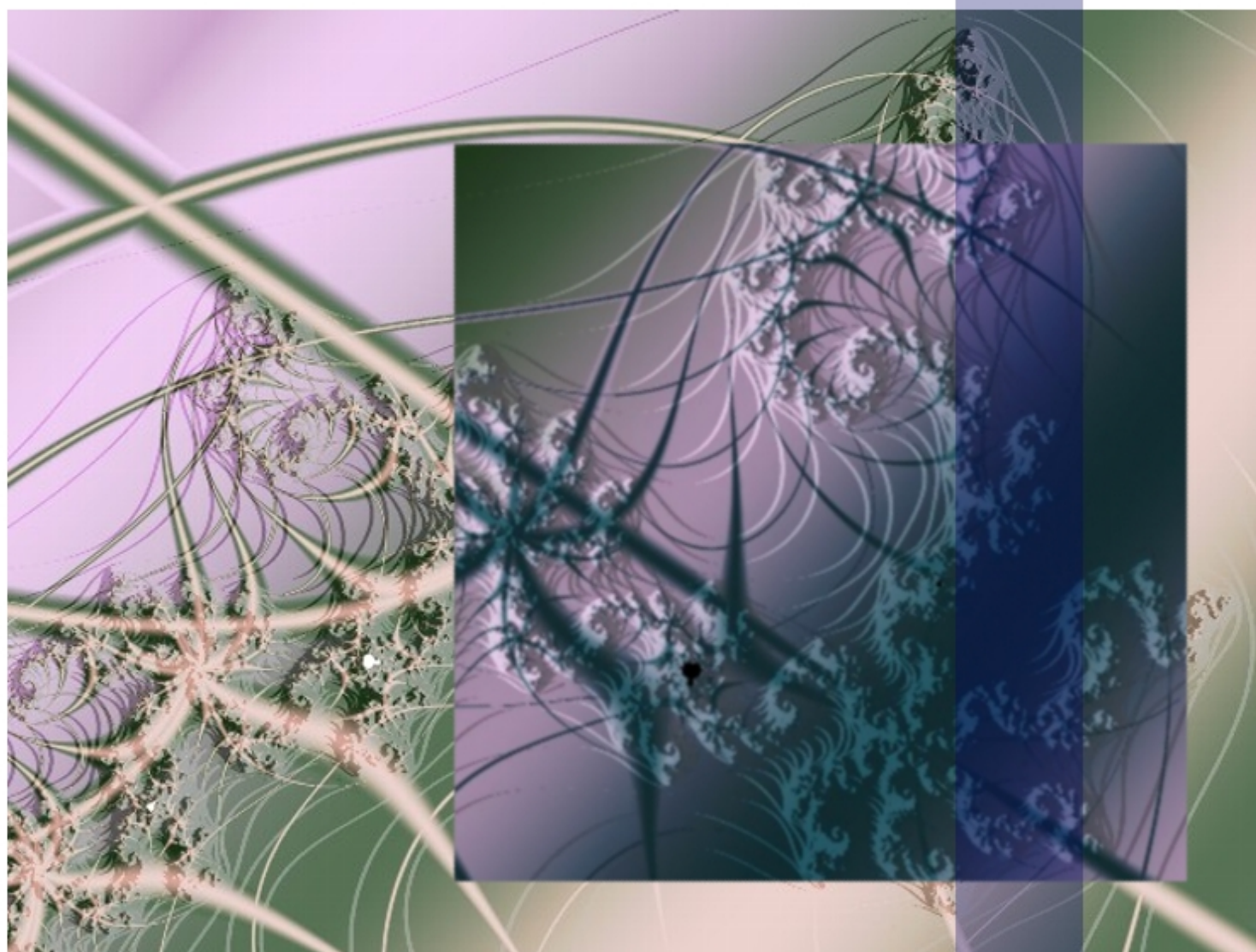




VO-Neural

Support Vector Machines



User Manual v. 1.1


VONeural Project

SVM User Manual

Doc. : SVM_user_manual_VONEURAL-MAN-NA-0003-Rel1.1

Issue: 1.1

Date: 15/07/2008

What	Who	When	Signatures
<p>Prepared by</p>	<p>VO-Neural Team</p>	<p>14/02/2008</p>	

1 Project Members

- Massimo Brescia^{2,6} (P.M.)
- Stefano Cavuoti^{2,5}
- Raffaele d'Abrusco^{2,5}
- Giovanni d'Angelo^{2,5}
- Elisabetta de Filippis⁵
- Natalia Deniskina^{2,5} (GRID manager)
- Omar Laurino⁵ (System manager)
- Giuseppe Longo^{5,3} (P.I.)
- Roberto Tagliaferri⁸

External collaborators

- Anna Corazza⁶
- Antonino Staiano⁷
- Ciro Donalek¹

Students involved

- Diego Capozzi⁴
- Mauro Garofalo⁵
- Alfonso Nocella⁵

1. California Institute of Technology – Department of Astronomy
2. I.N.A.F. – Astronomical Observatory of Capodimonte
3. I.N.F.N. Naples
4. John Moore University, Liverpool
5. University Federico II, Naples – Department of Physical Sciences
6. University Federico II, Naples – Department of Informatics
7. University Parthenope, Naples
8. University of Salerno

INDEX

1 Project Members.....3
 2 Abbreviations & Acronyms.....5
 3 Introduction.....6
 4 The Support Vector Machines.....6
 5 General introduction on the use of VONeural_SVM.....7
 5.1 Step by step procedure.....8
 6 Copyright.....13
 7 Bibliography.....14

TABLE INDEX

Tab. 1: Abbreviations and acronyms.....5
 Tab. 2: Summary of parameters involved in each of the operating modes.....7

FIGURE INDEX

Fig. 1: Virtual Observatory Login.....8
 Fig. 2: AstroGrid Workbench9
 Fig. 3: Task Launcher.....9
 Fig. 4: Selection MLP Methods Interface.....10
 Fig. 5: Parameters Interface.....10
 Fig. 6: Summary Information Window.....11
 Fig. 7: Parameters Selection.....12
 Fig. 8: Process Status Interface.....12
 Fig. 9: Results Window.....13

2 Abbreviations & Acronyms

A & A	Meaning
SVM	Support Vector Machines
NaN	Not a Number
MLP	Multi Layer Perceptron
PPS	Principal Probabilistic Surfaces
NN	Neural Networks
VO	Virtual Observatory

Tab. 1: Abbreviations and acronyms

3 Introduction

VO-Neural is a package of soft computing (machine learning) tools specifically tailored to work on massive astronomical data sets within a distributed computing environment. Its main structure is derived from the Astroneural Matlab toolbox extensively discussed in Tagliaferri et al. 2003, [4].

The reasons to adopt the ASTRONEURAL code instead of other NN packages were mainly:

- i) the fact that this software has been extensively applied to many fields (astrophysics, seismology, environmental monitoring and bioinformatics),
- ii) the fact that it includes all the options which are needed for scientific applications.

The porting of ASTRONEURAL under VO has required the re-engineering and porting under C++ of the kernels while for visualization purposes we preferred to use the existing TOPCAT facility.

In its final version the software will include three main tools:

- **VONeural_MLP**: MultiLayer Perceptron
- **VONeural_SVM**: Support Vector Machines
- **VONeural_PPS**: Probabilistic Principal Surfaces

We want to stress that the execution of VONeural_SVM, discuss in this manual, is realised in GRID using VO-Neural GRID Launcher. The VO-Neural GRID Launcher is an interface between VO and Grid infrastructures. It allows an VO user to start a computational tasks in Grid as from the user interface of VO (for example Workbench or VODesktop of AstroGrid) as from Desktop of user. The user must chose SVM application in the programs list of VODesktop (or Workbench). It will be launched to GRID, executed in GRID, and will retrieve results to VO Storage automatically. The user of VO can start a computational task in GRID without a personal GRID certificate because INFN GRID "robot certificate" is incorporated in VO-Neural GRID Launcher that offers to an general user the resources of GRID S.Co.P.E. (S.Co.P.E. is GRID project of the University Federico II, Naples).

The following sections will describe the use of the VONeural_SVM released with version VONeural 1.0.

4 The Support Vector Machines

SVM may be used to solve regression and classification problems.

Given a training set of instance-label pairs $(x_i, y_i); i = 1, \dots, l$ where $x_i \in \mathbb{R}^n$ and $y_i \in \{1, -1\}$, the support vector machines (SVM) (Boser, Guyon, and Vapnik 1992; Cortes and Vapnik 1995) require the solution of the following optimization problem:

$$\begin{aligned} \min_{w, b, \xi} \quad & \frac{1}{2} w^T w + C \sum_{i=1}^l \xi_i \\ \text{subject to} \quad & y_i (w^T \phi(x_i) + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0. \end{aligned}$$

Here training vectors x_i are mapped into a higher (maybe infinite) dimensional space by the ϕ function ϕ . Then SVM finds a linear separating hyperplane with the maximal margin in this higher dimensional space. $C > 0$ is the penalty parameter of the error term. Furthermore, $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$ is called the kernel function. You may find in VONeural_SVM the following four basic kernels:

- linear: $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$.
- polynomial: $K(\mathbf{x}_i, \mathbf{x}_j) = (\gamma \mathbf{x}_i^T \mathbf{x}_j + r)^d, \gamma > 0$.
- radial basis function (RBF): $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2), \gamma > 0$.
- sigmoid: $K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\gamma \mathbf{x}_i^T \mathbf{x}_j + r)$.

Here, γ , r , and d are kernel parameters.

Details about SVM theory and about different models implemented in our SVM may be found on the paper written by authors of LIBSVM: Chih-Chung Chang and Chih-Jen Lin: LIBSVM: a library for Support Vector Machines that may be downloaded at <http://www.csie.ntu.edu.tw/~cjlin/papers/libsvm.pdf>

5 General introduction on the use of VONeural_SVM

First of all make sure that your data set (DATASET) are stored in Astrogrid Myspace. For a given experiment you may need up to two auxiliary types of data sets: (TRAINING and TEST)

TRAINING: containing a subset of DATASET for which the target information is available. On this data set the SVM will learn how to recognize a pattern.

TEST: may contain a subset of the whole data set for which the target information is available. This data set may be used for evaluating the performances of the network.

VO-Neural_SVM offers to you three methods (TRAIN, TEST, FULL) to work on your datasets; the chose strictly depends on what kind of work you need:

- **TRAIN:** to train a new neural net and to validate it if you select a validation dataset. The output is a zip file that contains mainly a file with the created SVM (example.model) and log files .
- **TEST:** session is the way to apply an existing net to a new dataset without re-training it, may be used to evaluate the performances of a net (either produced by the TRAIN method or by other tools) by comparing its prediction with the known targets. The outputs are: a file with *output vectors* (example.test) and two log files, if you want you may have an other file (example.test.target) that contains target and output vectors.
- **FULL:** It is the union of the TRAIN and TEST sessions. The outputs are mainly a file containing the *net* (example.model) of TRAIN session; a file with *target* and *output vectors* (example.test) of the TEST session and log files, if you want you may have an other file (example.test.target) that contains target and output vectors.

Each output files is compressed in a zip file and deployed on the Astrogrid Myspace.

In the following schema are listed all the input parameters of the methods.

TRAIN	TEST	FULL
file format SVM type (optional) kernel type (optional) degree (optional) gamma (optional) coef0 (optional) cost (optional) nu (optional) epsilon (optional) tolerance (optional) shrinking (optional) probability (optional) weight (optional) nfold (optional) trainset	file format probability (optional) model targetvector testset output file	file format SVM type (optional) kernel type (optional) degree (optional) gamma (optional) coef0 (optional) cost (optional) nu (optional) epsilon (optional) tolerance (optional) shrinking (optional) probability (optional) weight (optional) targetvector trainset

Tab. 2: Summary of parameters involved in each of the operating modes

Below all input parameters are explained:

Input parameters

- **svm type** : set type of SVM (default 0)
 - 0 -- C-SVC
 - 1 -- nu-SVC
 - 2 -- one-class SVM
 - 3 -- epsilon-SVR
 - 4 -- nu-SVR
- **kernel type** : set type of kernel function (default 2)
 - 0 -- linear: $u^T v$
 - 1 -- polynomial: $(\gamma u^T v + \text{coef0})^{\text{degree}}$
 - 2 -- radial basis function: $\exp(-\gamma |u-v|^2)$
 - 3 -- sigmoid: $\tanh(\gamma u^T v + \text{coef0})$
 - 4 -- precomputed kernel (kernel values in training_set_file)
- **degree** : set degree in kernel function (default 3)
- **gamma** : set gamma in kernel function (default $1/k$)
- **coef0** : set coef0 in kernel function (default 0)
- **cost** : set the parameter C of C-SVC, epsilon-SVR, and nu-SVR (default 1)
- **nu** : set the parameter nu of nu-SVC, one-class SVM, and nu-SVR (default 0.5)
- **epsilon** : set the epsilon in loss function of epsilon-SVR (default 0.1)
- **tolerance** : set tolerance of termination criterion (default 0.001)
- **shrinking**: whether to use the shrinking heuristics, 0 or 1 (default 1)
- **probability** : whether to train a SVC or SVR model for probability estimates, 0 or 1 (default 0)
- **weight**: set the parameter C of class i to $\text{weight} \cdot C$, for C-SVC (default 1)
- **n-fold** : n of folder in cross validation mode
- **target vector** : type yes if you want target vector in the output file, no if you don't want it;
- **model**: input neural net file;
- **trainset**: training dataset file;
- **testset**: Test dataset file;

- **output file:** output file name

5.1 Step by step procedure

The detailed procedure, step by step is the following (to be completed during or after test):

Preliminary operations:

1. split the dataset in two sections: train and test; in the example it's used the 80% of the original dataset for train and 20% for the test. These sections are randomly extracted to exclude any strange effect caused by sorting of original dataset;
2. launch the AstroGrid Workbench application;
3. sign in to your MySpace account selecting "Data Analysis" panel and clicking on "Myspace". A simple interface like this will pop up:



Fig. 1: Virtual Observatory Login

Now the user is requested to provide the Community (choosing from the list), his User name and Password; click on OK to confirm. User is then signed in and have access to his "Myspace".

4. create three empty file inside "Myspace" and upload to this file the table containing parameters and target for train set, validation set and test set. If you want to use a file stored on your local drive, skip this step.

Running the experiment:

1. select the "Task Launcher" window from the same "Data Analysis" panel or from "Data Discovery" panel:



Fig. 2: AstroGrid Workbench

2. type 'VONeural' in the search field of the Chooser tab of "Task Launcher" window and click on "Search" in order to retrieve all resources connected to the "VONeural" package:

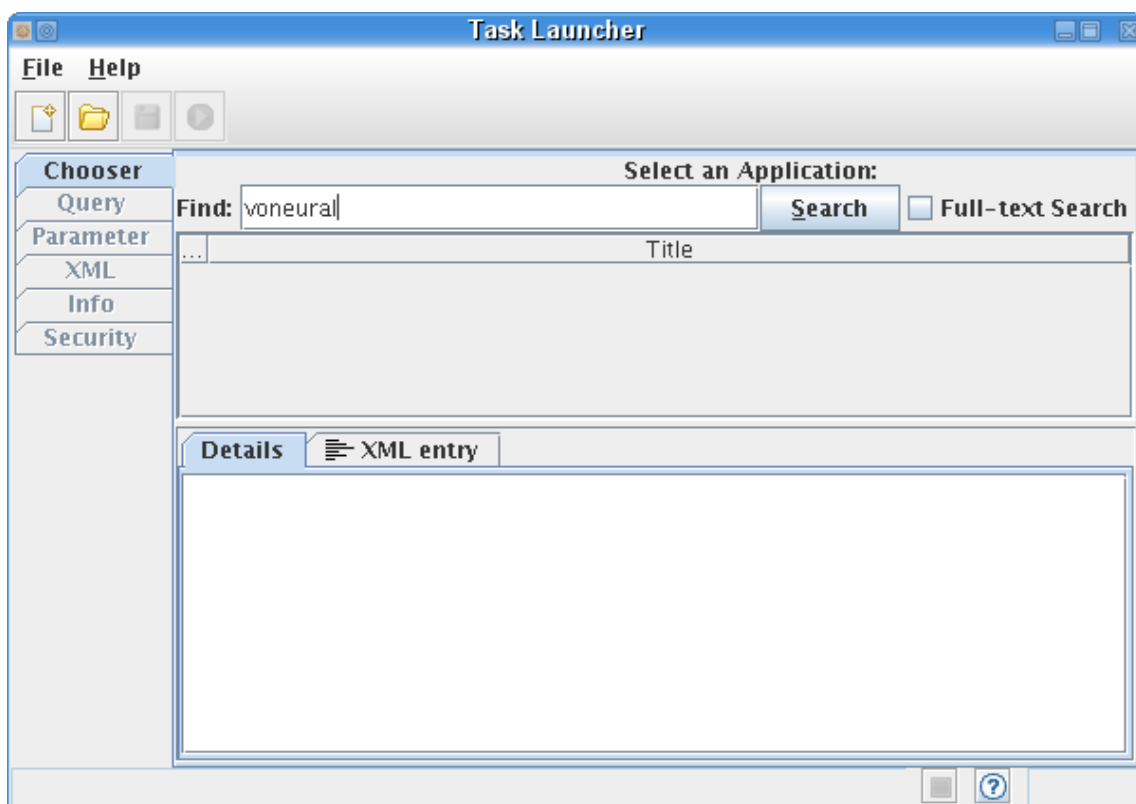


Fig. 3: Task Launcher

When all the possible choices are listed in the window, select “VONeural SVM”.

3. A new window appears, with different options for the SVM interface to be used. In the case of this simple experiment, select “Full”:

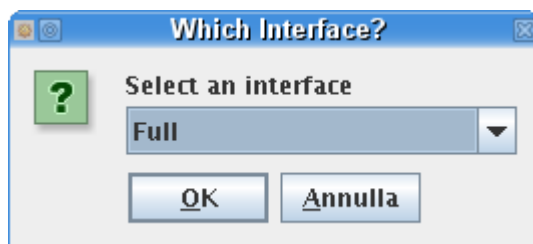


Fig. 4: Selection MLP Methods Interface

4. You come back to the “Parameter” tab of “Task Launcher” window, where the values of the parameters of the MLP neural networks are to be set, together with the name of the input and output file. Notice that other tabs are active (XML, Info, Security and Chooser).

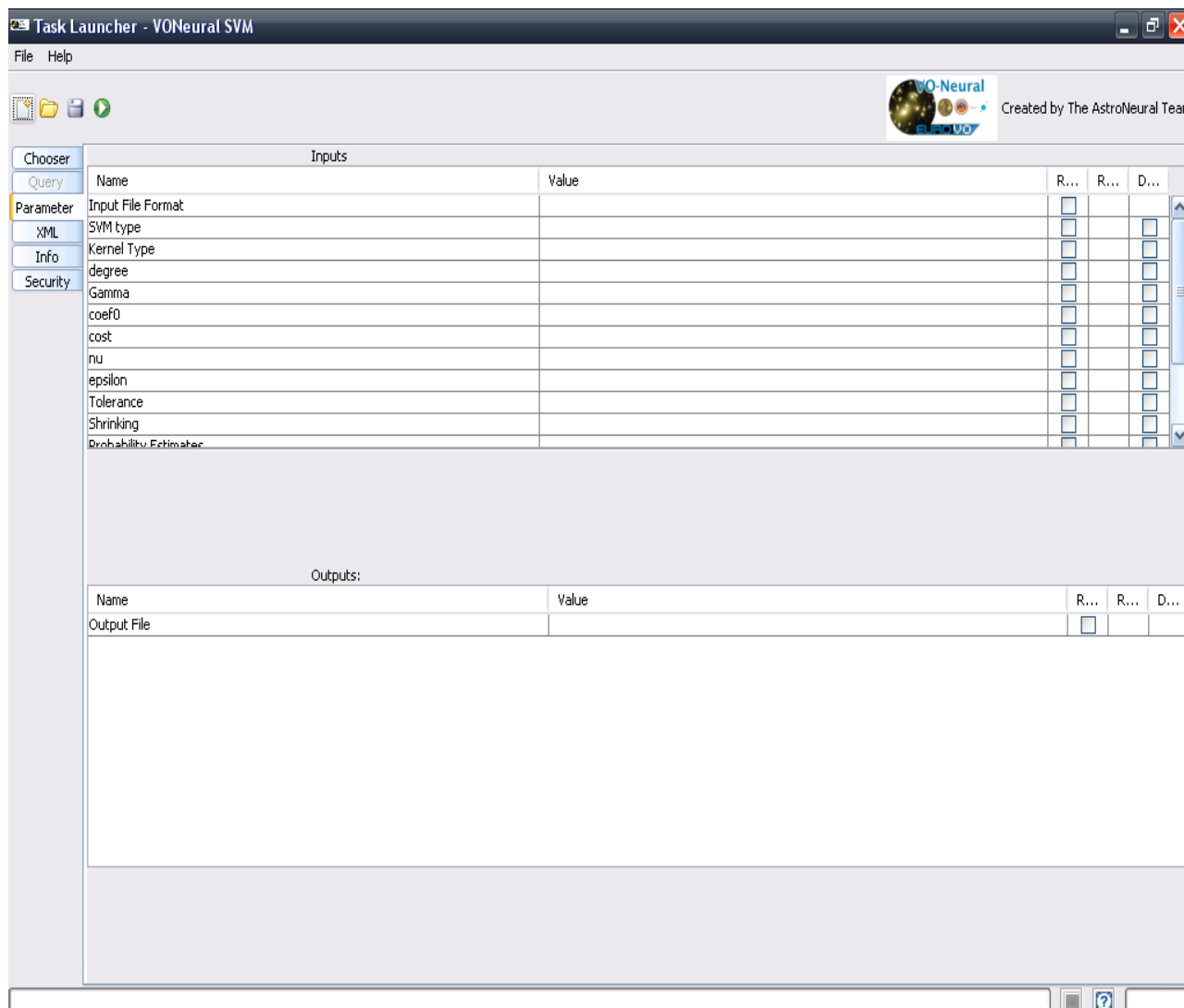


Fig. 5: Parameters Interface

A web-page containing technical documentation about VONeural SVM implementation and all the other algorithms included in the VONeural package can be reached by clicking on the VONeural logo at the top of the window, if you don't need to use a parameter you may delete it by the checkbox on the right side. A summary of the informations of this web page can be found in the Chooser tab:

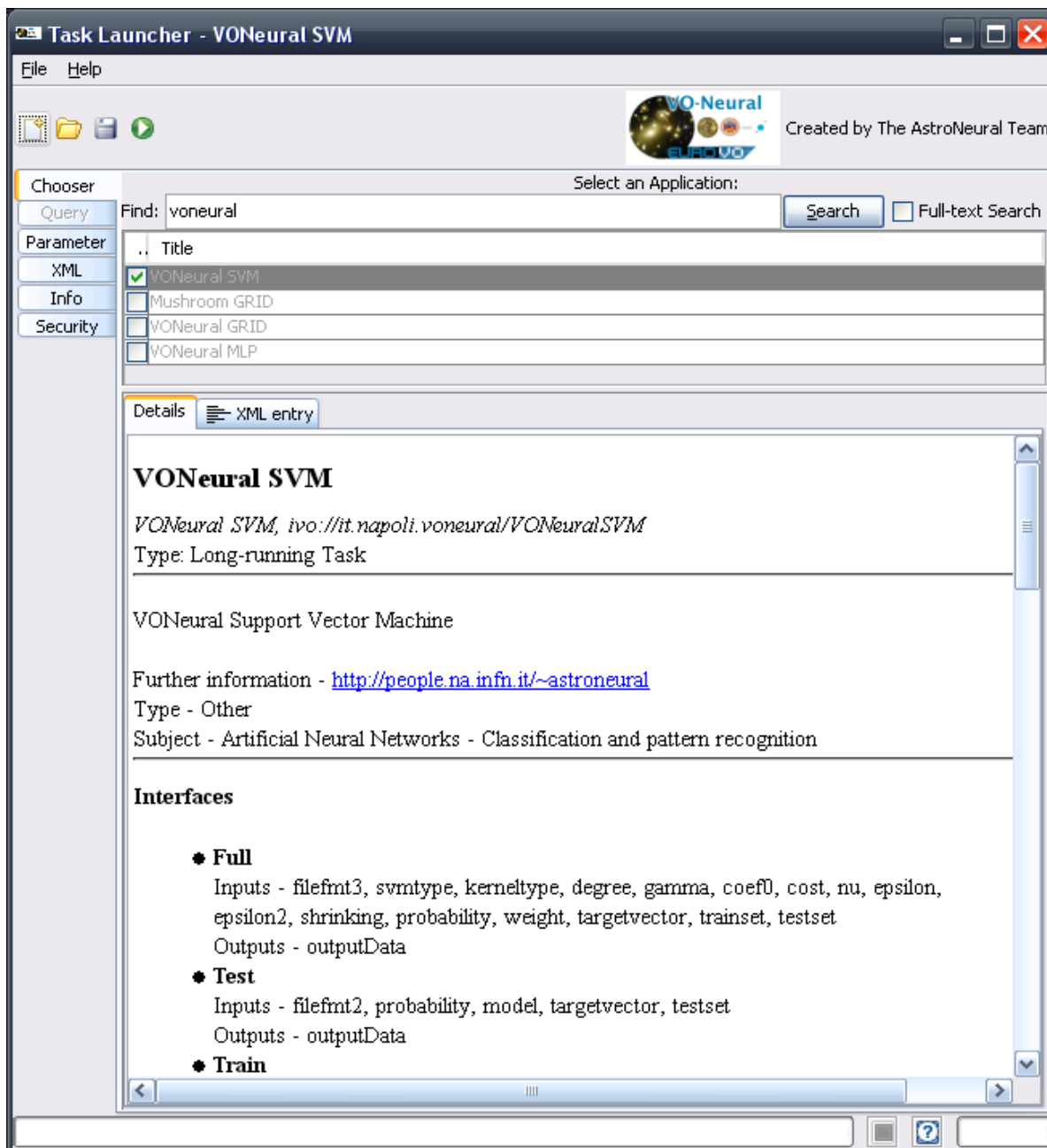


Fig. 6: Summary Information Window

5. Set all parameters to the reference settings values reported in the “Data and settings” section of this document by clicking on the 'value' field and typing the corresponding value to each numerical parameter. By selecting the boxes corresponding to the input and output file, a window pops up to browse the local disk or “Myspace”, depending on where the file containing the training is located. Select your train set, test set and output file name (in case it does not exist, remember to create an empty file with the same name inside “Myspace” before assigning the output file name.)

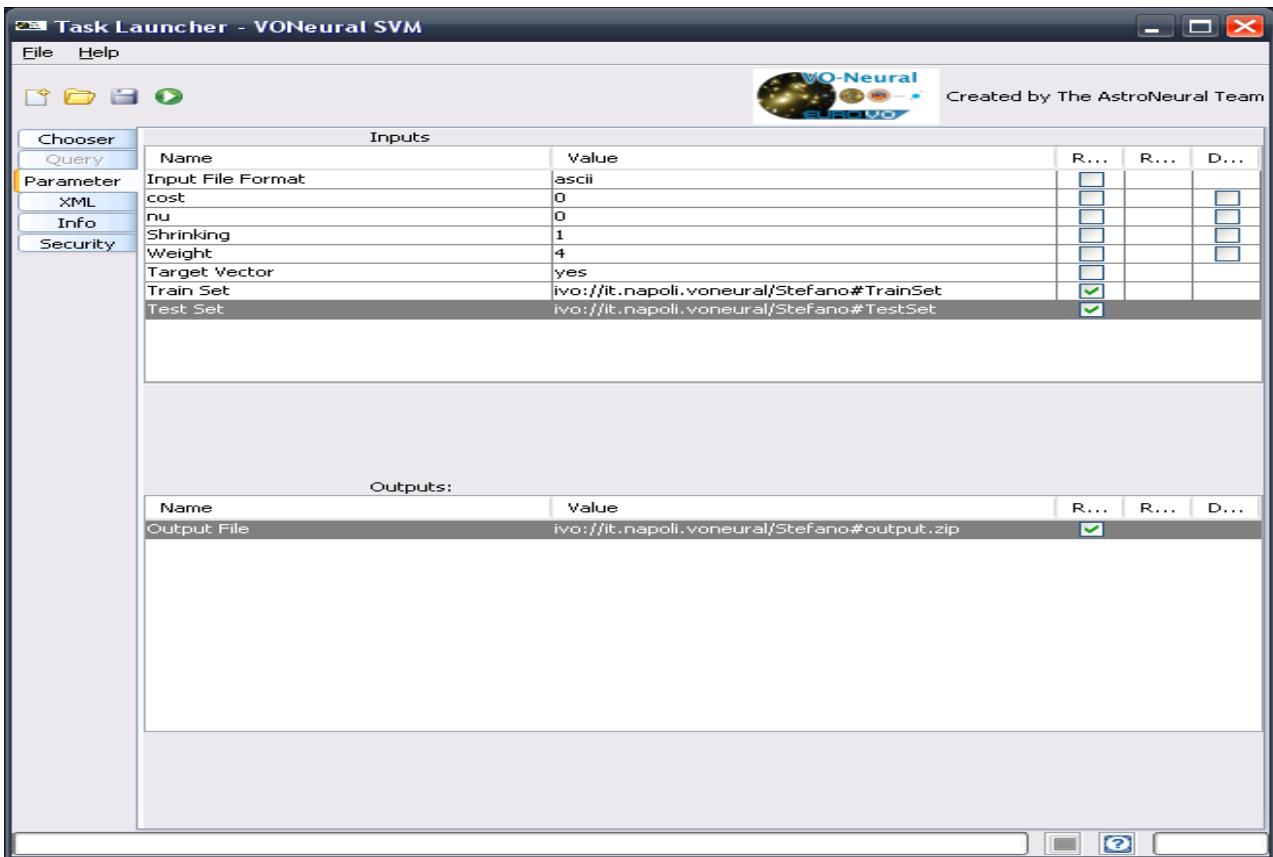


Fig. 7: Parameters Selection

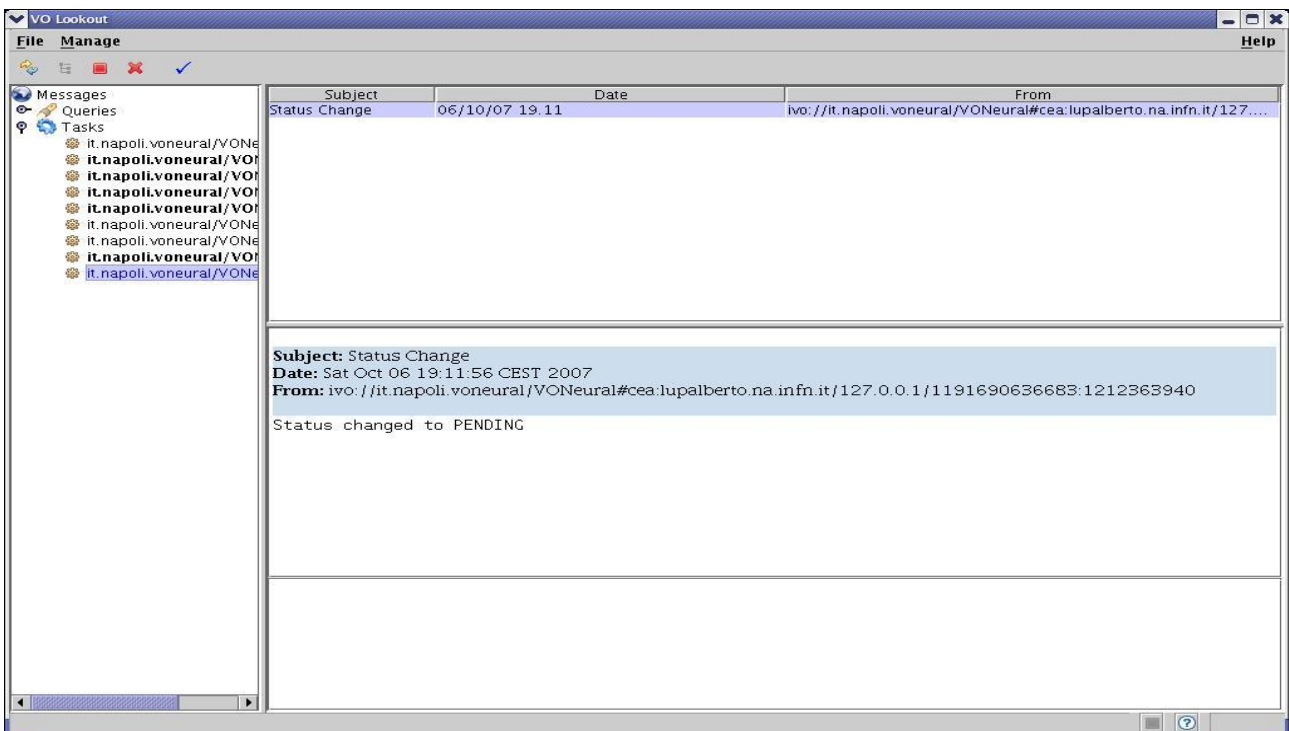


Fig. 8: Process Status Interface

6. Run the experiment by clicking on the green and white symbol located in the left upper side of the window.
7. You can check the state of the process by looking at the “VO Lookout” window that can be found in the “System Services” panel of Astrogrid Workbench. The process is listed under 'Tasks'.
8. When the process launched turns green (i.e. it has been successfully completed), select the “sample_dataset_photo_output” file located in the “Myspace”. This is a zipped collection of output files. Select the output file from “Myspace” by accessing the filetree through “Myspace” window in “System Services” panel in Astrogrid Workbench. Download it to the local disk by right-clicking on it.

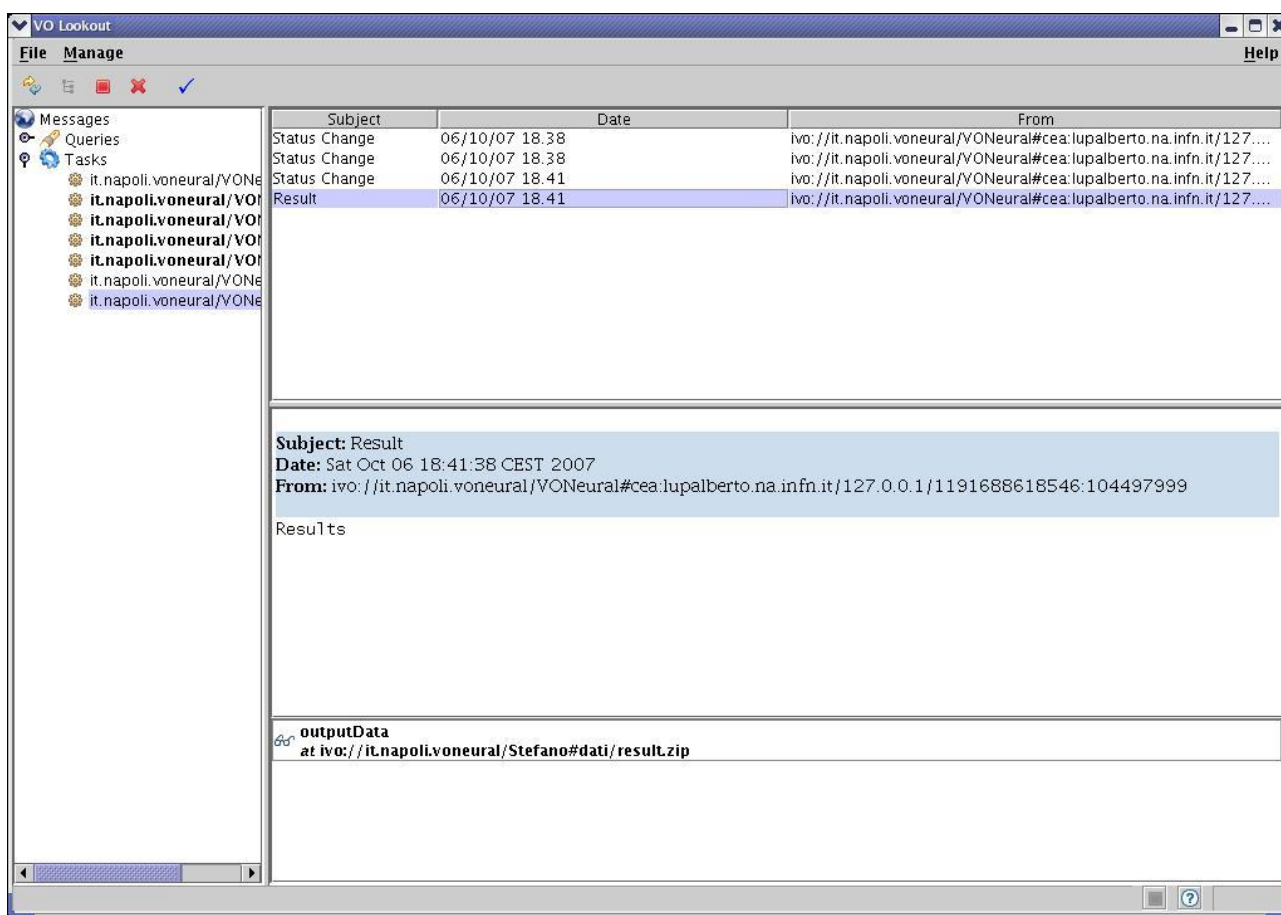


Fig. 9: Results Window

9. Unzip the file on local disk and use “Topcat” to read .tra file. This file is a table formed by two columns, the first containing the target column of the training set (spectroscopic redshifts of the sample of galaxies) and the second containing the values of the photometric redshifts estimated by the neural network (see Fig. 2).

6 Copyright

LIBSVM are used by the authorization of authors, this is the copyright:

Copyright (c) 2000-2008 Chih-Chung Chang and Chih-Jen Lin
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither name of copyright holders nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

7 Bibliography

1. Chih-Wei Hsu, Chih-Chung Chang and Chih-Jen Lin: Pratical guide to Support Vector Classification
2. Chih-Chung Chang and Chih-Jen Lin: LIBSVM: a library for Support Vector Machines

