Mining Discrete Patterns via Binary Matrix Factorization

Bao-Hong Shen
Arizona State University
Tempe, AZ 85287, USA
bhshen@ieee.org

Shuiwang Ji
Arizona State University
Tempe, AZ 85287, USA
shuiwang.ji@asu.edu

Jieping Ye
Arizona State University
Tempe, AZ 85287, USA
jieping.ye@asu.edu

ABSTRACT

Mining discrete patterns in binary data is important for subsampling, compression, and clustering. We consider rank-one binary matrix approximations that identify the dominant patterns of the data, while preserving its discrete property. A best approximation on such data has a minimum set of inconsistent entries, i.e., mismatches between the given binary data and the approximate matrix. Due to the hardness of the problem, previous accounts of such problems employ heuristics and the resulting approximation may be far away from the optimal one. In this paper, we show that the rank-one binary matrix approximation can be reformulated as a 0-1 integer linear program (ILP). However, the ILP formulation is computationally expensive even for small-size matrices. We propose a linear program (LP) relaxation, which is shown to achieve a guaranteed approximation error bound. We further extend the proposed formulations using the regularization technique, which is commonly employed to address overfitting. The LP formulation is restricted to medium-size matrices, due to the large number of variables involved for large matrices. Interestingly, we show that the proposed approximate formulation can be transformed into an instance of the minimum s-t cut problem, which can be solved efficiently by finding maximum flows. Our empirical study shows the efficiency of the proposed algorithm based on the maximum flow. Results also confirm the established theoretical bounds.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications - Data Mining

General Terms

Algorithms

Keywords

Binary matrix factorization, rank-one, integer linear program, regularization, minimum cut, maximum flow

KDD'09, June 28–July 1, 2009, Paris, France.

1. INTRODUCTION

Many applications involve data with discrete attributes and high dimensionality, such as binary images and document-term associations. For such applications, it is desirable to retain the discrete nature of the original data in the factorized components. PCA [8] is a commonly used method for reducing the dimension of continuous data, and it has been used widely in computer vision and many other applications [16, 18]. PCA computes a set of orthogonal projection vectors by the Singular Value Decomposition (SVD) [4], which capture the largest variations in the data. PCA and many other matrix-based techniques [12] deal with data of continuous attributes. Analysis of discrete data sets, however, generally leads to NP-complete/hard problems, especially when physically interpretable results in discrete spaces are desired. For discrete data sets, where all the entries are either 0 or 1, it is desirable to find decomposition where all the entries involved are from $\{0, 1\}$.

A technique, called PROXIMUS [13] was proposed to mine discrete patterns from binary data. PROXIMUS is based on the binary matrix factorization, which separates a data set based on the entries of a binary vector and on recursive partition in the direction of such vectors. The final product for a given binary matrix by PROXIMUS is a tree structure, representing various degrees of similarity among data points. Similar to SVD, the rank-one approximation is at the core of PROXIMUS. A binary rank-one approximation is given by the outer product of two binary vectors, one associated with data points and the other with their attributes. Starting from the root node of a tree, the entries in the data vector serves as indicators for bisection. That is, data points with 1 are made into one group while those with 0 are into the other, e.g., two disjoint submatrices. By this, two child nodes of the root are constructed. The tree is completed by this way recursively for each submatrix until some termination criterion is met, e.g., bisection is not possible for a data vector of all ones. PROXIMUS has been applied for mining high-dimensional discrete-attribute data [13, 14, 15].

The problem of binary rank-one approximations for a given binary matrix is to find two binary vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ such that the outer product of $\mathbf{x}_1$ and $\mathbf{x}_2$ is at the minimum Hamming distance from the given binary matrix [13]. In particular, for a binary matrix $\mathbf{A} \in \{0,1\}^{I_1 \times I_2}$, where I_1 and I_2 are some positive integers, $\mathbf{x}_1 \in \{0,1\}^{I_1}$ and $\mathbf{x}_2 \in \{0,1\}^{I_2}$ are computed by solving the following optimization problem:

$$\min_{\mathbf{x}_1,\mathbf{x}_2} \sum_{i,j=1}^{I_1,I_2} |(\mathbf{A} - \mathbf{x}_1\mathbf{x}_2^T)_{i,j}| \equiv \min_{\mathbf{x}_1,\mathbf{x}_2} \left\| \mathbf{A} - \mathbf{x}_1\mathbf{x}_2^T \right\|_F^2, \qquad (1)$$

where $||\cdot||_F$ denotes the Frobenius norm [4] of a matrix. The problem in (1) is conjectured to be NP-hard [14], and an iterative heuristic for rank-one approximation was proposed in PROXIMUS [13], whose solutions are sensitive to the initialization. Eight heuristics were proposed to initialize the approximation vectors. However, there is no theoretical guarantee on the quality of the approximations [15].

In this paper, we propose high-quality low-rank approximations of discrete data sets with guaranteed error bounds. Our main contributions include:

- We show that the rank-one binary matrix approximation can be formulated as a maximum weight problem in Section 2. We further show that this maximum weight problem can be solved exactly by a 0-1 integer linear program (ILP). Optimal approximations of small-scale data may be found in acceptable time, and their results may be required by error sensitive applications, e.g., a matrix usually of small scale representing associations between patients and pathological symptoms for medical study.

- Based on the equivalent ILP formulation, we propose a linear program (LP) relaxation in Section 3, which scales to medium-size matrices. We show that the solution from the LP relaxation achieves an approximation error bounded by two-fold of that from an optimal approximation. We further integrate the LP and an iterative procedure to improve the quality of the solution

- We extend the proposed exact and approximate formulations using the regularization technique, which is commonly employed to address overfitting. The regularization algorithms require minimal modifications to the standard ones and provide guaranteed solution quality. The error by the regularized approximation algorithm is a function of the regularization parameter $\lambda > 0$, and it is shown to be upper bounded by $\frac{2}{1+\min\{1,\lambda\}}$ in Section 4.

- The algorithms based on the liner programs are still computationally expensive for large matrices, due to the large number of variables involved. In Section 5, we show that the proposed approximate formulation can be transformed into an instance of minimum s-t cut problem, which can be solved efficiently by finding maximum flows.

- We have conducted empirical studies using both synthetic and real data in Section 6. Empirical results show the efficiency of the proposed algorithm based on the maximum flow. Results also confirm the consistency and tightness of the established theoretical bounds.

2. REFORMULATION AS A MAXIMUM WEIGHT PROBLEM

In this section, the problem of rank-one approximation is transformed into a *maximum weight problem* (definition follows). The simple form of the objective function in this problem permits elegant mathematical formulations for optimal solutions.

$$\text{Maximize} \quad \sum_{i,j=1}^{I_1,I_2} \mathbf{W}_{i,j} z_{i,j},$$

subject to

$$-x_{1i} - x_{2j} + 2z_{i,j} \le 0 \qquad \text{for } \mathbf{W}_{i,j} = 1, \tag{3}$$

$$x_{1i} + x_{2j} - z_{i,j} \le 1 \qquad \text{for } \mathbf{W}_{i,j} = -1, \tag{4}$$

$$x_{1i}, x_{2j}, z_{i,j} \in \{0,1\} \qquad \forall i,j \text{ for ILP, or} \tag{5}$$

$$x_{1i}, x_{2j}, z_{i,j} \in [0,1] \qquad \forall i,j \text{ for LP1.} \tag{6}$$

Figure 1: The 0-1 integer linear program (0-1 ILP) including constraints (3)(4)(5) for the maximum weight problem and its relaxation (LP1) including constraints (3)(4)(6).

Since every entry of the given matrix $\mathbf{A}$ is either 0 or 1, the objective function of the binary rank-one approximation can be simplified as follows:

$$\begin{aligned}
\sum_{i,j=1}^{I_1,I_2} \left|(\mathbf{A} - \mathbf{x}_1\mathbf{x}_2^T)_{i,j}\right| &= \sum_{i,j=1}^{I_1,I_2} (\mathbf{A}_{i,j} - \mathbf{x}_{1i}\mathbf{x}_{2j})^2 \\
&= \sum_{i,j=1}^{I_1,I_2} (\mathbf{A}_{i,j}^2 - 2\mathbf{A}_{i,j}\mathbf{x}_{1i}\mathbf{x}_{2j} + \mathbf{x}_{1i}^2\mathbf{x}_{2j}^2) \\
&= \sum_{i,j=1}^{I_1,I_2} \mathbf{A}_{i,j} - \sum_{i,j=1}^{I_1,I_2} \mathbf{x}_{1i}(2\mathbf{A}_{i,j} - 1)\mathbf{x}_{2j} \\
&= \|\mathbf{A}\|_F^2 - \mathbf{x}_1^T\mathbf{W}\mathbf{x}_2,
\end{aligned} \tag{2}$$

where $\mathbf{W} \in \{-1,1\}^{I_1 \times I_2}$ is the weight matrix defined for a given binary matrix $\mathbf{A} \in \{0,1\}^{I_1 \times I_2}$ as $\mathbf{W}_{i,j} = 1$, if $\mathbf{A}_{i,j} = 1$, and $\mathbf{W}_{i,j} = -1$, otherwise. Since $\mathbf{A}$ is fixed in equation (2), an optimal solution can be found by solving $\max_{\mathbf{x}_1,\mathbf{x}_2} \mathbf{x}_1^T\mathbf{W}\mathbf{x}_2$. We call this the "Maximum Weight Problem" (MWP) as formally stated below:

Maximum Weight Problem (MWP):
Given a weight matrix $\mathbf{W} \in \{-1,1\}^{I_1 \times I_2}$, find two vectors $\mathbf{x}_1 \in \{0,1\}^{I_1}$ and $\mathbf{x}_2 \in \{0,1\}^{I_2}$ such that $\sum_{i,j=1}^{I_1,I_2} \mathbf{W}_{i,j}\mathbf{x}_{1i}\mathbf{x}_{2j}$ is maximized.

A 0-1 integer linear program (ILP) is given for the exact solution of MWP in Figure 1. We will show that solving this 0-1 ILP is equivalent to solving MWP. The constraint (5) is employed in the 0-1 ILP instead of (6), which will be used for the LP relaxation in Section 3.

There are three types of binary variables x_{1i}, x_{2j}, and $z_{i,j}$ in the ILP for $1 \le i \le I_1$ and $1 \le j \le I_2$. Variables x_{1i} and x_{2j} represent the i-th and j-th entries of solution vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ respectively. Variable $z_{i,j}$ is an auxiliary variable whose value is 1, if and only if both the values of x_{1i} and x_{2j} are 1. The objective function of MWP can therefore be rewritten as $\sum_{i,j=1}^{I_1,I_2} \mathbf{W}_{i,j} z_{i,j}$ as in Figure 1. Constraints (3) and (4) are for the relation between $z_{i,j}$ and the pair of x_{1i} and x_{2j}. Notice that variable $z_{i,j}$ for each pair of i and j appears exactly twice in two places of the ILP: (i) the term $\mathbf{W}_{i,j} z_{i,j}$ in the objective function and (ii) the one constraint for $\mathbf{W}_{i,j}$. This implies that the value of $z_{i,j}$ is determined by the sign of $\mathbf{W}_{i,j}$ and the other two variables involved in the constraint for $\mathbf{W}_{i,j}$.

$$\begin{aligned}
\text{Maximize} \quad & \frac{1}{2}\sum_{\mathbf{W}_{i,j}=1}(x_{1i}+x_{2j}) - \sum_{\mathbf{W}_{i,j}=-1} z_{i,j}, \\
\text{subject to} \quad & \\
& x_{1i}+x_{2j}-z_{i,j} \le 1 \quad \text{for } \mathbf{W}_{i,j}=-1, \quad (7)\\
& x_{1i}, x_{2j} \in [0,1] \quad \text{for } \forall i,j, \text{ and} \quad (8)\\
& z_{i,j} \in [0,1] \quad \text{for } \mathbf{W}_{i,j}=-1. \quad (9)
\end{aligned}$$

Figure 2: LP2: A linear program for the relaxed problem.

For each positive $\mathbf{W}_{i,j}$, constraint (3) ensures that $z_{i,j}$ equals zero if the value of x_{1i} or x_{2j} is zero. If x_{1i} and x_{2j} both equal to one, the value of $z_{i,j}$ is free to be either 0 or 1; however, the maximization forces the value of $z_{i,j}$ to be one since $\mathbf{W}_{i,j}$ is positive. For each negative $\mathbf{W}_{i,j}$, constraint (4) makes the value of $z_{i,j}$ one if x_{1i} and x_{2j} both equal to one. If any of x_{1i} and x_{2j} equals zero, then the value of $z_{i,j}$ is free to be either 0 or 1; nevertheless, at optimality, $z_{i,j}$ must equal zero since $\mathbf{W}_{i,j}$ has negative contribution to the objective in this case. Consequently, the relation between $z_{i,j}$ and the pair x_{1i} and x_{2j} for some i and j at optimality is as follows: $z_{i,j}=1$, if and only if $x_{1i}=1$ and $x_{2j}=1$. The relation between the ILP and MWP is summarized below.

PROPOSITION 1. *For a given matrix $\mathbf{A} \in \{0,1\}^{I_1 \times I_2}$, the 0-1 ILP in Figure 1 defines an optimal solution $S = \left(\{x_{1i}\}_{i=1}^{I_1}, \{x_{2j}\}_{j=1}^{I_2}, \{z_{i,j}\}_{i,j=1}^{I_1,I_2} \right)$ of binary values. That is, matrix $\mathbf{x}_1\mathbf{x}_2^T$ is an optimal rank-one approximation for $\mathbf{A}$, where $\mathbf{x}_1 = [x_{11} \cdots x_{1I_1}]^T$, and $\mathbf{x}_2 = [x_{21} \cdots x_{2I_2}]^T$. Furthermore, $\|\mathbf{A}\|_F^2 - \sum_{i,j=1}^{I_1,I_2} \mathbf{W}_{i,j} z_{i,j}$ is the minimum value over all possible pairs of $\mathbf{x}_1$ and $\mathbf{x}_2$ for $\mathbf{A}$.*

3. ERROR-BOUNDED APPROXIMATIONS

The ILP formulation in Figure 1 defines the optimal solutions for MWP. However, it is computationally expensive to solve even for small-scale problems. In this section, we propose an efficient algorithm based on the linear programming relaxation for computing error-bounded rank-one approximations. The error by this algorithm is shown to be no more than twice of that by the optimal solution.

3.1 Relaxation for Maximum Weight Problem

We relax the integral constraint (5): $x_{1i}, x_{2i}, z_{ij} \in \{0,1\}$, to the constraint (6): $x_{1i}, x_{2i}, z_{i,j} \in [0,1]$, and obtain a linear program (LP) in Figure 1. We call this LP as *LP1*. Each entry of $\mathbf{x}_1$ and $\mathbf{x}_2$ in LP1 is allowed to have any nonnegative value no more than 1.

Solving LP1 is less difficult than solving its original ILP in Figure 1. Linear programs are polynomial-time solvable [10]. Simplex Method [2] and its variants are widely used to solve LP's. Despite their exponential worst-case complexity [11], simplex algorithms have persistently shown to perform very well in practice [17]. A randomized polynomial-time simplex algorithm is given by [9]. We use simplex algorithms to solve LP relaxations.

A compact LP, named *LP2*, that defines the same set of optimal solutions for LP1 is given in Figure 2. LP2 has fewer constraints and variables, i.e., more efficient than LP1 in terms of running time. The main difference between the two formulations is that all the constraints at (3) in LP1 are incorporated into the objective function of LP2. This can be done because of the following facts: (*i*) LP1 is a maximization problem, (*ii*) the $\mathbf{W}_{i,j}$ for the $z_{i,j}$ in (3) is positive, (*iii*) each $z_{i,j}$ appears in exactly one constraint, (*iv*) a $z_{i,j}$ in (3) is bounded above by $\frac{1}{2}(x_{1i}+x_{2j})$, and (*v*) every variable in LP1 can have a value in the interval $[0,1]$.

Those facts imply that the equality at (3) must hold for LP1 at optimality. Suppose, as contradiction, that there exists a case that the equality at (3) dose not hold at optimality, i.e., $z_{i,j} < \frac{1}{2}(x_{1i}+x_{2j})$ for some i and j. Due to the facts that $\mathbf{W}_{i,j}=1$ and $z_{i,j}$ does not involve in any other constraint, increasing the value of $z_{i,j}$ gives a feasible solution with better objective value, a contradiction. On the other hand, the equality of each constraint at (4), for $\mathbf{W}_{i,j}=-1$, may not hold at optimality, e.g., $z_{i,j}=-1$ for $x_{1i}=0$ and $x_{2j}=0$ if the equality would hold but $z_{i,j} \ge 0$.

Each variable $z_{i,j}$ for positive $\mathbf{W}_{i,j}$ in LP1 becomes redundant and is removed from LP2. However, their values can be obtained from the solution for LP2 as

$$z_{i,j} = \frac{1}{2}(x_{1i}+x_{2j}) \text{ for each } \mathbf{W}_{i,j}=1. \quad (10)$$

Since removing or relaxing a constraint of a maximization program does not reduce an optimal value, the relations among the optimal values l_{ILP}, l_{LP1}, and l_{LP2} of the ILP, LP1, and LP2, respectively, are follows: $l_{\text{ILP}} \le l_{\text{LP1}} = l_{\text{LP2}}$ for the same given weight matrix.

PROPOSITION 2. *For the same problem instance, the optimal value of MWP is no more than that of LP2.*

Algorithm 1: **LPIT** — Algorithm for binary rank-one approximation

Data: matrix $\mathbf{A} \in \{0,1\}^{I_1 \times I_2}$
Result: a pair of vectors in $\{0,1\}^{I_1}$ and $\{0,1\}^{I_2}$

Phase One:
Formulate an instance of LP2 for $\mathbf{A}$;
Find an optimal solution $\mathbf{x}_1 \in \mathbb{R}^{I_1}, \mathbf{x}_2 \in \mathbb{R}^{I_2}$;

Phase Two:
Construct a weight matrix $\mathbf{W}$ from $\mathbf{A}$;
Apply the iterative procedure to improve $\mathbf{x}_1$ and $\mathbf{x}_2$;

3.2 Approximation Algorithm

Our algorithm for binary rank-one approximation consists of two phases: (*i*) find a solution by LP2, and (*ii*) apply an iterative procedure to further improve the solution. The iterative procedure [13] can be done as follows: Since the objective value for MWP is $\mathbf{x}_1^T\mathbf{W}\mathbf{x}_2$, one of the vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ can be quickly determined if the other is fixed. The procedure alternatively finds one of the vectors, while fixing the other, until a termination criterion is met. The method, named *LPIT*, is summarized in Algorithm 1.

Two questions arise from the algorithm LPIT: (1) *Is a solution found in Phase One of LPIT valid? That is, does LP2 find a pair of binary vectors?* and (2) *What is the quality of solutions found by LPIT?*

Every variable in LP2 is allowed to have any value within the interval between 0 and 1. A matrix with fractional values cannot be a binary approximation. The solution by simplex algorithms for LP2 is shown in Section 3.3 to have binary property; namely, every entry in $\mathbf{x}_1$ and $\mathbf{x}_2$ is either 0 or 1.

In addition to feasibility, the quality of solutions by approximation is also important. The second phase of the

algorithm LPIT is a heuristic, whose improvement on solutions is less predictable. The assurance on error-bounded approximations therefore depends on the initial vectors, i.e., optimal solutions for LP2. The discrepancy between objective values of MWP and LP2 is the focus of analysis on the error bound, which is covered in Section 3.4.

3.3 Solutions with the Binary Property

The main result of this section is summarized in the theorem below.

THEOREM 3. *The value of every variable found by the simplex algorithm for LP2 is either 0 or 1.*

The following definitions are relevant to the coefficient matrix of the constraints in LP2. A square matrix is *unimodular* if its determinant is -1 or 1. A matrix $\mathbf{A}$ is *totally unimodular* if every square submatrix $\mathbf{B}$ of $\mathbf{A}$ is unimodular or singular, i.e., $\det \mathbf{B} \in \{-1, 0, 1\}$. A matrix $\mathbf{A}$ is totally unimodular if the condition below holds [5].

SUFFICIENT CONDITION 1 (TOTAL UNIMODULARITY). *Suppose that every column of matrix $\mathbf{A} \in \{-1, 0, 1\}^{I_1 \times I_2}$ has two nonzero entries or fewer. There exists a pair of disjoint sets S_1 and S_2, with $S_1 \cup S_2 = \{1, 2, \ldots, I_1\}$, such that the following condition holds: For every column j with two nonzero entries $\mathbf{A}_{i,j}$ and $\mathbf{A}_{k,j}$, where $i \in S_p$ and $k \in S_q$, $p, q \in \{1, 2\}$, (*i*) $\mathbf{A}_{i,j} \neq \mathbf{A}_{k,j}$ for $p = q$, and (*ii*) $\mathbf{A}_{i,j} = \mathbf{A}_{k,j}$ for $q \neq p$.*

A standard unit vector $\mathbf{e}$ is a vector with exactly one entry of one and the others are of zero, e.g., $[0, 0, 1, 0]^T$. The lemma below can be proved by using basic facts about determinants.

LEMMA 4. *Suppose matrix $\mathbf{A} \in \mathbb{R}^{I_1 \times I_2}$ is totally unimodular. The following matrices are totally unimodular: (a) its transpose $\mathbf{A}^T$, (b) the matrix $[\mathbf{A} \mid -\mathbf{e}] \in \mathbb{R}^{I_1 \times (I_2+1)}$, and (c) a matrix obtained from $\mathbf{A}$ by interchanging two rows or columns.*

PROOF. (a) is due to the fact that $\det \mathbf{B}^T = \det \mathbf{B}$ for any square matrix $\mathbf{B}$. (c) is due to the fact that interchanging two rows or columns changes only the sign of a determinant.

For any matrix $\mathbf{C} \in \mathbb{R}^{n \times n}$ and any row i of $\mathbf{C}$, $\det \mathbf{C} = \sum_{k=1}^{n} (-1)^{i+k} \mathbf{C}_{i,k} \det \mathbf{C}_{[i,k]}$, where $\mathbf{C}_{[i,k]}$ denotes the square submatrix of $\mathbf{C}$ obtained by removing row i and column k from $\mathbf{C}$. A square submatrix $\mathbf{C}$ of the matrix $[\mathbf{A}| - \mathbf{e}] \in \mathbb{R}^{I_1 \times (I_2+1)}$ is either (*i*) a square submatrix of $\mathbf{A}$, (*ii*) $[\mathbf{D}|\mathbf{0}]$, or (*iii*) $[\mathbf{D}| - \mathbf{e}]$, where $\mathbf{D} \in \mathbb{R}^{n \times (n-1)}$ is a submatrix of $\mathbf{A}$ for some n, $1 \leq n \leq \min\{I_1, I_2 + 1\}$. The matrix $\mathbf{C}$ in case (*i*) is unimodular due to the total unimodularity of $\mathbf{A}$. Since matrix $\mathbf{C}$ in case (*ii*) has a column consisting entirely of zeros, $\det \mathbf{C} = 0$, which means $\mathbf{C}$ is unimodular. For a row i corresponding to the nonzero entry of $\mathbf{e}$ in case (*iii*), $\det \mathbf{C} = (-1)^{i+n+1} \det \mathbf{D}_{[i,\varnothing]} \in \{-1, 0, 1\}$ in this case since $\mathbf{D}_{[i,\varnothing]}$ is a square submatrix of $\mathbf{A}$. Thus, $[\mathbf{A} \mid -\mathbf{e}]$ is totally unimodular. □

Now consider the coefficient matrix $\mathbf{C}$ of the constraints in LP2. Rearranging constraints and variables in different way results in a distinct matrix for $\mathbf{C}$. However, due to (c) in Lemma 4, total unimodularity is invariant under such operations. Without loss of generality, the rows and columns of $\mathbf{C}$ are assumed to be properly ordered with respect to the format in Figure 2. Let $\mathbf{e}_{i,n}$ denote a standard unit vector of size n with i-th entry being 1. Suppose that a given $I_1 \times I_2$ weight matrix $\mathbf{W}$ has negative entries $\mathbf{W}_{i_1,j_1}$, $\mathbf{W}_{i_2,j_2}$, $\ldots$, and $\mathbf{W}_{i_k,j_k}$ for some k, $1 \leq k \leq I_1 I_2$. Thus, $\mathbf{C}$ can be partitioned into

$$\mathbf{C} = \left[\begin{array}{c|c|c} \mathbf{e}_{i_1,I_1}^T & \mathbf{e}_{j_1,I_2}^T & -\mathbf{e}_{1,k}^T \\ \mathbf{e}_{i_2,I_1}^T & \mathbf{e}_{j_2,I_2}^T & -\mathbf{e}_{2,k}^T \\ \vdots & \vdots & \vdots \\ \mathbf{e}_{i_k,I_1}^T & \mathbf{e}_{j_k,I_2}^T & -\mathbf{e}_{k,k}^T \end{array}\right] = [\mathbf{X}_1 \,|\, \mathbf{X}_2 \,|\, -\mathbf{I}_k],$$

where $\mathbf{X}_1 \in \{0, 1\}^{k \times I_1}$, $\mathbf{X}_2 \in \{0, 1\}^{k \times I_2}$. By convention, the constraints of variable bounds, e.g., (8) and (9) in Figure 2, are not included in a coefficient matrix.

LEMMA 5. *The coefficient matrix of constraints in LP2 is totally unimodular.*

PROOF. Let $\mathbf{X}$ be the matrix $[\mathbf{X}_1|\mathbf{X}_2]$, where $\mathbf{X}_1$ and $\mathbf{X}_2$ are submatrices defined for $\mathbf{C}$. Its transpose $\mathbf{X}^T$ is totally unimodular since it satisfies the total unimodularity condition in Sufficient Condition 1. By Lemma 4, $\mathbf{X}$ is totally unimodular; so does the matrix $[\mathbf{X}| - \mathbf{I}_k] = \mathbf{C}$. □

To characterize an optimal solution by simplex algorithms, consider the following. A *polyhedron* is a solution set of some finite number of linear equalities and inequalities. A bounded polyhedron is called a *polytope*. Let $S \subseteq \{0, 1\}^n$ be the feasible solution set for some problem instance of the ILP in Figure 1. The *convex hull* of S is the smallest convex set containing S. Since S is a finite set, the convex hull is a polytope; that is, the feasible region of LP1 is a polytope; so does LP2. A point $\mathbf{x}$ in a polytope P is an *extreme point* if and only if $\mathbf{x}$ is not a convex combination of any pair of distinct points in P. If an LP has optimal solutions, then one of the optimal solutions must be an extreme point due to linear property. Simplex algorithms are based on repeatedly searching for a better extreme point on a polyhedron. Suppose that a polytope P is defined by

$$\left\{\mathbf{x} \in \mathbb{R}^{I_2} \mid \mathbf{b}_l \leq \mathbf{A}\mathbf{x} \leq \mathbf{b}_u \text{ and } \mathbf{c}_l \leq \mathbf{x} \leq \mathbf{c}_u\right\},$$

where $\mathbf{A} \in \mathbb{R}^{I_1 \times I_2}$, $\mathbf{b}_l \in \mathbb{R}^{I_1}$, $\mathbf{b}_u \in \mathbb{R}^{I_1}$, $\mathbf{c}_l \in \mathbb{R}^{I_2}$, and $\mathbf{c}_u \in \mathbb{R}^{I_2}$. It has been shown that the coordinates of every extreme point of P are integral if the following holds [7].

SUFFICIENT CONDITION 2 (INTEGRAL EXTREME POINTS). *The coefficient matrix $\mathbf{A}$ is totally unimodular, and the vectors $\mathbf{b}_l$, $\mathbf{b}_u$, $\mathbf{c}_l$, and $\mathbf{c}_u$ are integral.*

PROOF OF **Theorem 3**. For a given matrix of all 1's, LP2 finds vectors of all 1's, due to empty set of constraints (7) in this case. For other cases, the coefficient matrix of LP2 is totally unimodular, by Lemma 5. All the constants involved in the constraints of LP2 are integral. This also includes an implicit lower bound 0 for each constraint (7) of LP2. Thus, the Sufficient Condition 2 is satisfied. It follows that every extreme point of a polytope for LP2 is integral. More specifically, each of those integers is either 0 or 1 due to the bounds on variables in LP2. The feasible region of an instance for LP2 is a polytope, i.e., a bounded region. A simplex algorithm always finds an extreme point in the optima if the LP is feasible and bounded. □

We assume that an optimal solution for LP2 hereinafter has this binary property.

3.4 Analysis on Solution Quality

The main result of this section is summarized in the following theorem.

THEOREM 6. *Let $\mathbf{x}_1\mathbf{x}_2^T$ be the binary rank-one approximation by the algorithm LPIT for a binary matrix $\mathbf{A}$. The Hamming distance $\left\|\mathbf{A}-\mathbf{x}_1\mathbf{x}_2^T\right\|_F^2 \le 2\times\min_{\mathbf{p},\mathbf{q}}\left\|\mathbf{A}-\mathbf{p}\mathbf{q}^T\right\|_F^2$ for binary vectors $\mathbf{p}$ and $\mathbf{q}$.*

The corollary below follows directly from this theorem.

COROLLARY 7. *For a given rank-one binary matrix $\mathbf{A}$, the algorithm LPIT finds a pair of vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ such that $\mathbf{x}_1\mathbf{x}_2^T=\mathbf{A}$.*

Theorem 6 shows an error-bound on solutions by LPIT. Several definitions, relevant to its proof, are defined as follows:

DEFINITION 1. *Given an optimal solution*

$$S=\left(\{x_{11},\ldots,x_{1I_1}\},\{x_{21},\ldots,x_{2I_2}\},\{z_{i,j}\}_{\mathbf{W}_{i,j}=-1}\right)$$

for LP2 by a simplex algorithm with an input matrix $\mathbf{A}\in\{0,1\}^{I_1\times I_2}$, define the following:
1. $W_1=\frac{1}{2}\left|\{\mathbf{W}_{i,j}\,|\,x_{1i}+x_{2j}=1,\mathbf{W}_{i,j}=1\}\right|$,
2. $W_2=\left|\{\mathbf{W}_{i,j}\,|\,x_{1i}+x_{2j}=2,\mathbf{W}_{i,j}=1\}\right|$,
3. $W_{(-)}=-\left|\{\mathbf{W}_{i,j}\,|\,x_{1i}+x_{2j}=2,\mathbf{W}_{i,j}=-1\}\right|$,
4. $W_{opt}=$ The objective value of S for LP2, and
5. $W_{app}=$ The objective value of a solution for MWP by the algorithm LPIT for the matrix $\mathbf{A}$.

LEMMA 8. *$W_{opt}=W_1+W_2+W_{(-)}$.*

PROOF. Note that variable $z_{i,j}=1$ for $\mathbf{W}_{i,j}=-1$ if and only if $x_{1i}+x_{2j}=2$ at optimality. By Theorem 3, every variable in the solution equals 0 or 1. Using the objective function of LP2, W_{opt} can be expressed, by Definition 1, as

$$\begin{aligned}W_{\text{opt}}&=\frac{1}{2}\sum_{\mathbf{W}_{i,j}=1}(x_{1i}+x_{2j})-\sum_{\mathbf{W}_{i,j}=-1}z_{i,j}\\&=\frac{1}{2}\sum_{\substack{\mathbf{W}_{i,j}=1\\x_{1i}+x_{2j}=1}}1+\frac{1}{2}\sum_{\substack{\mathbf{W}_{i,j}=1\\x_{1i}+x_{2j}=2}}2-\sum_{\substack{\mathbf{W}_{i,j}=-1\\x_{1i}+x_{2j}=2}}1\\&=W_1+W_2+W_{(-)}.\end{aligned}$$

□

LEMMA 9. *$W_{app}\ge W_2+W_{(-)}$.*

PROOF. Let $\mathbf{x}_1$ and $\mathbf{x}_2$ be the two vectors found in Phase One of the algorithm LPIT. The objective value is $\mathbf{x}_1^T\mathbf{W}\mathbf{x}_2$ by equation (2). After the completion of LPIT, we have $W_{\text{app}}\ge\mathbf{x}_1^T\mathbf{W}\mathbf{x}_2$ due to the iterative procedure in Phase Two. Moreover, by the definition of MWP, the value can be expressed, by Definition 1, as

$$\begin{aligned}\mathbf{x}_1^T\mathbf{W}\mathbf{x}_2&=\sum_{i,j}\mathbf{W}_{i,j}x_{1i}x_{2j}=\sum_{i,j:\,x_{1i}+x_{2j}=2}\mathbf{W}_{i,j}\\&=\sum_{\substack{\mathbf{W}_{i,j}=1\\x_{1i}+x_{2j}=2}}1-\sum_{\substack{\mathbf{W}_{i,j}=-1\\x_{1i}+x_{2j}=2}}1=W_2+W_{(-)}.\end{aligned}$$

□

LEMMA 10. *$W_1\le\frac{1}{2}\left(\|\mathbf{A}\|_F^2-W_2\right)$, where $\mathbf{A}$ is a binary input matrix for the algorithm LPIT.*

PROOF. Suppose that $\left(\{x_{11},\ldots,x_{1I_1}\},\{x_{21},\ldots,x_{2I_2}\},\{z_{i,j}\}_{\mathbf{W}_{i,j}=-1}\right)$ is an optimal solution for an instance of LP2 with weight matrix $\mathbf{W}$ of $\mathbf{A}$. Since every entry in $\mathbf{A}$ is binary and $\mathbf{A}_{i,j}=1$ if and only if $\mathbf{W}_{i,j}=1$, by Definition 1, it follows that

$$\begin{aligned}\|\mathbf{A}\|_F^2&=\left|\{\mathbf{W}_{i,j}\,|\,\mathbf{W}_{i,j}=1\}\right|\\&\ge\left|\{\mathbf{W}_{i,j}\,|\,x_{1i}+x_{2j}=1,\mathbf{W}_{i,j}=1\}\right|\\&\quad+\left|\{\mathbf{W}_{i,j}\,|\,x_{1i}+x_{2j}=2,\mathbf{W}_{i,j}=1\}\right|\\&=2\mathbf{W}_1+\mathbf{W}_2.\end{aligned}$$

□

PROOF OF **Theorem 6**. The Hamming distance by an optimal approximation for $\mathbf{A}$ is

$$\begin{aligned}&\min_{\mathbf{p},\mathbf{q}}\left\|\mathbf{A}-\mathbf{p}\mathbf{q}^T\right\|_F^2\\&=\min_{\mathbf{p},\mathbf{q}}\left(\|\mathbf{A}\|_F^2-\mathbf{p}^T\mathbf{W}\mathbf{q}\right)&&\text{by (2)}\\&\ge\|\mathbf{A}\|_F^2-W_{\text{opt}}&&\text{by Proposition 2}\\&=\|\mathbf{A}\|_F^2-W_2-W_{(-)}-W_1&&\text{by Lemma 8}\\&\ge\|\mathbf{A}\|_F^2-W_2-W_{(-)}-\frac{1}{2}\left(\|\mathbf{A}\|_F^2-W_2\right)&&\text{by Lemma 10}\\&=\frac{1}{2}\left(\|\mathbf{A}\|_F^2-W_2-W_{(-)}\right)-\frac{1}{2}W_{(-)}\\&\ge\frac{1}{2}\left(\|\mathbf{A}\|_F^2-W_2-W_{(-)}\right)&&\text{by } W_{(-)}\le 0\\&\ge\frac{1}{2}\left(\|\mathbf{A}\|_F^2-W_{\text{app}}\right)&&\text{by Lemma 9}\\&=\frac{1}{2}\left\|\mathbf{A}-\mathbf{x}_1\mathbf{x}_2^T\right\|_F^2&&\text{by (2).}\end{aligned}$$

□

4. REGULARIZATION

Regularization is one of key techniques underlying many well-known data mining and machine learning methods, including support vector machines (SVM) and ridge regression. In the regularization framework, the smoothness of solution functions is typically optimized along with the error term and a regularization parameter is used to control the tradeoff between them. This generally results in the following objective function: $f=f_{\text{err}}+\lambda f_{\text{reg}}$, where f_{err}, f_{reg}, and λ are the original error function, regularizer, and regularization parameter, respectively. Specifically, we propose to solve the following optimization problem:

$$\min_{\mathbf{x}_1,\mathbf{x}_2}\left(\left\|\mathbf{A}-\mathbf{x}_1\mathbf{x}_2^T\right\|_F^2+\lambda\left\|\mathbf{x}_1\right\|_2^2\left\|\mathbf{x}_2\right\|_2^2\right),\qquad(11)$$

where $\mathbf{A}\in\{0,1\}^{I_1\times I_2}$, $\mathbf{x}_1\in\{0,1\}^{I_1}$, and $\mathbf{x}_2\in\{0,1\}^{I_2}$. The value $\left\|\mathbf{A}-\mathbf{x}_1\mathbf{x}_2^T\right\|_F^2$ in (11) is the total number of mismatches between the given matrix $\mathbf{A}$ and the solution matrix $\mathbf{x}_1\mathbf{x}_2^T$. The nonnegative regularization parameter λ in (11) controls the balance between the approximation error and the solution complexity. The problem at (1) is a special case of (11) when $\lambda=0$.

It follows from (2) that an optimal solution to (11) can be found by solving the following maximization problem:

$$\max_{\mathbf{x}_1,\mathbf{x}_2}\left(\mathbf{x}_1^T\mathbf{U}\mathbf{x}_2\right)\equiv\max_{\mathbf{x}_1,\mathbf{x}_2}\left(\mathbf{x}_1^T\mathbf{W}\mathbf{x}_2-\lambda\left\|\mathbf{x}_1\right\|_2^2\left\|\mathbf{x}_2\right\|_2^2\right),\quad(12)$$

where $\mathbf{U} \in \{-(1+\lambda), 1-\lambda\}^{I_1 \times I_2}$ is called the *λ-regularized weight* (λ-RW) *matrix* of $\mathbf{A}$ in this paper and is defined as $\mathbf{U}_{i,j} = 1-\lambda$, if $\mathbf{A}_{i,j} = 1$, and $\mathbf{U}_{i,j} = -(1+\lambda)$, otherwise. The equivalent relation in (12) is due to the equality below, with $\mathbf{1}$ being a vector of all 1's:

$$\begin{aligned}\mathbf{x}_1^T\mathbf{W}\mathbf{x}_2 - \lambda \|\mathbf{x}_1\|_2^2 \|\mathbf{x}_2\|_2^2 &= \mathbf{x}_1^T\mathbf{W}\mathbf{x}_2 - \lambda \mathbf{x}_1^T\mathbf{1}\mathbf{1}^T\mathbf{x}_2 \\ &= \mathbf{x}_1^T\left(\mathbf{W} - \lambda\mathbf{1}\mathbf{1}^T\right)\mathbf{x}_2 = \mathbf{x}_1^T\mathbf{U}\mathbf{x}_2.\end{aligned}$$

Next, we present the proposed methods for exact and error-bounded solutions for regularized rank-one approximation at (11). Our methodology is similar to the one in Section 3: the exact solutions can be found by integer linear programming; a relaxed integer program, called *ILP2*, is proposed for error-bounded solutions.

4.1 Exact and Relaxed Formulations

A 0-1 integer linear program, called *ILP1*, for exact solutions of the problem at (12) is given in Figure 3. There are three types of binary variables x_{1i}, x_{2j}, and $z_{i,j}$ in the ILP1 for $1 \le i \le I_1$ and $1 \le j \le I_2$. Variables x_{1i} and x_{2j} represent the i-th and j-th entries of solution vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ respectively. Variable $z_{i,j}$ is an auxiliary variable whose value is 1, if and only if both the values of x_{1i} and x_{2j} are 1. The objective function at (12) can therefore be rewritten as $\sum_{i,j=1}^{I_1,I_2} \mathbf{U}_{i,j} z_{i,j}$ as in Figure 3.

$$\begin{aligned}&\text{Maximize} \quad \sum_{i,j=1}^{I_1,I_2} \mathbf{U}_{i,j} z_{i,j}, \\ &\text{subject to} \\ &-x_{1i} - x_{2j} + 2z_{i,j} \le 0 \qquad \text{for } \mathbf{A}_{i,j} = 1, \qquad (13)\\ &x_{1i} + x_{2j} - z_{i,j} \le 1 \qquad \text{for } \mathbf{A}_{i,j} = 0, \text{ and} \qquad (14)\\ &x_{1i}, x_{2j}, z_{i,j} \in \{0,1\} \qquad \forall i,j. \qquad (15)\end{aligned}$$

Figure 3: ILP1: The exact formulation.

Constraints (13) and (14) enforce the relationship between $z_{i,j}$ and the pair of x_{1i} and x_{2j} for all the i and j. Specifically, for any pair of i and j, $z_{i,j} = 1$ if and only if $x_{1i} = 1$ and $x_{2j} = 1$ at optimality.

$$\begin{aligned}&\text{Maximize } \frac{1-\lambda}{2} \sum_{\mathbf{A}_{i,j}=1} (x_{1i} + x_{2j}) - (1+\lambda) \sum_{\mathbf{A}_{i,j}=0} z_{i,j}, \\ &\text{subject to} \\ &x_{1i} + x_{2j} - z_{i,j} \le 1 \qquad \text{for } \mathbf{A}_{i,j} = 0, \qquad (16)\\ &x_{1i}, x_{2j} \in \{0,1\} \qquad \text{for } \forall i,j, \text{ and} \qquad (17)\\ &z_{i,j} \in \{0,1\} \qquad \text{for } \mathbf{A}_{i,j} = 0. \qquad (18)\end{aligned}$$

Figure 4: ILP2: The relaxed formulation.

The corresponding variable $z_{i,j}$ for each positive $\mathbf{A}_{i,j}$ is bounded above by $\frac{1}{2}(x_{1i}+x_{2j})$ at (13). Replacing every such $z_{i,j}$ by its upper bound $\frac{1}{2}(x_{1i}+x_{2j})$ results in an ILP whose optimal value is not less than that of the ILP1. Figure 4 shows this resulting formulation ILP2. The constraint for $\mathbf{A}_{i,j} = 1$ becomes redundant and is removed from Figure 4. Since removing or relaxing a constraint does not make an objective value of the same instance worse, the proposition below follows:

PROPOSITION 11. *The optimal value at* (12) *is no more than that of ILP2 for the same problem instance.*

The main result of the error bounded approximation is given as Theorem 12 (the proof follows similar techniques in Section 3 and is omitted).

THEOREM 12. *Let $\mathbf{x}_1\mathbf{x}_2^T$ be the solution matrix for regularized rank-one approximation found by ILP2 for a binary matrix $\mathbf{A}$ and a regularization parameter $\lambda \in \mathbb{R}_+$. The cost $\left\|\mathbf{A} - \mathbf{x}_1\mathbf{x}_2^T\right\|_F^2 + \lambda \|\mathbf{x}_1\|_2^2 \|\mathbf{x}_2\|_2^2$ is bounded above by $\frac{2}{1+\min\{1,\lambda\}} \times \min_{\mathbf{p},\mathbf{q}} \left(\|\mathbf{A} - \mathbf{p}\mathbf{q}^T\|_F^2 + \lambda \|\mathbf{p}\|_2^2 \|\mathbf{q}\|_2^2\right)$ for binary vectors $\mathbf{p}$ and $\mathbf{q}$.*

Similar to ILP1 in Section 3, we relax the integral constraint of ILP2, i.e., x_{1i}, x_{2i}, $z_{i,j} \in [0,1]$, and solve its corresponding linear program (LP). The solution found by LP relaxation of ILP2 is feasible for ILP2 mainly because the coefficient matrix of ILP2 is *totally unimodular* [5]. That is, we essentially get integral x_{1i}, x_{2i}, $z_{i,j} \in \{0,1\}$ for "free" by solving the LP relaxation.

The algorithm, called **rLPIT** for (regularized) binary rank-one approximation is similar to **LPIT** and it consists of two phases: (*i*) find an error-bounded solution, and (*ii*) apply the iterative procedure [13] to further improve the solution.

5. EFFICIENT APPROXIMATION

In Sections 3 and 4, we compute the binary matrix approximation by solving liner programs. However, solving the LP for a large matrix is still computationally expensive, e.g., too many variables. In this section, we transform the problem into an instance of minimum s-t cut problem, which can be solved efficiently by finding maximum flows.

5.1 Approximation by Minimum s-t Cuts

Consider the *Generalized Independent Set Problem* (GIS) [6]: The input of GIS is (*i*) an undirected graph $G = (V, E)$, (*ii*) a nonnegative weight $w(v)$ for each vertex $v \in V$, and (*iii*) a nonnegative penalty $p(e)$ for each edge $e \in E$. The problem of GIS is to find a vertex subset $S \subseteq V$ so as to maximize the gain: $\sum_{v \in S} w(v) - \sum_{v_1,v_2 \in S, \{v_1,v_2\} \in E} p(v_1, v_2)$.

Observation. ILP2 in Figure 4 defines an instance for GIS, and the corresponding graph is *bipartite*:

1. Every vertex x_{1i} is in the same partite while a vertex x_{2j} is in the other. There is an edge between x_{1i} and x_{2j} if $\mathbf{A}_{i,j} = 0$.
2. The weight of a vertex x_{1i} (or x_{2j}) is $\frac{1-\lambda}{2} \sum_j \mathbf{A}_{i,j}$ (or $\frac{1-\lambda}{2} \sum_i \mathbf{A}_{i,j}$).
3. By constraint (16), $z_{i,j} = 1$ if and only if $x_{1i} = 1$ and $x_{2j} = 1$ at optimality. Thus, the variable $z_{i,j}$ indicates whether both the endpoints of an edge $\{x_{1i}, x_{2j}\}$ are in the solution set.
4. The penalty of each edge $\{x_{1i}, x_{2j}\}$ is $1 + \lambda$.
5. The values of entries in $\mathbf{x}_1$ and $\mathbf{x}_2$ of a solution for ILP2 indicate the solution set $S \in V$ of vertices for GIS.

It follows that the objective function of this instance for GIS is the same as the one in Figure 4. GIS is at least as difficult as *maximum independent set problem*; consequently,

GIS is NP-hard. However, it has been shown that GIS for *bipartite graphs* can be solved exactly in polynomial time by transformation [6] to an instance of minimum st-cut problem. Techniques in [6] implies the following transformation from problem at (11) to the maximum flow problem: Given $\mathbf{A} \in \{0,1\}^{I_1 \times I_2}$ and $\lambda \in \mathbb{R}_+$, the input is reduced to a network $G = (V, E)$, a source $s \in V$, a destination $t \in V$, and a capacity $c(e)$ for each link $e \in E$ as follows:

Nodes: Create (i) a node set V_1 of size I_1 for rows in $\mathbf{A}$, (ii) a node set V_2 of size I_2 for columns of $\mathbf{A}$, and (iii) a node pair s and t. Let the node set $V = V_1 \cup V_2 \cup \{s, t\}$, where s and t are the source and destination nodes respectively.

Links: For each node pair $v_i \in V_1$ and $v_j \in V_2$, create a link $(i, j) \in E$ if $\mathbf{A}_{i,j} = 0$. Also create (i) a link $(s, v_i) \in E$ for each $v_i \in V_1$ and (ii) a link $(v_j, t) \in E$ for each $v_j \in V_2$.

Capacities: Denote $s_{\text{row},i}$ and $s_{\text{col},j}$ as the sums of the row i and column j of $\mathbf{A}$ respectively. (i) For link (s, v_i), $v_i \in V_1$, let $c(s, v_i) = \frac{1-\lambda}{2} \times s_{\text{row},i}$. (ii) For link (v_j, t), $v_j \in V_2$, let $c(v_j, t) = \frac{1-\lambda}{2} \times s_{\text{col},j}$. (iii) For link $(v_i, v_j) \in E$, where $v_i \in V_1$ and $v_j \in V_2$, let $c(v_i, v_j) = 1 + \lambda$.

For example, Figure 5(c) depicts the network from the input matrix in Figure 5(a) and $\lambda = \frac{1}{2}$.

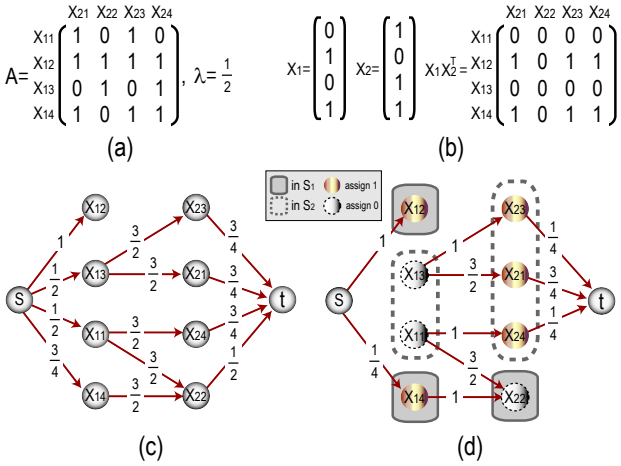


Figure 5: (a) A matrix consists of 4 data points and a regularization parameter 0.5. (b) The solution found by the minimum s-t cut. (c) The network transformed from the input in (a). (d) The residual network showing the cut and its assignment.

The network can be solved efficiently by any algorithm for finding maximum flows. The result partitions the node set $V \setminus \{s, t\}$ into two sets S_1 and S_2, where the destination t is reachable from every node $v_2 \in S_2$ in the final residual network but not for any node $v_1 \in S_1$ (page 928, Goldberg & Tarjan [3]). Figure 5(d) shows the residual network as a result of applying a maximum flow algorithm on the network in Figure 5(c).

The solution vectors $\mathbf{x}_1 \in \{0,1\}^{I_1}$ and $\mathbf{x}_2 \in \{0,1\}^{I_2}$ for ILP2 is obtained from the two sets S_1 and S_2 as follows:

1. The value of $\mathbf{x}_{1i}$ is 1 if and only if its corresponding node $v_i \in V_1$ is in S_1.
2. The value of $\mathbf{x}_{2j}$ is 1 if and only if its corresponding node $v_j \in V_2$ is in S_2.
3. All the other entries have values of zero.

Figure 5(b) shows the corresponding assignments and the solution for the example.

6. EXPERIMENTS

In this section, we present experimental results to evaluate the proposed algorithms. We present comparisons for the results by the minimum s-t cut in Phase 1 (P1) of the **rLPIT** algorithm, the improvement by the iterative process in Phase (P2) of **rLPIT**, theoretical upper bounds, and running time of regularized rank-one approximations by the minimum s-t cut. All algorithms are implemented using `C++`.

6.1 Experimental Setup

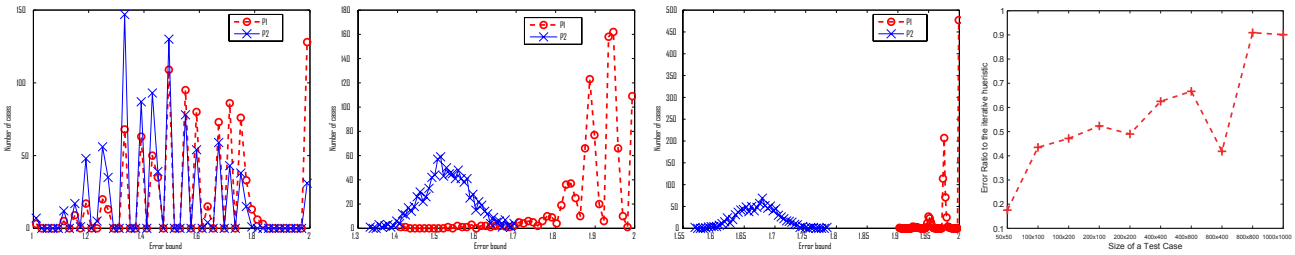
We use both the synthetic and real data in the experiments. We collected 3000 *Drosophila* gene expression pattern images of lateral view from three early developmental stage ranges, i.e., 1-3, 4-6, 7-8, from the FlyExpress database[1]. The images from this database had gone through labor intensive segmentation, alignment, and enhancement. The collection contains 128×320 black-and-white images of staining patterns and were subsampled down to 64×160 to reduce the computational cost for the purpose of this experiment. Each image is represented as a vector of length 10240. We randomly generate binary matrices with the density of non-zeros around 30%, which is close to the average density of gene pattern expression images. We present the cases up to 1000×1000 in size, and 1000 cases for distinct size for non-regularization. For regularization, we varied the parameter λ from 0.0 up to 1.0 with an increment of 0.1. The optimal objective values for ILP2 were used to provide bounds for our test cases, since the optimal value of (12) is no more than that of ILP2 as summarized in Proposition 11. For example, if the ratio by the minimum s-t cut to ILP2 is 1.5, then the exact ratio is 1.5 or less.

To figure out error ratios in acceptable time, i.e., avoiding integer programming, the linear programming (LP) relaxation for ILP2 were solved, e.g., $x_{1i}, x_{2j}, z_{i,j} \in [0, 1]$. Note that an optimal solution for the LP is also an optimal solution for ILP2, mainly due to the total unimodularity of ILP2's coefficient matrix, assuming a simplex method for the LP is applied. The implementation was coded in `C++` with the LP solvers *lpsolve*[2], which has a simplex-base algorithm. For large test cases, solving the corresponding LP's becomes a challenge. For example, there are one million variables or more for the size of 1000×1000. For such cases, the errors are compared with iterative heuristic, e.g., with random initial vectors proposed in [14].

6.2 Results on Rank-one Approximation without Regularization

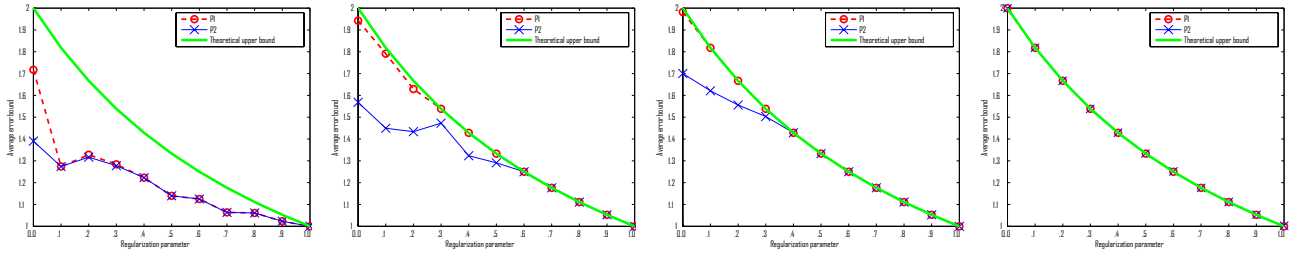
Each chart of Figure 6 shows the histogram of an error-bound distribution for a given set of fixed-size test cases. The bound is a ratio of the Hamming distance by solution from a minimum s-t cut to the objective value of ILP2. In a case that a distance by ILP2 is zero, the ratio is 1 because the distance by a minimum s-t cut is also zero for all the test

[1] `http://www.flyexpress.net`
[2] `http://sourceforge.net/projects/lpsolve`



(a) 5×5 test cases (b) 20×20 test cases (c) 50×50 test cases (d) Mixed test cases

Figure 6: Distributions of error bounds for the test cases without regularization. P1 and P2 correspond to the first and second phases of LPIT, respectively. P2 improves the result of P1 by the iterative procedure.



(a) 5×5 test cases (b) 20×20 test cases (c) 50×50 test cases (d) 400×100 test cases

Figure 7: Parameterized error bounds for the test cases with regularization. P1 and P2 correspond to the first and second phases of rLPIT, respectively. P2 improves the result of P1 by the iterative procedure.

cases. All the figures indicate that none of the test cases has any ratio exceeding 2. This is consistent with the result in Theorem 12.

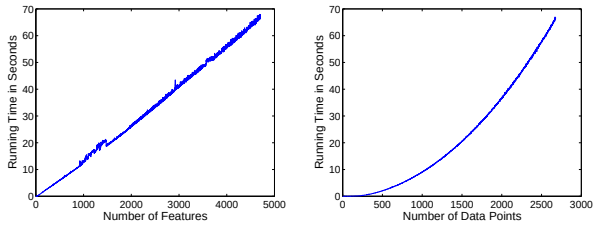
Figure 6(a) depicts that the iterative procedure has negligible improvement on the solution quality in this case. Comparing this result with those of Figures 6(b) and 6(c), we found that the role of iterative procedure is getting important as the input matrix gets complex. This is due to the lack of dominant patterns in randomly generated large matrices. In all those test cases, the mean of error bounds is steady in the range of 1.45 to 1.70 by the algorithm. The vector pair found by the minimum s-t cut serves good starting points for the iterative procedure in those test cases. Figure 6(d) shows the error ratios of the algorithm to the iterative heuristic given in [15].

6.3 Results on Rank-one Approximation with Regularization

Figure 7 shows the solution quality with varying regularization parameters for the test cases. In all those figures, none of the test cases has any error ratio exceeding the theoretical bound $\frac{2}{1+\lambda}$, $0 \leq \lambda \leq 1$. The bounds are tight for some of the test cases. For highly complex matrices, the proposed algorithm obtains solutions with errors close to theoretical bounds, e.g., the results in Figures 7(b) and 7(c). The improvement by the iterative procedure appears less persistent for regularization. This phenomenon is likely due to shorter Hamming distances of optimal solutions as penalty gets very high for complex solutions.

6.4 Running Time Comparison

We use the gene expression images for evaluating the running time of finding error-bounded approximations using



(a) Change in features (b) Change in data points

Figure 8: Results of running time tests.

minimum s-t cuts. We used the implementation of a maximum flow algorithm given in [1], which is reported to have good average performance in the context of image applications. In our test, there were two scenarios, each we fixed one of the dimensions while incrementally changed the other. Any of the fixed dimensions has a size of 1000. The regularization parameter was set to 0.5.

Figure 8(a) shows the running time for increasing the number of features. The running time is linear with respect to the change for our test cases. Figure 8(b) shows the running time for increasing the number of data points. It is clear that solving a matrix with a higher dimension in data points is more costly by the algorithms, and such cost gets higher faster. However, we can always improve the performance by solving the transpose of such a matrix or simply switching the source and destination sides of the network.

6.5 Hierarchical Tree Visualization

A total of three trees, each for a stage range, were constructed. Each tree has 1000 leaf nodes, the number of input

images for a stage range. Extracted patterns of a tree were represented as an image. To illustrate the typical traits of trees built by the proposed approximation algorithm, Figure 9 shows a result for a set of 40 input images from stage range 4-6. We observe that the shape of a tree relies on the regularization parameter λ and the degree of variation of the input images. When $\lambda = 0$, i.e., without regularization as in [13, 15], both $\mathbf{x}_1$ and $\mathbf{x}_2$ tend to be either all zeros or all ones, which is undesirable for tree construction. In our experiments, we set $\lambda = 0.4$. We can observe from the figure that images from the same branch share similar patterns.

7. CONCLUSION

In this paper, we study the problem of computing rank-one binary matrix approximations, which have been previously used for subsampling, compression, and clustering. Specifically, we reformulate the rank-one binary matrix approximation problem as a maximum weight problem, based on which we propose a linear program (LP) formulation, which is shown to achieve a guaranteed approximation error bound. We further extend the proposed formulations using the regularization technique. In addition, we show that the proposed approximate formulation can be transformed into an instance of minimum s-t cut problem, which can be solved efficiently by finding maximum flows. Our empirical evaluation shows the efficiency of the proposed algorithm based on minimum s-t cuts. Results also confirm the consistency and tightness of the established theoretical bounds.

As a sample application, we apply the proposed algorithm for the hierarchy construction and pattern discovery for *Drosophila* gene expression pattern images. We plan to examine the biological significance of the resulting hierarchy. We plan to apply the proposed algorithm to other applications involving binary data. We are currently investigating how the solutions of the proposed algorithm depend on the regularization parameter as well as its estimation.

Acknowledgments

This work was supported by NSF IIS-0612069, IIS-0812551, CCF-0811790, NIH R01-HG002516, and NGA HM1582-08-1-0016.

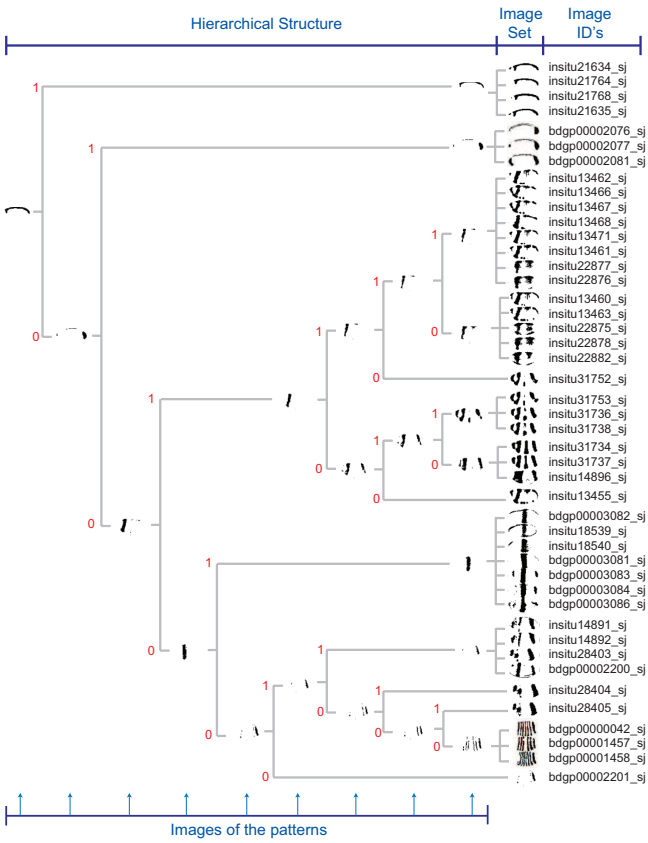


Figure 9: The hierarchy and patterns for a set of 40 images from developmental stage range 4-6.

8. REFERENCES

[1] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE TPAMI*, Sep 2004.

[2] G. Dantzig. Maximization of a linear function of variables subject to linear inequalities. In *Activity Analysis of Production and Allocation*. Wiley, 1951.

[3] A. V. Goldberg and R. E. Tarjan. A new approach to the maximum-flow problem. *J. ACM*, 35(4):921–940, 1988.

[4] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, MD, USA, 3rd edition, 1996.

[5] I. Heller and C. B. Tompkins. An extension of a theorem of Dantzig's. Ann. of Math. Stud., no. 38, pages 247–254. 1956.

[6] D. S. Hochbaum and A. Pathria. Forest harvesting and minimum cuts: a new approach to handling spatial constraints. *Forest Science*, 43(4):544–554, 1997.

[7] A. J. Hoffman and J. B. Kruskal. Integral boundary points of convex polyhedra. Annals of Mathematics Studies, no. 38, pages 223–246. Princeton University Press, 1956.

[8] I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, New York, 1986.

[9] J. A. Kelner and D. A. Spielman. A randomized polynomial-time simplex algorithm for linear programming. In *ACM STOC 2006*, pages 51–60, 2006.

[10] L. G. Khachiyan. A polynomial algorithm in linear programming [in russian]. *Doklady Akademii Nauk SSSR*, 244:1093–1096, 1979. English translation: Soviet Mathematics Doklady 20 (1979), 191-194.

[11] V. Klee and G. J. Minty. How good is the simplex algorithm? In *Inequalities, III*, pages 159–175. Academic Press, New York, 1972.

[12] T. G. Kolda and D. P. O'Leary. A semidiscrete matrix decomposition for latent semantic indexing information retrieval. *ACM Trans. Inf. Syst.*, 16(4):322–346, 1998.

[13] M. Koyutürk and A. Grama. PROXIMUS: a framework for analyzing very high dimensional discrete-attributed datasets. In *ACM SIGKDD*, pages 147–156, 2003.

[14] M. Koyutürk, A. Grama, and N. Ramakrishnan. Compression, clustering, and pattern discovery in very high-dimensional discrete-attribute data sets. *IEEE TKDE*, 17(4):447– 461, April 2005.

[15] M. Koyuturk, A. Grama, and N. Ramakrishnan. Nonorthogonal decomposition of binary matrices for bounded-error data compression and analysis. *ACM Trans. Math. Softw.*, 32(1):33–69, 2006.

[16] K. Nishino, Y. Sato, and K. Ikeuchi. Eigen-texture method: appearance compression based on 3d model. In *IEEE CVPR 1999*, pages 618–624, 1999.

[17] D. A. Spielman and S.-H. Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *Journal of ACM*, 51(3):385–463, 2004.

[18] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.