IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 21 July 2023, accepted 14 August 2023, date of publication 21 August 2023, date of current version 24 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3306783

RESEARCH ARTICLE

An Improved AdaBoost Algorithm for Highly Imbalanced Datasets in the Co-Authorship Recommendation Problem

VO DUC QUANG[1,2], HOANG HUU VIET[1], VU HOANG LONG[2], AND TRAN DINH KHANG[2]

[1]Faculty of Information Technology, Vinh University, Vinh 43108, Vietnam

[2]School of Information and Communication Technology, Hanoi University of Science and Technology, Hanoi 113000, Vietnam

Corresponding author: Tran Dinh Khang (khangtd@soict.hust.edu.vn)

ABSTRACT The co-authorship recommendation problem is attractive since it helps researchers extend collaboration to improve the quality of scientific articles as well as promote innovation. This problem involves suggesting authors join research groups based on their research interests, areas of expertise, and past collaborative experiences to write scientific articles together. In this paper, we tackle the co-authorship recommendation problem by modeling it as a co-authorship network, where each author is represented as a vertex, and each collaboration between two authors is represented as an edge. Since the number of author pairs without collaboration is much larger than those with collaboration, datasets created from co-authorship networks are typically two-class imbalanced datasets. Accordingly, we propose an improved algorithm of AdaBoost combined with the W-SVM algorithm, called Im.AdaBoost.W-SVM, to solve the classification problem with two-class imbalanced datasets. To evaluate the performance of our Im.AdaBoost.W-SVM algorithm for the co-authorship recommendation problem, we collected author and article data from the website *www.sciencedirect.com* through ScienceDirect APIs and created two-class imbalanced datasets. Our experimental results for our self-built co-authorship datasets with different sizes and imbalance ratios showed that our Im.AdaBoost.W-SVM algorithm outperforms the AdaBoost.DecisionTree and AdaBoost.W-SVM algorithms for the co-authorship recommendation problem.

INDEX TERMS AdaBoost, co-authorship network, imbalanced dataset, recommendation problem, support vector machine.

I. INTRODUCTION

The co-authorship recommendation problem is to predict the ability of co-authors to collaborate in the future [1]. In publishing scientific research works, the collaboration of authors is demonstrated by their joint authorship in articles. The relationship between authors and articles is a many-to-many relationship since an author can participate in writing many articles, and an article can have one or many authors, thereby creating an academic network called a co-authorship network [2], [3], [4]. In other words, a co-authorship network is a specialized social network that represents the collaborative relationship between scientists in publishing scientific research works, where each author is represented as a vertex, and each collaboration between two authors is represented as an edge.

The co-authorship recommendation problem is attractive and has been received significant attention from researchers since it helps researchers to share knowledge and foster innovation. So far, the main approaches to solving the co-authorship recommendation problem have been proposed, such as network analysis [5], [6], [7], [8], machine learning [9], [10], [11], and similar-content analysis [12], [13]. Among these approaches, machine learning has emerged as a popular and effective approach for solving the co-authorship recommendation problem in co-authorship networks. This is because, from the current co-authorship network, machine learning algorithms can learn to identify the potential author pairs that are likely to collaborate in the future, which is referred to as co-authorship candidate pairs. In the machine

The associate editor coordinating the review of this manuscript and approving it for publication was Wentao Fan.

learning approach, we have to compute various attributes for every pair of vertices in a co-authorship network, which may be correlated with the likelihood of a future collaboration between the authors. Once computed, the set of attribute values for a pair of vertices is referred to as a feature vector. Then, we need to label the feature vectors by checking whether or not two authors have previously collaborated on some article. If the author pair is co-authorships in an article, then the feature vector is labeled as $+1$, else it is labeled as -1. Finally, we apply classification algorithms to the dataset of labeled feature vectors to predict co-authorship collaborations.

In practice, co-authorship networks often exhibit a highly imbalanced distribution of connections. This is because authors typically collaborate with only a small subset of all possible authors, and the number of potential collaborations is much larger than the actual collaborations in the network. Consequently, datasets created from co-authorship networks are typically two-class imbalanced datasets, where the class of minority samples is labeled as $+1$, and the class of majority samples is labeled as -1. Hereafter, we refer to the samples labeled $+1$ as positive samples and those labeled -1 as negative samples.

Learning on imbalanced datasets is one of the challenging problems in the machine learning field. Typically, classification algorithms attempt to achieve classification models with the highest possible accuracy rate. However, such classification models will be biased toward recognizing the majority class label for two-class imbalanced datasets. In a co-authoring network, the training dataset often exhibits a high imbalance between positive and negative samples. As a result, classification models trained on such a dataset tend to misclassify positive samples as negative samples. This means that the classification models give a very high accuracy but fail to correctly classify almost all positive samples. According to [14], approaches commonly used to improve classification on imbalanced datasets are:

- Using preprocessing techniques on imbalanced datasets such as: (*i*) reducing the number of negative samples; (*ii*) generating additional positive samples; or (*iii*) combining both. These techniques aim to reduce the imbalance of datasets to improve the performance of traditional machine learning algorithms [15], [16], [17].
- Using techniques to improve algorithms such as: (*i*) error weight assignment; (*ii*) cost-based learning [18], [19], [20], [21], [22], [23], [24]; or (*iii*) applying deep learning models to imbalanced datasets [25], [26], [27], [28]. These techniques aim to modify traditional classification algorithms to classify positive samples better.

Among the above approaches, cost-based learning algorithms assign a higher cost weight when the training models misclassify positive samples as negative ones. Consequently, these algorithms offer several advantages such as: (*i*) keeping the original characteristics of datasets; (*ii*) providing various methods to improve the training parameters; and (*iii*) minimizing the error cost function through loops and parameter tuning. Although using a single classification algorithm can be effective in some cases, it may not take into account all the characteristics of datasets. Hence, several researchers have proposed combining multiple classifiers to create a more accurate and robust aggregate classifier [29], [30].

The AdaBoost algorithm proposed by Freund [31] is an ensemble learning method that combines several weak classifiers to form a strong classifier. Recently, AdaBoost has been improved by many researchers, notably the studies of combining AdaBoost with Support Vector Machine (SVM) [32], [33], [34], [35], [36], [37], [38], [39]. These improvements aim to take advantage of AdaBoost's adaptive iterability and SVM's scalability on datasets with different characteristics. In [33], Wonji et al. proposed a method that combines AdaBoost with Weight SVM (W-SVM), namely AdaBoost.W-SVM, to improve classification efficiency on two-class imbalanced datasets. However, we recognize that AdaBoost.W-SVM [33] and other studies [32], [35], [36], [37], [38], [39] using AdaBoost on two-class imbalanced datasets initialize equal error weights for each data sample. Thus, such algorithms can be improved to make them more efficient for the classification problem of two-class imbalanced datasets that require prioritizing the accurate classification of positive samples. In addition, AdaBoost computes the confidence weight of the membership classifiers by considering the total error in the entire dataset without considering the specific characteristics of individual samples in a given dataset, while the attributes of each sample are so important in enhancing the accurate classification of positive samples. Therefore, in this study, we propose an improved algorithm of AdaBoost combined with W-SVM algorithm, so called Im.AdaBoost.W-SVM, for the co-authorship recommendation problem. Firstly, we propose two significant improvements to the original AdaBoost: (*i*) initializing the set of different error weights adapted to the imbalance rate between positive and negative samples of datasets; and (*ii*) calculating the confidence weights of member classifiers based on their sensitivity to the total error caused by positive samples, *i.e.*, if the member classifier misclassified more positive samples, the confidence weights of misclassified samples will be decreased. Secondly, we apply the W-SVM algorithm as a membership classifier to our Im.AdaBoost.W-SVM to enhance the correct classification rate of positive samples. To evaluate the performance of our Im.AdaBoost.W-SVM algorithm for the co-authorship recommendation problem, we collected information on scientific articles from the website *www.sciencedirect.com* through ScienceDirect APIs and built two-class imbalanced datasets. Our experimental results on various experimental scenarios show that our Im.AdaBoost.W-SVM is more efficient than AdaBoost.DecisionTree [29] (*i.e.*, AdaBoost combined with Decision Tree algorithm) and AdaBoost.W-SVM algorithms [33] for the co-authorship recommendation problem.

The remainder of this paper is structured as follows: Section II provides preliminaries about the co-authorship recommendation problem, AdaBoost and Weighted-SVM algorithms; Section III presents our Im.AdaBoost.W-SVM; Section IV describes our experimental results and discussions; Section V concludes our work and offers directions for future research.

II. PRELIMINARIES

A. CO-AUTHORSHIP RECOMMENDATION PROBLEM

We approach the co-authorship recommendation problem by modeling it as a co-authorship network. A co-authorship network is represented by $G^T = (V^T, E^T, P^T, T)$, where (i) $T = \{t_1, t_2, \ldots, t_k\}$ is a set of timestamps; (ii) $V^T = \{v_1, v_2, \ldots, v_N\}$ is the set of vertices representing the authors appearing in the articles at some time in T; (iii) $P^T = \{p_1, p_2, \ldots, p_M\}$ is the set of articles at some time in T; and (iv) $E^T = \{v_i, v_j, p_k, t_h\}$ is the set of links between authors at some time in T, where two authors $(v_i, v_j) \in V^T \times V^T$ have the same article $p_k \in P^T$ at the time $t_h \in T$. In addition, the set V^T can add attributes that carry specific information about each author such as her/his information about nationality, affiliation, and study topics. The set of attributes is denoted by $A^T = \{a_1, a_2, \ldots, a_N\}$, where a_i is a feature vector containing information about an author/vertex pair $(v_i, v_j) \in V^T \times V^T$. Given a co-authorship network G^T, the co-authorship recommendation problem aims to predict the ability of authors to collaborate in the future. In other words, we have to predict the potential links among authors based on the information given in the sets E^T and A^T. Given two connected vertices $v_i \in V^T$ and $v_j \in V^T$ in G^T, the link measures are classified into three main groups:

a) **Link measures based on neighbors**: Given a vertex $v_i \in V^T$, we let $\Gamma(v_i)$ be the set of neighbors of v_i. Then, some popular link measures are Common Neighbor (*CN*), Adamic Adar (*AA*), Jaccard Coefficient (*JC*), Preferential Attachment (*PA*), and Resource Allocation (*RA*) [40]:

1) *CN* is calculated by the number of common neighbors of v_i and v_j:
$$CN(v_i, v_j) = |\Gamma(v_i) \cap \Gamma(v_j)|. \tag{1}$$
This show that the greater the number of common neighbors is, the greater the *CN* value is, *i.e.,* the higher the probability that the pair of authors (v_i, v_j) will collaborate in the future.
2) *AA* is calculated by the number of neighbor vertices of each common neighbor:
$$AA(v_i, v_j) = \sum_{v_k \in \Gamma(v_i) \cap \Gamma(v_j)} \frac{1}{log(|\Gamma(v_k)|)}, \tag{2}$$
where v_k is a common neighbor of both v_i and v_j.
3) *JC* is calculated by the ratio of the number of common neighbors to the total number of neighbors of v_i and v_j:
$$JC(v_i, v_j) = \frac{\Gamma(v_i) \cap \Gamma(v_j)}{\Gamma(v_i) \cup \Gamma(v_j)}. \tag{3}$$
4) *PA* is calculated by the number of neighbors of each vertice v_i and v_j:
$$PA(v_i, v_j) = |v_i| \times |v_j|, \tag{4}$$
where $|v_i|$ and $|v_j|$ are the number of degrees of v_i and v_j, respectively.
5) *RA* is similar to *AA* and it is calculated as follows:
$$RA(v_i, v_j) = \sum_{v_k \in \Gamma(v_i) \cap \Gamma(v_j)} \frac{1}{|\Gamma(v_k)|}. \tag{5}$$

b) **Link measures based on paths**: Some popular link measures are Shortest Path (*SPath*) and *Katz* [40]:

1) The *SPath* metric considers the path distance between two vertices v_i and v_j, denoted by $d(v_i, v_j)$, through intermediate vertices:
$$SPath(v_i, v_j) = \frac{1}{d(v_i, v_j)}. \tag{6}$$
If there is no path between two vertices v_i and v_j, then $SPath(v_i, v_j) = 0$.
2) *Katz* is the number of possible paths between two vertices v_i and v_j:
$$Katz(v_i, v_j) = \sum_{l=1}^{\infty} \beta^l |path^l_{v_i,v_j}| = \beta A + \beta A^2 + \cdots, \tag{7}$$
where $A^l = \{path^l_{v_i,v_j} | (v_i, v_j) \in V^T \times V^T\}$ is the set of paths from v_i to v_j with length l ($l \geq 1$) and β is an optional constant. $Katz(v_i, v_j)$ means that the farther the path is, the less influence it has on the measurement.

c) **Link measures based on additional personal information**: Some popular link measures are Similar Work (*SW*) and Common Country (*CC*) [41]:

1) *SW* is the similarity of the workplace and nationality among vertices $v_1, v_2, \cdots, v_N$:
$$SW(v_1, \cdots, v_N) = \begin{cases} 2, \text{if } S_1(v_1) = \cdots = S_1(v_N), \\ 1, \text{if } S_2(v_1) = \cdots = S_2(v_N), \\ 0, \text{otherwise}, \end{cases} \tag{8}$$
where $S_1(v_i)$ and $S_2(v_i)$ are information about the workplace and nationality of v_i ($i = 1, 2, \cdots, N$).
2) *CC* is the similarity of the workplace between two authors in the same national or university:
$$CC(v_i, v_j) = SW(v_i, v_j) + \sum_{v_k \in \Gamma(v_i) \cap \Gamma(v_j)} SW(v_k, v_i, v_j). \tag{9}$$

The link measures are used for determining the collaborative levels of authors in a co-authorship network. Considering a set of timestamps in T_1, the co-authorship candidate table built on G^{T_1} has the following structure: each row is the information about a pair (v_i, v_j) of authors and the columns

TABLE 1. Table of co-authorship candidates.

Index	Candidate pairs (v_i, v_j)	Linking measures $x = \{CN(v_i,v_j), AA(v_i,v_j), JC(v_i,v_j), PA(v_i,v_j), RA(v_i,v_j), SPath(v_i,v_j), Kazt(v_i,v_j), CC(v_i,v_j)\}$	Labels $y \in \{+1, -1\}$
1	(v_1, v_2)	$\{CN(v_1,v_2), AA(v_1,v_2), JC(v_1,v_2), PA(v_1,v_2), RA(v_1,v_2), SPath(v_1,v_2), Kazt(v_1,v_2), CC(v_1,v_2)\}$	$+1$
2	(v_1, v_3)	$\{CN(v_1,v_3), AA(v_1,v_3), JC(v_1,v_3), PA(v_1,v_3), RA(v_1,v_3), SPath(v_1,v_3), Kazt(v_1,v_3), CC(v_1,v_3)\}$	-1
3	(v_1, v_4)	$\{CN(v_1,v_4), AA(v_1,v_4), JC(v_1,v_4), PA(v_1,v_4), RA(v_1,v_4), SPath(v_1,v_4), Kazt(v_1,v_4), CC(v_1,v_4)\}$	-1
4	(v_2, v_3)	$\{CN(v_2,v_3), AA(v_2,v_3), JC(v_2,v_3), PA(v_2,v_3), RA(v_2,v_3), SPath(v_2,v_3), Kazt(v_2,v_3), CC(v_2,v_3)\}$	$+1$
...	...	...	...

are candidate pairs, link measures, and labels of the candidate pair of authors. Assuming that T_2 is a set of timestamps after T_1, the co-authorship candidate table built on a co-authorship network G^{T_2} is considered to assign labels with true collaboration (*i.e.*, labeled $+1$) or no collaboration (*i.e.*, labeled -1) for data samples of candidate pairs, as illustrated in Table 1.

Visually, we see that the candidate data table can be considered as a set of co-authored data samples with the full attributes and class labels. Therefore, the co-authorship recommendation problem is transformed into a classification problem on a two-class labeled dataset, where one class is labeled as $+1$ to represent a future collaboration, and the other is labeled as -1 to represent no future collaboration. Furthermore, in a large co-authorship network, the candidate pairs can exponentially explode, resulting in many cases where collaboration between authors does not exist, corresponding to the label -1. This leads to the creation of a two-class imbalanced dataset. Accordingly, with the classification model approach, the co-authorship recommendation problem becomes a classification problem on two-class imbalanced datasets.

B. AdaBoost ALGORITHM

In the classification problem for a two-class imbalanced dataset, using a single algorithm may not fully consider the characteristics of datasets. Therefore, many studies combine multiple classification algorithms, called membership classifiers, to form a stronger classifier [29], [30]. The AdaBoost algorithm proposed by Freund [31] is such a strong classifier. Give a dataset, the main idea of AdaBoost is that in each iteration, it assigns each data sample in the dataset to an error weight and re-evaluates the classification results of the membership classifiers, thereby allowing it to obtain better values of parameters for the next iteration.

Specifically, AdaBoost is presented in Alg. 1. The inputs of the algorithm include: (*i*) X is a dataset of N samples (x_i, y_i) $(i = 1, 2, \cdots, N)$, where x_i is an attribute vector and $y_i \in \{-1, +1\}$ is a class label of x_i; (*ii*) M is the maximum number of iterations; and (*iii*) h_1 is a membership classifier. At the beginning, the algorithm assigns a set of equal error weights $D^1 = \{w_i^1 = \frac{1}{N}\}$ for each sample $x_i \in X$. At each iteration t $(t = 1, 2, \cdots, M)$, the classifier h_t classifies the dataset X (line 3). The classification quality of h_t is evaluated through the sum of error ε_t for the next iteration (line 4) and the confidence weight α_t (line 5). Then, the algorithm updates the error weight distribution ω_i^{t+1} (line 6). Finally, an aggregate classification model is calculated according to the formula $H(x)$ (line 7). The classification label of the sample is determined based on the sign function: label $+1$ when $H(x) > 0$ and label -1 when $H(x) < 0$. If the total error ε_t on the dataset is equal to 0.5, then $\alpha_t = 0$, meaning that the classifier h_t does not contribute to the classification decision of the ensemble classifier H.

Recently, Lee et al. proposed an AdaBoost.W-SVM algorithm [33], where W-SVM [42] is a membership classifier in AdaBoost algorithm. At each iteration t, AdaBoost.W-SVM uses parameters z_i^t to adjust the weights of samples $x_i \in X$ in W-SVM. The value of z_i^t is calculated based on the number of samples $x_i \in X$ distributed in the SVM marginal space as follows:

1) If the sample x_i is in the bounded support vectors (BSV), then

$$z_i^t = \begin{cases} \dfrac{N_{BSV}}{2N_{BSV_-}}, & \text{if } y_i = -1, \\ \dfrac{N_{BSV}}{2N_{BSV_+}}, & \text{if } y_i = +1, \end{cases} \tag{10}$$

where N_{BSV} is the total number of samples in BSV, N_{BSV_-} is the total number of negative samples in BSV, and N_{BSV_+} is the total number of positive samples in BSV.

2) If the sample x_i is on the margin boundaries of support vectors (SV), then

$$z_i^t = \begin{cases} \dfrac{N_{SV}}{2N_{SV_-}}, & \text{if } y_i = -1, \\ \dfrac{N_{SV}}{2N_{SV_+}}, & \text{if } y_i = +1, \end{cases} \tag{11}$$

where N_{SV} is the total number of samples on SV, N_{SV_-} is the total number of negative samples on SV, and N_{SV_+} is the total number of positive samples on SV.

3) If the sample x_i is noise, then

$$z_i^t = exp(\frac{N_{noisy}}{N_+}), \tag{12}$$

where N_{noisy} is the total number of noise samples and N_+ is the total number of positive samples.

III. PROPOSED ALGORITHM

In this section, we propose two methods consisting of initializing adaptive weights and adjusting positive label-sensitive

Algorithm 1: AdaBoost

Input: A dataset $X = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ with $y = \{-1, +1\}$; M: maximum iteration; h_1: a member classifier.
Output: H: Ensemble classifier.

Initialize the error weight $D^1 = \{\omega_i^1 | \omega_i^1 = \frac{1}{N}\}$ on each sample $(x_i, y_i) \in X$ ($i = 1, \cdots, N$);
for $t = 1$ ***to*** M **do**
  Train a classifier h_t on X with the error weight D^t;
  Calculate the total error of h_t: $\varepsilon_t = \sum_{i=1}^{N} \omega_i^t, y_i \neq h_t(x_i)$;
  Calculate the confident weight of h_t: $\alpha_t = \frac{1}{2} ln \frac{1-\varepsilon_t}{\varepsilon_t}$;
  Calculate the error weight: $\omega_i^{t+1} = \frac{\omega_i^t e^{-\alpha_t y_i h_t(x_i)}}{L_t}$, where L_t is a normalization constant and $\sum_{i=1}^{N} \omega_i^{t+1} = 1$;
return $H(x) = sign(\sum_{t=1}^{M} \alpha_t h_t(x))$;

confidence weights of the membership classifier to overcome the weakness of AdaBoost for two-class imbalanced datasets. Subsequently, we propose an Im.AdaBoost.W-SVM algorithm to deal with highly imbalanced two-class datasets in the co-authorship recommendation problem.

A. INITIALIZE ADAPTIVE AdaBoost WEIGHTS

We find in the AdaBoost given in Alg. 1 that the set of the initial error weights $\{\omega_i^1 = \frac{1}{N}, i = 1, 2, \cdots, N\}$ assigned to each data sample is initialized equally. At each iteration, AdaBoost evaluates the classification result of each membership classifier and updates the weights for each sample. Specifically, if a sample is misclassified, its error weight is increased, while if a sample is correctly classified, its error weight is decreased. However, in the case of an imbalanced dataset, we need to adjust the algorithm to better adapt for positive samples, that is, we should assign a higher error weight to these samples. To overcome this weakness of AdaBoost for two-class imbalanced datasets, we propose a novel method to initialize error weights to better adapt to the imbalance ratio of positive and negative samples of a dataset. Our method aims to assign a higher initial error weight to positive samples.

We assume that N_{min} and N_{maj} are the number of positive and negative samples (*i.e.*, the number of samples of the minority and majority classes), respectively, where $N_{min} + N_{maj} = N$ and $N_{min} \leq N_{maj}$. We modify the error weights by adding a Δ_{min} value to the error weights of positive samples and subtracting a Δ_{maj} value from the error weights of negative samples. This means the error weight ω_i^1 of each sample $(x_i, y_i) \in X$ $(i = 1, 2, \ldots, N)$ is defined as follows:

$$\omega_i^1 = \begin{cases} \frac{1}{N} + \Delta_{min}, & \text{if } y_i = +1, \\ \frac{1}{N} - \Delta_{maj}, & \text{if } y_i = -1, \end{cases} \tag{13}$$

where Δ_{min} and Δ_{maj} must be satisfied two following conditions:

1) Error weights are greater than 0 and less than $\frac{1}{N}$, or

$$0 < \Delta_{min}, \Delta_{maj} < \frac{1}{N}. \tag{14}$$

2) The total error on the samples is equal to 1, or

$$\frac{N_{min}}{N} + N_{min} \times \Delta_{min} + \frac{N_{maj}}{N} - N_{maj} \times \Delta_{maj} = 1. \tag{15}$$

We consider Eq. (15) along with $N_{min} + N_{maj} = N$, we have:

$$\frac{N_{min} + N_{maj}}{N} + N_{min} \times \Delta_{min} - N_{maj} \times \Delta_{maj} = 1, \tag{16}$$

or

$$N_{min} \times \Delta_{min} = N_{maj} \times \Delta_{maj}. \tag{17}$$

We assume that the ratio of the number of positive samples to that of negative samples is $\delta = \frac{N_{min}}{N_{maj}}$, where $0 < \delta \leq 1$, then from Eq. (17), we have:

$$\Delta_{min} = \frac{N_{maj}}{N_{min}} \times \Delta_{maj} = \frac{\Delta_{maj}}{\delta}. \tag{18}$$

From Eqs. (14) and (18), we have the following conditions for Δ_{min} and Δ_{maj}:

$$\begin{cases} 0 < \Delta_{min}, \Delta_{maj} < \frac{1}{N}, \\ \Delta_{min} = \frac{\Delta_{maj}}{\delta}. \end{cases} \tag{19}$$

We propose to choose $\Delta_{maj} = \frac{1-\delta}{N}$, thus $\Delta_{min} = \frac{1-\delta}{\delta \times N}$. From Eq. (13), the set of initial weights is defined by $D^1 = \{\omega_i^1, i = 1, 2, \ldots, N\}$, where

$$\omega_i^1 = \begin{cases} \frac{1}{N} + \Delta_{min} = \frac{1}{N} + \frac{1-\delta}{\delta \times N}, & \text{if } y_i = +1, \\ \frac{1}{N} - \Delta_{maj} = \frac{1}{N} - \frac{1-\delta}{N}, & \text{if } y_i = -1. \end{cases} \tag{20}$$

It can be seen that, when applying Eq. (20) to datasets with different imbalanced rates, the error weights of positive samples tend to increase, while those of negative samples tend to decrease, depending on the value of $\delta = \frac{N_{min}}{N_{maj}}$. When the dataset is balanced, meaning that $\delta = 1$, $\Delta_{min} = 0$, and $\Delta_{maj} = 0$, and therefore the initial weights D^1 in our proposed method are the same as those of D^1 in the AdaBoost (*i.e.*, the error weights on all samples are $\frac{1}{N}$).

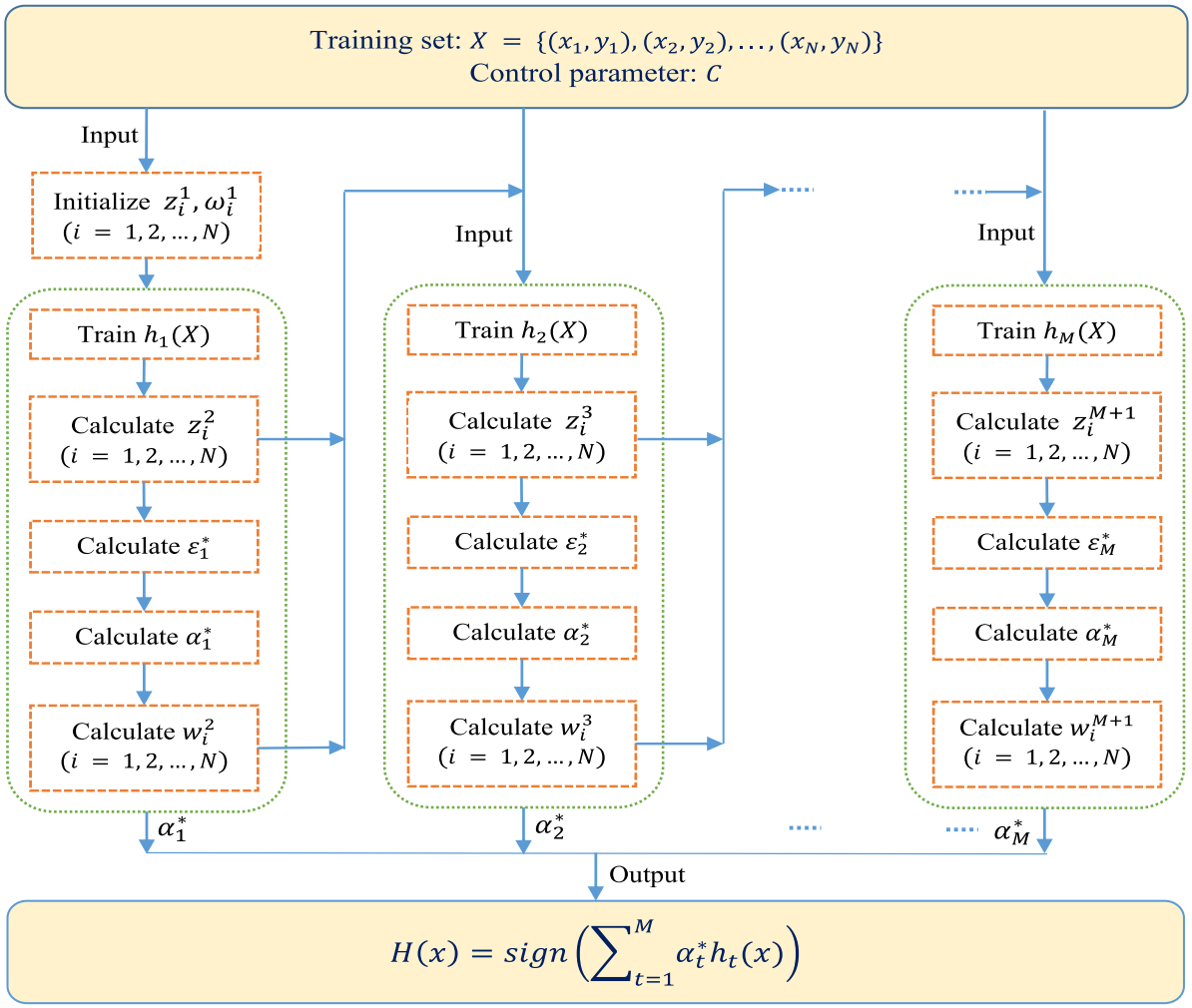


FIGURE 1. Scheme of our proposed Im.AdaBoost.W-SVM algorithm.

In addition, to dynamically adjust the Δ_{min} and Δ_{maj} values according to the individual characteristics of datasets, we propose a more general formula using an exponential parameter θ as follows:

$$\Delta_{maj} = \frac{(1-\delta)^\theta}{N} \text{ and } \Delta_{min} = \frac{(1-\delta)^\theta}{\delta \times N}. \quad (21)$$

For each dataset, we can find the best value of θ through testing on a set of given values. Obviously, our improvements given in Eqs.(13) and (21) make AdaBoost more generalizable on datasets with different imbalance rates.

B. ADJUST CONFIDENCE WEIGHTS OF THE MEMBERSHIP CLASSIFIER

We find in the AdaBoost given in Alg. 1 that it calculates the confidence weight of each membership classifier based on the total error on the entire dataset without considering the characteristics of positive and negative samples in two-class imbalanced datasets, while these characteristics can have an important influence on the results of membership classifiers. To overcome this weakness of AdaBoost for two-class imbalanced datasets, we propose a novel method to adjust the confidence weights of member classifiers based on their sensitivity to the total error caused by positive samples. Our method aims to decrease the confidence weights of misclassified samples if the member classifier misclassifies more positive samples.

We consider the line 5 in the AdaBoost algorithm, the confidence weight α_t of the membership classifier h_t is calculated by a function that is inversely proportional to the total error ε_t. This total error is considered equally across the misclassified samples. For a two-class imbalanced dataset, AdaBoost should be modified such that it gives priority to assigning a high error weight when it misclassifies many positive samples. To do this, we propose a new total error ε_t^* instead of ε_t, which is calculated by the total error of positive samples, denoted by ε_t^+, and that of negative samples, denoted by ε_t^-, *i.e.*,

$$\varepsilon_t^* = \varepsilon_t^- + \varepsilon_t^+, \quad (22)$$

Algorithm 2: Im.AdaBoost.W-SVM

Input: A dataset with N samples $X = \{(x_1, y_1), \ldots, (x_N, y_N)\}$; M: maximum iteration; h_1: a W-SVM classifier; C: a control parameter of W-SVM.

Output: H: Ensemble classifier.

Initialize $\{z_i^1 = 1\}$ and the error weight $D^1 = \{\omega_i^1\}$ for $i = 1, 2, \ldots, N$ using Eqs.(20) and (21) ;

for $t = 1$ ***to*** M **do**

  Train a classifier h_t (*i.e.,* W-SVM) on X with the error weight $D^t = \{\omega_i^t\}$ and $\{z_i^t\}$ for $i = 1, 2, \ldots, N$;

  Calculate $\{z_i^{t+1}\}$ using Eqs.(10), (11), and (12);

  Calculate the total error of h_t: ε_t^* (using Eq.(24));

  Calculate the confidence weight of h_t: α_t^* (using Eq.(25));

  Calculate the error weight: $\omega_i^{t+1} = \frac{\omega_i^t e^{-\alpha_t^* y_i h_t(x_i)}}{L_t}$, where L_t is a normalization constant and $\sum_{i=1}^N \omega_i^{t+1} = 1$;

return $H(x) = sign(\sum_{t=1}^M \alpha_t^* h_t(x))$.

where

$$\begin{cases} \varepsilon_t^+ = \sum_{i=1}^N \omega_i^t, y_i \neq h_t(x_i), y_i = +1, \\ \varepsilon_t^- = \sum_{i=1}^N \omega_i^t, y_i \neq h_t(x_i), y_i = -1. \end{cases} \tag{23}$$

Obviously, ε_t^* depends on ε_t^+ and ε_t^-, and if we want the membership classifier h_t to classify positive samples correctly, then we need to increase ε_t^+. To do this, we define a parameter γ to adjust the effect of the total error ε_t^+ on ε_t^*. Therefore, we redefine ε_t^* as follows:

$$\varepsilon_t^* = \varepsilon_t^- + \gamma * \varepsilon_t^+, \text{ subject to } \gamma > 1. \tag{24}$$

Since $0 < \varepsilon_t^- + \varepsilon_t^+ < 1$, we choose $\gamma = 2 - (\varepsilon_t^- + \varepsilon_t^+)$. Then, the confidence weight α_t of the membership classifier h_t in AdaBoost using our proposed method becomes:

$$\alpha_t^* = \frac{1}{2} ln \frac{1 - \varepsilon_t^*}{\varepsilon_t^*}. \tag{25}$$

It is evident that the total error value ε_t^* in Eq. (24) increases with ε_t^+ and γ, resulting in the confidence weight value α_t^* being adjusted down accordingly. This allows the algorithm can correctly classify as many positive samples as possible.

C. Im.AdaBoost.W-SVM ALGORITHM

This section proposes a novel algorithm to efficiently classify two-class imbalanced datasets in the co-authorship recommendation problem. We call the AdaBoost algorithm enhanced by our two improvements given in Secs. III-A and III-B an Im.AdaBoost algorithm. As shown in [33], AdaBoost combined W-SVM [42] gives a efficient classifier for two-class imbalanced datasets. Therefore, our Im.AdaBoost utilizes W-SVM as a membership classifier, so called Im.AdaBoost.W-SVM. The Im.AdaBoost.W-SVM scheme is described in Fig. 1 and Im.AdaBoost.W-SVM algorithm is shown in Alg. 2, where h_t is a W-SVM membership classifier.

Our Im.AdaBoost.W-SVM algorithm initializes $\{z_i^1 = 1\}$ and the set of adaptive error weights $D^1 = \{\omega_i^1\}$ for $i = 1, 2, \ldots, N$ using by Eqs. (20) and (21) (*i.e.*, the first input in Fig. 1). The algorithm runs in M iterations and in each iteration t ($t = 1, 2, \cdots, M$), it performs as follows. First, the algorithm uses h_t (*i.e.,* W-SVM) to classify the dataset X by using the set of error weights $D^t = \{\omega_i^t\}$ and the parameters $\{z_i^t\}$ for $i = 1, 2, \cdots, N$ (*i.e.,* Train $h_t(X)$ in Fig. 1). Then, the algorithm computes and updates the parameters z_i^{t+1}, ε_t^*, α_t^*, and $D^{t+1} = \{\omega_i^{t+1}, i = 1, 2, \ldots, N\}$ based on the *true/false* classification results on the samples (*i.e.*, Calculate z_i^{t+1}, Calculate ε_t^*, Calculate α_t^*, and Calculate w_i^{t+1} in Fig. 1). The values of z_i^{t+1} and $D^{t+1} = \{\omega_i^{t+1}, i = 1, 2, \ldots, N\}$ are used for the next loop, while the value of α_t^* is used for the ensemble classifier H. It should be noted that the confidence α_t^* is calculated by Eq. (25). After completing M iterations, the ensemble classifier H predicts the class labels for the samples using the formula: $H(x) = sign(\sum_{t=1}^M \alpha_t^* h_t(x))$, where h_t is a W-SVM classifier and x is a feature vector of some sample.

It should be emphasized that we can use the classification algorithms such as Decision Tree [43] or SVM [44] as a member classifier in Im.AdaBoost.

IV. EXPERIMENTS

In this section, we present experiments to evaluate the performance of our Im.AdaBoost.W-SVM algorithm. As we mentioned above, our Im.AdaBoost can utilize the Decision Tree (so called Im.AdaBoost.DecisionTree) or SVM (so called Im.AdaBoost.SVM) algorithm as a member classifier. Therefore, in our following experiments, we not only compare the performance of Im.AdaBoost.W-SVM with that of other popular classification algorithms consisting of Decision Tree [43], SVM [44], CNN [45], W-SVM [42], AdaBoost.DecisionTree [29], AdaBoost.SVM [35], AdaBoost.W-SVM [33], but also compare the performance of Im.AdaBoost.W-SVM with that of Im.AdaBoost. DecisionTree and Im.AdaBoost.SVM on three real-world co-authorship datasets. We chose the Decision Tree, SVM, W-SVM since they are not only popular shallow learning techniques but also commonly used as membership classifiers in the AdaBoost algorithm. While, we chose CNN

TABLE 2. Statistics of the journals from year 2000 to year 2017.

ID	Journals	ISSN	Number of articles	Number of authors	Average of articles per year
1	Chemical Physics Letters	00092614	18931	41806	1113
2	Journal of Molecular Biology	00222836	10806	35217	635
3	Biochemical and Biophysical Research Communications	0006291X	34848	134448	2049

since it is a popular deep learning technique. Moreover, the AdaBoost.DecisionTree, AdaBoost. SVM, and AdaBoost.W-SVM algorithms are efficient ensemble techniques for imbalanced datasets. We ran all experiments of algorithms in PyThon 3.11 software on a laptop computer with Core i7-8550U CPU 1.8 GHz and 16 GB RAM on Windows 11.

A. DATASETS

In practice, authors often submit their articles to related journals rather than one. Therefore, we collected articles from related journals to predict the ability of co-authors to collaborate in the future. To create datasets for the co-authorship recommendation problem, we collected information about authors and articles from *ScienceDirect* published in three journals: *Chemical Physics Letters*, *Journal of Molecular Biology*, *Biochemical and Biophysical Research Communications*, since they are related journals in the Biochemical and Biophysical field. Specifically, we retrieved the information from the website *www.sciencedirect.com* through ScienceDirect APIs consisting of the *article title*, *publication year*, *content summary*, *keyword list*, and *author information* in the period from year 2000 to year 2017. The statistics of the number of articles, the number of authors, and the average number of articles per year of these journals are shown in Table 2. We used the information from journals from year 2010 to year 2014 to calculate the link measures between author pairs and built one data table of co-authorship candidates as follows: (*i*) each data sample is a pair of co-authorship candidates; (*ii*) each sample has attributes of link measures as given in Table 1. We used the information from journals from year 2015 to year 2017 to determine the labels of data samples. Specifically, the label of each data sample is determined by checking whether or not two authors have collaborated on some article. If two authors are co-authors in an article, then the data sample is labeled $+1$. Otherwise, if there is no co-authorship in any article, then the data sample is labeled as -1. By doing so, we created a co-authorship dataset consisting of 372400 samples.

To comprehensively evaluate the performance of our algorithm, we used a bootstrap technique on our co-authorship dataset to create 21 sub-datasets, where the sub-datasets are divided into three groups of small, middle, and large sizes. Each group has seven datasets with the percentage of positive samples in the total number of samples to be 20%, 15%, 10%, 8%, 6%, 4%, and 2% (*i.e.*, the imbalanced ratios of positive and negative samples are 1:4, 1:5.67, 1:9, 1:11.50, 1:15.67, 1:24, and 1:49, respectively). In other words, we evaluate the performance of our algorithm for each imbalanced ratio on three datasets of small, middle, and large sizes. The details of our experimental datasets are shown in Table 3. It should be emphasized that the datasets with a percentage of positive samples being less than 10% are highly imbalanced.

TABLE 3. Description of Co-Authorship datasets.

Dataset		Samples	Pos. Samples	Neg. Samples	Pos. Percentage
Group I (Avg. size = 1607)	Co-Author 1	1800	360	1440	20%
	Co-Author 2	1710	270	1440	15%
	Co Author 3	1620	180	1440	10%
	Co-Author 4	1584	144	1440	8%
	Co-Author 5	1548	108	1440	6%
	Co-Author 6	1512	72	1440	4%
	Co-Author 7	1476	36	1440	2%
Group II (Avg. size = 2678)	Co-Author 8	3000	600	2400	20%
	Co-Author 9	2850	450	2400	15%
	Co-Author 10	2700	300	2400	10%
	Co-Author 11	2640	240	2400	8%
	Co-Author 12	2580	180	2400	6%
	Co-Author 13	2520	120	2400	4%
	Co-Author 14	2460	60	2400	2%
Group III (Avg. size = 4478)	Co-Author 15	4800	600	4200	20%
	Co-Author 16	4650	450	4200	15%
	Co-Author 17	4500	300	4200	10%
	Co-Author 18	4440	240	4200	8%
	Co-Author 19	4380	180	4200	6%
	Co-Author 20	4320	120	4200	4%
	Co-Author 21	4260	60	4200	2%

B. METRICS

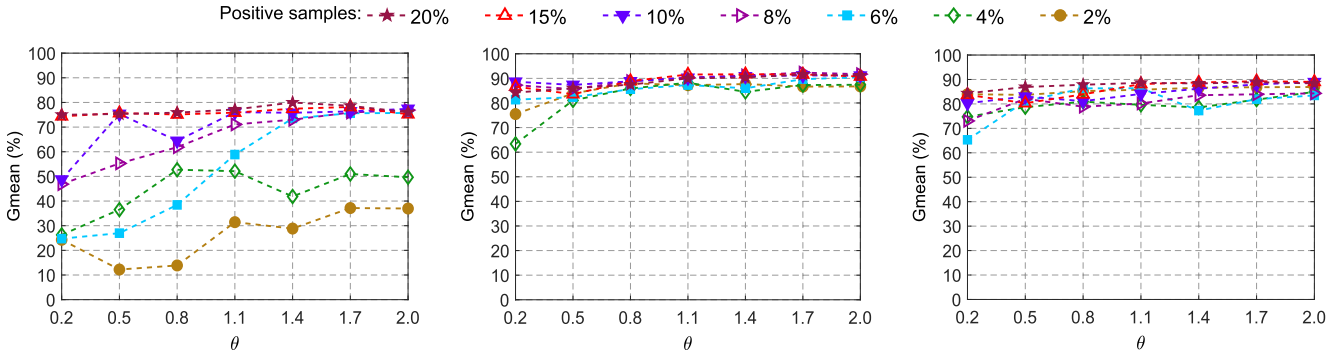
For a two-label classification problem, a confusion matrix built based on the results of a classification algorithm is defined as follows [46]:

$$\begin{bmatrix} TP\ (True\ Positive) & FP\ (False\ Positive) \\ FN\ (False\ Negative) & TN\ (True\ Negative) \end{bmatrix}, \tag{26}$$

where (*i*) *TP* is the number of samples that the algorithm correctly classified positive samples; (*ii*) *FP* is the number of samples that the algorithm misclassified positive samples as negative samples; (*iii*) *FN* is the number of samples that the algorithm misclassified negative samples as positive samples; and (*iv*) *TN* is the number of samples that the algorithm correctly classified negative samples. Accordingly, the accuracy metric of the model achieved by the algorithm is defined by:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \tag{27}$$

However, when we work with two-class imbalanced datasets, the accuracy metric is insufficient to measure the achieved model. For example, if a dataset consists of only 1% of positive and 99% of negative samples, a classification algorithm that achieves a 99% accuracy ratio may misclassify all positive samples, resulting in a useless algorithm. For this reason, several popular metrics used for evaluating a classification



(a) The datasets in Group I (b) The datasets in Group II (c) The datasets in Group III

FIGURE 2. The values of *Gmean* over θ found by Im.AdaBoost.W-SVM.

TABLE 4. The best values of the parameter θ for Im.AdaBoost.DecisionTree, Im.AdaBoost.SVM, and Im.AdaBoost.W-SVM algorithms.

ID	Algorithms	Group I (Co-Author 1 – Co-Author 7)							Group II (Co-Author 8 – Co-Author 14)							Group III (Co-Author 15 – Co-Author 21)						
		20%	15%	10%	8%	6%	4%	2%	20%	15%	10%	8%	6%	4%	2%	20%	15%	10%	8%	6%	4%	2%
1	Im.AdaBoost.DecisionTree	1.7	1.1	1.4	0.5	1.4	2	1.1	1.7	2.0	0.2	1.1	0.8	1.4	1.4	2.0	1.4	1.4	0.2	2.0	2.0	1.7
2	Im.AdaBoost.SVM	1.7	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	1.7	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
3	Im.AdaBoost.W-SVM	1.4	1.7	2.0	2.0	2.0	0.8	1.7	1.7	1.4	1.7	1.7	2.0	1.1	0.8	1.7	1.7	2.0	2.0	1.1	2.0	2.0

model of two-class imbalanced datasets are given as follows:

$$Precision = \frac{TP}{TP + FP}, \tag{28}$$

$$Sensitivity = SE = \frac{TP}{TP + FN}, \tag{29}$$

$$Specificity = SP = \frac{TN}{TN + FP}. \tag{30}$$

Precision is a metric to measure the percentage of positive samples that are actually positive samples, but it does not provide any insight into the number of samples from the positive samples that were mislabeled as negative samples. Therefore, using only *Precision* as a metric is insufficient to evaluate the performance of a classification model. On the other hand, *Sensitivity* (*SE*), also known as the *Recall*, measures the percentage of positive samples that were correctly classified, but it does not take into account the number of negative samples that were misclassified as positive samples. A high sensitivity value indicates that a classification model has a low rate of incorrectly classifying positive samples as negative samples. *Specificity* (*SP*) measures the percentage of negative samples that were correctly classified, but it does not consider the number of positive samples that were misclassified as negative samples. A high specificity value indicates that the classification model has a low rate of incorrectly classifying negative samples as positive samples. To evaluate a more accurate classification model for both positive and negative samples, two other metrics have been proposed as follows:

$$F1\text{-}score = \frac{(1 + \beta^2) \times SE \times Precision}{\beta^2 \times SE + Precision}, \tag{31}$$

$$Gmean = \sqrt{SP \times SE}. \tag{32}$$

*F*1-*score* (*F*1*S*) utilizes the harmonic mean of the precision and specificity metrics, where β is a coefficient to adjust the relative importance of the precision and sensitivity metrics. *Gmean* (geometric mean) utilizes the square root of the product of *SP* and *SE*, and therefore it considers the percentage of both negative and positive samples correctly classified.

Another popular metric to measure the quality of a classification model for two-class imbalanced datasets is the AUC (Area Under the Curve). The AUC is the area under the Receiver Operating Characteristic (ROC) curve, where the ROC is a curve connecting the points represented by the false positive rate (FPR) and the true positive rate (TPR) for different classification thresholds:

$$FPR = \frac{FP}{FP + TN} \text{ and } TPR = \frac{TP}{TP + FN} \text{ (\textit{i.e., SE})}. \tag{33}$$

A higher AUC value means the model is performing better and has a stronger ability to distinguish between positive and negative samples.

Since our Im.AdaBoost.W-SVM is to classify highly imbalanced datasets, we focus on analyzing the values of *SP*, *SE*, *Gmean*, and *AUC* to compare the performance of Im.AdaBoost.W-SVM with those of other algorithms.

C. EXPERIMENTAL RESULTS

In this section, we describe four experiments. The first one aims to find the best parameter values for Im.AdaBoost.DecisionTree, Im.AdaBoost.SVM, and Im.AdaBoost.W-SVM algorithms, while the other ones aim to compare the performance of Im.AdaBoost.W-SVM with that of the different classification algorithms on the datasets of Groups I, II, and III.

1) EXPERIMENT 1

In this experiment, we ran experiments to find the best parameter values for Im.AdaBoost.DecisionTree, Im.AdaBoost.SVM, and Im.AdaBoost.W-SVM on the created datasets. To do this, we inherited the values of parameters $M = 10$ and $C = 10000$ given in [47]. Then, we tested the values of the parameter $\theta \in \{0.2, 0.5, 0.8, 1.1, 1.4, 1.7, 2.0\}$ for Im.AdaBoost.DecisionTree, Im.AdaBoost.SVM, and Im.AdaBoost.W-SVM algorithms. For each value θ, we found the value of *Gmean* and determined the best value of θ such that *Gmean* is maximum.

Figure 2 shows a case where the values of *Gmean* were found by Im.AdaBoost.W-SVM. It should be noted that if there were some values of θ that *Gmean* is maximum, we took one of the θ values.

Table 4 shows the best values of the parameter θ for Im.AdaBoost.DecisionTree, Im.AdaBoost.SVM, and Im.AdaBoost.W-SVM ran on 21 datasets. Obviously, when each dataset is created with a different percentage of positive samples, the value of θ is also different.

2) EXPERIMENT 2

In this experiment, we compared the performance of Im.AdaBoost.W-SVM with that of the classification algorithms mentioned above on the datasets with a small size in Group I.

Table 5 shows the results of this experiment. Accordingly, we can give some main observations based on the values of *Gmean*, *SE*, *SP*, and *AUC* as follows.

- When the percentage of positive samples in the datasets decreases from 20% to 2%, meaning that the imbalance ratio of positive samples increases, only Im.AdaBoost.DecisionTree and Im.AdaBoost.W-SVM found the maximum values of *Gmean* and *AUC*. This shows that both Im.AdaBoost.DecisionTree and Im.AdaBoost.W-SVM have a stronger ability to distinguish between positive and negative instances than the other algorithms. Moreover, Im.AdaBoost.W-SVM found the maximum value of *Gmean* in four of seven datasets and Im.AdaBoost.DecisionTree found the maximum value of *Gmean* in the remaining datasets. However, in three cases where Im.AdaBoost.DecisionTree found the maximum value of *Gmean*, Im.AdaBoost.W-SVM found a higher value of *SE* than Im.AdaBoost.DecisionTree, meaning that Im.AdaBoost.W-SVM classifies positive samples more correctly than Im.AdaBoost.DecisionTree.
- When the percentage of positive samples in the datasets is 10%, 8%, 6%, or 2%, Im.AdaBoost.SVM found 100% of *SE*, 0% of *Gmean*, and 50% of *AUC*, meaning that it correctly classified all positive samples, while misclassifying all negative samples. In these cases, Im.AdaBoost.W-SVM is better than Im.AdaBoost.SVM for classifying both positive and negative samples.
- When the percentage of positive samples is low (*e.g.,* 2%), *i.e.,* the imbalanced rate of positive and negative samples is high, all the SVM, CNN, W-SVM, AdaBoost.DecisionTree, and AdaBoost.W-SVM algorithms misclassified all positive samples since $SE = 0$, while Im.AdaBoost.SVM misclassified all negative samples since $SP = 0$. Only three algorithms consisting of Decision Tree, Im.AdaBoost.DecisionTree, and Im.AdaBoost.W-SVM can classify both negative and positive samples. However, among these algorithms, Im.AdaBoost.W-SVM is the best algorithm for correctly classifying positive samples since it gave the highest value of *SE* compared to Decision Tree and Im.AdaBoost.DecisionTree.

Figure 3 shows a visual comparison of values of *Gmean*, *SE*, and *SP* found by CNN, W-SVM, AdaBoost.W-SVM, Im.AdaBoost.DecisionTree, Im.AdaBoost.SVM, and Im.AdaBoost.W-SVM. We chose CNN, W-SVM, and AdaBoost.W-SVM since (*i*) CNN is a popular deep learning technique; (*ii*) W-SVM is a membership classifier in our Im.AdaBoost.W-SVM; and (*iii*) AdaBoost.W-SVM is an efficient ensemble classifier for imbalanced datasets. Among these algorithms, we see that Im.AdaBoost.DecisionTree and Im.AdaBoost.W-SVM outperforms the others in classifying positive and negative samples. However, since Im.AdaBoost.W-SVM found higher values of *SE* than Im.AdaBoost.DecisionTree, indicating that Im.AdaBoost.W-SVM classified positive samples more correctly than Im.AdaBoost.DecisionTree.

3) EXPERIMENT 3

In this experiment, we compared the performance of all the algorithms mentioned in Experiment 2 on the seven datasets with a middle size in Group II.

Table 6 shows the results of this experiment. Accordingly, we can draw several conclusions as follows:

- Im.AdaBoost.W-SVM found not only the maximum values of *Gmean* and *AUC*, but also the maximum values of *SE* for all seven datasets, meaning that it classified positive samples better than the other algorithms.
- When the percentage of positive samples in the datasets is 20%, the deviation of the values of *Gmean*, *AUC*, and *SE* found by Im.AdaBoost.W-SVM compared with those found by the best algorithm (*i.e.*, Im.AdaBoost.DecisionTree) among the remaining algorithms is 2.76% (91.58% − 88.82%), 2.50%(91.63% − 89.13%), and 13.00% (94.67% − 81.67%), respectively. When the percentage of positive samples in the datasets is 2%, the deviation of the values of *Gmean*, *AUC*, and *SE* found by Im.AdaBoost.W-SVM compared with those found by the best algorithm (*i.e.*, AdaBoost.W-SVM) is 6.33% (87.99% − 81.66%), 5.64%(88.77% − 83.13%), and 26.67% (100% − 73.33%), respectively. Overall, we see that when the percentage of positive samples in the datasets decreases from 20% down to 2%,

TABLE 5. The classification results of datasets in Group I.

Dataset	Algorithm	θ	SP (%)	SE (%)	Gmean (%)	F1-score (%)	Precision (%)	Accuracy (%)	AUC (%)
Co-Author 1 (*Positive: 20%*)	Decision Tree	None	92.96	59.26	74.18	60.90	62.72	87.35	76.11
	SVM	None	99.85	04.81	21.71	09.11	86.67	84.01	52.33
	CNN	None	98.59	09.63	30.02	16.27	55.83	83.77	54.11
	W-SVM	None	92.59	38.89	60.00	44.21	51.26	83.64	65.74
	AdaBoost.DecisionTree	None	94.07	32.22	53.21	37.65	53.52	83.77	63.15
	AdaBoost.SVM	None	100.00	00.00	00.00	00.00	00.00	83.33	50.00
	AdaBoost.W-SVM	None	97.63	12.59	34.94	20.14	56.13	83.46	55.11
	Im.AdaBoost.DecisionTree	1.7	89.70	75.93	**82.40**	66.59	59.44	87.41	**82.81**
	Im.AdaBoost.SVM	1.7	92.59	38.89	60.00	44.21	51.26	83.64	65.74
	Im.AdaBoost.W-SVM	1.4	74.30	**86.30**	79.98	54.90	40.38	76.30	80.30
Co-Author 2 (*Positive: 15%*)	Decision Tree	None	94.00	45.10	65.02	48.77	53.29	87.58	69.55
	SVM	None	100.00	01.47	09.76	02.87	66.67	87.07	50.74
	CNN	None	99.78	00.98	08.08	01.90	33.33	86.81	50.38
	W-SVM	None	98.74	08.82	23.37	13.64	33.21	86.94	53.78
	AdaBoost.DecisionTree	None	97.56	10.29	30.98	15.90	37.71	86.10	53.92
	AdaBoost.SVM	None	99.93	01.96	08.08	03.65	26.67	87.07	50.94
	AdaBoost.W-SVM	None	98.44	07.84	26.80	12.81	50.31	86.55	53.14
	Im.AdaBoost.DecisionTree	1.1	90.96	64.22	76.35	57.26	51.81	87.45	77.59
	Im.AdaBoost.SVM	0.2	98.74	08.82	23.37	13.64	33.21	86.94	53.78
	Im.AdaBoost.W-SVM	1.7	72.37	**85.78**	**78.15**	47.00	32.96	74.13	**79.08**
Co Author 3 (*Positive: 10%*)	Decision Tree	None	96.22	43.70	64.56	48.24	54.69	91.45	69.96
	SVM	None	100.00	00.00	00.00	00.00	00.00	90.91	50.00
	CNN	None	99.70	02.96	16.93	05.51	77.78	90.91	51.33
	W-SVM	None	100.00	00.00	00.00	00.00	00.00	90.91	50.00
	AdaBoost.DecisionTree	None	99.41	02.96	16.91	05.37	37.30	90.64	51.19
	AdaBoost.SVM	None	100.00	00.00	00.00	00.00	00.00	90.91	50.00
	AdaBoost.W-SVM	None	99.85	01.48	07.01	02.72	16.67	90.91	50.67
	Im.AdaBoost.DecisionTree	1.4	93.04	56.30	72.35	49.83	44.78	89.70	74.67
	Im.AdaBoost.SVM	0.2	00.00	**100.00**	00.00	16.67	09.09	09.09	50.00
	Im.AdaBoost.W-SVM	2.0	63.78	94.07	**77.39**	34.27	21.03	66.53	**78.93**
Co-Author 4 (*Positive: 8%*)	Decision Tree	None	96.81	31.48	55.05	36.78	44.41	91.98	64.15
	SVM	None	100.00	00.00	00.00	00.00	00.00	92.59	50.00
	CNN	None	99.78	00.00	00.00	00.00	00.00	92.39	49.89
	W-SVM	None	100.00	00.00	00.00	00.00	00.00	92.59	50.00
	AdaBoost.DecisionTree	None	99.33	01.85	07.84	03.33	16.67	92.11	50.59
	AdaBoost.SVM	None	100.00	00.00	00.00	00.00	00.00	92.59	50.00
	AdaBoost.W-SVM	None	100.00	01.85	07.86	03.51	33.33	92.73	50.93
	Im.AdaBoost.DecisionTree	0.5	94.22	41.67	62.39	39.00	36.70	90.33	67.94
	Im.AdaBoost.SVM	0.2	00.00	**100.00**	00.00	13.79	07.41	07.41	50.00
	Im.AdaBoost.W-SVM	2.0	65.63	90.74	**76.89**	29.81	17.97	67.49	**78.19**
Co-Author 5 (*Positive: 6%*)	Decision Tree	None	98.00	29.63	52.68	35.79	45.81	94.13	63.81
	SVM	None	100.00	00.00	00.00	00.00	00.00	94.34	50.00
	CNN	None	100.00	00.00	00.00	00.00	00.00	94.34	50.00
	W-SVM	None	100.00	00.00	00.00	00.00	00.00	94.34	50.00
	AdaBoost.DecisionTree	None	99.11	00.00	00.00	00.00	00.00	93.50	49.56
	AdaBoost.SVM	None	100.00	00.00	00.00	00.00	00.00	94.34	50.00
	AdaBoost.W-SVM	None	100.00	01.23	06.42	02.38	33.33	94.41	50.62
	Im.AdaBoost.DecisionTree	1.4	94.44	51.85	69.76	42.19	35.73	92.03	73.15
	Im.AdaBoost.SVM	0.2	00.00	**100.00**	00.00	10.71	05.66	05.66	50.00
	Im.AdaBoost.W-SVM	2.0	59.56	96.30	**75.72**	22.13	12.50	61.64	**77.93**
Co-Author 6 (*Positive: 4%*)	Decision Tree	None	98.07	33.33	56.81	36.54	40.60	95.58	65.70
	SVM	None	100.00	00.00	00.00	00.00	00.00	96.15	50.00
	CNN	None	99.85	01.85	07.85	03.33	16.67	96.08	50.85
	W-SVM	None	99.78	01.85	07.83	03.03	08.33	96.01	50.81
	AdaBoost.DecisionTree	None	99.11	11.11	26.68	15.56	44.44	95.73	55.11
	AdaBoost.SVM	None	100.00	00.00	00.00	00.00	00.00	96.15	50.00
	AdaBoost.W-SVM	None	97.41	11.11	18.48	06.78	04.88	94.09	54.26
	Im.AdaBoost.DecisionTree	2.0	95.70	38.89	**60.65**	33.34	29.77	93.52	67.30
	Im.AdaBoost.SVM	0.2	99.78	01.85	07.83	03.03	08.33	96.01	50.81
	Im.AdaBoost.W-SVM	0.8	44.15	**96.30**	52.70	14.68	08.05	46.15	**70.22**
Co-Author 7 (*Positive: 2%*)	Decision Tree	None	99.04	11.11	33.17	14.17	21.43	97.31	55.07
	SVM	None	100.00	00.00	00.00	00.00	00.00	98.04	50.00
	CNN	None	99.93	00.00	00.00	00.00	00.00	97.97	49.96
	W-SVM	None	100.00	00.00	00.00	00.00	00.00	98.04	50.00
	AdaBoost.DecisionTree	None	99.70	00.00	00.00	00.00	00.00	97.75	49.85
	AdaBoost.SVM	None	100.00	00.00	00.00	00.00	00.00	98.04	50.00
	AdaBoost.W-SVM	None	100.00	00.00	00.00	00.00	00.00	98.04	50.00
	Im.AdaBoost.DecisionTree	1.1	97.19	22.22	**45.37**	16.60	14.04	95.72	**59.70**
	Im.AdaBoost.SVM	0.2	00.00	**100.00**	00.00	03.85	01.96	01.96	50.00
	Im.AdaBoost.W-SVM	1.7	18.44	77.78	37.16	03.79	01.94	19.61	48.11

the deviation of the values of *Gmean*, *AUC*, and *SE* was found by Im.AdaBoost.W-SVM compared with the best values of *Gmean*, *AUC*, and *SE* found by the remaining algorithms increases. This means that Im.AdaBoost.W-SVM outperforms the other algorithms when the imbalanced rate of positive and negative samples increases.

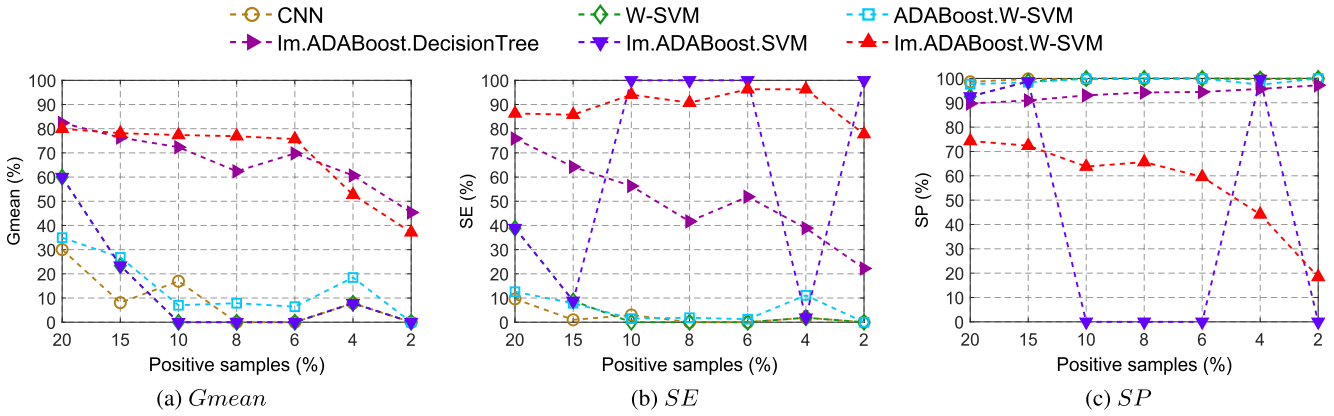


FIGURE 3. The values of *Gmean*, *SE*, and *SP* found on the datasets in Group I.

- When we applied the membership classifiers consisting of Decision Tree, SVM, and W-SVM to our Im.AdaBoost, Im.AdaBoost.DecisionTree, Im.AdaBoost.SVM, and Im.AdaBoost.W-SVM gave higher values of *Gmean*, *AUC*, and *SE* than AdaBoost.DecisionTree, AdaBoost.SVM, and AdaBoost.W-SVM, respectively, for all seven datasets. This shows that our Im.AdaBoost significantly improved the ability to differentiate between positive and negative samples compared with the original AdaBoost.

Figure 4 compares values of *Gmean*, *SE*, and *SP* found by CNN, W-SVM, AdaBoost.W-SVM, Im.AdaBoost.DecisionTree, Im.AdaBoost.SVM, and Im.AdaBoost.W-SVM. We see that Im.AdaBoost.W-SVM found much higher values of *Gmean*, *SE*, and lower values of *SP* than the compared algorithms. This means that Im.AdaBoost.W-SVM found a classification model with a high rate of correctly classifying positive samples. Moreover, when the percentage of positive samples in the datasets decreases from 20% to 2%, Im.AdaBoost.W-SVM achieved the value of *Gmean* decreased from 91.58% down to 87.99%, *SE* increased from 94.67% to 100%, and *SP* decreased from 88.60% to 77.53%. This means that the values of *Gmean*, *SE*, and *SP* found by Im.AdaBoost.W-SVM insignificantly changed even when the percentage of positive samples significantly decreased. In other words, Im.AdaBoost.W-SVM is efficient for classifying samples in the highly imbalanced datasets in Group II.

4) EXPERIMENT 4

In this experiment, we compared the performance of algorithms given in Experiment 1 for all seven datasets in Group III, in which the datasets have a much larger number of samples than that of the datasets in Group I and Group II.

Table 7 shows the results of this experiment. Accordingly, we can outline several main observations as follows:

- Im.AdaBoost.W-SVM found the maximum values of *Gmean* and *AUC* for all seven datasets and the maximum values *SE* for six datasets. However, when the percentage of positive samples in the datasets is 2%, Im.AdaBoost.SVM found 100% of *SE* and 0% of *SP*, meaning that Im.AdaBoost.SVM misclassified all negative samples as positive samples. Therefore, Im.AdaBoost.W-SVM performs better than the compared algorithms in correctly classifying the positive samples in the datasets.
- When the percentage of positive samples in the datasets is 20%, the deviation of the values of *Gmean*, *AUC*, and *SE* found by Im.AdaBoost.W-SVM compared with these second-highest values found by Im.AdaBoost.DecisionTree is 3.72% (88.87% − 85.15%), 3.20%(88.87% − 85.67%), and 11.67% (87.92% − 76.25%), respectively. However, these deviation values increase 16.25% (84.92% − 68.67%) for *Gmean*, 12.00%(85.17% − 73.17%) for *AUC*, and 43.75% (91.67% − 47.92%) for *SE* when the percentage of positive samples in the datasets is 4%. When the percentage of positive samples in the datasets is 2%, the deviation of the values of *Gmean*, *AUC*, and *SE* found by Im.AdaBoost.W-SVM compared with those found by AdaBoost.W-SVM is 10.41%(86.92% − 76.51%), 8.96%(87.04% − 78.08%), and 29.17%(91.67% − 62.50%), respectively. Overall, we see that when the percentage of positive samples decreases, the deviation of the values of *Gmean*, *AUC*, and *SE* found by Im.AdaBoost.W-SVM compared with the highest values of *Gmean* and *SE* found by the other algorithms increases. This means that Im.AdaBoost.W-SVM is more efficient than the other algorithms when the imbalanced rate of positive and negative samples increases.
- As in the results of Experiment 2, we see that when we applied the membership classifiers consisting of Decision Tree, SVM, and W-SVM to our Im.AdaBoost, Im.AdaBoost.DecisionTree, Im.AdaBoost.SVM, and Im.AdaBoost.W-SVM gave higher values of *Gmean*, *AUC*, and *SE* than AdaBoost.DecisionTree, AdaBoost.SVM, and AdaBoost.W-SVM, respectively,

TABLE 6. The classification results of datasets in Group II.

Dataset	Algorithm	θ	SP (%)	SE (%)	Gmean (%)	F1-score (%)	Precision (%)	Accuracy (%)	AUC (%)
Co-Author 8 (*Positive: 20%*)	Decision Tree	None	96.87	80.67	88.39	82.19	83.89	62.78	88.77
	SVM	None	95.87	75.33	84.96	76.84	78.45	61.63	85.60
	CNN	None	95.93	53.00	71.24	61.05	72.27	59.19	74.47
	W-SVM	None	95.87	75.67	85.15	77.05	78.53	61.67	85.77
	AdaBoost.DecisionTree	None	94.93	78.33	86.22	76.91	75.56	61.44	86.63
	AdaBoost.SVM	None	94.73	58.33	71.44	59.26	79.27	59.11	76.53
	AdaBoost.W-SVM	None	96.07	72.33	83.34	75.32	78.61	61.41	84.20
	Im.AdaBoost.DecisionTree	1.7	96.60	81.67	88.82	82.23	82.80	62.74	89.13
	Im.AdaBoost.SVM	0.2	95.87	75.67	85.15	77.05	78.53	61.67	85.77
	Im.AdaBoost.W-SVM	1.7	88.60	**94.67**	**91.58**	75.23	62.42	59.74	**91.63**
Co-Author 9 (*Positive: 15%*)	Decision Tree	None	97.67	66.81	80.74	73.24	81.21	62.42	82.24
	SVM	None	97.20	64.60	79.24	70.53	77.66	61.95	80.90
	CNN	None	98.67	34.07	57.50	46.97	82.24	60.14	66.37
	W-SVM	None	96.67	73.89	84.52	75.40	76.96	62.46	85.28
	AdaBoost.DecisionTree	None	96.07	62.83	77.36	66.07	71.78	61.14	79.45
	AdaBoost.SVM	None	97.33	47.79	68.15	57.68	73.01	60.56	72.56
	AdaBoost.W-SVM	None	96.67	62.39	77.66	67.63	73.83	61.45	79.53
	Im.AdaBoost.DecisionTree	2.0	97.27	68.14	81.39	73.13	79.05	62.30	82.70
	Im.AdaBoost.SVM	0.2	96.67	73.89	84.52	75.40	76.96	62.46	85.28
	Im.AdaBoost.W-SVM	1.4	88.33	**95.13**	**91.67**	69.80	55.13	59.48	**91.73**
Co-Author 10 (*Positive: 10%*)	Decision Tree	None	97.93	66.00	80.35	70.71	76.15	63.35	81.97
	SVM	None	98.53	35.33	58.98	47.12	70.73	61.86	66.93
	CNN	None	98.27	31.33	54.20	40.37	75.11	61.45	64.80
	W-SVM	None	97.00	58.67	75.39	62.12	66.11	62.34	77.83
	AdaBoost.DecisionTree	None	98.33	46.00	67.20	56.51	74.26	62.38	72.17
	AdaBoost.SVM	None	98.60	38.67	61.50	50.28	74.13	62.10	68.63
	AdaBoost.W-SVM	None	97.67	49.33	69.39	57.13	67.86	62.18	73.50
	Im.AdaBoost.DecisionTree	0.2	95.60	66.67	79.79	63.25	60.37	61.98	81.13
	Im.AdaBoost.SVM	0.2	97.00	58.67	75.39	62.12	66.11	62.34	77.83
	Im.AdaBoost.W-SVM	1.7	88.47	**94.00**	**91.19**	60.78	44.90	59.31	**91.23**
Co-Author 11 (*Positive: 8%*)	Decision Tree	None	98.13	54.17	72.74	60.82	69.56	63.25	76.15
	SVM	None	99.00	29.17	53.45	40.80	70.71	62.55	64.08
	CNN	None	99.13	28.33	53.00	40.76	73.17	62.59	63.73
	W-SVM	None	97.13	53.33	71.84	56.22	59.58	62.59	75.23
	AdaBoost.DecisionTree	None	98.13	48.33	68.86	56.31	67.44	62.96	73.23
	AdaBoost.SVM	None	98.67	40.83	62.88	50.95	71.94	62.92	69.75
	AdaBoost.W-SVM	None	98.07	45.00	66.25	52.96	64.78	62.76	71.53
	Im.AdaBoost.DecisionTree	1.1	97.40	65.83	79.99	66.26	67.04	63.37	81.62
	Im.AdaBoost.SVM	0.2	97.13	53.33	71.84	56.22	59.58	62.59	75.23
	Im.AdaBoost.W-SVM	1.7	88.20	**96.67**	**92.33**	56.17	39.59	59.22	**92.43**
Co-Author 12 (*Positive: 6%*)	Decision Tree	None	98.67	46.67	67.77	55.16	68.48	63.82	72.67
	SVM	None	99.73	21.11	45.74	33.51	82.31	63.52	60.42
	CNN	None	99.13	34.44	58.38	46.70	74.15	63.65	66.79
	W-SVM	None	97.67	44.44	65.69	48.21	53.31	63.10	71.06
	AdaBoost.DecisionTree	None	98.27	35.56	59.08	43.20	55.41	63.14	66.91
	AdaBoost.SVM	None	99.13	31.11	55.19	42.25	72.66	63.52	65.12
	AdaBoost.W-SVM	None	97.47	47.78	67.34	49.39	59.05	63.10	72.62
	Im.AdaBoost.DecisionTree	0.8	98.40	52.22	71.68	58.64	67.46	63.86	75.31
	Im.AdaBoost.SVM	1.7	97.67	44.44	65.69	48.21	53.31	63.10	71.06
	Im.AdaBoost.W-SVM	2.0	87.67	**93.33**	**90.44**	46.83	31.28	58.66	**90.50**
Co-Author 13 (*Positive: 4%*)	Decision Tree	None	99.00	43.33	65.50	51.49	63.45	64.57	71.17
	SVM	None	100.00	08.33	28.72	15.34	100.00	64.32	54.17
	CNN	None	100.00	08.33	28.72	15.34	100.00	64.32	54.17
	W-SVM	None	99.33	28.33	52.83	38.83	70.07	64.40	63.83
	AdaBoost.DecisionTree	None	99.60	30.00	53.88	42.27	72.86	64.62	64.80
	AdaBoost.SVM	None	99.87	05.00	15.79	08.57	30.00	64.15	52.43
	AdaBoost.W-SVM	None	96.87	46.67	63.90	37.94	42.46	63.29	71.77
	Im.AdaBoost.DecisionTree	1.4	97.13	50.00	69.64	45.35	42.32	63.55	73.57
	Im.AdaBoost.SVM	0.2	99.33	28.33	52.83	38.83	70.07	64.40	63.83
	Im.AdaBoost.W-SVM	1.1	81.93	**95.00**	**88.23**	29.53	17.49	54.96	**88.47**
Co-Author 14 (*Positive: 2%*)	Decision Tree	None	98.93	30.00	54.39	32.63	36.04	65.05	64.47
	SVM	None	100.00	13.33	35.27	22.92	100.00	65.53	56.67
	CNN	None	100.00	00.00	00.00	00.00	00.00	65.36	50.00
	W-SVM	None	99.60	23.33	47.02	31.11	58.33	65.40	61.47
	AdaBoost.DecisionTree	None	99.60	16.67	38.64	22.55	47.22	65.32	58.13
	AdaBoost.SVM	None	100.00	00.00	00.00	00.00	00.00	65.36	50.00
	AdaBoost.W-SVM	None	92.93	73.33	81.66	27.44	17.08	61.70	83.13
	Im.AdaBoost.DecisionTree	1.4	99.20	40.00	62.08	43.33	48.89	65.36	69.60
	Im.AdaBoost.SVM	0.2	99.60	23.33	47.02	31.11	58.33	65.40	61.47
	Im.AdaBoost.W-SVM	0.8	77.53	**100.00**	**87.99**	15.88	08.66	51.98	**88.77**

for all seven datasets. This shows again that our Im.AdaBoost significantly outperformed the original AdaBoost in ability to distinguish between positive and negative samples.

TABLE 7. The classification results of datasets in Group III.

Dataset	Algorithm	θ	SP (%)	SE (%)	Gmean (%)	F1-score (%)	Precision (%)	Accuracy (%)	AUC (%)
Co-Author 15 (*Positive: 20%*)	Decision Tree	None	97.83	70.00	82.75	77.42	86.60	93.19	83.92
	SVM	None	94.42	72.08	82.50	72.08	72.08	90.69	83.25
	CNN	None	94.33	67.92	80.04	69.21	70.56	89.93	81.13
	W-SVM	None	94.42	72.50	82.74	72.35	72.20	90.76	83.46
	AdaBoost.DecisionTree	None	94.58	72.08	82.57	72.38	72.69	90.83	83.33
	AdaBoost.SVM	None	99.17	27.08	51.82	41.27	86.67	87.15	63.13
	AdaBoost.W-SVM	None	95.17	62.92	77.38	67.26	72.25	89.79	79.04
	Im.AdaBoost.DecisionTree	2.0	95.08	76.25	85.15	75.93	75.62	91.94	85.67
	Im.AdaBoost.SVM	0.2	94.42	72.50	82.74	72.35	72.20	90.76	83.46
	Im.AdaBoost.W-SVM	1.7	89.83	**87.92**	**88.87**	73.65	63.36	89.51	**88.87**
Co-Author 16 (*Positive: 15%*)	Decision Tree	None	97.67	63.33	78.65	70.81	80.28	93.19	80.50
	SVM	None	95.25	60.56	75.95	63.01	65.66	90.72	77.90
	CNN	None	97.33	40.56	62.83	51.23	69.52	89.93	68.94
	W-SVM	None	95.08	62.78	77.26	64.20	65.70	90.87	78.93
	AdaBoost.DecisionTree	None	96.08	51.67	70.46	58.13	66.43	90.29	73.88
	AdaBoost.SVM	None	96.75	48.33	68.38	56.86	69.05	90.43	72.54
	AdaBoost.W-SVM	None	96.67	49.44	69.13	57.61	68.99	90.51	73.06
	Im.AdaBoost.DecisionTree	1.4	98.00	66.67	80.83	74.07	83.33	93.91	82.33
	Im.AdaBoost.SVM	0.2	95.08	62.78	77.26	64.20	65.70	90.87	78.93
	Im.AdaBoost.W-SVM	1.7	86.42	**92.22**	**89.27**	65.23	50.46	87.17	**89.32**
Co-Author 17 (*Positive: 10%*)	Decision Tree	None	97.83	49.17	69.36	57.56	69.41	93.41	73.50
	SVM	None	99.17	30.00	54.54	43.37	78.26	92.88	64.58
	CNN	None	98.92	32.50	56.70	45.35	75.00	92.88	65.71
	W-SVM	None	96.92	60.00	76.26	62.88	66.06	93.56	78.46
	AdaBoost.DecisionTree	None	98.08	49.17	69.44	58.42	71.95	93.64	73.63
	AdaBoost.SVM	None	98.33	46.67	67.74	57.14	73.68	93.64	72.50
	AdaBoost.W-SVM	None	96.83	48.33	68.41	53.70	60.42	92.42	72.58
	Im.AdaBoost.DecisionTree	1.4	95.75	59.17	75.27	58.68	58.20	92.42	77.46
	Im.AdaBoost.SVM	0.2	96.92	60.00	76.26	62.88	66.06	93.56	78.46
	Im.AdaBoost.W-SVM	2.0	81.08	**97.50**	**88.91**	50.43	34.01	82.58	**89.29**
Co-Author 18 (*Positive: 8%*)	Decision Tree	None	98.00	59.38	76.28	64.41	70.37	95.14	78.69
	SVM	None	99.67	15.63	39.46	26.09	78.95	93.44	57.65
	CNN	None	97.83	33.33	57.11	41.56	55.17	93.06	65.58
	W-SVM	None	97.67	33.33	57.06	41.03	53.33	92.90	65.50
	AdaBoost.DecisionTree	None	99.00	32.29	56.54	44.60	72.09	94.06	65.65
	AdaBoost.SVM	None	98.75	28.13	52.70	39.13	64.29	93.52	63.44
	AdaBoost.W-SVM	None	98.42	25.00	49.60	34.53	55.81	92.98	61.71
	Im.AdaBoost.DecisionTree	0.2	96.92	59.38	75.86	60.00	60.64	94.14	78.15
	Im.AdaBoost.SVM	0.2	97.67	33.33	57.06	41.03	53.33	92.90	65.50
	Im.AdaBoost.W-SVM	2.0	75.83	**93.75**	**84.32**	37.82	23.68	77.16	**84.79**
Co-Author 19 (*Positive: 6%*)	Decision Tree	None	98.08	44.44	66.02	50.39	58.18	95.05	71.26
	SVM	None	99.75	23.61	48.53	36.96	85.00	95.44	61.68
	CNN	None	99.33	23.61	48.43	35.05	68.00	95.05	61.47
	W-SVM	None	98.25	36.11	59.56	43.70	55.32	94.73	67.18
	AdaBoost.DecisionTree	None	98.50	44.44	66.16	52.46	64.00	95.44	71.47
	AdaBoost.SVM	None	99.25	31.94	56.31	44.23	71.88	95.44	65.60
	AdaBoost.W-SVM	None	98.33	37.50	60.72	45.38	57.45	94.89	67.92
	Im.AdaBoost.DecisionTree	2.0	97.08	52.78	71.58	52.41	52.05	94.58	74.93
	Im.AdaBoost.SVM	0.2	98.25	36.11	59.56	43.70	55.32	94.73	67.18
	Im.AdaBoost.W-SVM	1.1	84.00	**88.89**	**86.41**	39.02	25.00	84.28	**86.44**
Co-Author 20 (*Positive: 4%*)	Decision Tree	None	98.67	35.42	59.11	41.98	51.52	96.23	67.04
	SVM	None	99.83	12.50	35.33	21.43	75.00	96.47	56.17
	CNN	None	99.67	27.08	51.95	40.00	76.47	96.88	63.37
	W-SVM	None	99.17	25.00	49.79	34.29	54.55	96.31	62.08
	AdaBoost.DecisionTree	None	99.50	27.08	51.91	38.81	68.42	96.71	63.29
	AdaBoost.SVM	None	99.17	18.75	43.12	26.87	47.37	96.07	58.96
	AdaBoost.W-SVM	None	96.92	35.42	58.59	33.33	31.48	94.55	66.17
	Im.AdaBoost.DecisionTree	2.0	98.42	47.92	68.67	51.11	54.76	96.47	73.17
	Im.AdaBoost.SVM	0.2	99.17	25.00	49.79	34.29	54.55	96.31	62.08
	Im.AdaBoost.W-SVM	2.0	78.67	**91.67**	**84.92**	25.29	14.67	79.17	**85.17**
Co-Author 21 (*Positive: 2%*)	Decision Tree	None	99.58	37.50	61.11	47.37	64.29	98.37	68.54
	SVM	None	100.00	00.00	00.00	00.00	00.00	98.04	50.00
	CNN	None	99.67	16.67	40.76	25.00	50.00	98.04	58.17
	W-SVM	None	100.00	00.00	00.00	00.00	00.00	98.04	50.00
	AdaBoost.DecisionTree	None	99.50	25.00	49.87	33.33	50.00	98.04	62.25
	AdaBoost.SVM	None	100.00	00.00	00.00	00.00	00.00	98.04	50.00
	AdaBoost.W-SVM	None	93.67	62.50	76.51	26.09	16.48	93.06	78.08
	Im.AdaBoost.DecisionTree	1.7	99.08	45.83	67.39	47.83	50.00	98.04	72.46
	Im.AdaBoost.SVM	0.2	00.00	**100.00**	00.00	03.85	01.96	01.96	50.00
	Im.AdaBoost.W-SVM	2.0	82.42	91.67	**86.92**	17.12	09.44	82.60	**87.04**

Figure 5 compares *Gmean*, *SE*, and *SP* found by CNN, W-SVM, AdaBoost.W-SVM, Im.AdaBoost.DecisionTree, Im.AdaBoost.SVM, and Im.AdaBoost.W-SVM. We see that Im.AdaBoost.W-SVM achieved not only the highest values

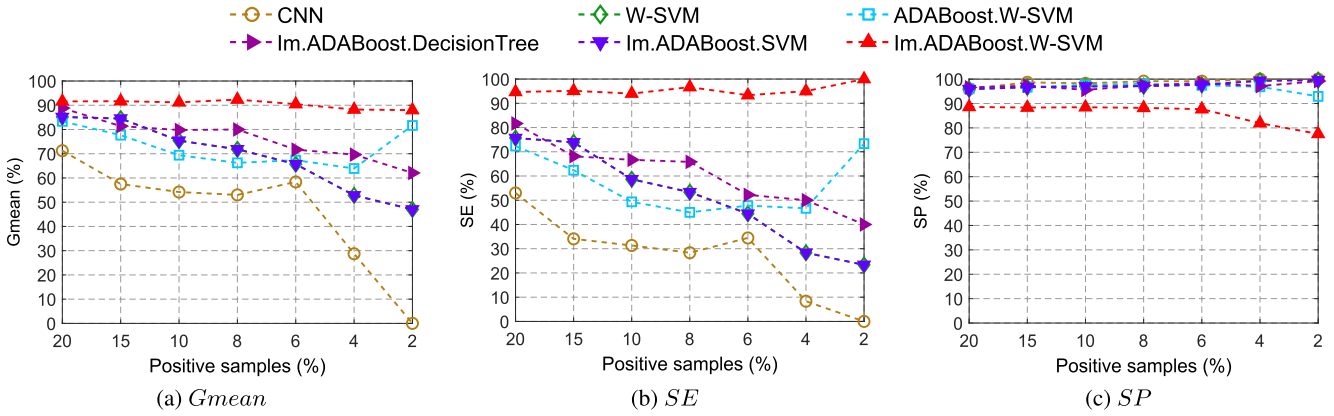


FIGURE 4. The values of *Gmean*, *SE*, and *SP* found on the datasets in Group II.

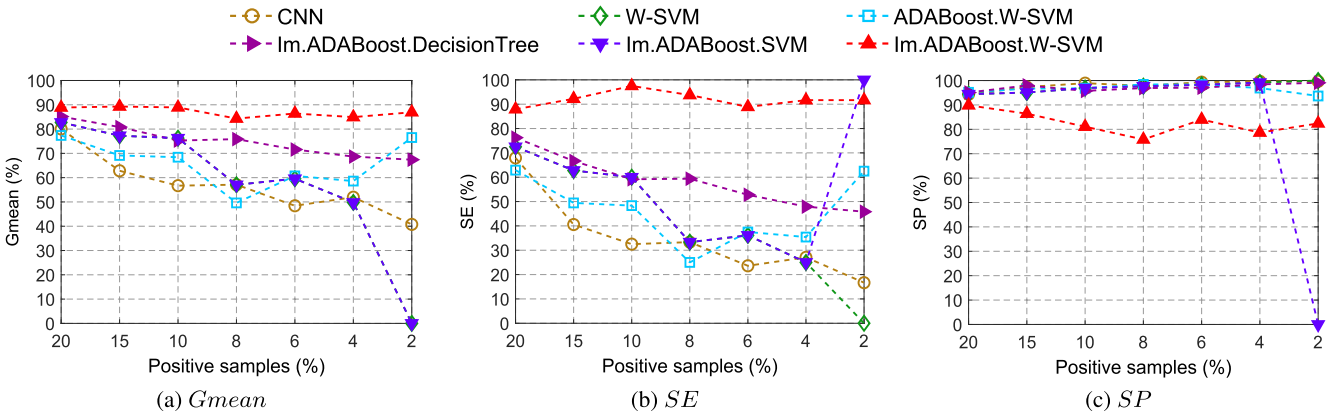


FIGURE 5. The values of *Gmean*, *SE*, and *SP* found on the datasets in Group III.

of *Gmean* but also the highest values of *SE*. This means that Im.AdaBoost.W-SVM has the highest ability among the algorithms to detect accurate positive samples. Moreover, the values of *Gmean*, *SE*, and *SP* found by Im.AdaBoost.W-SVM lightly changed even when the percentage of positive samples significantly decreases. Specifically, the value of *Gmean* only decreases from 88.87% to 86.92%, while the percentage of positive samples decreases from 20% to 2%. In other words, Im.AdaBoost.W-SVM is efficient for classifying samples in the highly imbalanced datasets in Group III.

In summary, the results of the three experiments above show that (*i*) Im.AdaBoost.W-SVM outperforms the compared algorithms when the imbalanced rate of positive and negative samples increases; (*ii*) Even when the percentage of positive samples significantly decreases, Im.AdaBoost.W-SVM is efficient for classifying samples with highly imbalanced rates in datasets; (*iii*) When the number of data samples in datasets significantly increases, Im.AdaBoost.W-SVM exhibits more explicit performance in correctly classifying the positive samples in the datasets. This is because in Im.AdaBoost.W-SVM, we proposed two significant improvements to the original AdaBoost. Firstly, we initialize the set of different error weights adapted to the imbalance rate between positive and negative samples of datasets. Secondly, we calculate the confidence weights of member classifiers based on their sensitivity to the total error caused by positive samples. By doing so, our Im.AdaBoost.W-SVM increases the influence of positive samples and decreases the influence of negative samples in constructing a classifier model.

V. CONCLUSION

In this paper, we solved the co-authorship recommendation problem as a classification problem. Since the datasets generated from this problem exhibit a significant class imbalance, making it challenging to address. To overcome this issue, we proposed a novel algorithm named Im.AdaBoost.W-SVM to effectively handle the challenges caused by these highly imbalanced two-class datasets. Specifically, we first proposed two significant improvements to the original AdaBoost: (*i*) initializing the set of different error weights adapted to the imbalance rate of datasets, which is adjusted by a parameter θ; and (*ii*) calculating the confidence weights of the member classifiers based on sensitivity to the total error caused on positive-class samples such that if the member

classifier misclassifies more positive-class sample, then the confidence weights of these samples are increased. We then used the W-SVM algorithm as a membership classifier in our Im.AdaBoost.W-SVM to classify the imbalanced datasets. To evaluate the performance of our Im.AdaBoost.W-SVM algorithm for the co-authorship recommendation problem, we collected information about authors and articles from the website www.sciencedirect.com and built two-class imbalanced datasets. Our experimental results for our self-built co-authorship datasets showed that our Im.AdaBoost.W-SVM algorithm is efficient for the co-authorship recommendation problem with highly imbalanced two-class datasets. In the future, we will apply our algorithm to datasets created from the co-author recommendation problem using various metrics to improve the quality of collaborative recommendations among authors. Another potential future extension is that we combine the Im.AdaBoost algorithm with weighted membership classifiers to comprehensively evaluate the performance of our algorithm. In addition, we find that the classification problem in highly imbalanced datasets is a common problem in many fields such as health, agriculture, transportation, etc. Therefore, we intend to extend the proposed algorithm to apply to classification problems in these fields to effectively solve the challenges of imbalanced datasets.

REFERENCES

[1] I. Makarov and O. Gerasimova, "Predicting collaborations in co-authorship network," in *Proc. 14th Int. Workshop Semantic Social Media Adaptation Personalization (SMAP)*, Larnaca, Cyprus, Jun. 2019, pp. 1–6.

[2] M. Savić, M. Ivanović, and C. Lakhmi, "Co-authorship networks: An introduction," in *Complex Networks in Software, Knowledge, and Social Systems*, vol. 148, no. 1. Cham, Switzerland: Springer, 2019, pp. 179–192.

[3] S. Kumar, "Co-authorship networks: A review of the literature," *Aslib J. Inf. Manage.*, vol. 67, no. 1, pp. 55–73, Jan. 2015.

[4] T. Iino, H. Inoue, Y. U. Saito, and Y. Todo, "How does the global network of research collaboration affect the quality of innovation?" *Jpn. Econ. Rev.*, vol. 72, no. 1, pp. 5–48, Jan. 2021.

[5] W. Wang, F. Xia, J. Wu, Z. Gong, H. Tong, and B. D. Davison, "Scholar2vec: Vector representation of scholars for lifetime collaborator prediction," *ACM Trans. Knowl. Discovery from Data*, vol. 15, no. 3, pp. 1–19, Jun. 2021.

[6] M. W. Rodrigues, M. A. J. Song, and L. E. Zárate, "Effectively clustering researchers in scientific collaboration networks: Case study on ResearchGate," *Social Netw. Anal. Mining*, vol. 11, no. 1, p. 71, Dec. 2021.

[7] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowl.-Based Syst.*, vol. 151, pp. 78–94, Jul. 2018.

[8] M. Abufouda and K. A. Zweig, "Link classification and tie strength ranking in online social networks with exogenous interaction networks," in *Proc. 6th Int. Workshop Mining Ubiquitous Social Environ.*, Montreal, QC, Canada, Apr. 2016, pp. 1–27.

[9] M. A. Hasan, V. Chaoji, S. Salem, and M. Zaki, "Link prediction using supervised learning," in *Proc. 4th Workshop Link Anal., Counterterrorism Secur.*, Bethesda, MD, USA, Apr. 2006, pp. 798–805.

[10] P. M. Chuan, L. H. Son, M. Ali, T. D. Khang, L. T. Huong, and N. Dey, "Link prediction in co-authorship networks based on hybrid content similarity metric," *Int. J. Speech Technol.*, vol. 48, no. 8, pp. 2470–2486, Aug. 2018.

[11] X. Song, Y. Zhang, R. Pan, and H. Wang, "Link prediction for statistical collaboration networks incorporating institutes and research interests," *IEEE Access*, vol. 10, pp. 104954–104965, 2022.

[12] H. Cho and Y. Yu, "Link prediction for interdisciplinary collaboration via co-authorship network," *Social Netw. Anal. Mining*, vol. 8, no. 1, pp. 1–12, Dec. 2018.

[13] M. Nayyeri, G. M. Cil, S. Vahdati, F. Osborne, M. Rahman, S. Angioni, A. Salatino, D. R. Recupero, N. Vassilyeva, E. Motta, and J. Lehmann, "Trans4E: Link prediction on scholarly knowledge graphs," *Neurocomputing*, vol. 461, pp. 530–542, Oct. 2021.

[14] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, *Learning from Imbalanced Data Sets*. Cham, Switzerland: Springer, 2018.

[15] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002.

[16] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," *IEEE Trans. Syst., Man, Cybern. B Cybern.*, vol. 39, no. 2, pp. 539–550, Apr. 2009.

[17] M. Zeng, B. Zou, F. Wei, and X. Liu, "Effective prediction of three common diseases by combining SMOTE with Tomek links technique for imbalanced medical data," in *Proc. IEEE Int. Conf. Online Anal. Comput. Sci.*, Chongqing, China, May 2016, pp. 225–228.

[18] R. Akbani, S. Kwek, and N. Japkowicz, "Applying support vector machines to imbalanced datasets," in *Proc. 15th Eur. Conf. Mach. Learn.*, Pisa, Italy, Sep. 2004, pp. 39–50.

[19] C. Elkan, "The foundations of cost-sensitive learning," in *Proc. 17th Int. Joint Conf. Artif. Intell.*, Seattle, WA, USA, Aug. 2001, pp. 973–978.

[20] H. Guo and H. L. Viktor, "Learning from imbalanced data sets with boosting and data generation: The DataBoost-IM approach," *ACM SIGKDD Explorations Newslett.*, vol. 6, no. 1, pp. 30–39, Jun. 2004.

[21] T. D. Khang, N. D. Vuong, M. K. Tran, and M. Fowler, "Fuzzy C-means clustering algorithm with multiple fuzzification coefficients," *Algorithms*, vol. 13, no. 7, p. 158, Jun. 2020.

[22] C.-F. Lin and S.-D. Wang, "Fuzzy support vector machines," *IEEE Trans. Neural Netw.*, vol. 13, no. 2, pp. 464–471, Mar. 2002.

[23] Y. Sun, M. S. Kamel, A. K. C. Wong, and Y. Wang, "Cost-sensitive boosting for classification of imbalanced data," *Pattern Recognit.*, vol. 40, no. 12, pp. 3358–3378, Dec. 2007.

[24] X. Tao, Q. Li, C. Ren, W. Guo, Q. He, R. Liu, and J. Zou, "Affinity and class probability-based fuzzy support vector machine for imbalanced data sets," *Neural Netw.*, vol. 122, pp. 289–307, Feb. 2020.

[25] X. Dong, H. Gao, L. Guo, K. Li, and A. Duan, "Deep cost adaptive convolutional network: A classification method for imbalanced mechanical data," *IEEE Access*, vol. 8, pp. 71486–71496, 2020.

[26] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *J. Big Data*, vol. 6, no. 1, pp. 1–54, Dec. 2019.

[27] T. D. Khang, M. K. Tran, and M. Fowler, "A novel semi-supervised fuzzy c-means clustering algorithm using multiple fuzzification coefficients," *Algorithms*, vol. 14, no. 9, pp. 1–12, 2021.

[28] Y. Yan, M. Chen, M.-L. Shyu, and S.-C. Chen, "Deep learning for imbalanced multimedia data classification," in *Proc. IEEE Int. Symp. Multimedia (ISM)*, Miami, FL, USA, Dec. 2015, pp. 483–488.

[29] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, Aug. 1997.

[30] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, and F. Herrera, "Ordering-based pruning for improving the performance of ensembles of classifiers in the framework of imbalanced datasets," *Inf. Sci.*, vol. 354, pp. 178–196, Aug. 2016.

[31] Y. Freund, "Boosting a weak learning algorithm by majority," *Inf. Comput.*, vol. 121, no. 2, pp. 256–285, Sep. 1995.

[32] B. X. Wang and N. Japkowicz, "Boosting support vector machines for imbalanced data sets," *Knowl. Inf. Syst.*, vol. 25, no. 1, pp. 1–20, Oct. 2010.

[33] W. Lee, C.-H. Jun, and J.-S. Lee, "Instance categorization by support vector machines to adjust weights in AdaBoost for imbalanced data classification," *Inf. Sci.*, vol. 381, pp. 92–103, Mar. 2017.

[34] V. D. Quang, T. D. Khang, and N. M. Huy, "Improving AdaBoost algorithm with weighted SVM for imbalanced data classification," in *Proc. 8th Int. Conf. Future Data Secur. Eng.*, Nov. 2021, pp. 125–136.

[35] X. Li, L. Wang, and E. Sung, "AdaBoost with SVM-based component classifiers," *Eng. Appl. Artif. Intell.*, vol. 21, no. 5, pp. 785–795, Aug. 2008.

[36] N. H. C. Lima, A. D. D. Neto, and J. D. de Melo, "Creating an ensemble of diverse support vector machines using Adaboost," in *Proc. Int. Joint Conf. Neural Netw.*, Atlanta, GA, USA, Jun. 2009, pp. 1802–1806.

[37] R. Sundar and P. Murugesan, "Performance enhanced boosted SVM for imbalanced datasets," *Appl. Soft Comput.*, vol. 83, no. 1, Oct. 2019, Art. no. 105601.

[38] A. Tharwat and T. Gabel, "Parameters optimization of support vector machines for imbalanced data using social ski driver algorithm," *Neural Comput. Appl.*, vol. 32, no. 11, pp. 6925–6938, Jun. 2020.
[39] T. Turki and Z. Wei, "Boosting support vector machines for cancer discrimination tasks," *Comput. Biol. Med.*, vol. 101, pp. 236–249, Oct. 2018.
[40] H. A. Hasin and D. Hassan, "Link prediction in co-authorship networks," *Sci. J. Univ. Zakho*, vol. 10, no. 4, pp. 235–257, Nov. 2022.
[41] T. D. Khang, V. D. Quang, and N. D. T. Anh, "Co-authorship recommendation systems," *Hue Univ. J. Sci.*, vol. 127, no. 2A, pp. 109–120, 2018.
[42] X. Yang, Q. Song, and A. Cao, "Weighted support vector machine for data classification," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Montreal, QC, Canada, Dec. 2005, pp. 859–864.
[43] J. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA, USA: Morgan Kaufmann, 1993.
[44] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, pp. 273–297, Apr. 1995.
[45] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," 2015, *arXiv:1511.08458*.
[46] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009.
[47] S. Luo, Z. Dai, T. Chen, H. Chen, and L. Jian, "A weighted SVM ensemble predictor based on AdaBoost for blast furnace ironmaking process," *Int. J. Speech Technol.*, vol. 50, no. 7, pp. 1997–2008, Jul. 2020.

VO DUC QUANG received the B.S. and M.S. degrees in computer science from the Hanoi University of Science and Technology, Vietnam, in 2010 and 2014, respectively, where he is currently pursuing the Ph.D. degree in computer science. He is currently a Lecturer with the Faculty of Information Technology, Vinh University, Vietnam. His current research interests include artificial intelligence, machine learning, and fuzzy logic.

HOANG HUU VIET received the B.S. degree in mathematics from Vinh University, Nghe An, Vietnam, in 1994, the B.S. and M.S. degrees in computer science from the Hanoi University of Science and Technology, Hanoi, Vietnam, in 1998 and 2002, respectively, and the Ph.D. degree in computer engineering from Kyung Hee University, Republic of Korea, in 2013. He is currently an Associate Professor with the Faculty of Information Technology, Vinh University. His current research interests include artificial intelligence, machine learning, and adaptive signal processing.

VU HOANG LONG is currently pursuing the B.S. degree with the Hanoi University of Science and Technology, Vietnam. His current research interests include artificial intelligence, machine learning, and data mining.

TRAN DINH KHANG received the Diploma degree in mathematics from Technische Universitaet Dresden, Germany, in 1986, and the Ph.D. degree in computer science from the Hanoi University of Technology, Vietnam, in 1999. He has been an Associate Professor with the School of Information and Communication Technology, Hanoi University of Science and Technology, since 2000. His main research topics include computational intelligence, hedge algebras, fuzzy information processing, and decision support systems.