Monochromatic Bi-Clustering

Sharon Wulff SHARON.WULFF@INF.ETHZ.CH
Department of Computer Science, ETH Zurich, Switzerland

Ruth Urner RURNER@CS.UWATERLOO.CA
Shai Ben-David SHAI@CS.UWATERLOO.CA
D.R.C. School of Computer Science, University of Waterloo, Canada, ON, N2L 3G1

Abstract

We propose a natural cost function for the bi-clustering task, the *monochromatic cost*. This cost function is suitable for detecting meaningful homogeneous bi-clusters based on categorical valued input matrices. Such tasks arise in many applications, such as the analysis of social networks and in systems-biology where researchers try to infer functional grouping of biological agents based on their pairwise interactions. We analyze the computational complexity of the resulting optimization problem. We present a polynomial time approximation algorithm for this bi-clustering task and complement this result by showing that finding (exact) optimal solutions is NP-hard. As far as we know, these are the first positive approximation guarantees and formal NP-hardness results for any bi-clustering optimization problem. In addition, we show that our optimization problem can be efficiently solved by deterministic annealing, yielding a promising heuristic for large problem instances.

1. Introduction

Common clustering tasks take as input a data set and a similarity (or distance) function over it, with the aim of finding a partition of the data into groups of mutually similar elements. Bi-clustering is a variant of this general task, in which the input data comes from two domain sets, and instead of having a distance function over its elements, the input is some relation over these sets. For example, a set of documents and a set of words and the relation indicating the membership of words in the documents. In this setting, the bi-clustering task is to find partitions of each of the two domain sets into groups, such that words in the same group appear in the same groups of documents and documents that share a group are likely to contain words from the same groups of words. Namely, the relation values in each of the resulting blocks (i.e., the product of two groups, one from each domain set) are as homogeneous as possible.

Similarly to clustering, bi-clustering is an unsupervised method for detecting meaningful structure in data. While common clustering tasks require some user-defined notion of similarity between data points, the bi-clustering task we analyze is fully determined by a matrix of an empirically measured pairwise relation (or interaction) between the domain points. The only additional learner's prior knowledge it requires, is the number of row and column groups. In this sense bi-clustering can be viewed as a more "objective" method.

Bi-clustering is far from being a new framework. Biologists apply bi-clustering techniques to detect groups of similar genes based on their gene expression levels over a set of different treatments. In recommender systems it is used to determine groups of similar customers based on the matrix of their preferences for a set of products. Bi-clustering is also applied in text categorization and other diverse data mining settings. In spite of its wide scope of applications, bi-clustering tasks have hardly been analyzed in terms of their computational and sample complexity. Many popular bi-clustering paradigms are only defined through the algorithms used to carry them out. Furthermore, the various tasks that are defined as optimization problems with respect to a clear objective function, lack proven bounds on their computational complexity.

The main contributions of this paper are as fol-

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28. Copyright 2013 by the author(s).

lows: We propose a new cost-function for the bi-clustering problem, the *monochromatic cost.* The monochromatic cost is a natural formalization of the goal of detecting structure in the form of label-homogeneous sub-matrices, (which is shared by most existing bi-clustering works). Unlike many existing cost-functions, the monochromatic cost can easily handle missing values in the input matrix.

We show that the monochromatic cost complies with the minimum description length (MDL) principle, by presenting a natural encoding of the input matrix whose length corresponds to our cost.

We provide a complete theoretical analysis of the optimization problem induced by the monochromatic cost function. We prove that optimizing this cost is NP-hard, as well as design a polynomial time approximation scheme (PTAS) with proven performance guarantees (it is generally assumed that the various existing formulations of bi-clustering are NP-hard, we are not aware of a formal hardness proof for any formulation).

Finally, we show that the problem of minimizing the monochromatic cost can be solved efficiently by a deterministic annealing scheme, leveraging on ideas used to derive the PTAS. We use such a solver to detect homogeneous bi-clusters in a real-world animal-feature dataset. Albeit not having performance guarantees for this scheme, we empirically demonstrate that this solver performs well in recovering the monochromatic objective on synthetic data.

Due to lack of space, this paper contains only a presentation of our results with succinct proof ideas, while all proofs are deferred to the appendix.

2. Related Work

Much of the more practically oriented work on bi-clustering discusses algorithms that lack an explicit objective function as an optimization goal. This line of work is less relevant to our work and we refer the reader to the survey by Tanay et al. (2006) for further details.

The first definition of bi-clustering as an optimization problem over some well defined objective function is probably due to Hartigan (1972). Hartigan considers real valued matrices and proposes several objective functions, including the sum of block's variances. Hartigan also proposes a heuristic for finding a low-cost bi-clustering, however, no guarantees are proven for its performance.

Cheng & Church (2000), were the first to introduce bi-clustering to gene expression analysis. They formally define a cost function, called the *low mean squared residue*, which can be viewed as a variant of Hartigan's minimum variance cost. They propose an iterative greedy search algorithm which may converge to a local minima. Other notable objective functions used in bioinformatics include *loss in mutual information* in (Dhillon et al., 2003) and the *square residue*, (Cho et al., 2004). In all of these papers, neither optimization quality guarantees nor computational complexity bounds are proven.

The Infinite Relational Model (IRM), described in (Kemp & Tenenbaum, 2006; 2008) is a Bayesian model for the automated discovery of structure in complex data. The approach assumes a generative model, in which the entries of the matrix $\boldsymbol{M}$ are generated by a Binomial distribution. The number of bi-clusters is automatically adjusted by a Chinese restaurant process. Unlike the IRM, the work presented here takes a combinatorial view of the problem of detecting homogeneous structure. It is more objective in the sense of not making any assumptions about the generating process. Their approach on the other hand is more general, and is suitable for more types of input data. Their work contains an extensive experimental setup, but it does not provide formal guarantees.

Various studies (Chakrabarti et al., 2004; Papadimitriou et al., 2008; Hirai et al., 2011) propose cost functions that are inspired by the minimum description length (MDL) principle. These cost functions try to minimize a variation of the Shannon entropy, and justify this by its relation to encoding length. However, this relation is of asymptotic nature, it holds only in the limit, when instance sizes go to infinity. Minimizing this measure for a specific input matrix does not necessarily yield an actual short description of the matrix. In contrast to this, the monochromatic cost function does correspond to the length of a compressed representation (see the discussion in 3.3), and can hence be also viewed as implementing the MDL principle. Those studies do not provide theoretical analysis of the approximation quality of their outcome or of the computational complexity of the resulting bi-clustering optimization task [1].

The algorithms and complexity community analyzed the computational complexity of the related problem of correlation clustering (the monochromatic bi-clustering cost can be viewed as a generalization of correlation clustering). The problem of correlation clus-

[1]Both (Chakrabarti et al., 2004) and (Papadimitriou et al., 2008) claim that the task is NP-hard. However, a closer look reveals that their argument is based on intuition and does not yield an actual proof of the claim.

tering with a fixed number of clusters has been studied in (Giotis & Guruswami, 2006), yielding a PTAS for the minimization problem for all k, and a PTAS for the maximization version along with an NP-hardness result. Our approximation algorithm is inspired by the PTAS for Max k-CUT by Goldreich et al. (1998).

3. The Monochromatic Bi-Clustering Cost Function

Given an input matrix over some fixed finite domain D of values, and integers K and L, the monochromatic bi-clustering task is to find a partition of the rows of the matrix into K groups and its columns into L groups, such that the resulting matrix blocks are as homogeneous as possible. We define the cost of a partition as the fraction of matrix entries that reside in blocks in which they are not the majority value entries.

3.1. Motivation

As an example consider gene expression profiling. DNA micro-arrays are designed to simultaneously measure the expression level of many genes within a particular mRNA sample, under various clinical conditions. The result of such an experiment is a matrix with rows corresponding to genes, columns to conditions and entries to the measured expression. Often the genes are not homogeneous under the performed tests, and thus one of the main challenges in the analysis of such data-sets, is to discover local patterns of sets of genes that exhibit coherent expression across subsets of experimental conditions. The expression values are often discretized or even threshold-ed such that the post-processed data has $\{0, 1\}$ entries. The raw micro-array data may contain missing values caused by various factors such as, insufficient resolution, image corruption, or even a systematic robotic failure in the generation process. These missing entries do not carry any information and thus should be disregarded for the purpose of gene grouping. In this work we assign missing entries the symbol $\star$ and design our cost function such that these values do not affect the ranking of the different solutions.

3.2. Formal Definition

Formally, for some finite domain D, let $\boldsymbol{M} \in (D \cup \{\star\})^{m\times n}$ be an input matrix (in sections 5 and 6, we consider a binary domain $D = \{0, 1\}$ for concreteness), and let $R = [m]$ and $C = [n]$ denote the set of $\boldsymbol{M}$'s row indices and column indices respectively. Given integers K and L, let $P = (P_R = \{R_1, \dots R_K\}, P_C = \{C_1, \dots C_L\})$ denote a partition of R into K subsets, and of C into L subsets. We use $\boldsymbol{M}[R', C']$ for a submatrix defined by a subset of rows $R' \subseteq R$ and subset of columns $C' \subseteq C$.

The monochromatic cost of a partition is defined as

$$Mon_{K,L}(\boldsymbol{M}, P) := \frac{1}{mn} \sum_{k\in[K],l\in[L]} \phi(\boldsymbol{M}[R_k, C_l]) \quad (1)$$

where $\phi : \{D \cup \star\}^{s\times t} \to \mathbb{R}$ is a function returning the number of the non-$\star$ entries in a (sub) matrix that differ from the majority, non-$\star$ value. Formally, for a matrix $A \in (D \cup \{\star\})^{s\times t}$, let d_{max} denote the majority non-$\star$ entry in A, then

$$\phi(A) = |\{(i, j) : A[i, j] \neq d_{max} \text{ and } A[i, j] \neq \star\}|$$

The majority value here refers to the most frequent value among the non-$\star$ entries, the majority value does not necessarily occur in more than half of the entries.

We formally define the monochromatic bi-clustering problem as:

Definition 3.1. *Given an input matrix $\boldsymbol{M} \in (D \cup \{\star\})^{m\times n}$, the K, L-MonochromaticBiclustering (K, L-MCBC) problem is finding a partition P of $\boldsymbol{M}$ that minimizes the monochromatic cost.*

We also consider the version of the monochromatic bi-clustering in which instead of penalizing non-homogeneous bi-clusters, we promote bi-cluster homogeneity (or agreement). We refer to this version as the *monochromatic agreement*. The monochromatic agreement of an input matrix $\boldsymbol{M}$ and a partition P is simply $1 - Mon_{K,L}(\boldsymbol{M}, P)$ (naturally a matrix partition that optimizes one of these costs also optimizes the other, but there is a difference when it comes to measuring the approximation ratio of a close-to-optimal partitioning).

3.3. Monochromatic Compression Scheme

One can think of the monochromatic bi-clustering problem as a compression of an input $m \times n$ matrix into a $K \times L$ blocks matrix, typically, $K \ll m$ and $L \ll n$. The compressed representation requires $K \cdot L + m \log(K) + n \log(L)$ bits (the majority label of each block, augmented by the cluster index of each row and each column), instead of the straightforward $m \cdot n$ bit representation. The monochromatic cost is essentially the error (or the information loss) of such a compressed representation. For an error-free compression, we need to add an encoding of the outliers in each block, adding $\sum_{k\in[K],l\in[L]} \phi(\boldsymbol{M}[R_k, C_l]) \log(mn)$ to the length of the encoding. This shows that (for fixed M,K and L) the length of our encoding corresponds to the monochromatic cost.

3.4. Symmetric Version

While the the bi-clustering problem described above models situations in which the input objects come from disparate domains (e.g. movies and viewers or text documents and words), one can also consider a single domain symmetric variant of the problem. In the symmetric problem, the input matrix is a square symmetric matrix, and the output partitions of the rows and columns are required to be identical. This version models the case where the input matrix records pairwise relations among the elements of a single set. The results of this paper apply to both versions. However, for the sake of concreteness, our presentation is in terms of the non-symmetric version. It is interesting to note that *correlation clustering*, as defined by Bansal et al. (2004), while being a clustering rather than a bi-clustering problem, can also be viewed as a special case of symmetric monochromatic bi-clustering

4. NP-Hardness

In this section, we show that the K, L-MCBC problem is computationally hard (NP-hard) for input matrices over $D \cup \{\star\}$. We use a reduction from MaxCut.

Theorem 4.1. *The K, L-MCBC over a domain set D is NP-hard for every finite domain set D of size ≥ 2 and any (K, L) such that $K \geq 2$ and $K \leq L \leq |D|^{K-1}$, or $L \geq 2$ and $L \leq K \leq |D|^{L-1}$.*

Proof Sketch: We first construct a reduction from MaxCut to the $2, 2$-MCBC, and then extend the reduction to larger (fixed) values of K and L. A *cut* in a graph $G = (V, E)$ is a partition of the vertex set into V_1 and V_2. The size of the cut is the number of edges in E that connect vertices from V_1 to vertices from V_2. The decision version of MaxCut is defined as follows:

Input A graph $G = (V, E)$, an integer r.
Question Is there a cut of G of size at least r?

Given instance $(G = (V, E), r)$ of MaxCut we construct an instance $\boldsymbol{M}_G$ of $2, 2$-MCBC and prove that this constitutes a reduction by showing:

Lemma 4.2. *G has a cut of size at least r if and only if $\boldsymbol{M}_G$ has a $2, 2$-bi-clustering of monochromatic cost at most $\frac{2(|E|-r)}{|\boldsymbol{M}_G|}$.*

Finally, we extend the hardness results to the K, L-MCBC problem for larger values of K and L. Given K and L as in the theorem, and an input matrix $\boldsymbol{M}$ to the $2, 2$-MCBC problem, we construct another matrix $\boldsymbol{N}$ such that an optimal K, L-MCBC partitioning of $\boldsymbol{N}$ will induce an optimal $2, 2$-MCBC partitioning of $\boldsymbol{M}$.

Discussion Note that for K and L such that $K > |D|^L$, K, L-MCBC is the same as $|D|^L, L$-MCBC, since, for L column blocks, there are at most $|D|^L$ possible patterns for the rows. Thus, our current reductions, miss only few relevant combinations for K and L. Theorem 4.1 further implies, that over an infinite domain, K, L-MCBC is NP-hard for *all* combinations of $K \geq 2$ and $L \geq 2$. We conjecture that this holds for finite domains as well. Furthermore, our NP-hardness results for fixed K and L imply that the MCBC problem version, where K and L are part of the input, is NP-hard.

NP-hardness for matrices with arbitrary fraction of $\star$ entries Adding blocks of $\star$-entries to a matrix does not make it any easier to find its optimal bi-clustering. Therefore, our hardness result applies to arbitrarily sparse inputs as well. However, for instances with a large fraction of $\star$-entries, approximating the monochromatic agreement is easy: any bi-clustering has relatively low cost, yielding a trivial approximation algorithm for such instances. We thus provide an NP-hardness result for the case of input matrices with only a small fraction of $\star$-entries:

Theorem 4.3. *For any fixed $K \geq 2$, $L \geq 2$, satisfying the condition of Theorem 4.1, and any $\epsilon \geq 0$, the $K+1, L+1$-MCBC problem restricted to instances with at most ϵ-fraction of missing $\star$ entries, is NP-hard.*

We conjecture that the K, L-MCBC is NP-hard even restricted to input matrices without $\star$-entries.

5. Approximation Algorithm With Guarantees

In this section we present a randomized polynomial time approximation scheme (PTAS) for solving the K, L-MCBC problem. Given an accuracy parameter ϵ and confidence threshold δ, for a binary input matrix $\boldsymbol{M} \in \{0, 1, \star\}^{m \times n}$, the algorithm outputs a bi-clustering that, with probability $\geq 1 - \delta$, has agreement score within a multiplicative $(1 - \epsilon)$ factor of the score of the best possible bi-clustering for that matrix. The run time of the algorithm is polynomial in the size of the input matrix, $m \cdot n$.

5.1. Algorithm Overview

There are two key ideas underlying the algorithm. To introduce the first concept, we need to define the monochromatic cost with respect to a pattern. The output of the monochromatic bi-clustering is a partition of the original matrix into $K \times L$ sub-matrices. We define the pattern of the solution as the resulting $K \times L$ binary matrix with entries corresponding to the majority values in each sub-matrix of the partition. One can also consider the reverse operation, where the pattern is fixed in advance, and the cost of a partition is computed with respect to the values defined by the pattern rather than the true majorities emerging from the partition.

Formally, let $\boldsymbol{A} \in \{0,1\}^{K \times L}$ be a pattern matrix, we define the monochromatic cost with respect to $\boldsymbol{A}$ as

$$Mon_{\boldsymbol{A}}(\boldsymbol{M}, P) := \frac{1}{mn} \sum_{k \in [K], l \in [L]} \phi_{\boldsymbol{A}[k,l]}(\boldsymbol{M}[R_k, C_l]) \tag{2}$$

where, for a matrix $\boldsymbol{M}$ and a value $a \in \{0,1\}$, we define

$$\phi_a(\boldsymbol{M}) = |\{(i,j) : \boldsymbol{M}[i,j] \neq a \text{ and } \boldsymbol{M}[i,j] \neq \star\}|$$

Note that in the above definition the parameters K and L are given implicitly by the dimensions of $\boldsymbol{A}$.

Our algorithm iterates over all (discarding symmetries) possible patterns, and for each pattern finds the best (approximated) partition. Naturally, the pattern of the best partition will be considered, and it is easy to see that the partition minimizing the cost with respect to the optimal pattern, is the optimal monochromatic solution.

Claim 5.1. *For a fixed pattern matrix $\boldsymbol{A}$, and a fixed partition P_R of the rows of $\boldsymbol{M}$, it is possible to find the partition of the columns that yields the lowest monochromatic pattern cost, i.e. $\min_{P'_C} Mon_{\boldsymbol{A}}(\boldsymbol{M}, (P_R, P'_C))$, in polynomial time.*

According to claim 5.1, if we knew the optimal partition of the rows, and conditioning on the pattern, we could efficiently recover the optimal partition of the columns. The claim is stated with respect to the rows but holds just the same for the columns

The second component in the derivation of the PTAS, is showing that it is sufficient to know the optimal partition of only a (fixed-size) uniformly sampled subset of the rows, to obtain a partition of the columns which is guaranteed to yield an "ϵ-close" agreement score to the optimal partition, with high probability (over the sample). The main technical part of the proof is showing that a sample size depending only on ϵ and δ (but independent of $|\boldsymbol{M}|$) suffices to guarantee the generation of an ϵ-approximation with high probability.

Putting the two ideas together, the approximation algorithm performs the following steps; For each $K \times L$ pattern matrix $\boldsymbol{A}$, the algorithm randomly picks a sample of the rows and a sample of the columns. For each partition of the rows sample into K groups, it computes the implied partition of the entire set of columns, having the optimal monochromatic cost with respect to $\boldsymbol{A}$. Similarly, for every partition of the columns sample, compute the corresponding optimal partition of the entire set of rows (w.r.t. the pattern $\boldsymbol{A}$). Next, for every resulting partitioning of the rows and every resulting partitioning of the columns, compute the monochromatic cost of the full partition, and, finally, the solution with minimal cost is returned. A pseudo-code of the monochromatic approximation algorithm is given in Algorithm 1.

Algorithm 1 Monochromatic Approximation Algorithm

```
1:  Input: $M \in \{0,1,\star\}^{m \times n}$, $K, L \in \mathbb{N}_{\geq 1}$, $\epsilon, \delta > 0$.
2:  Initialize $t = \frac{1}{2\epsilon^2} \log \frac{KL}{\delta\epsilon}$.
3:  for each pattern matrix $A \in \{0,1\}^{K \times L}$ do
4:      Sample $t$ rows and columns: $R^S, C^S$
5:      for each partition $P^S_R = \{R^S_1, \ldots R^S_K\}$ of $R^S$ and
        $P^S_C = \{C^S_1, \ldots C^S_L\}$ of $C^S$ do
6:          for each row $i \in R$ and column $j \in C$ do
7:              assignment$(i) = \operatorname{argmin}_{1 \leq k \leq K} Err(i, k | A, P^S_C)$
8:              assignment$(j) = \operatorname{argmin}_{1 \leq l \leq L} Err(j, l | A, P^S_R)$
9:          end for
10:         compute the cost of the partition
11:     end for
12: end for
13: Return the partition with the minimal cost
```

The following additional notation is used in the pseudo-code and the analysis of the algorithm. Let $S = (R^S, C^S)$ denote the sample of the rows and columns and $t = |R^S| = |C^S|$. We use $P^S = (P^S_R, P^S_C)$ to denote the partition of the sample, where $P^S_R = \{R^S_1, \ldots R^S_K\}$ and $P^S_C = \{C^S_1, \ldots C^S_L\}$.

Given a partition of the rows $P_R = \{R_1, \ldots, R_K\}$, and a pattern $\boldsymbol{A}$, we define the following error function for each column $j \in C$ and a column block $1 \leq l \leq L$,

$$Err(j, l | \boldsymbol{A}, P_R) = \frac{1}{t} \sum_{k \in [K]} \sum_{i \in R_k} |\boldsymbol{M}[i,j] - \boldsymbol{A}[k,l]|$$

The sum above ignores rows i for which $\boldsymbol{M}[i,j] = \star$. This is essentially the error with respect to the pattern $\boldsymbol{A}$ incurred by placing the column j in the l block, when the partition of the rows is given by P_R. Similarly $Err(i, k | \boldsymbol{A}, P_C)$ is defined for each row i.

5.2. Algorithm Analysis

Next we show that taking a sample size of $O(\frac{1}{\epsilon^2})$ suffices to approximate $OPT(\boldsymbol{M})$, within an additive factor of 4ϵ with high probability. OPT here denotes the optimal monochromatic solution cost for an input matrix $\boldsymbol{M}$. For the homogeneity maximization version, using a lower bound on the monochromatic agreement, we show that this is equivalent to a multiplicative approximation of $OPT' \cdot (1-\epsilon)$, where $OPT' = 1 - OPT$ is the optimal agreement score. Since the sample size is $O(\frac{1}{\epsilon^2})$, the run time of the basic step of our algorithm, is then $\exp(O(\frac{1}{\epsilon^2}))$.

The main technical claim we prove, is that given parameters $\epsilon > 0$ and $\delta > 0$, there exists some sample size, $t = t(\epsilon, \delta, K, L)$, for which for every input matrix $\boldsymbol{M}$ (of any size), if R^S and C^S are samples of size t drawn uniformly and independently from the set of rows and columns of $\boldsymbol{M}$, then for any fixed $K \times L$ bi-clustering pattern $\boldsymbol{A}$, there exists a partition of R^S into K groups and of C^S into L groups such that the $Mon_{\boldsymbol{A}}$ cost of the induced solution is ϵ-close to the $Mon_{\boldsymbol{A}}$ cost of the optimal partition with probability exceeding $1 - \delta$. With this, we get:

Theorem 5.2. *There is a randomized algorithm for the monochromatic bi-clustering problem, that given an input matrix $\boldsymbol{M}$ and an accuracy parameter ϵ, runs in time $|\boldsymbol{M}|^2 \exp\left(\frac{c}{\epsilon^2}\right)$ (for some constant c) and with high probability outputs a partition P of $\boldsymbol{M}$ with $Mon(\boldsymbol{M}, P) \leq OPT + 4\epsilon$.*

This implies:

Corollary 5.3. *For every K, L there exists a randomized polynomial time approximation scheme for the $K \times L$ monochromatic bi-clustering agreement maximization problem.*

The proof uses a lower bound of $\frac{1}{2}$ on the agreement of the optimal solution and shows that an additive 4ϵ approximation translates into a multiplicative $(1 - \epsilon)OPT'$ bound. The corollary follows from Theorem 5.2. Detailed proofs can be found in the appendix.

6. Deterministic Annealing

For relatively large values of K and L, the run-time of the approximation algorithm presented in section 5 (PTAS), which is polynomial in the input size, but exponential in K, L, can be too high, making it impractical for most real-world bi-clustering applications.

In this section we present a solution of the monochromatic objective by means of deterministic annealing (DA). The DA algorithm derived here, is directly motivated by one of the ideas which also underline the PTAS derivation. This property is formulated in claim 5.1. It is essentially saying that conditioned on a specific solution pattern, and given a partition of one of the item sets (i.e. rows or columns), the partition of the other item set, can be carried out efficiently in a decoupled manner, i.e. the assignment of the individual items in the second set is independent.

Rather than thinking of a clustering as a partition, here we represent a clustering as an assignment of the rows and columns to clusters. Let x_i and y_j denote the assignment of the i-th row and the j-th column to the row and column clusters, respectively. Hence, $x_i \in \{1, \ldots, K\}$ and $y_j \in \{1, \ldots, L\}$. To make the conditioning on the pattern explicit we define auxiliary variables $A[k, l] \in \{0, 1\}$ for $1 \leq k \leq K$ and $1 \leq l \leq L$ coding the majority of the bi-cluster k, l.

The DA solution alternates between updating the three blocks of variables: row assignments $\boldsymbol{x}$, column assignments $\boldsymbol{y}$ and the bi-cluster majorities $\boldsymbol{A}$. The row partition is sampled according to the following probabilities:

$$P(x_i = k | \boldsymbol{y}, \boldsymbol{A}) \propto \exp(-\frac{1}{T} \sum_{j \in [n]} |\boldsymbol{M}[i, j] - A[k, y_j]|),$$

whereas the column probabilities are given as follows:

$$P(y_j = l | \boldsymbol{x}, \boldsymbol{A}) \propto \exp(-\frac{1}{T} \sum_{i \in [m]} |\boldsymbol{M}[i, j] - A[x_j, l]|).$$

For $v \in \{0, 1\}$, the bi-cluster majority probabilities are

$$P(A[k, l] = v | \boldsymbol{x}, \boldsymbol{y}) \propto \exp(-\frac{1}{T} \sum_{x_i = k, y_j = l} |\boldsymbol{M}[i, j] - v|).$$

T is a temperature parameter that is decreased during the run of the algorithm in order to eventually only sample low-cost configurations. In case entries in $\boldsymbol{M}$ are missing, the corresponding elements do not contribute to the costs.

6.1. Deterministic Annealing Variant

In the experiments in section 7 we benchmark the DA solution, denoted as *DA*, against a variant of this solution which performs sampling without conditioning on the bi-cluster majorities. We denote this solver *DA-no-auxiliary*. This modification requires computing the assignments of each row and column individually, given the assignments of the rest of the variables. The clear disadvantage of this solver is an increase of the running time by a factor of $|\boldsymbol{M}|$. The reason it is interesting to compare with this solver, is to get empirical evidence on whether the introduction of the auxiliary variables affects the quality of the solution of a deterministic annealing scheme.

7. Experiments

The nature of the bi-clustering task as an unsupervised learning method, having no clear ground truth for real data, makes it hard to compare different approaches aiming at optimizing different cost functions. In order to draw connections to existing work, in section 7.2 we apply the monochromatic DA solver to the Osherson dataset of animal-features matrix gathered in a psychological experiment (Osherson et al., 1991). This data was used in (Kemp & Tenenbaum, 2006) to motivate a relational learning model. In section 7.1 we empirically demonstrate that the DA solver for the monochromatic objective performs well on synthetic data, despite not having provable guarantees on the solution. Even on relatively large instances it finds a solution with monochromatic cost close to the assumed minimal cost, given by the noise introduced to the data. We compare the cost with a variant of the solver detailed in section 6.1.

7.1. Synthetic Data

For a matrix size (m, n), parameters (K, L), and a noise level σ, we generate a data instance $\boldsymbol{M}_\sigma$ in the following way. First we generate a pattern matrix of size (K, L) by uniformly at random assigning each entry a value in $\{0, 1\}$. After fixing the pattern, for each row and column we choose uniformly at random an assignment to a row cluster $1 \leq k \leq K$, and column cluster $1 \leq l \leq L$ respectively. Finally, for each entry $\boldsymbol{M}_\sigma[i, j]$, we choose the value matching the assignment of i and j with probability $1 - \sigma$, and with probability σ we choose the other value. We report the results on three synthetic dataset sizes, $50, 500, 1000$ with $(5, 5), (10, 10), (10, 10)$ number of bi-clusters used for generation. For each size and number of bi-clusters parameters, we generate 5 instances, the reported values are the averaged errors. For the noise level we use 10 values in the interval $[0.01, 0.35]$, since we consider binary data, higher noise level may contain very little structure. We tried several values of temperature T and update schemes, and finally choose the run with the lowest final monochromatic cost.

On each dataset instance we run the solvers *DA* and *DA-no-auxiliary* (see section 6.1), until convergence (determined by a fixed number of iterations with no change to the lowest cost found). On the largest data set, some of the runs of *DA-no-auxiliary* did not converge within a limit of 8 hours. We report the lowest values obtained within the time limit.

Figure 1 presents the results of the synthetic data experiment. Both variants achieve the expected optimal cost on the smallest dataset. On the other two, *DA* achieves a lower cost compare to *DA-no-auxiliary*. In terms of running-time *DA-no-auxiliary* is on average at least 10 times slower (partially did not converge) compared to *DA*.

7.2. Feature-Animal Dataset

In this experiment we use the deterministic annealing solver *DA*, on the Osherson dataset of animal-features matrix gathered in a psychological experiment (Osherson et al., 1991), and used in (Kemp & Tenenbaum, 2006). The rows of the matrix correspond to 50 animals and the columns to 85 features of the animals, including physical properties, behavioral elements, affinity to different habitats etc'. The entries of the original matrix are real values in the interval $0 - 100$. We threshold the values to obtain a binary matrix, we use 25 as a threshold, which is the integer with lowest sum of squared errors to resulting class means. We use $K = 11$ and $L = 15$, these values peaked in the reduction of the overall cost as a function of the number of clusters introduced. In (Kemp & Tenenbaum, 2006) the number of animal clusters used was 12, while the number of feature clusters was much higher.

The monochromatic solution on this data is presented in Figure 2. The partition of the animals is very similar to the one given in (Kemp & Tenenbaum, 2006). In terms of the feature partitions, the monochromatic partition seems very reasonable. In (Kemp & Tenenbaum, 2006) many features are assigned to a singleton cluster, which essentially means that they do not take part in the lower dimensionality structure. This could be a result of their specific choice of the parameter that controls the penalty for new clusters.

8. Conclusions

We introduce a formulation of bi-clustering which is both natural, in the sense that it captures the aim of most existing bi-clustering works, and offers a convenient framework to analyze the bi-clustering task from a theoretical point of view. We show that the resulting optimization problem is NP-hard but can be approximated up to any multiplicative factor in polynomial time. We also provide an efficient deterministic annealing solver based on ideas from the PTAS.

Acknowledgments

We would like to thank Patrick Pletscher for helping out with the experimental setup and proof reading the paper, as well as Joachim Buhmann for insightful discussions.

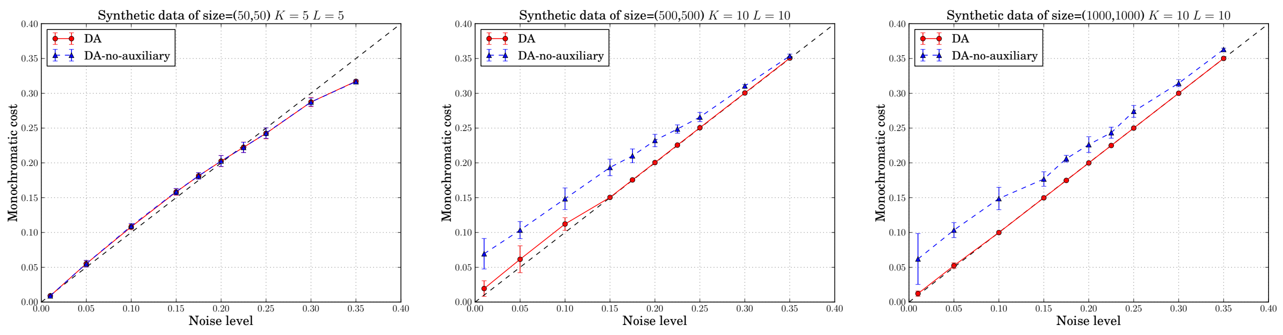

Figure 1. Comparison of the average monochromatic cost on synthetic data with varying sizes and number of bi-clusters. Both methods are deterministic annealing schemes for the monochromatic bi-clustering problem. *DA* uses auxiliary variables to represent the block majorities and performs sampling to determine their value. *DA-no-auxiliary* is a straight forward deterministic annealing scheme with no auxiliary variables. The dotted line represents the expected optimal monochromatic cost, which is the amount of noise introduced. *Left*: data size: 50×50, $K = 5, L = 5$, *Middle*: data size: 500×500, $K = 10, L = 10$, *Right*: data size: 1000×1000, $K = 10, L = 10$.

F0	chewteeth, walks, quadrapedal, oldworld, ground
F1	flippers, swims, fish, coastal, ocean, water
F2	big, strong, muscle
F3	vegetation, timid, group
F4	blue, strainteeth, plankton, skimmer, arctic
F5	gray, hairless
F6	small, forager, forest
F7	toughskin, bulbous, slow, inactive
F8	lean, fast, active, agility, smart
F9	black, brown, furry, tail, newworld
F10	hooves, longleg, horns, grazer, plains, fields
F11	hands, smelly, bipedal, jungle, tree, nestspot
F12	paws, meatteeth, claws, meat, hunter, stalker, fierce, solitary
F13	white, buckteeth, tunnels, weak, nocturnal, hibernate, domestic
F14	orange, red, yellow, patches, spots, stripes, pads, longneck, tusks, flys, hops, insects, scavenger, desert, bush, mountains, cave

A0	polar bear
A1	blue whale, humpback whale, walrus
A2	antelope, horse, zebra, deer
A3	gorilla, chimpanzee, giant panda
A4	hippopotamus, elephant, rhinoceros, pig
A5	spider monkey, squirrel, bat
A6	moose, ox, sheep, giraffe, buffalo, cow
A7	skunk, mole, hamster, rabbit, mouse
A8	killer whale, beaver, seal, otter, dolphin
A9	persian cat, siamese cat, fox, chihuahua, rat, weasel, raccoon
A10	grizzly bear, dalmatian, german shepherd, tiger, leopard, wolf, bobcat, lion, collie

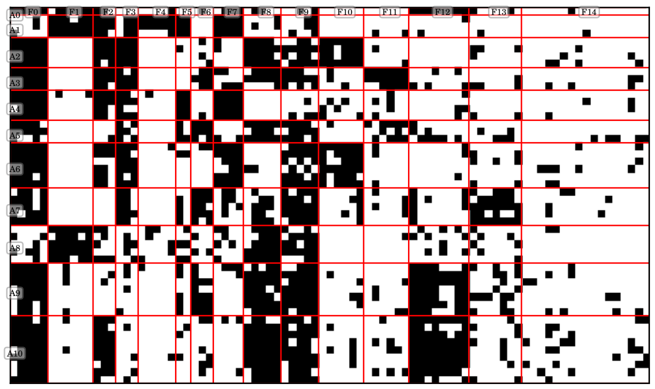

Figure 2. The monochromatic deterministic annealing solution *DA*, for the Osherson dataset of animal and features $(50 \cdot 85)$. *Upper Left*: The original (binarized) matrix. *Upper right*: The partition of the features into 15 clusters. *Lower left*: The partition of the animals into 11 clusters. *Lower right*: The permuted matrix according to the bi-clustering solution.

References

Bansal, N., Blum, A., and Chawla, S. Correlation clustering. *Machine Learning, Special Issue on Clustering*, 56:89–113, 2004.

Chakrabarti, D., Papadimitriou, S., Modha, D. S., and Faloutsos, C. Fully automatic cross-associations. In *KDD*, pp. 79–88, 2004.

Cheng, Y. and Church, G. M. Biclustering of expression data. In *International Conference on Intelligent Systems for Molecular Biology (ISMB)*, pp. 93–103, 2000.

Cho, H., Dhillon, I. S., Guan, Y., and Sra, S. Minimum sum-squared residue co-clustering of gene expression data. In *Proceedings of the Fourth SIAM International Conference on Data Mining*, pp. 114–125. SIAM, 2004.

Dhillon, I. S., Mallela, S., and Modha, D. S. Information-theoretic co-clustering. In *KDD*, pp. 89–98, 2003.

Giotis, I. and Guruswami, V. Correlation clustering with a fixed number of clusters. *Theory of Computing*, 2(13):249–266, 2006.

Goldreich, O., Goldwasser, S., and Ron, D. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, July 1998.

Hartigan, J.A. Direct clustering of a data matrix. *Journal of the American Statistical Association*, 67 (337):123–129, 1972.

Hirai, Hiroshi, Chou, Bin-Hui, and Suzuki, Einoshin. A parameter-free method for discovering generalized clusters in a network. In *Discovery Science*, pp. 135–149, 2011.

Kemp, C. and Tenenbaum, J. B. Learning systems of concepts with an infinite relational model. In *In Proceedings of the 21st National Conference on Artificial Intelligence*, 2006.

Kemp, C. and Tenenbaum, J. B. The discovery of structural form. *Proceedings of the National Academy of Sciences of the United States of America*, 2008.

Osherson, D., Stern, J., Wilkie, O., Stob, M., and Smith, E. E. Default probability. *Cognitive. Science*, 15:251–270, 1991.

Papadimitriou, S., Sun, J., Faloutsos, C., and Yu, P. S. Hierarchical, parameter-free community discovery. In *ECML/PKDD (2)*, pp. 170–187, 2008.

Tanay, A., Sharan, R., and Shamir, R. Biclustering Algorithms: A Survey. In *Handbook of Computational Molecular Biology*, 2006.