

Scopira: A Pattern Recognition Application Framework for Biomedical Datasets

Rodrigo A. Vivanco, Aleksander Demko, Nick J. Pizzi
Institute for Biodiagnostics, National Research Council Canada
Rodrigo.Vivanco@nrc-cnrc.gc.ca

Abstract

Machine learning techniques are widely used in the analysis of biomedical datasets. Modern devices tend to produce voluminous, high-dimensional datasets for which medical practitioners require high-performance, user-friendly programs and researchers need effective algorithm development and testing platforms. Interactive development systems, such as MATLAB, provide for rapid prototyping of algorithms and visualization but at the cost of computational efficiency. We present Scopira, a C++, open source programming framework for the development of biomedical data analysis applications.

1. Introduction

Pattern recognition techniques are widely used in the biomedical domain, solving problems ranging from cancer research [1,2] to the detection of neural activations in the human brain [3]. Most of the basic algorithms are either programmed into proprietary applications or part of an interpreted system such as MATLAB [4], the statistical scripting environment R [5], or Mathematica[®] [6].

These interpreted programming environments, with a rich set of mathematical operations and visualization tools, are well suited for prototyping and testing pattern recognition algorithms. However, their lack of modern software engineering constructs, the difficulty of fully and seamlessly integrating with computationally efficient programming languages such as C/C++, and their interpreted nature, does not make these systems ideal platforms for the development of complex, high-performance software applications suitable for the medical practitioner, especially with the emergence of voluminous high-dimensional datasets.

Scopira has been designed as a comprehensive, C++ object-oriented programming framework for the development of applications geared towards the data analysis and visualization of biomedical datasets.

2. Scopira

The emphasis with Scopira has been on high-performance, open source development and the ability to easily integrate other C/C++ libraries used in the biomedical imaging and analysis field by providing a common application programming interface (API) with a suite of built-in subsystems that ease the burden of complex application development.

As well, since some machine learning algorithms, such as evolutionary computation with genetic algorithms, lend themselves well to parallelization, and with the advent of Beowulf clusters [7], Scopira has been augmented to take full advantage of the message passing interface (MPI) library [8] for parallel computing.

2.1 N-Dimensional data arrays

Scopira offers developers an efficient and flexible, n-dimensional data array structure, *narray*, applicable to any numerical data type of arbitrary dimension (1D-vectors, 2D-matrices, 3D-cubes, etc), which is tuned for a particular data type and dimension. *narray* also provides run-time bounds checking to facilitate robust application development and debugging, but can be removed on release builds resulting in performance equivalent to native C implementations.

The *nslice* template class is a virtual n-dimensional array that is simply a *reference* to an *narray*. The class only contains dimension specification information and element access translates directly to the host *narray*. An *nslice* can have any dimensionality less than or equal to the host. This flexibility is very powerful; one could have a one-dimensional data-vector slice from a matrix, cube or five-dimensional array without copying data.

The *narray* class also provides hooks for alternate memory allocation systems such as DirectIO. Using the memory mapping facilities of the operating system, a disk file may be mapped into memory. When this memory space is accessed, the pages of the files are loaded into memory transparently to the user of *narray*. Furthermore,

as the *narray* class is 64-bit clean, on 64-bit architectures very large files may be used as datasets and the operating system will page portions of the file into memory as needed. Figure 1 illustrates a visualization module in

Scopira for a large (25 GB) optical coherence tomography image of a tooth, used to monitor dental health and search for micro cavities.

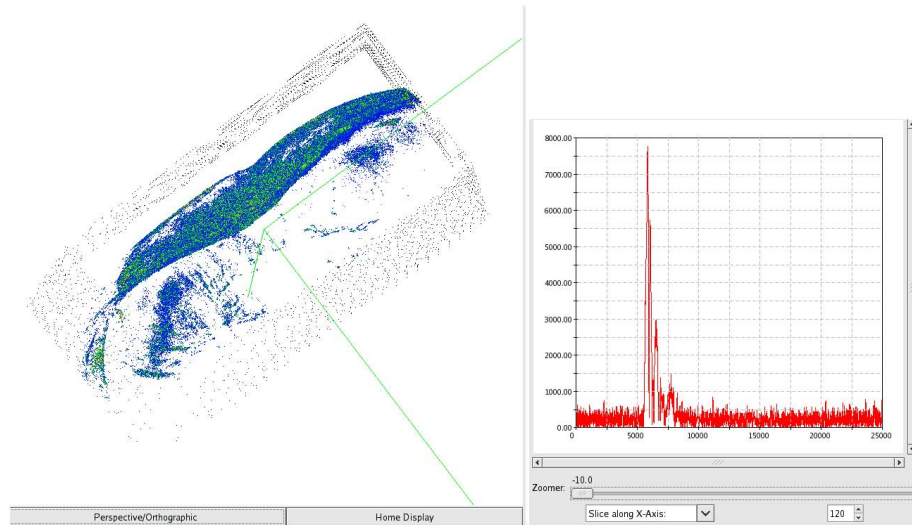


Figure 1: OCT scan of a tooth (25 GB dataset) and spectrum plot for a specified voxel.

2.2 Tools subsystem

Scopira consists of modular subsystems that can be used as needed by developers. The tools subsystem provides facilities useful in many programming domains, not just numerical and scientific computing. A reference counting scheme provides the basis for memory management and avoids many of the C++ memory leak problems common with regular pointers. Other useful utilities such as random number generation, XML file processing and dynamic library loading are provided.

Scopira also provides a flexible input/output system. Data flow objects may be linked dynamically to form I/O flow streams. Scopira includes flow objects to terminate or initiate a data flow for standard files, network sockets and memory buffers that are aware of the basic Scopira data structures. Transform I/O objects simply translate data from one form to another, such as binary-to-hex, buffer consolidating and ASCII encoding. Future transformers could include CRC calculators, compressors and cryptographic ciphers.

Serialization I/O objects provide a mechanism for objects to encode their data into a persistent stream. Through this interface, large complex data objects can quickly and easily encode themselves to disk (useful to save an application's data state in order for it to be restored the next it is launched) or over a network to be reconstructed at a remote location (useful for process migration on a cluster for load balancing issues).

2.3 Graphical user interface

This subsystem provides a basic graphical API wrapped around GTK+ [9] and consists of widget and window classes that become the foundation for all graphical user interface (GUI) objects in Scopira. More specialized and complex widgets, particular useful to numerical computing and visualization are also provided. This includes components useful for the display of matrices (eg, confusion matrices from classifiers), 2D images (eg, magnetic resonance (MR) images) and line plots (eg, MR spectra). Developers can use the basic GUI components provided to create more complex viewers for a particular application domain. A complementary subsystem provides the base OpenGL-enabled widget class that utilizes the GTKGLExt library [10]. The GTKGLExt library enables GTK+ based applications to utilize OpenGL for 2D and 3D visualization.

2.4 Type registration subsystem

This subsystem provides the basic mechanism through which applications and external plug-ins register their data models and visualization GUI components. Third party developers can load their plug-in extensions into a Scopira application to add new functionality to registered data models. The open source C++ medical image processing library ITK [11], which provides many basic algorithms such as segmentation with fuzzy region growing and multi-model image registration, has been successfully integrated into Scopira.

2.5 MPI extension subsystem

This subsystem provides a set of *narray* aware functions and input/output objects that allow developers to easily interface with the MPI programmer's API. Using C++ trait classes for type information and the size data already stored in *narray*, the Scopira wrapper functions drastically reduce the amount of parameters needed from the programmer thereby reducing common mistakes when using MPI.

2.6 Pattern recognition algorithms

Various pattern recognition algorithms have been implemented in Scopira and are available to developers. Linear discriminate analysis (LDA) [12] has been used extensively with Scopira applications as a supervised classifier for cancer [13] and other biomedical datasets [14].

It is not uncommon for biomedical spectra to be characterized by high dimensionality (many features) and sparse sample size resulting in what is referred to the 'curse of dimensionality' [15]. Several preprocessing algorithms for feature selection are implemented in Scopira to handle datasets with these characteristics.

A multilayer perceptron, along with a stochastic feature selection algorithm, were implemented using Scopira's MPI parallel programming extensions and applied to the classification of yeast infections with MR spectra datasets [16]. A genetic algorithm is also available

with Scopira for classification feature subset selection of biomedical datasets [17].

A fast implementation of Bezdek's fuzzy clustering algorithm (FCA) [18] is available for general unsupervised classification problems. As well, a modified FCA and pre-processing algorithms have been developed specifically for the analysis of time-series data such as functional MR imaging (fMRI) [19].

EROICA uses frequency analysis of the time-series fMRI dataset as a pre-selection criterion to remove time courses (TC) that are not likely to be involved in (experiment-relevant) neural activation. The reduced dataset is then clustered with FCA, which converges to a solution much faster than when using all the available TC. The removed TC are then validated against the final cluster centroids to ensure that a rejected TC should be included in an activation cluster.

Other unsupervised classification methods available to Scopira users includes a hierarchical clustering algorithm (Ward's method). The hierarchical clustering method was used to classify brain tumors using MR spectra of human brain neoplasm [20].

The pattern recognition algorithms available in Scopira have been used mostly in biomedical datasets but have also been applied to software engineering problems. A parallel genetic algorithm was implemented to drastically reduce the execution time for feature selection in classification problems. The parallel GA was used to select the subset of software quality metrics, obtained from a Java/C++ application, which produced the best prediction in terms of software quality [21].

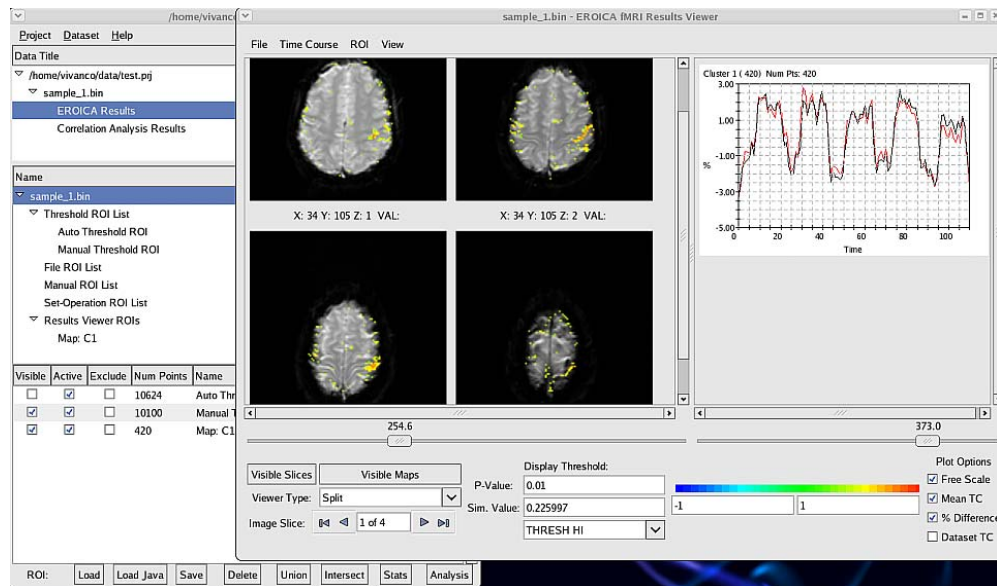


Figure 2: Neural activation maps of a finger tapping functional MRI experiment using fuzzy clustering.

3. Applications

The main focus during the evolution of Scopira has been as a high-performance application development framework for pattern recognition problems in the biomedical field. Scopira offers a programming template for applications that require a graphical user interface.

Developers can use this template to quickly provide users an interactive application. The entire source code for the Scopira software development framework is available for downloading at scopira.org. Currently, two applications and their source code, developed within the Scopira framework, are available from our institute for free downloading.

3.1 EvIdent[®]

EvIdent[®] is a data analysis application for fMRI experiments [22]. Initially coded with Java and C++, it has been ported to Scopira in order to improve its computational performance. FMRI datasets were in the order of 5–10 MB but with improved scanner technology, multi-slice high temporal resolution fMRI datasets are now commonly 50–100 MB.

EvIdent[®] provides model-based correlation analysis to find activation voxels that match a user specified time-

course. This time course is usually composed of the experimental stimulus and is convolved with mathematical models of the expected haemodynamic characteristics of the subject to the fMRI experiment. Model free cluster analysis is also available in EvIdent[®] with EROICA.

EvIdent[®] provides an intuitive and effective user interface and is illustrated in Figure 2, which shows the main application window and the neural activation results window. All of the image handling modules, dataset loading, image display, and region of interest management, are part of Scopira and available to programmers.

3.2 Visualization of high-dimensional patterns

Biomedical [MR, infrared (IR) or Raman] spectra of biofluids, mass spectra, e.g. from proteomics and microarrays expression profiles, are being acquired with increasingly higher dimensionality. Scopira provides a distance (similarity)-based method of visualizing these high-dimensional patterns and their relative relationships [23].

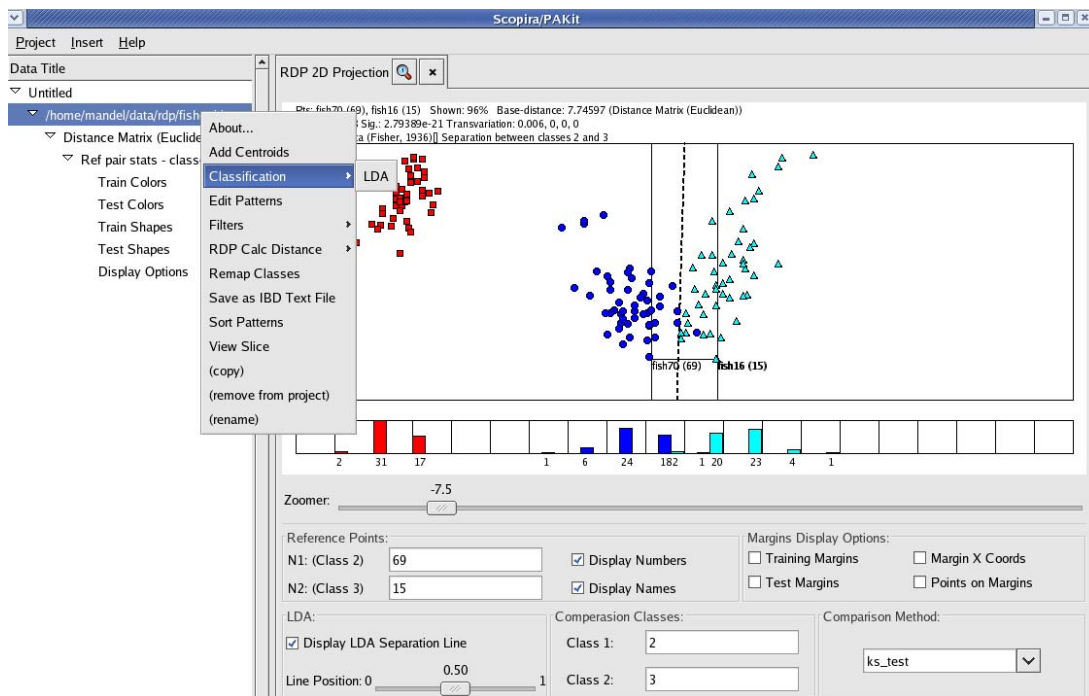


Figure 3: Scopira’s relative distance plane viewer module for a 3-class problem.

The original distances between points, with respect to any two reference patterns, are preserved in a special two-dimensional coordinate system, the relative distance plane (RDP), as illustrated in Figure 3. This visualization of the patterns permits a rapid assessment of class pattern

distributions and may be used as an initial data mining exploration before classification by some classifiers.

4. Discussion

Scopira has proven to be a successful programming framework for developing high-performance biomedical data analysis applications. It is based on C++, an efficient object-oriented language, and the source code is available as an open source project for other researchers to use and adapt in their own research endeavours. It has been compiled to work on Linux and Windows XP operating systems.

Scopira eases application development efforts by providing many useful subsystems, flexible and efficient data models, low level tools such as memory management and serialization, graphical user interface constructs, high-level visualization modules, and the ability to implement parallel algorithms with MPI.

Scopira is continuously under development and future capabilities will include the ability to develop distributed programs using agents, applicable to grid-computing data mining applications. Interpreted environments such as MATLAB are very useful as rapid prototyping platforms to algorithm developers and Scopira has been augmented with the ability to be used from within a MATLAB session. This gives MATLAB users the ability to use Scopira developed modules.

5. Acknowledgments

We would like to thank Brion Dolenko, Marina Mandelzweig and Mark Alexiuk of NRC for their contributions to the development of Scopira. Natural Sciences and Engineering Research Council of Canada (NSERC) is gratefully acknowledged for their support to this work.

6. References

- [1] T.R. Golub, D.K. Slonim, P. Tamayo, et al, "Molecular classification of cancer, class discovery and class prediction by gene expression monitoring", *Science*, 286, 1999, pp. 531–537.
- [2] J. Khan, J.S. Wei, M. Ringner, et al, "Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks", *Nat Med*, 7(6), 2001, pp. 1–10.
- [3] A. Wismüller, A. Meyer-Bäse, O. Lange, D. Auer, M. F. Reiser, D. Sumners, "Model-free functional MRI analysis based on unsupervised clustering", *J. Biomedical Informatics*, 37, 2004, pp.10–18.
- [4] K. Sigmon, T. Davis, *MATLAB Primer, 6th Edition*, CRC Press, Cleveland, 2001.
- [5] W.N. Venables, D.M. Smith, and the R Development Core Team, *An Introduction to R*, Network Theory Ltd, Bristol, 2004.
- [6] S. Wolfram, *The Mathematica Book, 5th Edition*, Wolfram Media, Champaign, 2003.
- [7] W. Gropp, E. Lusk, T. Sterling, *Beowulf Cluster Computing with Linux, Second Edition*, MIT Press, Cambridge, 2003.
- [8] M. Snir, S. Otto, S. Huss-Lederman, D. Walker, and J. Dongarra, *MPI: The Complete Reference*, MIT Press, Cambridge, 1998.
- [9] S. Logan, *GTK+ Programming in C*, Prentice Hall, New Jersey, USA, 2001.
- [10] GTK+ OpenGL Extension: <http://gtkglxext.sourceforge.net/>
- [11] L. Ibanez, W. Schroeder, L. Ng, J. Cates, *The ITK Software Guide*, Kitware Inc, New York, 2003.
- [12] A.C. Rencher, *Methods of Multivariate Analysis*, Wiley, New York, 1995.
- [13] C.L. Lean, R.L. Somorjai, I.C.P. Smith, P. Russel, C. E. Mountford, "Accurate diagnosis and prognosis of human cancers by proton MRS and a three stage classification strategy", *Annual Report NMR Spectroscopy*, 48, 2002, pp.71–111.
- [14] R.L. Somorjai, B. Dolenko, A. Nikulin, P. Nickerson, D. Rush, A. Shaw, M. Glogowski, J. Rendell, R. Deslauriers, "Distinguishing normal from rejecting renal allografts: application of a three-stage classification strategy to MR and IR spectra of urine", *Vib. Spectrosc*, 28, 2002, pp. 97–102, 2002.
- [15] R.L. Somorjai, B. Dolenko, R. Baumgartner, "Class prediction and discovery using gene microarray and proteomics mass spectroscopy data: curses, caveats, cautions", *Bioinformatics*, 19(2), 2003, pp.1484–91.
- [16] N.J. Pizzi, "Classification of biomedical spectra using stochastic feature selection", *Neural Network World*, 15(3), 2005, pp. 257-268.
- [17] A.E. Nikulin, B. Dolenko, T. Bezabeh, R.L. Somorjai, "Near-optimal region selection for feature space reduction: novel preprocessing methods for classifying MR spectra", *NMR in Biomedicine*, 11, 1998, pp. 209–216.
- [18] J. Kolen, T. Hutcheson, "Reducing the Time Complexity of the Fuzzy-C-Means Algorithm", *IEEE Trans. Fuzzy Systems*, 10, 2002, pp. 263–267.
- [19] M. Jarmasz, R.L. Somorjai, "EROICA: exploring regions of interest with cluster analysis in large functional magnetic resonance imaging data sets", *Concepts Magnetic Resonance Part A*, 16A, 2003, pp. 50–62.
- [20] H. Yang, N.J. Pizzi, "Biomedical data classification using hierarchical clustering", *Proc IEEE Canadian Conf Elect Comput Eng*, Niagara Falls, Canada, May, 2004, pp.1861–1864.
- [21] R.A. Vivanco, N.J. Pizzi, "Finding optimal software metrics to classify software maintainability using a parallel genetic algorithm", *Proceedings of the Genetic and Evolutionary Computation Conference*, Seattle, USA, June, 2004, pp. 1388–1399.

[22] N.J. Pizzi, R.A. Vivanco, R.L. Somorjai, “EvIdent: a functional magnetic resonance image analysis system”, *Artificial Intelligence in Medicine*, 21, 2001, pp. 263–269.

[23] R.L. Somorjai, B. Dolenko, A. Demko, M. Mandelzweig, A.E. Nikulin, R. Baumgartner, N.J. Pizzi, “Mapping high-dimensional data onto a relative distance plane – an exact method for visualizing and characterizing high-dimensional patterns”, *J. Biomedical Informatics*, 37, 2004, pp.366–379