33rd Annual International Conference of the IEEE EMBS
Boston, Massachusetts USA, August 30 - September 3, 2011

SVS: Data and knowledge integration in computational biology

Grzegorz Zycinski and Annalisa Barla and Alessandro Verri

Abstract— **In this paper we present a framework for structured variable selection (SVS). The main concept of the proposed schema is to take a step towards the integration of two different aspects of data mining: database and machine learning perspective. The framework is flexible enough to use not only microarray data, but other high-throughput data of choice (e.g. from mass spectrometry, microarray, next generation sequencing). Moreover, the feature selection phase incorporates prior biological knowledge in a modular way from various repositories and is ready to host different statistical learning techniques. We present a proof of concept of SVS, illustrating some implementation details and describing current results on high-throughput microarray data.**

I. Introduction

Modern biology is facing the major issue of integrating well-established statistical analysis approaches with the available knowledge stored in public databases and literature, in order to obtain interpretable results that may be applied in actual biological practice.

In the last decade, the central topic of biology has been to understand the complicated molecular interactions underlying the nature of diseases. As all biomedical sciences, molecular biology has undergone an actual revolution, driven by the introduction of new genome-wide technologies and of information processing techniques into daily research.

One key challenge comes from extracting knowledge from enormous datasets generated by high-throughput technologies. In this context, one popular example is microarray technology, which enables simultaneous measurement of thousands of gene expressions in the biological samples representing the process of interest. A common problem is the identification of predictive gene signatures from a small number of biological samples compared to the number of measured genes (tens vs. tens of thousands). Many statistical approaches were proposed to tackle this problem, such as *filtering, wrapper* and *embedded* methods [1], the most popular examples being GSEA [2], SVM-RFE [3], the lasso [4]. In order to avoid selection bias [5], any adopted statistical technique requires the validation of its results on independent datasets. When a new dataset is not available, cross-validation or boot-strapping procedures are needed, increasing the computational requirements. In practical implementations, such heavy computational demands usually force the adoption of advanced computational paradigms, as Grid, High-Performance or Cloud computing.

G. Zycinski, A. Barla and A. Verri are with DISI, Department of Information and Computer Science, University of Genova, I-16146 via Dodecaneso 35, Genova, Italy (e-mail: {grzegorz.zycinski, annalisa.barla, alessandro.verri}@unige.it).

On the other hand, a vast amount of structured biological knowledge has been digitized and stored in various repositories and databases, Gene Ontology (GO) [6], KEGG [7] and *Ensembl* [8], being the most used. As these data banks increase their size, another great challenge arising in the field is to be able to query them in a sensible way [9]. Indeed, various data representations and formats are employed to represent and store the information, making it very difficult to obtain effective interoperability.

II. Methods

A. Motivation and general ideas

Structured Variable Selection (SVS) aims at filling a gap between the paradigms of data integration traditionally employed in bioinformatics and machine learning-based approaches of data and knowledge integration. The general idea is to combine heterogenous high-throughput data with structured biological knowledge already available in the field. A complete review of the current state-of-the-art on the topic of knowledge and data integration is beyond the scope of our work, for a complete review see for instance [10] and references therein.

In principle, the integration between data and prior knowledge could be performed by direct injection of prior knowledge into the learning from data phase. The current implementation of this concept is one particular instance of the schema represented in Fig. 1 and it can be summarized with the two following steps: first, raw data produced by high-throughput (HT) technologies is integrated with prior biological knowledge available from GO; next, machine learning techniques are applied to such dynamically formed information ensemble, in order to discover meaningful biological knowledge applicable in clinical usage.

The integration of raw data with prior knowledge is performed as follows. Raw data from HT platform is collected, assembled and normalized using state-of-the-art normalization algorithms [11]. Then, vendor-specific platform-related annotations for raw data are collected and integrated as well. Finally, prior biological knowledge is retrieved from GO and included into the integration process performed by the *local integration framework*, see Fig. 1. The result of this phase is a dynamically created *information ensemble*. The idea behind it is based on the fact that nearly all pieces of information processed, namely raw data, bioinformatic annotations and prior knowledge records, may be associated to specific identifiers (also referred to as *keys*). The keys enable joining the content of many data streams in a fashion similar to what JOIN statements allow in SQL language. For example when using microarrays, an *expression measurement*

978-1-4244-4122-8/11/$26.00 ©2011 IEEE

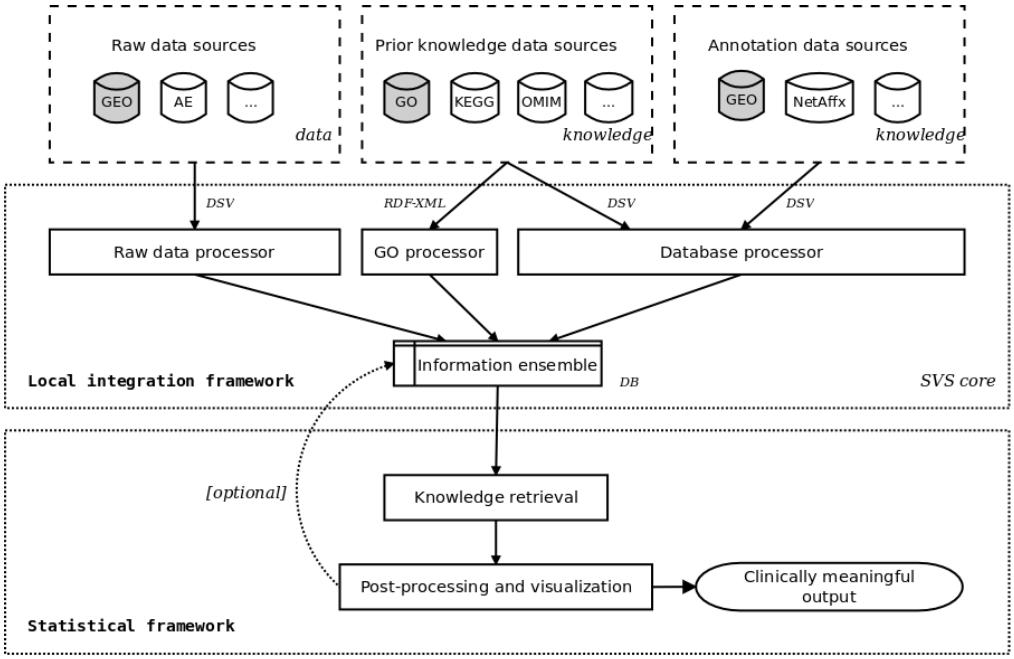

Fig. 1. General schema of Structure Variable Selection prototype system. Highlighted structures were used in the current implementation.

is associated with a *probeset name* that can be mapped to its corresponding *gene symbol*, which in turn enables to retrieve associated *GO terms* or *KEGG pathways*. All these relations may be used, in the following phase, by the *statistical framework*.

Based on the ensemble created previously, the statistical framework applies machine learning techniques to identify meaningful molecular entities (*knowledge retrieval* phase in Fig. 1) and then post-processes the results in order to prepare them for clinical usage.

B. SVS prototype implementation

Usually, machine learning approaches deal with bulk numerical datasets, requiring considerable computing power; to minimize this, the current SVS prototype follows the idea of guided partitioning of bulk datasets into smaller ones, and executing smaller computational tasks over remote computational resources. The prototype was implemented in Python [12] and R [13] programming languages. It was used to analyze microarray data in combination with prior knowledge coming from GO in order to provide a functional characterization of the biological phenomenon of interest. It is composed of two modules: Local Integration Framework and Statistical Framework.

As depicted in Fig. 1, the Local Integration Framework consists of three *processors*. We describe each of them separately, providing some implementation details; then we illustrate the main characteristics of their output, the *information ensemble*.

1) **Raw data processor:** Raw data processor uses normalization algorithms implemented in R language packages [14], [15]. Input data from Gene Expression Omnibus (GEO) [16] is provided as series of raw measurements performed for each biological sample and the output is a gene expression data matrix, stored in a Delimited Separated Values (DSV) file.

2) **GO processor:** GO is a database of controlled vocabularies (ontologies) structured as a directed acyclic graph (DAG) where each term has a defined relationship to one or more terms. The GO processor explicitly uses the DAG structure encoded in a RDF-XML file, available either as *termdb* or *assocdb* builds, from GO Consortium website [17]. The *termdb* build contains GO terms hierarchy, terms definitions and mappings to other databases. The *assocdb* build includes *termdb* and also contains manual and most of the electronic annotations compiled for terms. For the sake of simplicity our Python implementation of GO processor parses the *termdb* RDF-XML build, being much smaller and informative enough. As output, it produces a *tree structure* of GO terms, trading some information loss (namely, relation types and multiple parentage) for simplicity of usage. The output is in the form of serialized in-memory structure. This information is used when some information about GO terms is not available in vendor-specific microarray annotations, such as term hierarchy; it also allows for resolving synonymic terms and avoiding processing obsolete ones.

3) **Database processor:** The Database processor fuses information from previous processors, as needed, to build a dynamic information ensemble in the form of database tables. First, all DSV files are loaded into corresponding *raw tables*: the parser follows the structure of DSV file header to properly create the table and copies data into it. Next, *local data integration* is performed and derived tables, containing combined pieces of information from raw tables, are created. In the prototype Python implementation, we employed SQLite, its default SQL engine. Being an embedded

engine with small memory footprint, SQLite implements a subset of standard SQL dialect, sufficient to carry basic data manipulation needed by SVS. To minimize maintenance efforts and to provide elasticity, SQLite stores databases in single files. One database file can hold many tables and other structures (views, triggers, etc.) associated with them. Currently, the raw tables contain: expression measurements from gene expression data matrix (*GEDM*), vendor-specific annotations (*ANNO*), gene naming information (*HGNC*), and phenotype information for supervised analysis in the knowledge retrieval phase (*LABELS*). Note that the latter table is not required, for example in unsupervised learning problems. Each table row is identified by its key, unique within the table, namely *probeset name* for *GEDM* and *ANNO* tables, *gene symbol* for *HGNC*, and biological sample name for *LABELS*. Currently, SVS creates just two derived tables. The *term2probeset* derived table contains all the basic information associated with probeset names regarding GO terms such as ID, description, available evidence codes etc. Here, we discard control probesets such as AFFX probesets in Affymetrix platforms. Moreover, only GO terms that have any probeset associated according to *ANNO* data, are considered. This table was constructed to obtain the mapping GO term→probesets, more useful than the one available in *ANNO* table, namely probeset→GO terms.

The *probeset2gene* derived table contains associations between probeset names and their respective gene products, or sequence references. Here, if the original information from vendor-specific annotation *ANNO* data is confirmed in *HGNC* data, gene name is resolved unambiguosly; otherwise, the original GenBank accession number, provided by vendor, is used to pinpoint any external annotation for the probeset name in question. This table is constructed to control sequence references coming from *ANNO*, e.g. to filter out outdated or non-official gene symbols, known pseudo-genes, to resolve obsolete GenBank sequence records etc.

Current derived tables contain convenient mappings used later in knowledge retrieval phase, based solely on raw tables. No additional information is added there. New derived tables may be created in the future, if necessary. Also, useful information from knowledge retrieval phase may be included back here.

4) **Information ensemble:** Information ensemble encloses all information produced by relevant processors, in the form of SQLite database and serialized data structures. After *local data integration* has been performed, new kind of data is produced to utilize *data and knowledge integration* in *knowledge retrieval* phase. Using GO terms and vendor-specific annotations as a guide, for every GO term, only the expression measurements across all biological samples, related to this term, are considered and extracted in the form of submatrices. Each submatrix corresponds to one GO term. The set of such submatrices is the input to the statistical framework. Submatrices are available in the ensemble as serialized data structures. They are final output of the Local Integration Framework.

The Statistical Framework (Fig. 1) is implemented as standalone Python script. Its main goal is to address the biological question of interest. For example, in a supervised learning scenario, the question might be to identify the most relevant molecular variables and GO terms to discriminate healthy and diseased samples.

The set of submatrices from information ensemble is fed to the knowledge retrieval procedure, which, by means of machine learning methods, elaborates the data, selecting relevant variables and estimating the goodness of fit (e.g. classification error in supervised analysis) [18]. The prototype implementation takes as input the set of submatrices from Local Integration Framework and applies the feature selection procedure to each of them in distributed environment of local cluster of machines, built using Parallel Python (PP) [19] and PPlus (in-house, unpublished) Python applications. PPlus is a tiny wrapper over PP that hides all low-level activities of PP, and exposes all data needed for computation in shared central file storage. The Statistical Framework uses PPlus API to access individual computational resources (CPU cores) and to manipulate files; all intermediate data are made available in information ensemble.

In this phase, one can implement any feature selection technique of choice. We adopted $l_1 l_2$ with double optimization [20], a regularization method that enforces the sparsity over the objective function and, at the same time, retains the correlated discriminant variables. To avoid selection bias, the method is cast into a nested double cross-validation loop [21], [22]. Each submatrix is processed individually in parallel and associated to a classification error and to a list of selected probeset names. This information is available in the information ensemble as well.

The post-processing phase is implemented as standalone Python and R scripts that work further on information ensemble. Since the information obtained from knowledge retrieval is numerical, it must be re-annotated (with gene references, GO term details, etc.) to provide more straightforward and more biologically verifiable answers, avoiding as much as possible the manual post-processing of numerical results. Various statistics can be collected in this phase, such as mean and standard deviation of prediction error, histograms of selected variables etc., all of them automatically re-annotated. Specifically, in order to highlight the statistically meaningful results, the GO terms associated to the submatrices with a low prediction error are visualized in a non-interactive subgraph that can support the biologists to interpret the reliability of statistical results and to integrate it within existing domain knowledge. In the next future, automatic discovering of new meaningful entities, such as clusters of GO nodes, and subsequent updates of information ensemble with them, will be added as a feature of SVS.

III. Results

SVS was tested on a dataset of Affymetrix HG U133A microarray data on Parkinson's disease (PD) available in GEO (GSE20295) [23]. The dataset is composed of 53 controls and 40 PD samples from patients with late stage PD, resulting in a normalized 93×22215 GEDM. The

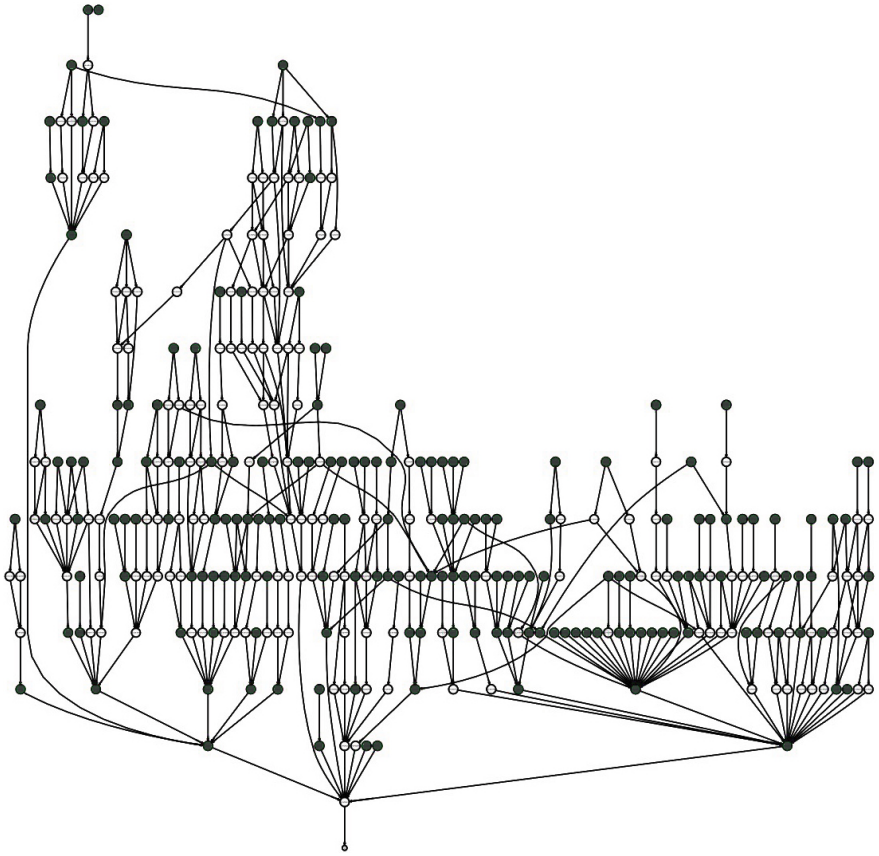

Fig. 2. Subgraph plot showing the selected GO terms on PD dataset.

main goal of this supervised experiment was to define a molecular characterization of PD. As proof of concept, we present the results obtained on one of the three domains of GO, that is the Molecular Function (MF) domain. The first phase, i.e. local integration framework, integrated the prior knowledge stored in MF obtaining 2574 submatrices. Each of them was associated to the binary classification problem of discriminating PD from healthy status. As described in Sec. II, in the knowledge retrieval phase the $l_1 l_2$ algorithm estimates a cross-validation error and a list of relevant variables for each submatrix. We considered as predictive only those matrices that scored a clasification accuracy higher than 70%, obtaining a final list of 188 GO terms, partially reported in Table I. The validity of the results goes beyond statistical robustness. Indeed, the selected GO terms are biologically sound and related with the disease. They belong to several GO classes: general (intracellular, cytoplasm, negative regulation of biological process), nervous system (neurotransmitter transport, transmission of nerve impulse, learning or memory), response to stimuli (behavior, temperature, organic substances, drugs or endogenous stimuli). The subgraph representing the relevant GO terms is shown in Fig. 2. Such visual representation is an efficient way to provide an easy-to-read biological information that highlights the similarity and the biological soundness of the extracted gene signatures.

TABLE I
TEN OF THE MOST RELEVANT GO TERMS SELECTED BY SVS

GO term	Description
GO:0020037	heme binding
GO:0003774	motor activity
GO:0005262	calcium channel activity
GO:0003777	microtubule motor activity
GO:0004896	cytokine receptor activity
GO:0051059	NF-kappaB binding
GO:0032395	MHC class II receptor activity
GO:0005388	calcium-transporting ATPase activity
GO:0015085	calcium ion transmembrane transporter activity
GO:0005539	glycosaminoglycan binding

IV. CONCLUSIONS

The SVS framework attempts to fill the gap between data-driven and knowledge-driven approaches for analysis of molecular data in computational biology. The general schema is designed to make use of different data sources as well as various publicly available repositories of biological domain knowledge.

In this work, we described in detail the prototype of SVS and presented experimental results on microarray data of Parkinson's disease.

We are aware of some limitations affecting the current implementation and we will work to overcome them in the near future. First, we aim at including other domain

knowledge sources, such as the KEGG repository. For example, one can assign GO terms to well established genes in KEGG pathways, and utilize this both as additional static annotation or in a different schema of guided partitioning of bulk numerical datasets. Moreover, we plan to annotate in this way all sub-reactions in the pathways of interest.

Also, we will work on the integration of more than one data type as well as various and possibly custom annotation data sources. For example, taking into account two high–throughput techniques exploring biological activity of DNA sequences – namely comparative genomic hybridization (CGH) and gene expression microarrays – we may relate the measurement of amplification of a particular DNA sequence, with the expression of mRNA derived from this sequence, and utilize this combined information as input for feature selection techniques.

Lastly, we will devise new methods to inject prior knowledge into the statistical learning phase. Currently, the available prior knowledge is used as a filter to restrict the search for meaningful features on a subset of the original group of measured features. Usually this *discarding* process significantly reduces the initial number of variables (tens of thousands) to some hundreds, diminishing the *curse of dimensionality* effect and promoting robustness in the statistical analysis. Taking advantage of the flexibility of machine learning methods, we aim at using prior knowledge in a more complex fashion. For instance, the structured prior knowledge may be used to design custom kernel functions or it could be used to define new penalty terms in the regularization functional.

ACKNOWLEDGMENT

The authors would like to thank Salvatore Masecchia, Margherita Squillario and Curzio Basso for useful comments and suggestions.

REFERENCES

[1] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *The Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[2] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, and J. P. Mesirov, "Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles," *PNAS*, vol. 102, no. 43, pp. 15 545–50, Oct 2005.

[3] I. Guyon, G. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine Learning*, Jan 2002.

[4] R. Tibshirani, "Regression Shrinkage and Selection via the Lasso," *Journal of the Royal Statistical Society, Series B*, vol. 56, pp. 267–288, 1996.

[5] C. Ambroise and G. McLachlan, "Selection bias in gene extraction on the basis of microarray gene-expression data," *PNAS*, vol. 99, no. 10, p. 6562, 2002.

[6] M. Ashburner, C. A. Ball, J. Blake, D. Botstein, H. Butler, J. M. Cherry, A. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock, "Gene ontology: tool for the unification of biology. The Gene Ontology Consortium." *Nature Genetics*, vol. 25, no. 1, pp. 25–9, May 2000.

[7] M. Kanehisa and S. Goto, "KEGG: Kyoto Encyclopedia of Genes and Genomes," *Nucleic Acids Res*, vol. 28, no. 1, pp. 27–30, Jan 2000.

[8] E. Birney, T. D. Andrews, P. Bevan, M. Caccamo, Y. Chen, L. Clarke, G. Coates, J. Cuff, V. Curwen, T. Cutts, T. Down, E. Eyras, X. M. Fernandez-Suarez, P. Gane, B. Gibbins, J. Gilbert, M. Hammond, H.-R. Hotz, V. Iyer, K. Jekosch, A. Kahari, A. Kasprzyk, D. Keefe, S. Keenan, H. Lehvaslaiho, G. McVicker, C. Melsopp, P. Meidl, E. Mongin, R. Pettett, S. Potter, G. Proctor, M. Rae, S. Searle, G. Slater, D. Smedley, J. Smith, W. Spooner, A. Stabenau, J. Stalker, R. Storey, A. Ureta-Vidal, K. C. Woodwark, G. Cameron, R. Durbin, A. Cox, T. Hubbard, and M. Clamp, "An overview of Ensembl," *Genome Res*, vol. 14, no. 5, pp. 925–8, May 2004.

[9] X. M. Fernández-Suárez and E. Birney, "Advanced genomic data mining," *PLoS Comput Biol*, vol. 4, no. 9, p. e1000121, Jan 2008.

[10] S. Philippi, "Data and knowledge integration in the life sciences," *Briefings in Bioinformatics*, Jan 2008.

[11] R. Irizarry, Z. Wu, and H. Jaffee, "Comparison of Affymetrix GeneChip expression measures," *Bioinformatics*, Jan 2006.

[12] [Online]. Available: http://www.python.org/

[13] [Online]. Available: http://www.r-project.org/

[14] R. C. Gentleman, V. J. Carey, D. M. Bates, *et al.*, "Bioconductor: Open software development for computational biology and bioinformatics," *Genome Biology*, vol. 5, p. R80, 2004.

[15] H. Bengtsson, P. Wirapati, and T. P. Speed, "A single-array preprocessing method for estimating full-resolution raw copy numbers from all Affymetrix genotyping arrays including GenomeWideSNP 5 & 6," *Bioinformatics*, vol. 25, no. 17, pp. 2149–2156, 2009.

[16] R. Edgar, M. Domrachev, and A. E. Lash, "Gene Expression Omnibus: NCBI gene expression and hybridization array data repository," *Nucleic Acids Res*, vol. 30, no. 1, pp. 207–10, Jan 2002.

[17] [Online]. Available: http://www.geneontology.org/GO.downloads.shtml

[18] T. Hastie, R. Tibshirani, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer-Verlag, 2009.

[19] [Online]. Available: http://www.parallelpython.com/

[20] C. De Mol, S. Mosci, M. Traskine, and A. Verri, "A Regularized Method for Selecting Nested Groups of Relevant Genes from Microarray Data," *Journal of Computational Biology*, vol. 16, pp. 1–15, Apr 2009.

[21] C. Ambroise and G. McLachlan, "Selection bias in gene extraction on the basis of microarray gene-expression data," *Proc. Natl. Acad. Sci. USA.*, vol. 99, no. 10, pp. 6562–6566, 2002.

[22] A. Barla, S. Mosci, L. Rosasco, and A. Verri, "A method for robust variable selection with significance assessment," *Proceedings of ESANN 2008*, Jan 2008.

[23] Y. Zhang, M. James, F. A. Middleton, and R. L. Davis, "Transcriptional analysis of multiple brain regions in Parkinson's disease supports the involvement of specific protein processing, energy metabolism, and signaling pathways, and suggests novel disease mechanisms," *Am J Med Genet B Neuropsychiatr Genet*, vol. 137B, no. 1, pp. 5–16, Aug 2005.