Tree Based Advanced Relative Expression Analysis

Marcin Czajkowski(✉), Krzysztof Jurczuk, and Marek Kretowski

Faculty of Computer Science, Bialystok University of Technology,
Wiejska 45a, 15-351 Bialystok, Poland
{m.czajkowski,k.jurczuk,m.kretowski}@pb.edu.pl

Abstract. This paper presents a new concept for biomarker discovery and gene expression data classification that rises from the Relative Expression Analysis (RXA). The basic idea of RXA is to focus on simple ordering relationships between the expression of small sets of genes rather than their raw values. We propose a paradigm shift as we extend RXA concept to tree-based Advanced Relative Expression Analysis (ARXA). The main contribution is a decision tree with splitting nodes that consider relative fraction comparisons between multiple gene pairs. In addition, to face the enormous computational complexity of RXA, the most time-consuming part which is scoring all possible gene pairs in each splitting node is parallelized using GPU. This way the algorithm allows searching for more tailored interactions between sub-groups of genes in a reasonable time. Experiments carried out on 8 cancer-related datasets show not only significant improvement in accuracy and speed of our approach in comparison to various RXA solutions but also new interesting patterns between subgroups of genes.

Keywords: Relative Expression Analysis · Decision trees · Gene expression data

1 Introduction

High-throughput technologies are generating large volumes of omics data at an unprecedented rate [11]. Traditional machine learning algorithms have been quite successful in automatically identifying complex patterns. Unfortunately, the overwhelming majority of systems focus on complex decision rules that are obstacles to mature applications [2]. Currently developed classification methods to biological data are usually designed for other purposes, such as improving statistical learning or applications to vision and speech, with little emphasis on transparency. The complexity of the decision rules that emerge from standard machine learning impedes biological understanding.

Comprehensive analysis poses new computational challenges and specialized computational approaches are required to effectively and efficiently carry out the predictions using biomedical data. It can be observed that there is a strong

V. V. Krzhizhanovskaya et al. (Eds.): ICCS 2020, LNCS 12139, pp. 496–510, 2020.
https://doi.org/10.1007/978-3-030-50420-5_37

need for such 'white box' models which may actually help in understanding and identifying relationships between specific features and improve biomarker discovery [22]. One of the solutions is Relative Expression Analysis (RXA) which is a powerful collection of easily interpretable computational methods for gene expression data classification. It focuses on finding interactions among a small collection of genes by studying relative ordering of their expressions rather than their raw values.

The most significant novelty in the proposed paper is the new, much more general concept of gene-gene interaction within RXA called Advanced Relative Expression Analysis (ARXA). By introducing relative fraction comparison between multiple gene pairs within a single individual we can detect not only the ordering shifts between the genes but also the percent changes in their relations. In addition, we have applied this strategy to the splitting nodes of the Decision Trees (DTs) in order to detect hierarchical relations as well. The traditional DTs have a long history in predictive modeling [17] but result in insufficient accuracy when applied to gene expression data. By combining and extending these two 'white box' algorithms we managed to significantly improve the classification accuracy on several publicly available gene expression datasets. Finally, to face up the enormous computational complexity which rises from an exhaustive analysis of all possible pairs of genes, we designed and implemented a graphic processing unit (GPU)-based parallelization.

The next section provides our motivations and a brief background on RXA, DTs and GPGPU parallelization. Section 3 describes in detail our concept of tree-based ARXA and its GPU-based implementation. In Sect. 4, experimental validation is performed and in the last section, the paper is concluded and possible future works are outlined.

2 Background

While great progress has been achieved in what entails biodata analysis, most of the research effort tends to focus almost exclusively on the prediction accuracy of core data mining tasks (e.g., classification and regression), and far less effort has gone into the crucial task of knowledge discovery itself. Specifically, the rules generated by nearly all standard, off-the-shelf techniques applied to genomics data [1], such as neural networks, random forests, SVMs, and linear discriminant analysis usually involve nonlinear functions of hundreds or thousands of genes, many parameters, and are therefore too complex to characterize mechanistically. Currently, deep learning approaches have been getting attention [23] as they can better recognize complex features through representation learning with multiple layers, and can facilitate the integrative analysis by effectively addressing the challenges discussed above. However, we know very little about how such results are derived internally.

In contrast to data mining systems, statistical methods for analyzing high-dimensional biomolecular data generated with high-throughput technologies permeate the literature in computational biology. Those analyses have uncovered

a great deal of information about biological processes [1], such as important mutations and lists of "marker genes" associated with common diseases and key interactions in transcriptional regulation. However, the analysis is often limited to a relatively small number of features thus a small set of informative variables needs to be identified out of a large number (or dimension) of candidates.

2.1 Relative Expression Analysis

The process of biomarker discovery and characterization provides opportunities for more sophisticated solutions that integrate statistical, data mining and expert knowledge-based approaches. One of the ideas for the gene expression data is the concept of Relative Expression Analysis which focuses on testing relative expression ordering among a small number of transcripts. In the pioneering research from 2004, a Top Scoring Pair (TSP) method is proposed [10] which is a straightforward prediction rule based on the RXA concept that utilizes building blocks of rank-altered gene pairs in case and control comparison. Such pairs of genes can be viewed as "biological switches" which can be directly related to regulatory "motifs" or other properties of transcriptional networks. The discriminating power of each pair of genes i, j was measured by the absolute difference between the probabilities P_{ij} of the event that gene i is expressed more than gene j in the two classes.

Let x_i and x_j be the expression values of two different genes from available set of genes and there are only two classes: *normal* and *disease*. At first, algorithm calculates the probability of the relation $x_i < x_j$ between those two genes in the objects from the same class: $P_{ij}(normal) = Prob(x_i < x_j | Y = normal)$ and $P_{ij}(disease) = Prob(x_i < x_j | Y = disease)$, where Y denotes the class of the objects. Next, the score for this pair of genes (x_i, x_j) is calculated: $\Delta_{ij} = |P_{ij}(normal) - P_{ij}(disease)|$. This procedure is repeated for all distinct pairs of genes and the pair with the highest score becomes titled top scoring pair. In the case of a draw, a secondary ranking that bases on raw genes expression differences in each class is used [24].

The k-TSP algorithm [24] is one of the first extensions of the TSP solution. It focuses on increasing the number of pairs in the prediction model and applies no more than k top scoring disjoint gene pairs with the highest score, where the parameter k is determined by the internal cross-validation. This method was later combined with a top-down induced decision tree in an algorithm called TSPDT [5]. In this hybrid solution, each non-terminal node of the tree divides instances according to a splitting rule that is based on TSP or k-TSP accuracy.

Different approaches for the TSP extension focus on the relationships between more than two genes. Algorithms Top Scoring Triplet (TST) [19] and Top Scoring N (TSN) [21] analyze all possible ordering relationships between the genes, however, the general concept of TSP is retained. One of the first heuristic approaches that applied the RXA concept was the evolutionary algorithm called EvoTSP [6] where the authors proposed an evolutionary search for the TSP-like rules. This approach, later extended with additional features ranking in REHA [7] showed that evolutionary search is a good alternative to the traditional RXA

algorithms. Finally, there are many variations of the TSP-family solutions that involve changes in ranking calculations, we can distinguish AUCTSP classifier that uses the ROC curve [15] or VH-k-TSP [12] that focuses on vertical and horizontal genes relations. What's more, the strength and simplicity of RXA has been recognized outside genomics data and is being successfully used in the proteomic and metabolomic analysis.

2.2 Decision Trees

The popularity of Decision trees (DTs) [17] can be explained by its ease of use, speed of classification and effectiveness. In addition, the hierarchical structure of the tree, where appropriate tests are applied successively from one node to the next, closely resembles the human way of making decisions. DT has a knowledge representation structure made up of nodes and branches, where: each internal node is associated with a test on one or more attributes; each branch represents the test result, and each leaf (terminal node) is designed by a class label. Induction of optimal DT is a known NP-complete problem [13]. As a consequence, practical DT learning algorithms must be heuristically enhanced.

DT represents a white-box approach and has considerable potential for biodata research and scientific modeling of the underlying processes. Unfortunately, there are not so many new solutions in the literature that focus on the classification of genomic data with comprehensive DT models. Existing attempts showed that decision tree algorithms often induce classifiers with the inferior predictive performance [8] and one of the alternatives is combining DTs with evolutionary approaches [18]. However, nowadays, much more interest is given in trees as sub-learners of an ensemble learning approach, such as Random Forests. These solutions alleviate the problem of low accuracy by averaging or adaptive merging of multiple trees. However, when modeling is aimed at understanding basic processes, such methods are not so useful because they generate more complex and less understandable models.

2.3 GPGPU Parallelization

Recent research on the parallelization of various evolutionary computation methods has seemed to focus on GPUs as the implementation platform. The popularity of GPUs results from their general availability, relatively low cost, and high computational power. Parallel evaluation of instances is considered much more scalable with respect to the size of the dataset than a population approach. It focuses on gradually distributing the entire dataset among the local memories of all processors.

In the literature, we can find a few systems where GPU-based parallelization of the induction of DTs was examined. One of the propositions was CUDT [20] that parallelized the top-down induction process. In each internal node, in order to find the best locally optimal splits, the attributes are processed in parallel. With this approach, the authors managed to reduce the induction time of a typical decision tree from 5 to 55 times when compared with the traditional

CPU version. The GPGPU parallelization was also introduced to evolutionary induced DTs [14]. In the case of RXA there exists also research considering GPU parallelization. In [21] authors managed to speed up calculations of basic TSP and TST solutions by two orders of magnitude.

2.4 Motivation and Contribution

Most recently proposed data mining methods for genomic data generate complex rules that constrain the process of uncovering new biological understanding that, after all, is the ultimate goal of data-driven biology. However, it is not enough to simply produce good outcomes but to provide logical reasoning just as clinicians do for medical treatments. In addition, whereas the need for statistical methods in biomedicine continues to grow, the effects on the clinical practice of existing classifiers based on genomic data are widely acknowledged to remain limited. One of the barriers is the study-to-study diversity in reported prediction accuracies, problems with data integration and the unfavorable ratio of the sample size to the number of potential biomarkers. The main TSP advances for gene expression data analysis are:

- the method is non-parametric since the method is constructed based on the relative ranking of gene pairs;
- the method is based on one or a few gene pairs. The biological interpretation of the model and the translational application are more straightforward;
- researchers have repeatedly found that the family of TSP algorithms provides good prediction performance in many transcriptomic data [1].

The main drawback of TSP-family algorithms is that they are focused only on gene expression data and can only be used locally and on a small scale. There are two reasons why: (i) focusing on simple "biological switches" may not work where more advanced relations occur; (ii) exhaustive search performed by TSP-solutions has enormous computational complexity which strongly limits the number of features and inter-relations that can be analyzed [16]. In our previous research, we managed to partially address both issues separately by using decision trees with TSP splits [5] and/or evolutionary algorithms [6,8].

Nonetheless, the true core of the problem (i) still remains as deliberately replacing the raw data values with the ordering relationships between the features obviously causes loss of potentially important information. Let us hypothetically assume that for some tested sample two genes $X1$ and $X2$ can discriminate normal class from cancer one. Figure 1 shows three simple scenarios (a), (b), (c) together with the outcome of RXA. The example (a) shows the ultimate goal of RXA as it illustrates the perfect "top scoring pair". We can observe that the ordering relations between genes X1 and X2 is opposite in different classes among all the instances. Unfortunately, RXA outcome for scenario (a) and scenario (b) would be the same as in both cases "biological switch" occurs, at least in theory. However, when we look at the expression image and the chart axis we see, that in fact, X1 and X2 have low expression values in both classes. Such

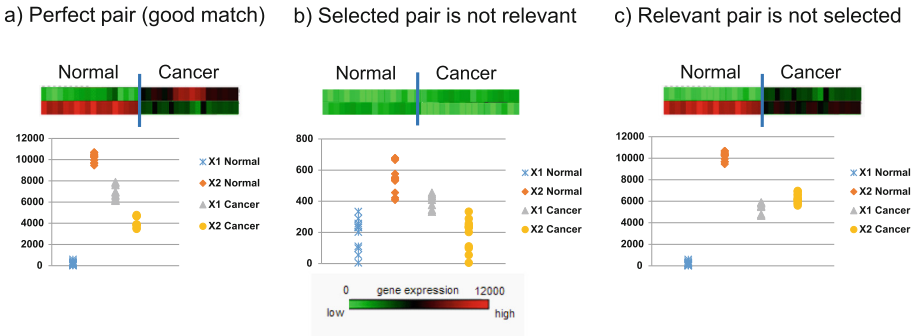

Fig. 1. Comparison of possible relations between two genes X1 and X2 in normal and cancer samples together with RXA outcome

selected pair is not relevant despite the fact that it will be promoted by RXA. An even worse scenario is presented in example (c) of Fig. 1 where undoubtedly relevant pair is not considered by the RXA despite significant variations in the expression values of genes in normal and cancer classes. As the simple ordering relationship between X1 and X2 is not changed between both classes, currently available RXA-family algorithms will never mark these genes as "top scoring pair". It might choose them with together other genes, by making multiple top pairs, but besides potential interpretability problems, lower accuracy issues may also arise. The issue (i) is also aggravated by the second (ii) problem which is computational complexity equals $O(T * k * M * N^Z)$, where T is the size of DT, k is the number of top-scoring groups, M is the number of instances, N is the number of analyzed genes and Z is the size of a group of genes which ordering relationships are searched. Sequential calculation of all possible gene pairs or gene groups strongly limits the number of genes and inter-relations that can be analyzed in a reasonable time and at the same time limits the number of features having similar expression values and being opposite to each other in different classes.

In this paper we propose the comparison of percentage changes of gene expressions in pairs among different classes. Within our approach the algorithm can easily ignore not relevant pairs (scenario (b)), select relevant ones (scenario (a) and (c)) and work even with smaller number of features. It should be noted that our new weight approach is even more computationally demanding than a typical RXA which will be shown in the following section. That is why we designed the GPU parallelization as an alternative to the above-mentioned evolutionary approaches to enable much faster RXA calculations.

3 Tree Based Advanced Relative Expression Analysis

The overall structure of the proposed solution is based on a typical top-down induced [17] binary classification tree. The greedy search starts with the root

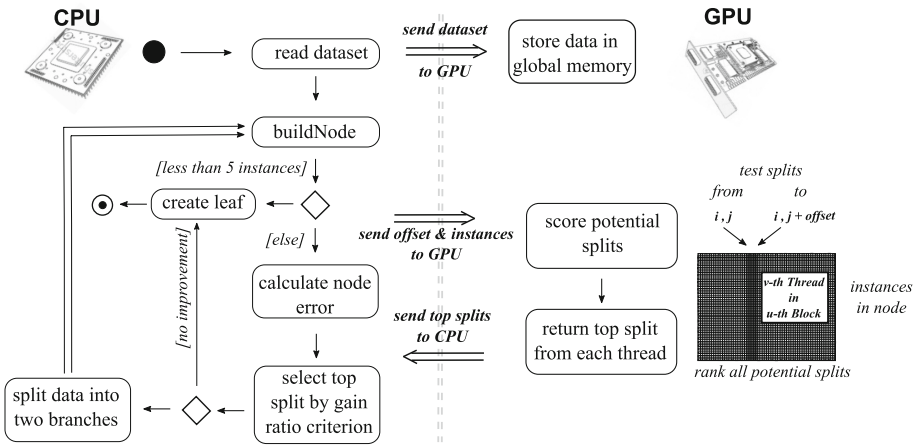

Fig. 2. General flowchart of a GPU-accelerated ARXA

node, where the locally optimal split (test) applies the new rank concept (denoted as ARXA). Then the training instances are redirected to the newly created nodes and this process is repeated for each node until the stop condition is met. Currently, we do not apply any form of post-pruning due to the small sample sizes, however, it should be considered in the future to improve the generalizing power of the predictive model.

The general flowchart of our GPU-accelerated ARXA is illustrated in Fig. 2. Each internal node contains information about a relation of pairs of genes that is later used to constitute the split. The basic idea to analyze relations within a single instance alike in RXA solutions, however, there are fundamental differences in scoring the collections of genes. It can be seen that the DT induction is run in a sequential manner on a CPU, and the most time-consuming operation which is scoring all potential splits is performed in parallel on a GPU. This way, the parallelization does not affect the behavior of the original algorithm.

Let us consider a gene expression microarray dataset consisting of N genes and M samples. Let the data be represented as an $N \times M$ matrix in which an expression value of u-th gene from v-th sample is denoted as x_{uv}. Each row represents observation of a particular gene X_u over M training samples, and each column Y_v represents a sample v described by the N genes. Let's for the simplicity of presentation assume that there are only two classes: C_1 and C_2, and instances with indexes from 1 to M_1 ($M_1 < M$) belong to the first class (C_1) and instances from range $(M_1 + 1, M\rangle$ to the second class (C_2).

At first, the ARXA method focuses on gene pair matching (i, j) $(i, j \in \{1, \ldots, N\}, i \neq j)$ for which there is the highest averaged over instances probability p of an event $\frac{x_{im}}{x_{jm}} < \frac{x_{in}}{x_{jn}}$ ($m \in C_1$ and $n \in C_2$). For each pair of genes (i, j) the probability p_{ij} is calculated:

$$p_{ij} = \frac{\sum_{m=1}^{M_1} \sum_{n=M_1+1}^{M} I(\frac{x_{im}}{x_{jm}} < \frac{x_{in}}{x_{jn}})}{(|C_1| * |C_2|)}$$

where $|C_1|$ denotes the number of instances from class C_1 and $I(\frac{x_{im}}{x_{jm}} < \frac{x_{in}}{x_{jn}})$ is the indicator function defined as:

$$I(\frac{x_{im}}{x_{jm}} < \frac{x_{in}}{x_{jn}}) = \begin{cases} 1, \text{if } \frac{x_{im}}{x_{jm}} < \frac{x_{in}}{x_{jn}} \\ 0, \text{if } \frac{x_{im}}{x_{jm}} \geq \frac{x_{in}}{x_{jn}} \end{cases}.$$

This computationally expensive calculations performed in each splitting node with complexity equals $O(N^2 * M^2)$ are handled by the GPU. Next, the top ranked pair from each thread is considered in building the splitting node. The threshold are calculated on the CPU and a single test that constitute splitting node has a form e.g. $\frac{x_i}{x_j} < 5$. It denotes that the instances can be divided into two sub-groups (branches or even classes) by simply checking if expression value of gene x_j is greater than 20% of gene x_i. Alike in k-TSP [24] we define maximum number of pairs that can constitute a node (the upper bound denoted as k can be set up before the classification) but instead of minimizing the prediction error we apply the gain ratio criterion. The number of pairs that creates the node may vary due to the internal cross-validation which throws away tests that do not contribute just as it is in k-TSP. The splitting criterion is guided by a majority voting mechanism in which all pair components of the split have the same weight. In the case of a draw, the vote of the primary pair is decisive.

3.1 ARXA Scoring on GPU

We propose two-level scoring due to the performance reasons as the major part of the scoring procedure is performed on the GPU and next the top gathered results are processed by the CPU (see Fig. 2). The RXA methods like TSP and TST exhibit characteristics that make them ideal for a GPU implementation as there is no data dependence between individual scores. As it is illustrated in Fig. 2, the dataset is first copied from the CPU main memory to the GPU device memory so each thread block can access it. It is performed only once before starting the tree induction as later only the indexes of the instances that remain in a calculated node are sent. In each node, possible relations X_i/X_j need to be processed and scored. Each thread on the device is assigned an equal amount of relations (called offset) to compute (see Fig. 2). This way each thread 'knows' which relations of genes it should analyze and where it should store the result.

In addition, number of instances for which the score is calculated varies in each tree node - from the full set of samples in a root to a few instances in the lower parts of the tree. Each thread loops over the instances that reach the node and calculates the scores to the assigned relations. After all thread blocks have finished, the results are copied from the GPU device memory back to the CPU main memory where the top split is established.

3.2 ARXA Scoring on CPU

After letting GPU know which instances residue in a current node and what offset is assigned to each thread, the CPU calculates the gain ratio for the node.

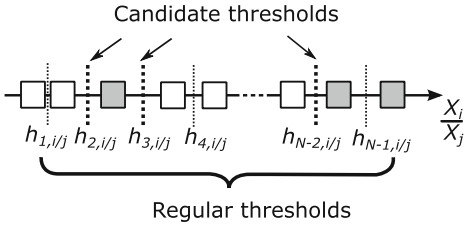

Fig. 3. Candidate thresholds for gene pair x_i/x_j

It is essential to check if potential splits returned from the GPU improves overall gain ratio as otherwise the leaf will be created. ARXA scoring on the CPU starts with sorting the results returned from threads according to their score (calculated on the GPU). Next, the results are filtered, alike in k-TSP solution, to leave only the k (default value: $k = 9$) top-ranked disjoint gene pairs. It should be noted that the GPU returns only the information about the relations and scores which is not enough to constitute a split.

Therefore, in the next step a set of tests is determined for further evaluation. Each test is constituted from a single top pair and has a form: $\frac{x_i}{x_j} < h_{i/j}$, where $h_{i/j}$ is the selected threshold. The search for the threshold only considers the relevant thresholds, called the candidate thresholds, which split instances from different classes as it is illustrated in Fig. 3. This way the algorithm does not consider e.g. $h_{1,i/j}$, $h_{4,i/j}$ and $h_{M-1,i/j}$ as those thresholds are useless for creating new tests because they split two training instances from the same class. The gain ratio criterion is used to determine the best possible threshold $h_{i/j}$, and the midpoint of the interval is applied as the value of this threshold. As an alternative to midpoint, we also performed experiments with smoothed threshold is e.g. an integer value (see enclosed results in Table 3). Finally, the choice of the number of gene pairs (parameter k) that constitute splitting node is determined by internal cross-validation.

4 Experimental Validation

In this section, we present a detailed experimental analysis to evaluate the relative performance of the proposed weight and hierarchical approach in RXA. Using several cancer-related gene expression datasets we have checked ARXA prediction power and confronted its results with popular RXA extensions.

4.1 Algorithms and Datasets

To make a proper comparison with the RXA algorithms, we use the same 8 cancer-related benchmark datasets (see Table 1) that are tested with the EvoTSP solution [6]. Datasets are deposited in NCBI's Gene Expression Omnibus and summarized in Table 1. A typical 10-fold cross-validation is applied and depending on the system, different tools are used:

Table 1. Details of gene expression datasets: abbreviation with name, number of genes and number of instances.

Datasets	Genes	Instances	Datasets	Genes	Instances
(a) GDS2771	22215	192	(e) GSE10072	22284	107
(b) GSE17920	54676	130	(f) GSE19804	54613	120
(c) GSE25837	18631	93	(g) GSE27272	24526	183
(d) GSE3365	22284	127	(h) GSE6613	22284	105

Table 2. Comparison of RXCT with top-scoring algorithms, including accuracy and the size of the classifier's model. The best accuracy for each dataset is bolded.

Dataset	TSP	TST	k-TSP		EvoTSP		TSPDT		ARXA		
	Acc.	Acc.	Acc.	Size	Acc.	Size	Acc.	Node size	Acc.	Node size	Tree size
(a)	57.2	61.9	62.9	10	65.6	4.0	60.1	15.4	**70.9**	5.7	3.6
(b)	88.7	89.4	90.1	6	96.5	2.1	**98.2**	1.0	92.5	1.0	1.0
(c)	64.9	63.7	67.2	10	78.1	2.8	72.3	5.8	**84.7**	7.6	1.4
(d)	93.5	92.8	94.1	10	**96.2**	2.1	88.3	2.0	95.0	3.0	1.0
(e)	56.0	60.5	58.4	14	66.9	3.1	68.1	4.7	**68.3**	6.7	3.4
(f)	47.3	50.1	56.2	18	66.2	2.7	67.2	10.9	**78.5**	8.1	2.2
(g)	81.9	84.2	87.2	14	86.1	4.1	88.6	3.3	**94.4**	6.7	1.0
(h)	49.5	51.7	55.8	10	53.6	6.1	59.6	7.0	**65.6**	5.9	2.4
Average	67.4	69.3	71.5	11.5	76.2	2.7	75.3	6.2	**81.2**	5.6	2.1

- evaluation of TSP, TST, and k-TSP was performed with the AUERA software [9], which is an open-source system for identification of relative expression molecular signatures;
- EvoTSP results were taken from the publication [6];
- original TSPDT and ARXA implementations are used.

Due to the performance reasons concerning other approaches, the Relief-F feature selection was applied and the number of selected genes was arbitrarily limited to the top 1000. In the experiments, we provide results for the proposed ARXA solution as well as its simplified variants which uses e.g. integer percentage split values.

Experiments were performed on a workstation equipped with Intel Core i5-8400 CPU, 32 GB RAM and NVIDIA GeForce GTX 1080 GPU card (8 GB memory, 2 560 CUDA cores). The sequential algorithm was implemented in C++ and the GPU-based parallelization part was implemented in CUDA-C (compiled by nvcc CUDA 10; single-precision arithmetic was applied).

4.2 Accuracy Comparison of ARXA to Popular RXA Counterparts

Table 2 summarizes classification performance for the proposed solution and its competitors. The model size of TSP and TST is not shown as it is fixed and

Table 3. Comparison of ARXA variants results with different model comprehensibility settings. Averaged value through all datasets are shown.

Algorithm	Accuracy	Node size	Tree size
$ARXA_{no\ round}$	**81.2**	5.6	2.1
$ARXA_{round\ 0.5}$	80.7	4.9	3.1
$ARXA_{round\ 1.0}$	79.6	4.6	3.4

equals correspondingly 2 and 3. It is clearly visible that the proposed ARXA solution managed to outperform all popular RXA classifiers in 6 out of 8 datasets. The statistical analysis of the obtained results using the Friedman test and the corresponding Dunn's multiple comparison test (significance level/p-value equals 0.05), as recommended by Demsar [9] showed that the differences in accuracy are significant. We have also performed an additional comparison between the datasets with the corrected paired t-test [24] with the significance level equals 0.05 and 9 degrees of freedom (n-1 degrees of freedom where n = 10 folds). It showed that ARXA significantly outperforms all algorithms on more than half datasets.

However, it should be noticed that improving classification accuracy was not our primary goal. We wanted to make a model in which gene pairs somehow interact with each other more deeply and also to promote finding sub-interactions between co-expressed genes and pairs. Such improvement in terms of classification accuracy was a surprise even for us, however, this may indicate the importance of the founded patterns.

4.3 ARXA Comprehensibility and GPGPU Acceleration

As we mentioned in Sect. 3.2, the fraction value which denotes the relation between two features can be rounded to improve model comprehensibility. Table 3 shows the ARXA average accuracy results from the performed experiments with different roundup of the threshold value. Therefore, in the first row we can see original ARXA version, in second row all the thresholds values in the tests that constituted splits we rounded to 0.5 and in last row the thresholds were rounded to the integer values. From the table we can observe, that, on average, as the thresholds are less precise, the number of tests in internal nodes decreases while the size of the tree increases. This outcome was consistent to all tested datasets.

In Figs. 4 and 5 we show an example decision trees induced by ARXA with and without threshold roundup. In both cases the prediction accuracy is similar but the structure and relations slightly differs. Although, there are a few similarities especially in the top nodes where e.g. two out of three relations in the root node from the DT illustrated in Fig. 5 is the same as in Fig. 4. There are also single genes that appear in both trees but constitute different pairs.

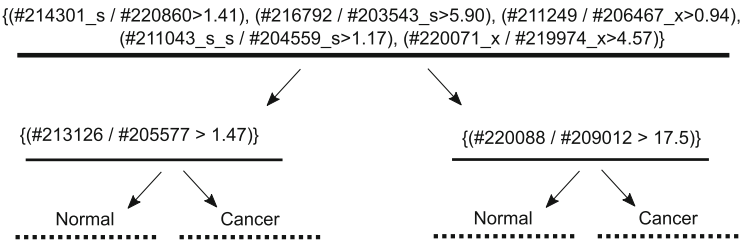

Fig. 4. An example decision tree induced by ARXA with detailed thresholds for GSE6613 Parkinson's disease.

Based on the description of the dataset (GSE6613 series) from GenBank NCBI [3] we performed a brief examination of one of the ARXA output prediction models (see Figs. 4 and 5). To check if founded genes or gene pair have some biological meaning we have decoded gene names from GSE6613 with GPL96 platform provided by NCBI. We found out that most of the founded genes are related with Parkinson's disease, for example #211249 (gene symbol: GPR68) is the top significantly deregulated gene identified through integrated analysis in Parkinson's disease [25] and gene LSM7 (#204559_s) is reported as significant in meta-analysis of genome-wide association studies of Parkinson's disease risk [4]. This is only an example of a fraction of knowledge discovered by ARXA but even the presented model is at some point supported by biological evidence in the literature.

Even with applied feature selection step (to make other algorithms work in a reasonable time), the number of possible relations for which the GPU needs to calculate is very high. For example, for N = 1000 genes and M = 100 instances the number of scores to calculate is over 10^9 in a root node (sub-nodes have fewer instances). However, if we would take the full dataset, this number drastically increases to 10^{13}. With the GPGPU acceleration the score ranking, on average through all datasets, was reduced from 20 s to 0.15 of a second which is over

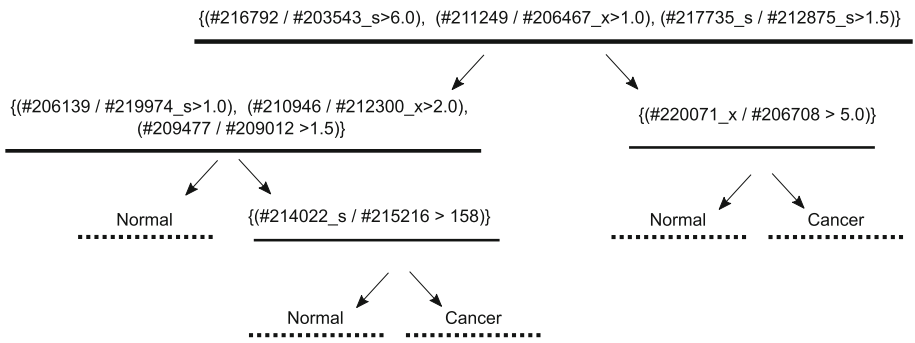

Fig. 5. An example decision tree induced by ARXA with rounded to 0.5 thresholds for GSE6613 Parkinson's disease.

two magnitude faster for a single run. The time included also the data transfer to and from the GPU. When, the whole data was used, the GPU took up to several minutes where a single sequential run of a single dataset took over a day. Through all the runs number of blocks equals to 1024 and threads equals to 128. In the profiling we noticed that processing too many possible relations by each thread (high load) slows down the parallelization. By decreasing the offset value we managed to improve load balancing and thus the overall ARXA speedup.

5 Conclusions

In this paper, we introduce a hybrid approach to analyze gene expression data which combines the problem-specific methodology with the popular white-box classifier. The Advanced Relative Expression Analysis (ARXA) fundamentally changes the RXA solution in the context of relations and pairs ranking. Our implementation considers involving GPGPU accelerated decision trees in order to open ARXA on finding interesting hierarchical patterns in subgroups of genes in a reasonable time. In addition, experiments show that knowledge discovered by ARXA is accurate, comprehensive and at some point supported by biological evidence in the literature.

We see many promising directions for future research. In particular, we are currently working with biologists and bioinformaticians to better understand the gene relations generated by ARXA. Next, there is still a lot of ways to improve the GPU parallelization of RXCT, e.g. load-balancing of tasks based on the number of instances in each node, simultaneous analysis of two branches, better GPU hierarchical memory exploitation.

It should be noted that most of RXA solutions are not fully detached decision model from the raw values of the dataset. Such an approach may reduce robustness to methodological and technical factors, study-specific biases as well as limits the potential of exploring merged data from different omics, platforms, and experiments. Most of TSP-family solutions use e.g. raw values in their secondary rankings, others mean or variance of a given gene in the data. ARXA does not consider analyzing raw values, therefore in the nearest future we want to validate our approach on multi-omics data.

Acknowledgments. This project was funded by the Polish National Science Center and allocated on the basis of decision 2019/33/B/ST6/02386 (first author). The second and third author were supported by the grant WZ/WI-IIT/3/2020 from BUT founded by Polish Ministry of Science and Higher Education.

References

1. Afsari, B., Braga-Neto, U.M., Geman, D.: Rank discriminants for predicting phenotypes from RNA expression. Ann. Appl. Stat. **8**(3), 1469–1491 (2014)

2. Bacardit, J., et al.: Hard data analytics problems make for better data analysis algorithms: bioinformatics as an example. Big Data **2**(3), 164–176 (2014)
3. Benson, D.A., et al.: GenBank. Nucleic Acids Res. **46**(D1), D41–D47 (2018)
4. Chang, D., Nalls, M.A., et al.: A meta-analysis of genome-wide association studies identifies 17 new Parkinson's disease risk loci. Nat Genet. **49**(10), 1511–1516 (2017)
5. Czajkowski, M., Kretowski, M.: Top scoring pair decision tree for gene expression data analysis. Adv. Exp. Med. Biol. **696**, 27–35 (2011)
6. Czajkowski, M., Kretowski, M.: Evolutionary approach for relative gene expression algorithms. Sci. World J. **2014**, 7 (2014). 593503
7. Czajkowski M., Kretowski M.: Relative evolutionary hierarchical analysis for gene expression data classification. In: GECCO 2019, pp. 1156–1164 (2019)
8. Czajkowski, M., Kretowski, M.: Decision tree underfitting in mining of gene expression data. An evolutionary multi-test tree approach. Expert Syst. Appl. **137**, 392–404 (2019)
9. Earls, J.C., et al.: AUREA: an open-source software system for accurate and user-friendly identification of relative expression molecular signatures. BMC Bioinform. **14**, 78 (2013). (Article 19)
10. Geman, D., et al.: Classifying gene expression profiles from pairwise mRNA comparisons. Stat. Appl. Genet. Mol. Biol. **3**(19) (2004)
11. Huang, S., Chaudhary, K., Garmire, L.X.: More is better: recent progress in multi-omics data integration methods. Front. Genet. **8**(84) (2017)
12. Huang, X., et al.: Analyzing omics data by pair-wise feature evaluation with horizontal and vertical comparisons. J. Pharm. Biomed. Anal. **157**, 20–26 (2018)
13. Hyafil, L., Rivest, R.L.: Constructing optimal binary decision trees is NP complete. Inf. Process. Lett. **5**(1), 15–17 (1976)
14. Jurczuk, K., Czajkowski, M., Kretowski, M.: Evolutionary induction of a decision tree for large scale data. A GPU-based approach. Soft Comput. **21**, 7363–7379 (2017)
15. Kagaris, D., Khamesipour A: AUCTSP: an improved biomarker gene pair class predictor. BMC Bioinform. **19**(244) (2018). (Article 244)
16. Kim, S., Lin, C.W., Tseng, G.C.: MetaKTSP: a meta-analytic top scoring pair method for robust cross-study validation of omics prediction analysis. Bioinformatics **32**(13), 1966–1973 (2016)
17. Kotsiantis, S.B.: Decision trees: a recent overview. Artif. Intell. Rev. **39**(4), 261–283 (2013)
18. Kretowski, M.: Evolutionary Decision Trees in Large-scale Data Mining. Studies in Big Data 59 (2019)
19. Lin, X., et al.: The ordering of expression among a few genes can provide simple cancer biomarkers and signal BRCA1 mutations. BMC Bioinform. **10**(256) (2009)
20. Lo, W.T., et al.: CUDT: a CUDA based decision tree algorithm. Sci. World J. **2014**, 12 (2014). 745640
21. Magis, A.T., Price, N.D.: The top-scoring 'N' algorithm: a generalized relative expression classification method from small numbers of biomolecules. BMC Bioinform. **13**(1), 227 (2012)
22. McDermott, J.E., et al.: Challenges in biomarker discovery: combining expert insights with statistical analysis of complex omics data. Expert Opin. Med. Diagn. **7**(1), 37–51 (2013)
23. Min, S., Lee, B., Yoon, S.: Deep learning in bioinformatics. Brief. Bioinform. **18**(5), 851–869 (2017)

24. Tan, A.C., Naiman, D.Q.: Simple decision rules for classifying human cancers from gene expression profiles. Bioinformatics **21**, 3896–3904 (2005)
25. Wang, J., Liu, Y., Chen, T.: Identification of key genes and pathways in Parkinson's disease through integrated analysis. Mol. Med. Rep. **16**(4), 3769–3776 (2017)