Semi-supervised associative classification using Ant Colony Optimization algorithm

Hamid Hussain Awan Corresp., 1 , **Waseem Shahzad** 1

1 Department of Computer Science, National Unibersity of Computer and Emerging Sciences Islamabad, Islamabad, Islamabad, Pakistan

Corresponding Author: Hamid Hussain Awan
Email address: hamidawan@gmail.com

Labeled data is the main ingredient for classification task. Obtaining labeled data is not always available and free. Semi-supervised learning solves the problem of labeling the unlabeled instances through some heuristic. Self-training is one of the most widely-used comprehensible approach for labeling the data. Traditional self-training approaches tend to show low classification accuracy when the majority of the data is unlabeled. A novel approach named Self-Training Associative Classification using Ant Colony Optimization (ST-AC-ACO) has been proposed in this paper to label and classify the unlabeled data instances to improve self-training classification accuracy by exploiting the association among attribute values (terms) and between a set of terms and class labels of the labeled instances. Ant Colony Optimization (ACO) has been employed to construct associative classification rules based on labeled and pseudo-labeled instances. Experiments demonstrate the superiority of the proposed associative self-training approach to its competing traditional self-training approaches.

Semi-supervised Associative Classification using Ant Colony Optimization Algorithm

Hanid Hussain Awan[1] **and Waseem Shahzad**[2]

[1]**National University of Computer and Emerging Sciences, H-11/4 Islamabad**
[2]**National University of Computer and Emerging Sciences, H-11/4 Islamabad**

Corresponding author:
Hamid Hussain Awan[1]

Email address: hamidawan@email.com

ABSTRACT

Labeled data is the main ingredient for classification task. Obtaining labeled data is not always available and free. Semi-supervised learning solves the problem of labeling the unlabeled instances through some heuristic. Self-training is one of the most widely-used comprehensible approach for labeling the data. Traditional self-training approaches tend to show low classification accuracy when the majority of the data is unlabeled. A novel approach named Self-Training Associative Classification using Ant Colony Optimization (ST-AC-ACO) has been proposed in this paper to label and classify the unlabeled data instances to improve self-training classification accuracy by exploiting the association among attribute values (terms) and between a set of terms and class labels of the labeled instances. Ant Colony Optimization (ACO) has been employed to construct associative classification rules based on labeled and pseudo-labeled instances. Experiments demonstrate the superiority of the proposed associative self-training approach to its competing traditional self-training approaches.

1 INTRODUCTION

Associative Classification combines frequent-pattern discovery of Association Rule Mining (ARM) Nguyen et al. (2018), Agrawal and Srikant (1994) with classification. The objective of ARM is to discover mutual association of items in itemsets for prediction of inter-dependence of items in given transactions . Frequent patterns are discovered to analyze whether a specific pattern of items is dependent on existence of another pattern Narvekar and Syed (2015). The difference between associative classification and ARM is that in an *associative classification rule* consequent is always a class label Aburub and Hadi (2018).

Semi-supervised Learning (SSL) is and emerging technique to label the instances where majority of the instances in given data is unlabeled Zhu et al. (2013). There are two types of SSL. One is called *Semi-Supervised Classification* in which SSL is used for classification purpose. The other type is called *Semi-Supervised Clustering* or *constrained clustering* which is used to improve clustering performance with the help of labeled instances Li et al. (2019), Triguero et al. (2015). Semi-Supervised classification (SSC) is the subject of this paper.

Self-labeling is one the most widely-used approach to perform SSC Yarowsky (1995), Li and Zhou (2005). It consists of two phases. In the first phase, labeled data is used to train traditional classifiers (e.g. C4.5 Quinlan (1993)) to find a mapping between data distribution and class labels. This knowledge is then used in the second phase to assign labels to unlabeled instances of the data set. There are two slightly different ways of training and assigning labels in semi-supervised learning. One is called the *inductive learning* in which only labeled instances are used during training and unlabeled instances are assigned labels only, without being part of the training. The other approach is called the *transductive learning* in which iterative procedure is followed to label the selected unlabeled instances and then use them as part of the labeled set to label remaining unlabeled instances Zhu et al. (2013). There are two types of self-labeling in literature named *self-training* and *co-training* Ling et al. (2009).

Self-training employs one classification algorithm to construct classification rules using labeled instances. It is retrained on *extended* labeled set of instances (see Definition 3) containing both the labeled and pseudo-labeled instances to refine classification model. Self-training doesn't make any specific assumptions about the underlying dataset except that it assumes its classification model is correct Zhu et al. (2013).

Co-training Fujino et al. (2008) splits the underlying datasets vertically. Each partition is called a view. Each view is used to train a traditional classifier independent of other views Blum and Mitchell (1998). After training of classifier on all views, the classifiers share their model with each other to teach each other about the most confident predictions. Co-training assumes that the underlying dataset can be split into multiple conditionally independent views Jiang et al. (2013).

Ant Colony Optimization (ACO)is a *meta heuristic* inspired by social behavior of ants Parpinelli et al. (2002). ACO does not guarantee optimum solution, but it *attempts* to discover *optimum* or *near-optimum* solution to the given problem. Despite of not providing the guaranteed optimal solution, ACO has been successfully applied in various optimization problems such as Constraint Satisfaction Problem Guan et al. (2021) and data mining problems to show promising results outperforming deterministic greedy algorithms Shahzad and Baig (2011).

The main motivation of the proposed approach is to improve classification accuracy of self-training by replacing the conventional classification algorithms with associative classification method assisted by the ACO meta-heuristic to achieve a diverse and more robust classifier.

This paper proposes a transductive *self-training* Semi-Supervised Classification by exploiting mutual association among attributes-values of underlying data. The proposed approach employs associative classification using ACO for self-training of labeling the unlabeled data and then classification based on the self-training. This technique is named Self-Training-based Associative Classification using Ant Colony Optimization (ST-AC-ACO). The reason for choosing self-training that it doesn't make any assumption about the data distribution. It makes only assumption that its class predictions or pseudo-labeling are correct Witten et al. (2011) Blum and Mitchell (1998). Unlike traditional semi-supervised algorithms, ST-AC-ACO employs associative classification which adds another step of discovering frequent patterns in the labeled instances to construct more robust classifier. The robustness comes from rule construction based on frequent patterns rather than one-step classification Hadi et al. (2018), Venturini et al. (2018). Associative classification as self-training is new to our knowledge and experiments show that it has outperformed existing self-training algorithms (see section 5). The significance of results of classification accuracy is tested using non-parametric Wilcoxon Signed Rank Test García et al. (2010) for each partition to verify the results.

The rest of the paper is as follows: Section 2 presents the preliminary background of SSL and ACO, section 3 presents related work, section 4 explains our proposed technique, section 5 demonstrates experimental results and comparison of the proposed technique with other self-training techniques and section 6 concludes the paper.

2 BACKGROUND

This section presents the basic definitions of terms related to SSL, Associative Classification and ACO.

2.1 Basic terms in SSL

Definition 1. Labeled set L is a subset of dataset D consisting of the data instances which have class labels.

Definition 2. Unlabeled set U is a subset of D consisting of the data instances which don't have class labels.

Mathematically:

$$D = L \cup U \tag{1}$$

Moreover

$$L \cap U = \phi \tag{2}$$

Definition 3. Extended labeled set EL is a sub set of D which is initially L (i.e. $EL = L$). Instances from U are assigned labels and included in the EL. Such instances that are assigned labels by some heuristic are called *pseudo=labeled* instances. When all the instances from U are labeled and added to EL, the EL becomes equal to D Triguero et al. (2014), Zhu et al. (2013), Triguero et al. (2015).

Definition 4. Enlargement of EL is the process of selecting instances from U, assigning them labels and moving them from U to EL. There are three proposed mechanisms for EL enlargement. Triguero et al. (2015). They are:

- **Incremental:** A fixed number of instances are chosen from U to move to EL after assigning most appropriate class to each instance Jiang et al. (2013).
- **Batch:** Each instance is evaluated under additional criteria before being added to EL. The basic criterion is the measure of confidence or similarity of an instance to some labeled instances for assigning the most appropriate class. After each instance is labeled, all pseudo-labeled instances are moved to EL in a single batch.
- **Amend:** Pseudo-labeled instances are continuously monitored and re-evaluated to measure any mis-labeling. Mis-labeled pseudo-labeled instances are re-labeled. This technique is more accurate than others but its much higher time complexity makes it impractical for application Li and Zhou (2005).

2.2 Basic terms of Associative Classification

Definition 5. Pattern is an associative classification rule that states association of an itemset X with a class label Y. The antecedent of a pattern is X while the consequent is Y Agrawal and Srikant (1994), Hadi et al. (2018).

Definition 6. Support of a pattern ($X => Y$) is calculated as:

$$Supp(X => Y) = P(X \cup Y) \tag{3}$$

where $Supp(X => Y)$ denotes the support of pattern $ifXthenY$ while $P(X \cup Y)$ represents the probability of occurrence of itemset X with class label Y Hadi et al. (2018), Nguyen et al. (2018).

Definition 7. Confidence of a pattern ($X => Y$) Venturini et al. (2018), Hadi et al. (2018) is calculated as:

$$Conf(X => Y) = P(Y|X) \tag{4}$$

where $Conf(X => Y)$ denotes confidence of the pattern $X => Y$ while $P(Y|X)$ represents the probability of occurrence of class label Y given the occurrence of itemset X Agrawal and Srikant (1994).

2.3 Ant Colony Optimization (ACO)

A problem can be represented as a 2-dimensional graph data structure in ACO algorithm Parpinelli et al. (2002). The *pheromone* and the *heuristic* are used to calculation of the *selection probability* of a *path* in the graph by an ant. ACO has most of its applications on categorical data sets. Each attribute of a discrete (categorical) dataset contains a finite set of discrete values called *terms*. Terms are represented by *nodes* and selection probabilities of a term being chosen are represented by edges of the graph as shown in figure 1. Terms of the same attribute can't be connected in the graph because only one term of an attribute can be selected in a pattern. For example $T1$ and $T2$ belong to same attribute in the given figure. The node marked with ∞ is the *sink* node which can be selected after selection of at least one term. The search process of an ant is terminated when an ant reaches sink node.

Definition 8. Pheromone in ACO acts for the material deposited by real ants when searching for food. It is used to guide other ants during search of the most optimum paths. Pheromone values can be initialized to zero or some arbitrary value between 0 and 1. A more appropriate way of initializing the pheromone values is given in equation 5 Shahzad and Baig (2011).

$$\tau_{ij} = \frac{1}{\sum_{i \in A} b_i} \tag{5}$$

where τ_{ij} denotes the pheromone value between nodes (terms) i and j, A represents set of attributes while b_i represents number of terms of the ith attribute.

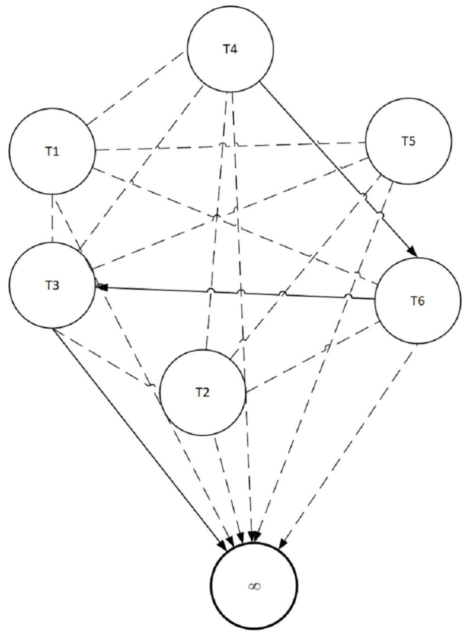

Figure 1. ACO representation as a graph data structure.

Definition 9. Heuristic is a problem-dependent value which usually evaluates the fitness of the solution component. An example heuristic can be the weight of the edge between two nodes. Ant Miner algorithm Parpinelli et al. (2002) uses *entropy* measure used in information theory. Heuristic value is calculated using equations 6 and 7.

$$P_{ij} = P(w|A_i = V_{ij}) \tag{6}$$

$$H(W|A_i = V_{ij}) = -\sum_{w \in C} P_{ij} log(P_{ij}) \tag{7}$$

where H represents heuristic value between nodes (terms) i and j, w represents the class label, C represents set of class labels, A_i represents the i-th attribute, V_{ij} represents j-th value of A_i and $P(w|A_i = V_{ij})$ represents the conditional probability of class label w given that $A_i = V_{ij}$ has occurred.

Definition 10. Selection probability is the guideline for ants to search for most optimal paths. Probability is a combination of pheromone and heuristic values Guan et al. (2021), Mohan and Baskaran (2012)

$$P_{ij} = \frac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{v \in V}[\tau_{iv}]^{\alpha}[\eta_{iv}]^{\beta}} \tag{8}$$

where P_{ij} denotes probability of selecting node j from node i and vice versa, τ_{ij} represents pheromone between nodes i and j, while η_{ij} represents problem-dependent heuristic value. Parameters α and β represent the *weights* of pheromone and heuristic values respectively.

Definition 11. Pheromone of search paths evaporates (decreases) over time. Pheromone evaporation rate ρ is usually kept constant in ACO and is a user-defined parameter. Its value is kept around 0.1 Parpinelli et al. (2002).

Definition 12. The increase in the pheromone values of paths with best results is called pheromone update. This update increases the selection probability of edges in best paths for future iterations by ants Shahzad and Baig (2011).

Definition 13. ACO algorithm is terminated when either a user-defined maximum number of iterations has been executed or the best searched path hasn't been changed for a (user-defined) number of iterations Mohan and Baskaran (2012).

3 RELATED WORK

Shahzad *et al* proposed a robust classifier using associative classification using Ant Colony Optimization for labeled data sets Shahzad and Baig (2011). This model uses the *select class first* approach to construct rules for a selected class only. Rules for all the classes are constructed ny choosing classes one-by-one. This technique experimentally showed much better accuracy than its competitors. This approach has been applicable to supervised classification problems only.

Aburub *et al* developed an associative classification algorithm for prediction of existence of underground water at a given place Aburub and Hadi (2018). Again this algorithm has been developed for associative classification of fully-labeled data.

Associative classification approaches have been applied for labeled datasets only and there exists no work on associative classification for semi-supervised learning of datasets containing unlabeled instances according to our knowledge.

Xiaojin *et al* put forward the initial formalization and classification of Semi-Supervised Learning (SSL) techniques Zhu and Goldberg (2009).

Triguero *et al* presented a taxonomic study of self-leveling techniques in Semi-Supervised Classification. This study provides a critical review of the self-labeling methods and also presents software tools for self-labeling SSC in Triguero et al. (2015). The main contribution of this research work includes proposing of new taxonomy of self-labeling methods, analysis and deduction of transductive and inductive capabilities of the self-labeling methods, and establishing an experimental methodology of the state-of-the-art self-labeling techniques along with the introduction of self-labeling module for KEEL software. The problem with this approach is that it compares self-training and co-training versions of traditional classification algorithms and no additional measure is used in classification process like feature selection or associative classification.

Zhu *et al* applied Semi-Supervised Learning approach for text representing and term classification based on term-weight in Zhu et al. (2013). The experimental results proved the effectiveness of results by the proposed method when compared to the results of supervised classification methods.

More recently, Li *et al* presented an incremental SSL method for classification of streaming data in Li et al. (2019). This approach proposes a model consisting of generative network used to learn representations from input (autoencoders), discriminant structure used to regularize the generative network by building pairwise similarity/dissimilarity (semi-supervised hashing), and the bridge which connects the generative network with the discriminant structure. The proposed approach employs transductive learning and falls in the category of generative methods of semi-supervised learning. They compared their incremental model on evolving streaming data with the state-of-the-art incremental learning approaches like Learn++, AdalinMLP, etc. This approach named ISLSD/ISLSD-E showed to be experimentally more accurate than the supervised incremental learning approaches in competition. Despite its good performance, the proposed approach doesn't provide comprehensible rule-based classifier.

As per our knowledge, there exists no associative classification approach for self-training, self-labeling or even entire semi-supervised classification.

We argue that since associative classification increases the robustness and confidence of classification rules Shahzad and Baig (2011)Hadi et al. (2018)Venturini et al. (2018), it is more logical and a natural way to incorporate associative classification for pseudo-labeling and rule construction in self-trained semi-supervised classification. Thus the main contribution of the proposed approach is the utilization of ACO-based associative classification for self-training and construction of comprehensible rule-based classifier to achieve higher classification accuracy than self-trained versions of classical classification algorithms.

In the proposed approach, an ACO-based *transductive* self-trained semi-supervised associative classification algorithm has been presented. This is a *rule-based* semi-supervised classifier

4 PROPOSED METHODOLOGY

The proposed approach consists of three components, the transductive self-training mechanism of SSL, principles of associative classification and rule construction by ACO.

Algorithm 1 illustrates the proposed ST-AC-ACO algorithm. The sets *L*, *EL* and *U* represent the set of labeled instances, the set of *extended* labeled instances and set of unlabeled instances respectively. The underlying dataset is initially partitioned into training set and test set. The training set is then partitioned into U and L according to a user-specific proportion. The EL consists of both the originally-labeled and pseudo-labeled (labeled by the algorithm) instances.

The $While$ loop (lines 5-19) executes until all the instances in U have been pseudo-labeled and moved to EL. It is important to note that the training and test sets are prepared using uniform class distribution. Similarly, instances from training set are randomly picked from each class according to the uniform class distribution to remove class labels before adding to U. The remaining instances are added to L. The key step is to maintain the specific proportion of labeled instances in L from the training set. Further detail has been explained in section 5.

Pheromone is initialized as illustrated in equation 9:

$$\tau_{ij} = \frac{1}{|Terms|} \tag{9}$$

where $Terms$ is the set of terms in the data set.

The Heuristic function is the second component for probabilistic selection of terms. Equation 10 is used to calculate heuristic value for the selection of the first term.

$$\eta_i = \frac{|term_i, class_k| + 1}{|term_i| + |classes|} \tag{10}$$

where η_i is the *heuristic value* for selection of the *i*th term as the 1st term of the rule antecedent, and $class_k$ represents *k*th class.

After the selection of the first term, heuristic function for the each subsequent term is calculated by equation 11.

$$\eta_{ij} = \frac{|term_i, term_j, class_k| \times |term_j, class_k|}{|term_i, class_k| \times |class_k|} \tag{11}$$

where η_{ij} is the heuristic value for link between the current item $term_i$ and a selection candidate item $term_j$ while $|term_i, term_j, class_k|$ represents the frequency of instances containing itemset $\{item_i, item_j, class_k\}$.

Since there can exist non-associative classification rules consisting of one term, $TRules$ set (line 10) would contain single-term rules constructed by the algorithm 2.

ARules is the list of rules constructed by ants (line 11) returned by the function $ConstructAntRules()$ demonstrated in algorithm 3. *Class_Rules* list is constructed by the union of $TRules$ and *ARules* (line 12). *Class_Rules* are in turn added to the global rule list named *Rules* (line 13).

After the rules for all classes are constructed, the *RulesList* is sorted (line 15) in the descending order of $confidence$ and then *support* (if two rules have equal value of $confidence$).

The process of randomly selection of unlabeled instances from U and assigning them the most suitable labels has been described in lines 16 and 17. The selected instances are called *pseudo-labeled* and are moved from U to EL set (line 18). Number of instances added from U to EL is illustrated in equation 12).

$$n = \begin{cases} |U|, & \text{if } \mu >= |U| \\ r, & \text{otherwise.} \end{cases} \tag{12}$$

where n represents the number of instances to be selected from U, mu is the user-defined parameter which sets the maximum number of instances to be selected in one iteration, U represents the set of unlabeled instances and r is a random number $[1, \mu]$. Moreover, the instances are chosen randomly from U to move to EL. This mechanism provides some level of dynamic extension of the EL as opposed to existing approaches like the approaches proposed in Jiang et al. (2013), Triguero et al. (2015), etc which employ the mechanism of selecting, pseudo-labeling and adding (to the EL set) a fixed static number of instances from U set.

The proposed algorithm uses the *RuleList* to label the selected instances. Terms of each of the selected instance are compared to antecedents the sorted rules. The consequent of the first rule whose antecedent matches an instance is assigned as the label of the instance.

Algorithm 1 SSAC-ACO

```
1: Set $RuleList \leftarrow \phi$
2: Initialize $L, U$
3: Set $EL \leftarrow L$
4: Initialize $minSupp, minConf, NoOfAnts$
5: while $U \neq \phi$ do
6:     Initialize $phermone$
7:     Initialize $heuristic$
8:     for Each class label $c$ do
9:         Set $Class\_Rules \leftarrow \phi$
10:        Set $TRules \leftarrow ConstructTermRules()$
11:        Set $ARules \leftarrow ConstructAntRules()$
12:        Set $Class\_Rules = TRules \cup ARules$
13:        Set $RuleList \leftarrow RuleList \cup Class\_Rules$
14:    end for
15:    Sort $RuleList$ by $confidence$ and $support$(in descending order).
16:    Randomly select $Instances$ from $U$.
17:    Assign class labels to each instance in $Instances$ using $RuleList$.
18:    Set $U \leftarrow U - Instances$ and $EL \leftarrow EL \cup Instances$.
19: end while
20: Prune $RuleList$ and remove duplicate rules (if any).
21: Test $RuleList$ on $TestSet$.
22: Display Results.
```

The constructed rules are then pruned to remove any redundant terms from rules (line 20) and then duplicate rules are removed if there exist any. Finally the *RuleList* is used to calculate the accuracy on *TestSet* and report the results (lines 21-22).

Algorithm 2 ConstructTermRules()

```
1: Set $Rules \leftarrow \phi$
2: for Each $term$ do                                        ▷ Rule for each term
3:     Construct 1-term $rule$ for $term$, such that $(term => c)$.
4:     Calculate $support$ and $confindence$ of $rule$.
5:     if then$support \leq minSupp$
6:         Set $pheromone \leftarrow 0$ for all $term$ trails.
7:     else if $support \geq minSupp$ And $confindence \geq minConf$ then
8:         Set $Rules \leftarrow Rules \cup rule$.
9:     end if
10:    Return $Rules$
11: end for
```

Algorithm 2 illustrates the process of construction of single-term rules. Such rules determine the association of each individual term of the dataset to class labels. Line 3 describes the calculations of support (equation 3) and confidence (equation 4) of the single-term rule. Line 6 is used to set *pheromone* trails to 0 from the *term* of the current rule if *support* is less then a user-defined *minSupport* threshold. If *support* and *confidence* values of the current rule meet the *minSupport* and *minConfidence* thresholds respectively, the *rule* is added to the *Rules* list (line 8).

Algorithm 3 illustrates the construction of associative classification rules by ants. Variable g represents the *generation index* of the ant rules. Each ant constructs an associative classification rule consisting of g number of terms in its antecedents. The initial value of g is set to 2 (line 1). The *while* construct (lines 2-17) present the evolutionary process of the rule construction. The variable *minCoverage* is a user-defined parameter which specifies the proportion of *EL* that has to be covered by the *MultiRules* rule list constructed by ants before termination of the rule construction process and its value is in range $[0,1]$. Lines 5-8 describe how each ant t constructs a rule consisting of at most g terms. The variable

Algorithm 3 ConstructAntRules()

```
1: Set $g \leftarrow 2$.                                          ▷ Generation counter
2: while $g \neq |attributes|$ And $coverage \leq minCoverage$ do
3:     Set $MultiRules \leftarrow \phi$                           ▷ Multi-term rules
4:     Set $t \leftarrow 1$                                       ▷ Ant index
5:     repeat
6:         Let ant $t$ construct a maximum of $g$-term $rule$ such that $(rule => c)$.
7:         Set $t \leftarrow t+1$
8:     until $t > noOfAnts$
9:     for Each $rule$ constructed by ants do
10:        Calculate $support$ and $confidence$ of $rule$
11:        if $support \geq minSupp$ And $\geq minConf$ then
12:            Set $MultiRules \leftarrow MultiRule \cup rule$
13:        end if
14:    end for
15:    Update $pheromone$.
16:    Set $g \leftarrow g+1$
17: end while
18: return $MultiRules$
```

c represents the selected class. For each rule in the ants-constructed rules, if $support$ and $confidence$ meet the threshold values, the rule is added to the $MultiRules$ list (lines 9-14). During construction of a multi-term rule, there are two steps involved. In the first step, an ant has to select first term using equation 10

The second step is to select subsequent terms of a multi-term rule. The pheromone (definition 8) for each possible ant path and heuristic function (definition 9) are the component of the calculation of the selection probability of each subsequent term (definition 10, equation 8). Every subsequent term is probabilistically selected and added to the rule of the current ant t.

The$pheromone$ and consequently the probability matrices are updated after all ants of the g-th generation construct their rules. The pheromone for each path from $term_i$ to $term_j$ is evaporated and is updated using equation 13.

$$\tau_{ij}(g+1) = \tau_{ij}(g) \times (1-\rho) \tag{13}$$

where ρ is a user-defined parameter called pheromone evaporation rate (definition 11).

The pheromone of paths used in construction of rules that were added to the $MutiRules$ list is increased and further updated by using equation 14.

$$\tau_{ij}(g+1) = \tau_{ij}(g) \times (1 - \frac{1}{1+conf_r}) \tag{14}$$

where r represents index of the rule in $Rules$ list.

5 EXPERIMENTAL RESULTS

For the purpose of evaluation of performance of the proposed SSAC-ACO algorithm and comparison with other proposed approaches, we have used 20 SSC datasets from KEEL dataset repository [1].

Table 1 displays the datasets used to evaluate the performance of the proposed approach and other self-training approaches. Most of the chosen datasets either consists of balanced class distribution or a class distribution that was made balanced by merging some of the low-frequency classes to a new class label during pre-processing. The column with heading $|Att|$ represents the number of attributes of datasets, $|Inst|$ represents the number of instances of datasets, $|Class|$ represents the number of classes of data datasets and the last column demonstrates whether a dataset is either balanced or imbalanced with respect to class distribution.

[1] `https://sci2s.ugr.es/keel/semisupervised.php`

Table 1. Datasets used for experiments

Sr. No	DataSet	$\|Att\|$	$\|Ins\|$	$\|Class\|$	Class dist
1	Appendicitis	7	106	2	Imbalanced
2	Australian	14	690	2	Balancesd
3	Automobile	24	159	4	Balancesd (Pre)
4	Breast Cancer	9	286	2	**Imbalanced**
5	Cleveland	13	297	2	Balancesd (Pre)
6	Contraceptive	9	1473	3	Balancesd
7	CRX	15	653	2	Balancesd
8	Flare	11	1066	5	Balancesd (Pre)
9	German	20	1000	2	Balancesd
10	Glass	9	214	3	Balancesd (Pre)
11	Haberman	3	306	2	Imbalanced
12	Heart	13	270	2	Balancesd
13	Iris	4	151	3	Balancesd
14	LED7Ligit	7	550	10	Balancesd
15	Lymphography	18	148	2	Balancesd (Pre)
16	Mammographic	5	830	2	Balancesd
17	Mushroom	22	5644	2	Balancesd
18	Pima	8	768	2	Balancesd
19	Saheart	9	462	2	Balancesd
20	Tae	5	151	3	Balancesd

Table 2 lists parameter values used in training phase of the ST-AC-ACO and competing state-of-the-art self-training classification algorithms. Number of ants, pheromone evaporation rate (ρ) and minimum coverage ($MinCoverage$) have been set as in Shahzad and Baig (2011), while values for minimum support and minimum confidence threshold have been specified by determining the most suitable values by experimentation. Minimum coverage value 1.0 means that training will stop when all instances of the EL have been covered by the list of discovered rules. Parameters for ST-C4.5 and ST-SMOG (SVM) have been set according to the setting in Zhu et al. (2013). Self-Training C4.5 (ST-C4.5) requires two parameters namely confidence level c and minimum number of itemsets per leaf of the decision tree.The algorithm post-prunes the tree. Self-training Sequential Minimal Optimization (ST-SMO) is SVM variant Kumar et al. (2020). Parameter C is set to value 1 to achieve higher training accuracy because the ST-SMO is trained on labeled data to correctly assign labels to unlabeled instances during training. The selected three competitors have been the best performing self-training algorithms in the KEEL tool Zhu et al. (2013). That is why they have been chosen for comparison with the performance of he proposed ST-AC-ACO algorithm.

Table 2. Algorithm parameters used in experiments

Algorithm	Parameter name	Value
ST-AC-ACO	No of ants	30
	Min support	0.05
	Min confidence	0.45
	rho	0.09
	Min coverage	1
ST-C4.5	c	0.25
	i	2
	Pruning	Post-prune
ST-NB	None	N/A
ST-SMO	Kernal type	Polynomial
	Polynomial degree	1
	Fit logistic model	TRUE
	C	1
	Tolerance parameter	0.001
	epsilom	1.00E-12

The proposed ST-AC-ACO algorithm has been implemented in C# while its competitor algorithms used in experimentation have been part of the Semi-Supervised Learning module of the KEEL Alcalá-Fdez et al. (2009) software. A significant difference between implementation of ST-AC-ACO and KEEL implementation is that ST-AC-ACO implementation does not require separate partition files for each partition of datasets. The software is developed to create partition during runtime and to remove labels of the instances of the unlabeled instances before training. Thus the user doesn't have to prepare labeled partitions for datasets. The implementation software for ST-AC-ACO and pre-processed datasets can be found online [2]. We have used 10-cross-fold validation mechanism for evaluation and comparisons. The classification accuracy of ST-AC-ACO has been compared with classification accuracies of ST-C4.5, ST-NB and ST-SMO algorithms.

The experimentation was setup for 4 sets consisting of 10%, 20%, 30% and 40% labeled data. Table 3 to Table 6 demonstrate the comparison of performance the classification accuracy comparison of the above-mentioned algorithms respectively. The Figure 2, Figure 3, Figure 4 and Figure 5 present the visualization of the appropriate tables mentioned above.

[2] `http://www.hamidawan.com.pk/research/`

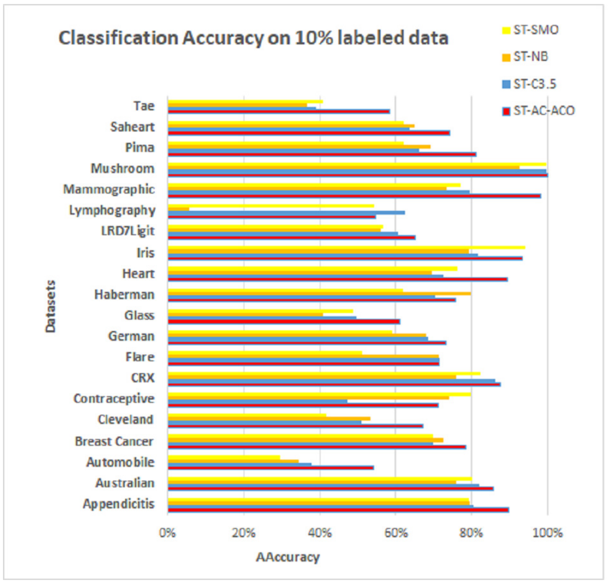

Figure 2. Accuracy comparison over 10% labeled data

Table 3. Classification comparison on 10% labeled data

Datasets	ST-AC-ACO	ST-C3.5	ST-NB	ST-SMO
Appendicitis	89.64%	80.25%	79.45%	79.15%
Australian	85.80%	81.93%	75.83%	80.02%
Automobile	54.25%	37.89%	34.67%	29.52%
Breast Cancer	78.34%	69.66%	72.42%	69.89%
Cleveland	67.03%	51.06%	53.39%	41.84%
Contraceptive	71.21%	47.33%	74.12%	79.88%
CRX	87.58%	86.00%	75.68%	82.26%
Flare	71.49%	71.57%	71.12%	51.24%
German	73.30%	68.68%	67.81%	59.02%
Glass	61.13%	49.66%	40.94%	48.93%
Haberman	75.80%	70.21%	79.69%	61.88%
Heart	89.26%	72.33%	69.59%	76.26%
Iris	93.33%	81.48%	79.26%	94.18%
LRD7Ligit	65.00%	60.74%	56.10%	56.81%
Lymphography	54.73%	62.32%	5.59%	54.22%
Mammographic	98.07%	79.39%	73.30%	77.10%
Mushroom	100.00%	99.55%	92.43%	99.39%
Pima	81.25%	66.10%	69.00%	62.07%
Saheart	74.03%	63.82%	64.78%	62.27%
Tae	58.29%	38.97%	36.61%	40.76%

As obvious from table 3, ST-AC-ACO algorithm comprehensively beat its competing algorithms on

Appendicitis (with 89.64% accuracy as compared to 80.25% accuracy of ST-C4.5 algorithm), *Automobile* (with 54.25% accuracy as compared to 37.89% accuracy of ST-C4.5 algorithm), *Breast cancer* (with 78.34% accuracy as compared to 72.42% accuracy of ST-Naive Bayesian algorithm), *Cleveland* (with 67.03% accuracy as compared to 53.39% accuracy of ST-NB), *Glass* (with 61.13% accuracy as compared to 49.66% accuracy of ST-C4.5), *Heart* (with 89.26% accuracy as compared to 76.26% accuracy of ST-SMO), *Mammographic* (with 98.07% accuracy as compared to 77.10% accuracy of ST-SMO), *Pima* (with 81.25% accuracy as compared to 69.00% accuracy of ST-NB), *Sahrart* (with 74.03% accuracy as compared to 64.78% accuracy of ST-NB) and *Tae* (with 58.29% accuracy as compared to 40.76% accuracy of ST-SMO). With the help of Wilcoxon's signed rank text García et al. (2010), we show that ST-AC-ACO beat non-associative self-training versions of classification algorithms in 17 of 20 datasets with a significant margin on 10% labeled data..

Table 4 presents accuracy comparison of the self-training algorithms on 20% labeled data. ST-C4.5 came closer to ST-AC-ACO over *German* dataset by showing comparable accuracy 69.18% to ST-AC-ACO's 70.20%). ST-NB showed comparable accuracy (89.00%) on *Appendicitis* to that of ST-AC-ACO (87.64%), while it was beaten by ST-AC-ACO on 10% labeled *Appendicitis* dataset. ST-NB beat ST-AC-ACO by showing 81.92% in comparison of ST-AC=ACO's 74.49% on *Haberman* dataset. S-SMO beat ST-AC-ACO on *Contraceptive* dataset by showing 84.05%accuracy against 74.54% of ST-AC-ACO. Wilcoxon's tests show that despite of being behind on a couple of occasions, ST-AC-ACO beat its competitors in accuracy on 15 out of 20 datasets. Figure 3 demonstrates the visual analysis of the results for the results displayed in table 4.

Table 4. Classification comparison on 20% labeled data

Datasets	**ST-AC-ACO**	**ST-C4.5**	**ST-NB**	**ST-SMO**
Appendicitis	87.64%	80.74%	89.00%	72.25%
Australian	97.54%	82.52%	77.02%	81.27%
Automobile	58.63%	45.34%	40.23%	44.26%
Breast Cancer	77.30%	70.22%	71.97%	62.95%
Cleveland	67.69%	53.11%	52.18%	43.72%
Contraceptive	74.54%	47.39%	73.96%	84.06%
CRX	96.94%	85.51%	76.32%	84.57%
Flare	71.11%	72.83%	73.26%	58.65%
German	70.20%	69.18%	68.54%	61.14%
Glass	67.32%	54.28%	42.72%	56.57%
Haberman	74.49%	70.96%	81.92%	65.43%
Heart	97.04%	73.44%	77.54%	77.85%
Iris	96.67%	88.43%	89.44%	91.94%
LED7Ligit	75.20%	67.94%	60.64%	62.72%
Lymphography	91.05%	70.65%	1.23%	66.02%
Mammographic	98.07%	82.43%	76.33%	78.87%
Mushroom	100.00%	99.83%	94.11%	99.77%
Pima	81.76%	68.78%	72.94%	65.56%
Saheart	81.81%	67.16%	67.22%	60.39%
Tae	65.54%	36.82%	41.06%	38.74%

Table 5 displays summary of accuracy comparison of self-training algorithms on 30% labeled data. Figure 4 presents the visual analysis of the same results. ST-AC-ACO didn't improve much its accuracy

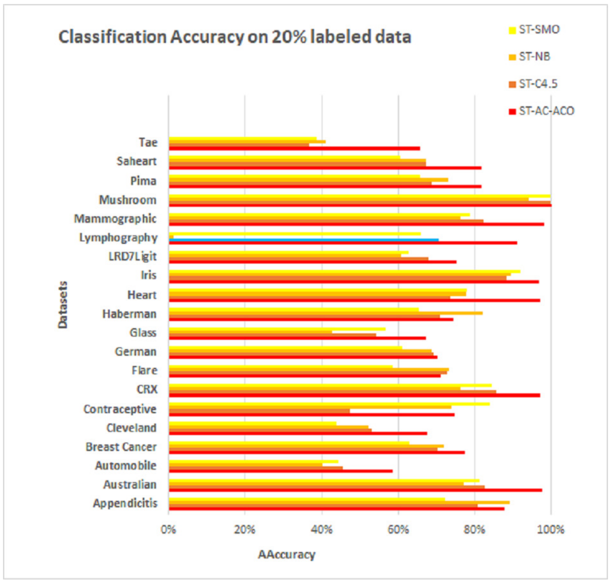

Figure 3. Accuracy comparison of ST-AC-ACO with other Self Training Algorithms over 20% labeled data

from its results on 20% labeled data while comparing to ST-C4.5 algorithm. ST-C4.5 showed comparable results on *Haberman* dataset by showing 70.70.32% accuracy as compared to 73.53% of ST-AC-ACO. ST-AC-ACO lost its lead from ST-NB on *German* dataset as both the algorithms showed almost same accuracy of about 70%. Nevertheless, ST-AC-ACO comprehensively beat ST-NB by showing 73.12% accuracy to 45.06%. While both the algorithms comparable accuracies to each other on 10% and 20% labeled data sets. Moreover, ST-SMO suddenly dropped its lead that it attained against ST-AC-ACO on 20% labeled *Counterceptive* data and showed only 47.89%. This shows the lack of robustness of ST-SMO as compared to probabilistic approaches like ACO and Naive Bayesian self-training approaches. Wilcoxon tests show that ST-C4.5 gave a little tougher competition to ST-AC-ACO despite the proposed approach still managed to show comprehensively higher accuracy on its competitors on 13 out of 20 datasets.

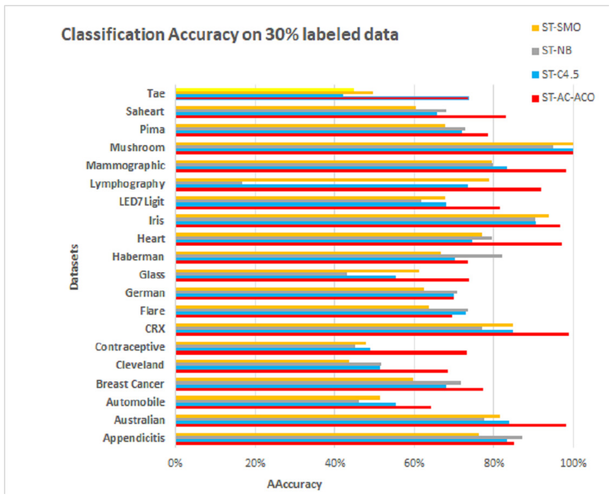

Figure 4. Accuracy comparison of ST-AC-ACO with other Self Training Algorithms over 30% labeled data

Table 5. Classification comparison on 30% labeled data

Datasets	**ST-AC-ACO**	**ST-C4.5**	**ST-NB**	**ST-SMO**
Appendicitis	85.09%	83.38%	87.08%	76.27%
Australian	98.12%	83.82%	77.59%	81.56%
Automobile	64.17%	55.31%	46.15%	51.29%
Breast Cancer	77.25%	67.96%	71.74%	59.56%
Cleveland	68.33%	51.44%	51.50%	43.65%
Contraceptive	73.12%	48.95%	45.06%	47.89%
CRX	98.77%	84.82%	76.93%	84.70%
Flare	69.34%	72.89%	73.56%	63.79%
German	70.00%	69.86%	70.79%	62.37%
Glass	73.85%	55.43%	43.09%	61.05%
Haberman	73.53%	70.32%	82.08%	66.75%
Heart	97.04%	74.44%	79.59%	77.08%
Iris	96.67%	90.63%	90.32%	93.89%
LED7Ligit	81.60%	67.94%	61.59%	67.59%
Lymphography	91.86%	73.37%	16.63%	78.77%
Mammographic	98.07%	83.26%	79.70%	79.45%
Mushroom	100.00%	99.90%	94.89%	99.90%
Pima	78.52%	72.00%	72.78%	67.68%
Saheart	83.12%	65.63%	68.03%	60.31%
Tae	73.54%	42.13%	49.65%	44.73%

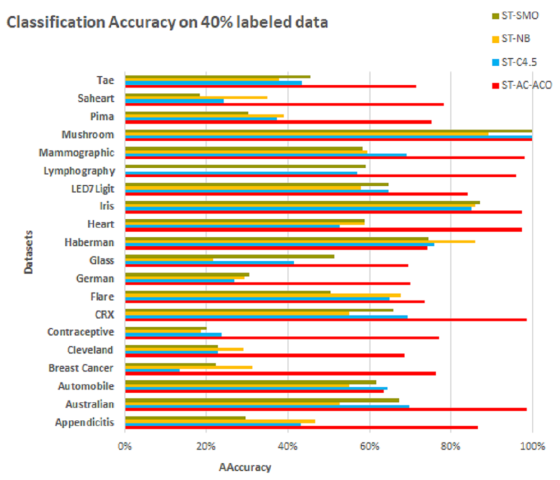

Figure 5. Accuracy comparison of ST-AC-ACO with other Self Training Algorithms over 40% labeled data

Table 6. Classification comparison on 40% labeled data

Datasets	**ST-AC-ACO**	**ST-C4.5**	**ST-NB**	**ST-SMO**
Appendicitis	86.64%	43.09%	46.72%	29.70%
Australian	98.55%	69.70%	52.81%	67.15%
Automobile	63.58%	64.53%	55.17%	61.61%
Breast Cancer	76.26%	13.48%	31.14%	22.40%
Cleveland	68.67%	22.78%	29.07%	22.77%
Contraceptive	76.92%	23.87%	18.62%	19.96%
CRX	98.47%	69.24%	55.03%	65.75%
Flare	73.46%	64.82%	67.69%	50.29%
German	70.10%	26.79%	29.33%	30.53%
Glass	69.57%	41.51%	21.63%	51.27%
Haberman	74.17%	75.90%	85.84%	74.43%
Heart	97.41%	52.74%	58.92%	58.84%
Iris	97.33%	85.00%	86.00%	87.00%
LED7Ligit	84.00%	64.74%	57.77%	64.54%
Lymphography	95.95%	56.90%	0.00%	59.14%
Mammographic	98.07%	69.00%	59.40%	58.31%
Mushroom	100.00%	100.00%	89.19%	99.81%
Pima	75.12%	37.28%	39.05%	30.28%
Saheart	78.12%	24.15%	34.96%	18.50%
Tae	71.50%	43.45%	37.80%	45.57%

Table 6 accompanied by figure 5 presents a comparative analysis of accuracies of self training algorithms on 40% labeled data. The proposed ST-AC-ACO algorithm regained comprehensive lead over ST-C4.5 that it lost on 20% labeled *Appendicitis*, *Flare* and *German* datasets as C4.5 failed to keep comparable accuracy on these datasets by showing 43.09%, 64.82% and 26.79% respectively. Similarly, ST-AC-ACO regained lead over ST-NB on *Appendicitis* (86.64% to 46.72%) and *German*(70.10% to 29.33%) datasets only because it kept its accuracy while ST-NB failed to maintain accuracy on more labeled ratio of labeled data. However, ST-NB kept its significant lead over ST-AC-ACO on *Haberman* dataset by showing 85.15% accuracy to 74.27%. Finally talking about comparison of accuracy comparison of ST-AC-ACO and ST-SMO, ST-AC-ACO gained a significant lead over ST-SMO on *Iris* dataset by showing 97.33% average classification accuracy in comparison to 87.00%.

To validate the results of experiments we performed statistical analysis using Wilcoxon Signed Rank Test García et al. (2010). The reason to use this test instead of other statistical significance tests like pair-wise t-test is that this test is non=parametric and makes no assumption about normal distribution of the underlying data. In our experimentation testing, our null hypothesis (H_0) states that there is no *significant* difference between the medians of accuracies (10-X folds) of ST-AC-ACO and its competitor on a specific dataset. The alternate hypothesis (H_1) states that there is a significant difference between medians of accuracies of ST-AC-ACO and its competitor on a specific dataset. When H_0 is not rejected, the accuracy of ST-AC-ACO is comparable (*Comp*) to its competitor. If average accuracy of ST-AC-ACO is higher than that of its competitor, we conclude that ST-AC-ACO has won (*Win*), otherwise we conclude that ST-AC-ACP has lost (*Loss*). The threshold ($w_{critical}$) is 8 for 10 readings (10-X fold validation). More details of the statistical test can be downloaded from the website [3] .

Table 7 presents the significance analysis of comparison of ST-AC-ACO with ST-C4.5, ST-NB and ST-SMO on 10% labeled data. The bottom three lines describe the summary of wins,defeats and comparable results achieved by ST-AC-ACO against ST-C4.5, ST-NB and ST-SMO respectively. It is important to note that the proposed ST-AC-ACO beat all the competitors on 15 datasets.

[3]`http://www.hamidawan.com.pk/research/`

Table 7. Wilcoxon Signed Rank Test Result on 10% labeled data - ACO vs others ($w_{critical} = 8$)

Dataset	vs ST-C4.5		vs ST-NB		vs ST-SMO	
	W-Stat	**Result**	**W-Stat**	**Result**	**W-Stat**	**Result**
Appendicitis	9	Comp	6	Win	9	Comp
Australian	0	Win	0	Win	0	Win
Automobile	2	Win	3	Win	0	Win
Breast-Cancer	1	Win	3	Win	2	Win
Cleveland	0	Win	0	Win	0	Win
Contraceptive	0	Win	20	Comp	-12	Comp
CRX	0	Win	0	Win	0	Win
Flare	-26.5	Comp	27	Comp	0	Win
German	0	Win	1	Win	0	Win
Glass	0	Win	0	Win	0	Win
Haberman	7	Win	-12	Comp	0	Win
Heart	0	Win	0	Win	0	Win
Iris	0	Win	1	Win	12	Comp
LED7Digit	2	Win	1	Win	1	Win
lymphography	0	Win	0	Win	0	Win
Mammographic	0	Win	0	Win	0	Win
mushroom	0	Win	0	Win	0	Win
Prima	0	Win	0	Win	0	Win
Sahara.	1	Win	1	Win	1	Win
Tae	0	Win	0	Win	1	Win
Win	**18**		**17**		**17**	
Loss	**0**		**0**		**0**	
Comp	**2**		**3**		**3**	

Table 8. Wilcoxon Signed Rank Test Result on 20% labeled data - ACO vs others ($w_{critical} = 8$)

Dataset	vs ST-C4.5		vs ST-NB		vs ST-SMO	
	W-Stat	**Result**	**W-Stat**	**Result**	**W-Stat**	**Result**
Appendicitis	11	Comp	-25	Comp	3	Win
Australian	0	Win	0	Win	0	Win
Automobile	7	Win	3	Win	8	Comp
Breast-Cancer	3	Win	4	Win	1	Win
Cleveland	0	Win	0	Win	0	Win
Contraceptive	0	Win	24	Comp	0	Loss
CRX	0	Win	0	Win	0	Win
Flare	-21	Comp	-18	Comp	0	Win
German	11	Comp	0	Win	0	Win
Glass	1	Win	0	Win	7	Win
Haberman	4	Win	-2	Loss	0	Win
heart	0	Win	0	Win	0	Win
Iris	8	Comp	3	Win	10	Comp
LED7Digit	11	Comp	4	Win	4	Win
lymphography	0	Win	0	Win	0	Win
Mammographic	0	Win	0	Win	0	Win
mushroom	0	Win	0	Win	0	Win
Pima	2	Win	3	Win	0	Win
Saheart.	0	Win	0	Win	0	Win
Tae	0	Win	2	Win	1	Win
Win	**15**		**16**		**17**	
Loss	**0**		**1**		**1**	
Comp	**5**		**3**		**2**	

Table 8 demonstrates the significance analysis of comparison on 20% labeled data. ST-AC-ACO lost only on 1 dataset each to ST-NB and ST-SMO. It is important to note that the proposed ST-AC-ACO beat all the competitors on 12 datasets.

Table 9. Wilcoxon Signed Rank Test Result on 30% labeled data - ACO vs others ($w_{critical} = 8$)

Dataset	vs ST-C4.5		vs ST-NB		vs ST-SMO	
	W-Stat	**Result**	**W-Stat**	**Result**	**W-Stat**	**Result**
Appendicitis	15.5	Comp	24	Comp	6	Win
Australian	0	Win	0	Win	0	Win
Automobile	13	Comp	8	Comp	16	Comp
Breast-Cancer	1	Win	2	Win	1	Win
Cleveland	0	Win	0	Win	0	Win
Contraceptive	0	Win	0	Win	0	Win
CRX	0	Win	0	Win	0	Win
Flare	-20.5	Comp	-17	Comp	1	Win
German	23	Comp	-12	Comp	0	Win
Glass	1	Win	0	Win	16	Comp
Haberman	8	Comp	-2	Loss	0	Win
heart	0	Win	0	Win	0	Win
Iris	11	Comp	2	Win	10	Comp
LED7Digit	10	Comp	5	Win	10	Comp
lymphography	0	Win	0	Win	3	Win
Mammographic	0	Win	0	Win	0	Win
mushroom	0	Win	0	Win	0	Win
Pima	2	Win	3	Win	0	Win
Saheart.	0	Win	0	Win	0	Win
Tae	1	Win	8	Comp	1	Win
Win	**13**		**14**		**16**	
Loss	**0**		**1**		**0**	
Comp	**7**		**5**		**4**	

Table 9 demonstrates the significance analysis of comparison on 30% labeled data. According to the results, ST-C4.5 came closer to ST-AC-ACO by showing comparable accuracy on 5 datasets. ST-AC-ACO won from all competitors on 11 datasets.

Table 10. Wilcoxon Signed Rank Test Result on 40% labeled data - ACO vs others ($w_{critical} = 8$)

Dataset	vs ST-C4.5		vs ST-NB		vs ST-SMO	
	W-Stat	**Result**	**W-Stat**	**Result**	**W-Stat**	**Result**
Appendicitis	0	Win	1	Win	0	Win
Australian	0	Win	0	Win	0	Win
automobile	-22	Comp	24	Comp	-23	Comp
Breast-Cancer	0	Win	0	Win	0	Win
Cleveland	0	Win	0	Win	0	Win
Contraceptive	0	Win	0	Win	0	Win
CRX	0	Win	0	Win	0	Win
Flare	7	Win	13	Comp	0	Win
German	0	Win	0	Win	0	Win
Glass	0	Win	0	Win	8	Comp
Haberman	-22.5	Comp	0	Loss	27	Comp
Heart	0	Win	0	Win	1	Win
Iris	4	Win	1	Win	5	Win
LED7Digit	11	Comp	6	Win	6	Win
Lymphography	0	Win	0	Win	0	Win
Mammographic	0	Win	0	Win	0	Win
Mushroom	0	Comp	0	Win	0	Win
Pima	0	Win	0	Win	0	Win
Saheart.	0	Win	0	Win	0	Win
Tae	2	Win	2	Win	5	Win
Win	**16**		**17**		**17**	
Loss	**0**		**1**		**0**	
Comp	**4**		**2**		**3**	

Table 10 demonstrates the significance analysis of comparison on 40% labeled data. As it is quite clear that ST-AC-ACO comprehensively beat other self-training algorithms on majority of datasets. ST-AC-ACO beat all competitors on 14 datasets.

As it has been shown that the power of *associative property* makes associative classification mechanism much more robust and reliable than other non-associative classifiers in semi-supervised classification problem.. Due to discovery of implicit relationship among non-class attributes to determine frequent patterns allows classification be more accurate and robust than merely constructing classification rules without finding association among non-class attributes.

6 CONCLUSION

A novel rule-based semi-supervised associative classification approach using ant colony optimization has been proposed in this paper. The primary task of the approach is to learn from a very smaller ratio of labeled data than unlabeled data to first label the unlabeled data and then apply the classification rules. This approach uses labeled data to first discover associative classification rules with ACO and then using those rules in transductive mechanism to label the unlabeled instances. The experimental results of the proposed technique demonstrate that the proposed ST-AC-ACO algorithm is not only superior in accuracy to its competing self-training algorithms but it is more robust as it tends to discover relationship between a frequent itemset of non-lass attributes and the class labels. This approach can further be combined with feature subset selection to remove unnecessary or redundant attributes for even better classification accuracy. Moreover, the proposed approach can also be utilized for labeling and classification of big data with a little fraction of labeled data. Another future direction is to develop a mechanism to find frequent patterns from the entire)labeled and unlabeled) dataset and assign the most confident class. A more fundamental task in this regard is to re-define the SSL problem by omitting the classification and presenting results just on pseudo-labeling, after all classification is a secondary task and its performance directly depends on pseudo-labeling of unlabeled instances.

REFERENCES

Aburub, F. and Hadi, W. (2018). A new associative classification algorithm for predicting groundwater locations. *JIKM*, 17(4):1850043.

Agrawal, R. and Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB ’94, pages 487–499, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Alcalá-Fdez, J., Sánchez, L., García, S., del Jesus, M. J., Ventura, S., i Guiu, J. M. G., Otero, J., Romero, C., Bacardit, J., Rivas, V. M., Fernández, J. C., and Herrera, F. (2009). KEEL: a software tool to assess evolutionary algorithms for data mining problems. *Soft Comput.*, 13(3):307–318.

Blum, A. and Mitchell, T. M. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory, COLT 1998, Madison, Wisconsin, USA, July 24-26, 1998.*, pages 92–100.

Fujino, A., Ueda, N., and Saito, K. (2008). Generative/discriminative classifier based on the maximum entropy principle. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(3):424–437.

García, S., Fernández, A., Luengo, J., and Herrera, F. (2010). Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Inf. Sci.*, 180(10):2044–2064.

Guan, B., Zhao, Y., and Li, Y. (2021). An improved ant colony optimization with an automatic updating mechanism for constraint satisfaction problems. *Expert Syst. Appl.*, 164:114021.

Hadi, W., Al-Radaideh, Q. A., and Alhawari, S. (2018). Integrating associative rule-based classification with naïve bayes for text classification. *Appl. Soft Comput.*, 69:344–356.

Jiang, Z., Zhang, S., and Zeng, J. (2013). A hybrid generative/discriminative method for semi-supervised classification. *Knowl. Based Syst.*, 37:137–145.

Kumar, B., Sinha, A., Chakrabarti, S., Khandelwal, A., Jain, H., and Vyas, O. P. (2020). Sequential minimal optimization for one-class slab support vector machine. *CoRR*, abs/2011.03243.

Li, M. and Zhou, Z.-H. (2005). Setred: Self-training with editing. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 611–621. Springer.

Li, Y., Wang, Y., Liu, Q., Bi, C., Jiang, X., and Sun, S. (2019). Incremental semi-supervised learning on streaming data. *Pattern Recognition*, 88:383–396.

Ling, C. X., Du, J., and Zhou, Z. (2009). When does co-training work in real data? In *Advances in Knowledge Discovery and Data Mining, 13th Pacific-Asia Conference, PAKDD 2009, Bangkok, Thailand, April 27-30, 2009, Proceedings*, pages 596–603.

Mohan, B. C. and Baskaran, R. (2012). A survey: Ant colony optimization based recent research and implementation on several engineering domain. *Expert Syst. Appl.*, 39(4):4618–4627.

Narvekar, M. and Syed, S. F. (2015). An optimized algorithm for association rule mining using fp tree. *Procedia Computer Science*, 45:101 – 110. International Conference on Advanced Computing Technologies and Applications (ICACTA).

Nguyen, L. T. T., Vo, B., Nguyen, L. T. T., Fournier-Viger, P., and Selamat, A. (2018). ETARM: an efficient top-k association rule mining algorithm. *Appl. Intell.*, 48(5):1148–1160.

Parpinelli, R. S., Lopes, H. S., and Freitas, A. A. (2002). Data mining with an ant colony optimization algorithm. *IEEE Trans. Evolutionary Computation*, 6(4):321–332.

Quinlan, J. R. (1993). *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Shahzad, W. and Baig, A. R. (2011). Hybrid associative classification algorithm using ant colony optimization. *International journal of innovative computingv and information control: IJICIC*, 7:6815–6826.

Triguero, I., García, S., and Herrera, F. (2015). Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study. *Knowl. Inf. Syst.*, 42(2):245–284.

Triguero, I., Sáez, J. A., Luengo, J., García, S., and Herrera, F. (2014). On the characterization of noise filters for self-training semi-supervised in nearest neighbor classification. *Neurocomputing*, 132:30–41.

Venturini, L., Baralis, E., and Garza, P. (2018). Scaling associative classification for very large datasets. *CoRR*, abs/1805.03887.

Witten, I. H., Frank, E., and Hall, M. A. (2011). *Data mining: practical machine learning tools and techniques, 3rd Edition*. Morgan Kaufmann, Elsevier.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *33rd annual meeting of the association for computational linguistics*, pages 189–196.

Zhu, X. and Goldberg, A. B. (2009). *Introduction to Semi-Supervised Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.

Zhu, Y., Yu, J., and Jing, L. (2013). A novel semi-supervised learning framework with simultaneous text representing. *Knowl. Inf. Syst.*, 34(3):547–562.