Chapter 11

Design and implementation of a Case-Based Classifier Approach[+]

11.1. Introduction

In connection with the development of knowledge based systems, there is no universal Case-Based Reasoning (CBR) method suitable for every application domain. Most CBR systems make use of general domain knowledge and are able to utilize the specific knowledge of previously experienced concrete problem situations (cases). CBR is also an approach to incremental sustained learning, since a new experience is retained each time a problem has been solved, making it immediately available for future problems. CBR seems to be advantageous for domains such as agriculture, medical systems, business planing etc. which are inherently case-based and don't have a well defined theory but do have a great abundance of data in the form of cases. But, still now the problems associated with CBR are the complexity and the accuracy of the retrieval phase. Both flat memory and inductive approaches suffer from serious drawbacks. The major pitfall that has been identified in the use of CBR is the inflexibility of the knowledge and strategies acquired [1].

In recent years, the classification task has been enjoying increasing interest in the machine learning community. In this context, several approaches like Artificial Neural Networks, Probabilistic Neural Networks, k-Nearest Neighbor approach, Beysian classifier, RBF Networks etc. have been reported [2-12]. Classifier uses a multitude of techniques to decide whether an unknown instance described by a vector of feature values belongs to a certain class. The task of determining which of these features are relevant to the classification task is one of the central problems [13]. The computational complexity of training algorithm is also a vital problem [14]. Moreover, some works draw attention regarding misconceptions of some classifiers [15,16].

Case-Based Reasoning (CBR) means reasoning based on previous experiences [17]. CBR favours learning from experience, since it is usually easier to learn by retaining a concrete problem solving experience than to generalize from it. Moreover, a very important feature of CBR is its coupling to learning.

Many theories of human concept learning posit that concepts are represented by prototypes [18] or exemplars [19]. Prototype models are relatively inflexible, they discard a great deal of information that people use during concept learning [20,21,22]. On the other hand, exemplar model represents concepts by their individual exemplars.

[+] This is based on the paper ["Design and Implementation of a Context-sensitive Case-Based Classifier Approach"] of the author. (Communicated)

Exemplar representations are far more flexible than prototype representations since they retain sensitivity to all of the information [23].

The knowledge base of our system is composed of rules embodying domain specific knowledge with category-exemplar type of case representation where a set or sub-set of features plans for all possible goal(s) (cases) which is/are likely to be achieved [24]. Due to the dynamic nature of the problem domain, we argue that the concept description cannot be represented only with category exemplars, but also with a set of specific weights associated with features and as well as exemplars. The weights associated with features must be adaptive to the changing environment in order to represent the acceptability of the exemplars (cases).

It is an attempt to couple a CBR method with classifier approach for learning from experience and using it for problem solving. The classifier is designed as a single layer neural network. The neural networks usually require a large set of training data and computationally complex training algorithm. The problems associated with learning of such systems are long computational time, fixation of reasonable starting values, guidance during training, non-convergence for some arbitrary data sets, etc. Moreover, the training samples are never stored in the network.

The above discussed problems are bypassed in our system by truncating the learning part of such classifiers. The task of learning is done by a CBR method and the classifier accepts the prior trained data as inputs. So no further learning of the classifier is required. We have proposed a model for case-based learning [25] in Chapter 10. This model can be used for selection of features and their relevance related to a context description. The system becomes rich with definite known cases obtained from the real field and sorts out the significant features for all possible categories and fixes the primary weights (Relative Feature Weights) of all features through an evaluation process [25]. It also reduces the uncertainty and defines the saturation of primary training. The training data is not assigned arbitrarily, but obtained from the region specific real field cases of the problem domain. Thus the learning makes sense within the context of the problem domain. So this work suggests a case-based classifier approach.

The features are considered as input values and the Relative Feature Weights as the synapse values respectively. The bias values are also obtained from Relative Feature Weights. When a set/sub-set of observed features is received, it is then transformed to a binary feature vector with elements 1 or 0. The input values thus processed reach to the classifier, it fires only when the weighted sum of all the potential values exceeds a certain limiting value (bias) of that particular case. The successful (fired) cases, if retained in the knowledge base, updates the Relative Feature Weights (synapses) of the concerned features.

In the next section, we present the theoretical foundations of our approach. In Section 11.3, we present the system architecture. Section 11.4 contains case illustrations for validation of the model. We conclude in section 11.5.

11.2. Theoretical foundation

Feature Vector (F)

Feature Vector (F) is a set of elements transformed from the feature set of all possible features in a domain. If N is the total number features in the feature set of a domain then the F can be represented as:

$$[F] = \begin{bmatrix} F_1 \\ F_2 \\ . \\ . \\ F_N \end{bmatrix} \qquad \text{.....} \quad 11.1$$

where each of the elements $F_1, F_2, \ldots .F_N$ possess a value either 1 or 0 depending upon the set of observed features. If for a particular observation, the i-th feature is observed, then $F_i = 1$ otherwise it is 0.

As an example, if a domain consists of total 6 features and for a particular case, features f_1, f_3 and f_6 are observed, then the feature set of this 6 features are transformed to a Feature Vector as:

$$[F] = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Bias vector (Φ)

Bias is a critical barrier of the weighted sum above which a case is considered to be fired and the fixation of the bias is a vital task to improve the performance of the system. In our system, the bias for cases under a category has been fixed on the basis of Relative Feature Weights (RFW). We have classified the features of each category in 3 sub-sets; (i) weekly-relevant (wr), (ii) relevant (r) and (iii) strongly relevant (sr). The features having RFW greater than the arithmetic mean (M_a) are labeled as 'strongly-relevant' features. The features having RFW in the range between $M_a/2$ to M_a is the 'relevant' one and features those having RFW less than $M_a/2$ are the 'weekly-relevant'.

The rationale for estimation of bias value is based on the theme that at least one member of each group should have the contribution and the features with highest RFW within each sub-set should have the highest priority to be considered in fixing the bias. The bias for i-th category is

$$\Phi_i = [\text{Max}(RFW_{wr}) + \text{Max}(RFW_r) + \text{Max}(RFW_{sr})]_i \quad \text{..... 11.2.}$$

The Bias Vector is a set where the elements represent the biases (case firing threshold) of all the possible categories in the domain. If M is the total number of categories in a domain, then the Bias Vector (Φ) can be represented as:

$$[\Phi] = \begin{bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ . \\ . \\ \Phi_M \end{bmatrix} \quad \text{..... 11.3}$$

Synapse Matrix (ω)

Synapse Matrix (ω) is a matrix whose elements are the Relative Feature Weights at saturation of all the features constructing the possible categories in a domain. After training of the system, the significant and non-significant features are identified and the corresponding Relative Feature Weights of all the categories are known. These Relative Feature Weights corresponding to a category constitutes the row of this Synapse Matrix. An element of the row will be considered as 0 (zero) if the corresponding feature has no contribution (non-significant feature) to that particular category. So ω is a M x N matrix where M is the number of categories and N is the total number of features in a domain. The ω can represented as:

$$[\omega] = \begin{bmatrix} \omega_{11} & \omega_{12} & \omega_{13} & . & . & . & . & . & \omega_{1N} \\ \omega_{21} & \omega_{22} & \omega_{23} & . & . & . & . & . & \omega_{2N} \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ \omega_{M2} & \omega_{M3} & . & . & . & . & . & . & \omega_{MN} \end{bmatrix} \quad \text{..... 11.4}$$

Where ω_{11} is the Relative Feature Weight (synapse) of feature-1 of the category-1, ω_{12} is that of feature-2 of the category-1 and so on.

Activation Value (X) and the System Output (Y)

The general model of the classifier consists of a summing part followed by an output part. In this case, we have considered the classifier as a single layer neural network of McCulloch-Pitts (MP) model of neurons [26]. The number of neurons or nodes in the network is equal to the number of possible categories in the problem domain. The Activation Value is given by a weighted sum of input values and a bias term. The output signal is typically a nonlinear transfer function of the activation value. For present problem, if $F_1, F_2, F_3 \ldots F_N$ be the elements of the Feature Vector (F) and $\omega_{i1}, \omega_{i2}, \omega_{i3}, \ldots\ldots \omega_{iN}$ are the synapse values for i-the node, then the Activation Value (X_i) for i-th node can be represented as:

$$X_i = \sum_{j=1}^{N} \omega_{ij} \cdot F_j - \Phi_i \qquad \ldots\ldots 11.5$$

where N is the total number of features in the domain.

Then the output value of the i-th node can be defined by the output function

$$Y_i = f(x_i)$$

The transfer function, that we have used is a binary function, where

$$f(x_i) = 1, \quad \text{if } x > 0$$
$$= 0, \quad \text{if } x \leq 0$$

If $f(x_i) = 1$, the i-th category is considered to be fired; otherwise not considered.

For present problem, McCulloch-Pitts model for i-th neuron is shown in fig. 11.1.

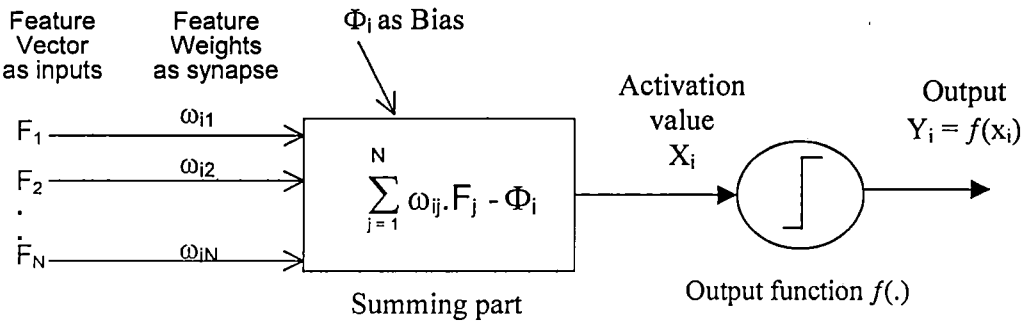

Fig.11.1. McCulloch-Pitts model of a neuron for present problem.

11.3. System architecture

In our system each case is represented with a set of features accepted as case descriptors and the collection of cases of the same type form a category. In the learning phase in Chapter 10, when the system gets trained with observed cases, the Relative Feature Weight of each feature approaches to saturation and when $| \omega_m - \omega_{(m+5)} | < \delta$, all significant features are obtained. The system is considered to be primarily trained and the primary case library is formed. At the end of primary training, the Relative Feature Weights of all the features of a category are known.

For a new case, the system receives the set of observed features and the Feature Vector Generator transforms observed feature set to Feature Vector (F). The system activates the Synapse Matrix Generator. The Synapse Matrix Generator then scans the case library to pick up the Relative Feature Weights of all the features for all categories and generates the Synapse Matrix (ω) according to equation 11.4. The Bias Vector Generator calculates the bias of each category and constructs the Bias Vector (Φ).

The classifier calculates the activation values (X) of the new case against each category and generates the outputs (Y) by using the transformation function. The categories for which the output is 1 are the classified category that is, the new case is of the type of this category. The block diagram of the system architecture is presented in fig. 11.2.

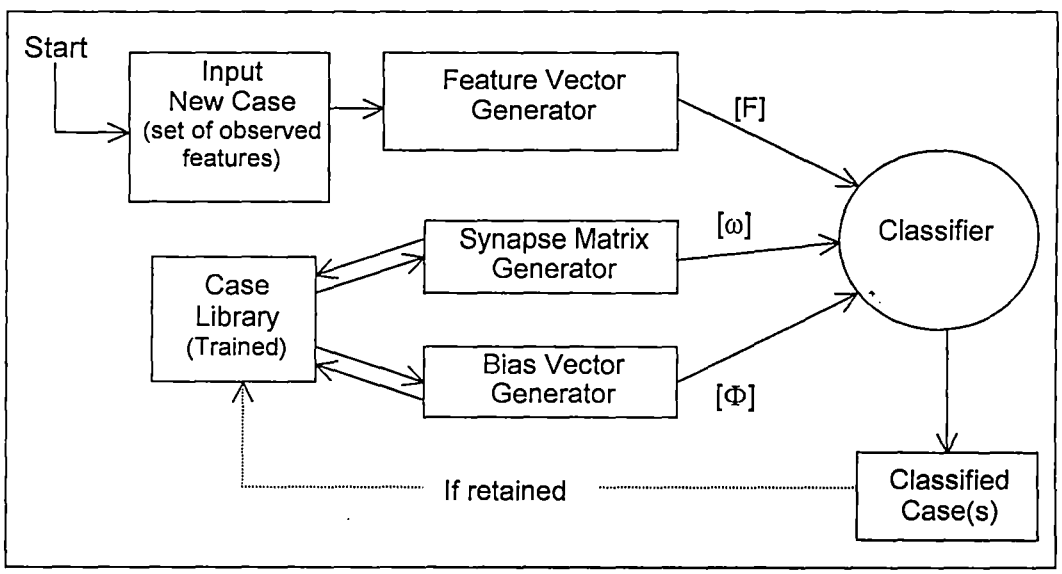

Fig. 11.2. Block diagram of the system architecture.

The classifier is a single layer neural network of McCulloch-Pitts model of neurons. If for a particular domain the number of possible categories is M, then the classifier

contains M neurons (nodes). Each neuron is responsible to classify a particular category. The block diagram of the classifier for M categories is presented in fig. 11.3.

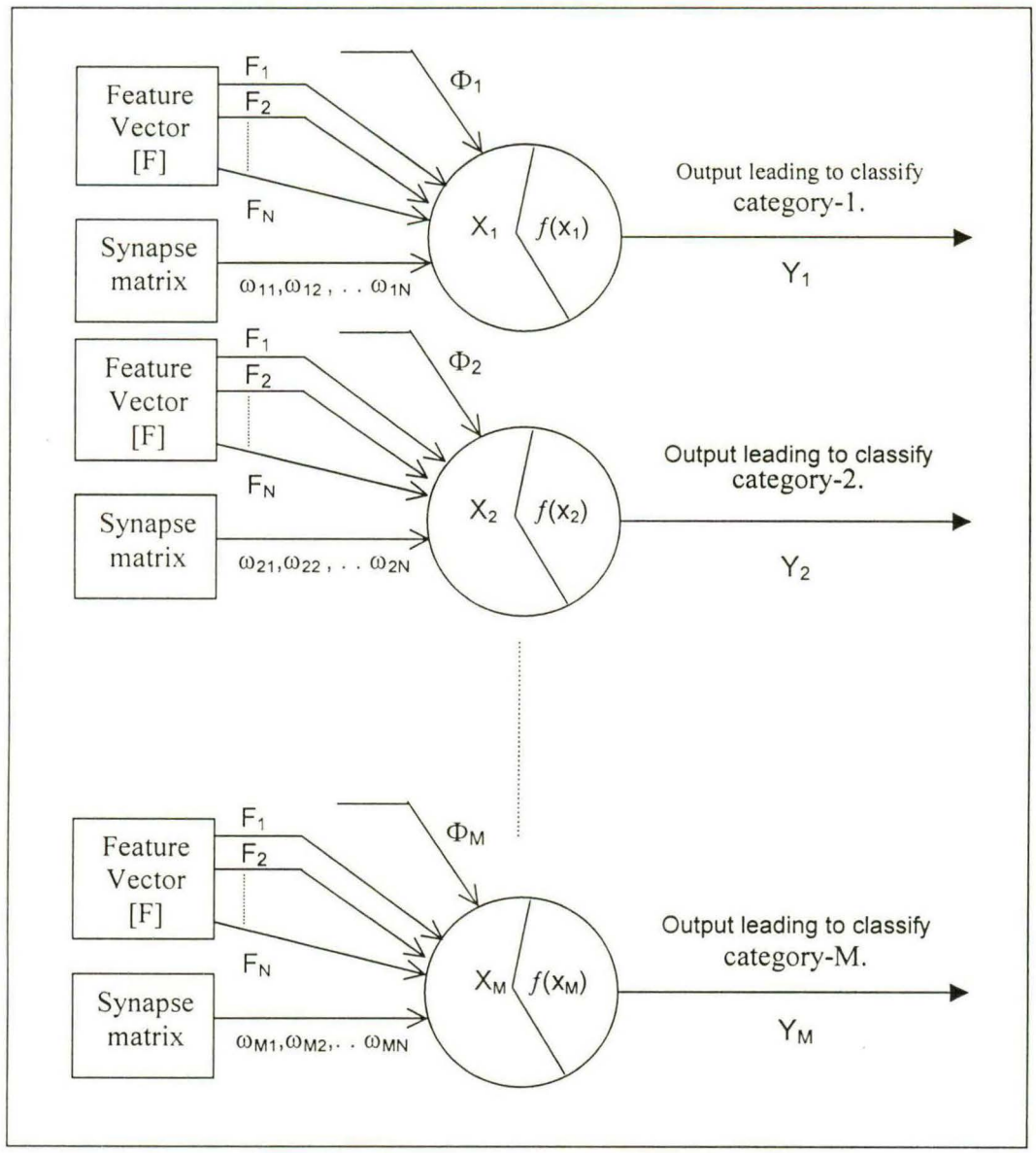

Fig. 11.3. The block diagram of the classifier for M categories.

Each neuron of this neural network accepts Feature Vector and Bias as inputs. The synapse values are accepted from the Synapse Matrix. Elements of each row of the Synapse Matrix provide the synapses to the corresponding neuron. The neuron then generates the Activation Value and gives output by using transfer function. Depending upon the output value, the case is considered to be classified.

Functionality of the proposed architecture is illustrated in the following steps:

Step 1. Input set of observed features for new case.
Step 2. Generate Feature Vector from the observed set of features.
Step 3. Generate Synapse Matrix by using case library.
Step 4. Generate Bias Vector by using case library.

Task of Classifier

Step 5. Calculate Activation Value of new case against each category.
Step 6. Calculate the Output against each category by using transfer function.
Step 7. If Output = 1 for any category, THEN the case is classified to that category.

Step 8. If the classified case is retained in the case library, THEN the Synapse Matrix and Bias Vector become updated, and dynamic learning starts.

As C programming language has proven to be a pleasant, expressive and versatile language for a wide variety of application, the system has been implemented through C and can run on both Windows and as well as Unix platform.

11.4. Case illustration

For a better understanding and validation of the method, real field problems in tea cultivation are being produced here. In tea gardens, tea bushes are often subject to the attack of various insect pests throughout the year. When a bush is attacked by a particular type of insect pest, some set of signs and symptoms should be observed on a specific part of the bush. These set of signs and symptoms (features) lead to proper identification of the insect pests active in the field in general. But tea industry is an wide spread one where the agro-climatic conditions differ from garden to garden even from section to section and thus the set of signs and symptoms become region specific. The complete set of features is very rare to be observed. Moreover, among all the features, some may have greater significance than the other less important one.

We have considered 8 insect pests of tea as examples. So the case library has been designed to contain 8 categories. Each category is allotted to contain the cases of attack due to a particular insect pest. The allotted category against each insect pest is presented in table 11.1.

Table 11.1. The allotted categories against 8 insect pests.

Insect pests	Allotted category
1. Red Spider	Category-1
2. Helopeltis	Category-2
3. Scarlet Mite	Category-3
4. Thrips	Category-4
5. Aphid	Category-5
6. Jussid	Category-6
7. Purple Mite	Category-7
8. Pink Mite	Category-8

31 possible sign and symptoms of attack leading to cover the identification of these 8 insect pests mentioned above, are identified by using domain knowledge and grouped under 11 rubrics as shown in table 11.2. This general feature set contains the all possible sub sets of features due to the attack of these 8 insect pests.

Table 11.2. General feature set of sign and symptoms for attack of 8 insect pests of tea crop.

Feature Groups	Serial number	Code and Description of features
1. Site of damage	1.	A1 - Young leaf
	2.	A2 - Matured leaf
	3.	A3 - Bud & young leaf
	4.	A4 - Tender stem
2. Leaf surface	5.	B1 - Upper
	6.	B2 - Lower
3. Leaf appearance	7.	C1 - Dry up
	8.	C2 - Leathery
	9.	C3 - Dusted white
	10.	C4 - Curved downwards
	11.	C5 - Curved upwards
	12.	C6 - Crinkled & curled
	13.	C7 - Deformed
	14.	C8 - Margin recurved & brown
4. Leaf colour	15.	D1 - Copper bronze
	16.	D2 - Yellow
	17.	D3 - Pale
	18.	D4 - Purple bronze
	19.	D5 - Brown
5. Leaf spot	20.	E1 - Brown ring
	21.	E2 - Reddish brown patch
	22.	E3 - Sand papery lines
6. Mid Rib colour	23.	F1 - Brownish
	24.	F2 - Upper side brownish
7. Edge colour	25.	G1- Brownish
	26.	G2 - Pinkish
8. Tip colour	27.	H1 - Brownish
9. Vein colour	28.	I1 - Brownish
10. Finger Tip test	29.	J1 - Red smear
11. Bush appearance	30.	K1- New flush stunted
	31.	K2 - Defoliation

Observed set of features leading to the definite known cases of attack of these 8 insect pests have been supplied to the system sequentially as first cum first server basis to train the system. After each entry, an iterative evaluation process as described in Chapter 10 generates the value of Relative Feature Weight of each feature corresponding to each category. After each entry, the system tests for saturation with $\delta \sim 10^{-3}$. A positive result of saturation testing indicates the end of primary training, that is, no more cases are required to train the system.

The features obtained with Relative Feature Weight # 0 are the significant features and that of with Relative Feature Weights = 0 are the non-significant or noisy features.

Relative Feature Weight = 0 indicates that a noisy feature has no contribution to a particular category. Moreover, it is observed that some features are common to different categories but with different values of Relative Feature Weights. The noisy features of each category have been ignored as they have no contribution.

The significant features and the Relative Feature Weights (which constitute the non-zero elements of Synapse Matrix) obtained against each category are presented in tables 11.3 – 11.10.

Table 11.3. The non-zero elements of Synapse Matrix for Category-1.

Feature Codes	Significant features	Non-zero elements of Synapse Matrix
A2	Site of damage IS Matured leaf	$\omega_{1,02} = 0.213$
D5	Leaf colour IS Brown	$\omega_{1,19} = 0.152$
E2	Leaf spots IS Reddish brown patch	$\omega_{1,21} = 0.374$
J1	Finger tip test IS Red smears	$\omega_{1,29} = 0.208$
K1	Bush appearance IS New flush stunted	$\omega_{1,30} = 0.053$

Table 11.4. The non-zero elements of Synapse Matrix for Category-2.

Feature Codes	Significant features	Non-zero elements of Synapse Matrix
A3	Site of damage IS Bud & young leaf	$\omega_{2,03} = 0.240$
A4	Site of damage IS Tender stem	$\omega_{2,04} = 0.057$
C7	Leaf appearance IS Deformed	$\omega_{2,13} = 0.260$
E1	Leaf spots IS Brown ring	$\omega_{2,20} = 0.335$
K1	Bush appearance IS New flush stunted	$\omega_{2,30} = 0.108$

Table 11.5. The non-zero elements of Synapse Matrix for Category-3.

Feature Codes	Significant features	Non-zero elements of Synapse Matrix
A1	Site of damage IS Young leaf	$\omega_{3,01} = 0.088$
A2	Site of damage IS Matured leaf	$\omega_{3,02} = 0.109$
B1	Leaf surface IS Upper	$\omega_{3,05} = 0.140$
C1	Leaf appearance IS Dry up	$\omega_{3,07} = 0.098$
D2	Leaf colour IS yellow	$\omega_{3,16} = 0.150$
F1	Mid Rib colour IS Brownish	$\omega_{3,23} = 0.040$
F2	Mid Rib colour IS Upper side brownish	$\omega_{3,24} = 0.266$
K2	Bush appearance IS Defoliation	$\omega_{3,31} = 0.108$

Table 11.6. The non-zero elements of Synapse Matrix for Category-4.

Feature Codes	Significant features	Non-zero elements of Synapse Matrix
A1	Site of damage IS Young leaf	$\omega_{4,01} = 0.270$
A3	Site of damage IS Bud & young leaf	$\omega_{4,03} = 0.092$
C5	Leaf surface IS Curved upwards	$\omega_{4,11} = 0.091$
C7	Leaf appearance IS Deformed	$\omega_{4,13} = 0.210$
E3	Leaf spots IS Sand papery lines	$\omega_{4,22} = 0.084$
H1	Tip colour IS Brownish	$\omega_{4,27} = 0.107$
K1	Bush appearance IS New flush stunted	$\omega_{4,30} = 0.145$

Table 11.7. The non-zero elements of Synapse Matrix for Category-5.

Feature Codes	Significant features	Non-zero elements of Synapse Matrix
A1	Site of damage IS Young leaf	$\omega_{5,01} = 0.104$
A3	Site of damage IS Bud & young leaf	$\omega_{5,03} = 0.124$
A4	Site of damage IS Tender stem	$\omega_{5,04} = 0.052$
B2	Leaf surface IS Lower	$\omega_{5,06} = 0.150$
C6	Leaf appearance IS Crinkled & curled	$\omega_{5,12} = 0.341$
D5	Leaf colour IS Brown	$\omega_{5,19} = 0.045$
K1	Bush appearance IS New flush stunted	$\omega_{5,30} = 0.184$

Table 11.8. The non-zero elements of Synapse Matrix for Category-6.

Feature Codes	Significant features	Non-zero elements of Synapse Matrix
A1	Site of damage IS Young leaf	$\omega_{6,01} = 0.193$
A4	Site of damage IS Tender stem	$\omega_{6,04} = 0.086$
B2	Leaf surface IS Lower	$\omega_{6,06} = 0.139$
C1	Leaf appearance IS Dry up	$\omega_{6,07} = 0.055$
C4	Leaf appearance IS Curved downwards	$\omega_{6,10} = 0.122$
C8	Leaf appearance IS Margin recurved & brown	$\omega_{6,14} = 0.058$
F1	Mid Rib colour IS Brownish	$\omega_{6,23} = 0.056$
I1	Vein colour IS Brownish	$\omega_{6,28} = 0.054$
K1	Bush appearance IS New flush stunted	$\omega_{6,30} = 0.236$

Table 11.9. The non-zero elements of Synapse Matrix for Category-7.

Feature Codes	Significant features	Non-zero elements of Synapse Matrix
A1	Site of damage IS Young leaf	$\omega_{7,01} = 0.015$
A2	Site of damage IS Matured leaf	$\omega_{7,02} = 0.206$
B1	Leaf surface IS Upper	$\omega_{7,05} = 0.183$
B2	Leaf surface IS Lower	$\omega_{7,06} = 0.035$
C1	Leaf appearance IS Dry up	$\omega_{7,07} = 0.128$
C2	Leaf appearance IS Leathery	$\omega_{7,08} = 0.013$
C3	Leaf appearance IS Dusted white	$\omega_{7,09} = 0.089$
D1	Leaf colour IS Copper bronze	$\omega_{7,15} = 0.153$
D4	Leaf colour IS Purple bronze	$\omega_{7,18} = 0.020$
G1	Edge colour IS Brownish	$\omega_{7,25} = 0.090$
G2	Edge colour IS Pinkish	$\omega_{7,26} = 0.066$

Table 11.10. The non-zero elements of Synapse Matrix for Category-8.

Feature Codes	Significant features	Non-zero elements of Synapse Matrix
A1	Site of damage IS Young leaf	$\omega_{8,01} = 0.217$
A4	Site of damage IS Tender stem	$\omega_{8,04} = 0.045$
B1	Leaf surface IS Upper	$\omega_{8,05} = 0.126$
C2	Leaf appearance IS Leathery	$\omega_{8,08} = 0.102$
C6	Leaf appearance IS Crinkled & curled	$\omega_{8,12} = 0.108$
D3	Leaf colour IS Pale	$\omega_{8,17} = 0.116$
G1	Edge colour IS Brownish	$\omega_{8,25} = 0.051$
I1	Vein colour IS Brownish	$\omega_{8,28} = 0.064$
K1	Bush appearance IS New flush stunted	$\omega_{8,30} = 0.171$

Now let us proceed to calculate the bias values (elements of Bias Vector) for each category. As a primary task to calculate the bias, the type of the features (i.e. strongly relevant, relevant or weekly relevant) are to be identified.

As an example, let us consider category-1.

The arithmetic mean of the Relative Feature Weights at saturation is $M_a = 0.200$
Depending upon the arithmetic mean, the type of the features of category-1 are labeled and shown in table 11.11.

Table 11.11. The type of features identified to calculate bias (Φ_1) of category-1.

Feature Codes	Significant features	RFWs (at saturation)	Type of features
A2	Site of damage IS Matured leaf	0.213	Strongly relevant
D5	Leaf colour IS Brown	0.152	Relevant
E2	Leaf spots IS Reddish brown patch	0.374	Strongly relevant
J1	Finger tip test IS Red smears	0.208	Strongly relevant
K1	Bush appearance IS New flush stunted	0.053	Weekly relevant

From the equation 11.2, the bias Φ_1 for the Category-1 is

$$\Phi_1 = 0.053 + 0.152 + 0.374$$
$$= 0.\ 579$$

Similarly, the bias values for other 7 categories are calculated. The bias values for 8 categories all together are presented in table 11.12.

Table 11.12. The calculated bias values of 8 categories.

Category	Insect pests	Bias values
Category-1	Red Spider	$\Phi_1 = 0.579$
Category-2	Helopeltis	$\Phi_2 = 0.500$
Category-3	Scarlet Mite	$\Phi_3 = 0.415$
Category-4	Thrips	$\Phi_4 = 0.377$
Category-5	Aphid	$\Phi_5 = 0.517$
Category-6	Jussid	$\Phi_6 = 0.376$
Category-7	Purple Mite	$\Phi_7 = 0.423$
Category-8	Pink Mite	$\Phi_8 = 0.376$

Now the system is ready to classify the new cases.

To test the performance of the system, more than 50 definite and real field cases of the problem domain were supplied to the system to classify the cases. The system classified all of them very accurately. Four examples are presented here.

Case example 1.

The observed set of features and feature codes for Case example 1 are:

1. Site of damage IS Young leaf (A1)
2. Leaf appearance IS Leathery (C2)
3. Leaf appearance IS Crinkled & curled (C6)
4. Leaf colour IS Pale (D3)
5. Vein colour IS Brownish (I1)

On the basis of the feature set, the classifier generates the activation values of the above case in respect of all categories and uses the transfer function to classify the case of specific category. The activation values and the classification of case by the system is shown in table 11.13. The actual observation in the field is also presented in the last column of the same table for comparison.

Table 11.13. System's output verses actual field observation for Case example 1.

Categories	$\sum \omega \cdot F$	Φ	X	Y	Classification by the system	Actual field observation
1. Red Spider	0.000	0.579	- 0.579	0		Pink Mite
2. Helopelties	0.000	0.500	- 0.500	0		
3. Scarlet Mite	0.088	0.415	- 0.327	0		
4. Thrips	0.270	0.377	- 0.107	0		
5. Aphid	0.445	0.517	- 0.072	0		
6. Jussid	0.247	0.376	- 0.129	0		
7. Purple Mite	0.028	0.423	- 0.395	0		
8. Pink Mite	0.607	0.376	+0.231	1	Pink Mite	

Case example 2.

The observed set of features and feature codes for Case example 2 are:

1. Site of damage IS Young leaf (A1)
2. Leaf surface IS Curved upwards (C5)
3. Leaf appearance IS Deformed (C7)
4. Tip colour IS Brownish (H1)

With the above observations, the output of the system is shown in table 11.14.

Table 11.14. System's output verses actual field observation for Case example 2.

Categories	$\sum \omega \cdot F$	Φ	X	Y	Classification by the system	Actual field observation
1. Red Spider	0.000	0.579	- 0.579	0		Thrips
2. Helopelties	0.260	0.500	- 0.240	0		
3. Scarlet Mite	0.088	0.415	- 0.327	0		
4. Thrips	0.678	0.377	+0.301	1	Thrips	
5. Aphid	0.104	0.517	- 0.413	0		
6. Jussid	0.193	0.376	- 0.183	0		
7. Purple Mite	0.015	0.423	- 0.408	0		
8. Pink Mite	0.217	0.376	- 0.159	0		

Case example 3.

The observed set of features and feature codes for Case example 3 are:

1. Site of damage IS Tender stem (A4)
2. Leaf appearance IS Deformed (C7)
3. Leaf spots IS Brown ring (E1)
4. Bush appearance IS New flush stunted (K1)

With the above observations, the output of the system is shown in table 11.15.

Table 11.15. System's output verses actual field observation for Case example 3.

Categories	$\sum \omega \cdot F$	Φ	X	Y	Classification by the system	Actual field observation
1. Red Spider	0.053	0.579	- 0.526	0		Helopeltis
2. Helopelties	0.760	0.500	+0.260	1	Helopeltis	
3. Scarlet Mite	0.000	0.415	- 0.415	0		
4. Thrips	0.355	0.377	- 0.022	0		
5. Aphid	0.185	0.517	- 0.332	0		
6. Jussid	0.322	0.376	- 0.054	0		
7. Purple Mite	0.000	0.423	- 0.423	0		
8. Pink Mite	0.216	0.376	- 0.160	0		

Case example 4

The observed set of features and feature codes for Case example 4 are

1. Site of damage IS Matured leaf (A2)
2. Leaf appearance IS Dry up (C1)
3. Leaf appearance IS Deformed (C7)
4. Leaf spots IS Reddish brown patch (E2)
5. Finger tip test IS Red smears (J1)
6. Bush appearance IS New flush stunted (K1)

With the above observations, the output of the system is shown in table 11.16.

Table 11.16. System's output verses actual field observation for Case example 4.

Categories	$\sum \omega_i F$	Φ	X	Y	Classification by the system	Actual field observation
1. Red Spider	0.848	0.579	+0.269	1	Red Spider	Red Spider
2. Helopelties	0.368	0.500	- 0.132	0		
3. Scarlet Mite	0.207	0.415	- 0.208	0		
4. Thrips	0.355	0.377	- 0.022	0		
5. Aphid	0.184	0.517	- 0.333	0		
6. Jussid	0.281	0.376	- 0.095	0		
7. Purple Mite	0.334	0.423	- 0.089	0		
8. Pink Mite	0.171	0.376	- 0.205	0		

11.5. Concluding remarks

From the above examples, it is evident that the proposed approach might be considered as a potential approach for classifier type of problems using CBR. In classification task, the consistency of training data is a vital problem which is managed easily in this system. Our system deals with data obtained from definite real field cases. As the system gets trained with real field cases, no heuristics or approximate estimation is concerned and thus reducing the uncertainty to a great extent. The only uncertainty concerned in our system is the uncertainty related to the real field observations for the new cases.

It is a hybrid system of CBR and Classifier concept. CBR eliminates the problems concerned with the training of a classifier and classifier removes the problems concerned with retrieval and adaptation of CBR. It is an attempt to design such a system where the merits of CBR and Classifier are clustered.

References

1. S. M. Williams, J. D. Bransford, N. J. Vye, S. R. Goldman and K. Carlson. Positive and Negative Effects of Specific Knowledge on Mathematical Problem-solving. Paper presented at Annual meeting, American Educational Research Association, Atlanta GA, 1993.

2. N. K. Bose and P. Liang. Neural Network Fundamentals with Graphs, Algorithms and Applications. McGraw-Hill, Int. Editions. 1996.

3. M. H. Hassoun. Fundamentals of Artificial Neural Networks. Cambridge, MA: MIT Press. 1995.

4. M. Sarkar and B. Yegnanarayana. An evolutionary programming-based probabilistic neural network construction technique. In Proc. of IEEE Int. Conference on Neural Networks. (Houston, USA), June 9-12, 1997.

5. R. M. Welch, S. K. Sengupta, A. K. Goroch, P. Rabindra, N. Rangaraj, and M. S. Navar. Polar cloud and surface classification using AVHRR imagery: An intercomparison of methods. Journal of Applied Meteorology, 31, 1992. pp. 405-420.

6. P. Domingos and M. Pazzani. Beyond independence: Conditions for the optimality of the simple Bayesian classifier. Proceedings of the 13th International Conference on Machine Learning, San Francisco, CA: Morgan Kaufmann. 1996. pp. 105-112.

7. B. V. Dasarathy. Nearest Neighbour(NN) Norms: NN Pattern Classification Techniques. Los Alamitos, California: IEEE Computer Society Press, 1991.

8. P. Langley and W. Iba. Average-case Analysis of a Nearest Neighbour Algorithm. Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI), San Mateo, CA: Morgan Kaufmann, 1993. pp. 889-894.

9. Machine Learning, Neural and Statistical Classification. Eds., D. Michie, D. J. Spiegelhalter, and C. C. Taylor, UK : Ellis Horwood Ltd., 1994.

10. J. Rachlin, S. Kasif, S. Salzberg and D. W. Aha. Towards a Better Understanding of Memory-Based Reasoning Systems. Proceedings of the 11th International Machine Learning Conference (ML94), San Francisco, CA: Morgan Kaufmann, 1994.

11. D. J. C. MacKay. Bayesian methods for supervised neural networks. The Handbook of Brain Theory and Neural Networks, (M. A. Arbib, ed.), Cambridge, MA: MIT Press, 1995. pp. 144-149.

12. D. Lowe. Radial basis function networks. The Handbook of Brain Theory and Neural Networks, (M. A. Arbib, ed.), Cambridge, MA: MIT Press, 1995. pp. 779-782.

13. T. R. Payne and P. Edwards. Implicit Feature Selection with the Value Difference Metric. 13th European Conference on Artificial Intelligence(ECAI-98), Ed. Henri Prade, John Wiley & Sons. Ltd, 1998.

14. M. Minsky and S. Papert. Perceptrons. The MIT Press, Cambridge, MA, 1988.

15. Asim Roy. "Summery of panel discussion at ICNN'97 on connectionist learning. Connectionist Learning: Is it Time to Reconsider the Foundations", JNNS Newsletter, appearing with Neural Networks, Vol. 11, No. 2, 1998.

16. Asim Roy. Artificial Neural Networks – A Science in Trouble. Vivek (A Quarterly in Artificial Intelligence, ISSN 0970-1618), Vol. 13, No. 2, 2000. pp. 17-24.

17. R. Schank. Dynamic Memory: A Theory of Learning in Computers and People. Cambridge, 1982.

18. S. K. Reed. Pattern recognition and categorization. Cognitive Psychology, 3, 1972. pp. 382-407.

19. D. L. Medin and M. M. Schaffer. Context theory of classification learning. Psychological Review, 85, 1978. pp. 207-238.

20. D. Homa and J. Cultice. Role of feedback, category size, and stimulus distortion on acquisition and utilization of ill-defined categories. Journal of Experimental psychology: Learning, Memory and Cognition, 8, 1984. pp. 37-50.

21. L. Fried and K. J. Holyoak. Induction of category distributions: A framework for classification learning. Journal of Experimental Psychology: Learning, Memory, and Cognition, 10, 1984. pp. 234-257.

22. B. W. A. Whittlesea. Preservation of specific experiences in the representation of general knowledge. Journal of Experimental Psychology: Learning, Memory, and Cognition, 1, 1987. pp. 3-17.

23. D. W. Aha and R. L. Goldstone. Concept Learning and Flexible Weighting. In Proceedings of the Cognitive Science Society, Bloomongton, IN: Lawrence Wrlbaum, 1992. pp. 534-539.

24. Agnar Aamodt and Enric Plaza. Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AICom – Artificial Intelligence Communications, IOS Press, Vol. 7, No. 1, 1994. pp. 39-59.

25. Indrajit Ghosh and R. K. Samanta. A Model for Case-Based Learning. The Journal of The Computer Society of India, Vol. 30, No. 4, 2000. pp. 9-14.

26. B. Yegnanarayana. Artificial Neural Networks. Prentice-Hall of India Pvt. Ltd., New Delhi, 1999. pp. 26-27.