13th International Conference on Intelligent Control and Information Processing (ICICIP 2025)
Muscat, Oman, February 6-11, 2025

Bi-clustering of Binary Data via Neurodynamics-driven Binary Matrix Factorization

Hongzong Li
Department of Computer Science
City University of Hong Kong
Kowloon, Hong Kong
hongzli2-c@my.cityu.edu.hk

Jun Wang
Department of Computer Science
& Department of Data Science
City University of Hong Kong
Kowloon, Hong Kong
jwang.cs@cityu.edu.hk

2025 13th International Conference on Intelligent Control and Information Processing (ICICIP) | 979-8-3315-1614-7/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICICIP64458.2025.10898111

***Abstract*—Bi-clustering, also known as co-clustering, is a powerful data analysis technique that simultaneously clusters rows and columns of a data matrix, revealing hidden patterns. In this paper, we propose a neurodynamics-driven binary matrix factorization approach for bi-clustering binary data. The proposed method utilizes multiple discrete Hopfield networks operating concurrently to explore local optimal solutions. Furthermore, a particle swarm optimization rule is iteratively applied to reinitialize the neuronal states for directing the search toward better solutions. Comparative evaluations across six benchmark datasets reveal that the proposed method outperforms five existing methods in terms of five internal and external indices.**

***Index Terms*—Bi-clustering; co-clustering; binary matrix factorization; quadratic unconstrained binary optimization (QUBO); collaborative neurodynamic optimization; Boltzmann machine**

I. Introduction

Bi-clustering has emerged as a powerful data analysis technique due to its ability to simultaneously cluster rows and columns of a data matrix, uncovering local patterns. It has found wide applications in various domains such as microarray and gene expression analysis [1], computational biology [2], biomedicine [3], text mining [4], natural language processing [5], marketing [6], collaborative filtering [7], etc.

Over the past decades, numerous biclustering algorithms have been developed to tackle the challenges inherent in simultaneously clustering both dimensions of data matrices, including Cheng and Church's algorithm [8], the extracting conserved gene expression motifs algorithm [9], the fast divide-and-conquer algorithm [10], the plaid algorithm [11], the large average submatrices algorithm [12], etc. Challenges such as handling overlapping biclusters and scaling to large datasets remain prevalent. Therefore, there is a continuous demand for innovative biclustering algorithms that can effectively address these issues, particularly for binary data matrices where the discrete nature of the data adds an extra layer of complexity.

In his seminal papers [13], [14], John Hopfield foresaw that recurrent neural networks can collectively serve as powerful computational models. Specifically, the Hopfield networks are developed for combinatorial optimization [14], [15]. Since then, a variety of neurodynamic optimization models have been developed for solving numerous optimization problems [16]. Despite the progress, it is acknowledged that an individual neurodynamic model faces challenges in effectively addressing combinatorial optimization problems because gradient-driven neurodynamic models are prone to be trapped in local minima. In recent years, collaborative neurodynamic optimization (CNO) has gained prominence as a hybrid intelligence framework, combining neurodynamic optimization with evolutionary optimization methods to tackle a wide array of complex optimization challenges. In analogy with scattered searches in swarm intelligence, the CNO approach utilizes multiple neurodynamic optimization models to probe local optima. Additionally, it integrates a meta-heuristic rule to update initial neuronal states for escaping from local optimal solutions and exploration of global optimal solutions. A mutation operator may be introduced to preserve a certain level of the diversity of initial neuronal states to prevent premature convergence. As demonstrated in [17], [18], collaborative neurodynamic approaches almost surely converge to the global optima. CNO-driven computationally intelligent problem solvers appear in many applications, including nonnegative matrix factorization [19], Boolean matrix factorization [20], bicriteria sparse nonnegative matrix factorization [21], binary matrix factorization [22], etc.

In this paper, we propose a neurodynamics-driven binary matrix factorization approach for bi-clustering binary data. We formulate the biclustering of binary data as binary matrix factorization and propose a collaborative neurodynamic optimization algorithm for binary matrix factorization. The proposed algorithm consists of two phases: one where discrete Hopfield networks with momentum terms (DHNm) are synchronously updated and another where discrete Hopfield networks (DHN)

This work was supported in part by the Research Grants Council of the Hong Kong Special Administrative Region of China under Grants 11202019, and 11203721; and in part by the InnoHK initiative, the Government of the Hong Kong Special Administrative Region, and the Laboratory for AI-Powered Financial Technologies.

are updated in synchronized batches. It employs numerous DHNs along with a particle swarm optimization update rule to reposition DHNs for escaping local optima towards global optima. We demonstrate that it outperforms five prominent baselines in terms of five internal and external indices.

The main contributions of this paper are outlined as follows:

- We develop a collaborative neurodynamic optimization algorithm that combines the discrete Hopfield network's effective exploration ability with momentum terms in scattered local searches, along with the gradient-free update characteristic of particle swarm optimization, to reinitialize neuronal searches to escape local optimal solutions.
- We validate the effectiveness of the proposed approach through extensive experiments on six benchmark datasets, showing improved clustering performance compared to existing methods in terms of six indices.

The remaining paper is structured as follows. Section II introduces essential preliminaries about neurodynamic optimization. Section III discusses the problem formulation and reformulation. Section IV describes the proposed neurodynamics-driven algorithm. Section V elaborates on the experimental results in six instances. Finally, Section VI provides the concluding remarks.

II. Preliminaries

A. Neurodynamic Optimization

1) Discrete Hopfield Network: DHN exemplifies a recurrent neural network characterized by its binary or bipolar states and a hard-limiter activation function [13]. Let W denote the neuron connection weight matrix, and θ denote the neuron bias vector. DHN operates in discrete time:

$$\begin{cases} u(t+1) = Wx(t) - \theta, \\ x(t) = \sigma(u(t)), \end{cases} \tag{1}$$

where u is the net-input vector, x is the neural state vector, and $\sigma(\cdot)$ is a vector-valued hard-limiter activation function expressed element-wise:

$$\sigma(u_i) = \begin{cases} 0 & \text{if} \;\; u_i(t) \le 0, \\ 1 & \text{if} \;\; u_i(t) > 0, \end{cases}$$

it is worth mentioning that for a search problem, the weights W and threshold θ are fixed and are used to represent a cost function.

As demonstrated in [13], the DHN is globally convergent to the local minima of a combinatorial optimization problem as follows:

$$\begin{aligned} \min \;\; & -\frac{1}{2}x^T W x + \theta^T x, \\ \text{s.t.} \;\; & x \in \{0,1\}^n. \end{aligned} \tag{2}$$

It is worth noting that the states of the DHN are determined solely by the sign of the negative gradient of the objective function (i.e., (1)) without being influenced by any historical effect.

If W in (2) is not symmetric, an equivalent approach is to replace it with $(W + W^T)/2$. In the view that the binary variables have $x_i^2 = x_i, \quad i = 1, 2, \ldots, n$, a linear term $\mathrm{diag}(w_{11}, \ldots, w_{nn})x$ is added to realize the zero diagonal elements of W.

As a variant of the DHN, DHNm is proposed in [23], and its neuronal states are updated as:

$$\begin{cases} u(t+1) = u(t) + Wx(t) - \theta, \\ x(t) = \sigma(u(t)). \end{cases} \tag{3}$$

The DHNm in (3) considers historical effects and enhances its dynamic behavior. As shown in [24], [25], the synchronously activated DHNm in (3) is convergent to the local optima of (2).

2) Collaborative Neurodynamic Optimization: Existing collaborative neurodynamic optimization (CNO) approaches utilize various neurodynamic models, including projection neural networks (e.g., [26], [27]), discrete Hopfield networks [22], [28]–[30], and Boltzmann machines [20], [29], [31]–[33]. Almost all of the CNO algorithms [20], [22], [28], [29] use a particle swarm optimization rule in [34] as follows:

$$v_i(t) = c_0 v_i(t-1) + c_1 r_1 (p_i^* - p_i(t-1)) + c_2 r_2 (p^* - p_i(t-1)), \tag{4a}$$

$$\text{if } (r_3 < S(v_i(t))), \text{then } p_i(t) = 1, \text{else } p_i(t) = 0, \tag{4b}$$

where p_i denotes the present position of the i-th particle, v_i denotes the velocity determining the searching direction, p_i^* denotes the present best solution of the i-th particle, p^* denotes the present best solution of a solution set, c_0 is an inertia weight, c_1 is a cognitive learning factor, c_2 is a social learning factor, and $r_1, r_2 \in [0,1]$ are random constants, and $S(\cdot)$ represents a sigmoid limiting transformation.

In a CNO approach, the diversity of initial states is essential for effective search, often enhanced by mutation operations to mitigate premature convergence. The diversity of initial states is quantified as follows:

$$\delta(p) = \frac{1}{Nn} \sum_{i=1}^{N} \| p^{(i)} - p^* \|_2, \tag{5}$$

where N denotes the population size (i.e., the total number of neurodynamic models), n is the dimensionality of the solution, $p^{(i)}$ is the initial states of the i-th neurodynamic model, and p^* is the current best solution across the entire population.

Bit-flip mutation, a commonly used mutation operator for combinatorial optimization [35], is expressed as:

$$p_j = \begin{cases} \neg p_j & \text{if} \;\; \kappa \le P_m, \\ p_j & \text{otherwise}, \end{cases} \tag{6}$$

where $\neg p_j$ denotes the negation of p_j, $\kappa \in [0,1]$ is a random number, and P_m is a preset mutation probability.

III. Problem Formulation

Let $A \in \{0,1\}^{m \times n}$ denote a binary data matrix, where m is the number of rows (data points) and n is the number of columns (features). The goal is to identify a set

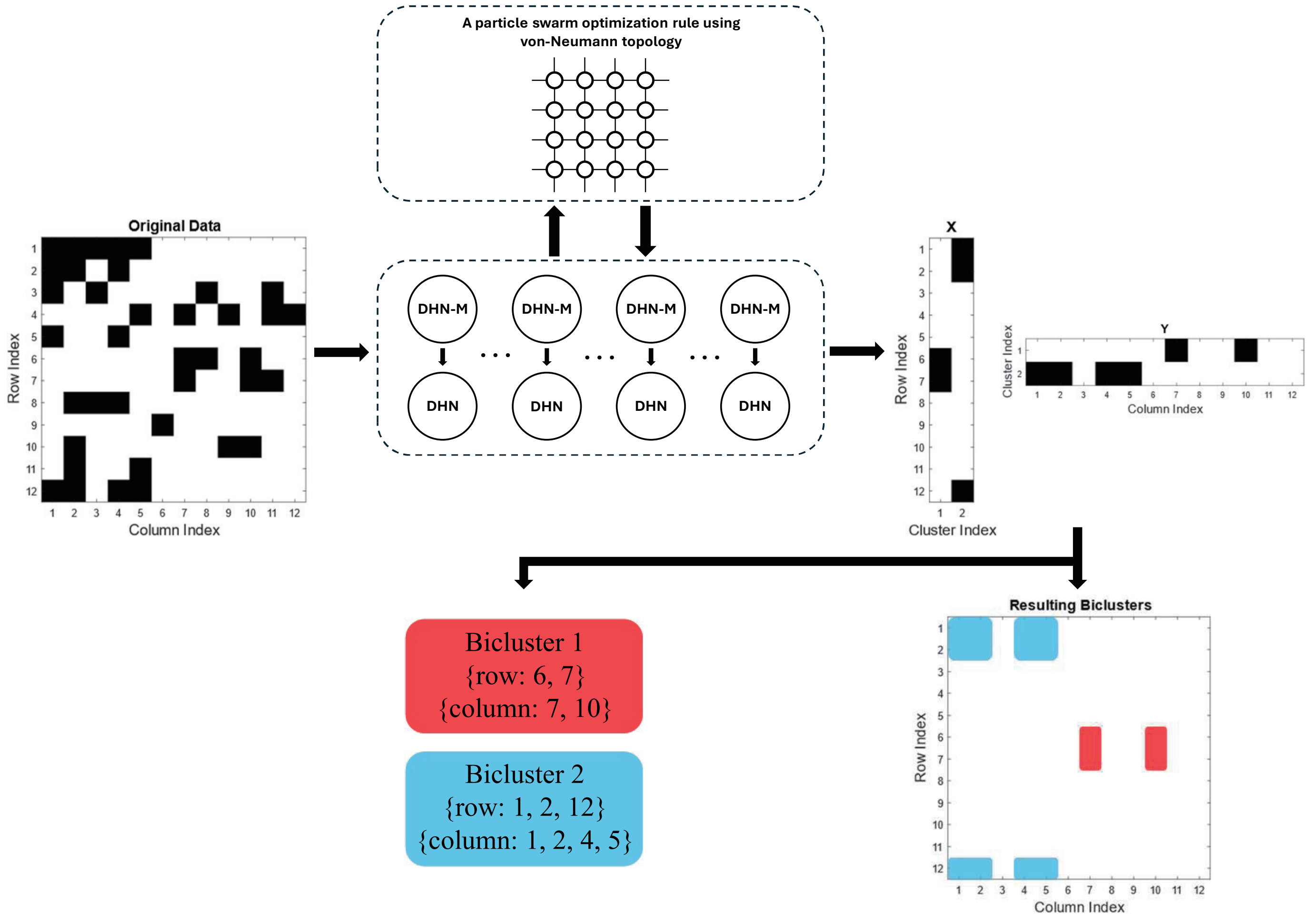

A particle swarm optimization rule using von-Neumann topology
Original Data
Row Index
Column Index
DHN-M
DHN
X
Y
Cluster Index
Resulting Biclusters
Bicluster 1 {row: 6, 7} {column: 7, 10}
Bicluster 2 {row: 1, 2, 12} {column: 1, 2, 4, 5}

Fig. 1. A schematic diagram of the CNO-BC algorithm.

of biclusters $\{I_k, J_k\}_{k=1}^{n_b}$, where $I_k \subseteq \{1, 2, \ldots, m\}$ and $J_k \subseteq \{1, 2, \ldots, n\}$ are the subsets of rows and columns, respectively; n_b is the number of desired biclusters. Each bicluster should exhibit a specific pattern or high similarity among its elements.

An approach to bi-clustering binary data is to solve a binary matrix factorization (BMF) problem [36]. In the BMF approach, the binary data matrix A is factorized approximately as two lower-dimensional binary matrices $X \in \{0,1\}^{m\times n_b}$ and $Y \in \{0,1\}^{n_b\times n}$:

$$A \approx XY,$$

where n_b is the rank of the factorization, corresponding to the number of biclusters. X and Y represent the memberships of rows and columns to biclusters, respectively, and .

Consider the following BMF problem for biclustering:

$$\begin{aligned} \min_{X,Y} \quad & f(X,Y) := \|XY - A\|_F^2, \\ \text{s.t.} \quad & X \in \{0,1\}^{n\times n_b},\ Y \in \{0,1\}^{n_b\times m}, \end{aligned} \tag{7}$$

where $\|\cdot\|_F$ is the Frobenius norm. By minimizing the objective function, it aims to find X and Y that capture the underlying structure of A while preserving as much information as possible.

IV. Algorithm Description

Let $\tilde{x}_i \in \{0,1\}^{n_b}$ represent the i-th row of the matrix X, and let $y_j \in \{0,1\}^{n_b}$ represent the j-th column of the matrix Y, where $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$.

$$\begin{aligned} &\|XY - A\|_F^2 = \sum_{i=1}^{n}\sum_{j=1}^{m}(\tilde{x}_i y_j - a_{ij})^2 = \\ &\sum_{i=1}^{n}\sum_{j=1}^{m}((\tilde{x}_i y_j)^2 - 2a_{ij}\tilde{x}_i y_j + a_{ij}^2) = \\ &\sum_{i=1}^{n}\sum_{j=1}^{m}(y_j^T \tilde{x}_i^T \tilde{x}_i y_j - 2a_{ij}\tilde{x}_i y_j + a_{ij}^2). \end{aligned} \tag{8}$$

In view that $x_{ik}^2 = x_{ik}$ and $y_{kj}^2 = y_{kj}$, the fourth-degree

Algorithm 1: CNO-BC

```
Input: Data matrix $A$, population size $N$, termination
       criterion $M$, ordered batch index sets
       $\mathcal{B}_X = \{1, 2, \ldots, n_b\}$ and $\mathcal{B}_Y = \{1, 2, \ldots, n_b\}$,
       particle swarm optimization based parameters
       $c_0$, $c_1$, and $c_2$.
Output: $X^*$ and $Y^*$.
1  For $k = 1, 2, \ldots, N$, generate random initial neuronal
   state matrices $X_k(0) \in \{0,1\}^{n \times n_b}$ and
   $Y_k(0) \in \{0,1\}^{n_b \times m}$, velocity matrices
   $V_k^X \in [-1,1]^{n \times n_b}, V_k^Y \in [-1,1]^{n_b \times m}$, set initial
   group-best matrix and initial individual-best matrices
   $X^* = \bar{X}_k = 0$ and $Y^* = \bar{Y}_k = 0$. Set $q = 0$;
2  while $q \leq M$ do
3      for $k = 1$ to $N$ do
4          $u_k^X(0) \leftarrow X_k(0) \times m \times n_b$,
           $u_k^Y(0) \leftarrow Y_k(0) \times n \times n_b$;
5          while $(f(X_k(t), Y_k(t)) - f(X_k(t+1), Y_k(t+1)))/f(X_k(t), Y_k(t)) < \epsilon$ do
6              Update $X_k(t)$ and $Y_k(t)$ according to (3)
               with $u_k^X(t+1)$ and $u_k^Y(t+1)$;
7          end
8          Shuffle the order of $\mathcal{B}_X$ and $\mathcal{B}_Y$;
9          while $X_k(t) \neq X_k(t+1)$ and
           $Y_k(t) \neq Y_k(t+1)$ do
10             Update every column of $X_k(t)$ in the order
               of $\mathcal{B}_X$ according to (13) ;
11             Update every row of $Y_k(t)$ in the order of
               $\mathcal{B}_Y$ according to (14) ;
12         end
13         if $f(X_k, Y_k) < f(\bar{X}_k, \bar{Y}_k)$ then
14             $\bar{X}_k \leftarrow X_k$ and $\bar{Y}_k \leftarrow Y_k$;
15         end
16     end
17     $(\hat{X}, \hat{Y}) =$
       $\arg\min\{f(X_1(t), Y_1(t)), \ldots, f(X_N(t), Y_N(t))\}$;
18     if $f(\hat{X}, \hat{Y}) < f(X^*, Y^*)$ then
19         $X^* \leftarrow \hat{X}$, $Y^* \leftarrow \hat{Y}$, and $q \leftarrow 0$;
20     else
21         $q \leftarrow q + 1$;
22     end
23     for $k = 1$ to $N$ do
24         Update $X_k$ and $Y_k$ according to (4);
25     end
26     Compute $\delta(q)$ according to (5);
27     if $\delta(q) < \delta_{\min}$ then
28         Perform the bit-flip mutation according to (6);
29     end
30 end
31 Select biclusters $Sr$ and $Sc$ according to (15);
```

monomial in (8)

$$y_j^T \tilde{x}_i^T \tilde{x}_i y_j = \sum_{k=1}^{n_b} \sum_{l=1}^{n_b} x_{ik} x_{il} y_{kj} y_{lj} = \sum_{k=1}^{n_b} \sum_{l \neq k} x_{ik} x_{il} y_{kj} y_{lj} + \sum_{k=1}^{n_b} x_{ik} y_{kj}. \tag{9}$$

As a result,

$$||XY - A||_F^2 = \sum_{i=1}^{n} \sum_{j=1}^{m} \{ \sum_{k=1}^{n_b} \big[\sum_{l \neq k} x_{ik} x_{il} y_{kj} y_{lj} + (1 - 2a_{ij}) x_{ik} y_{kj} \big] + a_{ij}^2 \}. \tag{10}$$

This expansion reformulates the original problem into a higher-order binary optimization problem, which can be transformed into a QUBO problem suitable for neurodynamic optimization.

The problem in (10) is solved as two separate quadratic binary optimization problems: one concerning X while keeping Y fixed, and the other concerning Y while keeping X fixed.

The partial derivatives of $f(X, Y)$ with respect to the elements x_{ij} and y_{jk} are computed as follows, for $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, n_b$, and $k = 1, 2, \ldots, m$:

$$\frac{\partial ||XY - A||_F^2}{\partial x_{ij}} = \sum_{k=1}^{m} \left(\sum_{l \neq j} 2x_{il} y_{lk} + (1 - 2a_{ik}) \right) y_{jk}, \tag{11}$$

$$\frac{\partial ||XY - A||_F^2}{\partial y_{jk}} = \sum_{i=1}^{n} \left(\sum_{l \neq j} 2x_{il} y_{lk} + (1 - 2a_{ik}) \right) x_{ij}. \tag{12}$$

In DHNs, the matrices X and Y represent the neuronal states. Then, the activation functions for updating X and Y are stated as follows:

$$\begin{aligned} U_X(t+1) &= -\nabla_X ||X(t)Y(t) - A||_F^2, \\ &= \left[-\sum_{k=1}^{m} \left(\sum_{l \neq j} 2x_{il} y_{lk} + (1 - 2a_{ik}) \right) y_{jk} \right]_{ij}, \\ X(t) &= g(U_X(t)), \end{aligned} \tag{13}$$

$$\begin{aligned} U_Y(t+1) &= -\nabla_Y ||X(t)Y(t) - A||_F^2, \\ &= \left[-\sum_{i=1}^{n} \left(\sum_{l \neq j} 2x_{il} y_{lk} + (1 - 2a_{ik}) \right) x_{ij} \right]_{jk}, \\ Y(t) &= g(U_Y(t)). \end{aligned} \tag{14}$$

Equation (11) reveals that the partial derivative of x_{ij} is depends solely on x_{il} for $l \neq j$. This suggests that neuronal states within each column of X can be updated simultaneously. Similarly, Equation (12) demonstrates that the partial derivative of y_{jk} is dependent only on y_{lk} for $l \neq j$, suggesting that the neuronal states within each row of Y can be updated concurrently. Consequently, both X and Y can be synchronously updated in n_b batches by updating the states within the same column of X and the same row of Y.

The biclusters are selected as follows:

$$Sr_k = \{i | x_{ik} = 1\}, \quad k = 1, ...n_b \tag{15}$$

$$Sc_k = \{i | y_{ki} = 1\}, \quad k = 1, ...n_b. \tag{16}$$

Fig. 1 illustrates the framework of the proposed CNO-BC algorithm. For the origin data, CNO-BC begins with a population of DHNm's (3) executing synchronously in the first phase to perform coarse searches, followed by the second phase, where DHNs (1) run synchronously in batches for fine searches. The particle swarm optimization rule utilizing a von-Neumann topology is applied to repeatedly reinitialize the neuronal states after they converge to a local optimum. After convergence, the factorized matrices X and Y are obtained, and biclusters are selected based on the factorized matrices.

Algorithm 1 describes the neurodynamic-based binary matrix factorization method for biclustering binary data. In the proposed algorithm, Steps 5-7 involve asynchronously updating X and Y using multiple DHNms until the objective function's decline rate falls below ϵ. Step 9 introduces randomness by shuffling the ordered sets $\mathcal{B}_X$ and $\mathcal{B}_Y$ to enhance solution diversity. In Steps 9-12, each column of X from the randomly shuffled index set $\mathcal{B}_X$ and each row of Y from the randomly shuffled index set $\mathcal{B}_Y$ are alternately updated using multiple DHNs until convergence. Steps 13-15 and 17-22 refine the individual-best and population-best solutions, respectively. In Steps 23-25, the particle swarm optimization rule is applied to X and Y to avoid local minimal solutions during the global search. In Step 26, the diversity of the solution sets is assessed using the metric in (5). If the diversity measure falls below the predefined threshold $\delta_{\min}$, the bit-flip mutation operator in (6) is applied in Steps 27-29. Finally, in Step 31, biclusters Sr and Sc are selected based on (15).

V. Experimental Results

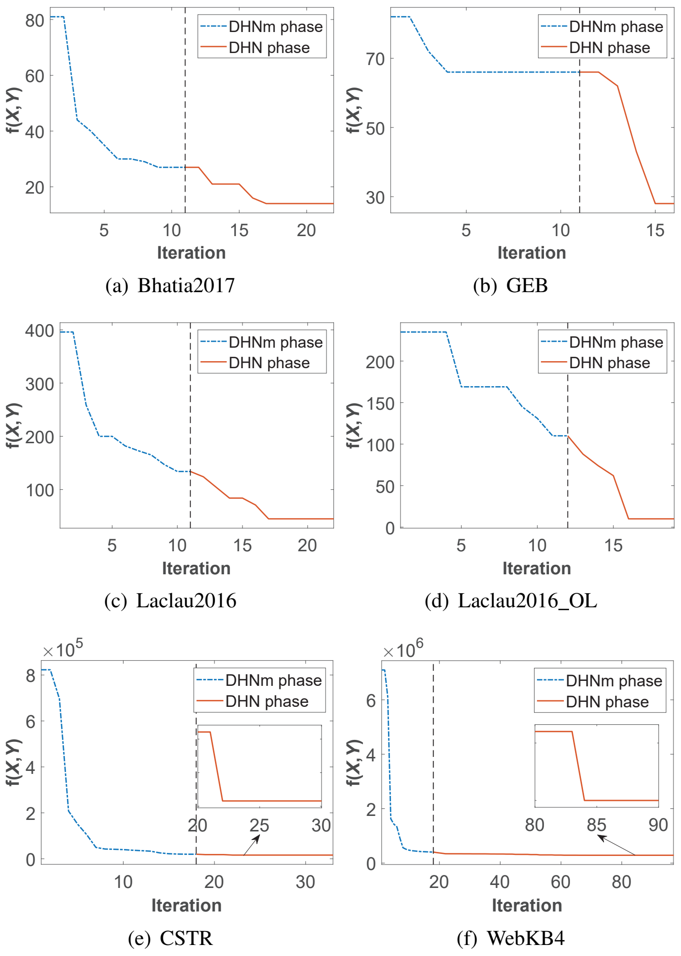
A. Performance Criteria

Evaluating the quality of biclusters obtained from binary data matrices is crucial for assessing the performance of biclustering algorithms. In this study, we employ several performance metrics to quantitatively measure the effectiveness of the biclustering results.

The Jaccard Index measures the similarity between the set of elements in the bicluster and the set of elements in the ground truth bicluster. It is defined as:

$$\text{Jaccard Index} = \frac{|\mathcal{C} \cap \mathcal{T}|}{|\mathcal{C} \cup \mathcal{T}|},$$

where $\mathcal{C}$ is the set of indices (positions) of elements in the bicluster, $\mathcal{T}$ is the set of indices of elements in the ground truth bicluster, and $|\cdot|$ denotes the cardinality (number of elements) of a set.



(a) Bhatia2017 (b) GEB (c) Laclau2016 (d) Laclau2016_OL (e) CSTR (f) WebKB4

Fig. 2. Snapshots of $f(X,Y)$ in (7) during the inner loop of the CNO-BC algorithm on the six datasets, where the blue dotted lines correspond to the DHNm update phase (Steps 5-7), and the red lines represent the DHN update phase (Steps 9-12).

Precision quantifies the proportion of correctly identified positive elements (true positives) among all elements identified as positive in the bicluster. It is defined as:

$$\text{Precision} = \frac{N_{\text{TP}}}{N_{\text{TP}} + N_{\text{FP}}},$$

where $N_{\text{TP}} = |\mathcal{C} \cap \mathcal{T}|$ is the number of true positives (correctly identified ones), and $N_{\text{FP}} = |\mathcal{C}| - N_{\text{TP}}$ is the number of false positives (zeros incorrectly included).

Recall, also known as sensitivity or true positive rate, measures the proportion of actual positive elements (ones) in the ground truth that are correctly identified in the bicluster. It is defined as:

$$\text{Recall} = \frac{N_{\text{TP}}}{N_{\text{TP}} + N_{\text{FN}}},$$

where $N_{\text{FN}} = |T| - N_{\text{TP}}$ is the number of false negatives (ones in the ground truth not included in the bicluster).

TABLE I
THE MEAN VALUES AND STANDARD DEVIATIONS OF THE FIVE INTERNAL AND EXTERNAL INDICES VALUES USING CNO-BC AND THE FIVE BASELINES ON BHATIA2017, GEB, LACLAU2016, LACLAU2016_OL, CSTR, AND WEBKB4, WHERE — INDICATES "NOT APPLICABLE".

Datasets	n	m	n_b	indices	CCA	xMotifs	Bimax	Plaid	LAS	CNO-BC
Bhatia2017	10	7	3	Jaccard Index↑	0.1154 ± 0.0463	0.1237 ± 0.0429	0.3889 ± 0.0000	0.6614 ± 0.1851	0.6667 ± 0.0000	**0.7667 ± 0.0373**
				Precision↑	0.1421 ± 0.0624	0.1478 ± 0.0772	0.4222 ± 0.0000	0.7486 ± 0.1302	0.6667 ± 0.0000	**0.8400 ± 0.0149**
				Recall↑	0.2785 ± 0.1165	0.3333 ± 0.0806	0.5833 ± 0.0000	0.7090 ± 0.2200	0.6667 ± 0.0000	**0.9167 ± 0.0000**
				F1-score↑	0.1776 ± 0.0666	0.1903 ± 0.0659	0.4889 ± 0.0000	0.7146 ± 0.1853	0.6667 ± 0.0000	**0.8508 ± 0.0213**
				NNG↑	-0.2241 ± 0.1441	-0.2473 ± 0.0949	**0.3393 ± 0.0000**	0.2527 ± 0.0554	0.2679 ± 0.0000	0.2976 ± 0.0000
GEB	47	4	2	Jaccard Index↑	0.4524 ± 0.0000	0.4894 ± 0.0000	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	—	**1.0000 ± 0.0000**
				Precision↑	0.5000 ± 0.0000	0.4894 ± 0.0000	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	—	**1.0000 ± 0.0000**
				Recall↑	0.8261 ± 0.0000	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	—	**1.0000 ± 0.0000**
				F1-score↑	0.6230 ± 0.0000	0.6571 ± 0.0000	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	—	**1.0000 ± 0.0000**
				NNG↑	0.0000 ± 0.0000	-0.0270 ± 0.0000	**0.6216 ± 0.0000**	**0.6216 ± 0.0000**	—	**0.6216 ± 0.0000**
Laclau2016	20	20	3	Jaccard Index↑	0.1167 ± 0.0000	0.1333 ± 0.0815	**1.0000 ± 0.0000**	0.5429 ± 0.1609	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**
				Precision↑	0.1167 ± 0.0000	0.1333 ± 0.0815	**1.0000 ± 0.0000**	0.5429 ± 0.1609	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**
				Recall↑	0.3333 ± 0.0000	0.3333 ± 0.0000	**1.0000 ± 0.0000**	0.5500 ± 0.1631	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**
				F1-score↑	0.1728 ± 0.0000	0.1765 ± 0.0844	**1.0000 ± 0.0000**	0.5455 ± 0.1610	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**
				NNG↑	-0.5735 ± 0.0000	-0.1827 ± 0.4843	**0.3333 ± 0.0000**	0.1660 ± 0.0166	**0.3333 ± 0.0000**	**0.3333 ± 0.0000**
Laclau2016_OL	20	17	2	Jaccard Index↑	0.3739 ± 0.0215	0.2701 ± 0.0062	0.5992 ± 0.0000	—	0.7889 ± 0.0000	**1.0000 ± 0.0000**
				Precision↑	0.5588 ± 0.0000	0.3175 ± 0.0118	0.8250 ± 0.0000	—	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**
				Recall↑	0.5410 ± 0.0392	0.3922 ± 0.0052	0.6111 ± 0.0000	—	0.7889 ± 0.0000	**1.0000 ± 0.0000**
				F1-score↑	0.5432 ± 0.0230	0.3507 ± 0.0053	0.6656 ± 0.0000	—	0.8819 ± 0.0000	**1.0000 ± 0.0000**
				NNG↑	0.1626 ± 0.0663	0.1718 ± 0.0207	0.3524 ± 0.0000	—	0.4119 ± 0.0000	**0.5220 ± 0.0000**
CSTR	475	1000	4	Jaccard Index↑	**0.2936 ± 0.0512**	0.0795 ± 0.0131	0.0490 ± 0.0000	—	0.2382 ± 0.0552	0.2537 ± 0.0168
				Precision↑	0.4293 ± 0.0656	0.1905 ± 0.0422	0.2381 ± 0.0000	—	**0.7498 ± 0.1329**	0.4142 ± 0.0155
				Recall↑	**0.4191 ± 0.0662**	0.2138 ± 0.0416	0.0495 ± 0.0000	—	0.2468 ± 0.0564	0.4190 ± 0.0323
				F1-score↑	**0.4137 ± 0.0604**	0.1339 ± 0.0192	0.0820 ± 0.0000	—	0.3623 ± 0.0834	0.3809 ± 0.0233
				NNG↑	-2.4002 ± 0.0259	-0.8708 ± 0.2019	0.0026 ± 0.0000	—	-0.2454 ± 0.0124	**0.0127 ± 0.0000**
WebKB4	4199	1000	4	Jaccard Index↑	0.1172 ± 0.0156	0.1454 ± 0.0331	**0.2517 ± 0.0000**	0.0555 ± 0.0000	0.1824 ± 0.0016	0.2513 ± 0.0312
				Precision↑	0.2814 ± 0.0321	0.3514 ± 0.0593	0.3006 ± 0.0000	0.0804 ± 0.0000	**0.6221 ± 0.0015**	0.2955 ± 0.0164
				Recall↑	0.1799 ± 0.0214	0.2115 ± 0.0385	0.6673 ± 0.0000	0.1043 ± 0.0000	0.2019 ± 0.0015	**0.6822 ± 0.0331**
				F1-score↑	0.2075 ± 0.0239	0.2498 ± 0.0467	0.3669 ± 0.0000	0.0908 ± 0.0000	0.3003 ± 0.0021	**0.3924 ± 0.0405**
				NNG↑	-0.5015 ± 0.0062	-0.1035 ± 0.0069	0.0132 ± 0.0000	-0.0017 ± 0.0000	-0.1789 ± 0.0018	**0.0478 ± 0.0001**

The F1-Score is the harmonic mean of precision and recall, providing a single metric that balances both precision and recall. It is defined as:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

As external indices are label-dependent and labels may not be available, internal indices are preferred. In this paper, we introduce an internal index called the normalized net coverage (NNG). NNG measures correctly identified positive elements (ones) and incorrectly included negative elements (zeros) within bi-clusters, normalized by the total number of ones, as follows:

$$\text{NNG} = \frac{N_{\text{ones}} - N_{\text{zeros}}}{N_{\text{total\_ones}}},$$

where N_{ones} is the number of ones in biclusters, N_{zeros} is the number of zeros in biclusters, and $N_{\text{total\_ones}}$ is the total number of ones in the data matrix. Specifically,

$$N_{\text{ones}} = \sum_{i \in I} \sum_{j \in J} a_{ij},$$

$$N_{\text{zeros}} = |I| \times |J| - N_{\text{ones}},$$

$$N_{\text{total\_ones}} = \sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij}.$$

Substituting them into the NNG formula:

$$\text{NNG} = \frac{N_{\text{ones}} - N_{\text{zeros}}}{N_{\text{total\_ones}}} = \frac{2 \sum_{i \in I} \sum_{j \in J} a_{ij} - |I| \times |J|}{\sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij}}.$$

It is worth noting that the NNG index may yield negative values if the number of zeros exceeds that of ones within bi-clusters. The maximum value of NNG (i.e., the perfect case) is 1.

B. Experiment Setups

The parameters for the CNO-BC algorithm used in the experiments are specified as follows. $N = 10$, $M = 50$, $c_0 = 1$, $c_1 = c_2 = 2$, $\epsilon = 0.01$, $\delta_{\min} = 0.004$, and $P_m = 0.01$.

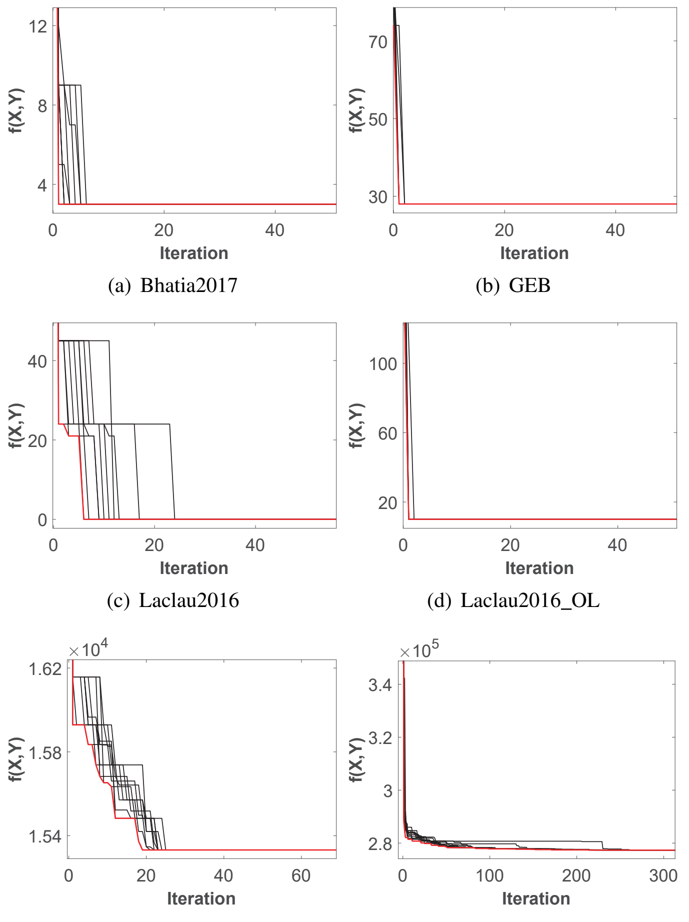
The experiments are based on six datasets: Bhatia2017 [37], GEB[1], Laclau2016 [38], Laclau2016_OL [38], CSTR[2], WebKB4[3], with their key parameters summarized in Table I.

The performance of CNO-BC is evaluated against five state-of-the-art algorithms for bi-clustering: Cheng and Church's algorithm (CCA) [8], the extracting conserved gene expression motifs algorithm (xMotifs) [9], the fast divide-and-conquer algorithm (Bimax) [10], the plaid algorithm (Plaid) [11], the large average submatrices algorithm (LAS) [12]. The codes of CCA, xMotifs, Plaid, and LAS are obtained from a Python package: biclustlib[4]. The code of Bimax is obtained from Github[5].

C. Neurodynamic Behaviors

Fig. 2 presents six snapshots depicting the convergence behavior of the objective function (i.e., $f(X, Y)$ in (7)) as

[1]https://www.bionumerics.com/download/sample-data
[2]https://github.com/franrole/cclust_package/tree/master/datasets
[3]https://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/
[4]https://github.com/padilha/biclustlib
[5]https://github.com/nikitasigal/biclustlib



(a) Bhatia2017 (b) GEB

(c) Laclau2016 (d) Laclau2016_OL

(e) CSTR (f) WebKB4

Fig. 3. The convergent behavior of CNO-BC.

a result of DHNm and DHN updates during the inner loop of CNO-BC across six datasets. The blue dotted lines correspond to the DHNm update phase (Steps 5-7), while the red lines represent the DHN update phase (Steps 9-12). As shown in Fig. 2, the objective function values consistently decrease and stabilize, reaching stationary points within 90 iterations. It indicates the efficient convergence behavior of the proposed method.

Fig. 3 illustrates the convergence behavior of the objective function $f(X, Y)$ using CNO-BC on the six datasets, with the red envelopes indicating the $f(X, Y)$ values of the group-best solutions. It demonstrates that the $f(X, Y)$ values decrease monotonically, and CNO-BC achieves convergence within 250 iterations.

D. Performance Comparisons

Table I presents the mean values and standard deviations of the five internal and external indices values obtained from CNO-BC and the five baseline algorithms, averaged over 50 runs on the six datasets. As shown in Table I, the proposed approach achieves better performance than the five baselines statistically in terms of the five indices.

VI. Concluding remarks

In this paper, we propose a neurodynamics-driven binary matrix factorization approach for bi-clustering binary data. By formulating the biclustering problem as a binary matrix factorization problem, the proposed method achieves statistically superior performance compared to the baselines, attributed to the enhanced exploration capabilities of the discrete Hopfield network, coupled with the highly effective optimization power of the collaborative neurodynamic optimization framework. Experimental results on benchmark datasets demonstrate the effectiveness and superiority of the proposed approach in terms of five internal and external indices. Future work may focus on extending the approach to handle noisy or incomplete data, exploring the incorporation of additional constraints or prior knowledge, and applying the method to larger-scale problems in real-world applications.

References

[1] A. Oghabian, S. Kilpinen, S. Hautaniemi, and E. Czeizler, "Biclustering methods: biological relevance and application in gene expression analysis," *PloS One*, vol. 9, no. 3, p. e90801, 2014.

[2] S. C. Madeira and A. L. Oliveira, "Biclustering algorithms for biological data analysis: a survey," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 1, no. 1, pp. 24–45, 2004.

[3] J. Xie, A. Ma, A. Fennell, Q. Ma, and J. Zhao, "It is time to apply biclustering: a comprehensive review of biclustering applications in biological and biomedical data," *Briefings in Bioinformatics*, vol. 20, no. 4, pp. 1450–1465, 2019.

[4] S. Busygin, O. Prokopyev, and P. M. Pardalos, "Biclustering in data mining," *Computers & Operations Research*, vol. 35, no. 9, pp. 2964–2987, 2008.

[5] Q. Wu, A. Hare, S. Wang, Y. Tu, Z. Liu, C. G. Brinton, and Y. Li, "Bats: A spectral biclustering approach to single document topic modeling and segmentation," *ACM Transactions on Intelligent Systems and Technology*, vol. 12, no. 5, pp. 1–29, 2021.

[6] B. Wang, Y. Miao, H. Zhao, J. Jin, and Y. Chen, "A biclustering-based method for market segmentation using customer pain points," *Engineering Applications of Artificial Intelligence*, vol. 47, pp. 101–109, 2016.

[7] M. G. Silva, S. C. Madeira, and R. Henriques, "A comprehensive survey on biclustering-based collaborative filtering," *ACM Computing Surveys*, 2024.

[8] Y. Cheng and G. Church, "Biclustering of expression data," in *Proceedings of International Conference on Intelligent Systems for Molecular Biology*, vol. 8, 02 2000, pp. 93–103.

[9] T. Murali and S. Kasif, "Extracting conserved gene expression motifs from gene expression data," in *Biocomputing 2003*. World Scientific, 2002, pp. 77–88.

[10] A. Prelić, S. Bleuler, P. Zimmermann, A. Wille, P. Bühlmann, W. Gruissem, L. Hennig, L. Thiele, and E. Zitzler, "A systematic comparison and evaluation of biclustering methods for gene expression data," *Bioinformatics*, vol. 22, no. 9, pp. 1122–1129, 2006.

[11] H. Turner, T. Bailey, and W. Krzanowski, "Improved biclustering of microarray data demonstrated through systematic performance tests," *Computational Statistics & Data Analysis*, vol. 48, no. 2, pp. 235–254, 2005.

[12] A. A. Shabalin, V. J. Weigman, C. M. Perou, and A. B. Nobel, "Finding large average submatrices in high dimensional data," *The Annals of Applied Statistics*, pp. 985–1012, 2009.

[13] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the National Academy of Sciences*, vol. 79, no. 8, pp. 2554–2558, 1982.

[14] J. J. Hopfield and D. W. Tank, "Computing with neural circuits - a model," *Science*, vol. 233, no. 4764, pp. 625–633, 1986.

[15] D. W. Tank and J. J. Hopfield, "Simple 'neural' optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit," *IEEE Trans. Circuits and Systems*, vol. 33, no. 5, pp. 533–541, 1986.

[16] Y. Xia, Q. Liu, J. Wang, and A. Cichocki, “A survey of neurodynamic optimization,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 8, no. 4, pp. 2677–2696, 2024.

[17] Z. Yan, J. Fan, and J. Wang, “A collective neurodynamic approach to constrained global optimization,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 5, pp. 1206–1215, 2017.

[18] H. Che and J. Wang, “A collaborative neurodynamic approach to global and combinatorial optimization,” *Neural Networks*, vol. 114, pp. 15 – 27, 2019.

[19] ——, “A nonnegative matrix factorization algorithm based on a discrete-time projection neural network,” *Neural Networks*, vol. 103, pp. 63–71, 2018.

[20] X. Li, J. Wang, and S. Kwong, “Boolean matrix factorization based on collaborative neurodynamic optimization with Boltzmann machines,” *Neural Networks*, vol. 153, pp. 142–151, 2022.

[21] H. Che, J. Wang, and A. Cichocki, “Bicriteria sparse nonnegative matrix factorization via two-timescale duplex neurodynamic optimization,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 8, pp. 4881–4891, Aug. 2023.

[22] H. Li, J. Wang, N. Zhang, and W. Zhang, “Binary matrix factorization via collaborative neurodynamic optimization,” *Neural Networks*, vol. 176, p. 106348, 2024.

[23] Y. Takefuji and K.-C. Lee, “A near-optimum parallel planarization algorithm,” *Science*, vol. 245, no. 4923, pp. 1221–1223, 1989.

[24] Y. Takefuji and K. C. Lee, “Artificial neural networks for four-coloring map problems and k-colorability problems,” *IEEE Transactions on Circuits and Systems*, vol. 38, no. 3, pp. 326–333, 1991.

[25] G. Galán-Marín and J. Muñoz-Pérez, “Design and analysis of maximum Hopfield networks,” *IEEE Transactions on Neural Networks*, vol. 12, no. 2, pp. 329–339, 2001.

[26] Z. Xia, Y. Liu, and J. Wang, “A collaborative neurodynamic approach to distributed global optimization,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 53, no. 5, pp. 3141–3151, May 2023.

[27] Z. Chen, J. Wang, and Q.-L. Han, “Chiller plant operation planning via collaborative neurodynamic optimization,” *IEEE Transactions on Systems, Man and Cybernetics: Systems*, vol. 53, no. 8, pp. 4623–4635, 2023.

[28] X. Li, J. Wang, and S. Kwong, “Hash bit selection via collaborative neurodynamic optimization with discrete Hopfield networks,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 10, pp. 5116–5124, Oct. 2022.

[29] H. Li and J. Wang, “A collaborative neurodynamic algorithm for quadratic unconstrained binary optimization,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2024, in press.

[30] H. Li, J. Wang, and J. Wang, “Solving the travelling salesman problem based on collaborative neurodynamic optimization with discrete Hopfield networks,” in *2021 11th International Conference on Information Science and Technology (ICIST)*. IEEE, 2021, pp. 456–465.

[31] H. Li and J. Wang, “Capacitated clustering via majorization-minimization and collaborative neurodynamic optimization,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 6, pp. 6679–6692, Jun. 2024.

[32] ——, “A collaborative neurodynamic optimization algorithm based on Boltzmann machines for solving the traveling salesman problem,” in *2021 11th International Conference on Intelligent Control and Information Processing (ICICIP)*. IEEE, 2021, pp. 325–333.

[33] ——, “Collaborative neurodynamic algorithms for solving sudoku puzzles,” in *2022 12th International Conference on Information Science and Technology (ICIST)*. IEEE, 2022, pp. 8–17.

[34] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proceedings of ICNN’95-International Conference on Neural Networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[35] Y. Zhang, S. Wang, P. Phillips, and G. Ji, “Binary PSO with mutation operator for feature selection using decision tree applied to spam detection,” *Knowledge-Based Systems*, vol. 64, pp. 22–31, 2014.

[36] Z.-Y. Zhang, T. Li, C. Ding, X.-W. Ren, and X.-S. Zhang, “Binary matrix factorization for analyzing gene expression data,” *Data Mining and Knowledge Discovery*, vol. 20, pp. 28–52, 2010.

[37] P. S. Bhatia, S. Iovleff, and G. Govaert, “blockcluster: An r package for model-based co-clustering,” *Journal of Statistical Software*, vol. 76, pp. 1–24, 2017.

[38] C. Laclau and M. Nadif, “Hard and fuzzy diagonal co-clustering for document-term partitioning,” *Neurocomputing*, vol. 193, pp. 133–147, 2016.