Simone Bassis
Anna Esposito
Francesco Carlo Morabito
Editors

SMART INNOVATION, SYSTEMS AND TECHNOLOGIES ■ 26

Recent Advances of Neural Network Models and Applications

Proceedings of the 23rd Workshop of the Italian Neural Networks Society (SIREN), May 23–25, Vietri sul Mare, Salerno, Italy

SIREN

Springer

Smart Innovation, Systems and Technologies

Volume 26

For further volumes:
http://www.springer.com/series/8767

About this Series

The Smart Innovation, Systems and Technologies book series encompasses the topics of knowledge, intelligence, innovation and sustainability. The aim of the series is to make available a platform for the publication of books on all aspects of single and multi-disciplinary research on these themes in order to make the latest results available in a readily-accessible form. Volumes on interdisciplinary research combining two or more of these areas is particularly sought.

The series covers systems and paradigms that employ knowledge and intelligence in a broad sense. Its scope is systems having embedded knowledge and intelligence, which may be applied to the solution of world problems in industry, the environment and the community. It also focusses on the knowledge-transfer methodologies and innovation strategies employed to make this happen effectively. The combination of intelligent systems tools and a broad range of applications introduces a need for a synergy of disciplines from science, technology, business and the humanities. The series will include conference proceedings, edited collections, monographs, handbooks, reference books, and other relevant types of book in areas of science and technology where smart systems and technologies can offer innovative solutions.

High quality content is an essential feature for all book proposals accepted for the series. It is expected that editors of all accepted volumes will ensure that contributions are subjected to an appropriate level of reviewing process and adhere to KES quality principles.

Simone Bassis · Anna Esposito
Francesco Carlo Morabito
Editors

Recent Advances of Neural Network Models and Applications

Proceedings of the 23rd Workshop of the Italian Neural Networks Society (SIREN), May 23–25, Vietri sul Mare, Salerno, Italy

Springer

Editors
Simone Bassis
Department of Computer Science
University of Milano
Milano
Italy

Anna Esposito
Department of Psychology
Second University of Naples Caserta,
and
Institute for Advanced Scientific Studies
(IIASS)
Vietri sul Mare (Salerno)
Italy

Francesco Carlo Morabito
DICEAM
Department of Civil, Energy,
Environmental, and Materials
Engineering
University Mediterranea of Reggio Calabria
Reggio Calabria
Italy

ISSN 2190-3018 ISSN 2190-3026 (electronic)
ISBN 978-3-319-04128-5 ISBN 978-3-319-04129-2 (eBook)
DOI 10.1007/978-3-319-04129-2
Springer Cham Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013956359

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

This volume collects a selection of contributions which have been presented at the 23rd Italian Workshop on Neural Networks (WIRN 2013), the yearly meeting of the Italian Society for Neural Networks (SIREN). The conference was held in Italy, Vietri sul Mare (Salerno), during May 23–24, 2013. The annual meeting of SIREN is sponsored by International Neural Network Society (INNS), European Neural Network Society (ENNS) and IEEE Computational Intelligence Society (CIS). The workshop, and thus this book, is organized in two main components, a special session and a group of regular sessions featuring different aspects and point of views of artificial neural networks, artificial and natural intelligence, as well as psychological and cognitive theories for modeling human behaviours and human machine interactions, including Information Communication applications of compelling interest.

More than 50 papers were presented at the Workshop, and most of them are reported here. The review process has been carried out in two steps, one before and one after the workshop in order to meet the Publisher requirements. The selection of the papers was made through a peer-review process, where each submission was evaluated by at least two reviewers. The submitted papers were authored by peer scholars from different countries (the Italian component was anyway preponderant). The acceptance rate was high because authors got the chance to review in two steps their work and also because they are experts in the field, being most of them involved in the organization of SIREN research activities for more than 20 years. In addition to regular papers, the technical program featured keynote plenary lectures by worldwide renowned scientists (Sankar Kumar Pal, India; Sara Rosenblum, Israel; Laurence Likforman, France; Virginio Cantoni, Italy).

The special session on Social and Emotional Networks for Interactional Exchanges was organized by Gennaro Cordasco, Anna Esposito and Maria Teresa Riviello (Department of Psychology, Second University of Naples, Italy). The Session explored new ideas and methods for developing automatic systems capable to detect and support users psychological wellbeing gathering information and meanings from the behavioral analysis of individual interactions both at

the micro (dyadic and small groups) and macro level (information and opinion transfer over a large population). Of particular interest was the analysis of sequences of group actions explicated through a series of visual, written and audio signals and the corresponding computational efforts to detect and interpret their semantic and pragmatic contents. Social networking and connectedness as the ability to spread around thinking and related effects on social network behaviors were also considered, as well as ICT applications detecting the health status and affective states of their users. The special session's invited lectures were given in honour of Professors Maria Marinaro and Luigi Maria Ricciardi which directed the activities of the hosting Institute, the International Institute for Advanced Scientific Studies (IIASS), for more than a decade, sustaining the Workshop and sponsoring SIREN's activities.

The organization of an International Conference gathers the efforts of several people. We would like to express our gratitude to everyone that has cooperate to its organization, by offering their commitment, energy and spare time to make this event a successful one. Finally, we are grateful to the contributors of this volume for their cooperation, interest, enthusiasm and lively interactions, making it not only a scientifically stimulating gathering but also a memorable personal experience.

May 2013

Simone Bassis
Anna Esposito
Francesco Carlo Morabito

Organization

WIRN 2013 is organized by the Italian Society of Neural Networks (SIREN) in co-operation with the International Institute for Advanced Scientific Studies (IIASS) of Vietri S/M (Italy).

Executive Committee

Bruno Apolloni	University of Milano, Italy
Simone Bassis	University of Milano, Italy
Anna Esposito	Second University of Naples and IIASS, Italy
Francesco Masulli	University of Genova, Italy
Francesco Carlo Morabito	University Mediterranea of Reggio Calabria, Italy
Francesco Palmieri	Second University of Napoli, Italy
Eros Pasero	Polytechnic of Torino, Italy
Stefano Squartini	Polytechnic University of Marche, Italy
Roberto Tagliaferri	University of Salerno, Italy
Aurelio Uncini	University "La Sapienza" of Roma, Italy
Salvatore Vitabile	University of Palermo, Italy

Program Committee

Conference Chair:	Francesco Carlo Morabito (University Mediterranea of Reggio Calabria, Italy)
Conference Co-Chair:	Simone Bassis (University of Milan, Italy)
Program Chair:	Bruno Apolloni (University of Milan, Italy)
Organizing Chair:	Anna Esposito (Second University of Napoli and IIASS, Italy)
Special Tracks:	Gennaro Cordasco, Anna Esposito and Maria Teresa Riviello (Second University of Napoli and IIASS, Italy)

Referees

V. Arnaboldi
S. Bassis
G. Boccignone
A. Borghese
F. Camastra
P. Campadelli
R. Carbone
C. Ceruti
A. Ciaramella
M. Corazza
G. Cordasco
V. D'Amato
A. De Candia
R. De Rosa
F. Epifania
A. Esposito
A. Esposito
M. Fiasch
R. Folgieri
M. Frasca
C. Gallicchio
S. Giove
G. Grossi
F. La Foresta
D. Malchiodi
U. Maniscalco
F. Masulli
M. Mesiti
A. Micheli
P. Mottoros
C. Orovas
F. Palmieri
E. Pasero
F. Piazza
G. Piscopo
M. Re
M.T. Riviello
A. Roli
S. Rovetta
A. Rozza
M. Russolillo
G. Sarn
S. Scardapane
M. Scarpiniti
R. Serra
A. Sperduti
S. Squartini
A. Staiano
R. Tagliaferri
A. Uncini
G. Valentini
L. Valerio
S. Valtolina
M. Villani
S. Vitabile
J. Vitale
A. Zanaboni
A. Zippo
C. Zizzo
I. Zoppis

Sponsoring Institutions

International Institute for Advanced Scientific Studies (IIASS)
of Vietri S/M (Italy)
Department of Psychology, Second University of Napoli (Italy)
Provincia di Salerno (Italy)
Comune di Vietri sul Mare, Salerno (Italy)

Contents

WIRN 2013

Part I: Algorithms

Part II: Signal Processing

Part I
Algorithms

Identifying Emergent Dynamical Structures in Network Models

Marco Villani[1,2], Stefano Benedettini[1], Andrea Roli[1,3], David Lane[1,4], Irene Poli[1,5], and Roberto Serra[1,2]

[1] European Centre for Living Technology, Ca' Minich, S. Marco 2940, 30124 Venezia, Italy
[2] Dept. of Physics, Informatics and Mathematics,
University of Modena e Reggio Emilia, v. Campi 213b, 41125 Modena, Italy
{marco.villani,roberto.serra}@unimore.it
[3] DISI Alma Mater Studiorum University of Bologna Campus of Cesena,
via Venezia 52, I-47521 Cesena, Italy
andrea.roli@unibo.it
[4] Dept. of Communication and Economics, University of Modena e Reggio Emilia,
v. Allegri 9, 41121 Reggio emilia, Italy
lane@unimore.it
[5] Department of Environmental Sciences, Informatics and Statistics,
University Ca'Foscari, Venice, Italy
irene.poli@unive.it

Abstract. The identification of emergent structures in dynamical systems is a major challenge in complex systems science. In particular, the formation of intermediate-level dynamical structures is of particular interest for what concerns biological as well as artificial network models. In this work, we present a new technique aimed at identifying clusters of nodes in a network that behave in a coherent and coordinated way and that loosely interact with the remainder of the system. This method is based on an extension of a measure introduced for detecting clusters in biological neural networks. Even if our results are still preliminary, we have evidence for showing that our approach is able to identify these "emerging things" in some artificial network models and that it is way more powerful than usual measures based on statistical correlation. This method will make it possible to identify mesolevel dynamical structures in network models in general, from biological to social networks.

Keywords: Dynamical systems, emergent dynamical structures, cluster index, boolean networks, emergent properties.

1 Introduction

Emergent phenomena are among the most intriguing ones in natural as well as in artificial systems. Indeed, it can be argued [1] that neural networks represent an attempt at shaping the emergent properties of a set of models in order to perform some required tasks. An intriguing aspect is the "sandwiched" nature of most emergent

S. Bassis et al. (eds.), *Recent Advances of Neural Network Models and Applications*,
Smart Innovation, Systems and Technologies 26,
DOI: 10.1007/978-3-319-04129-2_1, © Springer International Publishing Switzerland 2014

phenomena: while past researches were almost exclusively focused on bottom-up emergence in two-level systems (like e.g. Benard-Marangoni convection cells emerging from the interaction of the water molecules [2]) it is becoming increasingly clear that in the most interesting cases the new entities and levels do emerge between pre-existing ones. The paradigmatic example may be that of organs and tissues in multicellular organisms: both the lower (cellular) level and the upper one (organism) predate the appearance of the intermediate structures. Other examples come from the physical world (e.g. mesolevel structures in climate) and social systems (e.g. various factions within political parties). What is more interesting in the present case is that also in artificial systems, like neural networks, one observes the formation of intermediate-level circuits between the single neurons and the global properties. It goes without saying that some neural architectures have been devised precisely to stimulate the formation of these mesolevel structures, but here we are concerned with structures that come into being by spontaneous processes without being explicitly designed from the outside (although a certain type of design may ease or prevent the formation of these spontaneous structures).

A central question is then that of identifying the emerging "things": these may be either static entities or dynamical patterns, or some mixture of the two. In dynamical networks, static emergent structures take the form of topological features, like e.g. motifs in genetic networks or communities in a broader context. There is an extensive literature on community detection, so we will concentrate here on a different type of mesolevel structures, namely those that are created by the dynamical interactions in the network. Nodes may work together although they are not directly linked, since the dynamical laws may give rise to different parts working together. If the topology were regular, these nodes might be identified by visual inspection, but in the case of irregular topologies this approach seems hopeless.

In this paper we present a first step towards the development of formalized methods to identify these mesolevel "things": since they may have a topological as well as a dynamical nature, we refer to them as mesolevel dynamical structures (MDS). The task of identifying MDSs is a formidable one, so we will show here the outline of a promising approach and some preliminary results, while remarking that there are still more open questions than answers. However, the interest of the task motivates in our opinion the opportunity to report our preliminary results.

In order to escape "bird's eye" detection methods, we will consider different subsets of the network, looking for those whose nodes appear to be well coordinated among themselves and have a weaker interaction with the rest of the nodes. For each subset of nodes we will measure its so-called cluster index, a measure based on information theory that had been proposed by Tononi and Edelman [3]. After a suitable normalization procedure (see the following for the details) we rank the various subsets in order to identify those that are good candidates for the role of partially independent "organs" (note that they not necessarily exist in any network).

The cluster index has been defined so far for quasi-static systems, and we will discuss its extension to nonlinear dynamical systems. We will also show the result of the application of this ranking method to some model systems, including some synthetic dynamical networks and some genetic regulatory networks proposed by

examining the biological literature. The method draws our attention on subsets that are functionally correlated and that represent an interesting hypothesis about possible MDSs. In the end we will also comment on the fact that our method, although not yet fully developed, already outperforms usual correlation techniques.

2 Some Useful Definitions

For the sake of definiteness, let us consider a system U, our "universe" that is a network of N nodes that can change in discrete time, taking one of a finite number l of discrete values (in the examples we will choose l=2 for simplicity). The value of node i at time t+1, $x_i(t+1)$, will depend upon the values of a fixed set of input nodes at time t, possibly including the i-th (self-loops are not prohibited). In several cases, networks start with a random state and change according to the evolution rules so the initial state may bear no relationship to the system itself. Since we are interested in finding out some properties of the networks themselves, we will consider their behaviors after an adequate relaxation time. For the time being we will also ignore external influences on some nodes, although these might be easily included.

The entropy of a single node is estimated from a long time series by taking frequencies f_v of observed values in time as proxies for probabilities, and is defined as

$$H_i = -\sum_{v=1}^{m} f_v \log f_v \tag{1}$$

where the sum is taken over all the possible values a node can take.

If the system is deterministic and is found in a fixed point attractor, H_i=0 for every node, since each node takes its value with frequency one. In order to apply entropy-based methods, Edelman and Tononi considered a system subject to gaussian noise around an equilibrium point. In our case it is however appropriate to deal with a richer time behavior since nonlinear networks can have several different attractors, each attractor contributing to the behavior of the system (though in different times). So our "long data series" will be composed by several repetitions of a single attractor, followed by repetitions of another one, etc. (ignoring the short transients between two attractors) . The number of times a single attractor is represented in the data series should be weighted in some way: there are possible several different strategies, depending on the nature of the system we are analyzing. In case of noisy systems a possibility is that of estimating the weights of the attractors by measuring the persistence time of the systems in each of them [4]; deterministic systems might be analyzed by weighting attractors with their basins of attraction. For simplicity in the following we opt for this second choice.

Now let us look for interesting sets of nodes (clusters, from now on). A good cluster should be composed by nodes (i) that possess high integration among themselves and (ii) that are more loosely coupled to other nodes of the system. The measure will define, called the cluster index, is not a Boolean one, but it provides a measure of "clusterness" that can be used to rank various candidate clusters (i.e., emergent intermediate-level sets of coordinated nodes).

3 Measuring the Cluster Index

Following Edelman and Tononi [3], we define the cluster index *C(S)* of a set of *k* nodes *S*, as the ratio of a measure of their integration *I(S)* to a measure of the mutual information *M(S\U-S)* of that cluster with the rest of the system.

The integration is defined as follows: let *H(S)* be the entropy (computed with time averages) of the elements of *S*. This means that each element is a vector of *k* nodes, and that the entropies are computed by counting the frequencies of the k-dimensional vectors. Then:

$$I(S) = \sum_{j \in S} H(x_j) - H(S) \tag{2}$$

The first term is the sum of the single-node entropies, the last one is computed using vectors of length *k*, so *I* measures the deviation from statistical independence of the *k* elements in S[1]. The mutual information of *S* to the rest of the world *U-S* is also defined by generalizing the usual notion of mutual information between nodes to *k* dimensional vectors

$$M(S;U-S) \equiv H(S) - H(S \mid U-S) = H(S) + H(U-S) - H(S,U-S) \tag{3}$$

where, as usual, *H(A|B)* is the conditional entropy and *H(A,B)* the joint entropy.

Finally, the cluster index *C(S)* is defined by

$$C(S) = \frac{I(S)}{M(S;U-S)} \tag{4}$$

The cluster index vanishes if $I=0$, $M \neq 0$, and is not defined whenever $M=0$. For this reason, the approach based upon cluster indices does not work properly when the mutual information of *S* with the rest of the system vanishes; these cases, in which *S* is statistically independent from the rest of the system – a significant property because they signal particularly strong structures - can be diagnosed in advance.

In this way, for every subsystem *S* we will get a measure of its quality as a cluster. In order to identify potential MDSs it is necessary to compare the indices of various candidate clusters. It is straightforward to compare clusters of the same size using *C(S)*, but unfortunately *C* scales with the size of the subsystem, so that a loosely connected subsystem may have a larger index than a more coherent, smaller one. In order to deal with these cases we need to normalize the clusters with respect to their size. The analysis may turn out quite cumbersome, but in most cases we found it sufficient to use a simple prescription, used by Tononi and Edelman in their original paper, which results in the calculation process outlined in the following.

The first step is to define a "null system", i.e., a non-clustered homogeneous system, from which we sample a series. This system provide us with a null hypothesis

[1] H(S) is estimated from the same time series used to calculate the frequencies f_v of eq. (1). So, to compute H(S) we calculate the frequencies f^S_v of the observed values of S seen as a whole.

and allows us to calculate a set of normalization constants, one for each subsystem size. For each subsystem size, we compute average integration $\langle I_h \rangle$ and mutual information $\langle M_h \rangle$ (subscript h stands for "homogeneous"); we can then normalize the cluster index value of any subsystem S in it universe U using the appropriate normalization constants dependent on the size of S:

$$C'(S) = \frac{I(S)}{\langle I_h \rangle} \Big/ \frac{M(S;U-S)}{\langle M_h \rangle} \tag{5}$$

We apply this normalization to both the cluster indices in the analyzed system and in the null system.

The definition of "null system" is critical: it could be problem-specific, but we prefer a simple solution which is fairly general: given a series of Boolean vectors, we compute the frequency of ones b and generate a new random Boolean series where each bit has the same probability b of being one. This random null hypothesis is easy to calculate, related to the original data and parameter-free; moreover we believe it satisfies the requirements set by Tononi of homogeneity and cluster-freeness.

The second step involves the computation of a statistical significance index, called T_c, that is used to rank the clusters in the analyzed system. The T_c of a cluster S is:

$$T_c(S) = \frac{C'(S) - \langle C'_h \rangle}{\sigma(C'_h)} \tag{6}$$

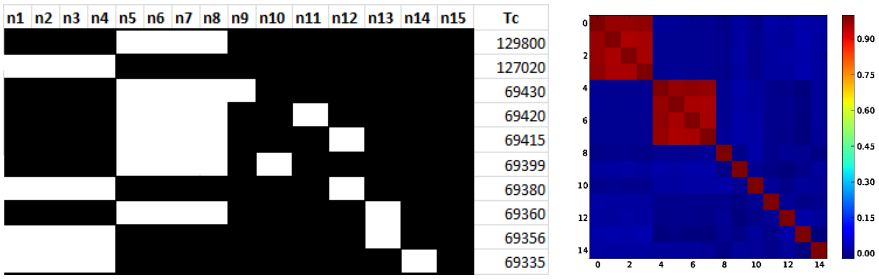
where $\langle C'h \rangle$ and $\sigma(C'h)$ are respectively the average and the standard deviation of the population of normalized cluster indices with the same size of S from the null system [5].

4 A Controlled Case Study

As a first step, we show the results of the application of our method on simple cases in which the systems analyzed have clusters by construction. These experiments make it possible to assess the effectiveness of the approach on controlled case studies, in which the expected outcome of the analysis is known a priori. Since our method aims at finding clusters of nodes which work together --- independently of their connections --- on the basis of sample trajectories of the system, we directly generated trajectories in which some groups of values behave coherently, i.e., they are the clusters to be detected. The trajectories are sequences of binary vectors of length n, $[x_1(t),x_2(t),...,x_n(t)]$. At each time step t, the values of the first c vector positions are generated according to the following procedure: $x_1(t)$, the leader, is a randomly chosen value in {0,1}; the values from position 2 to c, the followers, are a noisy copy of the leader, i.e., $x_i(t)=x_1(t)$ with probability 1-p and $x_i(t)=\sim x_1(t)$ otherwise, being p the noise rate. Values $x_{c+1}(t),...,x_n(t)$ are randomly chosen in {0,1}. This way, the first block of the vector is composed of strongly correlated values and it should be clearly distinguished from the rest of the positions. Besides series with one cluster, with the

same procedure we also generated trajectories with two independent clusters of size c_1 and c_2, respectively. In this case, the clusters can be found in positions $1,..,c_1$ and $c_{1+1},...,c_1+c_2$, where leaders are x_1 and x_{c1+1}. The trajectories were generated with p in $\{0, 0.01, 0.1\}$.

We applied our method based on the calculation of the normalized cluster index and we sorted the clusters as a function of the significance index T_c. In all the cases, the score based on T_c returns correctly the clusters in the first positions of the ranking. As an example, in Figure 1a we show the results of a representative case with two clusters with c_1=c_2=4 in a trajectory with 15 vector positions and p=0.01. The figure shows the ten T_c highest values and the corresponding cluster size. The bars are sorted in not increasing order of T_c. The highest peaks correspond indeed to the two clusters created in the trajectories. Each row of the matrix represents a cluster: white cells are the vector positions included in the cluster and they are ranked, from the top, by not increasing values of T_c. We can see that the first two clusters detected are indeed the ones corresponding to the positions 5,...,8 and 1,...,4 (small differences in T_c between the two clusters are due to noise).



(a) (b)

Fig. 1. (a) Matrix illustrating the elements of the clusters and the corresponding T_c values. The first two clusters are the ones introduced in the trajectory. (b) The heatmap shows the correlation values between pairs of vector positions in the trajectory.

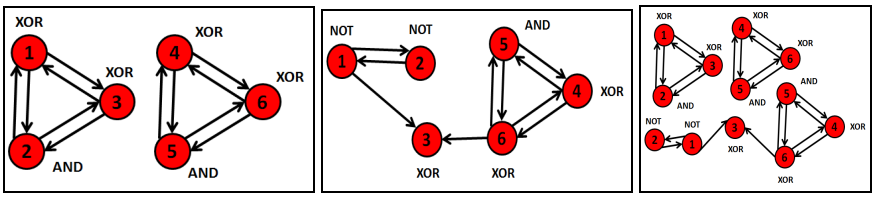
The detected clusters are composed of correlated values, therefore we expect to find them also by simply calculating the correlation between every pair of positions. The correlation is computed by taking the value of the Pearson correlation coefficient for each pair of variables; in the case of binary variables, this quantity is usually called the phi coefficient. Since we are interested indifferently in both positive and negative correlations, we take the absolute value of the phi coefficients. Results can be plotted as heatmaps, with correlation values associated to colors from blue (lowest) to red (highest). An example is given in Figure 1b. As we can observe, the blocks composing the clusters are clearly detected and this result holds for all the trajectories we analyzed. This result is not surprising, as the vector positions composing a cluster are indeed strongly correlated (the only variance is introduced by noise). One might then object that the correlation measure is sufficient to detect clusters. In fact, this argument is only valid in some simple cases and does not extend to the general case. The reason is that correlation is a pairwise measure, while the cluster index accounts

for multiple relations. These first tests enable us to state that our method based on the cluster index can be effectively used to capture multiple correlations among variables. In the next section, we will show that this approach can be particularly powerful in detecting clusters of nodes in networks.

5 Cluster Indices Applied to Network Models

The case study we are going to examine consists of three synchronous deterministic Boolean Networks (BNs) – the BN being a very interesting case of complex systems [6] [7], also applied to relevant biological data [8] [9] [10] and processes [11] [12]. The aim of this case study is to check whether CI analysis is capable of recognizing special topological cases, such as causally (in)dependent subnetworks and oscillators, where the causal relationships are more than binary. Note that in all the following cases the phi analysis is ineffective (doesn't relate any variable, having values different from zero only on the diagonal of the matrix).

The first example is a BN made of two independent sub-networks (RBN1 - Figure 2a); in this case we expect the analysis to identify the two subsystems. The second example (RBN2 - Figure 2b) is a BN made of an oscillator (nodes 0 and 1) and one of the subnetworks form the previous example, node 2 has no feedback connections. In the last example we simply merge the networks form the previous examples (RBN3 system). Figures 3 show the top 5 most relevant clusters according to T_c. CI analysis is able to correctly identify the two subnetworks in the first example (first and second rows). The analysis clusters together 5 of 6 nodes of RBN2: those already clustered in RBN1, plus nodes 1 and 2 (which negates each other - figure 2b) and the node that compute the XOR of the signal coming from the two just mentioned groups. Indeed, all these nodes are needed in order to correctly reconstruct the RBN2 series.

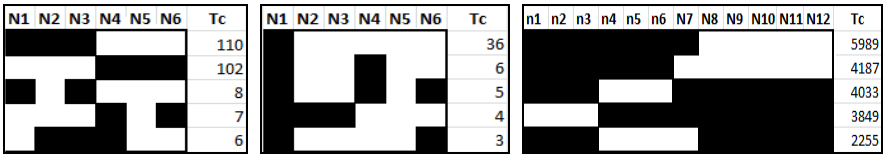


(a) (b) (c)

Fig. 2. (a) independent Boolean networks (RBN1); (b) interdependent networks (RBN2); (c) A system composed by both the previous networks (RBN3). Beside each boolean node there is the boolean function the node is realizing.

In the third example the top two clusters correspond respectively to the 5 nodes already recognized in RBN2 and to the whole RBN2 system, while the third and fourth rows correspond to the independent subgraphs of RBN1: all MDSs are therefore correctly identified.

We would like to point out that CI analysis does not require any knowledge about system topology or dynamics. This information is normally unavailable in real cases; on the other hand, our methodology just needs a data series.



(a) (b) (c)

Fig. 3. Matrix illustrating the elements of the clusters and the corresponding T_c values, for (a) RBN1, (b) RBN2 and (c) RBN3 systems

As a final note, it is important to point out that covariance analysis is inadequate in this scenario as it is not able to identify any cluster. We took the same series we applied CI analysis upon and computed the correlation matrix between the node variables; the correlation indices between nodes are uniformly low in magnitude. The inadequacy of this method can be explained by the fact that correlation only takes into account binary linear interactions between variables as opposed to *CI*, which does not necessitate these hypotheses. Experiments performed using asynchronous update yielded essentially the same results with respect to both *CI* and correlation analyses.

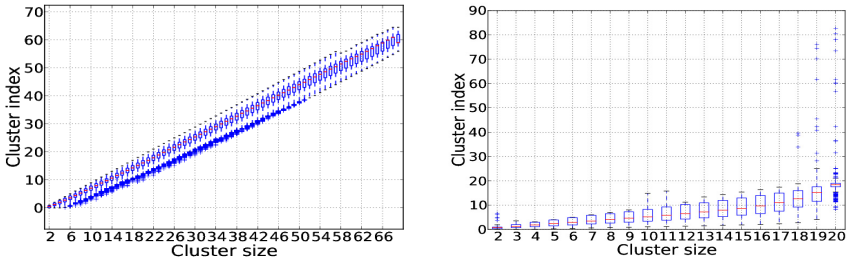
6 Evolved Network: Some Examples

We have shown that our method makes it possible to discover clusters of coordinated nodes in a network. We may then raise the question as to what extent this technique can be used to extract meaningful information about clusters in networks subject to evolution. This would provide insights both into biological and artificial evolutionary processes involving networks. The subject of evolution of clusters is strictly linked to the evolution of modularity [13][14] but we conjecture that clusters form only when certain conditions are verified in the evolutionary process: in particular, we expect that clusters are not needed if the environmental conditions do not require organized subsystems (devoted to specific tasks).

To test this conjecture, we studied cases in which BNs with random topology are evolved for maximizing the minimal distance between attractors and for two classification tasks [15]. These tasks are static and not intrinsically modular; therefore, we expect not to find clusters in these evolved networks. The outcome of our analysis is that all these tasks can be optimally solved by BNs possessing only two complementary attractors. It can be easily shown that in homogeneous cases (systems without clusters) the cluster index scales linearly with the number of nodes of the cluster. Take a subsystem S and compute $I(S)$; all $H(X_i)$ are equal to 1 (I observes exactly two equally probable symbols on every node); moreover, $H(S)=H(X)=H(X\backslash S)=1$ because on any subsystem I again observes only two symbols with equal probability. To sum it up:

$$C(S) = \frac{I(S)}{M(S; X - S)} = \frac{N-1}{1} = N - 1 \quad [7]$$

where N is the number of nodes in S.



(a) (b)

Fig. 4. Distribution of maximum of CI for each cluster dimensions for evolved system, where (a) the task is the maximization of the minimal distance between attractors (systems with 70 nodes) and (b) the task is the Density Classification Problem (DCP), a simple counting problem [16] and a paradigmatic example of a problem hardly solvable for decentralized systems (the results regard networks with 21 nodes). Essentially, it requires that a binary dynamical system recognize whether an initial binary string contains more 0s or more 1s, by reaching a fixed attractor composed respectively by only 0s or 1s.

In figure 4 you can indeed observe this kind of behavior (note that only the averages have some meaning, because of no T_c has significant value – so, the few exceptions to the general behavior on the right side of figure 4b can be discarded. More details are available in [15]).

These are just preliminary experiments and we are currently studying cases in which the formation of clusters is indeed expected. Note however that there are data of evolved systems having well defined clusters: indeed, biological evolution is affecting living systems since 3.8 billion years.

In particular we are analyzing the gene regulatory network shaping the developmental process of Arabidopsis thaliana, a system composed by 15 genes and 10 different asymptotical behaviors [17]: our tool was able to group together the three genes core of the system (the first two clusters resulting from T_c ranking): in this case we are identifying clusters having T_c values very significant (see [18] for details).

7 Conclusions

A central question in distributed dynamical system is that of identifying the emerging "things": these may be either static entities or dynamical patterns, or some mixture of the two (neural networks representing an attempt at shaping the emergent properties of a set of models in order to perform some required tasks). In this paper we present a first step towards the development of formalized methods – a research initially started within studies on the brain activities [3] - to identify these mesolevel organizations

(MDSs in the work), which may have a topological as well as a dynamical nature. As examples of application we used time series of simple artificial systems and more complex data coming from Boolean Networks and biological gene regulatory systems (*A.thaliana*). So, the analysis performed by our system is able to identify several interesting mesolevel dynamical structures, and we think it could suggest interesting new ways in dealing with artificial and biological systems.

Acknowledgments. This article has been partially funded by the UE projects "MD – Emergence by Design", Pr.ref. 284625 and "INSITE - The Innovation Society, Sustainability, and ICT" Pr.ref. 271574, under the 7th FWP - FET programme.

References

1. Serra, R., Zanarini, G.: Complex Systems and Cognitive Processes - A Combinatorial Approach. Springer (1990)
2. Haken, H.: Synergetics. Springer, Heidelberg (2004)
3. Tononi, G., McIntosh, A.R., Russell, D.P., Edelman, G.M.: Functional Clustering: Identifying Strongly Interactive Brain Regions in Neuroimaging Data. Neuroimage 7 (1998)
4. Villani, M., Serra, R.: On the dynamical properties of a model of cell differentiation. EURASIP Journal on Bioinformatics and Systems Biology 2013, 4 (2013)
5. Benedettini, S.: Identifying mesolevel dynamical structures ECLT (European Center for Living Technologies) technical report, Venice (2013)
6. Kauffman, S.A.: The Origins of Order. Oxford University Press, Oxford (1993)
7. Kauffman, S.A.: At Home in the Universe. Oxford University Press, Oxford (1995)
8. Serra, R., Villani, M., Semeria, A.: Genetic network models and statistical properties of gene expression data in knock-out experiments. Journal of Theoretical Biology 227, 149–157 (2004)
9. Shmulevich, I., Kauffman, S.A., Aldana, M.: Eukaryotic cells are dynamically ordered or critical but not chaotic. Proc. Natl. Acad. Sci. 102, 13439–13444 (2005)
10. Villani, M., Serra, R., Graudenzi, A., Kauffman, S.A.: Why a simple model of genetic regulatory networks describes the distribution of avalanches in gene expression data. J. Theor. Biol. 249, 449–460 (2007)
11. Serra, R., Villani, M., Barbieri, B., Kauffman, S.A., Colacci, A.: On the dynamics of random boolean networks subject to noise: attractors, ergodic sets and cell types. Journal of Theoretical Biology 265, 185–193 (2010)
12. Villani, M., Barbieri, A., Serra, R.A.: Dynamical Model of Genetic Networks for Cell Differentiation. PLoS ONE 6(3), e17703 (2011), doi:10.1371/journal.pone.0017703
13. Espinosa-Soto, C., Wagner, A.: Specialization Can Drive the Evolution of Modularity. PLoS Comput. Biol. 6(3) (2010)
14. Clune, J., Mouret, J.-B., Lipson, H.: The evolutionary origins of modularity. Proceedings of the Royal Society B 280, 20122863 (2013)
15. Benedettini, S., Villani, M., Roli, A., Serra, R., Manfroni, M., Gagliardi, A., Pinciroli, C., Birattari, M.: Dynamical regimes and learning properties of evolved Boolean networks. Neurocomputing 99, 111–123 (2013)

16. Packard, N.: Adaptation toward the edge of chaos. In: Kelso, J., Mandell, A., Shlesinger, M. (eds.) Dynamic Patterns in Complex Systems. World Scientific, Singapore (1988)
17. Chaos, A., Aldana, M., Espinosa-Soto, C., Ponce de Leon, B.G., Garay Arroyo, A., Alvarez-Buylla, E.R.: From Genes to Flower Patterns and Evolution: Dynamic Models of Gene Regulatory Networks. J. Plant Growth Regul. 25, 278–289 (2006)
18. Villani, M., Filisetti, A., Benedettini, S., Roli, A., Lane, D., Serra, R.: The detection of intermediate-level emergent structures and patterns. In: Proceeding of ECAL 2013, the 12th European Conference on Artificial Life. MIT Press (2013) ISBN: 9780262317092

Experimental Guidelines for Semantic-Based Regularization

Claudio Saccà, Michelangelo Diligenti, and Marco Gori

Dipartimento di Ingegneria dell'Informazione,
Università di Siena, via Roma 54, Siena, Italy
claudiosacc@gmail.com, {diligmic,marco}@dii.unisi.it

Abstract. This paper presents a novel approach for learning with constraints called Semantic-Based Regularization. This paper shows how prior knowledge in form of First Order Logic (FOL) clauses, converted into a set of continuous constraints and integrated into a learning framework, allows to jointly learn from examples and semantic knowledge. A series of experiments on artificial learning tasks and application of text categorization in relational context will be presented to emphasize the benefits given by the introduction of logic rules into the learning process.

1 Introduction

Recent studies in machine learning enlightened the improvements given by the incorporation of a significant amount of prior knowledge into the learning process with a capable of bridging abstract descriptions of the environment with collections of supervised and unsupervised examples. In past few years remarkable approaches to provide a unified treatment of logic and learning were suggested by [3] in which the background knowledge on the problem at hand can be injected into the learning process mainly by encoding it into the kernel function. A related approach to combining first-order logic and probabilistic graphical models in a single representation are Markov Logic Networks [4]. In [2] and successively in [6],[7] it has been proposed a different approach to incorporate logic clauses, that are thought of as abstract and partial representations of the environment and are expected to dictate constraints on the development of an agent which also learns from examples. The approach is based on a framework that integrates kernel machines and logic to solve multi-task learning problems. The kernel machine mathematical apparatus allows casting the learning problem into a primal optimization of a function composed of the loss on the supervised examples, the regularization term, and a penalty term deriving from forcing the constraints converting the logic. This naturally allows to get advantage of unsupervised patterns in the learning task, as the degree of satisfaction of the constraints can be measured on unsupervised data. In particular, constraints are assumed to be expressed in First-Order Logic (FOL). The mathematical apparatus of Semantic Based Regularization (SBR) that converts the external knowledge into a set of real value constraints, which are enforced over the values assumed by

S. Bassis et al. (eds.), *Recent Advances of Neural Network Models and Applications*,
Smart Innovation, Systems and Technologies 26,
DOI: 10.1007/978-3-319-04129-2_2,

the learned classification functions, has been introduced in [2]. We introduced the new concept of binary predicates and given relations to exploit better the high expressivity of FOL rule. Furthermore, we developed a new software implementing SBR. Hence in this paper, we give some guidelines to the use of this software and present some representative benchmark experiments and a text-categorization task to show how we can take advantage of the integration of logic knowledge into the learning process to improve classification performace respect to a plain SVM classifier.

The paper is organized as follows: the next section introduces learning from constraints with kernel machines. The translation of any FOL knowledge into real-valued constraints is described in section 3, but can be examined in depth in [7] and some experimental results are reported in section 5 providing some guidelines in section 4 on how to execute the experiments through the related software for Semantic-Based Regularization (SBRS)[1].

2 Learning with Constraints

Let us consider a multitask learning problem as formulated in [7], where each task works on an input domain where labeled and unlabeled examples are sampled from. Each input pattern is described by a vector of features that are relevant to solve the tasks at hand. Let $\mathcal{D}_k$ and $f_k : \mathcal{D}_k \rightarrow I\!R$ be the input domain and the function implementing task k, respectively. We indicate as $\mathbf{x}_k \in \mathcal{D}_k$ a generic input vector for the k-th task. Task k is implemented by a function f_k, which may be known may be known a priori (*GIVEN* task) or it must be inferred (*LEARN* task). In this latter case it is assumed that each task function lives in an appropriate Reproducing Kernel Hilbert Space $\mathcal{H}_k$. Let us indicate with $\mathcal{T}$ the total number of tasks, of which the first T are assumed to be the learn tasks. For remaining evidence tasks, it will hold that all sampled data is supervised as the output of the task is known in all data points: $\mathcal{S}_k = \mathcal{L}_k$,$\mathcal{U}_k = \emptyset$ (*close-world assumption*).

The learning procedure can be cast as an optimization problem that aims at computing the optimal *LEARN* functions $f_k \in \mathcal{H}_k,\ k = 1, \ldots, T$ where $f_k : \mathcal{D}_k \rightarrow I\!R$. The optimization problem consists of three terms: a data fitting term, penalizing solutions that do not fit the example data, a regularization term, penalizing solutions that are too complex and a constraint term, penalizing solutions that do not respect the constraints:

$$E[f_1, \ldots, f_{\mathcal{T}}] = \sum_{k=1}^{T} \lambda_k^{\tau} \cdot \frac{1}{|\mathcal{L}_k|} \sum_{(\mathbf{x}_k, \mathbf{y}_k) \in \mathcal{L}_k} L_k^e(f_k(\mathbf{x}_k), \mathbf{y}_k) + \sum_{k=1}^{T} \lambda_k^r \cdot ||f_k||^2_{\mathcal{H}_k} + \\ + \sum_{h=1}^{H} \lambda_h^v \cdot L_h^c(\phi_h(f_1, \ldots, f_{\mathcal{T}}))\ . \tag{1}$$

[1] `https://sites.google.com/site/semanticbasedregularization/home/software`

where L_k^e and L_h^c are a loss function that measures the fitting quality respect to the target $\mathbf{y}_k$ for the data fitting term and the constraint degree of satisfaction, respectively. Clearly, if the tasks are uncorrelated, the optimization of the objective function is equivalent to T stand-alone optimization problems for each function.

The optimization of the overall error function is performed in the primal space using gradient descent [1]. The objective function is non-convex due to the constraint term. Hence, in order to face the problems connected with the presence of sub-optimal solutions, the optimization problem was split in two stages. In a first phase, as commonly done by kernel machines it is performed regularized fitting of the supervised examples. Only in a second phase, the constraints are enforced since requiring a higher abstraction level [2],[7]. The constraints can also be gradually introduced. As common practice in constraint satisfaction tasks, more restrictive constraints should be enforced earlier.

3 Translation of First-Order Logic (FOL) Clauses into Real-Valued Constraints

We have focused the attention on knowledge-based descriptions given by first-order logic. Let's consider as example that our knowledge-base (KB) is composed by generic FOL clauses in the following format $\forall v_i \; E(v_i, \mathcal{P})$, where $v_i \in \mathcal{D}_i$ is a variable belonging to the set of the variables $\mathcal{V} = \{v_1, \ldots, v_N\}$ used in the KB, $\mathcal{P}$ is the set of predicates used in the KB, and $E(v_i, \mathcal{P})$ represents the generic propositional (quantifier-free) part of the FOL formula. Without loss of generality, we focused our attention to FOL clauses in the Prenex-Conjunction Normal form [2]. A FOL rule is translated in a continuous form where a predicate $P(x)$ is approximated via a function $f_P(x)$ implemented by a Kernel Machine. The conversion process of a clause into a constraint functional consists of the following three steps:

I. Predicate substitution: substitution of the predicates with their continuous implementation realized by the functions f composed with a squash function, mapping the output values into the interval $[0, 1]$.
II. Conversion of the Propositional Expression: conversion of the quantifier-free expression $E(v_i, \mathcal{P})$ using T-norms, where all atoms are grounded as detailed in [5],[7]
III. Quantifier conversion: conversion of the universal and existential quantifiers as explained in [2],[5],[7].

4 Semantic-Based Regularization Simulator Guidelines

Semantic-Based Regularization Simulator (SBRS) is the software implementing SBR that have used for the experiments presented in the next section. SBRS is able to use some of the most used kernels. Regarding to constraints evaluation, it is possible to use FOL clause with n-ary predicates and to learn or verify the satisfaction of logic constraints. The input is split in four separate ASCII files:

– data definition file: this file contains the patterns available in the dataset and each line represents a pattern. Each line is composed by fields which are delimited by the (;) symbol

patternID0;domainXY;X:0.2,Y:0.3

– predicate definition file: this file contains the definitions of predicates. As explained before, predicates are approximated with a function. They can be defined as *learn* if the correspondent function must be inferred from examples and constraints or *given* if it is known a priori. For *given* predicates can be specified a default value (T,F) for the correspondent function when the example is not provided.

DEF A(domainXY);LEARN
DEF R(domainXY,domainXY);GIVEN;F

– examples files: this files contains the desired output for specific groundings of the predicates (i.e. examples in machine learning context).

A(patternID0)=1
A(patternID1)=0

– FOL rules definition: this file containts the rules that have to be integrated in learning process. Rules are expressed in CNF and they are defined using a specific syntax. For each each rule, it must be specified the norms used to convert the propositional part (*product_ tnorm*, *minimum_ tnorm*) and quantifiers (*L1*,*LINF*) of the logic formula. Rules can be defined as *learn* if they have to be used in the learning process, or *verify* if we want to verify their satisfaction on a different sample of data.

forall p [(NOT A(p) OR NOT B(p) OR C(p))];LEARN;PRODUCT_TNORM;L1
forall p [(NOT C(p) OR A(p))];VERIFY;MINIMUM_TNORM;LINF

Due to a lack of space, more details on SBRS can be found in the manual[2]. A simple tutorial[3] with a few examples is also provided.

5 Experimental Results

For the fisrt part of this section we designed 2 artificial benchmarks to show how it is possible to define logic rules and the benefits of their integration into the learning process. All datasets assume a uniform density distribution and some prior knowledge is available on the classification task, that is expressed by logic clauses. A two-stage learning algorithm as described in [2,7] is exploited in the

[2] https://sites.google.com/site/semanticbasedregularization/SBRS_manual.pdf
[3] https://sites.google.com/site/semanticbasedregularization/SBRS_tutorial.pdf

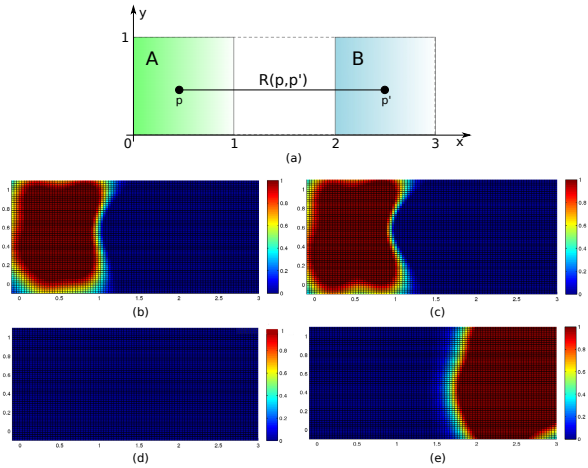


Fig. 1. (a) Input problem definition for benchmark 1 (b) Activation map of class A when not using constraints (c) Activation map of class A when using constraints (d) Activation map of class B when not using constraints (e) Activation map of class B when using constraints

experiments. All presented results are an average over multiple runs performed over different samples of the training and test sets. We considered learning tasks with examples drawn in $\mathbb{R}^2$ that allow us to plot the activation maps and assess the performance in a low dimensional space. Moreover, all the experiments have been designed so that we have small set of supervised examples and a great amount of unsupervised one since our approach it is expected to take advantage of unsupervised patterns to learn from the constraints. Therefore, we will present an application of text categorization on CiteSeer dataset where relational context permits to define different logic rules.

Benchmark 1: Universal and Existential Quantifiers. In this benchmark, it is supposed to check the effects of both universal and existential quantifier in a generic rule. In this benchmark it is also introduced the notion of *GIVEN* predicate and the difference between learn and given functions. The dataset is composed by patterns belonging to two classes: A and B. Dataset has been generated so that it is consistent with the following FOL rule:

$$\forall p \; A(p) \Rightarrow \exists p' \; B(p') \wedge R(p, p')$$

where $R(x, y)$ is *GIVEN* predicate. This means that, considering $p \equiv (x, y)$, its value is known for each groundings of its variables according to the following definition: $R(p, p') = 1$ if $\; | \, \|x' - x\| - 2 \, | \leq 0.01 \, , \; \|y' - y\| \leq 0.01$, otherwise $R = 0$. Patterns are distributed uniformly over $\{(x, y) : x \in [0, 3], y \in [0, 1]\}$, but given a generic grounding for variable p, we have that $A(p) = 1$ iff $\;\; p \in \{p : 0 \leq x \leq 1 \, , \, 0 \leq y \leq 1\}$, while $B(p) = 1$ iff $\;\; p \in \{p : 2 \leq x \leq 3 \, , \, 2 \leq y \leq 3\}$. In figure

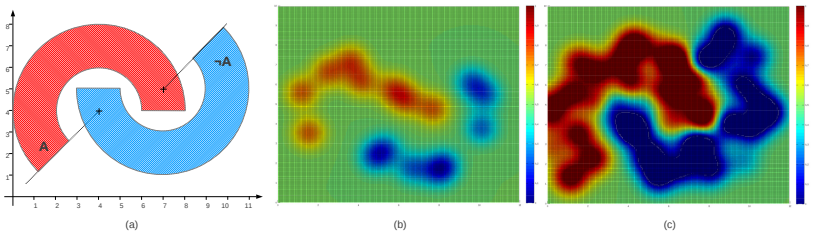


Fig. 2. (a) Input problem definition for benchmark 2 (b) Activation map of class A when not using constraints (c) Activation map of class A when using constraints

1 (a) it is shown the input problem definition. In addition, we provide some declaration of the examples only for the predicate $A(p)$. Also in this benchmark, the goal is to learn the corresponding functions associated to both predicate $A(p)$ and $B(p)$ and to compare with the results of a plain SVM without constraints, to show the benefits of integrating logic knowledge into the learning process. In particular, since we don't have examples for class B, we want to learn it only through the constraint. The parameters λ_l, λ_r and λ_c have been set to 1, 0.1 and 5, respectively. We exploited a Gaussian kernel with variance equal to 0.5. After a previous phase of cross-validation, we decided to use LINF/P-GAUSS norm to translate the FOL rule into a real-valued constraint.

As we expected, when no supervised examples for B are provided, in figure 1 (c) the activation area is all set to zero because without example it is not possible to infer a function. On the other hand, SBR can benefits of the logic knowledge to infer the activation map for class B with a good approximation.

Benchmark 2: Generalization of Manifold Regularization - The Two Moons. In this experiment, it will be shown how manifold regularization can be considered a special case of SBR. It will be assumed that for each couple of patterns in the dataset the information about their neighbourhood is provided through a *GIVEN* predicate $N(p, p')$ that is true if the patterns are near ($\|p - p'\| \leq 0.2$) to each other or false otherwise. The dataset is composed by patterns distributed uniformly over the two moons shown in figure 2 (a) and could belong or not to class A. The assumption that two points connected by an edge in the manifold must belong to the same class could be translated in logic by the following rule:

$$\forall p \; \forall p' \; N(p, p') \Rightarrow A(p) \Leftrightarrow A(p').$$

This logic formula, can be seen as the logic equivalent of manifold assumption, because predicate $N(p, p')$ holds true if and only if p, p' are connected on the manifold built using the relation of neighbourhood. Dataset has been generated so that, given a generic grounding for variable p, we have $A(p) = 1$ iff p belongs to the red moon, while $A(p) = 0$ iff p belongs to the blue moon. In addition, we provide very few supervised examples for the class A while a great amount of patterns remain unsupervised. The classification task consists to learn the

function associated to predicate $A(p)$ when integrating the logic rule into the learning process. The parameters λ_l, λ_r and λ_c have been set to 1, 0.1 and 1, respectively. A Gaussian kernel, with variance equal to 0.8, has been exploited. Results in figures 2 (b)(c) show that, when adding the FOL rule, SBR can infer the activation map for class A with a better approximation respect to the case when no rules are used to train the classifier.

Using a test-set to evaluate the classification performance of the learned functions, we can se that our approach improves consistently the F1-score from 0.806 $\pm$ 0.009 when no constraints are used to 0.982 $\pm$ 0.014 when we force the satisfaction of the FOL rule, defined before, during the learning process.

CiteSeer: Text Categorization in Relational Context. The CiteSeer dataset[4] consists of 3312 scientific publications classified into at least one of six classes. The citation network consists of 4732 links. Each publication in the dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. Five folds have been generated by selecting randomly 15% and 35% of the papers for validation and test set respectively. For the remaining 50% of the training set, n% (n=5,10,25,50) of the papers where selected randomly keeping the supervisions. The others remain unsupervised. The knowledge base collects different collateral information which is available on the dataset. CiteSeer makes available a list of citations for each papers. Our algorithm can exploit these relations assuming that a citation represents a common intent between the papers that are therefore suggested to belong to the same set of categories. This can be expressed via a set of 6 clauses (one per category) such that foreach $i = 1, \ldots, 6$:

$$\forall x \in \mathcal{P} \; \forall y \in \mathcal{P} \quad Link(x, y) \Rightarrow (C_i(x) \wedge C_i(y)) \vee (\neg C_i(x) \wedge \neg C_i(y))$$

where $\mathcal{P}$ is the domain of all papers in the dataset and $Link(x, y)$ is a binary predicate which holds true iff paper x cites paper y. In the dataset we know that all papers belonging to *ML* class have been also tagged as *AI*. This information can be exploited through the following rule: $\forall\, x \in \mathcal{P} \;\; ML(x) \Rightarrow AI(x)$. Furthermore, the following rule defines a close-world assumption $\forall\, x \in \mathcal{P}\; C_1(x) \vee \ldots \vee C_6(x)$, where $C_1 \ldots C_6$ are the six categories of this problem. Finally using the supervised examples available in training, we add a prior to each class adding for each category this rule:

$$\exists_n x \;\; C_i(x) \wedge \exists_m \; x \;\; \neg C_i(x) \quad : \quad n + m = N$$

where n and consequently m are choosen basing on the number of supervised examples in training set for that class. For each subsample size of the training set, one classifier has been trained. As a comparison, we also trained for each set a standard SVM (using only the supervised labels), a Transductive SVM (implemented in the svmlight software package). The validation set has been used to select the best values for λ_r and λ_c. The F1-score has been compute as

[4] Available at: `http://linqs.cs.umd.edu/projects//projects/lbc/index.html`

Table 1. Micro F1 metrics averaged over 5 runs using SVM, Transductive SVM (TSVM) and Semantic Based Regularization (SBR)

	5%	10%	25%	50%
SVM	0.237 ±0.015	0.442 ±0.023	0.589 ±0.016	0.644 ±0.008
TSVM	0.604 ±0.023	0.623 ±0.021	0.631 ±0.02	0.655 ±0.007
SBR	0.637 ±0.019	0.656 ±0.02	0.661 ±0.022	0.679 ±0.013

an average over five fold. Table 1 summarizes the results for a different number of supervised data. SBR provides a statistically significant F1 gain with the respect to a standard SVM that do not exploit logic knowledge and it improves in average the classification performance of a trasductive SVM.

6 Conclusions

In this paper we give some insights on how to integrate prior-knowledge in form of logic clause into the general framework of regularization with kernel machines. This apparatus makes it possible to use a semi-supervised scheme in which the unsupervised examples, often abundant, play a crucial role in the approximation of the penalty term associated with the logic constraints. These preliminary experiments suggest the possibility to exploit a new class of semantic-based regularization machines in which the introduction of prior knowledge takes into account constraints on the tasks. The general principles at the base of this approach can be applied to several fields like bioinformatics for prediction of proteins interactions that we are going to explore.

Acknowledgements. This research was partially supported by the research grant PRIN2009 "Learning Techniques in Relational Domains and Their Applications" (2009LNP494) from the Italian MURST.

References

1. Chapelle, O.: Training a support vector machine in the primal. Neural Computation 19(5), 1155–1178 (2007)
2. Diligenti, M., Gori, M., Maggini, M., Rigutini, L.: Bridging logic and kernel machines. Machine Learning, 1–32 (2011)
3. Frasconi, P., Passerini, A.: Learning with kernels and logical representations. In: De Raedt, L., Frasconi, P., Kersting, K., Muggleton, S.H. (eds.) Probabilistic ILP 2007. LNCS (LNAI), vol. 4911, pp. 56–91. Springer, Heidelberg (2008)
4. Richardson, M., Domingos, P.: Markov logic networks. Machine Learning 62(1-2), 107–136 (2006)

5. Saccà, C., Diligenti, M., Maggini, M., Gori, M.: Integrating logic knowledge into graph regularization: an application to image tagging. In: Ninth Workshop on Mining and Learning with Graphs - MLG (KDD) (2011)
6. Saccà, C., Diligenti, M., Maggini, M., Gori, M.: Learning to tag from logic constraints in hyperlinked environments. In: ICMLA, pp. 251–256 (2011)
7. Saccà, C., Frandina, S., Diligenti, M., Gori, M.: Constrained-based learning for text categorization. In: Workshop on COmbining COnstraint solving with MIning and LEarning - CoCoMiLe (ECAI) (2012)

A Preliminary Study on Transductive Extreme Learning Machines

Simone Scardapane, Danilo Comminiello, Michele Scarpiniti, and Aurelio Uncini

Department of Information Engineering, Electronics and Telecommunications (DIET),
"Sapienza" University of Rome,
via Eudossiana 18, 00184, Rome
{simone.scardapane,danilo.comminiello,
michele.scarpiniti}@uniroma1.it, aurel@ieee.org

Abstract. Transductive learning is the problem of designing learning machines that succesfully generalize only on a given set of input patterns. In this paper we begin the study towards the extension of Extreme Learning Machine (ELM) theory to the transductive setting, focusing on the binary classification case. To this end, we analyze previous work on Transductive Support Vector Machines (TSVM) learning, and introduce the Transductive ELM (TELM) model. Contrary to TSVM, we show that the optimization of TELM results in a purely combinatorial search over the unknown labels. Some preliminary results on an artifical dataset show substained improvements with respect to a standard ELM model.

Keywords: Transductive learning, extreme learning machine, semi-supervised learning.

1 Introduction

In the classical Machine Learning setting [1], starting from a limited set of data sampled from an unknown stochastic process, the goal is to infer a general predictive rule for the overall system. Vapnik [2] was the first to argue that in some situations, this target may be unnecessarily complex with respect to the actual requirements. In particular, if we are interested on predictions limited to a given set of input patterns, then a learning system tuned to this specific set should outperform a general predictive one. In Vapnik words, the advice is that, "*when solving a problem of interest, do not solve a more general problem as an intermediate step*" [2]. Vapnik also coined a term for this setting, which he called *Transductive Learning* (TL).

In [2] he studied extensively the theoretical properties of TL, and his insights led him to propose an extension to the standard Support Vector Machine (SVM) algorithm, namely the Tranductive SVM (TSVM). While SVM learning results in a quadratic optimization problem, TSVM learning is partly combinatorial, making it a difficult non-convex optimization procedure. However, a number of interesting algorithms have been proposed for its efficient solution. The interested reader can find a comprehensive review of them in Chapelle et al. [3].

By drawing theoretical and practical ideas from TSVMs, in this paper we extend *Extreme Learning Machine* (ELM) theory [4] to the transductive setting. ELM models

S. Bassis et al. (eds.), *Recent Advances of Neural Network Models and Applications*,
Smart Innovation, Systems and Technologies 26,
DOI: 10.1007/978-3-319-04129-2_3,

have gained some attention as a conceptual unifying framework for several families of learning algorithms, and possess interesting properties of speed and efficiency. An ELM is a two-layer feed-forward network, where the input is initially projected to an highly dimensional feature space, on which a linear model is subsequently applied. Differently from other algorithms, the feature space is fully fixed before observing the data, thus learning is equivalent to finding the optimal output weights for our data. We show that, in the binary classification case, Transductive ELM (TELM) learning results in a purely combinatorial search over a set of binary variables, thus it can be solved more efficiently with respect to TSVM. In this preliminary work we use a simple Genetic Algorithm (GA) [5] as a global optimizer and test the resulting algorithm on an artificial dataset. Results show promising increase in performance for different sizes of the datasets.

Transductive learning has been throughly studied lately due to the interest in Semi-Supervised Learning (SSL) [6]. In SSL, additional unlabelled data is provided to the algorithm (as in TL), but the goal is to infer a general predictive rule as in classical inductive learning. In this respect, unlabelled data is seen as additional information that the algorithm can use to deduce general properties about the geometry of input patterns. Despite TL and SSL have different objectives, their inner workings are in some respects similar, and many TL and SSL algorithms can be used interchangeably in the two situations. In particular, TSVMs are known as Semi-Supervised SVM (S3VM) [3] in the SSL community. Hence, our work on TELM may be of interest as a first step towards the use of ELM models in a SSL setting.

The rest of this paper is organized as follows: in Section 2 we introduce some basic concepts on TL, and detail the TSVM optimization procedure. Section 3 summarizes the main theory of ELM. Section 4, the main contribution of this work, extends ELM theory using concepts from Section 2. Section 5 shows some preliminary results on an artificial dataset. Although we provide a working algorithm, two fundamental questions remain open, and we confront with them in Section 6. Finally, we make some final remarks in Section 7.

2 Transductive Learning

2.1 Inductive Learning and Support Vector Machines

Consider an unknown stochastic process described by the joint probability function $p(\mathbf{x},y) = p(\mathbf{x})p(y|\mathbf{x}), \mathbf{x} \in X, y \in Y$, where X and Y are known as the *input* and *output* spaces respectively. In this work we restrict ourselves to the binary classification case, i.e., $Y = \{-1,+1\}$. Given a loss function $L(\mathbf{x},y,\hat{y}) : X \times Y \times Y \to \mathbb{R}$ that measures the loss we incur by estimating $\hat{y} = f(\mathbf{x})$ instead of the true y, and a set of possible models H, the goal of inductive learning is to find a function that minimizes the *expected risk*:

$$I[f] = \int_{X \times Y} L(\mathbf{x},y,f(\mathbf{x}))p(\mathbf{x},y)d\mathbf{x}dy \tag{1}$$

We are given only a limited dataset of N samplings from the process $S = (\mathbf{x}_i, y_i)_{i=1}^N$, that we call the *training set*. The *empirical risk* is defined as:

$$I_{emp}[f;S] = \frac{1}{N}\sum_{i=1}^{N} L(\mathbf{x}_i, y_i, f(\mathbf{x}_i)) \tag{2}$$

Vapnik [2] derived several bounds, known as *VC bounds*, on the relation between (1) and (2) for limited datasets. All bounds are in the following form and are valid with probability $1-\eta$:

$$I[f] \leq I_{emp}[f;S] + \Phi(h,N,\eta) \tag{3}$$

where h is the *VC-dimension* of the set H, and $\Phi(h,N,\eta)$ is known as a *capacity* term. In general, such term is directly proportional to h. Thus, for two functions $f_1, f_2 \in H$ with the same error on the dataset, the one with lower VC-dimension is preferable. Practically, this observation can be implemented in the Support Vector Machine (SVM) algorithm, as we describe below.

Consider a generic *Reproducing Kernel Hilbert Space* H as set of models. There is a direct relationship [7] between h and the inverse of $\|f\|_H$, $f \in H$, where $\|f\|_H$ is the norm of f in H. Thus, the optimal function is the one minimizing the error on the dataset and of minimum norm. When using the *hinge loss* $L(\mathbf{x}_i, y_i, f(\mathbf{x})) = \max\{0, 1-y_i f(\mathbf{x}_i)\}$ as loss function, we obtain the SVM for classification [8]. It can be shown that learning corresponds to a quadratic optimization problem:

$$\begin{aligned} \underset{f}{\text{minimize}} \quad & \frac{1}{2}\|f\|_H^2 + C_s \sum_{i=1}^{N} \zeta_i \\ \text{subject to} \quad & y_i f(\mathbf{x}_i) \geq 1-\zeta_i,\ \zeta_i \geq 0,\ i = 1,\ldots,N. \end{aligned} \tag{4}$$

where ζ_i are a set of *slack variables* that measures the error between predicted and desired output and C_s is a regularization parameter set by the user. Solution to (4) is of the form $f(\mathbf{x}) = \sum_{i=1}^{N} a_i k(\mathbf{x}, \mathbf{x}_i)$, where $k(\cdot,\cdot)$ is the *reproducing kernel* associated to H.

2.2 Transductive Learning and Transductive SVM

In *Transductive learning* (TL) we are given an additional set[1] $U = (\mathbf{x}_i)_{i=N+1}^{N+M}$, called the *testing set*, and we aim at minimizing $I_{emp}[f;U]$. An extension of the theory described above [2] leads to minimizing the error on both S and U.

By denoting with $\mathbf{y}^* = [y^*_{N+1}, \ldots, y^*_{N+M}]^T$ a possible labelling of the elements in U, this results in the following (partly combinatorial) optimization problem, known as the *Transductive SVM* (TSVM):

$$\begin{aligned} \underset{f,\mathbf{y}^*}{\text{minimize}} \quad & \frac{1}{2}\|f\|_H^2 + C_s \sum_{i=1}^{N} \zeta_i + C_u \sum_{i=N+1}^{N+M} \zeta_i \\ \text{subject to} \quad & y_i f(\mathbf{x}_i) \geq 1-\zeta_i,\ \zeta_i \geq 0,\ i = 1,\ldots,N. \\ & y^*_i f(\mathbf{x}_i) \geq 1-\zeta_i,\ \zeta_i \geq 0,\ i = N+1,\ldots,N+M \end{aligned} \tag{5}$$

[1] Note the peculiar numbering on the dataset.

where we introduce an additional regularization term C_u. In particular, equation (5) is combinatorial over $\mathbf{y}^*$, since each label is constrained to be binary. This makes the overall problem highly non-convex and difficult to optimize in general. Some of the algorithms designed to efficiently solve it are presented in [3].

Typically, we also try to enforce an additional constraint on the proportion of labellings over $\mathbf{U}$, of the form:

$$\rho = \frac{1}{M}\sum_{i=1}^{M} y_i^*$$

where ρ is set *a-priori* by the user. This avoids unbalanced solutions in which all patterns are assigned to the same class.

3 Extreme Learning Machine

An Extreme Learning Machine (ELM) [9,4] is a linear combination of an L-dimensional feature mapping of the original input:

$$f(\mathbf{x}) = \sum_{i=1}^{L} h_i(\mathbf{x})\beta_i = \mathbf{h}(\mathbf{x})^T\beta \tag{6}$$

where $\mathbf{h}(\mathbf{x}) = [h_1(\mathbf{x}),\ldots,h_L(\mathbf{x})]^T$ is called the *ELM feature vector* and β is the vector of expansion coefficients. The feature mapping is considered fixed, so the problem is that of estimating the optimal β. Starting from a known function $g(\mathbf{x},\theta)$, where θ is a vector of parameters, it is possible to obtain an ELM feature mapping by drawing parameters θ at random from an uniform probability distribution, and repeating the operation L times. Huang et al. [4] showed that almost any non-linear function can be used in this way, and the resulting network will continue to be an universal approximator. Moreover, they proposed the following regularized optimization problem, where we aim at finding the weight vector that minimizes the error on S and is of minimum norm:

$$\begin{aligned} \underset{\beta}{\text{minimize}} \quad & \frac{1}{2}\|\beta\|_2^2 + \frac{C_s}{2}\sum_{i=1}^{N}\zeta_i^2 \\ \text{subject to} \quad & \mathbf{h}^T(\mathbf{x}_i)\beta = y_i - \zeta_i,\ i = 1,\ldots,N. \end{aligned} \tag{7}$$

As for SVM, C_s is a regularization parameter that can be adjusted by the user, and $\zeta_i, i = 1,\ldots,N$ measure the error between desired and predicted output. The problem is similar to (4), but has a solution in closed form. In particular, a possible solution to (7) is given by [4]:

$$\beta = \mathbf{H}^T\left(\frac{1}{C_s}\mathbf{I}_{N\times N} + \mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{y} \tag{8}$$

where $\mathbf{I}_{N\times N}$ is the $N \times N$ identity matrix, and we defined the hidden matrix $\mathbf{H} = [\mathbf{h}(\mathbf{x}_1),\ldots,\mathbf{h}(\mathbf{x}_N)]$ and the output vector $\mathbf{y} = [y_1,\ldots,y_N]^T$. When using ELM for classification, a decision function can be easily computed as:

$$f'(\mathbf{x}) = sign(f(\mathbf{x}))$$

4 Transductive ELM

Remember that in the TL setting we are given an additional dataset $U = (\mathbf{x}_i)_{i=N+1}^{N+M}$ over which we desire to minimize the error. To this end, similarly to the case of TSVM, we consider the following modified optimization problem:

$$\begin{aligned} \underset{\beta, y^*}{\text{minimize}} \quad & \frac{1}{2}\|\beta\|_2^2 + \frac{C_s}{2}\sum_{i=1}^{N}\zeta_i^2 + \frac{C_u}{2}\sum_{i=N+1}^{N+M}\zeta_i^2 \\ \text{subject to} \quad & \mathbf{h}^T(x_i)\beta = y_i - \zeta_i,\ i = 1,\ldots,N. \\ & \mathbf{h}^T(x_i)\beta = y_i^* - \zeta_i,\ i = N+1,\ldots,N+M. \end{aligned} \tag{9}$$

We call (9) the *Transductive ELM* (TELM). At first sight, this may seems partly combinatorial as in the case of TSVM. However, for any possible choice of the labelling $\mathbf{y}^*$, the optimal β is given by (8), or more precisely, by a slightly modified version to take into account different parameters for C_s and C_u:

$$\beta = \mathbf{H}^T(\mathbf{C}^{-1}\mathbf{I} + \mathbf{H}\mathbf{H}^T)^{-1}\begin{bmatrix}\mathbf{y} \\ \mathbf{y}^*\end{bmatrix} \tag{10}$$

Where $\mathbf{C}$ is a diagonal matrix with the first N elements equal to C_s and the last M elements equal to C_u, and the hidden matrix is computed over all $N+M$ input patterns:

$$\mathbf{H} = [\mathbf{h}(\mathbf{x}_1),\ldots,\mathbf{h}(\mathbf{x}_N),\mathbf{h}(\mathbf{x}_{N+1}),\ldots,\mathbf{h}(\mathbf{x}_{N+M})]$$

Back-substituting (10) into (9), we obtain a fully combinatorial search problem over $\mathbf{y}^*$. This can be further simplified by considering:

$$\hat{\mathbf{H}} = \mathbf{H}^T(\mathbf{C}^{-1}\mathbf{I} + \mathbf{H}\mathbf{H}^T)^{-1} = \begin{bmatrix}\hat{\mathbf{H}}_1 & \hat{\mathbf{H}}_2\end{bmatrix} \tag{11}$$

Where $\hat{\mathbf{H}}_1$ is the submatrix containing the first N columns of $\hat{\mathbf{H}}$, and the other block follow. Equation (10) can be rewritten as:

$$\beta = \hat{\mathbf{H}}_1\mathbf{y} + \hat{\mathbf{H}}_2\mathbf{y}^* \tag{12}$$

Where the vector $\hat{\mathbf{H}}_1\mathbf{y}$ and the matrix $\hat{\mathbf{H}}_2$ are fixed for any choice of the labeling of U. Any known algorithm for combinatorial optimization [5] can be used to train a TELM model, and form (12) is particularly convenient for computations. We do not try to enforce a specific proportion of positive labels (although this would be relatively easy) since in our experiments the additional constraint never improved performance.

5 Results

The TELM algorithm was tested on an artificial dataset known in literature as *the two moons*, a sample of which is shown in Fig. 1. Two points, one for each class, are shown in red and blue respectively. All simulations were performed by MATLAB 2012a, on an Intel i3 3.07 GHz processor at 64 bit, with 4 GB of RAM available, and each result is averaged over 100 runs. The TELM is solved using a standard *Genetic*

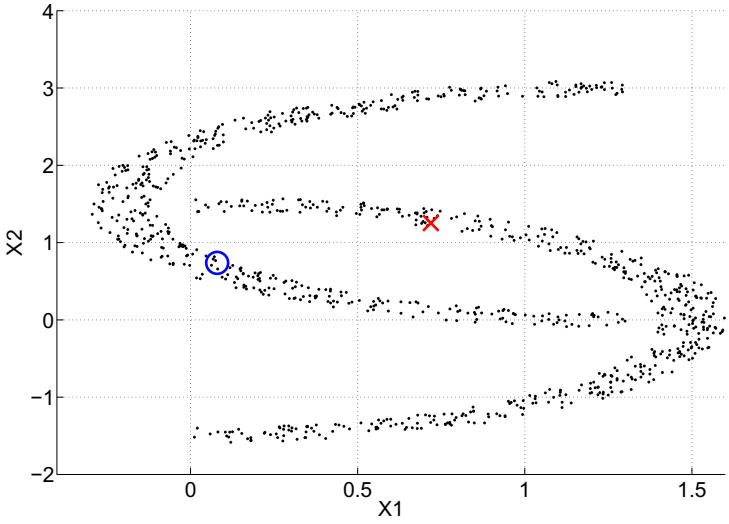


Fig. 1. Sample of the dataset

Algorithm [5]. For comparison, we implemented as baseline a standard ELM model and a binary SVM.

Sigmoid additive activation functions are used to construct the ELM feature space:

$$g(\mathbf{x}) = \frac{1}{1 + e^{-(\mathbf{a}\mathbf{x}+b)}} \tag{13}$$

Using standard default choices for the parameters, we consider 40 hidden nodes, and set $C = 1$. Parameters $\mathbf{a}$ and b of equation (13) were generated according to an uniform probability distribution. The SVM uses the Gaussian kernel:

$$k(\mathbf{x}, \mathbf{y}) = \exp\{-\gamma \|\mathbf{x} - \mathbf{y}\|_2^2\} \tag{14}$$

Parameter γ in (14) was also set to 1 in all the experiments. Algorithms were tested using five different sizes of the datasets. For the first four experiments, a total of 100 samples was considered, and the training size was gradually increased. In the last experiment, instead, we considered two datasets of 100 elements each. For each method we present the classification accuracy in Table 1, where the highest accuracy in each row is highlighted in boldface.

As can be seen, TELM outperforms both methods for every combination we considered. In particular, it gives a small improvement when trained using very small training datasets (first two rows), very large increments with datasets of medium size (third and fourth row), and is able to reach 100% classification accuracy with sufficient samples (fifth row).

Table 1. Experimental results: classification accuracy

	SVM	ELM	TELM
$N = 4$, $M = 98$	0.77	0.75	**0.79**
$N = 10$, $M = 90$	0.81	0.75	**0.86**
$N = 40$, $M = 60$	0.85	0.80	**0.93**
$N = 60$, $M = 40$	0.85	0.81	**0.97**
$N = 100$, $M = 100$	0.93	0.95	**1**

6 Open Questions

Two main questions remain to be answered for an effective implementation of the TELM algorithm. We detail them briefly in this Section.

1. Our formulation suffers from a major drawback which is encountered also on TSVMs. In particular, it cannot be easily extended to the regression case. It is easy to show that any minimizer β of the first two terms of equation (10) automatically minimizes the third with the trivial choice $y_i^* = h(\mathbf{x}_i)^T \beta$. Thus, some modifications are needed, for example following [10].
2. The genetic algorithm imposes a strong computational effort in minimizing (10). This can be addressed by developing specialized solvers able to take into consideration the specific nature of the problem. To this end, we imagine that many of the algorithms used for TSVMs can be readily extended to our context.

7 Conclusions

In this work we presented an initial study for the extension of ELM theory to the transductive learning framework. We showed that this results in a fully combinatorial optimization problem. In our experiments, we solved it using a standard GA. Results are highly promising in the dataset we considered. However, there is the need of further optimizing the learning algorithm before a successful real-world application.

References

1. Cherkassky, V., Mulier, F.: Learning from data: concepts, theory, and methods (2007)
2. Vapnik, V.: The nature of statistical learning theory, 2nd edn., vol. 8. Springer (January 1999)
3. Chapelle, O., Sindhwani, V., Keerthi, S.: Optimization techniques for semi-supervised support vector machines. Journal of Machine Learning Research 9, 203–233 (2008)
4. Huang, G.B., Zhou, H., Ding, X., Zhang, R.: Extreme learning machine for regression and multiclass classification. IEEE Transactions on Systems, Man, and Cybernetics 42(2), 513–529 (2012)
5. Luke, S.: Essentials of metaheuristics (2009)

6. Chapelle, O., Schölkopf, B., Zien, A.: Semi-supervised learning (2006)
7. Evgeniou, T., Pontil, M., Poggio, T.: Regularization networks and support vector machines. Advances in Computational Mathematics 13, 1–50 (2000)
8. Steinwart, I., Christmann, A.: Support vector machines, 1st edn. (2008)
9. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: Theory and applications. Neurocomputing 70(1-3), 489–501 (2006)
10. Cortes, C., Mohri, M.: On transductive regression. In: Advances in Neural Information Processing Systems (2007)

Avoiding the Cluster Hypothesis in SV Classification of Partially Labeled Data

Dario Malchiodi[1] and Tommaso Legnani[2]

[1] Dipartimento di Informatica, Università degli Studi di Milano
malchiodi@di.unimi.it
[2] Dipartimento di Matematica "F. Enriques", Università degli Studi di Milano

Abstract. We propose a Support Vector-based methodology for learning classifiers from partially labeled data. Its novelty stands in a formulation not based on the *cluster hypothesis*, stating that learning algorithms should search among classifiers whose decision surface is far from the unlabeled points. On the contrary, we assume such points as specimens of uncertain labels which should lay in a region containing the decision surface. The proposed approach is tested against synthetic data sets and subsequently applied to well-known benchmarks, attaining better or at least comparable performance w.r.t. methods described in the literature.

1 Introduction

The problem of classification consists in assigning objects in a given domain to one among a prefixed set of classes. In its simplest version, this problem is solved in the machine learning context through *learning* a classifier (that is, a mapping from objects to classes) on the basis of a *labeled sample* consisting of pairs (point, class). Such a problem admits several variations, and this paper focuses on a special setting characterized by the presence in the sample of points not associated to any specific class. This happens for instance in many real world situations where collecting objects is extremely easy (e.g., when mining the Internet) but labeling them is expensive (tipycally because some sort of human intervention is required). Such cases are dealt within the field of semi supervised learning [1] taking into account the so-called *cluster hypothesis*, stating that unlabeled points should not be close to the decision surfaces of the learnt classifiers. In this paper, instead, we require that unlabeled points be confined in a region of the space containing the decision function of the learnt classifier. Indeed, in some situations unlabeled points are characterized by some inherent form of uncertainty, rather than on the difficulty of labeling them. Web spam detection is a typical example of such a situation: in many cases, even humans reading the text contained in a web page are not able to definitely classify it as spam or non-spam, or experts produce different classifications on a same page [2].

The paper is organized as follows: Sect. 2 describes the proposed method, while Sect. 3 applies it to artificial and real-world data sets. Finally, Sect. 4 is devoted to outlooks and concluding remarks.

S. Bassis et al. (eds.), *Recent Advances of Neural Network Models and Applications*, Smart Innovation, Systems and Technologies 26,
DOI: 10.1007/978-3-319-04129-2_4,