Aggregation Pheromone Density Based Pattern Classification

Anindya Halder*

Center for Soft Computing Research, Indian Statistical Institute, Kolkata 700108, India

anindya_t@isical.ac.in

Ashish Ghosh

Machine Intelligence Unit and Center for Soft Computing Research

Indian Statistical Institute, Kolkata 700108, India

ash@isical.ac.in

Susmita Ghosh

Department of Computer Science Engineering and Jadavpur University

Kolkata 700032, India

Abstract. The study of ant colonies behavior and their self-organizing capabilities is of interest to machine learning community, because it provides models of distributed adaptive organization which are useful to solve difficult optimization and classification problems among others. Social insects like ants, bees deposit pheromone (a type of chemical) in order to communicate between the members of their community. Pheromone, that causes clumping behavior in a species and brings individuals into a closer proximity, is called aggregation pheromone. This article presents a new algorithm (called, APC) for pattern classification based on this property of aggregation pheromone found in natural behavior of real ants. Here each data pattern is considered as an ant, and the training patterns (ants) form several groups or colonies depending on the number of classes present in the data set. A new test pattern (ant) will move along the direction where average aggregation pheromone density (at the location of the new ant) formed due to each colony of ants is higher and hence eventually it will join that colony. Thus each individual test pattern (ant) will finally join a particular colony. The proposed algorithm is evaluated with a number of benchmark data sets as well as various kinds of artificially generated data sets using three evaluation measures. Results are compared with four other well known conventional classification techniques. Experimental results show the potentiality of the proposed algorithm in terms of all the evaluation measures compared to other algorithms.

Keywords: Swarm intelligence, Ant colony optimization, Aggregation pheromone, Pattern classification.

*Address for correspondence: Center for Soft Computing Research Indian Statistical Institute, Kolkata 700108, India

1. Introduction

The task of classification occurs in various ranges of human activities. A classification problem includes the assignment of an object to a predefined class according to its characteristics [7]. At its broadest, the term could cover any context in which some decision or forecast is made on the basis of currently available information, and a classification procedure is a formal method for repeatedly making such judgments in new situations.

A wide variety of techniques exists in the literature [7, 12, 14, 18] since several decades for pattern classification. Broadly, three classes of classifiers exist [14] in the literature. They consist of classifiers that depend on (i) similarity maximization methods, (ii) probabilistic methods, and (iii) geometric methods. The first class of classifiers have some similarity metrics and assign class labels for maximizing the similarity. Minimum distance classifier, K-NN classifier [3] are the most common examples of that type. Probabilistic methods, for which the Bayesian classifier is the most known, depend on the prior probabilities of classes and class-conditional densities of the instances. In addition to Bayesian classifiers, logistic classifiers which is based on maximum likelihood approach, belong to this type of classifiers. The third category of classifiers is the geometric classifiers, which build decision boundaries by directly minimizing the error criterion. Neural networks [29], Fishers linear discriminant methods are examples of geometric classifiers.

Apart from these above mentioned three categories of classifiers there are various kinds of classifiers such as decision trees [30], support vector machine [32], fuzzy set based classifiers [19] etc.

In this article, an aggregation pheromone density based classifier is proposed which is inspired by the natural behavior found in real ants and other social insects. The social insects' behavior such as finding the best food source, building of optimal nest structure, brooding, protecting the larva, guarding etc. show intelligent behavior on the swarm level [8, 15]. A swarms' behavior is not determined just by the behavior of itself, but the interactions among individuals play a vital role in shaping the swarm behavior [8, 15]. Computational modeling of swarms' behavior is found to be useful in various application domains like, function optimization [35, 36], finding optimal routes [4], scheduling [6], image and data analysis [37]. Different applications originated from the study of different types of swarms. Among them, most popular ones are ant colonies and bird flocks [8]. Ant Colony Optimization (ACO) [5, 6] and Aggregation Pheromone Systems (APS) [35, 36] are computational algorithms modeled on the behavior of ant colonies. ACO [5, 6] algorithms are designed to emulate ants' behavior of laying pheromone on the ground while moving to solve optimization problems. Pheromone is a type of chemical emitted by an organism to communicate between members of the same species. Pheromone, which is responsible for clumping or clustering behavior in a species and brings individuals into closer proximity, is known as aggregation pheromone. Thus, aggregation pheromone causes individuals to aggregate around positions which in turn produce more pheromone to attract individuals of the same species. In APS [35, 36], a variant of ACO, this behavior of ants is used to solve real parameter optimization problems. A model for solving continuous optimization problems [33] was also proposed as an extension of ant colony optimization (ACO) problem.

In the present article an aggregation pheromone density based algorithm, APC is proposed for pattern classification. In order to show the effectiveness of the proposed algorithm we have considered a number of real life benchmark data sets and various kinds of synthetic data sets. Results are compared with other standard popular classification algorithms. Experimental results justify the potentiality of the proposed APC method in terms of the classification accuracy for most of the data sets.

Rest of the paper is organized in four sections. In Section 2 motivation for the work and other related research is described. Proposed method is demonstrated in Section 3. In Section 4, we illustrate the experimental outcome, describe the data sets used, other classification techniques compared with, theoretical details of the performance evaluation measures and analysis of the experimental results. Finally, in Section 5 conclusions are drawn.

2. Motivation and Related work

Several species of ants group their corpses into "cemeteries" in an effort to clean up their nests. Experimental work illustrates that ants group corpses, which are initially randomly distributed in space, into clusters, within a few hours. It seems that some feedback mechanism (using local density or similarity of data items) determines the probability that an ant will pick up or drop a corpse. This concept was generalized by numerous researchers and proposed various algorithm for unsupervised classification or clustering. A comprehensive survey can be found in [13].

Inspired by the ants' property of piling up the corpses to clean the nest various ant based clustering algorithms are proposed. Besides nest cleaning, many functions of aggregation behavior have been observed in ants and ant like agents [1, 24, 34]. These include foraging-site marking and mating, finding shelter and defense. For example, after finding safe shelter, cockroaches produce a specific pheromone with their excrement, which attracts other members of their species [34]. Based on the similar property i.e., ants need to find comfortable and secure environment to sleep, Chen et al. [2] proposed *Ant Sleeping Model* which makes ants to group with those that have similar physiques. They defined a fitness function to measure the ants' similarity with their neighbors. They stated that when an ant's fitness is low, it has a higher probability to wake up and stay in active state. Thus an ant will leave its original position to search for a more secure and comfortable position to sleep. Since each individual ant uses only a little local information to decide whether to be in active state or sleeping state, the whole ant group dynamically self organizes into distinctive, independent subgroups. Using similar concept Tsutsui et al. [35, 36] used *Aggregation Pheromone Systems* for continuous function optimization where aggregation pheromone density is defined by a density function in the search space.

As mentioned above, many functions of aggregation behavior have been observed in ants and ant like agents. Inspired by the aggregation pheromone system found in ants and other similar agents, in earlier works, attempts are made for solving clustering [9, 16], image segmentation [11, 10], and change detraction [17] problems with encouraging results. As mentioned earlier, here in this article, our work on pattern classification is based on aggregation pheromone.

Though a large number of techniques exist for ant based unsupervised classification (i.e clustering) in the literature, only few attempts have been made for (supervised) classification. *AntMiner* is the first of this kind, proposed by Parpinelli et al. [27] to extract *if-then* classification rule from categorical data. Each rule in *AntMiner* contains a condition part as the antecedent and a predicted class. The condition part is a conjunction of attribute-operator-value tuples. A rule condition is added to the current partial rule that the ant is constructing with some defined probability value.

In *AntMiner*, the heuristic value is taken to be an information theoretic measure for the quality of the term to be added to the rule. Here in this case the quality is measured in terms of entropy for preferring this term to others. As soon as the rule construction part is over, rule pruning is undertaken to increase the comprehensibility and accuracy of the rule. Rule pruning procedure iteratively removes the term whose

removal will cause a maximum increase in the (defined) quality of the rule. After the pruning step, the rule may be assigned a different predicted class based on the majority class in the cases covered by the rule antecedent. After each ant completes the construction of its rule, pheromone updating is carried out. Liu et al. further extended the algorithm to reduce the computational complexity in *AntMiner2* [20] and to increase the classification accuracy in *AntMiner3* [21]. Martens et al. in *AntMiner+* [22] modified the existing versions of *AntMiner* by (i) implementing better performing *MAX-MIN* Ant System, (ii) using directed acyclic graph for the environment to choose more effective path by the ants with the inclusion of the class variable to handle multi-class problems and (iii) applying early stopping criteria to prevent the rule base from the effect of noisey training data. Moreover, system parameters are set in an automated, dynamic manner in this version.

3. Proposed Methodology

As mentioned in the previous sections, aggregation pheromone brings individuals into closer proximity. This group formation nature of aggregation pheromone (found in natural behavior of real ants) is being used as the basic idea of the proposed technique. Here each data pattern is considered as an ant, and the training patterns (ants) form several colonies or homogeneous groups depending on the number of classes present in the data set. Each ant (in the group) emits pheromone around its local neighborhood. The intensity of pheromone (emitted by an individual ant) is maximum at the position where the ant is situated and it decays uniformly with the distance from the position of the ant. Hence pheromone intensity is modeled by the Gaussian function keeping the ant at the center. When a new ant (test pattern) comes in the system it tries to join to one of the existing colonies/groups. A new ant will move towards a colony for which average aggregation pheromone density (at the location of that new ant) is higher than that of the other colonies; and hence eventually the new ant will join that colony. Here average aggregation pheromone density of a colony is the average of the cumulative effect of pheromone intensity (at the location of the test ant) emitted by each individual ant belonging to that colony. Thus each individual new ant will join a particular colony.

3.1. Aggregation pheromone density based classification

Consider a data set with m classes, which (by our assumption) forms m homogeneous groups/colonies of ants or training patterns. Let $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \ldots, \boldsymbol{x}_{|C_i|}$ be the training data patterns in the class C_i and considered as a population of $|C_i|$-ants $a_1, a_2, a_3, ..., a_{|C_i|}$ which forms a group/colony C_i; where an ant $a_j \in C_i$ represents the training data pattern $\boldsymbol{x}_j$. The intensity of pheromone emitted by an individual ant a_j (located at $\boldsymbol{x}_j$) decreases with its distance from $\boldsymbol{x}_j$. Thus the pheromone intensity at a point closer to $\boldsymbol{x}_j$ is more than those at other points that are farther from it. To achieve this, the pheromone intensity emitted by ant $a_j \in C_i$ is modeled by a Gaussian distribution. The pheromone intensity deposited at $\boldsymbol{x}$ by an ant a_j (located at $\boldsymbol{x}_j$) is thus computed as

$$\Delta\tau(a_j, \boldsymbol{x}) = \exp^{-\frac{d(\boldsymbol{x}_j, \boldsymbol{x})^2}{2\delta^2}} \tag{1}$$

where, δ denotes the spread of Gaussian function and $d(\boldsymbol{x}_j, \boldsymbol{x})$ is the Euclidian distance between $\boldsymbol{x}_j$ and $\boldsymbol{x}$.

Total aggregation pheromone density at $\boldsymbol{x}$ deposited by the entire population of $|C_i|$ ants belonging to the colony C_i is computed as.

$$\Delta\tau_i(\boldsymbol{x}) = \sum_{x_j \in C_i} \exp^{-\frac{d(\boldsymbol{x}_j, \boldsymbol{x})^2}{2\delta^2}}. \tag{2}$$

Now a new (test pattern) ant a_t at $\boldsymbol{x_t}$ appears in the system. The average aggregation pheromone density (at the location of that new ant a_t) deposited by the colony C_i is given by

$$\Delta\overline{\tau}_i(\boldsymbol{x_t}) = \frac{1}{|C_i|} \sum_{x_j \in C_i} \exp^{-\frac{d(\boldsymbol{x}_j, \boldsymbol{x_t})^2}{2\delta^2}}. \tag{3}$$

The new ant a_t will move towards a colony for which average aggregation pheromone density (at the location of that new ant) is higher than that of other colonies. Hence finally that ant will join the colony determined by the following equation.

$$ColonyLabel(\boldsymbol{x_t}) = \arg\max_i(\Delta\overline{\tau}_i(\boldsymbol{x_t})). \tag{4}$$

Thus each of the test ants will join a colony and that colony label will be the class label of that test pattern (ant).

The proposed aggregation pheromone density based classification algorithm is given below.

Algorithm 1 : Aggregation pheromone density based classifier

for each new (test) ant a_t located at $\boldsymbol{x_t}$ **do**
  for each colony C_i **do**
    Calculate the average aggregation pheromone density at location $\boldsymbol{x_t}$ due to (all ant in) colony C_i using equation 3.
  end for
  Compute the $ColonyLabel(\boldsymbol{x_t})$ of the ant a_t by equation 4. // Ties are broken arbitrarily.
end for

4. Experimental Evaluation

For the purpose of our study, we used ten real life benchmark data sets from the UCI repository [23] and [25] (for Indian Telugu Vowel data) as well as 5 synthetic data sets. Experiments were carried in two different ways in order to test the classification accuracy. In the approch, from a data set, a certain percentage of data is taken out randomly to make the training set and the rest is considered as the test set. This process is repeated 20 times. In the second approach, k (=10) fold cross validation is used. The whole data set is randomly divided into k mutually exclusive and (nearly) equal sized subsets. For each subset, considered as the test set, the classifier is trained on the union of all other subsets. Then, cross validation is run (10 times) for each training and test set pair.

4.1. Description of the data sets

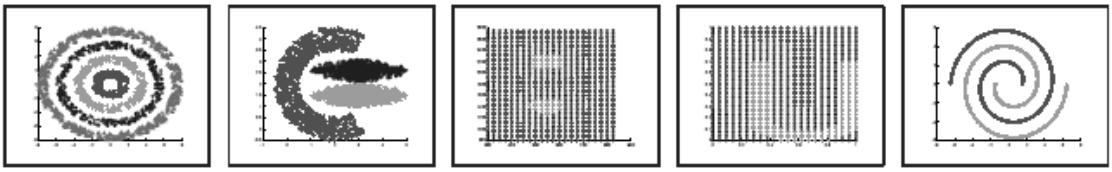
Out of the 10 real life data sets 9 data sets are taken from UCI repository [23] and one from [25] (for Indian Telugu Vowel data). Also we have used 5 different kinds of synthetic data. A summary of the data sets is given in Table 1.

Table 1. Summary of the data sets used. N is the total number of data, D and C represent dimensionality and number of classes, respectively.

Real Life data			
Data Set	**N**	**D**	**C**
Iris	150	4	3
WBC	683	9	2
Sonar	208	60	2
Thyroid	215	5	3
Glass	214	9	6
Balance scale	625	4	3
Telugu vowel	871	3	6
English vowel	990	10	11
Diabetes	768	8	2
Ionosphere	351	34	2
Synthetic data			
Data Set	**N**	**D**	**C**
Annular	1400	2	4
Ellipse	3000	2	3
Pat1	880	2	3
Pat2	880	2	3
Spiral	1000	2	2

Real life data sets:

To start with, as a toy problem, we have chosen perhaps the most common Iris data with 150 instances, 4 features (sepal length, sepal width, petal length, petal width) and distributed into 3 types of Iris plant. Wisconsin Breast Cancer (WBC) data contains 699 instances distributed in two categories described by nine features of which 16 instances with the missing values are ignored. Sonar data has 208 instances described by sixty attributes distributed in two classes. Thyroid data set has 215 instances of patients with five features describing whether patient has euthyroidism, hypothyroidism and hyperthyroidism (three classes). Glass data set has 214 instances describing six categories of glass on the basis of nine features. Balance scale data was generated to model the psychological experimental results. It has 625 instances described by four features, distributed in three classes. The Indian Telugu vowel data [25] is the formant frequency of sounds in consonant-vowel-consonant context uttered by three speakers in the age group 30-35 years. The data set consist of 871 instances with 3 formant frequencies (features) which were obtained through the spectrum analysis of the speech data. The data patterns are distributed



(a) (b) (c) (d) (e)

Figure 1. (a) Annular data, (b) Ellipse data (c) Pat1 data (d) Pat2 data (e) Spiral data

in six overlapping classes and their boundaries are ill-defined. English vowel (deterding data) having 990 instances with 10 features is a data set for speaker independent recognition of the eleven steady state vowels of British English. Diabetes data set has 768 instances, 8 attributes and is distributed in two classes "tested positive" and "tested negative". Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females of at least 21 years old of Pima Indian heritage. Ionosphere is a radar data which consist of 351 instances each with 34 continuous features distributed in two classes namely "good" and "bad". This radar data was collected by a system in Goose Bay, Labrador. This system consists of a phased array of 16 high-frequency antennas with a total transmitted power of the order of 6.4 kilowatts. The targets were free electrons in the ionosphere. "Good" radar returns are those showing evidence of some type of structures in the ionosphere. "Bad" returns are those that do not; their signals pass through the ionosphere.

Synthetic data sets

The classifiers are tested with 5 different kinds of artificially generated data sets also. First we have generated and experimented with annular data having 1400 instances and 2 features. The data is distributed in 4 concentric rings of different radius. The annular data is shown if Fig. 1 (a). Next ellipse data (shown in Fig. 1 (b)) is considered which has 3000 data points distributed in 3 classes with 2 features. Pat1 data [26] (shown in Fig. 1 (c)) having 880 instances in 3 classes and 2 features is used next. Part 2 data [26] (shown in Fig. 1 (d)) has 880 instances, 2 features and distributed in 3 classes. Spiral data (shown in Fig. 1 (e)) contains 1000 data points distributed in two spirals with 2 features. It is worth mentioning that all the synthetic data sets considered here are non linearly separable. Note that all the synthetic data sets in Fig. 1 are shown with different color and symbol for (data points in) different classes.

4.2. Comparison with other methods

The proposed method is compared with 4 other standard classification techniques, namely, (i) K nearest neighbor [3] (with k=5 and k=7 are taken as representative), (ii) Minimum distance classifier, (iii) Multi layer perception (MLP) [29], and (iv) Support vector machines (SVM) with sequential minimal optimization (SMO) [28, 32]. We have used Weka package [38] for MLP and SVM (SMO).

4.3. Performance evaluation measures

In order to evaluate the performance of the proposed classifier, in this article we have used following three kinds of performance measures.

Percentage accuracy

Here we have reported the results on test case accuracy only; that is percentage of correctly classified test patterns out of total test patterns.

Macro averaged F_1 measure

Macro averaged F_1 is derived from precision and recall [31]. The precision of a class i is defined as

$$precision_i(p_i) = \frac{\#\ patterns\ correctly\ classified\ into\ class\ i}{\#\ patterns\ classified\ into\ class\ i}, \tag{5}$$

and recall of class i is defined as

$$recall_i(r_i) = \frac{\#\ patterns\ correctly\ classified\ into\ class\ i}{\#\ patterns\ that\ are\ truly\ present\ in\ class\ i}. \tag{6}$$

Then F_1, the harmonic mean between precision and recall, of class i is defined as

$$(F_1)_i = \frac{2 \times p_i \times r_i}{p_i + r_i}. \tag{7}$$

F_1 measure gives equal importance to both precision and recall.

The macro-averaged F_1 measure is computed by first computing the F_1 scores for each category (class) and then averaging these per-category scores to compute the global means. $Macro{-}averaged\ F_1$ (or $Macro\ F_1$ in short) is defined as

$$Macro\ averaged\ F_1 = \frac{1}{m}\sum_{i=1}^{m}(F_1)_i, \tag{8}$$

where m is the number of category (class). Macro-averaged F_1 gives equal weight to each category. The value of Macro-averaged F_1 lies between 0 and 1. *More close the value of macro-averaged F_1 to 1, the better is the classification.*

Micro-averaged F_1 measure

The micro-averaged F_1 measures are computed by first creating a global contingency table whose cell values are the sum of the corresponding cells in the per-category contingency tables. Then use this global contingency table to compute the micro-averaged performance scores. Micro-averaged F_1 gives equal weightage on each sample (test case). $Micro - averaged\ F_1$ (or $Micro\ F_1$ in short) is defined as

$$Micro\ averaged\ F_1 = \frac{2 \times \frac{1}{m}\Sigma_{i=1}^{m} p_i \times \frac{1}{m}\Sigma_{i=1}^{m} r_i}{\frac{1}{m}\Sigma_{i=1}^{m} p_i + \frac{1}{m}\Sigma_{i=1}^{m} r_i}, \tag{9}$$

The value of Micro-averaged F_1 lies between 0 and 1. *More close the value of micro-averaged F_1 to 1, the better is the classification.*

4.4. Experimental results

The proposed aggregation pheromone density based classifier has only one parameter δ (spread of the Gaussian). For the optimal performance of the proposed classifier, we have experimented with wide range of δ for each data set. The δ value for which best result, in terms of performance evaluation measures, occurs is reported in Table 2 and Table 3. That selected δ value is put within bracket with APC

Table 2. Experimental evaluation on real life data for APC, K-NN, MDC, MLP and SVM classifiers using 10 percent of total data as training set and rest of the data as test set

Data	Method		% Accuracy	Macro F_1	Micro F_1	Time (in ms)
Iris	APC (δ=0.4)		93.666672 #1 (2.655067)	0.936044 #1 (0.027495)	0.939927 #1 (0.023075)	3.95 #3 (6.84)
	K-NN	K=5	90.333336 #3 (3.493870)	0.901006 #3 (0.039162)	0.910622 #3 (0.028172)	1.55 #2 (5.48)
		K=7	89.222214 (2.203316)	0.890289 (0.023724)	0.897831 (0.017741)	2.30 (4.65)
	MDC		91.814819 #2 (1.994162)	0.917606 #2 (0.020370)	0.921039 #2 (0.018701)	0.00 #1 (0.00)
	MLP		82.66668 (11.74583)	0.82136 (0.121781)	0.8478 (0.095351)	386 (33.2265)
	SVM		71.25924 (7.157253)	0.64342 (0.109071)	0.70794 (0.120788)	404 (38.78144)
WBC	APC (δ=3.2)		96.5397 #1 (0.509428)	0.96164 #1 (0.005809)	0.96166 #1 (0.005622)	24.95 #3 (7.83)
	K-NN	K=5	96.128250 (0.521697)	0.957060 (0.005987)	0.957364 (0.005761)	18.75 #2 (6.37)
		K=7	95.852280 (0.382357)	0.953874 (0.004439)	0.954325 (0.004208)	18.87 (6.27)
	MDC		96.217545 #3 (0.531270)	0.958094 #3 (0.006068)	0.958324 #3 (0.005861)	3.15 #1 (6.30)
	MLP		93.77778 (1.28327)	0.92988 (0.015179)	0.93066 (0.01481)	2674 (0.561514)
	SVM		96.355515 #2 (0.726681)	0.959629 #2 (0.007993)	0.959856 #2 (0.00798)	176 (93.0806)
Sonar	APC (δ=0.43)		65.00 #3 (2.655067)	0.642357 #3 (0.027495)	0.650773 #3 (0.023075)	3.90 #3 (6.84)
	K-NN	K=5	58.111702 (3.965799)	0.566007 (0.044301)	0.581504 (0.038698)	3.85 #2 (6.67)
		K=7	57.686176 (4.948453)	0.559964 (0.056456)	0.577353 (0.052228)	3.87 (6.67)
	MDC		59.893616 (4.055829)	0.592969 (0.042072)	0.601412 (0.040189)	2.25 #1 (5.36)
	MLP		66.8439 #2 (7.62616)	0.6667 #2 (0.07573)	0.67493 #2 (0.07969)	15050 (164.3168)
	SVM		68.40422 #1 (3.237267)	0.6821 #1 (0.03187)	0.69526 #1 (0.030342)	210 (125.698)
Thyroid	APC (δ=1.6)		85.438148 #1 (3.762797)	0.769650 #1 (0.065778)	0.791321 #1 (0.059440)	5.45 #3 (7.43)
	K-NN	K=5	76.056702 (3.893289)	0.498423 (0.121202)	0.582944 (0.108642)	1.55 #1 (6.84)
		K=7	69.587639 (0.000008)	0.273556 (0.000000)	0.273556 (0.000000)	3.95 (4.65)
	MDC		83.505150 (4.309599)	0.750875 #2 (0.072063)	0.770221 #3 (0.064120)	2.30 #2 (5.48)
	MLP		85.30928 #2 (3.480186)	0.704933 (0.109706)	0.715933 (0.117456)	685 (44.3113)
	SVM		84.43296 #3 (4.69497)	0.72368 #3 (0.095615)	0.7756 #2 (0.072271)	434 (58.51495)
Glass	APC (δ=0.13)		53.116722 #1 (2.729592)	0.400773 #1 (0.054025)	0.420660 #1 (0.064672)	9.40 #3 (7.68)
	K-NN	K=5	43.086739 (4.806423)	0.190268 (0.048741)	0.205177 (16.525463)	1.60 #1 (6.30)
		K=7	38.750004 (4.205626)	0.145445 (0.024202)	0.153085 (0.065397)	3.15 (4.80)
	MDC		13.775511 (0.000001)	0.040359 (0.000000)	0.040359 (0.000001)	4.65 #2 (7.11)
	MLP		51.9862 #2 (7.1085)	0.35362 #2 (0.064148)	0.370433 #2 (0.060149)	1851.667 (49.1313)
	SVM		49.84456 #3 (3.380186)	0.26172 #3 (0.03415)	0.30714 #3 (0.05251)	1988 (156.256)

Table2 Continued..						
Data	**Method**		**% Accuracy**	**Macro F_1**	**Micro F_1**	**Time (in ms)**
Balance-scale	APC (δ=0.30)		76.336281 (2.40545)	0.594873 (0.019518)	0.596314 (0.018957)	25.80 #3 (7.41)
	K-NN	K=5	80.336281 (1.959652)	0.571284 (0.022120)	0.586133 (0.031176)	14.00 #2 (4.69)
		K=7	80.911499 #3 (1.930970)	0.565143 (0.017209)	0.576407 (0.036716)	14.85 (5.99)
	MDC		69.796455 (3.117897)	0.598410 #3 (0.027168)	0.631183 #2 (0.033910)	3.85 #1 (6.67)
	MLP		88.21788 #1 (2.128149)	0.7553 #1 (0.05264)	0.764483 #1 (0.05467)	1535 (72.7438)
	SVM		86.99824 #2 (1.091485)	0.603 #2 (0.007208)	0.60392 #3 (0.00686)	464 (52)
Telugu Vowel	APC (δ=11.8)		77.91242 #1 (1.114165)	0.754243 #1 (0.030467)	0.757838 #1 (0.030069)	74.80 #3 (6.40)
	K-NN	K=5	76.176834 #3 (2.253696)	0.729180 #2 (0.022419)	0.734841 #2 (0.021028)	23.40 #2 (7.81)
		K=7	74.293877 (2.015436)	0.700342 (0.024606)	0.709387 (0.022805)	29.70 (6.72)
	MDC		69.465645 (1.827815)	0.675203 (0.020818)	0.682223 (0.020702)	8.65 #1 (7.83)
	MLP		77.277351 #2 (1.172304)	0.70745 #3 (0.01204)	0.729083 #3 (0.017214)	3445 (65.2559)
	SVM		68.75 (7.44663)	0.56984 (0.089007)	0.61218 (0.068327)	2176 (89.57678)
English Vowel	APC (δ= 0.50)		65.841751 #1 (2.939633)	0.660388 #1 (0.029055)	0.668256 #1 (0.029404)	100.70 #3 (12.48)
	K-NN	K=5	47.957355 #3 (2.462776)	0.471048 #3 (0.023784)	0.493678 #2 (0.022325)	39.30 #2 (7.71)
		K=7	45.342308 (2.096394)	0.442498 (0.023811)	0.468215 (0.023263)	41.40 (7.43)
	MDC		48.383842 #2 (2.278892)	0.482557 #2 (0.021116)	0.489458 #3 (0.020410)	15.75 #1 (0.43)
	MLP		46.03816 (2.257424)	0.45588 (0.01586)	0.47706 (0.018125)	32462 (327.6828)
	SVM		25.97084 (1.218338)	0.24794 (0.00865)	0.28068 (0.01679)	8256 (459.9826)
Diabetes	APC (δ=16.8)		69.703751 #3 (3.663801)	0.654223 #3 (0.031438)	0.657159 #3 (0.029830)	32.65 #3 (4.79)
	K-NN	K=5	68.208084 (2.668094)	0.622328 (0.034729)	0.641612 (0.023075)	24.30 #2 (6.84)
		K=7	69.205208 (2.406525)	0.619605 (0.037236)	0.645976 (0.031250)	24.40 (9.23)
	MDC		66.950859 (3.809962)	0.630764 (0.047395)	0.632433 (0.046946)	7.05 #1 (7.80)
	MLP		70.1589 #2 (3.82767)	0.67125 #1 (0.04237)	0.67265 #2 (0.042589)	671.25 (42.37628)
	SVM		72.45666 #1 (2.79107)	0.66994 #2 (0.040233)	0.67972 #1 (0.03612)	198 (129.213)
Ionosphere	APC (δ=0.80)		80.615135 #3 (5.758697)	0.750431 #3 (0.106371)	0.792146 #3 (0.057387)	10.15 #3 (7.46)
	K-NN	K=5	73.769714 (6.954584)	0.617905 (0.144472)	0.721410 (0.070467)	4.65 #2 (7.11)
		K=7	70.914833 (6.522116)	0.560510 (0.141594)	0.680493 (0.092459)	7.00 (7.75)
	MDC		76.876976 (5.344280)	0.746983 (0.051898)	0.751699 (0.054081)	0.75 #1 (3.27)
	MLP		82.0886 #2 (1.956914)	0.7815 #2 (0.021089)	0.8057 #2 (0.02448)	9500 (41.95235)
	SVM		83.03798 #1 (3.316587)	0.78732 #1 (0.051897)	0.82098 #1 (0.031353)	232 (74.13501)

Table 3. Experimental evaluation on synthetic data for APC, K-NN, MDC, MLP and SVM classifiers using 10 percent of total data as training set and rest of the data as test set

Data	Method		% Accuracy	Macro F_1	Micro F_1	Time (in ms)
Annular	APC (δ=0.40)		98.464279 #1 (1.281758)	0.985941 #1 (0.012746)	0.986097 #1 (0.012504)	102.30 #3 (9.23)
	K-NN	K=5	91.194443 #2 (3.526837)	0.921799 #2 (0.030936)	0.923134 #2 (0.030281)	54.70 #2 (7.91)
		K=7	86.916664 (4.237551)	0.882359 (0.036768)	0.884641 (0.035331)	68.60 (7.79)
	MDC		28.222218 (2.975650)	0.252424 (0.023999)	0.267402 (0.021159)	10.15 #1 (7.46)
	MLP		84.58732 #3 (6.893613)	0.83166 #3 (0.10865)	0.86374 #3 (0.06685)	831.66 (108.6524)
	SVM		33.20636 (2.964732)	0.15676 (0.03461)	0.1611 (0.03776)	928 (102.0588)
Ellipse	APC (δ=0.35)		99.812973 #2 (0.175866)	0.998128 #2 (0.001761)	0.998134 #2 (0.001756)	386.70 #3 (8.36)
	K-NN	K=5	99.887039 #1 (0.136824)	0.998870 #1 (0.001368)	0.998874 #1 (0.001362)	236.20 #2 (4.62)
		K=7	99.833336 (0.196853)	0.998334 (0.001968)	0.998338 (0.001960)	282.90 (4.72)
	MDC		81.996292 (0.498130)	0.817814 (0.004231)	0.818785 (0.004754)	23.40 #1 (7.81)
	MLP		90.5037 #3 (4.439360)	0.90236 #3 (0.04565)	0.91072 #3 (0.043436)	5030 (54.40588)
	SVM		82.71852 (0.331782)	0.8238 (0.00373)	0.82576 (0.003383)	502 (91.9565)
Pat1	APC (δ=4.4)		77.345520 #1 (1.838376)	0.719335 #1 (0.034793)	0.721196 #1 (0.034889)	75.70 #3 (7.56)
	K-NN	K=5	69.268600 #2 (2.530912)	0.565623 #2 (0.028308)	0.576096 #2 (0.027571)	29.00 #2 (4.60)
		K=7	66.948288 (3.124440)	0.491452 (0.060469)	0.521020 (0.065949)	29.75 (5.48)
	MDC		32.036572 (6.634942)	0.249202 (0.034747)	0.332992 (0.033403)	7.75 #1 (7.76)
	MLP		55.42932 (3.497248)	0.43884 #3 (0.22571)	0.44898 #3 (0.23028)	1530 (67.8233)
	SVM		56.31312 #3 (4.179266)	0.26224 (0.032992)	0.2659 (0.04031)	450 (62.92853)
Pat2	APC (δ=0.0050)		80.756615 #1 (2.472527)	0.803176 #1 (0.026302)	0.804466 #1 (0.026496)	78.90 #3 (3.25)
	K-NN	K=5	72.061790 #2 (3.104161)	0.707022 #2 (0.032179)	0.710232 #2 (0.031780)	22.65 #2 (7.87)
		K=7	67.875153 (2.887981)	0.656549 (0.029284)	0.662427 (0.029878)	30.45 (6.06)
	MDC		47.679695 (2.599657)	0.482760 #3 (0.024480)	0.512758 (0.020395)	4.75 #1 (7.26)
	MLP		56.46462 #3 (3.636202)	0.46974 (0.052524)	0.52598 #3 (0.049856)	1530 (49.3963)
	SVM		53.28284 (1.628742)	0.26284 (0.06669)	0.26504 (0.071089)	548 (96.41577)
Spiral	APC (δ=0.04)		99.400009 #1 (0.847218)	0.993997 #1 (0.008480)	0.994102 #1 (0.008238)	100.75 #3 (7.67)
	K-NN	K=5	85.972237 #2 (3.711656)	0.859307 #2 (0.037474)	0.861510 #2 (0.036477)	29.70 #2 (4.75)
		K=7	77.127777 (3.292017)	0.770620 (0.032862)	0.772976 (0.033630)	34.35 (6.33)
	MDC		63.427776 #3 (1.722142)	0.633872 #3 (0.017464)	0.634552 #3 (0.017145)	8.65 #1 (7.83)
	MLP		59.9111 (5.208443)	0.56958 (0.08277)	0.6116 (0.04462)	1306 (77.3563)
	SVM		60.62222 (1.712132)	0.5968 (0.01712)	0.60872 (0.013036)	194 (73.648)

method. Note that for a wide range of δ values, the performance measures are observed to be fixed at nearly constant value or varies a little.

The CPU (execution) time, in milliseconds, needed by the algorithms are also given in the table for comparison. Rank of each algorithm is given depending on its performance measures and the execution time (separately) using '#' symbol followed by corresponding rank (from 1 to 3). For example '#1' indicates the best result with respect to either the corresponding performance measure or execution time.

4.4.1. Analysis of results of first kind of experiments (training with 10% data):

As mentioned earlier, for the first kind of experiments, 10% of data is taken out randomly to make the training set and the rest amount is considered as the test set. This process is repeated 20 times. Average result in terms of performance measures and execution time together with standard deviation (shown in bracket) of these 20 runs is reported in Table 2 and Table 3. Following section gives the detailed illustration and experimental outcome. The proposed aggregation pheromone density based classification method is compared with four other popular classification algorithms.

It is apparent from Table 2 that for real life data sets, in terms of percentage accuracy, macro averaged F_1 measure and micro averaged F_1 measure, the proposed APC performed the best for 6 data sets (Iris, WBC, Thyroid, Glass, Telugu Vowel and English Vowel) and the third best for 3 data sets (Sonar, Diabetes and Ionosphere), whereas SVM performed better for Sonar, Diabetes and Ionosphere data sets. Only in one case (Balancescale data) MLP outperformed other methods in terms of all performance measures.

For synthetic data sets, the proposed APC performed better for four out of five cases (except Ellipse data) in terms of all the performance measures compared to other classifiers. Only in case of Ellipse data K-NN outperformed others.

In terms of execution time, performance of minimum distance classifier (MDC) is the best, whereas K-NN and the proposed APC performed the $2nd$ best and the $3rd$ best respectively for all the data sets except Thyroid data. For Thyroid data K-NN (K=5) takes the least execution time.

On an average for most of the real life data sets considered, performance of the proposed APC is the best or is very close to the best one in terms of classification performance measures; and it takes less execution time than that of MLP and SVM, but more than MDC and K-NN.

4.4.2. Analysis of results of for the kind of experiments (k-fold cross validation):

In the second approach, k (=10) fold cross validation is used. The whole data set is randomly divided into k mutually exclusive and (nearly) equal sized subsets. For each subset, considered as the test set, the classifier is trained on the union of all the other subsets. Then, cross validation is run (total 10 times) for each training and test set pair. The results are averaged over all test set and training set pair in terms of performance measures and execution time together with standard deviation (shown in bracket). The results are reported in Tables 4 and 5.

From Table 4, it can be seen that the performance of the proposed APC method is the best for 6 data sets (namely Iris, WBC, Thyroid, Glass, Telugu Vowel, and Ionosphere) and the 2nd best for Balancescale data, whereas it performed the 3rd best for 3 data sets (Sonar, English Vowel, Diabetes). In case of Sonar, English Vowel and Balancescale data sets MLP outperformed other methods. For 4 data sets (Thyroid, Diabetes, Glass, Ionosphere) performance of the MLP is the 2nd best and the 3rd best for

Table 4. Experimental evaluation on real life data for APC, K-NN, MDC, MLP and SVM classifiers using 10 fold cross validation

Data	Method		% Accuracy	Macro F_1	Micro F_1	Time (in ms)
Iris	APC (δ=0.2)		96.666672 #1 (3.265985)	0.962399 #1 (0.038586)	0.966399 #1 (0.304879)	1.50 #3 (4.50)
	K-NN	K=5	96.666664 #2 (3.333332)	0.953933 (0.038478)	0.958891 (0.034205)	1.02 #2 (0.57)
		K=7	96.00 (3.333332)	0.961976 #2 (0.038851)	0.966029 #2 (0.034539)	1.60 (4.80)
	MDC		92.000000 (6.531973)	0.914886 (0.063683)	0.921216 (0.060490)	0.00 #1 (0.0000)
	MLP		96.4 #3 (0.5333)	0.96134 #3 (0.00268)	0.96146 #3 (0.00262)	428 (44.899)
	SVM		96.40002 (0.326615)	0.96394 (0.00326)	0.96454 (0.00326)	428 (52.6877)
WBC	APC (δ=2.9)		97.472890 #1 (2.229890)	0.974002 #1 (0.025074)	0.974394 #1 (0.035915)	20.40 #3 (7.17)
	K-NN	K=5	97.075874 (2.253604)	0.967985 (0.024345)	0.968359 (0.023970)	12.50 #2 (6.26)
		K=7	97.220802 #2 (2.304891)	0.969560 #2 (0.024754)	0.969868 #2 (0.024511)	15.60 (6.95)
	MDC		96.489769 (2.173517)	0.961090 (0.023392)	0.961387 (0.023348)	3.2000 #1 (6.4000)
	MLP		95.27898 (3.350445)	0.94776 (0.0338)	0.94778 (0.03379)	2716 (94.78)
	SVM		96.82402 #3 (0.107043)	0.96498 #3 (0.001144)	0.965 #3 (0.0012)	164 (34.9857)
Sonar	APC (δ=0.76)		76.92306 #3 (6.055346)	0.76734 #3 (0.063898)	0.768 #3 (0.059777)	7.90 #3 (7.91)
	K-NN	K=5	81.238098 #2 (3.996597)	0.805798 #2 (0.039728)	0.818402 #2 (0.027983)	7.70 #2 (7.71)
		K=7	75.928574 (5.857771)	0.749916 (0.056770)	0.762828 (0.050302)	9.60 (4.80)
	MDC		68.214294 (5.435662)	0.673079 (0.101016)	0.679201 (0.101306)	1.05 #1 (0.746)
	MLP		83.26922 #1 (1.53244)	0.83168 #1 (0.01546)	0.83178 #1 (0.01544)	15316 (79.39)
	SVM		75.880959 (0.860036)	0.742555 (0.00817)	0.762994 (0.00872)	200 (80)
Thyroid	APC (δ=1.1)		96.09306 #1 (5.167178)	0.94748 #1 (0.063673)	0.954424 #1 (0.306456)	4.60 #2 (7.03)
	K-NN	K=5	92.142860 #3 (5.836174)	0.872105 #3 (0.108553)	0.884973 #3 (0.103134)	1.50 #1 (4.50)
		K=7	90.735939 (4.120293)	0.837893 (0.088872)	0.860782 (0.086944)	1.60 (4.80)
	MDC		85.129875 (7.756683)	0.777132 (0.105355)	0.793752 (0.108443)	4.60 #2 (7.0314)
	MLP		95.865799 #2 (4.37208)	0.946259 #2 (0.04505)	0.94784 #2 (0.04519)	698 (41.66)
	SVM		89.3023 (5.294155)	0.83344 (0.055660)	0.85276 (0.05411)	428 (48.7442)
Glass	APC (δ=0.24)		68.744591 #1 (3.856398)	0.573391 #1 (0.037313)	0.596075 #1 (0.052334)	4.70 #3 (3.18)
	K-NN	K=5	64.523811 #3 (10.154930)	0.460668 #3 (0.085411)	0.481694 #3 (0.097158)	3.10 #2 (6.20)
		K=7	61.796532 (10.626472)	0.430365 (0.104615)	0.449541 (0.115375)	4.80 (7.33)
	MDC		39.199135 (8.481120)	0.350505 (0.120826)	0.381075 (0.128135)	1.50 #1 (4.50)
	MLP		66.93925 #2 (4.337086)	0.4924 #2 (0.046921)	0.510125 #2 (0.053051)	1797.5 (10.89725)
	SVM		57.1028 (0.68674)	0.3203 (0.003758)	0.34542 (0.00404)	2276 (141.929)

Table 4 Continued..

Data	Method		% Accuracy	Macro F_1	Micro F_1	Time (in ms)
Balance-scale	APC (δ=8.1)		87.987239 #2 (5.246451)	0.749494 #2 (0.092133)	0.757700#2 (0.254812)	20.20 #3 (7.30)
	K-NN	K=5	84.308754 (4.247334)	0.697587 (0.021098)	0.699380 (0.020356)	9.30 #2 (7.60)
		K=7	87.985153 #3 (3.505145)	0.713029 #3 (0.018146)	0.713865 #3 (0.018215)	14.00 (4.69)
	MDC		74.528938 (5.277471)	0.678964 (0.049212)	0.760272 (0.043933)	1.50 #1 (4.5000)
	MLP		91.104 #1 (3.45929)	0.82148 #1 (0.02968)	0.82432 #1 (0.02012)	1640 (87.4)
	SVM		87.904 (2.34465)	0.60996 (0.012394)	0.60996 (0.02039)	490 (68.4105)
Telugu Vowel	APC (δ=11.6)		85.991379 #1 (3.775894)	0.846151 #1 (0.043551)	0.851242 #1 (0.272442)	65.50 #3 (6.52)
	K-NN	K=5	85.876434 #2 (4.338057)	0.839221 #2 (0.054833)	0.844289 #2 (0.053589)	32.90 #2 (11.02)
		K=7	85.419266 (2.570991)	0.832156 (0.037082)	0.839496 (0.037432)	36.50 (7.10)
	MDC		71.068436 (3.583662)	0.685577 (0.048609)	0.698672 (0.044490)	2.74 #1 (1.647)
	MLP		83.05394 #3 (1.3452)	0.80662 #3 (0.02443)	0.80996 #3 (0.02416)	3454 (13.56)
	SVM		78.87488 (1.145243)	0.73016 (0.011515)	0.7487 (0.011915)	2090 (144.361)
English Vowel	APC (δ= 1.2)		70.606064 #3 (2.409467)	0.692361 (0.028975)	0.728935 #3 (0.231738)	67.20 #3 (7.08)
	K-NN	K=5	90.909081 #2 (3.226002)	0.904888 #2 (0.029228)	0.910257 #2 (0.028137)	34.20 #2 (6.16)
		K=7	85.050514 (3.187824)	0.847239 (0.027575)	0.858285 (0.026106)	40.40 (7.68)
	MDC		60.101006 (6.408372)	0.568240 (0.067130)	0.583804 (0.067154)	1.50 #1 (4.50)
	MLP		92.8283 #1 (2.75588)	0.9281 #1 (0.0277)	0.92908 #1 (0.02744)	33514 (346.2)
	SVM		70.40404 (1.20026)	0.70266 #3 (0.01065)	0.70382 (0.01089)	8380 (507.661)
Diabetes	APC (δ=14.3)		72.399185 #3 (2.677361)	0.707209 #3 (0.032043)	0.715370 #3 (0.228262)	26.90 #3 (7.15)
	K-NN	K=5	69.926521 (3.265679)	0.660209 (0.048316)	0.662434 (0.047569)	12.50 #2 (6.26)
		K=7	71.871155 (3.973670)	0.677057 (0.043629)	0.679883 (0.044213)	17.20 (4.62)
	MDC		63.672253 (6.262702)	0.591258 (0.067712)	0.595100 (0.067762)	1.58 #1 (4.27)
	MLP		74.89582 #2 (2.896056)	0.71788 #2 (0.0123)	0.71896 #2 (0.01163)	2800 (66.6)
	SVM		77.21354 #1 (1.184173)	0.72934 #1 (0.01248)	0.73866 #1 (0.01226)	212 (68.527)
Ionosphere	APC (δ=0.60)		89.473016 #1 (4.527817)	0.893275 #1 (0.049275)	0.8979 #1 (0.044130)	12.50 #3 (6.26)
	K-NN	K=5	83.198410 (5.890584)	0.788803 (0.072114)	0.815330 (0.061963)	4.70 #2 (7.60)
		K=7	82.626984 (5.755850)	0.777142 (0.081009)	0.806473 (0.068577)	6.20 (7.18)
	MDC		71.769844 (8.762713)	0.701233 (0.088937)	0.712467 (0.083369)	0.0000 #1 (0.0000)
	MLP		88.7983 #2 (0.72635)	0.874877 #2 (0.0084)	0.878668 #2 (0.00821)	9600 (76.5)
	SVM		88.37608 #3 (0.33225)	0.86558 #3 (0.00412)	0.87418 #3 (0.00374)	240 (67.5278)

Table 5. Experimental evaluation on synthetic data for APC, K-NN, MDC, MLP and SVM classifiers using 10 fold cross validation

Data	Method		% Accuracy	Macro F_1	Micro F_1	Time (in ms)
Annular	APC (δ=0.40)		100.00 #1 (0.00)	1.00 #1 (0.00)	1.00 #1 (0.00)	85.90 #3 (7.91)
	K-NN	K=5	100.00 #1 (0.00)	1.00 #1 (0.00)	1.00 #1 (0.00)	37.50 #2 (7.76)
		K=7	100.00 #1 (0.00)	1.00 #1 (0.00)	1.00 #1 (0.00)	47.00 (7.76)
	MDC		25.214289 (4.078940)	0.211035 #3 (0.029748)	0.221697 #3 (0.029450)	1.6000 #1 (4.8000)
	MLP		94.5714 #2 (1.3462)	0.94686 #2 (0.0133)	0.94842 #2 (0.0121)	3346 (76)
	SVM		35.7143 #3 (0)	0.1316 (0)	0.1316 (0)	902.5 (46.5698)
Ellipse	APC (δ=0.35)		99.100006 #2 (0.472582)	0.990936 #2 (0.004641)	0.991059 #2 (0.313417)	386.00 #3 (10.00)
	K-NN	K=5	100.00 #1 (0.00)	1.00 #1 (0.00)	1.00 #1 (0.00)	209.30 #2 (7.52)
		K=7	100.00 (0.00)	1.00 (0.00)	1.00 (0.00)	267.10 (4.66)
	MDC		81.833328 (1.815061)	0.815729 (0.020001)	0.816876 (0.020231)	1.6000 #1 (4.8000)
	MLP		98.30002 #3 (0.4967)	0.983 #3 (0.005)	0.9831 #3 (0.005)	5442 (35.4)
	SVM		82.1 (0.062365)	0.82005 (0.000642)	0.820175 (0.000642)	502.5 (113.880)
Pat1	APC (δ=4.4)		92.159088 #1 (3.600677)	0.924101 #1 (0.051472)	0.926410 #1 (0.049274)	72.00 #3 (10.26)
	K-NN	K=5	89.772728 (2.095351)	0.851910 (0.085776)	0.854222 (0.086123)	15.60 (6.95)
		K=7	91.409088 #2 (2.124440)	0.919219 #2 (0.047815)	0.924744 #2 (0.040493)	26.60 (7.17)
	MDC		31.249996 (7.262955)	0.226164 (0.048951)	0.261131 (0.052608)	0.0000 #1 (0.0000)
	MLP		62.2443 #3 (0.903286)	0.4033 #3 (0.010202)	0.403725 #3 (0.009932)	1552.5 (32.6917)
	SVM		58.9773 (4.179266)	0.2473 (0)	0.2473 (0)	503.333 (65.9966)
Pat2	APC (δ=0.005)		94.545456 #2 (1.818183)	0.944518 #2 (0.019194)	0.947499 #2 (0.017672)	76.40 #3 (4.80)
	K-NN	K=5	96.136368 (1.915035)	0.959809 (0.019588)	0.960424 (0.019116)	22.00 #2 (7.56)
		K=7	97.045456 #1 (2.500001)	0.969734 #1 (0.024594)	0.970206 #1 (0.024268)	28.30 (6.17)
	MDC		49.545452 (6.144775)	0.497767 (0.065957)	0.531423 (0.065048)	0.0000 (0.0000)
	MLP		64.34088 #3 (0.2203)	0.5703 #3 (0.0033)	0.614 #3 (0.0026)	1614 (52.8)
	SVM		56.47728 (0.592657)	0.37948 (0.00961)	0.40742 (0.007053)	452 (70.2567)
Spiral	APC (δ=0.05)		100.00 #1 (0.00)	1.00 #1 (0.00)	1.00 #1 (0.00)	92.10 #3 (8.57)
	K-NN	K=5	100.00 #1 (0.00)	1.00 #1 (0.00)	1.00 #1 (0.00)	21.90 #2 (7.65)
		K=7	100.00 #1 (0.00)	1.00 #1 (0.00)	1.00 #1 (0.00)	27.70 (6.25)
	MDC		64.900002 #3 (4.205948)	0.644116 #3 (0.042975)	0.644927 #3 (0.042825)	0.0000 #1 (0.0000)
	MLP		62.9 (1.71610)	0.628175 (0.017257)	0.6296 (0.017183)	1126 (565.7419)
	SVM		65.9 #2 (0.126491)	0.659 #2 (0.001265)	0.659 #2 (0.001265)	194 (71.4423)

Iris and Telugu Vowel data. SVM outperformed other methods for only Diabetes data; and for WBC and Ionosphere data it's performance is the 3rd best.

In case of considered synthetic data sets (refer Table 5), for Annular and Spiral data sets APC and K-NN both performed the best and they are 100% percent accurate. K-NN outperformed other methods for Ellipse and Pat2 data; in these cases APC performed the 2nd best, whereas for Pat1 data APC and K-NN performed the best and the 2nd best respectively. It is obvious from the experimental outcome that for Pat1, Pat2 and Spiral data sets MDC, MLP, and SVM failed to perform satisfactorily (as that of APC and K-NN). For Annular data also, performance of MDC and SVM is very poor; in this case MLP performed the 2nd best after APC and K-NN.

In terms of execution time MDC, K-NN and APC performed the best, the 2nd best, and the 3rd best respectively for both the real life and synthetic data sets except Thyroid data set. In case of Thyroid data set K-NN (K=5) takes the least execution time whereas APC and MDC takes 2nd least time.

In summary, using both kinds of experiments (discussed above), for the chosen real life as well as synthetic data sets, the proposed method, APC either performed the best or very close to the best performance produced by other classifiers with moderate execution time.

5. Conclusions

In this article we have proposed a computationally simple yet effective algorithm for pattern classification based on aggregation pheromone density, which is inspired by ants' property to accumulate around points with higher pheromone density. Performance of the proposed method is evaluated on 10 benchmark real life data sets as well as five synthetic data sets using three performance evaluation measures.

Comparative study of the experimental results on ten real life and synthetic data sets shows the potentiality of the proposed APC method. Time requirement is moderate.

Future work of the proposed method may be directed towards solving real world problems like land cover classification of remote sensing images, micro array gene classification, web-page classification and also to handle classification tasks for categorical data.

Acknowledgement

Support of the Department of Science and Technology, Govt. of India to the Center for Soft Computing Research is thankfully acknowledged by Mr. Anindya Halder, Research Scholar of the Center, Indian Statistical Institute, Kolkata.

References

[1] W. J. Bell. Chemo-orientation in walking insects. In W. J. Bell and R. T. Carde, editors, *Chemical Ecology of Insects*, pages 93–109, 1984.

[2] L. Chen, X. H. Xu, and Y. X. Chen. An adaptive ant colony clustering algorithm. In *Proceedings of the* 3^{rd} *Intrenational Conference on Machine Learning and Cybernetics*, pages 1387–1392, Shanghai, 2004.

[3] T. M. Cover and P. E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, IT-13(1):21–27, 1967.

[4] M. Dorigo and L. M. Gambardella. Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computing*, 1(1):53–66, 1997.

[5] M. Dorigo, V. Maniezzo, and A. Colorni. The ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics - part B*, 26(1):29–41, 1996.

[6] M. Dorigo and T. Stützle. *Ant Colony Optimization*. Prentice Hall of India Private Limited, New Delhi, 2005.

[7] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* Wiley, New York, 2000.

[8] A. P. Englebrecht. *Computational Intelligence: An Introduction*. John Wiley and Sons, 2002.

[9] A. Ghosh, A. Halder, M. Kothari, and S. Ghosh. Aggregation pheromone density based data clustering. *Information Sciences*, 178(13):2816–2831, 2008.

[10] A. Ghosh, M. Kothari, A. Halder, and S. Ghosh. Use of aggregation pheromone density for image segmentation. *Pattern Recognition Letters*, 2009. To appear.

[11] S. Ghosh, M. Kothari, and A. Ghosh. Aggregation pheromone density based image segmentation. In P. Kalra and S. Peleg, editors, *Proceedings of the* 5^{th} *Indian Conference on Computer Vision, Graphics and Image Processing*, pages 118–127, LNCS 4338. Springer-Verlag Berlin, Heidelberg, 2006.

[12] D. J. Hand and S. D. Jacka. *Discrimination and Classification*. Wiley, New York, 1981.

[13] J. Handl and B. Meyer. Ant-based and swarm-based clustering. *Swarm Intelligence*, 1:95–113, 2007.

[14] A. K. Jain, R. P. W. Duin, and J. Mao. Statistical pattern recognition: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000.

[15] J. Kennedy and R. C. Eberhart. *Swarm Intelligence*. Morgan Kaufman Publishers, 2001.

[16] M. Kothari, S. Ghosh, and A. Ghosh. Aggregation pheromone density based clustering. In S. P. Mohanty and A. Sahoo, editors, *Proceedings of the* 9^{th} *International Conference on Information Technology.*, pages 259–264. IEEE Computer Society Press, Los Alamitos, California, 2006.

[17] M. Kothari, S. Ghosh, and A. Ghosh. Aggregation pheromone density based change detection in remotely sensed images. In P. Pal, editor, *Proceedings of* 6^{th} *International Conference on Advances in Pattern Recognition*, pages 193–197, Singapore, 2007. World Scientific Publishing Co. Pvt. Ltd.

[18] S. R. Kulkarni, G. Lugosi, and S. S. Venkatesh. Learning pattern classification: A survey. *IEEE Transactions on Information Theory*, 44(6):2178–2206, 1998.

[19] L. I. Kuncheva. *Fuzzy Classifier Design*. Springer-Verlag, Heidelberg, 2000.

[20] B. Liu, H. A. Abbass, and B. McKay. Density-based heuristic for rule discovery with ant-miner. In *Proceedings of* 6^{th} *Australasia-Japan Joint Workshop on Intelligence Evololution System*, pages 180–184, Canberra, Australia, 2002.

[21] B. Liu, H. A. Abbass, and B. McKay. Classification rule discovery with ant colony optimization. In *Proceedings of IEEE/WIC International Conference Intellelligent Agent Technology*, pages 83–88, 2003.

[22] D. Martens, M. D. Backer, R. Haesen, J. Vanthienen, M. Snoeck, and B. Baesens. Data mining with an ant colony optimization algorithm. *IEEE Transactions on Evolutionary Computing*, 11(5):651–665, 2007.

[23] D. J. Newman, S. Hettich, C. L. Blake, and C. J. Merz. UCI repository of machine learning databases. University of California, Irvine, Department of Information and Computer Sciences, 1998. http://www.ics.uci.edu/∼mlearn/MLRepository.html.

[24] M. Ono, T. Igarashi, E. Ohno, and M. Sasaki. Unusual thermal defence by a honeybee against mass attack by hornets. *Nature*, 377(6547):334–336, 1995.

[25] S. K. Pal and D. Dutta Majumder. Fuzzy sets and decision making approaches in vowel and speaker recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 7:625–629, 1977.

[26] S. K. Pal and S. Mitra. Fuzzy versions of Kohonen's net and mlp-based classification: Performance evaluation for certain nonconvex decision regions. *Information Sciences*, 76:297–337, 1994.

[27] R. S. Parpinelli, H. S. Lopes, and A. A. Freitas. Data mining with an ant colony optimization algorithm. *IEEE Transactions on Evolutionary Computing*, 6(4):321–332, 2002.

[28] J. Platt. Using sparseness and analytic QP to speed training of support vector machines. In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in neural information processing systems*. MIT Press, 1999.

[29] B. D. Ripley and N. L. Hjort. *Pattern Recognition and Neural Networks*. Cambridge University Press, New York, NY, USA, 1995.

[30] S. R. Safavian and D. Landgrebe. A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(3):660–674, 1991.

[31] G. Salton and M. J. McGill. *An Introduction to Modern Information Retrieval*. McGrawHill, 1983.

[32] B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT press, 2002.

[33] K. Socha and M. Dorigo. Ant colony optimization for continuous domains. *European Journal of Operational Research*.

[34] M. Sukama and H. Fukami. Aggregation arrestant pheromone of the German cockroach, Blattella germanica (L.) (Dictyoptera: Blattellidae): isolation and structure elucidation of blasttellastanoside-A and B. *Journal of Chemical Ecology*, 19:2521–2541, 1993.

[35] S. Tsutsui. Ant colony optimization for continuous domains with aggregation pheromones metaphor. In *Proceedings of the* 5^{th} *International Conference on Recent Advances in Soft Computing*, pages 207–212, United Kingdom, December 2004.

[36] S. Tsutsui and A. Ghosh. An extension of ant colony optimization for function optimization. In *Proceedings of the* 5^{th} *Asia Pacific Conference on Simulated Evolution and Learning*, Pusan, Korea, 2004.

[37] X. N. Wang, Y. J. Feng, and Z. R. Feng. Ant colony optimization for image segmentation. In *Proceedings of the* 4^{th} *International Conference on Machine Learning and Cybernetics*, pages 5355–5360, 2005.

[38] Weka Machine Learning Project. Weka. URL http://www.cs.waikato.ac.nz/~ml/weka.