Data Mining Tools: From Web to Grid Architectures

Davide Anguita, Arianna Poggi,
Fabio Rivieccio, and Anna Marina Scapolla

DIBE Dept. of Biophysical and Electronic Engineering,
University of Genoa, Via Opera Pia 11a, 16145 Genova, Italy
{anguita, apoggi, rivieccio, ams}@dibe.unige.it

Abstract. The paradigm of Grid computing is establishing as a novel, reliable and effective method to exploit a pool of hardware resources and make them available to the users. Data-mining benefits from the Grid as it often requires to run time consuming algorithms on large amounts of data which maybe reside on a different resource from the one having the proper data-mining algorithms. Also, in recent times, machine learning methods have been available to the purposes of knowledge discovery, which is a topic of interest for a large community of users. The present work is an account of the evolution of the ways in which a user can be provided with a data-mining service: from a web interface to a Grid service, the exploitation of a complex resource from a technical and a user-friendliness point of view is considered. More specifically, the goal is to show the interest/advantage of running data mining algorithm on the Grid. Such an environment can employ computational and storage resources in an efficient way, making it possible to open data mining services to Grid users and providing services to business contexts.

1 Introduction

The availability of large amounts of data has raised the interest in various tasks having the common goal of finding complex relationships between the supplied data. These tasks, often referred to as *data mining*, include (among others) classification, regression and clustering: the first two deal with the prediction of a value (in an ordered or unordered set) and the last is related to estimating a probability density. Machine Learning [19] provides methods and theory to perform these tasks in an inductive and data-based fashion, where a data-set is regarded as a collection of samples coming from an input/output statistical relation. This relation is *learnt* by means of algorithms working on data and the resulting Machine (e.g. a classifier or a regressor) can be used to forecast future output values on the basis of some input. The Support Vector Machine [8] is currently rated among the best performing algorithms and can be used to accomplish any of the three tasks above (and more). The recent inclusion of this algorithm in the data-mining suite of a well known database product [6] witnesses this method is fit for a large scale practical exploitation.

In this scenario, the paradigm of Grid computing can be seen as a natural and fruitful evolution of the software layer on which data-mining builds. Data mining algorithms need a large amount of storage and computational resources, and Grid computing is a way to meet these requirements. A Grid enabled environment should provide the core processing capabilities with secure, reliable and scalable high bandwidth access to the various distributed data sources and formats across various administrative domains [14]. When running computationally intensive processes in a dynamic Grid environment, a further advantage comes from having an accurate representation of the available resources and their current status. An example of the adoption of data–mining techniques in conjunction with a grid–enabled technology is given by the `NEOS` server [15].

The next section clarifies the main technical issues related to the SVM algorithm while section 3 describes in greater detail the two realizations of the SVM model selection and testing as an on-demand service. Paragraph 3.1 deals with the deployment of SVM as a web-accessible service, while paragraph 3.2 is focused on the SVM Grid service. The paper ends with a hint on possible future developments of the Grid service (Sec.4).

2 Data Mining Through Support Vector Machines

The Support Vector Machine was presented by Vapnik and Cortes [8] in the mid of the nineties. The method has proven an effective data mining tool since it was successfully adopted to solve problems in diverse fields. A predictive model is generated by induction on the sole basis of "examples", i.e. couples of input/ouput objects which underlie an unknown relation. The goal of the method is to learn a model for this relation.

This paradigm lies within the so-called *Statistical Learning Theory* [19], developed by Vapnik and Chervonenkis in the early seventies. The theory gives the method a sound background from various points of view:

– The procedure which identifies the final model is principled as it embeds a requirement of smoothness together with the obvious requisite to score a low number of errors.
– Effective criteria are available to select a model endowed with good generalization capability. Novel methods which are theoretically well founded drive the choice of a performing model [4, 2].
– Performance bounds holding in probability exist to estimate the error rate scored on-line by the generated model [3].

The Support Vector Method provides algorithms to tackle very different sorts of problem: classification, regression and clustering. Each of these problems is casted into a constrained quadratic programming (CQP) problem offering many advantages, such as the existence of a unique solution and the availability of many algorithms to solve it in a fast and accurate way.

For the reader's convenience the SVM algorithm for classification is here detailed, anyhow the same way of reasoning can be likewise applied in the other

Table 1. Valid kernel functions

Kernel Type	**Kernel expression**
Linear	$k(x_i, x_j) = x_i \cdot x_j$
Gaussian	$k(x_i, x_j) = e^{-\gamma \vert x_i - x_j \vert^2}$
Polynomial	$k(x_i, x_j) = (\frac{(x_i \cdot x_j)}{ni} + 1)^p$

two data-mining tasks. Two issues are considered when seeking for a good Support Vector Classifier: a low number of training errors and a smooth classifier. These two requirements are embedded in an optimization problem, called *primal problem*. The solution for this problem, if $\mathbf{x} \in \Re^{ni}$, is of the form

$$f(\mathbf{x}) = (\mathbf{w} \cdot \mathbf{x}) + b \tag{1}$$

and $|\mathbf{w}|^{\mathbf{2}}$ is a smoothness measure: the lower the norm, the better the generalization capability. Actually, the margin between the two classes scales as the inverse of $|\mathbf{w}|^{\mathbf{2}}$, hence searching for a smooth function means searching for a large margin classifier. Extension to non-linear classification is possible through the so-called *kernel trick* [1]: the basic idea is to replace each dot product appearing in the problem formulation with a suitable *kernel function* thus enabling a non-linear separation of the data. The theoretical justification of this peculiar procedure is to map the data-points into a high dimensionality Hilbert space, where the chance to find a good linear separation is greatly augmented. The linear separation found in this mapped space (called *feature space*) is then mapped back into the input space and can be described by only means of dot products between mapped points: the dot product in the features space can be expressed by a kernel function in the input space, thus allowing to describe the linear separation in the feature space as a non-linear separation in the input space [7]. The (implicit) seek for a suitable hyperplane in the feature space and thus the identification of a non-linear classifier is made possible by the solution of a problem originating from the primal, called the *dual problem* which contains only dot products between input points (thus making possible the use of the kernel trick). The dual problem is a CQP problem and its solution is given by a set of lagrange multipliers related to the optimization problem (α_i) which minimize the dual cost function and define the classifier in the following way:

$$f(\mathbf{x}) = \sum_{i=1}^{np} \alpha_i y_i k(x_i, x) + b \tag{2}$$

where b is a bias, the value of which can also be assessed from the alphas, and $k(x_i, x)$ is the kernel function. Suitable kernels are:

It turns out that often the solution is sparse (some of the α_i are zero), then the solution can be described (or *supported*) by a number of points less than np, i.e. those points corresponding to the non-zero alphas; for this reason these points are also called *support vectors*. A by product of the whole SVM algorithm is then a selection of the input points which are most relevant to the sake of classification. The final form of the solution is a weighted sum involving the kernel function (see eq.2), where the CQP problem identifies the weights (i.e. the alphas) of the sum. Some other parameters are clearly not identified by the optimization procedures but must be set a priori in order to instruct the CQP. These parameters are sometimes called for this reason *hyperparameters* and they are:

– The values shaping the kernel function (e.g. for RBFs the value of the width, for polynomials the degree, etc.).
– The upper bound on the alpha value (as they should not grow to an infinite value); this also results in setting the overall error relevance.
– A parameter modeling the noise amount (only for regression problems).

Hyperparameters are tightly connected to the final system performance. Statistical Learning Theory features some probabilistic bounds which estimate the on-line error rate related to a certain model; as a by product, one can choose the model (i.e. a certain set of hyperparameters) on the basis of the forecasted performance. Many methods are available to this end; the *resampling* methods (e.g. the *bootstrap* [11]) are among the most used: they build on the concept of splitting the available data into subsets that are used for training and testing various classifiers. The on-line performance is then devised on the basis of the average test error. Despite the problem of how to select a model is quite thorny, practical methods often grant a fair performance: for an account of the vast amount of real-world problems solved through the use of SVM see [13].

3 Deploying a SVM-Based Classifier as an On-demand Service

From the above description it is clear that some efforts must be made to shrink the gap between unskilled users and data mining algorithms. The two following examples show how the Support Vector Algorithms can be offered to users in a friendly way: we first present the Internet-based Smart Adaptive Algorithm Computational (ISAAC) server, and then the implementation of SVM as a Grid service.

The goals shared by these two implementations are:

– No specific knowledge of what the hyperparameters are and of their values is required: the system seeks for the best values in a transparent way.
– The user is allowed to upload his own data without bearing the load of computation required by the algorithm: the mining engine resides in fact on a remote server.
– The architectural layout of the whole system should be easy to upgrade in order to widen the offered solution portfolio with new mining algorithms and facilities.

The main intent that motivates the present work is to set-up an on-demand data-mining service exploiting remote resources and freeing users from the necessity of having expensive hardware and/or a solid know-how in the field of

Machine Learning. Indeed, the remote elaboration is here carried on in the most user-transparent possible way, only requiring the user to perform some basic actions (like identification and data upload) in order to build a proper classifier.

3.1 The ISAAC Server

The ISAAC server hosts a sequence of web pages which allow a user to interactively launch a model selection procedure for finding a Support Vector Classifier based on the data provided by the user himself. The server builds on a cluster of PCs connected through a switch, thus avoiding possible bottlenecks. The user can connect to the system at the following URL:

`http://www.smartlab.dibe.unige.it/`

The system features 8 Intel P4 @ 2.4/2.8GHz nodes with 1GB RAM and an overall storage space of about a TeraByte. Each node of the cluster runs Linux Red Hat 9 and the hosting is granted by the `Apache` web server. Users are able to access the system (without having to register in the system and supply a password) and upload data through an upload form. The web connection offers the standard security provided by the `https` protocol. In this case the user navigates in different forms which allow him/her to upload the data, launch an elaboration including model selection with historical data and test the built model on new data. The user is finally sent the communication of the estimated output values via e-mail. A software infrastructure made of `bash` procedures called via `php` provides the necessary substrate for the system to function properly. No user registration other than email address storing has been provided here. In compliance with the desired features, the ISAAC system can be updated with new mining algorithms without affecting the general process architecture, as all the requests are forwarded to a unique Requests Queue Manager (RQM). This module decides when the overall load is low enough to launch another process: it gains information about each node load state through monitoring facilities and passes to the Process Launching Module (PLM) the reference to the job request. The PLM launches the jobs and is responsible to instruct the Results Delivering System (RDS) about which user has to be contacted when a certain job ends. The general architecture is depicted in Fig.1.

All the procedures running in the ISAAC system are specific and cannot be ported without a significant intervention to fit the code to novel hardware resources. Also, the addition of new nodes to the cluster is tightly bounded to the OS, which should run some Linux distribution in order to avoid a time consuming tuning phase. In other words, a scaling of the whole system to a wider cluster of resources could not be easily performed. As already pointed out, the security of the system (which is critical when uploading sensible data) is demanded to the standard `https` protocol. All the problems presented are commonplace in a cluster of workstations: in particular, the security issues are wholly demanded to the server, which should filter malicious actions coming from the outer world; security issues among the cluster nodes are clearly not

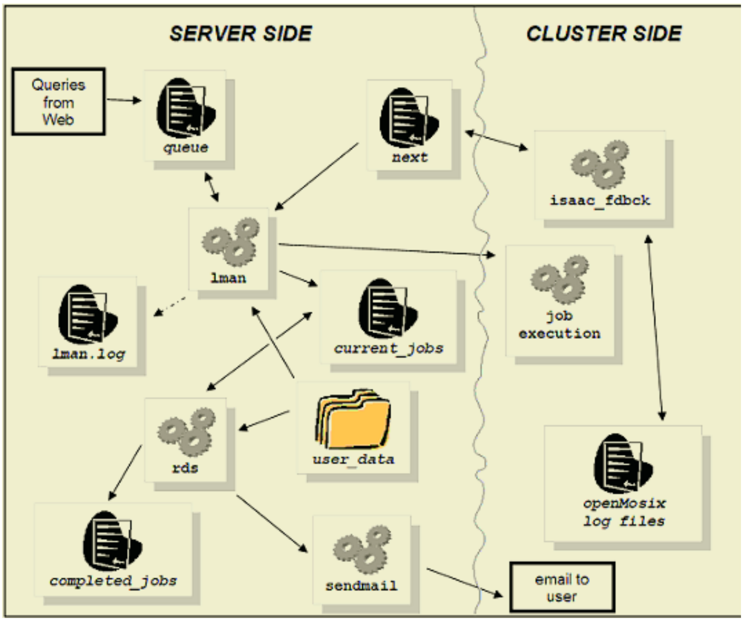


Fig. 1. The ISAAC process architecture

considered. The paradigm of Grid computing addresses these problems (among others) offering a more reliable and effective solution.

3.2 SVM as a Grid Component

Grid computing is establishing as the state-of-the-art technology to connect remote and heterogeneous resources and make them available to the users' community. The scenario opened by Grid computing features a dynamic optimization of the computational load across the whole pool of resources, also allowing the delivering of payback services in a business framework. A Grid infrastructure at DIBE (Department of Biophysical and Electronic Engineering) is being built up [10] as a embryonal testing environment for Grid distributed technology with a single administrative domain. This specific Grid is focused on data-mining problems, and the tool considered for tackling such problems is the SVM algorithm. The Grid structure also allows power users to add new mining tools so to set up a modular mining framework without having to complement every new method with some ad-hoc code. The environment is open to evolution incorporating other nodes (e.g. university departments or business organizations) and it is able to become an effective Grid-based system, geographically distributed and comprehensive of different organizations each one with its distinctive network and security policies. The hardware architecture is composed by some nodes running different Linux distributions and the Globus Toolkit v.3.2 middleware: the *de facto* standard to build a Grid infrastructure. One of these nodes is constituted by the above described Isaac System. More precisely, only the web server is a Grid node and the cluster remains in a private sub-network. The idea is to extend and improve the Isaac system, that has shown security and scalability limits, providing an on-demand data-mining service, a secure, reliable, scalable and portable service for the Grid users. In fact, the requirements for a data-mining algorithm are a secure transfer of data between the client application (user interface) and the Grid node, as well as a large amount of computational resources for a quick response to the user. The requests of data-mining elaborations are met through Grid Service techniques: a distributed computing technology that allows creating client/server applications and obtaining interoperability between different operating systems, programming languages and object models. Grid Services are an extension of Web Services technology so they inherit all advantages of that technology plus some features. They are platform-independent and language-independent, since they use the standard XML language and the Internet standard HTTP (HyperText Transfer Protocol). Distributed technologies already existing like CORBA, RMI, EJB, result in highly coupled distributed systems, where the client and the server are very dependent on each other; Web Services are instead oriented towards loosely coupled systems and meet the demands of an Internet-wide application, just like Grid-oriented applications [12]. Until now Grid Services can be regarded as a "patch" over the Web Services (WS) specification, but from the beginning of this year, a new WS specification

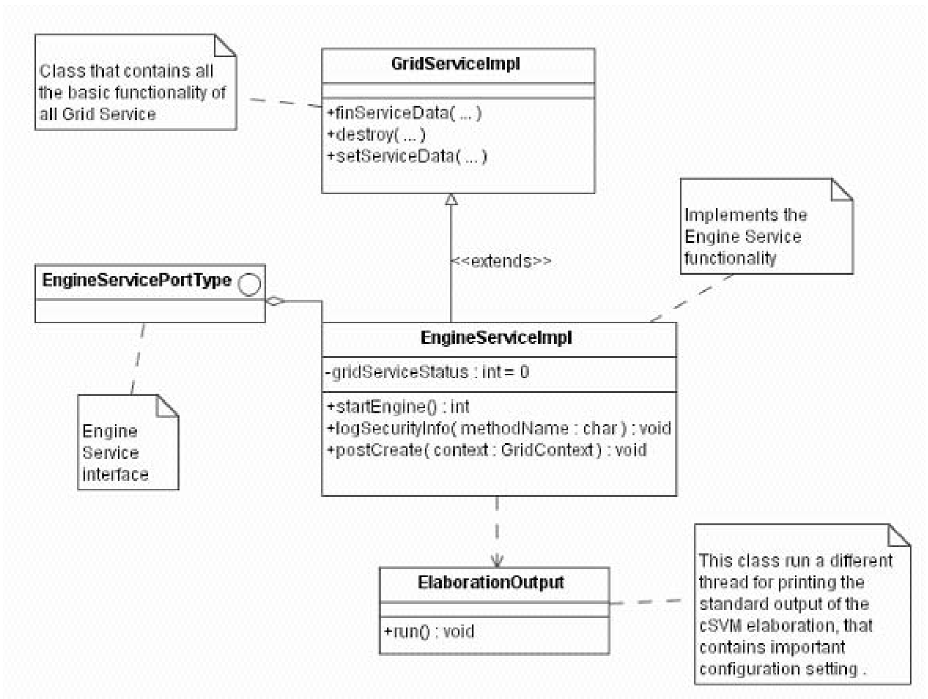
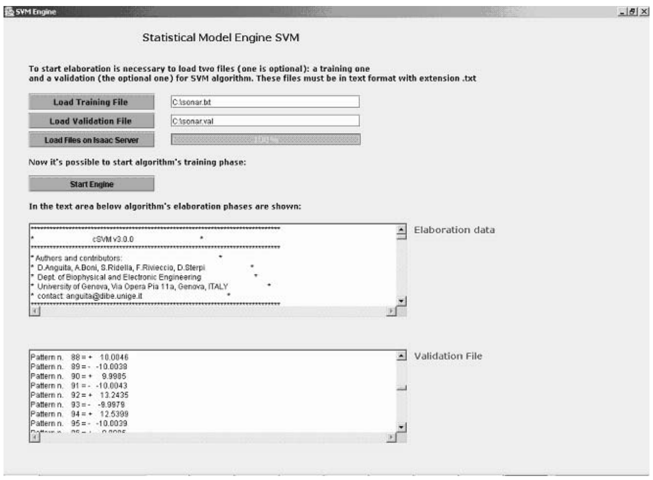

Class that contains all the basic functionality of all Grid Service
GridServiceImpl
+finServiceData(...)
+destroy(...)
+setServiceData(...)
Implements the Engine Service functionality
<<extends>>
EngineServicePortType
EngineServiceImpl
-gridServiceStatus : int = 0
+startEngine() : int
+logSecurityInfo(methodName : char) : void
+postCreate(context : GridContext) : void
Engine Service Interface
ElaborationOutput
+run() : void
This class run a different thread for printing the standard output of the cSVM elaboration, that contains important configuration setting .

Fig. 2. Engine Service Class Diagram

is coming up: the WSRF, Web Services Resource Framework [20]. This framework defines a family of specifications for accessing stateful resources using Web services and includes the WS-ResourceProperties, WS-ResourceLifetime, WS-BaseFaults, and WS-ServiceGroup specifications. The motivation for this is that while Web service implementations typically do not maintain the state information during their interactions, their interfaces have the need to frequently allow for the manipulation of the state, that is, data values that persist across and evolve as a result of Web service interactions [18]. The choice of developing a Grid Service is justified by the great portability, scalability and security of this kind of applications. In this secure environment, only accessible to authorized logins (i.e. certificated by a custom DIBE Certification Authority), users requiring data mining services have the peacefulness that their sensible data are transferred and elaborated over a secure environment. The service is called Engine Service (like the name of its PortType), and its class EngineServiceImpl extends Grid ServiceImpl, the base class that provides the basic functionality for a Grid Service. The only exposed method is startEngine() that starts the SVM elaboration on the Grid. The other two methods are not exposed in the Grid Service but they implement the service security (see Fig.2). A client application is provided to give a user-friendly interface to interact with the service. From this one, it is possible to transfer the input data files and start the elaboration, involving the identification and building of a proper model for a support vector classifier. The user can upload two files: one is used for training and is made of couples of input/output values, the other is only composed of input values (called *validation* data), the task of the classifier being an estimation of the correct labels. If the user does not set any of the hyperparameters, the algorithm performs the model selection procedure and seeks for the set of hyperparameters which minimizes the estimated generalization error. The results are displayed in two text areas: one for the model, which supplies detailed information about the learning phase, and one for the validation, which tells how fairly the model is performing on the validation data. The mechanism of file transfer is implemented using GridFTP: a high-performance, secure, reliable data transfer protocol optimized for high-bandwidth wide-area networks. The GridFTP protocol is based on FTP, the highly-popular Internet file transfer protocol [16](see Fig.3).

4 Final Considerations

The system presented above is oriented towards on-demand computing. A future refinement in the sense of a dynamic and effective distribution of the computational load will improve the overall quality of service offered to users. In a scenario in which one tries to exploit both the Grid capabilities and the cluster computational power, this first layer could be exclusively managed by the Grid. An evolution is planned to grow the complexity of the software architecture to enable a faster model selection procedure: once the data is uploaded by the user, the estimation of the generalization ability of the various candidate models could be spread across the available resources; this second layer could



Statistical Model Engine SVM

To start elaboration is necessary to load two files (one is optional): a training one
and a validation (the optional one) for SVM algorithm. These files must be in text format with extension .txt

C:\sonar.txt

C:\sonar.val

Now it's possible to start algorithm's training phase:

In the text area below algorithm's elaboration phases are shown:

```
****************************************************************
*                    cSVM v3.0.0                        *
****************************************************************
* Authors and contributors:                              *
*  D.Anguita, A.Boni, S.Ridella, F.Rivieccio, D.Sterpi          *
*  Dept. of Biophysical and Electronic Engineering               *
*  University of Genova, Via Opera Pia 11a, Genova, ITALY              *
*  contact: anguita@dibe.unige.it                        *
****************************************************************
```

```
Pattern n.   88 = +   10.0046
Pattern n.   89 = -  -10.0039
Pattern n.   90 = +    9.9985
Pattern n.   91 = -  -10.0043
Pattern n.   92 = +   13.2435
Pattern n.   93 = -   -9.9979
Pattern n.   94 = +   12.5399
Pattern n.   95 = -  -10.0039
```

Fig. 3. Engine Service Client Interface Appearance

be managed both by the Grid level and the cluster one. Each evaluation can be performed in a parallel fashion thus speeding up the identification of the estimated best model. Some methods are also available in literature to speed the CQP optimization based on previously available solutions related to a different set of hyperparameters (e.g. *alpha seeding* procedures [5]).

Besides these two layers into which the service can be allotted, a third and deeper level of parallelization can be foreseen. Recently some algorithms have been made available ([21]) which allow to parallelize each CQP problem, thus opening to new scenarios in which a single classification request will be distributed into several tasks on the basis of a two-folded parallelization: the classification task is at first split into the parallel testing of several potential models and secondly the evaluation of each of these models is shared among a number of nodes which contribute to solving the same CQP problem. This third layer could be exclusively performed by the cluster, thus enabling better performance.

The challenges lying in this envision are the proper allocation of the tasks coming from different levels of optimization and the handling of concurrent requests coming from different clients, in order to lower the response time perceived by the single user.

References

1. M. A. Aizerman, E. M. Braverman, and L. I. Rozoner. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.
2. D. Anguita, S. Ridella, F. Rivieccio, R. Zunino, Hyperparameter design criteria for support vector classifiers, *Neurocomputing*, Vol. 55, N. 1-2, pp. 109-134, 2003.

3. D. Anguita, A. Boni,D. Sterpi, S. Ridella, F. Rivieccio, *Theoretical and Practical Model Selection Methods for Support Vector Classifiers*, in "Support Vector Machines: Theory and Applications", by L. Wang (Ed.), Springer, in press.
4. P. L. Bartlett, S. Boucheron, and G. Lugosi. Model selection and error estimation. *Mach. Learn.*, 48(1-3):85–113, 2002.
5. D. DeCoste, K. Wagstaff, Alpha seeding for support vector machines, *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pp.345–359, 2000.
6. Oracle Corp. Discover patterns, make predictions, develop advanced BI A pplications, January 2004.
7. M. G. Genton, Classes of Kernels for Machine Learning: A Statistics Perspective, *Journal of Machine Learning Research*, vol. 2, pp. 299-312, 2001.
8. C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.
9. D. Anguita, A. Boni, and S. Ridella. Evaluating the generalization ability of support vector machines through the bootstrap. *Neural Processing Letters*, 11(1), 2000.
10. D. Anguita, A. Poggi, and A.M. Scapolla. Smart adaptive algorithm on the grid. *11th Plenary HP-OUVA Conference, June 2004, Paris*, page 2, 2004.
11. B. Efron and R. Tibshirani. *An Introduction to the Bootstrap.* Chapman and Hall, London, 1993.
12. I. Foster, C. Kesselman, J. Nick, and S. Tuecke. The physiology of the grid: An open grid services architecture for distributed systems integration, 2002.
13. I. Guyon. Online svm application list, available on web at: http://www.clopinet.com/isabelle/projects/svm/applist.html.
14. BeSC-Belfast e-Science Centre Online available on web at: http://www.qub.ac.uk/escience/projects/geddm/.
15. J. Czyzyk, M. P. Mesnier, J. J. Mor, "The NEOS Server", *IEEE Computational Science and Engineering*, Vol.5, N.3, pp. 68-75, 1998.
16. Globus-Project, "GridFTP, Universal Data Transfer for the Grid", *White Paper*, Sept. 2000.
17. T. Poggio and S. Smale, The mathematics of learning: Dealing with data, *Amer. Math. Soc. Notice*, Vol.50(5),pp. 537–544, 2003.
18. IBM-Report. Web services resource framework, March 2004.
19. V. Vapnik. *Statistical Learning Theory.* Wiley, 1998.
20. I. Foster, J. Frey, S. Graham, S. Tuecke, K.Czajkowski, D. Ferguson, F. Leymann, M. Nally, I. Sedukhin, D. Snelling, T.Storey, W.Vambenepe, S.Weerawarana, Modeling Stateful Resources with Web Services, whitepaper available at: http://www-106.ibm.com/developerworks/library/ws-resource/ws-modelingresources.pdf
21. G. Zanghirati, L. Zanni, "A Parallel Solver for Large Quadratic Programs in Training Support Vector Machines", *Parallel Computing*, 29 (2003), 535-551.