bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

Object-Attribute Biclustering for Elimination of Missing Genotypes in Ischemic Stroke Genome-Wide Data

Dmitry I. Ignatov[1], Gennady V. Khvorykh[2], Andrey V. Khrunin[2], Stefan Nikolić[1], Makhmud Shaban[1], Elizaveta A. Petrova[3], Evgeniya A. Koltsova[3], Fouzi Takelait[1], and Dmitrii Egurnov[1]

[1] National Research University Higher School of Economics, Russia
dignatov@hse.ru, http://www.hse.ru
[2] Institute of Molecular Genetics of National Research Centre "Kurchatov Institute", Russia
khvorykh@img.ras.ru, http://img.ras.ru
[3] Pirogov Russian National Research Medical University, Russia
http://rsmu.ru

Abstract. Missing genotypes can affect the efficacy of machine learning approaches to identify the risk genetic variants of common diseases and traits. The problem occurs when genotypic data are collected from different experiments with different DNA microarrays, each being characterised by its pattern of uncalled (missing) genotypes. This can prevent the machine learning classifier from assigning the classes correctly. To tackle this issue, we used well-developed notions of object-attribute biclusters and formal concepts that correspond to dense subrelations in the binary relation *patients* $\times$ *SNPs*. The paper contains experimental results on applying a biclustering algorithm to a large real-world dataset collected for studying the genetic bases of ischemic stroke. The algorithm could identify large dense biclusters in the genotypic matrix for further processing, which in return significantly improved the quality of machine learning classifiers. The proposed algorithm was also able to generate biclusters for the whole dataset without size constraints in comparison to the In-Close4 algorithm for generation of formal concepts.

Keywords: Formal Concept Analysis, Biclustering, Single Nucleotide Polymorphism, Missing Genotypes, Data Mining, Ischemic Stroke

1 Introduction

The recent progress in studying different aspects of human health and diversity (e.g., genetics of common diseases and traits, human population structure, and relationships) is associated with the development of high-throughput genotyping technologies, particularly with massive parallel genotyping of Single Nucleotide Polymorphisms (SNPs) by DNA-microarrays [1]. They allowed the determination of hundreds of thousands and millions of SNPs in one experiment and were

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

the basis for conducting genome-wide association studies (GWAS). Although thousands of genetic loci have been revealed in GWAS, there are practical problems with replicating the associations identified in different studies. They seem to be due to both limitations in the methodology of the GWAS approach itself and differences between various studies in data design and analysis [2]. The machine learning (ML) approaches were found to be quite promising in this field [3].

Genotyping by microarrays is efficient and cost-effective, but missing data appear. GWAS is based on a comparison of frequencies of genetic variants among patients and healthy people. It assumes that all genotypes are provided (usually, their percentage is defined by a genotype calling threshold). In this article, we demonstrate that missing data can affect not only statistical analysis but also the ML algorithms. The classifiers can fail because of missing values (uncalled genotypes) being distributed non-randomly. We assume that each set of DNA-microarray can possess a specific pattern of missing values marking both the dataset of patients and healthy people. Therefore, the missing data needs to be carefully estimated and processed without dropping too many SNPs that may contain crucial genetic information.

To overcome the problem of missing data, we aimed to apply a technique capable of discovering some groupings in a dataset by looking at the similarity across all individuals and their genotypes. The raw datasets can be converted into an integer matrix, where individuals are in rows, SNPs are in columns, and cells contain genotypes. For each SNP, the person can have either AA, AB, or BB genotype, where A and B are the alleles. Thus the genotypes can be coded as 0, 1, and 2, representing the counts of allele B.

The proposed method can simultaneously cluster rows and columns in a data matrix to find homogeneous submatrices [4], which can overlap. Each of these submatrices is called a bicluster, and the process of finding them is called biclustering [4,5,6,7,8].

Biclustering in genotype data allows identifying sets of individuals sharing SNPs with missing genotypes. A bicluster arises when there is a strong relationship between a specific set of objects and a specific set of attributes in a data table. A particular kind of bicluster if a formal concept in Formal Concept Analysis (FCA) [9]. A formal concept is a pair of the form (extent, intent), where extent consists of all objects that share the attributes in intent, and dually the intent consists of all attributes shared by the objects in extent. Formal concepts have a desirable property of being homogeneous and closed in the algebraic sense, which resulted in their extensive use in Gene Expression Analysis (GEA) [10,11,12,13].

A concept-based bicluster (or object-attribute bicluster) [14] is a scalable approximation of a formal concept withe the following advantages:

1. Reduced number of patterns to analyze;
2. Reduced computational cost (polynomial vs. exponential);
3. Manual (interactive) tuning of bicluster density threshold;
4. Tolerance to missing (object, attribute) pairs.

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

In this paper, we propose an extended biclustering algorithm of [15] that can identify large biclusters with missing genotypes for categorical data (many-valued contexts with a selected value). This algorithm can generate a smaller amount of dense object-attribute biclusters than that of existing exact algorithms for formal concepts like concept miner In-Close4 [16], and is, therefore, better suited for large datasets. Moreover, during experimentation with the ischemic stroke dataset, we found that the number of large dense biclusters identified by our algorithm is significantly lower than the number of formal concepts extracted by In-Close4[4] and Concept Explorer (ConExp[5]) [17].

The paper is organized as follows. In Section 2, we recall basic notions from Formal Concept Analysis and Biclustering. In Section 3, we introduce a method of FCA-based biclustering and its variants along with bicluster post-processing schemes, consider discussing the complexity of the proposed algorithm. In Section 4, we describe a dataset that consists of a sample of patients and their SNPs collected from various (independent) groups of patients. Then we present the results obtained during experiments on this dataset in Section 5 and mention the used hardware and software configuration. Section 6 concludes the paper.

2 Basic notions

2.1 Formal Concept Analysis

Definition 1. *A **formal context** in FCA [9] is a triple $\mathbb{K} = (G, M, I)$ consisting of two sets, G and M, and a binary relation $I \subseteq G \times M$ between G and M. The triple can be represented by a matrix consisting of set of rows G, called* **objects**, *and that of M, called* **attributes**, *of the context and crosses representing incidence relation I. The notation gIm or $(g, m) \in I$ means that the object g* **has** *the attribute m.*

Definition 2. *For $A \subseteq G$ and $B \subseteq M$, let*

$$A' \stackrel{\text{def}}{=} \{m \in M \mid gIm \text{ for all } g \in A\}, \text{ and } B' \stackrel{\text{def}}{=} \{g \in G \mid gIm \text{ for all } m \in B\}.$$

These two operators are the **derivation operators** *for $\mathbb{K} = (G, M, I)$.*

Proposition 1. *Let (G, M, I) be a formal context, for subsets $A, A_1, A_2 \subseteq G$ and $B \subseteq M$ we have*

1. $A_1 \subseteq A_2$ *if* $A_2' \subseteq A_1'$,
2. $A \subseteq A''$,
3. $A = A''$ *(hence,* $A'''' = A''$*),*
4. $(A_1 \cup A_2)' = A_1' \cap A_2'$,
5. $A \subseteq B' \Leftrightarrow B \subseteq A' \Leftrightarrow A \times B \subseteq I$.

Similar properties hold for subsets of attributes.

[4] https://sourceforge.net/projects/inclose/
[5] http://conexp.sourceforge.net

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

Definition 3. *A* **closure operator** *on set S is a mapping $\varphi : 2^S \to 2^S$ with the following properties:*
Let $X \subseteq S$, then

1. $\varphi(\varphi(X)) = \varphi(X)$ *(**idempotency**),*
2. $X \subseteq \varphi(X)$ *(**extensity**),*
3. $X \subseteq Y \Rightarrow \varphi(X) \subseteq \varphi(Y)$ *(**monotonicity**).*

For a closure operator φ the set $\varphi(\varphi(X))$ is called **closure** *of X, while a subset $X \subseteq S$ is called* **closed** *if $\varphi(\varphi(X)) = X$.*

It is evident from properties of derivation operators that for a formal context (G, M, I), the operators

$$(\cdot)'' : 2^G \to 2^G \text{ and } (\cdot)'' : 2^M \to 2^M$$

are closure operators.

Definition 4. *(A, B) is a* **formal concept** *of formal context $\mathbb{K} = (G, M, I)$ iff*

$$A \subseteq B,\ B \subseteq M,\ A' = B,\ \text{and } A = B'\ .$$

The sets A and B are called the **extent** *and the* **intent** *of the formal concept (A, B), respectively.*

This definition says that every formal concept has two parts, namely, its extent and intent. It follows an old tradition in philosophical concept logic, as expressed in the *Logic of Port Royal, 1662* [18].

Definition 5. *The set of all formal concepts $\underline{\mathfrak{B}}(B, M, I)$ is partially ordered, given by relation $\leq_{\mathbb{K}}$:*

$$(A_1, B_1) \leq_{\mathbb{K}} (A_2, B_2) \iff A_1 \subseteq A_2 \text{ (dually } B_2 \subseteq B_1)$$

$\underline{\mathfrak{B}}(B, M, I)$ is called **concept lattice** *of the formal context $\mathbb{K}$.*

In case an object has properties like colour or age the corresponding attributes should have values themselves.

Definition 6. *A* **many-valued context** *(G, M, W, J) consists of sets G, M and W and a ternary relation $J \subseteq G \times M \times W$ for which it holds that*

$$(g, m, w) \in J \text{ and } (g, m, v) \in I \text{ imply } w = v\ .$$

The elements of M are called **(many-valued) attributes** *and those of W* **attribute values**.

Since many-valued attributes can be considered as partial maps from G in W, it is convenient to write $m(g) = w$.

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

2.2 Biclustering

In [5], *bicluster* is defined as a homogeneous submatrix of an input object-attribute matrix of real values in general. Consider a dataset as a matrix, $A = (X, Y) \in \mathbb{R}^{n \times m}$, with a set of rows/objects/individuals $X = \{x_1, \ldots, x_n\}$ and set of columns/attributes/SNPs $Y = \{y_1, \ldots, y_m\}$. A submatrix constructed from a subset of rows $I \subseteq X$ and that of columns $J \subseteq Y$ is denoted as (I, J) is called a **bicluster** of A [5]. The bicluster should satisfy some specific homogeneity properties, which varies from method to another.

For instance, for the purpose of this research, we use the following FCA-based definition of a bicluster [19,14,15].

Definition 7. *For a formal context $\mathbb{K} = (G, M, I)$ any biset $(A, B) \subseteq I$ with $A \neq \emptyset$ and $B \neq \emptyset$ is called a* **bicluster**. *If $(g, m) \in I$, then the bicluster $(A, B) = (m', g')$ is called an object-attribute or* **OA-bicluster** *with* **density** $\rho(A, B) = \frac{|I \cap (A \times B)|}{|A| \cdot |B|}$.

The density $\rho(m', g')$ of a bicluster (m', g') is the bicluster quality measure that shows how many non-empty pairs the bicluster contains divided by its size.

Several basic properties of OA-biclusters are below.

Proposition 2.

1. *For any bicluster $(A, B) \subseteq 2^G \times 2^M$ it is true that $0 \leq \rho(A, B) \leq 1$,*
2. *OA-bicluster (m', g') is a formal concept iff $\rho = 1$,*
3. *If (A, B) is a OA-bicluster, there exists (at least one) its* **generating pair** *$(g, m) \in A \times B$ such that $(m', g') = (A, B)$,*
4. *If (m', g') is a OA-bicluster, then $(g'', g') \leq (m', m'')$.*
5. *For every $(g, m) \in I$, $(h, n) \in g'' \times m''$, it follows $(m', g') = (n', h')$.*

In Fig. 1, you can see the example of OA-bicluster, for a particular pair $(g, m) \in I$ of a certain context (G, M, I). In general, only the regions (g'', g') and (m', m'') are full of non-empty pairs, i.e. have maximal density $\rho = 1$, since they are object and attribute formal concepts respectively. The black cells indicate non-empty pairs, which one may found in less dense white regions.

Definition 8. *Let $(A, B) \in 2^G \times 2^M$ be a OA-bicluster and $\rho_{min} \in (0, 1]$, then (A, B) is called dense if it satisfies the constraint $\rho(A, B) \geq \rho_{min}$.*

The number of OA-biclusters of a context can be much less than the number of formal concepts (which may be $2^{\min(|G|,|M|)}$), as stated by the following propositions.

Proposition 3. *For a formal context $\mathbb{K} = (G, M, I)$ the largest number of OA-biclusters is equal to $|I|$ and all OA-biclusters can be generated in time $\mathcal{O}(|I|)$.*

Proposition 4. *For a formal context $\mathbb{K} = (G, M, I)$ and $\rho_{min} > 0$ the largest number of dense OA-biclusters is equal to $|I|$, all dense OA-biclusters can be generated in time $\mathcal{O}(|I||G||M|)$.*

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

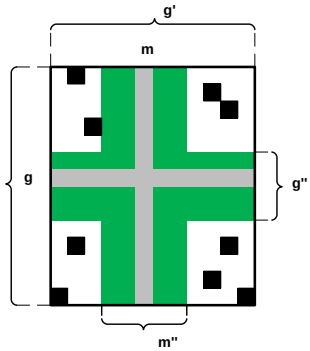

Fig. 1. OA-bicluster based on object and attribute primes.

3 Model and algorithm description

3.1 Parallel OA-biclustering algorithm

Algorithm 1 is a rather straightforward implementation, which takes an initial many-valued formal context and minimal density threshold as parameters and computes large dense biclusters for each (object, attribute) pair in the relation I that indicates which objects have SNP with missing values. However, since OA-biclusters for many-valued contexts were not formally introduced previously, we use an induced formal context with one-valued attributes denoting missing attribute-values of an original genotype matrix to correctly apply the definition of dense OA-bicluster.

Definition 9. *Let $\mathbb{K} = (G, M, W, J)$ is a many-valued context and $v \in W$ is a selected value (e.g., denoting the absence of an SNP value), then its* **derived context for the value** *v is $\mathbb{K}_v = (G, M, I)$ where gIm iff $(g, m, v) \in J$.*

For genotype matrices with missing SNP values represented by many-valued contexts, similar representation can be expressed in terms of co-domains of many-valued attributes (if the absence of the value of $m(g)$ means the absence of the corresponding SNP value) or by means of nominal scaling with a single attribute for the missing value v [9].

If we compare the number of output pattern for formal concepts and dense OA-biclusters, in the worst case these values are $2^{min(|G|,|M|)}$ versus $|I|$. The time complexity of our algorithm is polynomial, $\mathcal{O}(|G||M||I|)$, versus exponential in the worse case for BiMax [20], $\mathcal{O}(|G||M||L| \log |L|)$, or $\mathcal{O}(|G|^2|M||L|)$ for CbO algorithms family [21], where $|L|$ is a number of generated concepts (also considered as biclusters) and is exponential in the worst case $|L| = 2^{min(|G|,|M|)}$.

For calculating biclusters that fulfil a minimum density constraint, we need to perform several steps (see Algorithm 1). Steps 5-8 consists of applying the Galois operator to all objects in G and steps 9-12 then to all attributes in M within the induced context. The outer for loops are parallel (the concrete implementation

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

may differ), while the internal ones are ordinary for loops. Then all biclusters are enumerated in a parallel manner as well, and only those that fulfil the minimal density requirement are retained (Steps 13-16). Again, efficient implementation of set data-structure for storing biclusters and duplicate elimination of the fly in parallel execution mode are not addressed in the pseudo-code.

The novelties of this algorithm are that we used parallelization to generate the OA-bicluster giving as input a medium-sized dataset (e.g. $10^3 \times 10^4$), that is to make our program runs faster, and the possibility to work with selected values reducing many-valued context to contexts with one-valued attributes.

Algorithm 1: OA-bicluster generation for a many-valued context.

Data: $\mathbb{K} = (G, M, W, J)$ is a many-valued formal context, ρ_{min} is a threshold density value of bicluster density and $v \in W$ is a selected value

Result: $B = \{(A, B) | (A, B)$ is an OA-bicluster for value $v\}$

```
1  begin
2      Obj.Size := |G|
3      Attr.Size := |M|
4      B ⟵ ∅
5      parallel for g ∈ G do
6          for m ∈ M do
7              if m(g)=v then
8                  Obj[g].Add(m)
9      parallel for m ∈ M do
10         for g ∈ G do
11             if m(g)=v then
12                 Attr[m].Add(g)
13     parallel for (g, m, w) ∈ J do
14         if w=v then
15             if ρ(Attr[m], Obj[g]) ≥ ρ_min then
16                 B := B ∪ {(Attr[m], Obj[g])}
```

3.2 One-pass version of the OA-biclustering algorithm

Let us describe the online problem of finding the set of prime OA-biclusters based on the online OAC-Prime Triclustering [22]. Let $\mathbb{K} = (G, M, I)$ be a context. The user has no a priori knowledge of the elements and even cardinalities of G, M, and I. At each iteration, we receive a set of pairs ("batch") from I: $J \subseteq I$. After that, we must process J and get the current version of the set of all biclusters. It is important in this setting to consider every pair of biclusters different if they have different generating pairs even if their extents and intents are equal, because any other pair can change only one of them, thus making them different.

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

Also, the algorithm requires that the dictionaries containing the prime sets are implemented as hash-tables or similar efficient key-value structures. Because of this data structure, the algorithm can efficiently access prime sets.

The algorithm itself is also straightforward (Alg. 2). It takes a set of pairs (J) and current versions of the biclusters set ($\mathcal{B}$) and the dictionaries containing prime sets ($PrimesO$ and $PrimesA$) as input and outputs the modified versions of the bicluster set and dictionaries. The algorithm processes each pair (g, m) of J sequentially (line 1). On each iteration the algorithm modifies the corresponding prime sets: it adds m to g' (line 2) and g to m' (line 3).

Finally, it adds a new bicluster to the bicluster set. Note that this bicluster contains pointers to the corresponding prime sets (in the corresponding dictionaries) instead of their copies (line 4).

In effect, this algorithm is very similar to the original OA-biclustering algorithm with some optimizations. First of all, instead of computing prime sets at the beginning, we modify them on spot, as adding a new pair to the relation modifies only two prime sets by one element. Secondly, we remove the main loop by using pointers for the bicluster' extents and intents, as we can generate biclusters at the same step as we modify the prime sets. And third, it uses only one pass through the pairs of the binary relation I, instead of enumeration of different pairwise combinations of objects and attributes.

Algorithm 2: Online generation of OA-biclusters

Input: J is a set of object-attribute pairs;
$\mathcal{B} = \{\mathbf{b} = (*X, *Y)\}$ is the current set of OA-biclusters;
$PrimesO$, $PrimesA$;
Output: $\mathcal{B} = \{\mathbf{b} = (*X, *Y)\}$;
$PrimesO$, $PrimesA$;
1: **for all** $(g, m) \in J$ **do**
2: $PrimesO[g] := PrimesO[g] \cup \{m\}$
3: $PrimesA[m] := PrimesAC[m] \cup \{g\}$
4: $\mathcal{B} := \mathcal{B} \cup \{(\&PrimesA[m], \&PrimesO[g])\}$
5: **end for**

Each step requires constant time: we need to modify two sets and add one bicluster to the set of biclusters. The total number of steps is equal to $|I|$; the time complexity is linear $\mathcal{O}(|I|)$. Beside that the algorithm is one-pass.

The memory complexity is the same: for each of $|I|$ steps the size of each dictionary containing prime sets is increased either by one element (if the required prime set is already present), or by one key-value pair (if not). Since each of these dictionaries requires $\mathcal{O}(|I|)$ memory, the memory complexity is also linear.

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

3.3 Post-processing constraints

Another important step used as an addition to this algorithm is post-processing. Thus, we may want to remove additional biclusters with the same extent and intent from the output. Simple constraints like minimal support condition can be processed during this step without increasing the original complexity. It should be done only during the post-processing step, as the addition of a pair in the main algorithm can change the set of biclusters, and, respectively, the values used to check the conditions. Finally, if we need to fulfil more difficult constraints like minimal density condition, the time complexity of the post-processing will be higher than the time complexity of the original algorithm, but it can be also efficiently implemented.

To remove the same biclusters we need to use an efficient hashing procedure that can be improved by implementing it in the main algorithm. For this, for all prime sets, we need to keep their hash-values with them in the memory. And finally, when using hash-functions other than LSH function (Locality-Sensitive Hashing) [23], we can calculate hash-values of prime sets as some function of their elements (for example, exclusive disjunction or sum). Then, when we modify prime sets, we just need to get the result of this function and the new element. In this case, the hash-value of the bicluster can be calculated as the same function of the hash-values of its extent and intent.

Then it would be enough to implement the bicluster set as a hash-set in order to efficiently remove the additional entries of the same bicluster.

Pseudo-code for the basic post-processing (Alg. 3).

Algorithm 3: Post-processing for the online OA-biclustering algorithm.

Input: $\mathcal{B} = \{\mathbf{b} = (*X, *Y)\}$ is a full set of biclusters;
Output: $\overline{\mathcal{B}} = \{\mathbf{b} = (*X, *Y)\}$ is a processed hash-set of biclusters;

```
for all b ∈ B do
    Calculate hash(b)
    if hash(b) ∉ B̄ then
        B̄ := B̄ ∪ {b}
    end if
end for
```

If the names (codes) of the objects and attributes are small enough (the time complexity of computing their hash values is $\mathcal{O}(1)$), the time complexity of the post-processing is $\mathcal{O}(|I|)$ if we do not need to calculate densities, and $\mathcal{O}(|I||G||M|)$ otherwise. Also, the basic version of the post-processing does not require any additional memory; so, its memory complexity is $O(1)$.

Finally, the algorithm can be easily paralleled by splitting the subset of input pairs into several subsets, processing each of them independently, and merging the resulting sets afterwards, which may lead to distributed computing schemes for larger datasets (cf. [24]).

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

In case the output of the post-processing step is stored in a relational database along with the computed statistics and generating pairs, further usage of selection operators [25] is convenient to consider only a specific subset of biclusters.

We use the following operator resulting in a specific subset of biclusters

$$\sigma_{(\alpha_{min} \leq |A| \leq \alpha_{max}) \wedge (\beta_{min} \leq |B| \leq \beta_{max}) \wedge (\rho_{min} \leq \rho(A,B) \leq \rho_{max})}(\mathcal{B}),$$

where $|A|$ is the extent size, $|B|$ is the intent size, and $\rho(A, B)$ is the density of OA-bicluster $\mathbf{b} \in \mathcal{B}$, respectively. One more reason to use postprocessing is neither monotonic nor anti-monotonic character of the minimal density constraint in the sense of constraints pushing in pattern mining [10,15].

4 Data collection

Collection of patients with ischemic stroke and their clinical characterisation were made at the Pirogov Russian National Research Medical University. The DNA extraction and genotyping of the samples were described previously [26].

The dataset contains samples corresponding to individuals with a genetic portrait for each and a group label. The former represents the genotypes determined at many SNPs all over the genome. The latter takes values 0 or 1 depending on whether a person did not have or had a stroke. Each SNP is a vector that components can take values from $\{0, 1, 2, -1\}$, where 0, 1, and 2 denote the genotypes, and -1 indicates a missing value.

We represent the dataset as a many-valued formal context. In the derived context $\mathbb{K} = (G, M, I)$, where objects from G stand for samples and attributes from M stand for SNPs, gIm means that an individual g has a missing SNP m. The context has the following parameters $|G| = 1,323$, $|M| = 85,142$, and $|I| = 45,075$ which represents the total number of attributes with missing values in the dataset and cover 0.491% of the whole data matrix. The number of attributes without missing values is 40,067.

The genotypic data were obtained with DNA-microarrays. The dataset was compiled from several experiments where different types of microarrays were applied. Not all genotypes are equally measured during the experiment. Thus, there is a certain instrumental error. The quality of DNA can also affect the output of the experiments. Fig. 2 shows how many individuals have exactly N missing genotypes per SNP in the dataset.

For instance, many individuals have about 85 missing genotypes per SNP.

5 Experiments

5.1 Hardware and software configuration

The experimental results with OA-biclustering generation and processing were obtained on an Intel(R) Core(TM) i5-8265U CPU @ 1.80 GHz with 8 GB of RAM and 64-bit Windows 10 Pro operating system. We used the following software releases to perform our experiments: Python 3.7.4 and Conda 4.8.2.

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

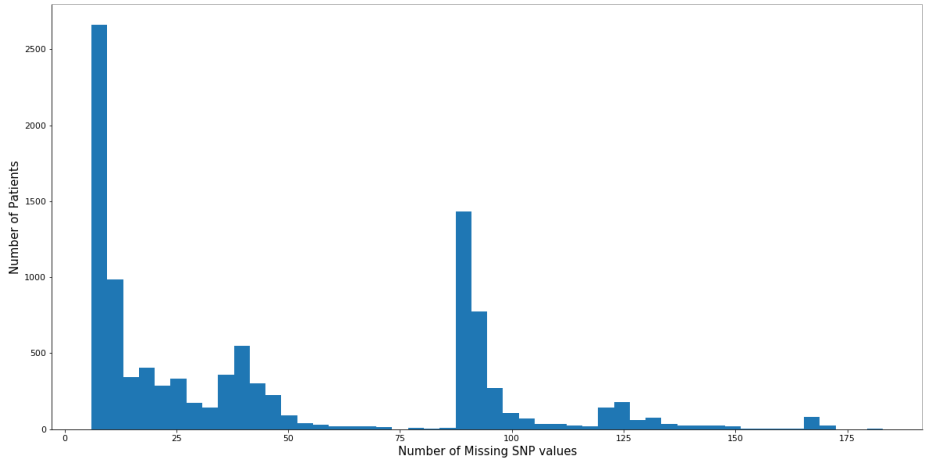

Fig. 2. The distribution of the number of missing SNP values by columns **before** elimination.

5.2 Identification of Biclusters with Missing SNPs

The following experiment was performed with ischemic stroke data collection: first of all, 383,733 OA-biclusters, with duplicates, were generated after applying the parallel biclustering algorithm to the dataset.

As we can see from the graph in Figure 3, there is a reasonable amount of biclusters with a density value greater than 0.9. The distributions of biclusters by extent and intent show that the majority of biclusters have about 90 samples and 2,600 SNPs, respectively.

For the selection of large dense biclusters, we set the density constraint to be $\rho_{min} = 0.9$. Additional constraints were set as follows: $3 \leq |m'| \leq 1,500$ for the extent size and $3 \leq |g'| \leq 80,000$ for the intent size. In total, we selected 98,529 OA-biclusters with missing values. For this selection, the graph in Fig. 4 shows the selected peaks of large dense biclusters for different extent sizes.

Example 1. Biclusters in the form $(patients, SNPs)$.
For generating pair $(g, m) = (1102, rs6704827_A)$ we have that

$$(m', g') \in \sigma_{(3 \leq |A| \leq 1,500) \wedge (3 \leq |B| \leq 80,000) \wedge (0.9 \leq \rho(A,B) \leq 1)}(\mathcal{B}), \text{ where}$$

$(m'g') = (\{1101, 1102, \ldots, 1114\}, \{rs10915587_G, rs284267_A, \ldots, rs12171249_A\})$, $\rho(m', g') \approx 0.91$, $|m'| = 14$ individuals, $|g'| = 758$ SNPs, 9,657 pairs out of 10,612 correspond to missing SNP values.

We studied further large dense biclusters and chose the densest ones with possibly larger sizes of their extents and intents from each of the peaks identified in their distributions, respectively (Fig. 3).

Here are some examples of these subsets with their associated graphs.

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

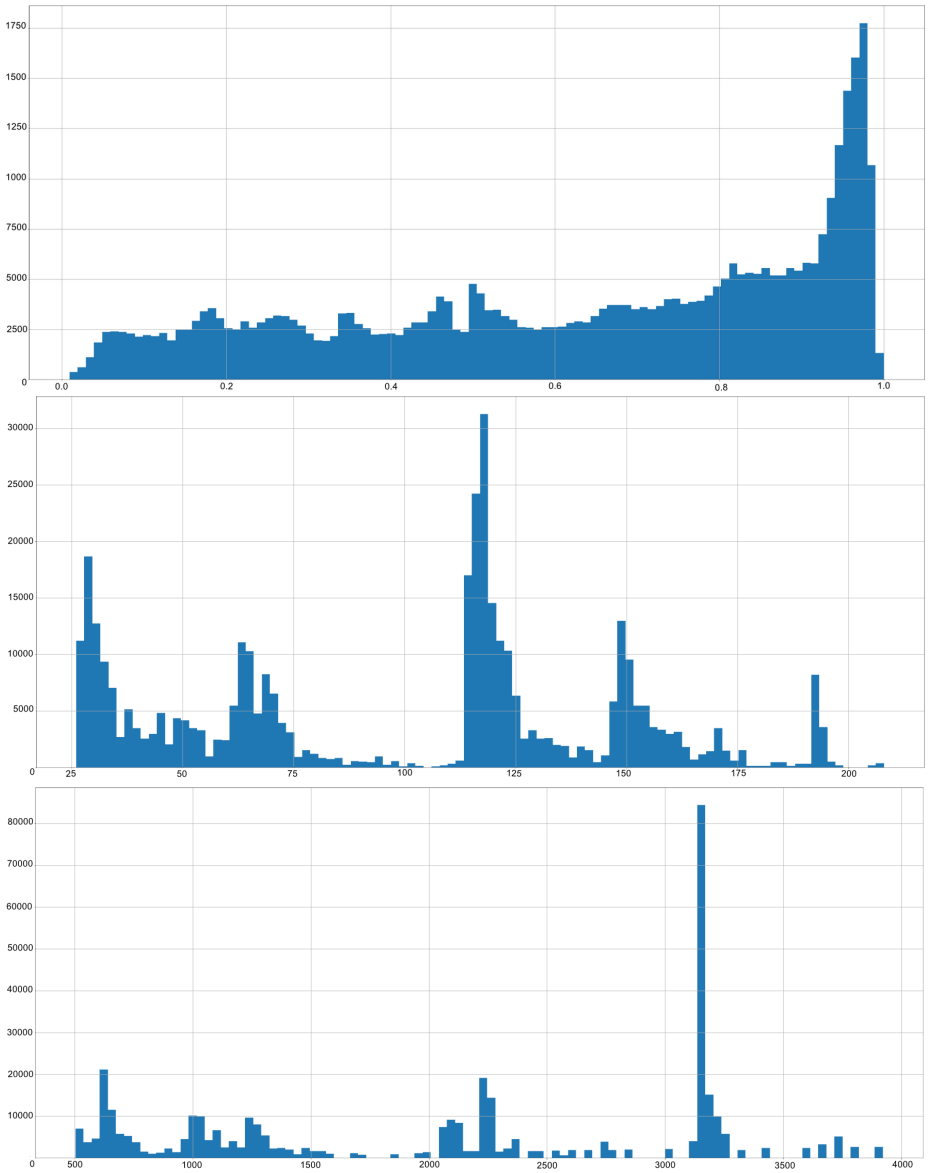

Fig. 3. The distribution of the number of biclusters by their density (top), extent (middle) and intent sizes (bottom).

Example 2. We can further narrow down the number of patterns in the previous selection by looking at the distribution of biclusters by their extent size and choosing proper boundaries. Thus, in Fig. 4, there is the third largest peak of the number of biclusters near the extent size 125.

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

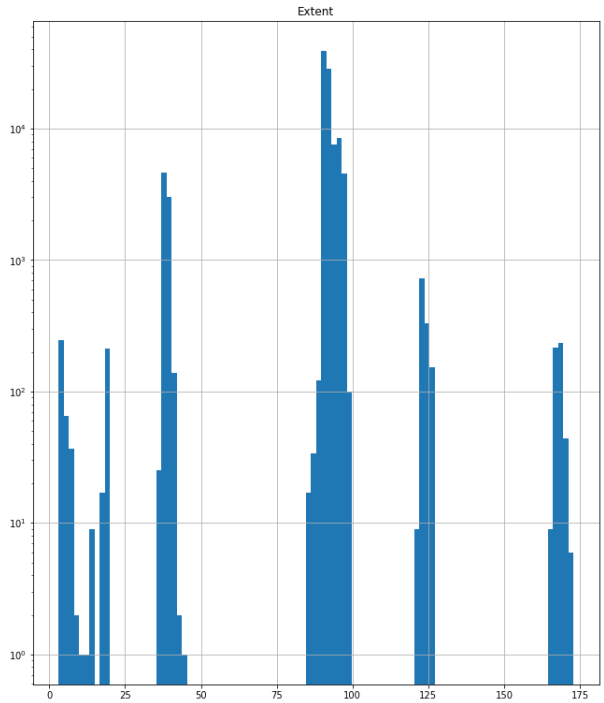

Fig. 4. The distribution of dense biclusters ($\rho_{min} = 0.9$) by their extent size.

For the constraints below

$$\rho_{min} = 94.08\% \wedge \rho_{max} = 100\% \wedge |g'| = 122 \wedge (3 \leq |m'| \leq 80,000)$$

the large dense bicluster with its intent size of 455 is identified and selected. Such bicluster has a large number of missing genotypes, which are subject to be eliminated later on.

Example 3. The selection around the rightmost peak (*see Fig. 4*) and further refining of the minimal value density

$$\rho_{min} = 95.4\% \wedge \rho_{max} = 100\% \wedge 160 \leq |g'| \leq 175 \wedge (3 \leq |m'| \leq 80,000)$$

resulted in the large bicluster with the extent size of 108 and the intent size of 166.

5.3 Elimination of Large Biclusters with Missing Genotypes

After applying the proposed biclustering algorithm to the collected dataset, all large biclusters with missing genotypes were identified and eliminated. That

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

resulted in a new data matrix ready for further analysis[6]. We consolidate the evolution of the two datasets before and after removing missing values in Table 1.

Table 1. Basic statistics of the datasets **before** and **after** elimination of missing values.

	no. samples	no. SNPs	no. NaNs	NaNs fraction
Before elimination	1,223	85,142	553,430	0.49%
After elimination	1,472	82,690	388,052	0.31%

As seen from Table 1, the biclustering algorithm application resulted in improvement in terms of entries corresponding to SNPs with missing genotypes, a fraction of such entries is reduced by 29.88%. The total number of biclusters generated before and after eliminating SNP with missing genotypes is 383,733 (with duplicates) and 259,440, respectively. The total amount of time for generating these biclusters before and after deleting missing data is 3433.2 and 2293.7 seconds (by Algorithm 1), respectively. As for online Algorithm 2, it has processed the original context (before elimination) in 1.5 seconds, while the post-processing Algorithm 3 for density computation has taken 907 seconds in sequential and 651 seconds in parallel (six cores) modes, respectively.

Fig. 5 shows the distribution of missing values in columns in the new data set (after elimination of missing data), which now has less ragged character.

5.4 Large Dense Biclusters Elimination and Classification Quality

We have conducted a number of machine learning experiments on our datasets to check the impact of eliminating missing data. Our proposed algorithm handled on the quality measures of supervised learning algorithms.

We choose to use gradient boosting on decision trees (GBDT). For this purpose, we selected two libraries where it is already implemented, CatBoost and LightGBM. Both implementations can handle missing values.

A genome can essentially be interpreted as a sequence of SNPs, so we made a decision to also use Long-Short Term Memory Network [27] as a strong approach to handling sequential data.

First dataset experiments. Firstly, we applied GBDT algorithm from CatBoost library to our initial dataset (before elimination of SNPs with missing genotypes). The following parameters were taken for the classifier:

- Maximum number of trees: 3;
- Tree depth limit: 3;

[6] `https://github.com/dimachine/OABicGWAS/`

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

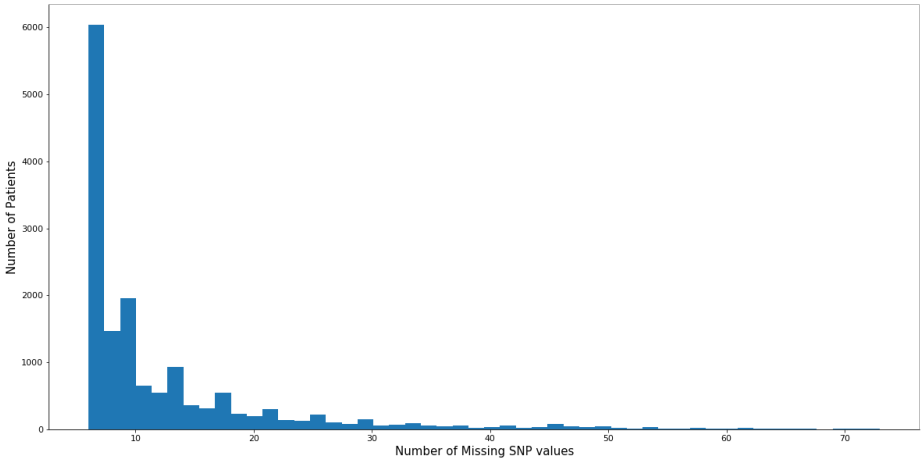

Fig. 5. Distribution of the number of SNPs with missing genotypes by columns **after** elimination.

– Loss function: binary cross-entropy (log-loss/binary cross-entropy).

We also applied LSTM approach the following way: the initial sequence was resized to 100 elements by a fully-connected layer, then the layer output was passed to the LSTM module element-wise. The hidden state of LSTM after the last element was passed to a fully-connected classification layer.

The scores on this dataset were evaluated with 3-fold cross-validation with stratified splits. Basic classification metrics' scores are present in Table 2.

Table 2. Classification scores on the test set **before** elimination of missing SNP values

	accuracy	**F1-score**	**precision**	**recall**
CatBoostClassifier	0.966	0.9758	0.9558	0.9967
FC+LSTM	0.890	0.926	0.880	0.982

These unexpectedly high scores were unrealistic because the GBDT model complexity was set to one of the lowest possible configurations, and the LSTM model, which is handling the data in a different way, also achieved high accuracy. For a lot of samples, the model learned to "understand" on which chip it was analyzed by looking at the patterns of missing genotypes, so the data leak was present.

Second dataset experiments. This dataset was obtained after the identification of large dense biclusters by application of our proposed algorithm with subsequent elimination. Table 3 recaps the experiments conducted on the dataset.

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

For the first and second experiments, we used *CatBoost classifier* with train/test split in the proportion of 8:2 and 3-fold cross-validation, respectively, while maintaining the balance of classes for model validation. In the third experiment, we used *LGBMClassifier* classifier with 3-fold cross-validation while maintaining the balance of classes for model validation. In the fourth experiment, the described earlier *LSTM* classifier was used with the aforementioned cross-validation.

Table 3. Scores results of different machine learning classifiers applied to the dataset **after** elimination of SNP with missing genotypes.

	no. trees	**depth**	**accuracy**	**F1-score**	**precision**	**recall**
CatBoostClassifier	2	2	0.715	0.834	0.715	**1.000**
	5	2	**0.773**	**0.862**	**0.761**	0.995
	5	3	**0.773**	**0.862**	**0.761**	0.995
CatBoostClassifier	4	3	**0.768**	**0.859**	**0.990**	**0.759**
	5	3	**0.768**	**0.859**	**0.990**	**0.759**
LGBMClassifier	5	3	0.753	0.852	**0.997**	0.744
	5	5	0.753	0.852	0.996	0.744
	4	4	0.751	0.851	**0.997**	0.742
	4	3	0.749	0.850	**0.997**	0.741
	5	4	**0.756**	**0.854**	0.996	**0.747**
FC+LSTM	-	-	0.731	0.839	0.735	0.981

From Table 3, one can see that scores are more realistic in comparison to those of Table 2, thus showing us that data leak and subsequent overfitting effects are gone. We realize that our proposed biclustering algorithm successfully identified large submatrices with missing data, which we eliminated and successfully removed the impact of data leak and overfitting.

5.5 Detecting concepts of missing SNP values under size constraints

In-Close4 is an open-source software tool [28], which provides a highly optimised algorithm from CbO family [21,29] to construct the set of concepts satisfying given constraints on sizes of extents and intents. In-Close4 takes as input a context and outputs a reduced concept lattice: all concepts satisfying the constraints given by parameter values ($|A| \geq m$ and $|B| \geq n$, where A and B are extent and intent of an output formal concept, and $m, n \in \mathbb{N}$).

To deal with our large real-world dataset, we changed the maximum default values used in the executable of In-Close4 parameters as follows:

```
#define MAX_CONS 30000000  //max number of concepts
#define MAX_COLS 90000     //max number of attributes
#define MAX_ROWS 2000      //max number of objects
#define MAX_FOR_B 90000000 //memory for storing intents
#define MAX_FOR_A 90000000 //memory for storing extents
```

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

From the following tables, one can see that the number of formal concepts generated by In-Close4 becomes several times larger than the number of OA-biclusters, in our case study. When we set the extent size constraint to 5 with the input context before and the extent and the intent size constraint to 20 and 0, respectively, after the elimination of missing data, the software crashed. Meanwhile, our proposed biclustering algorithms could manage to output all OA-biclusters in both cases.

As the author of InClose suggested in private communication, the tool was optimised for "tall" contexts with a large number of objects rather than attributes, while in bioinformatics the contexts are often "wide" like in our case when the number of SNPs is almost 57 times larger than that of individuals. So, the results on the transposed context along with properly set compilation parameters allowed to process the whole context for $m = 0$ and $n = 0$[7].

Table 4. The number of concepts and elapsed time generated by In-Close4 algorithm **before** eliminating SNPs with missing genotypes.

Min intent size	Min extent size	Total Time, s	No. of Concepts
0	45	21.2	18,617
0	40	23.6	34,400
0	30	35.8	68,477
0	20	46.1	165,864
0	10	64.3	214,007
0	5	188.3	1,220,576
0	0	143.43	1,979,439

Even though we still do not know the number of output concepts for the context after elimination of missing SNP values, their number is more than 10 times larger than that of OA-biclusters, which might be considered as argument in favour of their usage for the studied problem with rather low or no size constraints.

6 Conclusion

A new approach to process the missing values in datasets of SNP genotypes obtained with DNA-microarrays is proposed. It is based on OA-biclustering. We applied the approach to the real-world datasets representing the genotypes of patients with ischemic stroke and healthy people. It allowed us to estimate and eliminate the SNPs carefully with missing genotypes. Results of the OA-biclustering algorithm showed the possibility of detecting relatively large dense

[7] The last line in Table 4 and the last five lines in Table 5 corresponds to the experiments conducted for the final version of the paper on the transposed contexts.

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

Table 5. The number of concepts and elapsed time generated by In-Close4 algorithm **after** eliminating SNPs with missing genotypes.

Min intent size	Min extent size	Total Time, s	Number of Concepts
0	40	10.4	2,743
0	30	10.6	4,196
0	20	12.6	19,620
30	0	5.8	352,257
25	0	6.2	466,695
20	0	7.4	695,962
15	0	10.7	1,308,222
10	0	18.3	3,226,277

biclusters, which significantly helped in removing the effects of data leaks and overfitting while applying ML algorithms.

We compared our algorithm with In-Close4. The findings showed that the number of OA-biclusters generated by our algorithm is significantly lower than the number of formal concepts or biclusters generated by the In-Close4 algorithm. Besides, our algorithm has the advantage of using OA-bicluster without the need to experiment with finding the best minimum support, as in the case of using In-Close4 for generating formal concepts.

Since survey [30] mentioned frequent itemset mining (FIM) as a tool to identify strong associations between allelic combinations associated with diseases, the proposed algorithm needs further comparison with other approaches from FIM like DeBi [31] developed for GEA and anytime discovery approaches like Alpine [32] tested on GEA datasets as well; though their use may get complicated if we need to keep information about object names for decision-makers. It also requires further time complexity improvements to increase the scalability and quality of the extensive bicluster finding process for massive datasets.

Speaking about other possible applications of biclustering, we can suggest the development of a new technique to impute missing genotypes. Thus, biclustering has been recently applied to impute the missing values in gene expression data [33]. Both gene expression and SNP genotyping data are obtained with DNA-microarrays. The raw data can be represented as an integer matrix; therefore, biclustering can be potentially applied to impute the genotypes that facilitates statistical analyses and empowers of ML algorithms.

Acknowledgements. This study was implemented in the Basic Research Program's framework at the National Research University Higher School of Economics and the Laboratory of Models and Methods of Computational Pragmatics in 2020. The authors thank prof. Alexei Fedorov (University of Toledo College of Medicine, Ohio, USA) and prof. Svetlana Limborska (Institute of Molecular Genetics of National Research Centre "Kurchatov Institute", Moscow, Russia) for insightful discussions of the results obtained, and anonymous reviewers.

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

Funding. The study was funded by RFBR (Russian Foundation for Basic Research) according to the research project No 19-29-01151. The foundation had no role in study design, data collection and analysis, writing the manuscript, and decision to publish.

References

1. Bumgarner, R.: Overview of DNA microarrays: types, applications, and their future. Curr Protoc Mol Biol **Chapter 22** (Jan 2013) Unit 22.1.
2. Dehghan, A.: Genome-Wide Association Studies. Methods Mol. Biol. **1793** (2018) 37–49
3. Nicholls, H.L., John, C.R., Watson, D.S., Munroe, P.B., Barnes, M.R., Cabrera, C.P.: Reaching the End-Game for GWAS: Machine Learning Approaches for the Prioritization of Complex Disease Loci. Front Genet **11** (2020) 350
4. Tanay, A., Sharan, R., Shamir, R.: Discovering statistically significant biclusters in gene expression data. Bioinformatics **18**(suppl_1) (2002) S136–S144
5. Madeira, S.C., Oliveira, A.L.: Biclustering algorithms for biological data analysis: a survey. IEEE/ACM trans. on comp. biol. and bioinform. **1**(1) (2004) 24–45
6. Cheng, Y., Church, G.M.: Biclustering of expression data. In et al., P.E.B., ed.: Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology, 2000, AAAI (2000) 93–103
7. Tanay, A., Sharan, R., Shamir, R.: Biclustering algorithms: A survey. Handbook of computational molecular biology **9**(1-20) (2005) 122–124
8. Busygin, S., Prokopyev, O., Pardalos, P.M.: Biclustering in data mining. Computers & Operations Research **35**(9) (2008) 2964–2987
9. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. 1st edn. Springer-Verlag New York, Inc., Secaucus, NJ, USA (1999)
10. Besson, J., Robardet, C., Boulicaut, J., Rome, S.: Constraint-based concept mining and its application to microarray data analysis. Int. Data Anal. **9**(1) (2005) 59–82
11. Blachon, S., Pensa, R.G., Besson, J., Robardet, C., Boulicaut, J., Gandrillon, O.: Clustering formal concepts to discover biologically relevant knowledge from gene expression data. Silico Biol. **7**(4-5) (2007) 467–483
12. Kaytoue, M., Kuznetsov, S.O., Napoli, A., Duplessis, S.: Mining gene expression data with pattern structures in formal concept analysis. Inf. Sci. **181**(10) (2011) 1989–2001
13. Andrews, S., McLeod, K.: Gene co-expression in mouse embryo tissues. Int. J. Intell. Inf. Technol. **9**(4) (2013) 55–68
14. Ignatov, D.I., Kaminskaya, A.Y., Kuznetsov, S., Magizov, R.A.: Method of Biclusterzation Based on Object and Attribute Closures. In: Proc. of 8-th international Conference on Intellectualization of Information Processing (IIP 2011). Cyprus, Paphos, October 17–24, MAKS Press (2010) 140–143 (in Russian).
15. Ignatov, D.I., Kuznetsov, S.O., Poelmans, J.: Concept-based biclustering for internet advertisement. In: 2012 IEEE 12th International Conference on Data Mining Workshops, IEEE (2012) 123–130
16. Andrews, S.: In-close2, a high performance formal concept miner. In et al., S.A., ed.: Conceptual Structures for Discovering Knowledge - 19th Int. Conf. Conceptual Structures, ICCS 2011. Proceedings. Volume 6828 of LNCS., Springer (2011) 50–62
17. Yevtushenko, S.A.: System of data analysis "Concept Explorer". In: Proc. 7th National Conference on Artificial Intelligence (KII'00). (2000) 127–134

bioRxiv preprint doi: https://doi.org/10.1101/2020.10.22.349910; this version posted October 23, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY 4.0 International license.

18. Arnauld, A., Nicole, P.: La logique ou l'art de penser. Contenant outre les règles communes, plusieurs observations nouvelles, propres à former le jugement. Archives de la linguistique française. Ch. Savreuf, Guignart (1662)
19. Ignatov, D.: Models, Algorithms, and Software Tools of Biclustering Based on Closed Sets. PhD thesis, HSE University, Moscow (2010)
20. Prelic, A., Bleuler, S., Zimmermann, P., Wille, A., Bühlmann, P., Gruissem, W., Hennig, L., Thiele, L., Zitzler, E.: A systematic comparison and evaluation of biclustering methods for gene expression data. Bioinform. **22**(9) (2006) 1122–1129
21. Kuznetsov, S.O.: Mathematical aspects of concept analysis. Journal of Mathematical Sciences **80**(2) (1996) 1654–1698
22. Gnatyshak, D., Ignatov, D.I., Kuznetsov, S.O., Nourine, L.: A one-pass triclustering approach: Is there any room for big data? In Bertet, K., Rudolph, S., eds.: Proc. of the 11th Int. Conf. on Concept Lattices and Their Applications (CLA 2014). Volume 1252 of CEUR Workshop Proceedings., CEUR-WS.org (2014) 231–242
23. Leskovec, J., Rajaraman, A., Ullman, J.D.: Finding Similar Items. In: Mining of Massive Datasets, 3nd Ed. Cambridge University Press (2020) 73–134
24. Ignatov, D.I., Tochilkin, D., Egurnov, D.: Multimodal clustering of boolean tensors on mapreduce: Experiments revisited. In et al., D.C., ed.: Suppl. Proceedings of ICFCA 2019 Conference and Workshops. Volume 2378 of CEUR Workshop Proceedings., CEUR-WS.org (2019) 137–151
25. Codd, E.F.: A relational model of data for large shared data banks. Commun. ACM **13**(6) (June 1970) 377–387
26. Shetova, I.M., Timofeev, D.I., Shamalov, N.A., Bondarenko, E.A., Slominski, P.A., Limborskaia, S.A., Skvortsova, V.I.: The association between the DNA marker rs1842993 and risk for cardioembolic stroke in the Slavic population. Zh Nevrol Psikhiatr Im S S Korsakova **112**(3 Pt 2) (2012) 38–41
27. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9** (12 1997) 1735–80
28. Andrews, S.: Making Use of Empty Intersections to Improve the Performance of CbO-Type Algorithms. In Bertet, K., Borchmann, D., Cellier, P., Ferré, S., eds.: Formal Concept Analysis - 14th International Conference, ICFCA 2017. Proceedings. Volume 10308 of LNCS., Springer (2017) 56–71
29. Janostik, R., Konecny, J., Krajca, P.: LCM is well implemented cbo: Study of LCM from FCA point of view. In Valverde-Albacete, F.J., Trnecka, M., eds.: Proc. of the Fifteenth Int. Conf. on Concept Lattices and Their Applications, 2020. Volume 2668 of CEUR Workshop Proceedings., CEUR-WS.org (2020) 47–58
30. Naulaerts, S., Meysman, P., Bittremieux, W., Vu, T., Berghe, W.V., Goethals, B., Laukens, K.: A primer to frequent itemset mining for bioinformatics. Briefings Bioinform. **16**(2) (2015) 216–231
31. Serin, A., Vingron, M.: Debi: Discovering differentially expressed biclusters using a frequent itemset approach. Algorithms for Molecular Biology **6**(1) (2011) 18
32. Hu, Q., Imielinski, T.: ALPINE: progressive itemset mining with definite guarantees. In Chawla, N.V., Wang, W., eds.: Proceedings of the 2017 SIAM International Conference on Data Mining, SIAM (2017) 63–71
33. Chowdhury, H.A., Ahmed, H.A., Bhattacharyya, D.K., Kalita, J.K.: NCBI: A Novel Correlation Based Imputing Technique Using Biclustering. In: Computational Intelligence in Pattern Recognition. Springer (2020) 509–519