Frequent and Orthogonal Patterns Mining and its Application in Associative Classification

Leandro Souza Costa
Advisor: Wagner Meira Jr.

Department of Computer Science
Federal University of Minas Gerais

Master's Thesis Defense
April 18, 2008

Frequent Patterns

- Widely used in several applications, including association rules, classification, clustering, indexing, etc.;
- Minimize the result is still a challenge:
  - One of the properties of Frequent Patterns is anti-monotonicity.
- Minimize redundancy is another challenge:
  - There is limited work on finding those top-k patterns which demonstrate high-significance and low-redundancy simultaneously.

Frequent Patterns

Orthogonal Patterns

Our goal is to apply orthogonality in the frequent pattern mining problem, extracting a sub-set of patterns with high-significance and low-redundancy simultaneously.

Orthogonality Metrics

- It is necessary to define orthogonality metrics in order to evaluate sets;
- The Jaccard similarity coefficient complement applied to dataset coverage may be considered as an orthogonality metric applicable to two patterns:

$$D(p_1, p_2) = 1 - \frac{|TS(p_1) \cap TS(p_2)|}{|TS(p_1) \cup TS(p_2)|},$$

  where $TS(p)$ is the set of transactions covered by p.
- We are interested in metrics applicable to sets of different sizes.

Considering Pattern Structures

Motivation

Two patterns are orthogonal if they don't share items. We say that the patterns ABC and DEF are orthogonal, but ABC e CDE are not, since the item C is present in both patterns. This may be applied to bigger sets, for example, the patterns AB, CD e EF are orthogonal, but the patterns AB, BC e CD are not.

Considering Pattern Structures

- Let $\mathcal{I}$ be a set of items, $\mathcal{D}$ a dataset of transactions in $\mathcal{I}$, $\mathcal{F}$ the set of frequent patterns in $\mathcal{D}$, and $\mathcal{F}'$ a sub-set of $\mathcal{F}$ ($\mathcal{F}' \subseteq \mathcal{F}$);
- We say that $\mathcal{I}' \subseteq \mathcal{I}$ is the sub-set of items that appear in, at least, one of the patterns found in $\mathcal{F}'$;
- For each item $i \subseteq \mathcal{I}'$ is given a weight:

$$w_i = \frac{|\mathcal{F}'| - |\mathcal{F}'_i|}{|\mathcal{F}'| - 1},$$

where $\mathcal{F}'_i \subseteq \mathcal{F}'$ is the sub-set of patterns from $\mathcal{F}'$ that contains the item i;

Considering Pattern Structures

- The orthogonality based in patterns structure for the set is given by:

$$O_e = \frac{\sum_{i \subseteq \mathcal{I}'} w_i}{|\mathcal{I}'|}.$$

Considering Transaction Coverage

Motivation

Two patterns are orthogonal if they appear in different areas in dataset. In other words, if the sets of transactions covered by each pattern don't have elements in common.

Considering Transaction Coverage

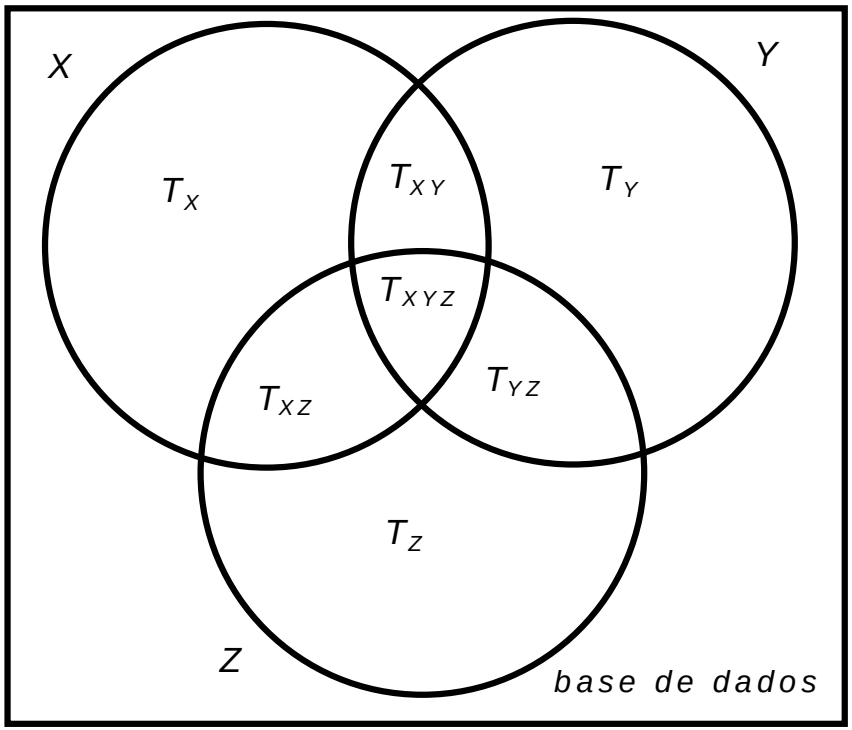


Figure: Visualization of Transaction Coverage in the Dataset

Considering Transaction Coverage

- Let $\mathcal{I}$ be a set of items, $\mathcal{D}$ a dataset of transactions in $\mathcal{I}$, $\mathcal{F}$ the set of frequent patterns in $\mathcal{D}$, and $\mathcal{F}'$ a sub-set of $\mathcal{F}$ ($\mathcal{F}' \subseteq \mathcal{F}$);
- We say that $\mathcal{D}' \subseteq \mathcal{D}$ is the sub-set of transactions covered by, at least, one of the patterns found in $\mathcal{F}'$;
- For each transaction $t \subseteq \mathcal{D}'$ is given a weight:

$$w_t = \frac{|\mathcal{F}'| - |\mathcal{F}'_t|}{|\mathcal{F}'| - 1},$$

where $\mathcal{F}'_t$ is the sub-set of patterns from $\mathcal{F}'$ that appear in the transaction t;

Considering Transaction Coverage

- The orthogonality based in transaction coverage is given by:

$$O_t = \frac{\sum_{t \subseteq \mathcal{D}'} w_t}{|\mathcal{D}'|}.$$

Considering Class Coverage

Motivation

Two patterns are orthogonal if they are found in transactions with distinct classes. In other words, the sets of transactions covered by the pattens may not have classes in common.

Considering Class Coverage

- Let $\mathcal{I}$ be a set of items, $\mathcal{D}$ a dataset of transactions in $\mathcal{I}$, $\mathcal{F}$ the set of frequent patterns in $\mathcal{D}$, $\mathcal{F}'$ a sub-set from $\mathcal{F}$ ($\mathcal{F}' \subseteq \mathcal{F}$) and $\mathcal{D}' \subseteq \mathcal{D}$ the sub-set of transactions covered by, at least, one of the patters found in $\mathcal{F}'$;
- We say that $\mathcal{C}$ is the set of classes associated to transactions in $\mathcal{D}$ and $\mathcal{C}' \subseteq \mathcal{C}$ the sub-set of classes associated to transactions in $\mathcal{D}'$;
- For each class $c \subseteq \mathcal{C}'$ is given a weight:

$$w_c = \frac{|\mathcal{F}'| - |\mathcal{F}'_c|}{|\mathcal{F}'| - 1},$$

where $\mathcal{F}'_c$ is the sub-set of patterns from $\mathcal{F}'$ that cover a number of transactions with class $c \subseteq \mathcal{C}'$ higher than 90% of the expected average;

Considering Class Coverage

- The orthogonality based in class coverage is given by:

$$O_c = \frac{\sum_{c \subseteq \mathcal{C}'} w_c}{|\mathcal{C}'|}.$$

Using Orthogonality in LAC

- For each test instance, LAC (*Lazy Associative Classifier*) builds a projection of training dataset with only transactions that share items with the instance;
- After that, the algorithm gets a set of frequent patterns according to a support given by the user;
- With these patterns, it generate association rules used to classify the instance.

Using Orthogonality in LAC

- In this work, orthogonality was used to extract, from the frequent patterns set, a sub-set of orthogonal patterns;
- The association rules were generated from this sub-set.

Heuristic to Get Orthogonal Sets

- Find the sub-set of patterns with best orthogonality metric is NP-Hard;
- It was developed a greedy heuristic that starts with an orthogonal set with two elements, and, iteratively, tries to obtain a new set with one more element adding candidate patterns and modifying the set in order to maximize the metric.

Heuristic to Get Orthogonal Sets

Require: $\mathcal{D}, \sigma$
1: $\mathcal{F} \leftarrow FindFrequentPatterns(\mathcal{D}, \sigma)$
2: $Sort(\mathcal{F})$
3: $\mathcal{O} \leftarrow GetFirstAvailablePattern(\mathcal{F})$
4: **repeat**
5: $\quad rate \leftarrow GetOrthogonalityRate(\mathcal{O})$
6: $\quad \mathcal{O}_c \leftarrow GetNextCandidateSet(\mathcal{O}, \mathcal{F})$
7: $\quad rate_c = GetOrthogonalityRate(\mathcal{O}_c)$
8: $\quad$ **if** $rate_c \geq rate$ **then**
9: $\quad\quad \mathcal{O} \leftarrow \mathcal{O}_c$
10: $\quad$ **end if**
11: **until** $rate_c < rate$
12: $\mathcal{R} \leftarrow \mathcal{O}$

Algorithm 1: OLAC

Heuristic to Get Orthogonal Sets

Require: $\mathcal{O}, \mathcal{F}$
1: $\mathcal{O}_c \leftarrow \mathcal{O} \cup GetFirstAvailablePattern(\mathcal{F})$
2: $rate_c = GetOrthogonalityRate(\mathcal{O}_c)$
3: **for** $P \in \mathcal{F}, P \notin \mathcal{O}_c$ **do**
4: $\quad S \leftarrow GetMoreSimilar(\mathcal{O}_c, P)$
5: $\quad \mathcal{O}_c \leftarrow \mathcal{O}_c \cup P \setminus S$
6: $\quad rate_{try} = GetRate(\mathcal{O}_c)$
7: $\quad$ **if** $rate_{try} > rate_c$ **then**
8: $\quad\quad rate_c \leftarrow rate_{try}$
9: $\quad$ **else**
10: $\quad\quad \mathcal{O}_c \leftarrow \mathcal{O}_c \cup S \setminus P$
11: $\quad$ **end if**
12: **end for**
13: **return** $\mathcal{O}_c$

Algorithm 2: OLAC - GetNextCandidateSet

Contextualizing

The **ORIGAMI** is a graph mining algorithm found in literature, where the authors introduce a definition for α-orthogonality and β-representativity, and present a new paradigm for mining a summary representation of the set of frequent graphs by considering the distances in the pattern space.

α-orthogonality Definition

- Let $\mathcal{F}$ be the set of all frequent sub-graphs of a collection;
- Let $sim : \mathcal{F} \times \mathcal{F} \rightarrow [0, 1]$ be a symmetric binary function that returns the *similarity* between two graphs;
- Given a collection of graphs $\mathcal{G}$, and an upper-bound for similarity $\alpha \in [0, 1]$, we say that the sub-set of graphs $\mathcal{R} \subseteq \mathcal{G}$ is α-**orthogonal** with respect to $\mathcal{G}$ if, and only if, for any $G_a, G_b \in \mathcal{R}, sim(G_a, G_b) \leq \alpha$ and for any $G_i \in \mathcal{G} \backslash \mathcal{R}$ there exists a $G_j \in \mathcal{R}, sim(G_i, G_j) > \alpha$;

α-orthogonality Definition

- Given a collection of graphs $\mathcal{G}$, an α-orthogonal set $\mathcal{R} \subseteq \mathcal{G}$ and a lower-bound for similarity $\beta \in [0, 1]$, we say that $\mathcal{R}$ **represents** a graph $G \in \mathcal{G}$ if there exists some $G_a \in \mathcal{R}$ such that $sim(G_a, G) \geq \beta$. Given $\Upsilon(\mathcal{R}, \mathcal{G}) = \{G \in \mathcal{G} : \exists G_a \in \mathcal{R}, sim(G, G_a) \geq \beta\}$, we say that $\mathcal{R}$ is a β-representative set for $\Upsilon(\mathcal{R}, \mathcal{G})$;

α-orthogonality Definition

- Given a collection of graphs $\mathcal{G}$, an α-orthogonal set and its β-representative set $\mathcal{R}$, we call **residue** of $\mathcal{R}$ the set of patterns not represented in $\mathcal{G}$, given as $\Delta(\mathcal{R},\mathcal{G}) = \mathcal{G} \setminus \{\mathcal{R} \cup \Upsilon(\mathcal{R},\mathcal{G})\}$, the *residue* of $\mathcal{R}$ is defined as the cardinality of its residue-set $|\Delta(\mathcal{R},\mathcal{G})|$. Finaly, we define the average residue similarity for $\mathcal{R}$ as follows: $ars(\mathcal{R},\mathcal{G}) = \frac{\sum_{G_b \in \Delta(\mathcal{R},\mathcal{G})} max_{G_a \in \mathcal{R}}\{sim(G_a,G_b)\}}{|\Delta(\mathcal{R},\mathcal{G})|}$.

α-orthogonality Definition

Objective

The goal is to find sets of α-orthogonal and β-representative graphs related to the maximal sub-graphs set $\mathcal{M}$.

The ORIGAMI Algorithm

Require: $\mathcal{D}, \sigma, \alpha, \beta$
1: $EM \leftarrow EdgeMap(\mathcal{D})$
2: $\mathcal{F}_1 \leftarrow FindFrequentEdges(\mathcal{D}, \sigma)$
3: $\widehat{\mathcal{M}} \leftarrow 0$
4: **while** $\neg StopCondition()$ **do**
5: $M \leftarrow RandomMaximalGraph(\mathcal{D}, \mathcal{F}_1, EM, \sigma)$
6: $\widehat{\mathcal{M}} \leftarrow \widehat{\mathcal{M}} \cup M$
7: **end while**
8: $\mathcal{R} \leftarrow OrthogonalRepresentativeSets(\widehat{\mathcal{M}}, \alpha, \beta)$

Algorithm 3: ORIGAMI

The ORIGAMI Adaptation

- It was implemented an adaptation of ORIGAMI for the Associative Classification Problem;
- It was implemented a heuristic to get maximal patterns based on the article;
- It was implemented a heuristic to get orthogonal patterns based on the article.

Heuristic to Get Maximal Patterns

- The algorithm starts with an empty result-set;
- On each iteration, it tries to get the largest frequent pattern possible, selection items randomly;
  - If the algorithm chooses an item that is already in the pattern, or that makes the pattern infrequent, a counter is decremented;
  - The stop-condition for the candidate pattern generation is that the number of wrong chooses for the random item should be, at most, equal to the test instance size.

Heuristic to Get Maximal Patterns

- After generate a new maximal pattern, the algorithm tries to insert it in the result-set;
  - If the pattern is already in the set, the algorithm decrements a second counter of tries;
  - The stop condition for the maximal patterns set generation is that number of candidate patterns generated that are not maximal or are already in the set should be, at most, equal to the test instance size.

Heuristic to Get the Orthogonal Patterns Set

- The algorithm starts with a residue equal to 0 (zero);
- On each iteration, it tries to get a new orthogonal set selecting, randomly, maximal patterns found in the first step of the algorithm, and adding them to the result-set;
  - If the algorithm selects a pattern that is already in the set or does not have similarity lower than α with all the other patterns of the set, the algorithm decrements a counter of tries;
  - The stop-condition is that the maximum number of wrong patterns chosen should be, at most, equal to the number of maximal patterns.

Heuristic to Get the Orthogonal Patterns Set

- After generate a new orthogonal set, the algorithm calculate its residue-value;
- If the residue is the best until now, it consider the set found as the new best result;
- The stop-condition for the algorithm is that the maximum number of orthogonal sets generated that don't improve the result should be, at most, equal to the number of maximal patterns.

The Applicative **olac**

- In the applicative **olac** we have the implementation of three different approaches of an association rules based classifier:
  - The LAC (*Lazy Associative Classifier*) approach, that is the *lazy* strategy in his original version(and non-orthogonal);
  - The OLAC (*Orthogonal Lazy Associative Classifier*) approach, that is a variant of the *lazy* strategy considering orthogonality;
  - The ORIGAMI, that is the adaptation presented for the ORIGAMI strategy.

Methodology

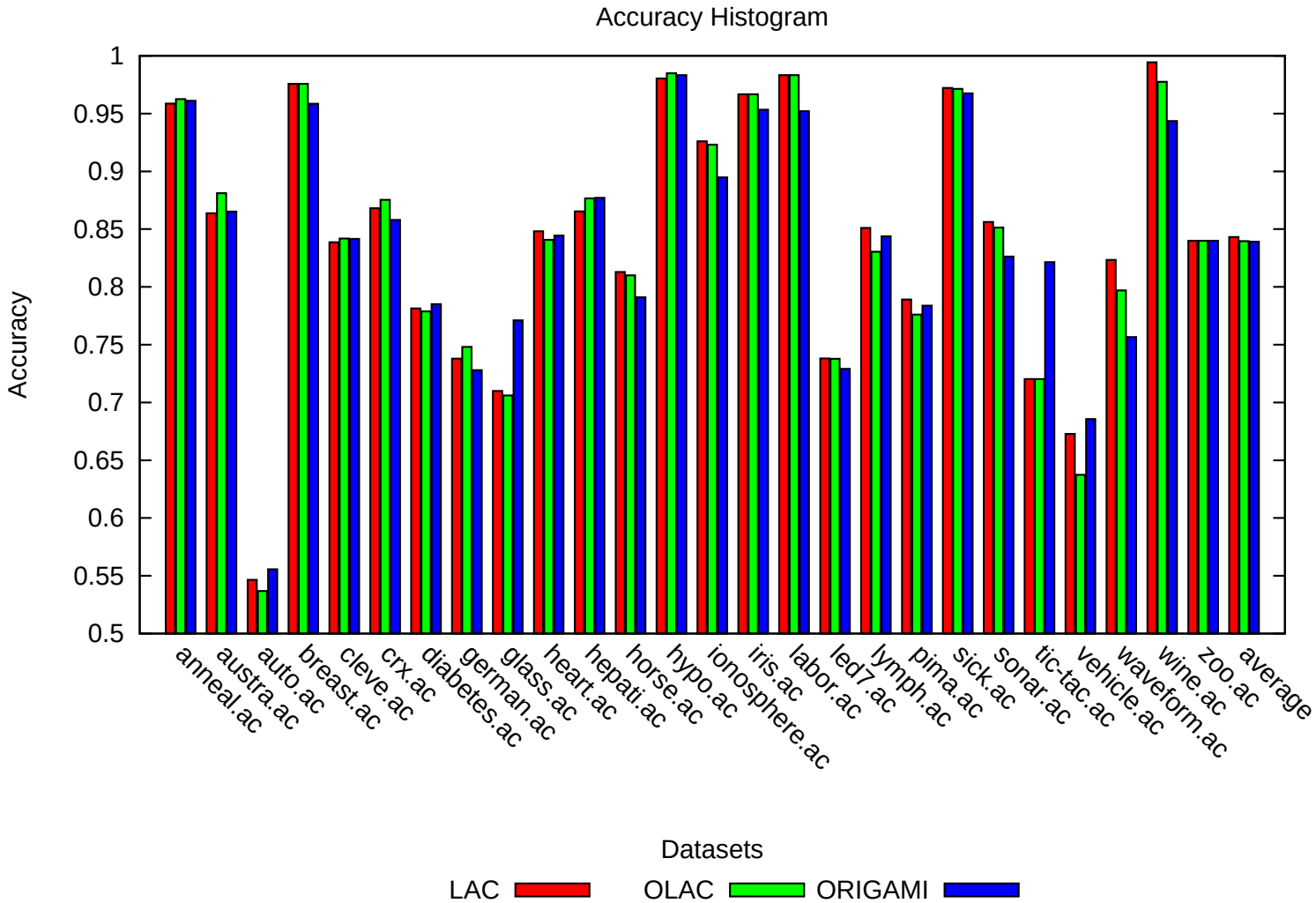
- We used 26 datasets from **UCI** (*UC Irvine Machine Learning Repository*);
- We used 10-fold cross-validation and the final results of each experiment represent the average of the ten runs.

Methodology

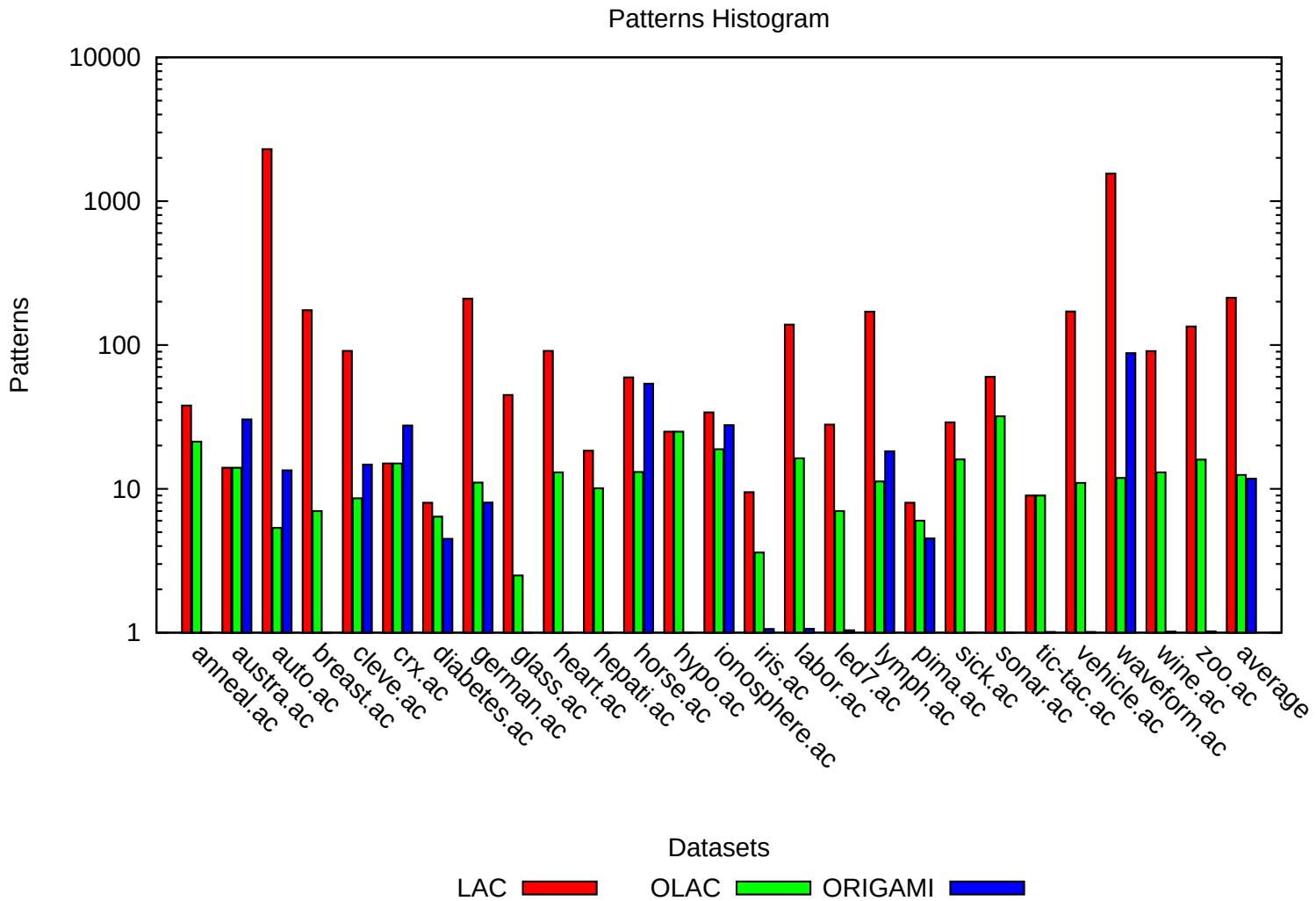
Parameters	Values
support	$\{0.0001, 0.001, 0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.99, 1\}$
confidence	$\{0.0001, 0.001, 0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.99, 1\}$
min-num-rules	$\{1\}$
max-num-rank-rules	$\{1, 10, 100, 1000, 10000, 100000, 1000000\}$
min-rule-len	$\{1\}$
max-rule-len	$\{1, 2, 3\}$
rule-measure	$\{s, c, j, k, o, n, e, p, l, i, v\}$
orth-metric	$\{e, c, l, a\}$
orth-method	$\{s, p\}$
orth-pat-ordering	$\{s, r, i, z, n\}$
origami-alpha	$\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$
origami-beta	$\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$

Table: Parameters Used for All Approaches

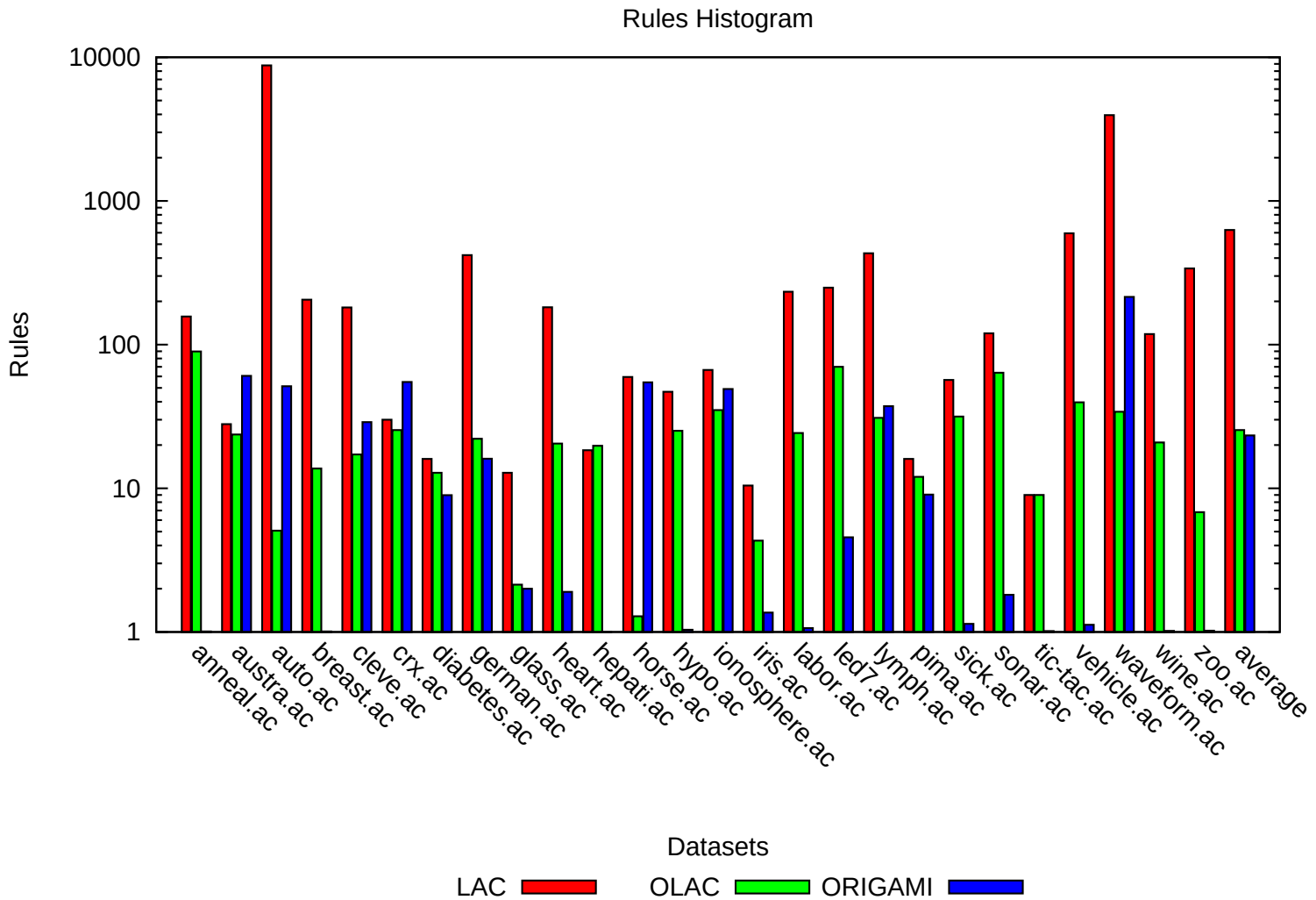
Better Results for Each Dataset



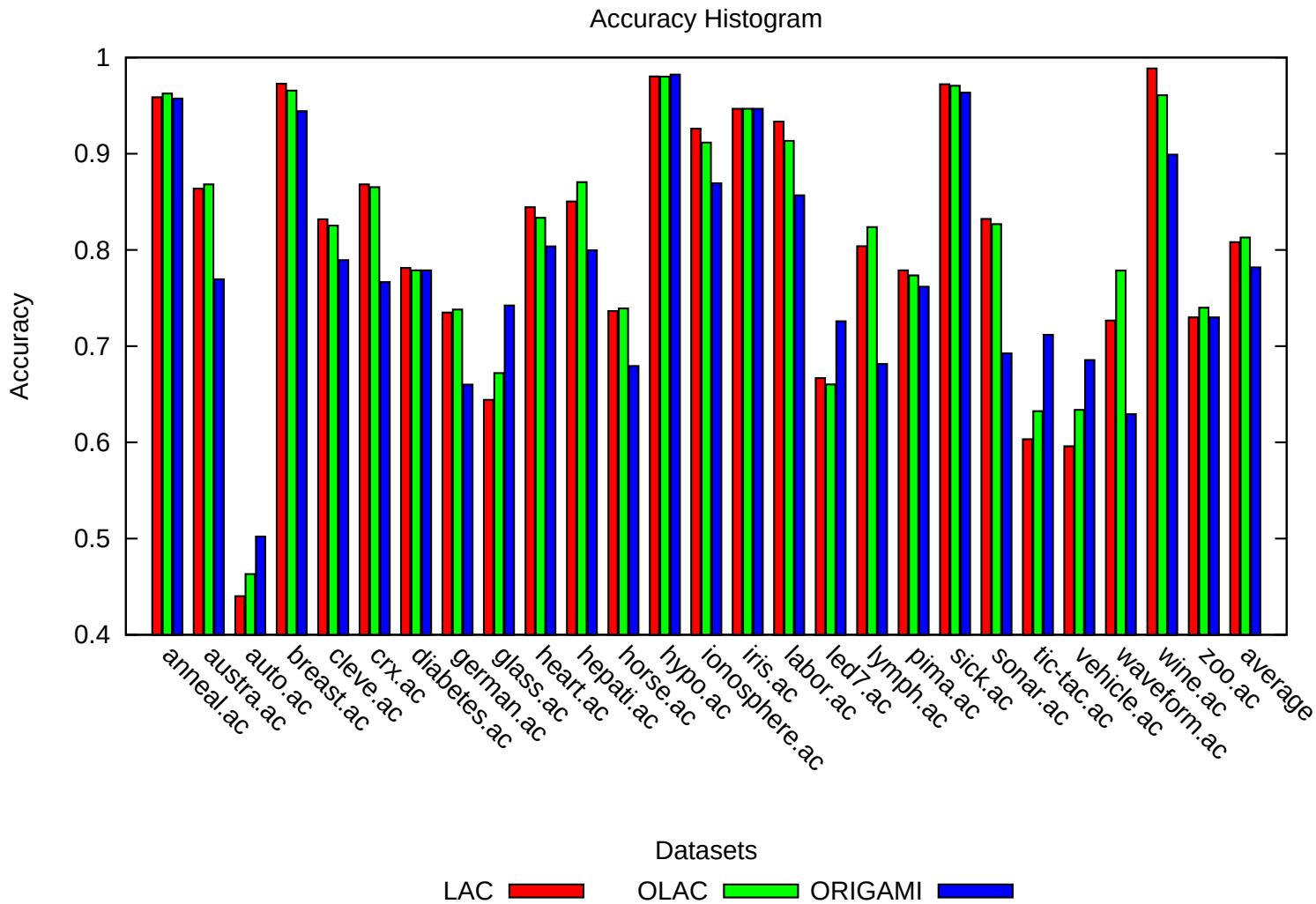
Better Results for Each Dataset



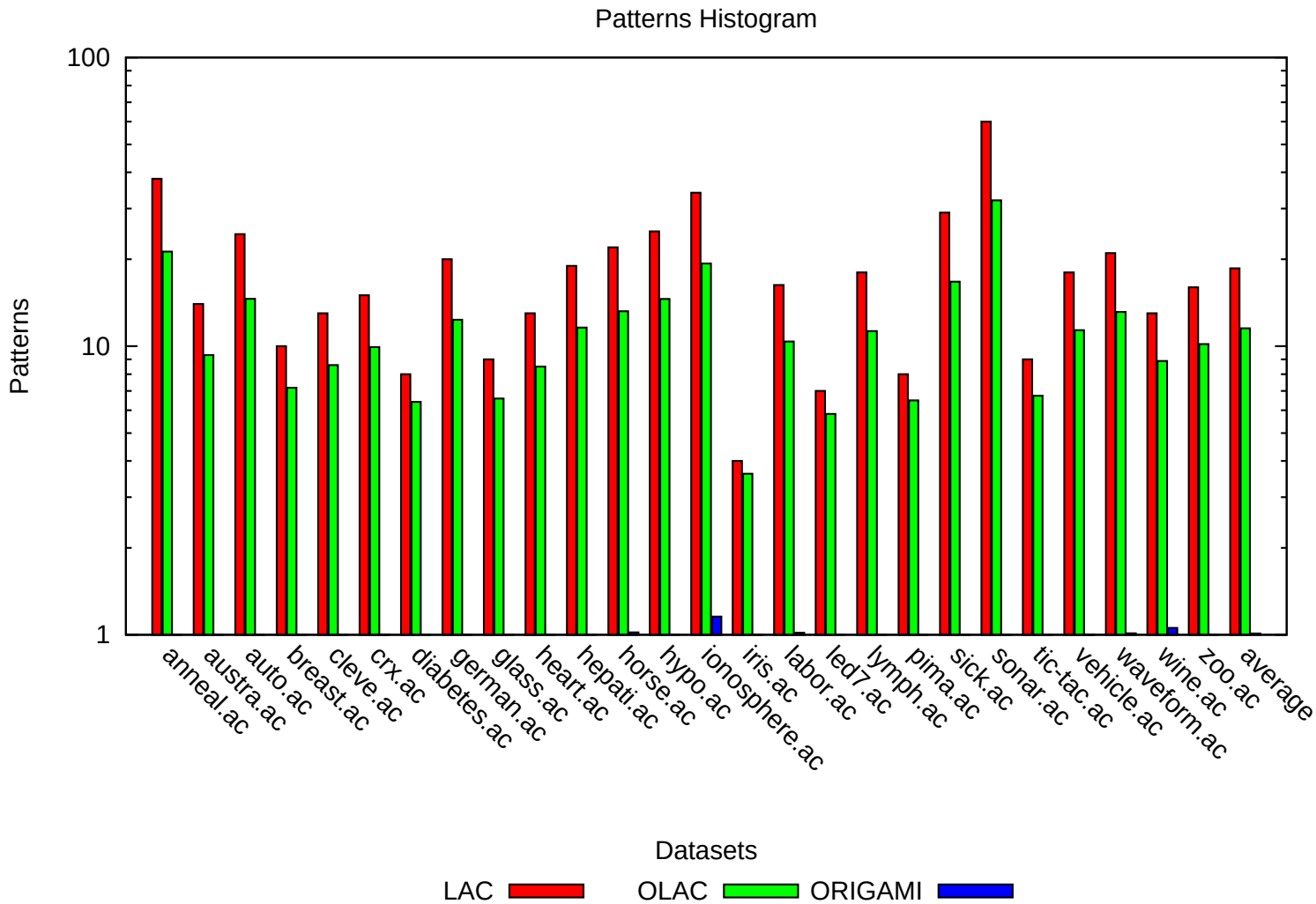
Better Results for Each Dataset



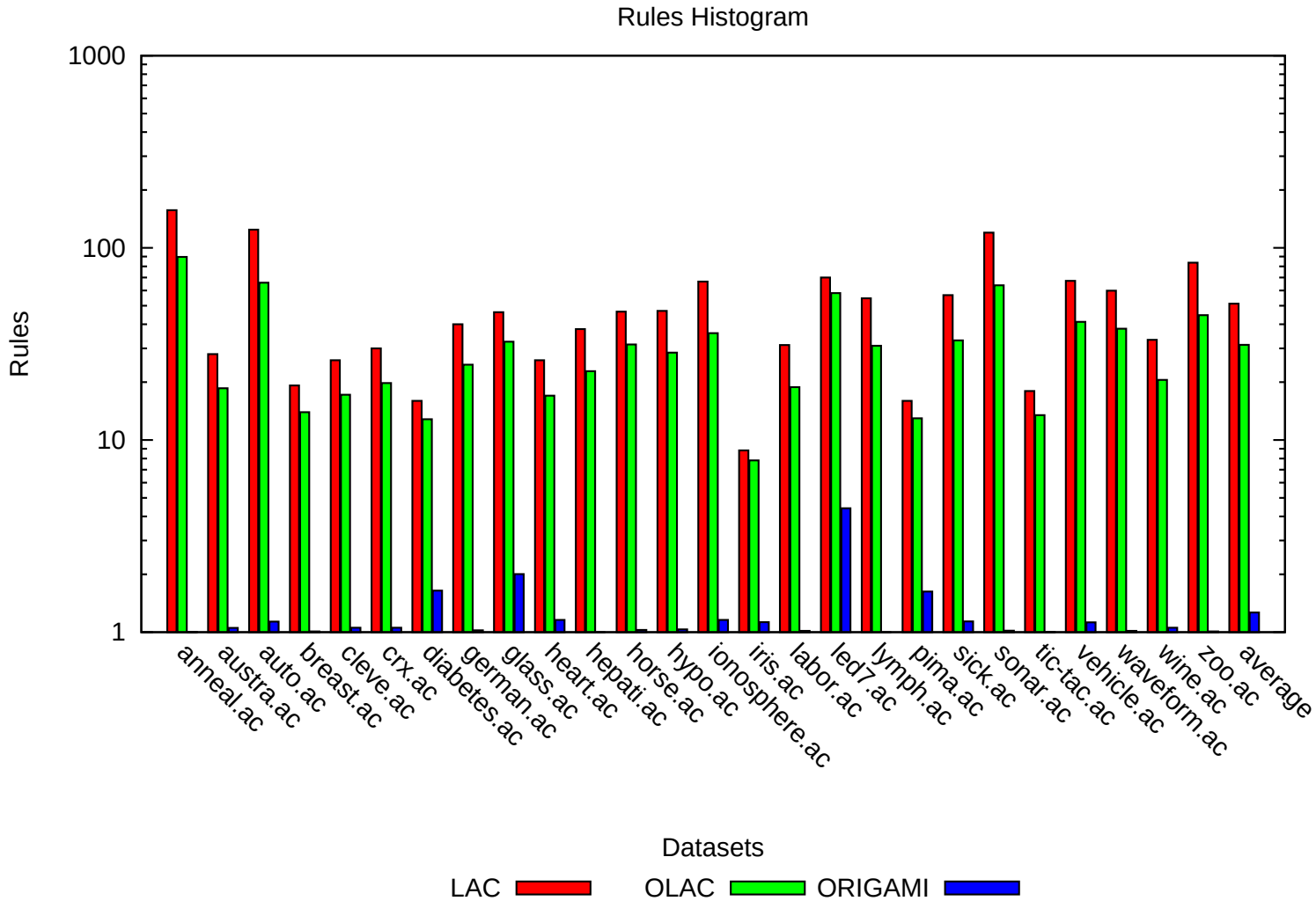
Better Results in Average for All Datasets



Better Results in Average for All Datasets



Better Results in Average for All Datasets



Execution Parameters

	LAC	OLAC	ORIGAMI
support	0.001	0.0001	0.0001
confidence	0.01	0.0001	0.0001
min-num-rules	1	1	1
max-num-rank-rules	1000	100	10
min-rule-len	1	1	-
max-rule-len	1	2	-
rule-measure	n	n	c
orth-metric	-	s	s
orth-method	-	s	-
orth-pat-ordering	-	s	-
origami-alpha	-	-	0.1
origami-beta	-	-	0.8

Table: Best Parameters for Each Run

Results obtained with LAC using the best parameters for OLAC

Patterns: 249.03
Rules: 628.12
Accuracy: 0.54

Comparing the Results

Datasets	OLAC & LAC	OLAC & ¬ LAC	¬ OLAC & LAC	¬ OLAC & ¬ LAC
anneal.ac	95.11	1.13	0.75	3.01
austra.ac	84.93	1.88	1.45	11.74
auto.ac	39.51	6.83	4.39	49.27
breast.ac	96.28	0.29	1.00	2.43
cleve.ac	81.19	1.32	1.98	15.51
crx.ac	84.93	1.59	1.88	11.59
⋮	⋮	⋮	⋮	⋮
wine.ac	96.07	0.00	2.81	1.12
zoo.ac	73.27	0.99	0.00	25.74
average	78.91	2.39	1.90	16.80

Table: Comparison between LAC and OLAC (number of correct and wrong classifications)

Accuracy

- With orthogonality based approaches we got good results, very close to the classical approach one:
  - Considering the best parameters for each dataset, the values for accuracies obtained by LAC, OLAC and ORIGAMI were, respectively, 0.843, 0.840 e 0.839;
  - Considering the best parameters for the average of the results, the values for accuracy obtained by LAC, OLAC and ORIGAMI were, respectively, 0.808, 0.813 e 0.782.

Patterns

- The number of patterns used to generate the rules by the orthogonal approaches were lower than by the classical approach:
  - Considering the best parameters for each dataset, the number of patters used by LAC, OLAC and ORIGAMI were, respectively, 213, 12 e 12;
  - Considering the best parameters for the average of the results, the number of patterns used by LAC, OLAC and ORIGAMI were, respectively, 19, 12 e 1.

Rules

- The number of rules generated by the orthogonal approaches were much lower than by the classical approach:
  - Considering the best parameters for each dataset, the number of rules generated by LAC, OLAC and ORIGAMI were, respectively, 628, 25 e 23;
  - Considering the best parameters for the average of the results, the number of rules generated by LAC, OLAC and ORIGAMI were, respectively, 51, 31 e 1.

Other Results

- The orthogonality metric based in pattern structure obtained the best results;
- The association rule metrics that obtained the best results were conviction by LAC and OLAC and confidence by ORIGAMI;
- Most of fails found in classification based in orthogonality was not caused by the low orthogonality metric. They were found because sometimes, the different patterns used, induced the results to the wrong class.

Next Steps

- Use orthogonality in another points of a classification algorithm;
- Research for new heuristics of orthogonal patterns extraction, in order to improve performance;
- Research for new algorithms of frequent pattern mining that consider orthogonality while generating the frequent patterns;
- Use of a hybrid approach OLAC-ORIGAMI.

Questions?