We are IntechOpen,
the world's leading publisher of
Open Access books
Built by scientists, for scientists

6,900
Open access books available

186,000
International authors and editors

200M
Downloads

154
Countries delivered to

Our authors are among the

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS

BOOK CITATION INDEX

INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

Automatically Building Diagnostic Bayesian Networks from On-line Data Sources and the SMILE Web-based Interface

Anucha Tungkasthan, Nipat Jongsawat, Pittaya Poompuang,
Sarayut Intarasema and Wichian Premchaiswadi
Graduate School of Information Technology in Business, Siam University
Thailand

1. Introduction

One of the most difficult obstacles in the practical application of probabilistic methods is the effort that is required for model building and, in particular, for quantifying graphical models with numerical probabilities. The construction of Bayesian Networks (BNs) with the help of human experts is a difficult and time consuming task, which is prone to errors and omissions especially when the problems are very complicated or there are numerous variables involved. Learning the structure of a BN model and causal relations from a dataset or database is important for extensive BNs analysis.

In general, the causal structure and the numerical parameters of a BN can be obtained using two distinct approaches. First, they can be obtained from an expert. Second, they can also be learned from a data set. The main drawback of the first approach is that sometimes there is not enough causal knowledge to establish the structure of the network model with certainty and estimation of probabilities required for a typical application is a time-consuming task because of the number of parameters required (typically hundreds or even thousands of values). Thus, the second approach can initially help human experts build a BN model and they can make it applicable at a later time. In practice, some combination of these two approaches is typically used. This paper essentially focuses on using the second approach.

This paper presents a practical framework for automating the building of diagnostic BN models from data sources obtained from the WWW and demonstrates the use of a SMILE web-based interface to represent them. This work proposes the following components: 1) an RSS agent that automatically gathers RSS feeds from diverse data sources in the WWW environment, 2) a transformation/conversion tool that transforms and converts the collected data for both continuous and discrete valued data sets 3) a reasoning engine that has the ability to learn and build the causal structure for BN models from data and provide functionality to perform a diagnosis, 4) the visualization of BN models on a website, and 5) a diagnosis of the BN model and the resulting reports. This article is organized as follows: Section 2 presents a little more detail about the basic concepts of Bayesian networks and tools. Section 3 addresses related work. Section 4 describes the design and implementation of a practical framework for automating the building of diagnostic BN models from online

data sources in more detail. Section 5 presents a conclusion and discusses some perspectives and ideas for future work. An acknowledgement is described in section 6.

2. Fundamentals

This section is intended to describe the fundamentals of Bayesian networks and the core reasoning engines of SMILE web-based interface development. They are described in the following sections.

2.1 Bayesian network and Bayesian updating

Bayesian networks (also called belief networks, Bayesian belief networks, causal probabilistic networks, or causal networks) (Pearl, 1988) are acyclic directed graphs in which nodes represent random variables and arcs represent direct probabilistic dependencies among them. The structure of a Bayesian network is a graphical, qualitative illustration of the interactions among the set of variables that it models. The structure of the directed graph can mimic the causal structure of the modeled domain, although this is not necessary. When the structure is causal, it provides a useful, modular insight into the interactions among the variables and allows for a prediction of the effects of external manipulation. Nodes of a Bayesian network are usually drawn as circles or ovals. A Bayesian network also represents the quantitative relationships among the modeled variables. Numerically, it represents the joint probability distribution among them. This distribution is described efficiently by exploring the probabilistic independence among the modeled variables. Each node is described by a probability distribution conditional on its direct predecessors. Nodes with no predecessors are described by prior probability distributions. Both the structure and the numerical parameters of a Bayesian network can be elicited from an expert. They can also be derived from data, as the structure of a Bayesian network is simply a representation of interdependencies in the data and the numbers are a representation of the joint probability distributions that can be inferred from the data. Finally, both the structure and the numerical probabilities can be a mixture of expert knowledge, measurements and objective frequency data.

Bayesian updating, also referred to as belief updating, or somewhat less precisely as probabilistic inference is based on the numerical parameters captured in the model (Cooper, 1990). The structure of the model which is an explicit statement of the interdependencies in the domain helps in making the algorithms for Bayesian updating more efficient (Dagum & Luby, 1997). All algorithms for Bayesian updating are based on a theorem proposed by Rev. Thomas Bayes (1702-1761) and are known as Bayes Theorem. Belief updating in Bayesian networks is computationally complex. In the worst case, belief updating algorithms are NP-hard (Cooper 1990). There exist several efficient algorithms, however, that make belief updating in graphs consisting of tens or hundreds of variables tractable. Pearl developed a message-passing scheme that updates the probability distributions for each node in a Bayesian network in response to observations of one or more variables (Pearl, 1986). Lauritzen and Spiegelhalter, Jensen et al, and Dawid proposed an efficient algorithm that first transforms a Bayesian network into a tree where each node in the tree corresponds to a subset of variables in the original graph (Lauritzen & Spiegelhalter, 1988: Jensen et al., 1990: Dawid, 1992). The algorithm then exploits several mathematical properties of this tree to perform probabilistic inference. Several approximate algorithms based on stochastic

sampling have been developed. Of these, best known are probabilistic logic sampling (Henrion, 1988), likelihood sampling (Shachter & Peot, 1989: Fung & Chang, 1989), and backward sampling (Fung & del Favero, 1994), Adaptive Importance Sampling (AISBN) (Cheng & Druzdzel, 2000), and Approximate Posterior Importance Sampling (APIS-BN) (Yuan & Druzdzel, 2003). Approximate belief updating in Bayesian networks has also been shown to be worst case NP-hard (Dagum & Luby, 1993).

2.2 The core reasoning engines

The core reasoning engines of the web-based interface development capability consist of SMILE (Structural Modeling, Inference, and Learning Engine), SMILEarn, and JSMILE. SMILE is a reasoning engine that is used for graphical probabilistic models and provides functionality to perform diagnosis. SMILEarn is used for obtaining data from a data source, preprocessing the data, and learning the causal structure of BN models. JSMILE is used for accessing the SMILE library from the web-based interface. This section provides some more detailed information about SMILE, SMILEarn and JSMILE wrapper.

SMILE (Structural Modeling, Inference, and Learning Engine) is a fully platform independent library of functions implementing graphical probabilistic and decision-theoretic models, such as Bayesian networks, influence diagrams (IDs), and structural equation models (Druzdzel, 1999). Its individual functions, defined in the SMILE Application Programmer Interface (API), allow creating, editing, saving, and loading graphical models, and using them for probabilistic reasoning and decision making under uncertainty. SMILE can be embedded in programs that use graphical probabilistic models as their reasoning engines. Models developed in SMILE can be equipped with a user interface that best suits the user of the resulting application. SMILE is written in C++ in a platform-independent manner and is fully portable. Model building and the reasoning process are under full control of the application program as the SMILE library serves merely as a set of tools and structures that facilitates them.

SMILEarn extends the functionality provided by SMILE. It provides a set of specialized classes that implement learning algorithms and other useful tools for automatically building graphical models from data. It is a C++ library that contains a set of data structures, classes, and functions that implement learning algorithms for graphical models and includes other functionality (such as data access, storage and preprocessing) that can be used in a model in conjunction with SMILE. Although SMILEarn is a module of SMILE, which means that it requires SMILE to be used, but one can use SMILE without the need to install and use SMILEarn.

JSMILE is a library of java classes for reasoning about graphical probabilistic models, such as Bayesian networks and influence diagrams. It can be embedded in programs that use graphical probabilistic models as a reasoning engine. It is a wrapper library that enables access to the SMILE and SMILEXML C++ libraries from java applications. JSMILE is not limited to stand-alone applications. It can also be used on the back-end side of a multi-tiered application.

3. Related works

There are various kinds of software applications that can be used to create decision theoretic models, learn the causal structure, and perform diagnosis based on BNs and IDs. There are both commercial and non-commercial software applications available. The commercial

software applications are widely used in a business environment. Many of them are integrated into business analysis software and used particularly for solving difficult business problems. The non-commercial software applications are extensively used for the educational purposes. This article reviews only the most relevant subset of non-commercial software applications based on BNs and IDs.

B-Course is an analysis tool that was developed in the fields of Bayesian and causal modelling (Mylltmaki et al., 2002). It is a free web-based online data analysis tool, which allows users to analyze data for multivariate probabilistic dependencies. It also offers facilities for inferring certain type of causal dependencies from the data. B-Course is used via a web-browser, and requires the user's data to be a text file with data presented in a tabular format typical for any statistical package (e.g., SPSS, Excel text format). It offers a simple three step procedure (data upload, model search, and analysis of the model) for building a BN dependency model. After searching the model, B-Course provides the best model to the user via a report. Users can continue to search for the next best model but they must make the decision for selecting the best model that fits their needs. Selecting the best model is sometimes very difficult for inexperienced users. In B-Course, there are no structural learning algorithms provided for the user to aid in selection. The analysis method, modelling assumptions, restrictions, model search algorithms, and parameter settings are totally transparent to the user.

Elvira is a tool for building and evaluating graphical probabilistic models (Lacave et al., 2007). It is a non web-based application. It is implemented in Java, so that it can run on different platforms. It contains a graphical interface for editing networks, with specific options for canonical models (e.g., OR, AND, MAX, etc.), exact and approximate algorithms for discrete and continuous variables, explanation facilities, learning methods for building networks from databases, algorithms for fusing networks, etc. Elvira is structured as four main modules: (1) data representation- containing the definition of the data structures that are needed for managing BNs and IDs in Java, (2) data acquisition- including the classes that are necessary for saving and loading a network from either a file or a database, (3) processing - implementing the algorithms for processing and evaluating models, and (4) visualization - defining the Elvira graphical user interface (GUI) which obviously makes use of the classes that are included in the previous modules.

GeNIe (Graphical Network Interface) is a versatile and user friendly development environment for building graphical decision models (Druzdzel, 1999). The original interface was designed for SMILE which is described in a previous section. GeNIe may be seen as an outer shell to SMILE. GeNIe is implemented in Visual C++ and draws heavily on the Microsoft foundation classes. GeNIe provides numerous tools for users such as an interface to build Bayesian network models or influence diagrams, to learn the causal relationships of a model using various algorithms, and to perform model diagnosis. In order to use GeNIe efficiently, the GeNIe software must be installed and the user should have some background knowledge about probabilistic graphical models and become familiar with the tools provided in GeNIe.

Poompuang, et al presents a development environment for building graphical decision-theoretic models based on Bayesian networks and influence diagrams working on the website by utilizing an original engine called "SMILE" (Poompuang, et al., 2007). They propose the idea of building and developing graphical decision-theoretic models on a web page in order to overcome such the limitation of Bayesian belief network software developed on a windows-based platform, which makes the models not easily portable and

is limited in its graphical representation across multiple system platforms. They present a prototype of Bayesian network models and influence diagrams in a World Wide Web environment, which can be displayed by a standard web browser.

Tungkasthan, et al presents a visualization of BN and influence Diagram models on a website (Tungkasthan et al., 2008). They develop an application based on the Macromedia Flash and Flash Remoting technologies. The application model on the client side is constructed by using the Macromedia Flash and the connection between a client and web server is developed by using the Flash Remoting technology. They use the capability of Marcomedia Flash and Flash Remoting technology to build richer, more interactive, more efficient, and more intuitive user interfaces for their applications than are possible with other web technologies such as JSP and Java applets. Their applications also provide a powerful, intuitive drag-and-drop graphical authoring tool that is comfortable for the users and have quick-loading and dynamic interfaces.

Jongsawat, et al presents a technique to dynamically feed data into a diagnostic Bayesian network model and a web-based user interface for the models (Jongsawat et al., 2008). In their work, the BN model (the students' attitude towards several factors in a college enrolment decision) is fixed and the data obtained from an online questionnaire are saved into a database and transferred to the model. The user can observe the changes in the probability values and the impact the changes have on each node in real-time after clicking on a belief update button. Users can also perform Bayesian inference in the model and they can compute the impact by observing values of a subset of the model variables on the probability distribution over the remaining variables based on real-time data.

Jongsawat, et al presents a SMILE web-based interface that permits users to build a Bayesian network causal structure from a dataset or database and perform Bayesian network diagnosis through the web (Jongsawat & Premchaiswadi, 2009). There are several learning algorithms such as Greedy Thick Thinning, PC, Essential Graph Search, and Naive Bayes provided for the user. The user can just select the desired learning algorithm and adjust its parameter settings to learn the model structure. After building the BN structure, the user is able to quantify uncertain interactions among random variables by setting observations (evidence) and use this quantification to determine the impact of the observations. The SMILE web-based interface was developed based on SMILE, SMILEarn, and SMILE.NET. It uses a novel, user-friendly interface which interweaves the steps in the BN analysis with brief support instructions on the web page.

4. Design and implementation

The following steps in this section describe how a practical framework and SMILE web-based interface are designed and implemented for automating the building of diagnostic BN models from online data sources. The structure of the proposed framework is presented in Fig. 1.

In an article by the U.S. News & World Report's "World's Best Universities rankings based on the Times Higher Education-QS World University Rankings in 2009" was selected to be the case study and the source of information for the BN model construction. The top 400 world's best universities were reported. There are six categories in each rank to be scored and reported on the web site. They consist of the following items: Academic Peer Review, Employer Review, Student to Faculty, International Faculty, International Students, and Citations per Faculty. In the data preparation process, we built RSS feeds from these online sources. The sample of RSS feeds is described below.
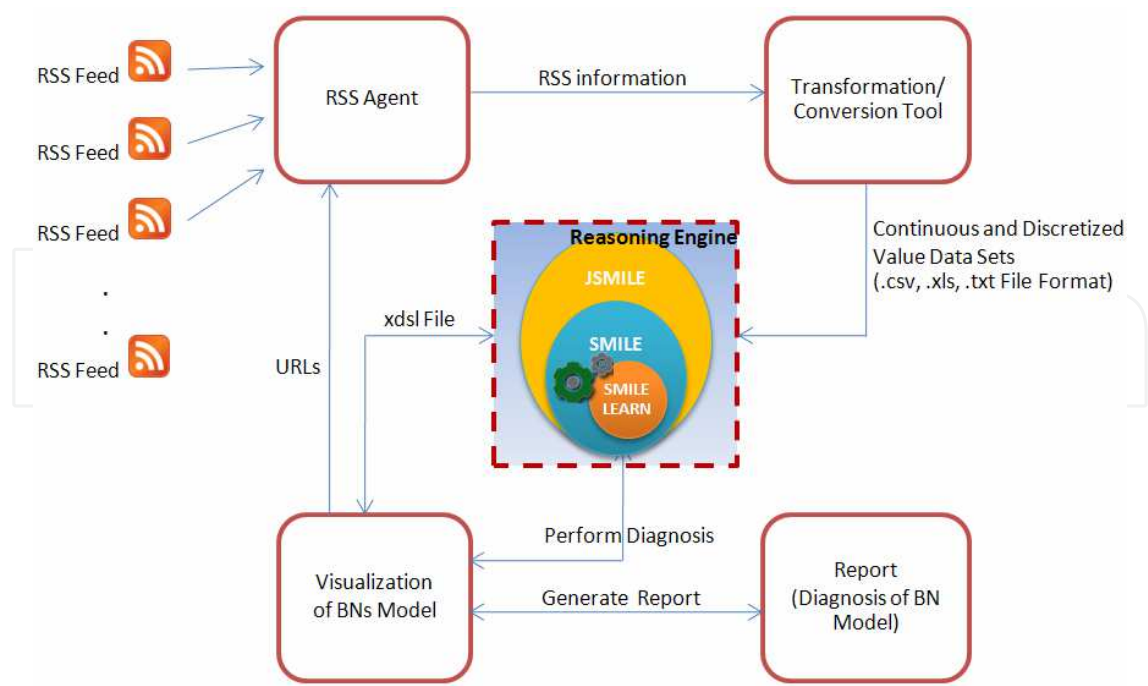
Fig. 1. A practical framework for automating the building of diagnostic BN models from online data sources

```xml
<?xml version="1.0" encoding="utf-8"?>
<rss version="2.0">
<channel>
        <title>Top 400 world's best universities</title>
        <link>http://www.usnews.com/articles/education/worlds-best-colleges/2009</link>
        <description> Top 400 world's best universities were reported</description>
        <lastBuildDate> Mon, 10 Aug 2009  18:37:00 GMT</lastBuildDate>
        <language>en-us</language>
<item>
        <uname>Harvard University</uname>
        <academicPeerReviewScore>100</academicPeerReviewScore>
        <employerReviewScore>100</employerReviewScore>
        <studentToFacultyScore>96</studentToFacultyScore>
        <internationalFacultyScore>87</internationalFacultyScore>
        <internationalStudentsScore>81</internationalStudentsScore>
        <citationsPerFaculty>100</citationsPerFaculty>
</item>
<item>
        <uname>Yale University</uname>
        <academicPeerReviewScore>100</academicPeerReviewScore>
        <employerReviewScore>100</employerReviewScore>
        <studentToFacultyScore>89</studentToFacultyScore>
        <internationalFacultyScore>71</internationalFacultyScore>
        <internationalStudentsScore>98</internationalStudentsScore>
        <citationsPerFaculty>100</citationsPerFaculty>
</item>
.
.
.
</channel>
</rss>
```

Beyond the creation of RSS feeds of the top 400 world's best universities, we also created RSS feeds for other topics such as World's Best Colleges in Asian and Middle Eastern Universities, Australian and New Zealand Universities, British and European Universities, and Canadian Universities.

First, the RSS agent is used to collect and read the provided RSS feeds according to the agent's predefined URLs. Fig. 2 shows the screenshot of the SMILE web-based interface. Users click on "Add URL to SMILElist" option to add URLs that contains RSS feeds to the list. Next, they click on the "SMILElist" option to display all added URLs in the table and then click on the checkbox to select the URLs (RSS feeds) that are required for the Bayesian belief network construction, then choose "Import Data" from the data list, and finally click on the OK button to pass the URLs to the agent. The agent gathers the RSS feeds according to the specified URLs and passes them to the transformation/conversion tool.
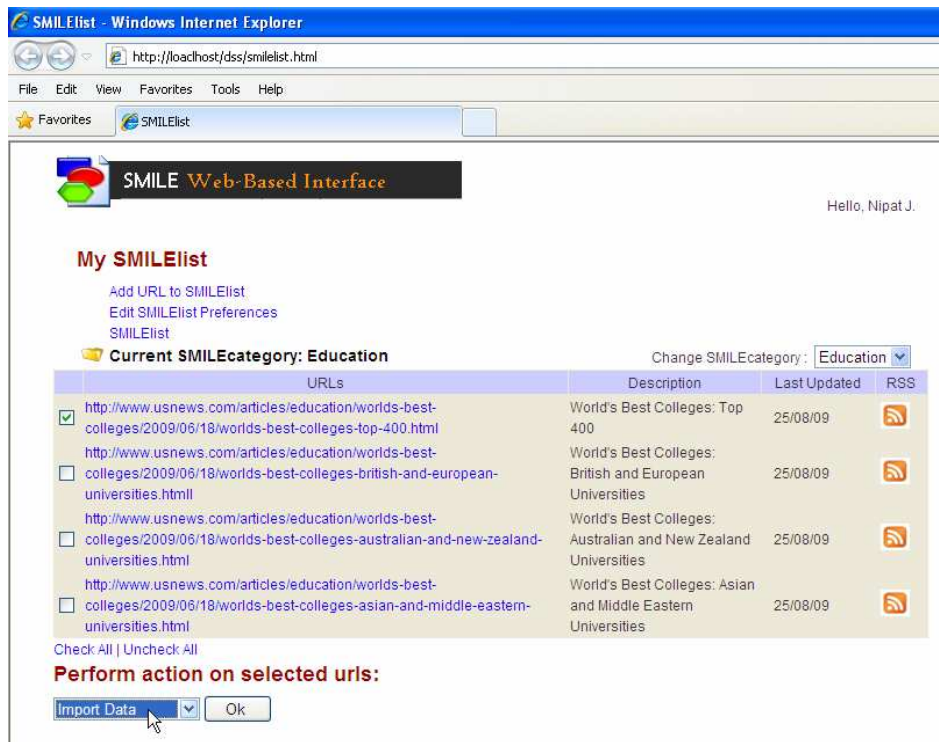
Fig. 2. Screenshot of SMILE web-based interface

Second, the transformation/conversion tool provides a method to cope with pre-processing the data obtained from the RSS feeds and handles missing values in the dataset, checks the compatibility and integration of collected data, and converts them into two separate data sets (continuous and discrete valued data sets). Fig. 3(a) and Fig. 3(b) show continuous and discrete valued data sets displayed using a datagrid view on the web. The users choose one of the issues from the list in a combo box above the data grid table. The checkbox for each variable is automatically displayed and the variables and their data are loaded into the datagrid table. Users can choose some of the variables by clicking on the checkbox or choose the "Select All" option to select all variables to be included for learning and building the causal structure of the BN model. If they select the type of data set (continuous or discrete) from the list in a combo box below the datagrid table, the selected data set is loaded and displayed in the datagrid view. To select a "Discrete" value data set, they must specify the properties of each variable such as state name, lower bound value, and upper bound value

for each state of the variables. The screenshot of defining and editing the properties of each variable are shown in Fig. 4(a) and Fig. 4(b). Next, they select a learning algorithm (Thick Thinning, PC, Essential Graph Search, and Naive Bayes) from the list in a combo box below and then click on the "Create Model" button to pass these parameter settings to the core engine.
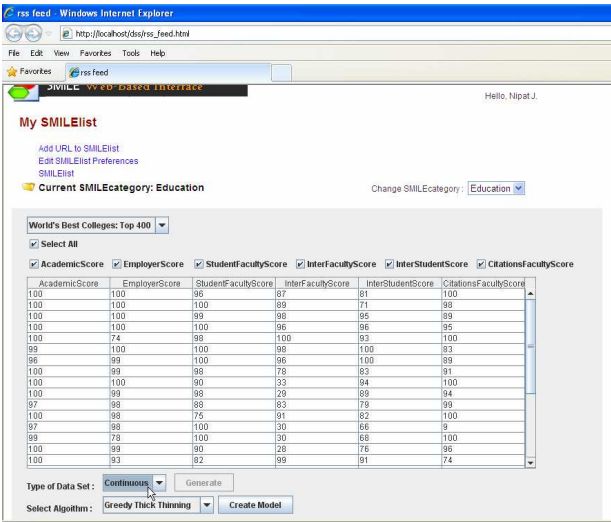

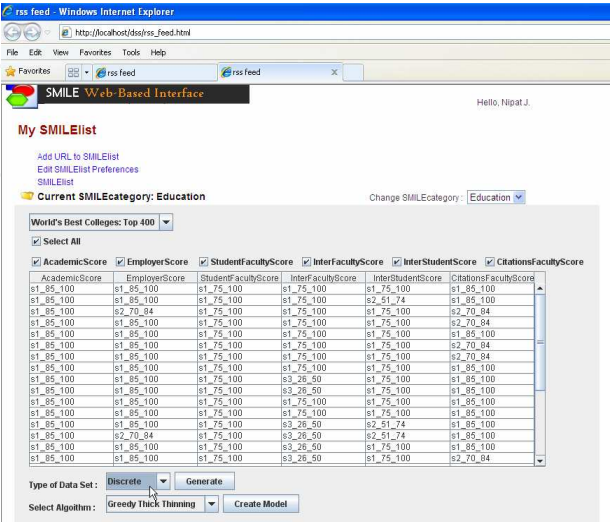Fig. 3(a). A continuous valued data set
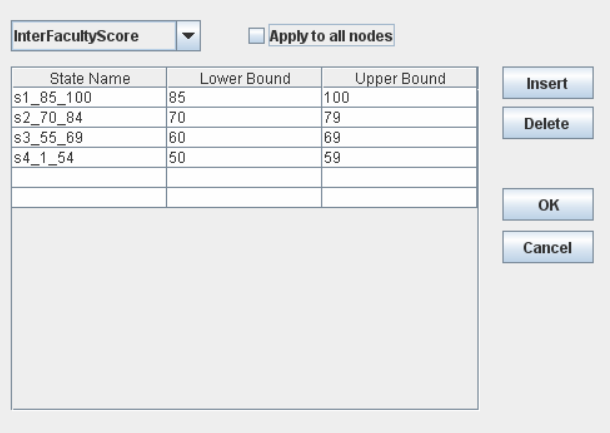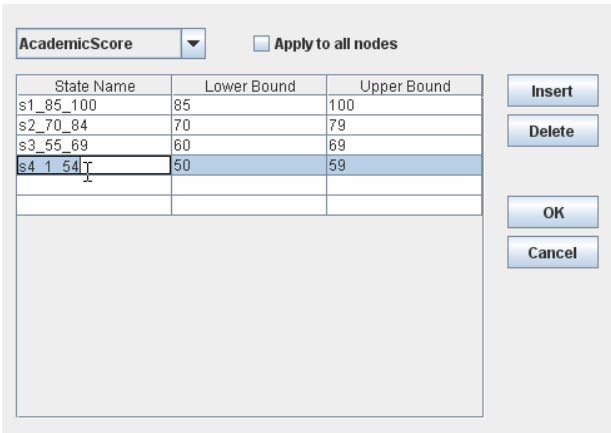

Fig. 3(b). A discrete valued data set

Fig. 4(a). Define the properties for the variable

Fig. 4(b). Edit the details of each variable

Third, the core reasoning engine receives continuous or discrete valued data sets, learns and builds the causal structure of the BN model, and performs diagnosis analysis on the BN model. This is one of the most important components of the framework. It consists of JSMILE, SMILEarn, and SMILE. JSMILE is the outermost shell of the core reasoning engine. It is used as a wrapper for accessing the SMILE and SMILEarn library from the SMILE web-based interface. It calls the functions, passes the parameter values to SMILE and SMILEarn and receives the return values. SMILEarn is actually a module of SMILE. It is used to cope with incoming data from previous steps, pre-processes the data (removing or filling missing values in the dataset, discretization of continuous values, etc.), and learns the causal structure of the BN model. It provides a set of specialized classes that implement learning algorithms and other useful tools for automatically building graphical models from data.

The sample of source code shown below is the class that implements the greedy thick thinning procedure for learning the structure and parameters of a Bayesian network.

```
class DSL_network;
class DSL_dataset;
class DSL_greedyThickThinning
    {
    public:
        DSL_greedyThickThinning()
        {
                maxParents = 5;
                priors = K2;
                netWeight = 1.0;
        }

        int Learn(const DSL_dataset &data, DSL_network &net);
        enum PriorsType { K2, BDeu };
        PriorsType priors;
        int maxParents;
        double netWeight;
        typedef std::vector<std::pair<int, int> > IntPairVector;
        IntPairVector forcedArcs;
        IntPairVector forbiddenArcs;
        IntPairVector tiers;
    };
```

There are several fields of this class that are used for defining some details of the learning algorithm. For example, the "maxParents" field defines the maximal number of parents a node can have. The "priors" field defines the type of priors (K2 method). The learning method, Learn (const DSL_dataset &data, DSL_network &net), performs the actual learning procedure. The first argument is the input dataset. The result from the learning procedure is stored in the DSL_network which is the second argument. The method returns DSL_OKAY if learning was successful and an error code otherwise.

SMILE is mainly used for graphical BN models and provides the functionality to perform a diagnosis. With SMILE diagnosis, a user can determine the state of the network by performing tests or observations. A user is able to select a test and perform it by setting the evidence for the test.

Fourth, the visualization part of the BN model is automatically built by using a java applet and placed on the SMILE web-based interface. In the BN model, the states of each variable or node are automatically altered in real-time when the RSS feeds are updated. The agent checks for the updated RSS feed and loads metadata about its content into the BN model. The update belief function of the core engine is called to update the probability values of the states of each variable and displays them on the model. Fig. 5 depicts a simplified representation of the communication between a web browser and a web server. In the first and second steps, the client makes requests for a web page from the web server. The web server returns an applet back to client side. In the third and last step, the applet will handle and communicate with a servlet application that runs on the web server. Fig. 6 shows the BN model on the SMILE web-based interface. Users select "Picking" from the list in a combo box and clicks on each node to drag and drop it independently on the applet area. They can zoom in or out of the model by clicking on "zoom in/zoom out" button. They click on the

"Update Belief" button to update the probability values of all variables in BN model. The tooltip text for the updated probability results appears when they move the mouse cursor over any node. The updated probability results on each node are shown in Fig. 7.

Lastly, for BN diagnosis, the user is allowed to perform a model diagnosis by entering observations (evidence) for some of the context and evidence variables. Fig. 8 shows the screenshot of the BN model diagnosis. The user begins the BN model diagnosis by performing a right click on a node and selects the state for setting the evidence for the test. After setting the evidence, they click on the "Update Belief" button to update the model.
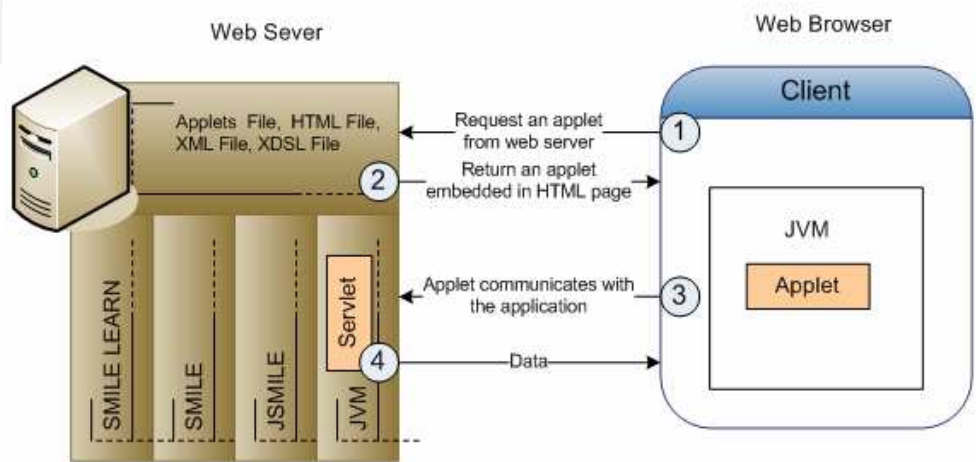
Fig. 5. SMILE web-based interface architecture
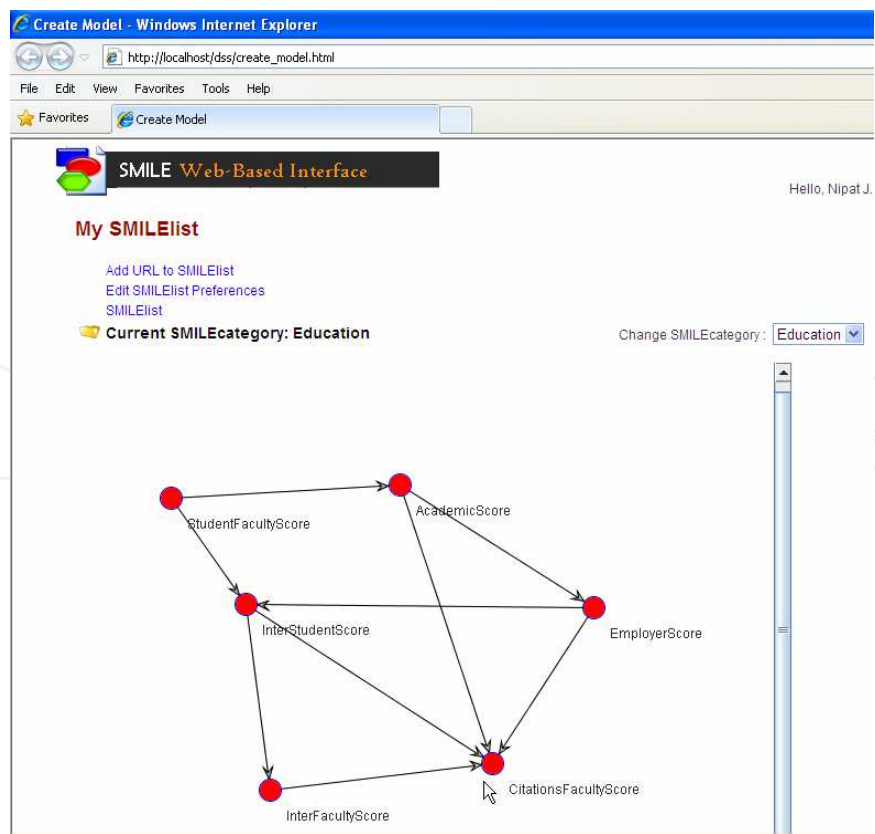
Fig. 6. Screenshot of the BN model on SMILE web-based interface
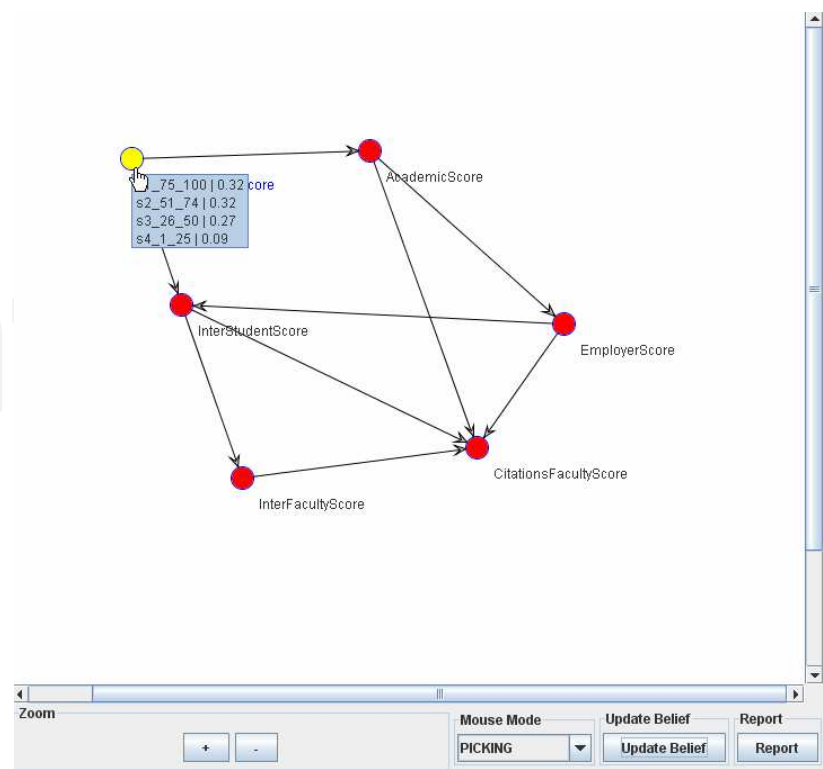
Fig. 7. Screenshot of the probability values on the node after updating belief
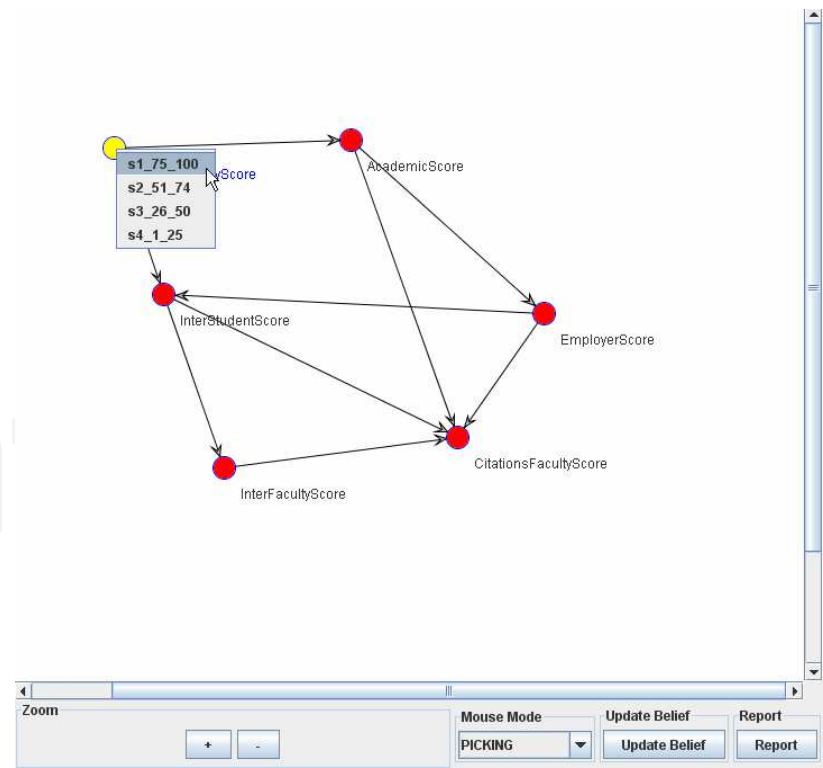
Fig. 8. Screenshot of a BN model diagnosis

Next, they click on the "Report" button to display a graphical representation of the BN model. The report graphically displays a BN model and its probability values with a horizontal bar graph. It is shown in Fig. 9 and Fig. 10.
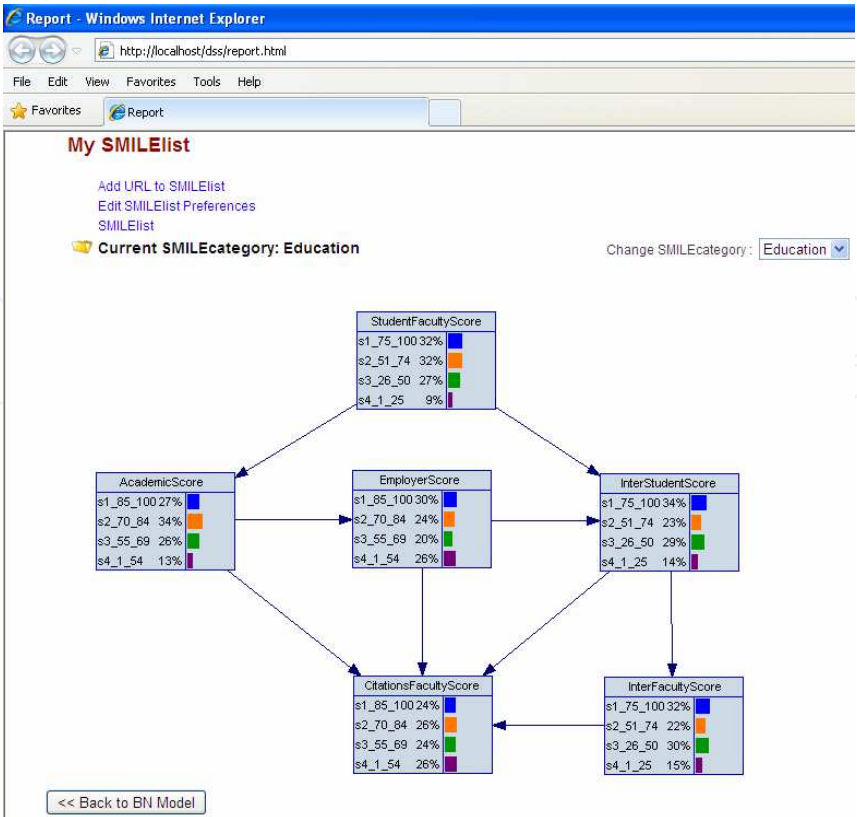
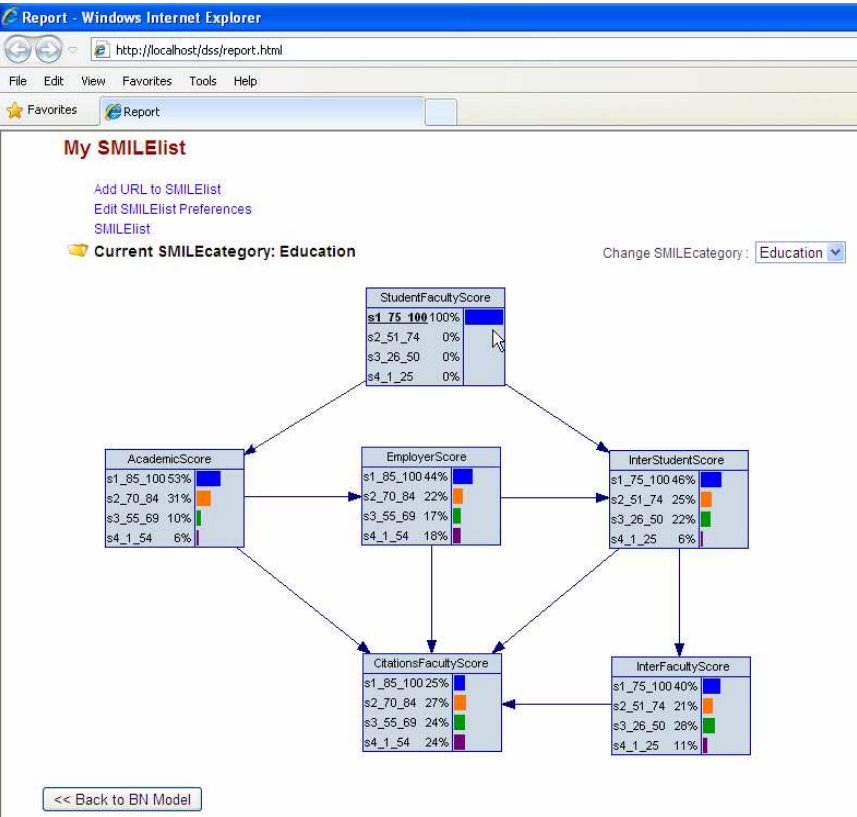Fig. 9. Screenshot of a graphical report on a BN model

Fig. 10. Screenshot of a graphical report on a BN model (Set Evidence)

5. Conclusion

This paper presented a practical framework for automating the building of diagnostic BN models from data sources obtained from the WWW and demonstrates the use of a SMILE web-based interface to represent them. The framework consists of the following components: RSS agent, transformation/conversion tool, core reasoning engine, and the SMILE web-based interface. The RSS agent automatically collects and reads the provided RSS feeds according to the agent's predefined URLs. A transformation/conversion tool provides a method to handle the pre-processing of the data obtained from the RSS feeds and handles missing values, checks the compatibility and integration of collected data, and converts them into two separate continuous and discrete valued data sets. The core reasoning engine learns and builds the causal structure for the BN model, and performs probabilistic inference for the Bayesian belief network. A SMILE web-based interface permits users to perform Bayesian network diagnosis through the web. They can quantify uncertain interactions among random variables by setting observations (evidence) and use this quantification to determine the impact of observations. A graphical structure for representing the BN model is shown as a report and displayed to the user.

The two main principles utilized in the proposed framework and SMILE web-based interface were transparency and ease of use. Our future work will focus on improving a decision-oriented diagnosis approach. The SMILE web-based interface has been extended to cope with influence or relevance diagrams. The next version of the application will allow users to quantify a decision maker's decision options and preferences and use these to determine an optimal decision policy.

6. Acknowledgement

7. References

Lacave, C.; Luque, M. and Díez, F.J. (2007). Explanation of Bayesian Networks and Influence Diagrams in Elvira. *IEEE Transactions on Systems, Man, AND Cybernetics-Part B: Cybernetics*, Vol. 37, No. 4, p. 952-965.

Yuan, C. & Druzdzel, M.J. (2003). An Importance Sampling Algorithm Based on Evidence Pre-propagation. *Nineteenth International Conference on Uncertainty in Artificial Intelligence*, Acapulco, Mexico, p. 624-631.

Dagum, P., & Luby, M. (1993). Approximate probabilistic reasoning in Bayesian belief network is NP-Hard. *Artificial Intelligence*, Vol. 60, p. 141-153.

Dagum, P. & Luby, M. (1997). An Optimal Approximation Algorithm for Bayesian Inference, *Artificial Intelligence*, Vol.93, p.1-27.

Jensen, F.V.; Olesen, K. G. and Andersen, S.K. (1990). An Algebra of Bayesian Belief Universes for Knowledge-Based Systems. *Networks: Special Issue on Influence Diagrams*, Vol.20, No. 5, August 1990, p.637-659.

Fung, R., & Chang, K.C. (1989). Weighing and Integrating Evidence for Stochastic Simulation in Bayesian Networks. *In Proceedings of the Fifth Conference on Uncertainty in Articial Intelligence (UAI-89)*, p.209-219, New York, N.Y., Elsevier Science Publishing Company, Inc.

Fung, R., & DelFavero, B. (1994). Backward Simulation in Bayesian Networks. *In Proceedings of the Tenth Annual Conference on Uncertainty in Artificial Intelligence (UAI–94)*, p.227-234, San Francisco, C.A., Morgan Kaufmann Publishers.

Cooper, G.F. (1990). The Computational Complexity of Probabilistic Inference using Bayesian Belief Networks, *Artificial Intelligent*, Vol. 42, No. 2-3, p. 393-405.

Cheng, J. & Druzdzel, M.J. (2000). AIS-BN: An Adaptive Importance Sampling Algorithm for Evidential Reasoning in Large Bayesian Networks. *Journal of Artificial Intelligence Research (JAIR)*, Vol. 13, p. 155-188.

Jongsawat, N.; Poompuang, P. & Premchaiswadi, W. (2008). Dynamic Data Feed to Bayesian Network Model and SMILE Web Application. *In Proceedings of IEEE on Software Engineering: Artificial Intelligence, Networking, and Parallel/Distributed Computing*, p. 931-936, Phuket, Bangkok, Thailand.

Jongsawat, N. & Premchaiswadi, W. (2009). A SMILE Web-Based Interface for Learning the Causal Structure and Performing a Diagnosis of a Bayesian Network. *Proceedings of IEEE on Systems, Man, and Cybernetic, Systems Science & Engineering (SMC2009)*, San Antonio, Texas, USA.

Druzdzel, M.J. (1999). SMILE: Structural Modeling, Inference, and Learning Engine and GeNIe: A Development Environment for Graphical Decision-Theoretic Models. *In Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI–99)*, p. 902-903, Orlando, FL.

Pearl, J. (1988). Probabilistic Reasoning in Intelligent Systems. *Networks of Plausible Inference*, San Mateo, CA, Morgan Kaufmann Publishers.

Pearl, J. (1986). Fusion, Propagation, and Structuring in Belief Networks. *Artificial Intelligence*, Vol. 29, No. 3, p. 241-288.

Mylltmaki, P.; Silander, T.; Tirri, H. & Uronen, P. (2002). B-Course a Web-based Tool for Bayesian and Causal Data Analysis. *International Journal on Artificial Intelligence Tools*, Vol. 11, No. 3, p. 369-387.

Poompuang, P.; Kungtasthan, A. ; Jongsawat, N. & Sutheebanjard, P. (2007). Graphical Decision-Theoretic Models on the Web. *Proceedings of Knowledge Management*, p. 163-170, Bangkok, Thailand.

Shachter, R.D. & Peot, M.A. (1989). Simulation Approaches to General Probabilistic Inference on Belief Networks. *In Uncertainty in Artificial Intelligence*, p. 221-231, New York, N.Y., Elsevier Science Publishing Company, Inc.

Lauritzen, S.L. & Spiegelhalter, D.J. (1988). Local Computations with Probabilities on Graphical Structures and their Application to Expert Systems (With Discussion). *Journal of the Royal Statistical Society Series B*, Vol. 50, No 2, p.157-224.

Tungkasthan, A.; Poompuang, P. & Premchaiswadi, W. (2008). SMILE Visualization with Flash Technologies, *Proceedings of IEEE on Software Engineering: Artificial Intelligence, Networking, and Parallel/Distributed Computing*, p. 551-556, Phuket, Bangkok, Thailand.
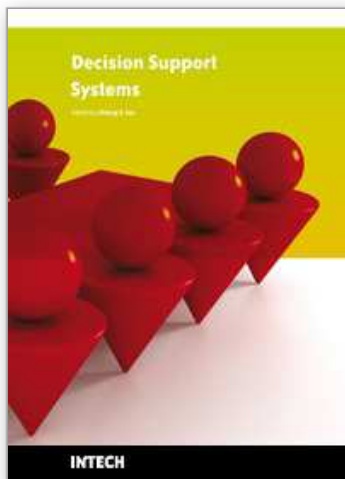
Decision Support Systems

Edited by Chiang S. Jao

Decision support systems (DSS) have evolved over the past four decades from theoretical concepts into real world computerized applications. DSS architecture contains three key components: knowledge base, computerized model, and user interface. DSS simulate cognitive decision-making functions of humans based on artificial intelligence methodologies (including expert systems, data mining, machine learning, connectionism, logistical reasoning, etc.) in order to perform decision support functions. The applications of DSS cover many domains, ranging from aviation monitoring, transportation safety, clinical diagnosis, weather forecast, business management to internet search strategy. By combining knowledge bases with inference rules, DSS are able to provide suggestions to end users to improve decisions and outcomes. This book is written as a textbook so that it can be used in formal courses examining decision support systems. It may be used by both undergraduate and graduate students from diverse computer-related fields. It will also be of value to established professionals as a text for self-study or for reference.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

INTECH

open science | open minds