Aggregation pheromone metaphor for semi-supervised classification

Anindya Halder a, Susmita Ghosh c, Ashish Ghosh b, *

a Department of Computer Application, North-Eastern Hill University, Tura Campus, Tura 794002, Meghalaya, India
b Center for Soft Computing Research, Indian Statistical Institute, Kolkata 700108, India
c Department of Computer Science & Engineering, Jadavpur University, Kolkata 700032, India

A R T I C L E   I N F O

Article history:
Received 9 February 2012
Received in revised form 16 November 2012
Accepted 3 January 2013
Available online 11 January 2013

Keywords:
Semi-supervised classification
Self-training
Ant colony
Aggregation pheromone

A B S T R A C T

This article presents a novel ‘self-training’ based semi-supervised classification algorithm using the property of aggregation pheromone found in real ants. The proposed method has no assumption regarding the data distribution and is free from parameters to be set by the user. It can also capture arbitrary shapes of the classes. The proposed algorithm is evaluated with a number of synthetic as well as real life benchmark datasets in terms of accuracy, macro and micro averaged $F_1$ measures. Results are compared with two supervised and three semi-supervised classification techniques and are statistically validated using paired t-test. Experimental results show the potentiality of the proposed algorithm.

© 2013 Elsevier Ltd. All rights reserved.

1. Introduction

Traditional machine learning methods for pattern classification require sufficient number of labeled data to assign an unlabeled pattern to a certain class. However, labeled patterns are often difficult, costly, and/or time consuming to obtain, as they require the effort of experienced human annotators. On the other hand, unlabeled data may be relatively easy to gather. Semi-supervised learning (classification) [7] methods make use of the large amount of available unlabeled data, along with a small amount of labeled data, to improve classification accuracy. As semi-supervised classification requires less human intervention and produces better accuracy, it is of great interest to the machine learning researchers in recent years.

A variety of semi-supervised learning methods exist in the literature. These can be broadly categorized as follows: self-training [38,45], co-training [5], transductive support vector machines (TSVMs) [2,8,9,44], graph-based methods [3,4,47], expectation maximization (EM) with generative mixture models [31] etc. A good review of semi-supervised classification methods is available in [48,49].

Self-training is a wrapper based method commonly used for semi-supervised learning. In this process a classifier is first trained using a small amount of labeled data. Then unlabeled data patterns are classified using the trained classifier. The classified (unlabeled) pattern whose predicted value is sufficiently high for belonging to a certain class is added to the training set along with its predicted class label. This is done for all the classified (unlabeled) patterns. Thus, the amount of training data increases due to the inclusion of the “high confidence” unlabeled patterns in the original training set. Re-training of the classifier is done using the new enlarged training set; and the procedure is repeated. One can imagine that a misclassification can reinforce itself. Therefore, some algorithms are used to avoid this problem by unlearning the unlabeled points if the prediction confidence drops below a threshold. Self-training has been applied to several natural language processing tasks [36,45]. Self-training was also applied to object detection from images [38], and showed that semi-supervised techniques perform better compared to the state-of-the-art object detectors. Self-training based semi-supervised method was used for motion estimation in dynamic systems [22]. Also self-training semi-supervised support vector machine (SVM) was proposed for electroencephalogram (EEG) based brain–computer interface system [24].

In co-training [5,29], features are split into two sets. Following two assumptions are considered in co-training: (i) each sub-set (of features) is sufficient to train a good classifier, and (ii) given a class, the two sets are conditionally independent. Initially, two separate classifiers are individually trained with the labeled data, on the two sub-sets. Co-training then utilizes the unlabeled data by adding the most confident predictions of one classifier to the training set of the other classifier; thereby, effectively allowing each individual classifier to train its counterpart.

Transductive support vector machine (TSVM) [44] is an extension of standard support vector machine for dealing with unlabeled
data. In standard SVM, only the labeled data is used, and the goal is to find a maximum margin linear boundary in the Kernel Hilbert Space [10]. In TSVM the unlabeled data is also used to find a labeling of the unlabeled data, so that a linear boundary has the maximum margin on both the original labeled data and the unlabeled data (with predicted label). The decision boundary has the smallest generalization error bound on unlabeled data [43]. Intuitively, unlabeled data guides the linear boundary away from denser regions. Initially developed TSVM algorithms [2] were not able to handle a large number of unlabeled data efficiently. To overcome this drawback, many variants like SVMLight-TSVM [21], low density separation (LDS) [8], concave-convex procedure (CCCP) for TSVM (CCCP-TSVM) [9] have been proposed in the literature. A semi-supervised support vector classifier designed using a quasi-Newton method for nonsmooth convex functions is proposed in [35].

Graph-based semi-supervised methods construct a graph where the nodes designate the labeled and unlabeled samples of the dataset and (weighted or unweighted) edges represent the similarity of samples. These methods are nonparametric, discriminative, and transductive in nature. Some of the graph based methods like mincut [34], harmonic [15], local and global consistency [47], manifold regularization [1] are discussed in [48,49]. A few recent graph-based semi-supervised classifiers are proposed particularly for graph construction [46], handling multiple graphs in gene networks [41], neighborhood graph construction [37], and betweenness computation on large sparse directed graphs [27].

Generative models [7] are possibly the oldest semi-supervised learning method. It assumes a probabilistic model where identifiable mixture distribution is known. With a large amount of unlabeled data, the mixture components can be identified. The model using EM method is being used for text classification [31].

In this article, a novel ‘self-training’ based semi-supervised algorithm is proposed using the aggregation pheromone density which is inspired by the natural behavior of real ants and other social insects. Different applications originated from the study of different types of swarms. Among them, most popular ones are ant colony and bird flocks [12]. Ant Colony Optimization (ACO) [11] and Aggregation Pheromone Systems (AP) [42] are computational algorithms modeled on the behavior of ant colony. ACO [11] 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 termed as aggregation pheromone [42]. Thus, aggregation pheromone causes individuals to aggregate around good positions which in turn produces more pheromone to attract individuals of the same species. In AP [42], a variant of ACO, this behavior of ants is used to solve real parameter optimization problems.

Inspired by the aggregation pheromone system found in ants and other similar agents, in earlier works, attempts were made for solving clustering [13], classification [16], image segmentation [14] problems, and land use map generation from multispectral remotely sensed images [17] with encouraging results.

Though a large number of techniques exists for ant based unsupervised classification (i.e., clustering) in the literature [19], a few attempts [25,26,28,34] have been made for (supervised) classification. In our earlier (conference) work one preliminary attempt [18] was also made for semi-supervised classification based on ant colony approach with promising results.

Motivated from the promising results, the earlier research [18] has been extended in this article to propose an advanced aggregation pheromone density based semi-supervised classification (called, APSSC) algorithm.

The proposed APSSC algorithm is ‘self-training’ in nature and consists of two phases namely, ‘self-training’ and ‘testing’. In self-training phase the classifier is first trained with the small amount of labeled (patterns) ants. Afterwards, the classifier is used to classify the unlabeled ants; and then among the unlabeled ants, the ‘high confidence’ ones are determined and they are added (together with their predicted labels) to the corresponding (class) colony in the training set. The classifier is re-trained (using the newly formed training set) and this procedure is repeated until colony formation is stabilized. In this way, a new enlarged training set is built. Once the colony formation is stabilized, in the testing phase, each test (pattern) ant is evaluated to assign to the colony for which the average aggregation pheromone density is more. In this way the classification accuracy obtained with a small amount of labeled ant can be improved by the use of the ‘high confidence’ unlabeled ants.

The proposed method has the advantage of not having any assumption regarding the data distribution. Moreover, opposed to the earlier version [18], it does not require to set any free parameter manually. In addition, it can better capture the arbitrary shapes of the classes by updating the covariance matrices of the classes with iterations.

The proposed semi-supervised classifier is compared with two conventional supervised classifiers (viz., multi layer perceptron and support vector machine) and three state-of-the-art semi-supervised classifiers (viz., semi-supervised classification by low density separation [8] and concave-convex procedure for transductive support vector machine [9], self-training semi-supervised support vector machine [24]) using five real life benchmark datasets and four artificially generated datasets. Performance of each of the methods is evaluated using percentage of overall accuracy, macro averaged $F_1$ measure, and micro averaged $F_1$ measure. Results of the investigations of the semi-supervised methods are also statistically validity using paired t-test [23]. Experimental results show the potentiality of the proposed semi-supervised method compared to other techniques for most of the datasets.

The rest of the article is organized as follows. Section 2 provides a detail description of the proposed ant based semi-supervised classification method using aggregation pheromone system. Details of the experiments and analysis of results are provided in Section 3, and finally, conclusions are drawn in Section 4.

2. Proposed methodology

As mentioned earlier, aggregation pheromone brings individuals into closer proximity. This group forming nature of aggregation pheromone (found in natural behavior of real ants) is being used as the basic idea of the proposed technique.

The proposed aggregation pheromone density based semi-supervised classification (APSSC) algorithm is ‘self-training’ in nature. It consists of two steps. The first step uses ‘self-training’ strategy, where the semi-supervised classifier is (re)trained iteratively using the small number of labeled ants along with the ‘high confidence’ unlabeled ants (described latter). The second step is ‘testing’. Once self-training is over (i.e., colony is stabilized) the new test ants are predicted to assign a particular class (colony) in the testing phase. The details of the proposed methods are described below.

2.1. Aggregation pheromone density based semi-supervised classification

Consider a dataset with $K$ classes and a small number of labeled data patterns from each class which, by our assumption,
forms $K$ homogeneous groups or colonies of ants in the training/labeled set $L$. Also, there is a (relatively large) $|U|$ number of unlabeled data patterns in the unlabeled set $U$.

Let $x_{i1}^r, x_{i2}^r, \ldots, x_{in}^r$ be the given original training data or labeled data patterns in the $k$th training colony $C_k^r$. These patterns are considered as a population of $|C_k^r|$ number of ants represented as $a_{i1}^r, a_{i2}^r, \ldots, a_{in}^r$. Hence, an ant $a_{ij}^r$ represents the $i$th training data pattern (in the $k$th initial training class) $x_{ij}^r \in C_k^r$.

Consider $x_{i1}^r, x_{i2}^r, \ldots, x_{in}^r$ to be the unlabeled data patterns represented as unlabeled ants $a_{i1}^u, a_{i2}^u, \ldots, a_{in}^u$, correspondingly.

Step 1: Self-training. At iteration $t = 0$, only labeled ants (patterns) are considered to form the initial training colony, i.e., the $k$th training colony $C_k^r$ is the same as $k$th initial training class/colony $C_k^0$.

Each labeled ant emits pheromone at its neighborhood. The intensity of pheromone emitted by the $i$th individual labeled ant $a_{ij}^r \in C_k^r$ located at $x_{ij}^r$ at iteration $t$ decreases with increase in its distance from $x_{ij}^r$. Thus, the pheromone intensity at a point closer to $x_{ij}^r$ is more than those at other points that are farther from it. To achieve this, the pheromone intensity emitted by ant $a_{ij}^r \in C_k^r$ at time $t$ is modeled by a Gaussian distribution. Hence, effect of the emitted pheromone density on the $j$th unlabeled ant $a_{kj}^u$ (located at $x_{kj}^u$) at iteration $t$ due to the $i$th labeled ant of $k$th colony ($a_{ij}^r \in C_k^r$) located at $x_{ij}^r$ is given by

$$\Delta^{t}(x_{ij}^r, x_{kj}^u) = \frac{1}{(2\pi)^{d/2} |\Sigma_k^t|^{1/2}} \exp \left(-\frac{1}{2} (x_{kj}^u - x_{ij}^r)^T (\Sigma_k^t)^{-1} (x_{kj}^u - x_{ij}^r) \right).$$

(1)

where, $\Sigma_k^t$, det$(\Sigma_k^t)$, and $d$ represent respectively, the covariance matrix of the $k$th class at iteration $t$, the determinant of the covariance matrix $\Sigma_k^t$, and number of dimensions of the dataset used.

The average effect of emitted (aggregated) pheromone on the $j$th unlabeled ant $a_{kj}^u$ due to $k$th training colony $C_k^r$ at iteration $t$ is given by

$$\Delta^{t}_k = \frac{1}{|C_k^r|} \sum_{m \in C_k^r} \Delta^{t}(x_{jm}^r, x_{kj}^u); \quad \forall j, \forall k.$$  

(2)

Thereafter, pheromone density $\tau_{jk}^t$ due to the $k$th colony $C_k^r$ on the $j$th unlabeled (pattern) ant at iteration $t$ is updated according to the following equation:

$$\tau_{jk}^t = (1 - \rho) \tau_{jk}^{t-1} + \rho \Delta^{t}_k; \quad \forall j, \forall k,$$

(3)

where, $0 \leq \rho \leq 1$ is the evaporation constant. With smaller values of $\rho$, the system uses lower values of the pheromone density of the past cycles more than with the larger values of $\rho$. Larger value of $\rho$ indicates that the effect of the pheromone emitted in the present iteration is more compared to the pheromone emitted in the previous iterations. $\rho$ acts as a trade-off factor of the emitted pheromone in the previous and the present iterations. Instead of keeping it constant [18] throughout the self-training process, it is reasonable to vary it with respect to time. As in the self-training process, the emitted pheromone (in the current iteration) at the location of an unlabeled ant is computed mainly due to the training ants; also, there is no (or, less) effect of the emitted pheromone from the earlier cycle. Therefore, the effect of emitted pheromone ($\Delta^{t}_k$) in the current iteration should be high during the initial stage. As time progresses the effect of the emitted pheromone in the current iteration should decrease and the effect of the pheromone density of the past cycles ($\tau_{jk}^{t-1}$) should increase. Hence, $\rho$ is a function of time and it is defined as

$$\rho = \frac{1}{1 + \log(t + 1)}.$$  

(4)

After pheromone density is updated, the gradation of belonging of an unlabeled ant $a_{kj}^u$ to colony $C_k^r$ is computed as

$$\mu_{kj}^r = \frac{\tau_{jk}^t}{\sum_{h=1}^{K} \tau_{jh}^t}; \quad \forall j, \forall k.$$  

(5)

This $\mu_{kj}^r$ is nothing but the normalized pheromone density (n.p.d.) at the location of an unlabeled ant $a_{kj}^u$ due to colony $C_k^r$.

Once the normalized pheromone density (n.p.d.) values of all the unlabeled ants are determined, ants are evaluated to be temporarily added to the training set for the next iteration ($t+1$). Ants, added to the training set, are termed as 'high confidence ants'. Evaluation of the unlabeled ants is done as follows.

2.1.1. Determination of 'high confidence' ants

As stated earlier, let $\mu_{kj}^r$ be the normalized pheromone density (n.p.d.) value associated with an unlabeled ant $a_{kj}^u$ due to the colony $C_k^r$. Let the highest normalized pheromone density corresponding to the unlabeled ant $a_{kj}^u$ be $\mu_{kj}^r = \max_k(\mu_{kj}^r)$. The ratio

$$\frac{\mu_{kj}^r}{\mu_{kj}^r} (\forall k \neq h)$$

represents the degree of similarity of an unlabeled ant $a_{kj}^u$ belonging to colony $C_k$ and the highest contributing colony $C_h$. Range of this ratio is in $[0,1]$. More the value of the ratio, more is the similarity of the unlabeled ant with two colonies $C_k$ and $C_h$; hence, less is the confidence (of the unlabeled ant for belonging to any colony). Therefore, if all such ratios between $\mu_{kj}^r(\forall k \neq h)$ and the highest n.p.d. value $\mu_{kj}^h$ of the unlabeled ant $a_{kj}^u$ are less than equal to $1/K$, (where, $K$ is the number of colony or class) then that unlabeled ant $a_{kj}^u$ becomes a 'high confidence' ant to be added to the training set for the next iteration ($t+1$). If for any colony $C_k$ the ratio

$$\frac{\mu_{kj}^r}{\mu_{kj}^r} (\forall k \neq h)$$

is greater than $1/K$, then the corresponding ant is considered to be a less confidence one, and is not added to any colony.

The methodology for determining the 'confidence' of an unlabeled ant $a_{kj}^u$ is summarized in Algorithm 1.

**Algorithm 1.** Determination of the 'confidence' of an unlabeled ant $a_{kj}^u$

1: for each n.p.d. $\mu_{kj}^r$ ($k \neq h$) due to $k$th training colony $C_k^r$ at iteration $t$ do
2: if $\left(\frac{\mu_{kj}^r}{\mu_{kj}^h} = \max_k(\mu_{kj}^r) \leq \frac{1}{K}\right)$ then
3: flag_variable = 1
4: else
5: flag_variable = 0
6: break;
7: end if
8: end for
9: if (flag_variable = 1) then
10: Add the unlabeled ant $a_{kj}^u$ to the appropriate training colony ($C_{k+1}^r$) for the next iteration ($t+1$) as $C_{k+1}^r = C_h^r \cup a_{kj}^u$
11: else
12: Do not add the unlabeled ant $a_{kj}^u$ to any colony.
13: end if

Note that, addition of an ant to the colony $C_h$ is done temporarily for the next iteration. In subsequent iterations it will
be added to the appropriate colony depending on its current membership value or it may not be included in any colony. Hence, in each iteration (re)assignment of the initial unlabeled ants occurs.

2.1.2. Stopping criterion of self-training

The self-training phase of the algorithm stops when there is no (re)assignment. This is done by computing the colony centers. If the colony centers in two successive iterations do not change then it can be said that there is no (re)partition. At that time colony formation by the unlabeled ants is over and the unlabeled ants are stabilized. It means either they have joined any colony with sufficient confidence, or (rest) have not joined any colony (with sufficient confidence). The unlabeled ants, which have joined in any colony are now considered as training samples, and thus, the size of the training set is increased with the help of the unlabeled patterns.

Step 2: Testing. After the colony formation (by the unlabeled ants) is over, the new ants (patterns) are tested as follows. If the test ant $a_n$ at $x_n$ appears in the system, the average aggregation pheromone density (at the location of the new ant $a_n$) by the colony $C_i$ is given by [as in Eq. (2)]

$$
\Delta \tau_{nk} = \frac{1}{|C_k|} \sum_{u \in C_k} \left( \frac{1}{(2\pi)^{D/2} |\text{det}(\Sigma_k)|^{1/2}} \exp\left(-\frac{1}{2}(x_n-x_j^u)^T \Sigma_k^{-1}(x_n-x_j^u)\right) \right)
$$

(6)

The test ant $a_n$ will move toward a colony for which the average aggregation pheromone density (at the location of that test ant) is higher than that of the other colonies. Hence, finally the said ant will join the colony that will be governed by the following equation:

$$
\text{ColonyLabel}(x_n) = \arg \max_k (\Delta \tau_{nk}).
$$

(7)

As opposed to the previous version [18], the present version of the algorithm does not have any free parameters to be set by the user manually; also in each iteration the algorithm updates the class covariance matrix (of Eq. (1)) and thereby is able to better capture the shape of the classes. The complete procedure is summarized in Algorithm 2.

Algorithm 2. Aggregation Pheromone density based Semi-Supervised Classification (APSSC)

1: \textbf{begin self-training()}
2: Initialize: Iteration counter $t \leftarrow 0$; initial pheromone density $\tau_{jk} \leftarrow 0$, $\forall j, k$.
3: \textbf{repeat}
4: \textbf{for} each unlabeled ant $a^u_j$ located at $x^u_j$ \textbf{do}
5: \textbf{for} each training colony $C_i$ at iteration $t$ \textbf{do}
6: \hspace{1em} Calculate the average aggregation pheromone density $\Delta \tau_{nk}$ on the $j$th unlabeled ant $a^u_j$ due to all ants in present training colony $C_i$ at iteration $t$ using equation (2).
7: \hspace{1em} Update pheromone density $\tau_{jk}$ due to $k$th colony $C_k$ on the $j$th unlabeled (pattern) ant at iteration $t$ by Equation (3).
8: \hspace{1em} \textbf{end for}
9: \textbf{for} each training colony $C_i$ at iteration $t$ \textbf{do}
10: \hspace{1em} Compute the n.p.d. $\mu_{jk}$ of each unlabeled ant $a^u_j$ due to each colony $C_i$ at iteration $t$ using Equation (5).
11: \hspace{1em} \textbf{end for}
12: \hspace{1em} Compute the ‘confidence’ of the unlabeled ant $a^u_j$ and add or do not add the ant to the appropriate colony for the next iteration ($t+1$) according to Algorithm 1.
13: \textbf{end for}
14: $t \leftarrow t+1$.
15: \textbf{until <StoppingCriteria>}
16: \textbf{end self-training}

\textbf{begin testing()}
\hspace{1em} for each new test ant $a_n$ located at $x_n$ \textbf{do}
\hspace{2em} for each colony $C_i$ \textbf{do}
\hspace{3em} Calculate the average aggregation pheromone density $\Delta \tau_{nk}$ at location $x_n$ due to all ants in colony $C_i$ using Eq. (6).
\hspace{3em} \textbf{end for}
\hspace{2em} Compute the ColonyLabel($x_n$) of the ant $a_n$ by Eq. (7). // Ties are broken arbitrarily.
\hspace{1em} \textbf{end for}
\textbf{end testing}

3. Experimental evaluation

In this section we report the details of experimental setup, along with the datasets and then analyze the results.

3.1. Datasets used

For the purpose of our study, we used four artificially generated datasets (shown in Fig. 1), and five real life data sets, four among them are from the UCI repository [30], and Telugu Vowel data is from [32].

Among the synthetic datasets Annullar dataset (Fig. 1(a)) has four concentric rings of different radius representing four classes having total 1400 data patterns. Ellipse data (Fig. 1 (b)) consists of two very close ellipse, partially confined within a half ellipse representing the three classes having 300 data patterns. Pat2 data (Fig. 1 (c)) [33] consists of three classes having 880 patterns. Spiral data (Fig. 1 (d)) contains 1000 data points distributed in two spirals shaped classes. All the synthetic datasets have two dimensions. Please note that, all the synthetic datasets are nonlinearly separable.

Among the real life datasets, the Ionosphere is a radar data which consists 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 kW. 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. The Indian Telugu Vowel data [32] is the formant frequency of sounds in consonant-vowel-consonant context uttered by three speakers in the age group 30–35 years. The dataset consists of 871 instances with three formant frequencies (features) which were obtained through the spectrum analysis of the speech data. The data patterns are distributed in six overlapping classes and their boundaries are ill-defined. Balance Scale data was generated to model the psychological experimental results. It has 625 instances described by four features, distributed in three classes. Sonar data has 208 instances described by 60 attributes distributed in two classes. Wisconsin Breast Cancer (WBC) data contains 699 instances distributed in two classes namely “good” and “bad”.

To test the classification accuracy, 5% of data is taken out randomly from a dataset to form the initial training set and the rest is considered as the unlabeled set. The process is repeated 10 times. The reported results are obtained considering the unlabeled data as the test set. A summary about the datasets is given in Table 1.
3.2. Methods compared with

The proposed method is compared with two traditional classifiers: multi-layer perceptron (MLP) [20], and support vector machine (SVM) [40], along with three semi-supervised techniques, namely, semi-supervised classification by low density separation (LDS) [8], concave-convex procedure for transductive support vector machine (CCCP-TSVM) [9] and self-training semi-supervised support vector machine (SS-SVM) [24]. For implementation of SVM and SS-SVM we have used the package as referred in [6] package. We have used the available source codes of LDS, and CCCP-TSVM, respectively, from [8], and [9] for the implementation. Note that, compared methods have a number of parameters. The MLP and APSSC algorithms are implemented in Matlab. We have suitably adjusted the parameters to get the optimum results.

3.3. 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. Namely, (i) percentage accuracy, (ii) macro averaged $F_1$ measure, and (iii) micro averaged $F_1$ measure.

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

**Macro averaged $F_1$ measure**: Macro averaged $F_1$ is derived from precision and recall [39]. The precision ($p_k$) of class $k$ is defined as

$$p_k = \frac{\# \text{ patterns correctly classified into class } k}{\# \text{ patterns classified into class } k},$$

and, recall ($r_k$) of class $k$ is defined as

$$r_k = \frac{\# \text{ patterns correctly classified into class } k}{\# \text{ patterns that are truly present in class } k}.$$  

Then $(F_1)_k$, the harmonic mean between precision and recall, of class $k$ is defined as

$$(F_1)_k = \frac{2 \times p_k \times r_k}{p_k + r_k}.$$  

$F_1$ 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 $F_1$ gives equal weight to each category.
Macro averaged $F_1$ measure (denoted as Macro $F_1$, in short) is defined as

$$\text{Macro } F_1 = \frac{1}{R} \sum_{k=1}^{K} (P_k)_{kk},$$

(11)

where $K$ is the number of categories (classes).

**Micro averaged $F_1$ measure:** It is computed by first creating a global contingency table whose cell values are sum of the corresponding cells in the per-category contingency tables. Then this global contingency table is used to compute the micro averaged performance scores. Micro averaged $F_1$ gives equal weight on each sample (test case).

Micro averaged $F_1$ measure (denoted as Micro $F_1$, in short) is defined as

$$\text{Micro } F_1 = \frac{2 \times \frac{1}{K} \sum_{k=1}^{K} p_k \times \frac{1}{K} \sum_{k=1}^{K} r_k}{\frac{1}{K} \sum_{k=1}^{K} p_k + \frac{1}{K} \sum_{k=1}^{K} r_k},$$

(12)

where $K$ is the number of categories (classes).

---

**3.4. Statistical significance test**

To test the significance of results statistically (in terms of percentage accuracy) of the investigation, paired t-test [23] has been performed with the proposed APSSC versus other semi-supervised methods at 5% level of significance, and results of t-test in terms of $p$-score are reported in Table 4.

**3.5. Experimental results and analysis**

The average results and standard deviations (shown in bracket) for 10 simulation runs (with 10 different labeled, unlabeled/test sets) of all the algorithms are reported in Tables 2 and 3 for synthetic and real life datasets, respectively. The CPU (execution)
time, in seconds, needed by the algorithms are also given in the table for comparison.

All the algorithms used in this article are implemented in Matlab and simulated in core 2 duo (2.2 GHz speed) processor using 2 GB of main memory in Windows environment.

Rank of each algorithm is given depending on its performance index using ‘#' symbol followed by corresponding rank (from 1 to 3).

For example ‘#1’ indicates the best result with respect to the corresponding performance index. The best results are also marked as bold.

It is seen from the experimental outcome of synthetic data (Table 2) that the proposed APSSC algorithm outperforms the other semi-supervised counterpart (LDS, CCCP-TSVM and SS-SVM), for Annular, Pat2 and Spiral datasets in terms of classification
accuracy, macro averaged $F_1$ (denoted as, Macro $F_1$ in tables) and 
micro averaged $F_1$ (denoted as, Micro $F_1$ in tables). In particular, 
the accuracy of the APSSC for the Spiral data is 100%. Also the 
small standard deviation of the average results produced by 
APSSC suggest the robustness of the proposed method with 
variation of the training sets. Though for Ellipse dataset LDS 
performs slightly better than other semi-supervised methods, 
still the accuracy produced by APSSC is as high as 99.61%. The 
very high value (> 0.9) of Macro $F_1$ and Micro $F_1$ measure 
produced by APSSC in case of Annular, Ellipse, and Spiral data 
indicates the very high (classwise) precision, and recall rates. 
Analyzing the high accuracy produced by the proposed method 
trained with very limited training samples (only 5% of the total 
data) on synthetic datasets with arbitrary geometrical classes 
suggests the effectiveness of the proposed method in capturing 
the different geometrical shapes having nonlinearly separable and 
nonconvex class distributions. In these cases, the performance of 
the supervised classifiers (MLP and SVM) trained with limited 
training samples are very poor.

For real life datasets (Table 3), the proposed APSSC is observed to 
perform better in terms of classification accuracy, macro averaged $F_1$ 
and micro averaged $F_1$ in three cases namely Telugu Vowel, Balance 
Scale and Sonar datasets. The improvement found by the proposed 
method over other methods is significantly high. For example, the 
improvement of the proposed algorithm in Telugu Vowel, Balance 
Scale and Sonar data compared to the second best method are 3.07%, 
2.78%, and 4.39%, respectively. Whereas, in case of the WBC dataset 
the accuracy produced by the APSSC is only 0.24% less as 
compared to the best one (CCCP-TSVM).

It is worth mentioning here that, as expected, all the semi-
supervised classifiers clearly dominate the supervised classifiers 
(MLP and SVM). This is because the use of the unlabeled patterns 
really helps to gain accuracy in semi-supervised case even though 
the number of training samples used in both the cases (supervised 
and semi-supervised) are the same.

As mentioned earlier, results of the investigation (in terms of 
percentage accuracy) are statistically validated using the paired 
t-test [23] performed with the proposed APSSC versus other semi-
supervised methods at 5% level of significance. Results of paired 
t-test in terms of p-score are reported in Table 4. Statistically 
significant results in terms of p-score of the paired t-test (at 5% 
level of significance) are marked as bold in Table 4. The up-
arrow (↑), and down-arrow (↓) are also shown in the table along 
with the p-scores to indicate, respectively, the significant 
 improvement found by the proposed method APSSC (compared to 
the other semi-supervised method), and significant improve-
ment found by the other method (compared to the proposed 
APSSC) during the paired t-test.

From the paired t-test it is found that in case of Annular and 
Ellipse datasets improvement in performances of the proposed 
APSSC method is statistically significant (at 5% level) compared to 
the other semi-supervised counterparts.

Table 4
Results of paired t-test performed with proposed APSSC versus other semi-
supervised methods in terms of p-score.

<table>
<thead>
<tr>
<th>Data</th>
<th>APSSC vs. LDS</th>
<th>APSSC vs. CCCP-TSVM</th>
<th>APSSC vs. SS-SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Annular</td>
<td>0.082</td>
<td>0.4963</td>
<td>0.001↑</td>
</tr>
<tr>
<td>Ellipse</td>
<td>0.0021↑</td>
<td>0.3449</td>
<td>0.0251↑</td>
</tr>
<tr>
<td>Pat2</td>
<td>$1.58 \times 10^{-7}$↑</td>
<td>0.0835</td>
<td>$7.48 \times 10^{-4}$↑</td>
</tr>
<tr>
<td>Spiral</td>
<td>0.0043↑</td>
<td>0.3378</td>
<td>0.0063↑</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>0.1123</td>
<td>0.0285</td>
<td>0.4398</td>
</tr>
<tr>
<td>Telugu Vowel</td>
<td>$9.79 \times 10^{-3}$↑</td>
<td>0.0173</td>
<td>0.0301↑</td>
</tr>
<tr>
<td>Balance Scale</td>
<td>0.0107↑</td>
<td>0.1223</td>
<td>0.0086↑</td>
</tr>
<tr>
<td>Sonar</td>
<td>0.0014↑</td>
<td>0.0254</td>
<td>0.0207↑</td>
</tr>
<tr>
<td>WBC</td>
<td>0.0138↑</td>
<td>0.0492</td>
<td>0.0085↑</td>
</tr>
</tbody>
</table>

4. Conclusions

This article presents a novel ‘self-training’ based semi-super-
vised classification algorithm using the metaphor of the aggrega-
tion pheromone found in natural behavior of real ants. The 
performance of the proposed method is compared with two 
supervised (namely, MLP and SVM) and three semi-supervised 
classification techniques (namely, LDS, CCCP-TSVM, and SS-SVM). 
The proposed method has the following advantages. (i) No free 
parameters need to be set by the users, (ii) during the self-
training process in each iteration the method updates the covar-
iance matrix of each class, and thereby it is able to capture the 
shapes of the classes, (iii) as the algorithm has no assumption 
regarding the data distribution, therefore, it can be applied for 
datasets having arbitrary distribution. On the other hand, the 
other semi-supervised methods have many parameters to be set 
by the user, and the optimal performance of the algorithm is 
sensitive to the choice of the parameter values (which varies with 
different datasets). Therefore, it is extremely difficult and time 
consuming to find the proper tuning of the parameters. In this 
respect also, the proposed APSSC has a significant advantage over 
the other semi-supervised counterpart.

The performance of the proposed semi-supervised algorithm is 
tested using a number of real life and synthetic datasets. Statistical 
significance of the experimental results (of different datasets) 
obtained using various semi-supervised methods is evaluated using 
paired t-test. Results of investigation justify the potentiality of the 
proposed APSSC algorithm in terms of classification accuracy, macro 
and micro averaged $F_1$ measures consuming moderate execution 
time. In most of the cases, the improvement in results obtained 
by the proposed method are found to be statistically significant 
compared to its other semi-supervised counterparts.
Future work of the proposed method may be directed toward solving real world problems like microarray gene classification, land-use map generation from multi-spectral remotely sensed images etc.

Conflicts of interest statement

None declared.

Acknowledgments

Support of the Center for Soft Computing Research, Indian Statistical Institute, Kolkata to carry out this research is thankfully acknowledged by Mr. Anindya Halder. The authors are also thankful to the reviewers for their thorough and stimulating comments, which helped to improve the quality of the manuscript.

References

He was a visiting scientist at Remote Sensing Laboratory, University of Trento, Italy during January 2010 to June 2010. He also worked as project linked personnel in Machine Intelligence Unit, ISI, Kolkata from December 2006 to June 2007. He was a Lecturer in Department of Computer Science at Kalyani Mahavidyalaya (College), Kalyani, India from August 2005 to December 2006. He has already published a couple of research articles in internationally reputed journals and refereed conferences. He won (with A. Ghosh and S. Ghosh) the best innovative paper award in the World Congress on Nature and Biologically Inspired Computing (NaBIC’09), Coimbatore, India during December 9-11, 2009. He has been selected for international travel support scheme for young scientist by Department of Science and Technology (DST), Government of India to present a paper in 7th International Conference on Swarm Intelligence, ANTS 2010, Brussels, Belgium, during September 8-10, 2010. His research interests include, machine learning, pattern recognition, swarm intelligence, soft computing, remote sensing image analysis etc.

Susmita Ghosh received the B.Sc. (with honors) in Physics and the B.Tech. in Computer Science and Engineering from the University of Calcutta, India, in 1988 and 1991, respectively, the M.Tech. in Computer Science and Engineering from the Indian Institute of Technology, Bombay, in 1993, and the Ph.D. (Engineering) from Jadavpur University, Kolkata, in 2004. She has visited various universities/academic institutes for collaborative research/ international projects and delivered lectures in different countries including Australia, China, Italy, Japan and the USA. She has published many research papers in internationally reputed journals and refereed conferences and has edited 3 books. She was a Lecturer and then a Reader with the Department of Computer Science and Engineering, Jadavpur University since 1997. She is currently an Associate Professor at the same department. Her research interests include genetic algorithms, neural networks, image processing, pattern recognition, soft computing, and remotely sensed image analysis.

Ashish Ghosh is a Professor with the Machine Intelligence Unit and is In-charge, Center for Soft Computing Research, Indian Statistical Institute. He has already published more than 150 research papers in internationally reputed journals and refereed conferences, and has edited eight books. His current research interests include pattern recognition and machine learning, data mining, image analysis, remotely sensed image analysis, video image analysis, soft computing, fuzzy sets and uncertainty analysis, neural networks, evolutionary computation, and bioinformatics. Dr. Ghosh received the prestigious and most coveted Young Scientists Award in Engineering Sciences from the Indian National Science Academy in 1995, and in Computer Science from the Indian Science Congress Association in 1992. He was selected as an Associate of the Indian Academy of Sciences, Bangalore, India, in 1997. He is a member of the founding team that established the National Center for Soft Computing Research at the Indian Statistical Institute, Kolkata, in 2004, with funding from the Department of Science and Technology, Government of India, and is currently the In-charge of the Center. He is acting as a member of the editorial boards of various international journals.