Volume 37 Number 1 March 2013

ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

Special Issue:

100 Years of Alan Turing and 20 Years of SLAIS

Guest Editors:

Dunja Mladenić
Stephen Muggleton
Ivan Bratko

1977

Editorial Boards, Publishing Council

Informatica is a journal primarily covering intelligent systems in the European computer science, informatics and cognitive community; scientific and educational as well as technical, commercial and industrial. Its basic aim is to enhance communications between different European structures on the basis of equal rights and international refereeing. It publishes scientific papers accepted by at least two referees outside the author's country. In addition, it contains information about conferences, opinions, critical examinations of existing publications and news. Finally, major practical achievements and innovations in the computer and information industry are presented through commercial publications as well as through independent evaluations.

Editing and refereeing are distributed. Each editor from the Editorial Board can conduct the refereeing process by appointing two new referees or referees from the Board of Referees or Editorial Board. Referees should not be from the author's country. If new referees are appointed, their names will appear in the list of referees. Each paper bears the name of the editor who appointed the referees. Each editor can propose new members for the Editorial Board or referees. Editors and referees inactive for a longer period can be automatically replaced. Changes in the Editorial Board are confirmed by the Executive Editors.

The coordination necessary is made through the Executive Editors who examine the reviews, sort the accepted articles and maintain appropriate international distribution. The Executive Board is appointed by the Society Informatika. Informatica is partially supported by the Slovenian Ministry of Higher Education, Science and Technology.

Each author is guaranteed to receive the reviews of his article. When accepted, publication in Informatica is guaranteed in less than one year after the Executive Editors receive the corrected version of the article.

Editors’s Introduction to the Special Issue on “100 Years of Alan Turing and 20 Years of SLAIS”

Alan Mathison Turing (1912 - 1954) was an English mathematician, logician, cryptanalyst, and computer scientist. He was highly influential in the development of computer science, providing a formalisation of the concepts of "algorithm" and "computation" by way of the Turing machine. His work played a pivotal role in the creation of the modern computer science. Turing is widely considered to be the father of Computer Science and Artificial Intelligence.

SLAIS (Slovenian Artificial Intelligence Society) is an association of researchers and practitioners in the field of Artificial Intelligence in Slovenia. Most of them come from universities and research institutes, but there are members from industrial and commercial organizations as well. The society promotes theoretical and applied research as well as the transfer of AI technology to industrial and commercial environments. SLAIS was founded in 1992 and is a member society of ECCAI (European Coordinating Committee for Artificial Intelligence).

This special issue is centered on the Alan Turing centenary or the Alan Turing Year and the 20th anniversary of the Slovenian Artificial Intelligence Society.

The first theme of the special issue is related to Turing’s unique impact on Mathematics, Computing, Computer Science, Informatics, Artificial Intelligence, Philosophy and computational aspects of Physics, Biology, Linguistics, Economics and the wider scientific world. In our call for papers relating to Turing, contributions were encouraged either concerning his life or his achievements. In addition, we also called for papers relating to Donald Michie (1923–2007), Turing's contemporary and active member of Slovenian national institutions (associate member of J. Stefan Institute, and corresponding member of Slovene Academy of Sciences and Arts).

The second theme of the special issue is related to the 20th anniversary of SLAIS, the Slovenian AI society. We called for papers about important achievements of Slovenian AI that importantly contributed to the field of AI in the national and international context, both in theory and practice. We encouraged authors to present achievements from an historical perspective, their contributions to the field of AI, their impacts to an information society and possible impacts on the advancement of AI.

We are delighted to present the nine papers comprising this special issue. Two papers focus on Alan Turing. *The Child Machine vs. the World Brain* discusses Alan Turing's Child Machine idea of learning as an incremental process based on a mixture of instruction and trial-and-error learning. *Alan Turing, Turing Machines and Stronger* discusses contributions of Alan Turing to Computer Science, comparing Turing's importance to that of Einstein within Physics. Seven papers focus on different aspects of Artificial Intelligence. Two of them provide a summary of research issues, results and systems in the field of *Mining Big Data in Real Time* and in the field *Data Stream Mining For Ubiquitous Environments.* The remaining five papers review research contributions of SLAIS members over substantial periods of time: *Automatic Text Analysis by Artificial Intelligence*, *Advances in Data Mining for Biomedical Research*, *Explanation and Reliability of Individual predictions* in machine learning, *DEX Methodology: Thirty three years of qualitative multi-attribute modeling*, and *ORANGE: Data Mining Fruitful and Fun*. The latter two papers describe development issues in two publicly available systems that have over decades attracted large user communities world wide.

Dunja Mladenić
Stephen Muggleton
Ivan Bratko

Editors of the special issue

The Child Machine vs the World Brain

Claude Sammut
School of Computer Science and Engineering, The University of New South Wales, Sydney, Australia
claude@cse.unsw.edu.au

Keywords: Turing, Machine Learning

Received: December 17, 2012

Machine learning research can be thought of as building two different types of entities: Turing's Child Machine and H.G. Wells' World Brain. The former is a machine that learns incrementally by receiving instruction from a trainer or by its own trial-and-error. The latter is a permanent repository that makes all human knowledge accessible to anyone in the world. While machine learning began following the Child Machine model, recent research has been more focussed on "organising the world's knowledge"

Povzetek: Raziskovanje strojnega učenja je predstavljeno skozi dve paradigmi: Turingov Child Machine in H.G. Wellsov World Brain.

1 Encountering Alan Turing through Donald Michie

My most immediate knowledge of Alan Turing is through many entertaining and informative conversations with Donald Michie. As a young man, barely out of school, Donald went to work at Bletchley Park as a code breaker. He became Alan Turing's chess partner because they both enjoyed playing but neither was in the same league as the other excellent players at Bletchley. Possessing similar mediocre abilities, they were a good match for each other. This was fortunate for young Donald because, when not playing chess, he learned much from Turing about computation and intelligence. AlthoughTuring's investigation of machine intelligence was cut short by his tragic death, Donald continued his legacy. After an extraordinarily successful career in genetics, Donald founded the first AI group in Britain and made Edinburgh one of the top laboratories in the world, and, through a shared interest in chess with Ivan Bratko, established a connection with Slovenian AI.

I first met Donald when I was as a visiting assistant professor at the University of Illinois at Urbana-Champaign, working with Ryszard Michalski. Much of the team that Donald had assembled in Edinburgh had dispersed as a result of the Lighthill report. This was a misguided and damning report on machine intelligence research in the UK. Following the release of the report, Donald was given the choice of either teaching or finding his own means of funding himself. He chose the latter. Part of his strategy was to spend a semester each year at Illinois, at Michalski's invitation, because the university was trying to build up its research in AI at that time. The topic of a seminar that Donald gave in 1983 was "Artificial Intelligence: The first 2,400 years". He traced the history of ideas that lead to the current state of AI, dating back to Aristotle. Of course, Alan Turing played a prominent role in that story. His 1950 Mind paper [1] is rightly remembered as a landmark in the history AI and famously describes the imitation game. However, Donald always lamented that the final section of the paper was largely ignored even though, in his opinion, that was the most important part. In it, Turing suggested that to build a computer system capable of achieving the level of intelligence required to pass the imitation game, it would have to be educated, much like a human child.

> Instead of trying to produce a programme to simulate the adult mind, why not rather try to produce one which simulates the child's? If this were then subjected to an appropriate course of education one would obtain the adult brain. Presumably the child-brain is something like a notebook as one buys from the stationers. Rather little mechanism, and lots of blank sheets... Our hope is that there is so little mechanism in the child-brain that something like it can be easily programmed. The amount of work in the education we can assume, as a first approximation, to be much the same as for the human child.

He went on to speculate about the kinds of learning mechanisms needed for the child machine's training. The style of learning was always incremental. That is, the machine acquires knowledge by being told or by its own exploration and this knowledge accumulates so that it can learn increasingly complex concepts and solve increasingly complex problems.

Early efforts in Machine Learning adopted this paradigm. For example, the Michie and Chambers [2] BOXES program learned to balance a pole and cart system by trial-and-error receiving punishments are rewards, much as Turing described, and like subsequent reinforcement learning systems. My own efforts, much later, with the Marvin program [3] were directed towards building a system that could accumulate learn and accumulate concepts expressed in a form of first order logic. More recent systems that learn in this manner include the Robot Scientist [4] and the Xpero robot learning project [5]. However, most current machine learning research has adopted a somewhat different style. Turing could not have foreseen the internet and the effect it would have on AI. Access to huge amounts of data and computing resources suggest a kind of intelligence that may be built in a very different way than human intelligence.

Another significant historical figure did anticipate something like the internet. At around the same time of Turing's paper on computing machines [6], H.G. Wells speculated on what he called "the world brain" [7], a permanent world encyclopedia that would provide, "...a sort of mental clearing house for the mind, a depot where knowledge and ideas are received, sorted, summarised, digested, clarified and compared.... any student, in any part of the world, will be able to sit with his projector in his own study at his or her convenience to examine any book, any document, in an exact replica."

Wells thought that the technology for this repository of knowledge would be microfilm. He could not have known that the World Wide Web, Wikipedia and internet search engines could bring about his vision on a far larger scale than he imagined. Indeed, Google's company mission is "...to organize the world's information and make it universally accessible and useful"[1]. A core technology for such search engines is machine learning. However, it is of a form that is somewhat different form that described by Turing for his Child Machine. Unlike a human child, acquiring new knowledge incrementally, machine learning systems can access the enormous amounts of data available throughout the internet to produce, in some instances, superhuman performance. However, this is usually *all-at-once* and *single-concept-at-a-time* learning that must still be guided by humans.

The first programs capable of efficiently learning from moderately large numbers of examples began with Michalski's Aq [8] and Quinlan's ID3 [9], both of whom were influenced by Donald Michie. Quinlan was a PhD student studying with psychologist Earl Hunt when he attended a lecture by Michie at Stanford University. Donald challenged the students to devise a method of learning to determine a win in a chess end-game and Quinlan responded with the first version of his decision tree learner. The utility of these programs for finding patterns in large data sets encouraged new research into "batch" learning systems, eventually leading to the large-scale statistical learning methods commonly in use today. A characteristic of most systems that are used for mining patterns in "big data" is that they use fairly simple features and rely of masses of data to build robust classifiers. Structure in representation is generally created by the human data analyst. Currently, the construction of the world brain is a partnership between human and machine. But how far can this continue? Will humans be able to structure truly large knowledge sources as the amount and complexity of information increases, or will machines have to take over at least some of that job too? If that is the case, what mechanisms can be employed to do this?

Some search engines are already beginning to incorporate some *semantic integration*. For example, Google's Knowledge Graph[2] uses what is essentially a semantic net to supplement search information with the properties of objects and their relationships to other objects. However, most of the semantic information is derived from human built ontologies, and this poses a problem. Handcrafted ontologies reflect a knowledge engineers assumptions about structure in the data, which are not necessarily true, whereas a machine built ontology should be more faithful to the actual evidence for a particular semantic structure. Learning systems are capable if finding relations between entities [10] but the complexity is greater than learning propositional representations. This complexity can be reduced if concepts are learned incrementally. That is simple concepts are learned first and become background knowledge for further learning. Learning can also be assisted if the systems is capable of selected its own examples to test hypotheses. When these elements are incorporated, machine learning research once again resembles Turing's model of a child machine, but perhaps not entirely as he envisaged because it will also include the capabilities of the "world brain".

The study of learning in the style of the child machine, which accumulates knowledge active experimentation as been neglected, although there are some notable exceptions [11, 12, 13, 14]. There are some very significant open questions that are yet to be answered for this approach to work. We discuss these in the next section.

2 Cumulative active learning

Many of the problems associated with building a Child Machine have been addressed in the past. An unfortunate effect of the success of the "big data" approach to machine learning has been to distract researchers from investigating what I have termed cumulative active learning, which is essential for the Child Machine. The structure of a cumulative, or never-ending, learning system that acquires data by experimentation was nicely illustrated by Mitchell *et al.* [15] in 1983 (Figure 1) and his more recent work on never-ending language learning [13] follows a similar structure. An experiment or test of a hypothesis is created by a problem generator, the problem solver performs the experiment whose outcome is judged by a critic. The critic labels the results of the experiments as positive, if the problem solver succeeded in its task or negative if it failed. The learner then updates its hypothesis, based on the new training data and the process repeats as long as there are remaining learning tasks.

A requirement for cumulative learning is a mechanism

[1] http://www.google.com/about/company/

[2] http://www.google.com/insidesearch/features/search/knowledge.html

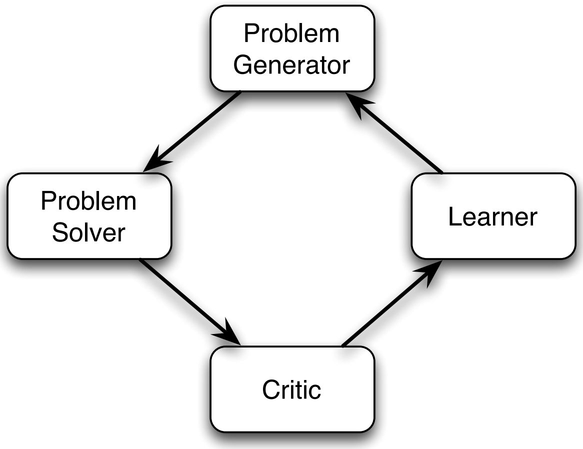


Figure 1: A theory is a network of concepts.

for storing and using learned concepts as *background knowledge*. Representing concepts in the form of expressions in first order logic makes this relatively easy. To my knowledge, Banerji [16] was the first propose such a mechanism and the first implementation of a learning system that could use learned concepts for further learning was by Cohen [17]. With Brian Cohen, I elaborated this idea to create a learning system in which the concepts were represented as logic programs and were, thus, executable [18]. This work evolved into Marvin [19] and other related systems [20, 21] developed similar capabilities. The ability to use of background knowledge has become one of the distinguish features Inductive Logic Programming [22]. However, even many ILP systems that use background knowledge to learn new concepts do not close the loop to allow those learned concepts to, themselves, become part of the background knowledge for future learning. Other systems, such NELL [13] currently only use background knowledge that is in the form of ground clauses [23].

3 Learning as debugging a logic program

Since, in practice, a learning agent can never be exposed to all instances of a concept, it is prone to making mistakes of two kinds:

- The system may observe an event for which there is no prior knowledge about how to respond.
- The system may observe an event for which there is prior knowledge. However, the known concepts are too general and the system may respond inappropriately.

When the system is learning only one concept at a time, repairing a theory is relatively easy. We just specialise an over-general theory or generalise a theory that is too specific. However, in a system that accumulates knowledge over time, learning many concepts, localising the error is not so easy. We can think of a theory as being a network of interdependent concepts, as in Figure 2, where the definition of concepts S and T rely on the definition of Q, which depends on P and E. Suppose we discover an error in the theory while attempting to use concept, T, but the real error is due to an incorrect definition of concept, E.

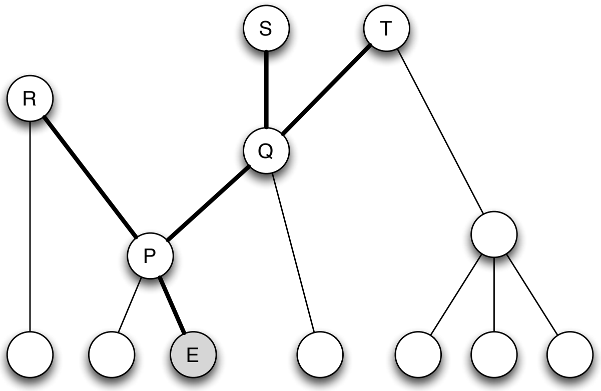


Figure 2: A concept hierarchy.

When we use a logical representation of concepts, a concept description is also an executable computer program, so one way of locating the problem is to trace through the execution of the program that lead to the unexpected result, testing each concept. That is, we debug the program, as Shapiro [24] did with MIS. To locate an error when an incorrect solution has been given (i.e. the theory contains an over-generalisation) Shapiro's debugging algorithm works backwards through the failed proof of a goal, searching for the procedure that caused the failure. In Figure 2, backtracing would begin with the last goal satisfied, that is, T. The debugger begins stepping back through the proof, i.e. down the dark path to node Q, then P if necessary, asking an oracle if the partial solution at each point is correct. If this is not true, then an erroneous clause has been found. Note that the algorithm assumes the existence of an infallible oracle. In a reactive environment, the learning program may do without an oracle since the program is able to perform experiments to test a concept. Thus, a failure suggests that the initial set of experiments that resulted in the formation of the concepts along the solution path was not extensive enough for at least one of the concepts.

A concept that is too specific may prevent the program from producing any solution at all. That is, the logic program that is supposed to satisfy the goal does not cover the initial conditions of the task. An attempt at debugging the theory can only be made when a correct solution has been seen, otherwise the learner has no indication that the task really is possible. A correct solution may be found, either by "mutating" the current theory in the hope that the goal can be satisfied by the mutant or the learner may observe another agent in the world performing the task. Shapiro's debugging method for programs that fail to produce an answer is equivalent to the second alternative, that is, the oracle supplies the correct solution. The debugger again tries to work backwards seeking clauses in the program which could have produced the given solution. Once such a clause is found, its body provides further goals that should be satisfied in order to arrive at the solution. The debugger considers each of these intermediate goals to see if they can also be produced by other clauses. Any goal that cannot be achieved indicates where the program or theory is deficient.

Up to this point, we have described mostly past research that addresses many of the problems that are needed to build a Child Machine but further work is still required. In the next section we look at a problem that has received very little attention, mainly because we have not yet built AI systems that are sufficiently complex.

4 The Economics of learning and theory revision

Detecting and repairing an error in a single concept is one thing, but repairing an entire theory is another matter. Remember that in Figure 2, we envisaged a world model or domain theory as a hierarchy of concepts. Using a horn clause representation, the head of a clause corresponds to a parent node and the goals in the body correspond to the children. These goals match other clause heads and form links to the rest of the network. Also in Figure 2, we imagined that one concept, represented by the shaded node, E, was in error. When the concept is repaired, what effect will that have on the concepts which referred to the old concept? Since P, Q, R, S and T refer, directly or indirectly, to the erroneous node they must have been learned in the presence of the error. Are they, therefore also in error or will correcting E alone correct them all?

When faced with the problem of ensuring the consistency of its knowledge base, two strategies are available to the learning system.

1. After correcting E, the system may test each of the concepts that depend on E. However revising all of the concepts dependent on one that just been modified could involve a lot of work if the network of concepts is very extensive.

2. The system may wait to see if any further errors show up. In this case, each concept will be debugged as necessary. Although more economical this method requires a method for tolerating errors if the program has been assigned a task which it must continue to perform.

When a learning system is connected to the physical world, as in a robot, it cannot rely on the accuracy of measurements from vision systems, touch sensors, etc. Thus, a program may fail because its world model does not accurately reflect the real state of the world. This being the case, the learning system must not revise its domain theory prematurely since there may not, in fact, be any errors in its theory. Therefore, a prudent approach to error recovery is to delay revision of a domain theory until sufficient evidence has accumulated to suggest the appropriate changes.

Richards and Mooney [25] proposed a theory revision system, based on earlier work by Muggleton [26, 20]. Wrobel [27] also proposed a first order theory refinement system. The main idea behind these methods is to consider repairing a theory as a kind of compression. That is, the set of clauses that form a theory may be rewritten so that the same or greater coverage is achieved by a theory that is more compact in an information theoretic sense. An important operation for this is predicate invention [20, 14]. That is, the learner must have the ability to invent its own concepts when it detects patterns that can be reused in the description of other concepts.

When an experiment fails, we must not invalidate the concepts used in planning the experiment for, as mentioned earlier, the failure may be due to noise. Instead, we note the circumstances of the failure and augment the failed concept with a description of these circumstances. Several things could happen to the concept when this is done:

- The description of the concept is modified to the extent that it becomes correct. If an alternative, correct description already existed, then the alternative domain theories of which these concepts were components, converge.

- After several failures, there is no generalisation which covers the circumstances of failure. In this case, the failures may be either due attributed to noise or to some phenomenon not yet known to the system. In either case, nothing can be done.

A truth maintenance system (TMS) [28] may be used to maintain the network of concepts that form a domain theory. The TMS stores dependencies which, when errors are found will indicate where other potential weaknesses in the theory lie. The TMS may also allow a learning program to experiment with alternative domain theories by maintaining multiple worlds.

Whatever mechanisms are used, some important questions to investigate are: what is the cost of learning and when it it worthwhile adopting a new theory in favour of an existing one. That is, putting up with occasional errors in an existing theory my cost less than building a new theory.

5 Deep knowledge vs shallow knowledge

Before concluding, I want to return to Turing's paper on the imitation game and what is needed in a child machine to be able to pass the test. Turing proposed the following criteria for success:

> I believe that in about fifty years time it will be possible to programme computers with a storage capacity of about 10^9 to make them play the imitation game so well that an average interrogator will not have more than 70 per cent chance of making the right identification after five minutes of questioning.

It should be noted that these conditions are not particularly onerous if the conversation is confined to superficial chat, but according to Donald Michie, Alan Turing never engaged in superficial chat, so the test should be understood to pose questions that require some thought. Turing, however, was not a typical conversationalist. Many human dialogues can be conducted with "canned" responses, for example, interactions at a supermarket or enquiries to a call centre, where, in fact, the operators follow quite strict scripts.

Leading up to the 50th anniversary of Turing's paper in 2000, Donald Michie suggested to me that we should try to build a conversational agent, to see how close AI was to Turing's prediction that the test could be passed within 50 years of the publication of the paper. We implemented several frameworks for building a conversational agent [29], borrowing partly from natural language processing theory, which represents deep understanding of an utterance, and also from chatterbots, which mostly rely on simple pattern matching rules, much like Eliza [30]. The idea of mixing deep and shallow knowledge is consistent with a theme of Donald's work, namely that many tasks that we think require conscious, deliberative thought are actually achieved by simpler, subcognitive processes. He was fond of a quote from A.N. Whitehead [31]:

> It is a profoundly erroneous truism... that we should cultivate the habit of thinking what we are doing. The precise opposite is the case. Civilisation advances by extending the number of important operations which we can perform without thinking about them.

This applies even to conversation. There are many circumstances in which we can, legitimately, talk without thinking. Even many aspects of an expert's decision making are subcognitive. An expert may be defined as someone who knows his or her job so well that it can be done with little thought, only requiring real deliberation when an unusual case is encountered. Learning subcognitive skills was one of Donald's great interests for many years. He coined the term *behavioural cloning* [32] to describe the process of building an operational model of an expert's skill by observing and recording traces of the behaviour and using them to create training examples for a learning program.

There are advantages, for the Child Machine, of having a mixture or representations, reasoning and learning methods. Shallow reasoning is usually fast and covers the most common cases encountered but may break when an unusual case is seen. Reasoning methods that require complex representations and inferences are computationally expensive to perform and to learn but they are more general and may work when shallow reasoning fails.

6 Conclusion

From its initial beginnings as an attempt to perform human-like learning, Machine Learning is now in danger of becoming just another branch of statistics. The majority of research done today only concerns itself with data analysis. The field has forgotten that there is more to learning than just looking for patterns in very large data sets. While this can be useful, machine learning will eventually hit the limitations of shallow representations and will have to face the problems that a human learner experiences in trying to make sense of the world over a lifetime of accumulated experience. So while we are currently preoccupied with building the world brain, eventually this effort will have to change to making the world brain into a child machine.

This essay reviewed some of the research that is needed to create a child machine and discussed open problems that are still unanswered. Indeed, some of the questions I posed are drawn from a 1991 paper [**?**] that set the program for much of our group's research since then, and will continue well into the future. In the words of Alan Turing, "We can only see a short distance ahead, but we can see plenty there that needs to be done" [6].

References

[1] Turing, A.: Computing machinery and intelligence. Mind **59**(236) (1950) 433–460

[2] Michie, D., Chambers, R.: Boxes: An experiment in adaptive control. In Dale, E., Michie, D., eds.: Machine Intelligence 2. Oliver and Boyd, Edinburgh (1968)

[3] Sammut, C.A.: Learning Concepts by Performing Experiments. Ph.d. thesis, Department of Computer Science, University of New South Wales (1981)

[4] King, R.D., Whelan, K.E., Jones, F.M., Reiser, P.G.K., Bryant, C H, Muggleton, S.H., Kell, D.B., Oliver, S.G.: Functional genomic hypothesis generation and experimentation by a robot scientist. Nature **427**(6971) (2004) 247–252

[5] Bratko, I.: Comparison of machine learning for autonomous robot discovery. In Koronacki, J., Ras, Z.W., Wierzchon, S.T., Kacprzyk, J., eds.: Advances in Machine Learning I. Volume 262 of Studies in Computational Intelligence. Springer (2010) 441–456

[6] Turing, A.: On computable numbers, with an application to the entscheidungsproblem. Proceedings of the London Mathematical Society **2**(42) (1936) 230–265

[7] Wells, H.: World brain: The idea of a permanent world encyclopaedia. In: Encyclopédie Française. (1937)

[8] Michalski, R.S.: Discovering classification rules using variable valued logic system vl1. In: Third International Joint Conference on Artificial Intelligence. (1973) 162–172

[9] Quinlan, J.R.: Discovering rules by induction from large collections of examples. In Michie, D., ed.: Expert Systems in the Micro-Electronic Age, Edinburgh (1979)

[10] Grobelnik, M., Mladenic, D.: Automated knowledge discovery in advanced knowledge management. Journal of knowledge management (9) (2005) 132–149

[11] Brown, S., Sammut, C.: A relational approach to tool-use learning in robots. In: International Conference on Inductive Logic Programming, Dubrovnik (September 2012)

[12] Bratko, I.: Discovery of abstract concepts by a robot. In Pfahringer, B., Holmes, G., Hoffmann, A.G., eds.: International Conference on Discovery Science. Volume 6332 of Springer Lecture Notes in Computer Science. (2010) 372–379

[13] Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka, Jr, E.R., Mitchell, T.M.: Toward an architecture for never-ending language learning. In: Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI 2010). (2010)

[14] Muggleton, S., Lin, D., Pahlavi, D., Tamaddoni-Nezhad, A.: Meta-interpretive learning: application to grammatical inference. Machine Learning (forthcoming)

[15] Mitchell, T.M., Utgoff, P.E., Banerji, R.B.: Learning by experimentation: Acquiring and refining problem-solving heuristics. In Michalski, R., Carbonell, J., Mitchell, T., eds.: Machine Learning: An Artificial Intelligence Approach. Tioga, Palo Alto (1983)

[16] Banerji, R.B.: A language for the description of concepts. General Systems **9** (1964) 135–141

[17] Cohen, B.L.: A Theory of Structural Concept Formation and Pattern Recognition. Ph.d. thesis, Department of Computer Science, University of New South Wales (1978)

[18] Sammut, C.A., Cohen, B.L.: A language for describing concepts as programs. In Tobias, J.M., ed.: Language Design and Programming Methodology. Volume 79 of Lecture Notes in Computer Science. Springer (1980) 111–116

[19] Sammut, C.A.: Concept learning by experiment. In: Seventh International Joint Conference on Artificial Intelligence, Vancouver (1981) 104–105

[20] Muggleton, S., Buntine, W.: Machine invention of first-order predicates by inverting resolution. In Michalski, R.S., Mitchell, T.M., Carbonell, J.G., eds.: Proceedings of the Fifth International Machine. Learning Conference. Morgan Kaufmann,, Ann Arbor, Michigan (1988) 339–352

[21] De Raedt, L., Bruynooghe, M.: An overview of the interactive concept-learner and theory revisor clint. In Muggleton, S., ed.: Inductive Logic Programming. Academic Press (1992) 163–191

[22] Muggleton, S.: Inductive logic programming. New Generation Computing **8** (1991) 295–318

[23] Quinlan, J.R.: Learning logical definitions from relations. Machine Learning **5** (1990) 239–266

[24] Shapiro, E.: An algorithm that infers theories from facts. In: Proceedings of the Seventh International Joint Conference on Artificial Intelligence, Vancouver, Morgan Kaufmann (1981) 446–451

[25] Richards, B., Mooney, R.: First-order theory revision. In: Proceedings of the Eighth International Machine Learning Workshop, Evanston, IL (1991) 447–451

[26] Muggleton, S.: Duce, an oracle based approach to constructive induction. In: Proceedings of the International Joint Conference on Artificial Intelligence, Milan (1987) 287–292

[27] Wrobel, S.: First order theory refinement. In: Advances in Inductive Logic programming. Volume 32. (1996) 14–33

[28] de Kleer, J.: An assumption-based tms. Artificial Intelligence **28**(1) (1986) 127–162

[29] Sammut, C.: Managing context in a conversational agent. Electronic Transactions on Artificial Intelligence **5**(B) (2001) 189–202 Full paper from Machine Intelligence 18.

[30] Weizenbaum, J.: Eliza - a computer program for the study of natural language communication between man and machine. Communications of the ACM **9**(1) (1966) 36–35

[31] Whitehead, A.: An Introduction to Mathematics. Henry Holt and Company, New York (1911)

[32] Michie, D., Bain, M., Hayes-Michie, J.: Cognitive models from subcognitive skills. In Grimble, M., McGhee, S., Mowforth, P., eds.: Knowledge-base Systems in Industrial Control. Peter Peregrinus (1990)

Alan Turing, Turing Machines and Stronger

Matjaž Gams
Department of Intelligent Systems,
Jožef Stefan Institute, Jamova cesta 39, 1000 Ljubljana, Slovenia
E-mail: matjaz.gams@ijs.si

Keywords: Alan Turing, Turing machine, Turing test, hypercomputing, computability, artificial intelligence

Received: October 23, 2012

This positional paper claims that the achievements of Alan Mathison Turing in computer science and informatics are comparable to those of Albert Einstein in physics. Turing's contributions are presented through his most important events and achievements, particularly through the concept of the hypercomputer; that is, computers that are stronger than the Universal Turing Machines. The paper analyzes several essential AI and human-intelligence concepts that Turing introduced. Part of the paper discusses Donald Michie, Alan Turing's co-worker and contemporary and an honorary member of the Jozef Stefan Institute. Even though 2012 marks a century since Turing's birth, he remains largely unknown around the world. This paper makes an appeal for Turing's full recognition and acknowledges contributions to Turing's career.

Povzetek: Prispevki Alana Turinga so predstavljeni s tezo, da je za računalništvo njegov prispevek tako pomemben kot prispevek Alberta Einsteina za fiziko.

1 Introduction

Alan Mathison Turing (23rd June 1912–7th June 1954) was a British mathematician and computer scientist. He invented a formalization of the concepts of "algorithm" and "computation" with the Turing machine, which can be considered a model of a general purpose computer. He also decoded the German Enigma machine (a corresponding book cover is presented in Figure 1).

In 2012, the centenary of Turing's birth, a number of Turing-related events were held around the world. Lectures and publications about Turing were made in Slovenia in the first half of 2012, such as [9, 10], but the most essential event was a conference in October, dedicated to Alan Turing and 20 years of Slovenian Artificial Intelligence Society, entitled "100 Years of Alan Turing and 20 years of SLAIS" (http://is.ijs.si/is/is2012). Among the world-renowned researchers who presented at the conference were Stephen Muggleton, Natasa Milic-Frayling, Albert Bifet, Claude Sammut, and Joao Gama. At the opening ceremony, Professor Dr. Ivan Bratko received an award for life-long achievements in artificial intelligence, both in Slovenia and internationally.

The following are some of the key dates related to Turing from our perspective:

1912 – Turing's birth
1936 – Creation of the Turing machine
1932–42 – Enigma decoded
1950 – Creation of AI, Turing test
1954 – Turing's death
2007 – Death of Donald Michie
2009 – Turing's official rehabilitation
2012 – Centenary of Turing's birth

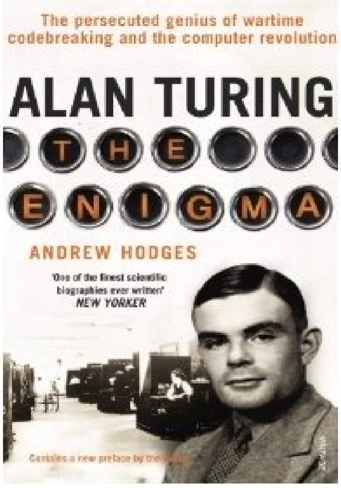

The persecuted genius of wartime codebreaking and the computer revolution
ALAN TURING
THE ENIGMA
ANDREW HODGES
'One of the finest scientific biographies ever written' NEW YORKER

Figure 1: A book about Alan Turing, published in 2012.

This paper analyzes some of these events. Section 2 discusses Turing's predictions of how to obtain machine intelligence, while Section 3 examines Turing's machine concepts. Section 4 deals with hypercomputing, and Section 5 with the Turing test. Donald Michie and Alan Turing are represented through the Slovenian connection in Section 6, before Section 7 concludes.

2 Turing's predictions about MI

In 2013, AI Communications will publish Stephen Muggleton's article entitled "Alan Turing and the development of Artificial Intelligence" [16]. The article analyzes Turing's predictions for the development of artificial intelligence through his 1950 "Mind" paper [22]. The novelty of Muggleton's article is that it examines the article published in 1950 from contemporary perspectives. Several of the issues that Turing raised in the Mind paper are as relevant and interesting as at any time. For example, Turing himself proposed three alternative possible ways to achieve artificial intelligence: by programming; by ab initio learning; and by combining several approaches such as learning, logics, probabilistic computing, and using knowledge. Turing also proposed the Child Machine, a machine that enabled learning as an infant. (Coincidentally, in 2012 neuroscientists managed to implant devices into the human brain, enabling tetraplegic patients to control binded artificial limbs, and later learned to improve performance in a way similar to how children do. See, for example, http://journals.lww.com/neurosurgery/Fulltext/2013/02000/Robotic_Arm_Control_Using_Extracellular_Action.3.aspx).

Over the last 50 years, AI researchers have tried all three of the above-mentioned approaches. Some systems like CYC (https://twitter.com/cyc_ai), which have been running for decades and are still very much alive, are based on humans imputing large amounts of knowledge, which an average human possesses, into a computer and adding several simple computing mechanisms. The other approach can be characterized as machine learning or data mining, which is currently one of the most successful areas of artificial intelligence. However, these systems clearly lack human properties and it seems that Turing was correct in denoting this approach as incapable of achieving true human-level artificial intelligence. Even programs like Nell (http://www.cs.cmu.edu/~tom/), one of the top AI self-learning programs from the Web, seems to have been saturated after successful learning of words and their relations.

The third approach is clearly the most promising for further research, both when Turing proposed it and today. As Steven Muggleton wrote in the closing remarks of his paper, AI and even Web systems like Google have significantly influenced the way we live today. Despite the major technological achievements that have occurred recently, several of Turing's ideas and discussions are as valid today as they were over 60 years ago – many lifetimes ago in information-society terms.

3 Turing machine, TM

Although Turing studied several areas related to mathematics, computer science, and artificial intelligence, his main discoveries are probably related to the Turing machine.

In 1936, Turing published a paper about the Turing machine (TM), the Universal Turing Machine (UTM), and the halting problem. He was interested in the undecidability of formal systems, as was his professor, David Hilbert. An undecidable problem is a decision problem for which it is impossible to construct a single algorithm that always leads to a correct yes/no answer. A decision problem is any arbitrary yes/no question on an infinite set of inputs.

Formally [19], a decision problem A is considered to be decidable or effectively solvable if A is a recursive set. A problem is considered to be partially decidable, semidecidable, solvable, or provable if A is a recursively enumerable set. Partially decidable problems and any other problems that are not decidable are referred to as undecidable. In computability theory, the halting problem is to decide whether the program finishes running or will run forever, given a description of a program and a finite input.

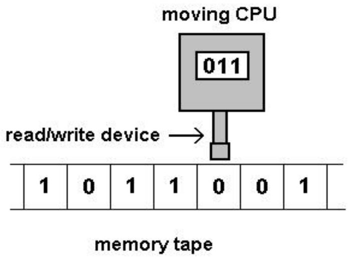


Figure 2: The Turing machine – a formal computing mechanism.

By constructing a counterexample, Alan Turing proved that a general algorithm running on a Turing machine that solves the halting problem for all possible program-input pairs cannot exist. He connected two Turing machines to each other in such a way that one stops when the other does not, creating a logically impossible situation. Therefore, since there are algorithms (programs) that no other algorithm can decide whether it will stop or not, the halting problem is undecidable for Turing machines. Another simple explanation related to algorithms is a version of the halting problem. The program:

```
while True: continue;
```

will loop forever, but the program

```
while False: continue;
```

stops very quickly. If we have a function "stops(x)" that takes a program X and returns true if X stops, we can make a program:

```
program p(x): while stops(x): Continue;
```

This program P takes program X and runs forever if X does not run forever.

What happens if we run program P on itself; i.e., X=P? This is the essential self-reference that causes the logical impossibility: P must work on every possible program, so it must be able to work on program P, so we can say P runs forever if P does not run forever. That is, by definition, not possible; hence, such a program does not exist for all possible situations.

Turing was not the first person to provide new discoveries regarding undecidability, since it was one of the most important open questions at that time. Church and Gödel presented their theorems first, but to a lesser extent than Turing. Gödel published his incompleteness theorems [18] as early as 1931. A weaker form of his first incompleteness theorem is a consequence of the undecidability of the halting problem. This weaker form differs from the standard statement of the incompleteness theorem, which means that the axiomatic system is a necessity if one wants to prove all true statements about natural numbers. There are actually two issues: provability and truth. In the first case the issue is just about proving, while in the former it is about being true but not provable. In both cases, a formal system contains statements that are true and cannot be proven, even though there are sentences (algorithms, programs, theories) that cannot be proven regardless of their truth value [1]. Turing's contribution was in proving the concept regarding computability (with regard to computing) and presenting it in algorithmic form, while Church and Gödel presented their conclusions as mathematical theorems. It was only discovered later that the concept is basically the same.

The importance of the Turing machine is that it does not consist of mathematical equations and symbols; rather, it is a formal computing mechanism, as presented in Figure 2. It consists of a simple head; that is, a simple read/write device and a tape with cells. Each cell contains just one symbol. When the head is above one cell, it can read the symbol in the cell or write a symbol into the cell. It does not matter whether the head moves or the tape, but one move is to one cell left or right. The head executes a program; that is, an algorithm that is attached to this head.

This computing mechanism represents the simplest and best known model of computing and of the digital computers we use today. Although there are disputes, some of which are looked at in this paper, the Turing machine is not importantly affected by them. For example, theoreticians are quick to point out that the Turing machine needs a tape of unlimited size in one direction (that is, an unlimited data storage equivalent), which means that computers are not true equivalents of the Turing machine, but rather a finite automata (albeit huge ones). This argument seems somewhat superficial. Surely, if the computer memory is big enough, then the difference is meaningless in all but theoretical terms. Memory capacities in computers do not represent a bottleneck in computing, so there is no need to worry about them.

In summary, the Turing machine represents such a basic computing concept, or computing principle, that it can be compared to Einstein's relativity principle in terms of importance and nature. Indeed, both are neither axioms nor theorems, but principles, one describing digital or symbolic computers and the other time-space relations. Similarly, both concepts remain valid for our current lives even though several decades have passed since they were first formulated. Both concepts are also essential for our understanding of the world – without Einstein we would not be able to understand our universe, and without Turing we would not be able to understand computers and computing.

4 Stronger than Turing machines - hypercomputing

The term "hypercomputation" was introduced by Copeland and Proudfoot in 1999 [2]. Machines are referred to as hypercomputers or super-Turing computational models if they are, in principle, stronger than the Turing machine, which means that they can solve tasks that the Turing machine cannot. The term super-Turing computing usually denotes physically realizable mechanisms. Hypercomputers and super-Turing computational models include the computation of non-Turing-computable functions, following super-recursive algorithms. Turing himself introduced stronger machines than the universal Turing machine. An example is a Turing machine that includes an oracle capable of correctly answering any question with Yes or No. However, the purpose of this section is simply to analyze machines or beings that do not need a superficial component, but are stronger than Turing machines.

Hypercomputing is related to several issues, including the question of human computing power compared to computing mechanism and computers. Consider, for example, the Church-Turing thesis [24]. It states that any function that is algorithmically computable can be computed by a Turing machine. Hypercomputers compute functions that a Turing machine cannot, which means that they are not computable in the Church-Turing sense. Some publications have indicated that no physical machine can be designed to surpass the Turing machines and that it is not possible to construct a counterproof. In other words, the hypercomputer ideas could be hypothetical and physically non-existent. In principle, however, there is no proof that hypercomputers are impossible in mental computational issues, just because they are not physically realizable. Accepting philosophical viewpoints such as dualism is sufficient for hypercomputers to become theoretically possible, if ever a concrete version were not designed in the form of a computing machine.

Several authors, such as Roger Penrose [17], have opposed the notion that computers are as powerful as digital computers, directly indicating that humans either possess stronger mechanisms than Turing machines or mechanisms that, while not in principle stronger, are so different in practical computer terms that computers cannot compete with them under real-life circumstances due to huge differences in complexity. In other words, the human brain is either stronger in principle or in reality for most real-world tasks.

The halting problem is one problem that a Turing machine cannot solve. Some hypercomputers can simulate a program for an infinite number of steps and tell the user whether or not the program halted. Some authors have claimed that the halting problem can be solved by humans even without using the additional information that is known to make it possible to solve the halting problem.

Therefore, the halting problem can be solved, even by a Turing machine, if additional information is provided; however, in the same circumstances only hypercomputing mechanisms and humans can solve the problem. Therefore, humans are computationally stronger than computers.

For a hypercomputing model, the author of the present paper introduced the Multiple Turing Machine, as presented in Figure 3. Unlike the multi-tape Turing machine, this model consists of two universal Turing machines, each of which writes on each other program, while at the same time obtaining information from the open outside world. The model is based on the principle of multiple knowledge [5]. The weak version of this principle states that a sensibly integrated model (or computing mechanism) generally outperforms each single model constituting the basic set of models. The strong version of the principle states that real or human-level intelligence can be achieved only when using the multiple algorithms that apply the principle. Current computer mechanism in digital computers cannot provide the multiple computations. This means that current computers are not as strong as humans in principle, although they are much faster at computing single operations and tasks that do not demand multiple computations. It is important to note that multiplicity may or may not include parallelism; rather, it resonates the interaction concept [23]. The principle of multiple knowledge has several representations and confirmations. In terms of the physical world, it demonstrates a strong similarity to the multiple-worlds theory [3] or the multiverse theory [4]. In both theories, there are a huge number of universes like ours in the super-universe. The open question remains where these universes are. Are they physical or just mental? If they are physical, where are they exactly? Besides the evidently non-problematic mental existence, there are also theories related to the physical existence of multiple worlds.

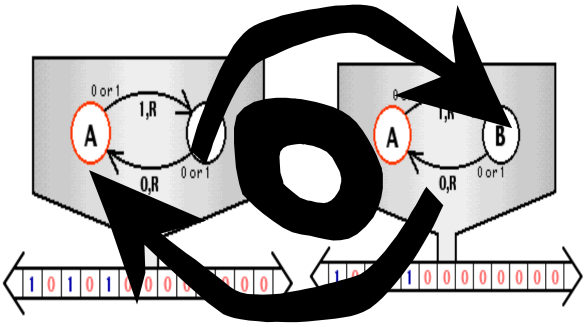


Figure 3: The Multiple Turing machine.

The controversial multiverse theory is based on physical observations that our universe is expanding at an increasingly fast rate, despite there not being enough matter or energy of any kind to support such an increase. However, the existence of multiple coexisting universes would explain the gravity that causes the expansion.

Whatever fascinating these new ideas that enable other kinds of computation, the TM the best corresponds to digital computers while there is no existing hypercomputer in our physical world and no super-universe has yet been confirmed. It should also be noted that for some kinds of computation we now understand the differences compared to the Turing computation. For example, quantum computers do note compute based on 0/1, but on a quantum superposition of the two; that is, in an essentially different computing way. However, while David Deutsch himself originally claimed, around 1988, that quantum computers have a computational power beyond UTMs, they are presently not considered that way, even by Deutsch, who now views them as a way to reduce cryptographic problems from the NP complexity class to the P class (see Shor's result on NP complexity in quantum computing).

The Turing machine and the halting machine are as fundamental concepts and principles as the other principles humans use, such as that of Einstein, and have not changed over time.

5 Turing test

In 1950, Turing published a paper describing the Turing test and a debate about artificial intelligence, which at that time was referred to as "machine intelligence". It was only in 1956 that McCarthy coined the term "artificial intelligence" term and the field started growing.

Regarding the Turing test (Turing 1950), the best known and discussed AI test of all times, the issue is in comparing behaviors of two computing mechanisms (originally one human and one computer) and identifying the computing type of each. There are dozens of different versions of the Turing test, such as a Total Turing Test [12], which includes physical tasks, and a Trully Total Turing test [20], which examines populations of test subjects.

In terms of the latest definitions [8], intelligent systems and intelligence are defined with the ability to learn. As a consequence, all machine learning systems are intelligent and every living being is intelligent including bacteria and plants. However, the Turing test deals with human-level intelligence, which includes testing at that level.

In practical terms, although computers have improved their performance by a factor of 100,000 over the last 20 years, human interrogators separate computers from humans in as many questions as before two decades. The reason is that current computers lack any human-level understanding. By verifying the understanding of any sentence – that is, its semantics – all computer programs display tabula-rasa performance. This is the empirical argument of weak AI, which claims that computers need major improvements in order to approach human-level intelligence and computing [5, 6].

The relation between computers and humans is presented in Figure 4. Computers progress exponentially according to Moore's law. In Figure 4, the x axis is linear and the y axis is logarithmic; therefore, the progress of computers is presented by a line. On the other hand, humans, in terms of their hardware, remain the same as they were hundreds of years ago. Due to the huge difference in the speed of progress, computers have bypassed humans in an increasing number of areas. For example, computers can beat humans in practically all formal mental games, with only a few exceptions. In 1997, Garry Kasparov lost to the Big Blue IBM computer, while in 2011 another IBM program, named Watson, beat two human world champions in "Jeopardy," an association knowledge game.

Today, however, humans use computers as tools, analogous to forks, chairs, or cars, not as beings, due to their lack of personality or self-awareness. Computers are purely calculating and information/retrieval tools that enable several services like web or social interaction. Humans using tools can perform better in tasks such as flying an airplane. Computers help us solve more complex information and calculation tasks; therefore our capability to solve problems dramatically increases due to the progress of computers, as represented in the upper line in Figure 4. Furthermore, this line helps understand the Flynn phenomenon of how each generation achieves higher IQ scores than the last. It is mainly due to the improved computers, which have a positive reverse effect on humans. As far as the author can ascertain, this explanation is the first of this kind of the Flynn phenomenon.

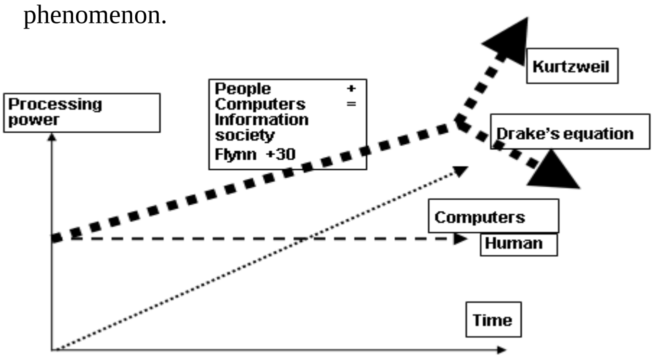


Figure 4: Modern viewpoint on relation between computers and humans.

Where will this progress lead us? According to optimists like Ray Kurtzweil, who was recently employed by Google, the constant progress of human civilization will lead to a qualitative leap forward. According to pessimists, humans are facing dire times, perhaps because of several reasons, such as overpopulation, global heating, and a lack of natural resources. Another grim prospect of human civilization comes from the Drake equation, which says that, based on the number of stars in the universe, there should be several civilizations with which ours should have already made contact. However, these are only possible negative predictions that need not happen.

Like Turing, who predicted true artificial intelligence (albeit by 2000), most scientists agree that true intelligent computers will emerge sooner or later. When that happens, human civilization will indeed leap forward.

6 Alan Turing, Donald Michie, and Slovenia

Slovenia has a strong connection to Alan Turing and his companion Donald Michie. Michie and Turing worked together to decode Enigma, the German encryption machine. By use of electronic machines, the counter-surveillance department where Michie and Turing worked was able to decode messages to German submarines. This has been described as the most relevant civil discovery during the Second World War and that it saved hundreds of thousands of lives. Because it was a secret, not much was known about this issue until decades later when the data became public.

Alan Turing died committing suicide with a poisoned apple, in an analogy to the Cinderella story. His death and sexual behavior (homosexuality was at that time treated as a risk to national security for anyone employed in governmental services and was therefore prosecuted) led to Turing's legal prosecution. Today, this would be considered a major injustice to a loyal and honorable citizen, and his conviction was only reversed in 2009.

Unlike Turing, Michie [21] went on to have a long and successful career in artificial intelligence. He was also the first person to establish an AI department in any institution in the world. Michie was awarded several established prizes, including the Feigenbaum Medal and the IJCAI award. Michie is especially important for Slovenian AI and the Slovenian society SLAIS due to his cooperation with Ivan Bratko. In recent decades, Michie has often spent one month each year at the Jozef Stefan Institute, in Ivan Bratko's room. The room is now named the Donald Michie room and is close to the central lecture hall at the institute. Ivan Bratko also spent a lot of time at the Turing Institute in UK, where several Slovenian researchers were invited for short and long periods.

Ivan Bratko, who has been active in many AI areas including machine learning [13], was the head of the AI department at the Jozef Stefan Institute. Due to the successful growth of the field, three departments emerged from the original one. Ivan Bratko is still the scientific head of the Intelligent Systems department. As well as the Donald Michie room, there is also an Alan Turing room, the office of the author of this paper, who in 2010 [7] wrote: "Let this be in memory of Donald Michie as Turing's contemporary and our dear colleague, and the extreme genius Alan Turing, who marked the lives of every human in the world as hardly anybody else has". The contributions of Turing and Michie were presented on several other occasions as well [8, 9, 10, 11].

Stephen Muggleton [16], Ivan Bratko, and the author of the present paper established an award named the Michie–Turing Award for life achievements in information society in Slovenia. This is our humble tribute to the great names that will resonate in Slovenia for a long time to come. At this opportunity, we wish to thank the living relatives of Donald Michie and Alan Turing, who agreed to the naming of the award. An international board will supervise the nominations to guarantee that the award recipients will comply with the desired high criteria. More about the award can be found at http://is.ijs.si/is/is2013/nagrade_is_eng.asp?lang=eng.

The Michie–Turing award was defined during the Alan Turing Conference in 2012 [14] in Slovenia and will be presented for the first time in 2013.

7 Discussion

Entering "Lady Gaga" into the Google search engine produces 400 million hits, compared to 132 million for "Mozart", 60 million for "Albert Einstein" and just 9 million for "Alan Turing". Amazon.com contains approximately 10,000 book references for "Albert Einstein", compared to 1700 for "Alan Turing". Yet, many scientists, including the author of the present paper, consider the Turing Award to be the computer science equivalent of the Nobel Prize, and Turing himself as the "Einstein of computer science". Today, several publications refer to Turing as the father of computer science, artificial intelligence, and mathematical biology.

Although it might not be logical to compare the fame of Lady Gaga to that of Turing, why is Turing so unrecognized compared to such geniuses as Einstein or Mozart? What good is it that Turing is widely recognized in computer science and informatics, if the average European has not heard of him but has heard of Einstein or Mozart? The fact that Turing was neglected 60 years ago does not change the fact that he is still neglected now, as criteria such as internet hits indicate.

It should be the responsibility of all scientists, not just the computer science community, but on all scientists to revive the fallen acknowledgement of an extraordinary scientist who was ruined by intolerant bureaucrats. It is only fair that the world accepts Turing as one of one of the most important scientists. Anyone who doubts such a claim should just look around and count the Turing machines embedded in nearby machines or read his seminal works.

We should also remember Donald Michie as Turing's companion on the list of computer geniuses who changed the world.

8 References

[1] D. Bojadžijev, M. Gams. Addendum to "Sloman's view of Gödel's sentence". *Artificial intelligence* 98, 1998.

[2] B. Copeland, D. Proudfoot. Alan Turing's forgotten ideas in computer science. *Scientific American,* April 1999.

[3] D. Deutsch. The Fabric of Reality: The Science of Parallel Universes – and Its Implications, Penguin books, 1998.

[4] G. Ellis. Does the Multiverse Really Exist? *Scientific American* 305 (2), 2011.

[5] M. Gams. Weak intelligence: through the principle and paradox of multiple knowledge. *Nova Science*, 2001.

[6] M. Gams. The Turing machine may not be the universal machine. *Minds and Machines*, 2002.

[7] M. Gams. Turing – the genius that influenced lives of all humans on the Earth (in Slovene). Delo, Sobotna pril., 3. jul. 2010, letn. 52, št. 151, str. 26–27.

[8] M. Gams. Natural and artificial intelligence. Video lecture. (In Slovene), 2011 http://videolectures.net/solomon_gams_inteligenca/

[9] M. Gams. Turing, computer Einstein. Videlo lecture. (In Slovene), 2012. http://videolectures.net/kolokviji_gams_turing/

[10] M. Gams. Alan M. Turing, the inventor of the universal computing machine (in Slovene). Organizacija znanja, 2, 2012.

[11] M. Gams. Alan Turing – Einstein of computer science. Proc. of the 15th Int. conf. Information society IS 2012, 8.-12. 2012, Ljubljana, Slovenia, 2012.

[12] S. Harnad. Other bodies, other minds: A machine incarnation of an old philosophical problem. *Minds and Machines*, 1, 1991.

[13] R.S. Michalski, I. Bratko, M. Kubat. Machine Learning and Data Mining: Methods and Applications. Wiley, 1998.

[14] D. Mladenic, M. Bohanec (eds.). 100 Years of Alan Turing and 20 Years of SLAIS. Conf. Proceedings, part of the Proc. of the 15th Int. conf. Information society IS 2012, October 8–12, 2012, Ljubljana, Slovenia, 2012.

[15] S. Muggleton. Artificial Intelligence With Donald Michie. In D. Mladenic, M. Bohanec (eds.). 100 Years of Alan Turing and 20 Years of SLAIS. Conf. Proceedings, part of the Proc. of the 15th Int. conf. Information society IS 2012, October 8–12, 2012, Ljubljana, Slovenia, 2012.

[16] S. Muggleton. Alan Turing and the development of Artificial Intelligence. *AI communications*, forthcoming, 2013.

[17] R. Penrose. The Emperor's New Mind: Concerning Computers, Minds, and the Laws of Physics, Oxford, 2002.

[18] G. Raguni. The Gödel's legacy: revisiting the Logic. Amazon Digital Services, Inc., 2012.

[19] E. Reiter, C. Johnson. Limits of Computation: An Introduction to the Undecidable and the Intractable. Chapman and Hall, 2012.

[20] P. Schweizer (1998). The Truly Total Turing Test. *Minds and Machines* 8 (2):263–272.

[21] A. Srinivasan, D. Michie: Machine intelligence, biology and more. Oxford, 2009.

[22] A. Turing. Computing Machinery and Intelligence. Mind, 1950.

[23] P. Wegner. Why interaction is more powerful than algorithms. *Communications of the ACM* 40, 5, 1997.

[24] Wikipedia. Philosophy of computer science: Philosophy of artificial intelligence, Church-Turing thesis, Technological singularity, Chinese Room. 2012.

Mining Big Data in Real Time

Albert Bifet
Yahoo! Research Barcelona
Avinguda Diagonal 177, 8th floor
Barcelona, 08018, Catalonia, Spain
E-mail: abifet@yahoo-inc.com

Keywords: big data, data streams, data mining

Received: December 15, 2012

Streaming data analysis in real time is becoming the fastest and most efficient way to obtain useful knowledge from what is happening now, allowing organizations to react quickly when problems appear or to detect new trends helping to improve their performance. Evolving data streams are contributing to the growth of data created over the last few years. We are creating the same quantity of data every two days, as we created from the dawn of time up until 2003. Evolving data streams methods are becoming a low-cost, green methodology for real time online prediction and analysis. We discuss the current and future trends of mining evolving data streams, and the challenges that the field will have to overcome during the next years.

Povzetek: Prispevek opisuje rudarjenje velikih količin podatkov na osnovi pretakanja podatkov v podjetjih.

1 Introduction

Nowadays, the quantity of data that is created every two days is estimated to be 5 exabytes. This amount of data is similar to the amount of data created from the dawn of time up until 2003. Moreover, it was estimated that 2007 was the first year in which it was not possible to store all the data that we are producing. This massive amount of data opens new challenging discovery tasks.

Data stream real time analytics are needed to manage the data currently generated, at an ever increasing rate, from such applications as: sensor networks, measurements in network monitoring and traffic management, log records or click-streams in web exploring, manufacturing processes, call detail records, email, blogging, twitter posts and others [17]. In fact, all data generated can be considered as streaming data or as a snapshot of streaming data, since it is obtained from an interval of time.

In the data stream model, data arrive at high speed, and algorithms that process them must do so under very strict constraints of space and time. Consequently, data streams pose several challenges for data mining algorithm design. First, algorithms must make use of limited resources (time and memory). Second, they must deal with data whose nature or distribution changes over time.

We need to deal with resources in an efficient and low-cost way. *Green computing* is the study and practice of using computing resources efficiently. A main approach to green computing is based on algorithmic efficiency. In data stream mining, we are interested in three main dimensions:

- accuracy
- amount of space (computer memory) necessary
- the time required to learn from training examples and to predict

These dimensions are typically interdependent: adjusting the time and space used by an algorithm can influence accuracy. By storing more pre-computed information, such as look up tables, an algorithm can run faster at the expense of space. An algorithm can also run faster by processing less information, either by stopping early or storing less, thus having less data to process. The more time an algorithm has, the more likely it is that accuracy can be increased.

The issue of the measurement of three evaluation dimensions simultaneously has led to another important issue in data stream mining, namely estimating the combined cost of performing the learning and prediction processes in terms of time and memory. As an example, several rental cost options exist:

- Cost per hour of usage: Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud. Cost depends on the time and on the machine rented (small instance with 1.7 GB, large with 7.5 GB or extra large with 15 GB).
- Cost per hour and memory used: GoGrid is a web service similar to Amazon EC2, but it charges by RAM-Hours. Every GB of RAM deployed for 1 hour equals one RAM-Hour.

In [11, 9] the use of RAM-Hours was introduced as an evaluation measure of the resources used by streaming algorithms. Every GB of RAM deployed for 1 hour equals one RAM-Hour.

The structure of this paper is as follows: Section 2 introduces Big Data, Section 3 shows some open source tools, Sections 5, 6, and 7 discuss future trends and introduce a new problem, application and technique for real time Big Data mining, and finally Section 8 concludes the paper.

2 Big data

Big Data is a new term used to identify the datasets that due to their large size, we can not manage them with the typical data mining software tools. Instead of defining "Big Data" as datasets of a concrete large size, for example in the order of magnitude of petabytes, the definition is related to the fact that the dataset is too big to be managed without using new algorithms or technologies.

The McKinsey Global Institute (MGI) published a report on Big Data [26] that describes the business oportunities that big data opens: a potential value of $300 billion in the US health care, $149 billion in European governement administration or improving the operating margin of retailer companies by 60 percent.

Big Data analytics is becoming an important tool to improve efficiency and quality in organizations, and its importance is going to increase in the next years.

There are two main strategies for dealing with big data: sampling and using distributed systems. Sampling is based in the fact that if the dataset is too large and we can not use all the examples, we can obtain an approximate solution using a subset of the examples. A good sampling method will try to select the best instances, to have a good performance using a small quantity of memory and time.

An alternative to sampling is the use of probabilistic techniques. Backstrom, Boldi, Rosa, Ugander and Vigna [7] computed the average distance of friendship links between users in Facebook. They repeated the experiment that Stanley Milgram did in 1967 [28], where he challenged people to send postcards to specific people around US using only direct acquaintances. Milgram obtained a number between 4.4 and 5.7, so the notion of six degrees of separation was confirmed. The experiments using Facebook showed a four degrees of separation pattern. To run these experiments, these researchers used HyperANF, a software tool by Boldi, Rosa and Vigna [13] that improved ANF. ANF is a fast and scalable tool for data mining in massive graphs [30] that computes approximations to the neighbourhood function of nodes in massive graphs. The *neighbourhood function* of a node n and distance h is defined as the number of nodes at a certain distance h reachable from node n. This function is computed using the set of nodes that can be reachable from a distance $h - 1$ using probabilistic counter data structures.

The number of users of Facebook is more than 800 million users, but they managed to compute the average distance between two users on Facebook only using one machine. As one of the authors of this paper, Paolo Boldi, once said "Big data does not need big machines, it needs big intelligence".

The most popular distributed systems used nowadays are based in the *map-reduce* framework. The map-reduce methodology started in Google, as a way to perform crawling of the web in a faster way. Hadoop is a open-source implementation of map-reduce started in Yahoo! and is being used in many non-streaming big data analysis.

The map-reduce model divides algorithms in two main steps: map and reduce, inspired in ideas in functional programming. The input data is split into several datasets and each split is send to a mapper, that will transform the data. The output of the mappers will be combined in reducers, that will produce the final output of the algorithm.

Nowadays, in business, more than size and scale, what it is important is speed and agility. As David Meerman Scott explains in his book "Real-Time Marketing & PR" [33], it is important for companies to know what is happening right now, in real time, to be able to react, and more important, to anticipate and detect new business opportunities.

Finally, we would like to mention the work that Global Pulse is doing [37] using Big Data to improve life in developing countries. Global Pulse is a United Nations initiative, launched in 2009, that functions as an innovative lab, and that is based in mining Big Data for developing countries. They pursue a strategy that consists of 1) researching innovative methods and techniques for analyzing real-time digital data to detect early emerging vulnerabilities; 2) assembling free and open source technology toolkit for analyzing real-time data and sharing hypotheses; and 3) establishing an integrated, global network of Pulse Labs, to pilot the approach at country level.

Global Pulse describe the main opportunities Big Data offers to developing countries in their White paper "Big Data for Development: Challenges & Opportunities"[22]:

- Early warning: develop fast response in time of crisis, detecting anomalies in the usage of digital media
- Real-time awareness: design programs and policies with a more fine-grained representation of reality
- Real-time feedback: check what policies and programs fails, monitoring it in real time, and using this feedback make the needed changes

The Big Data mining revolution is not restricted to the industrialized world, as mobiles are spreading in developing countries as well. It is estimated than there are over five billion mobile phones, and that 80% are located in developing countries.

3 Tools: open source revolution

The Big Data phenomenon is intrinsically related to the open source software. Large companies as Facebook, Yahoo!, Twitter, LinkedIn benefit and contribute working on open source projects. Big Data infrastructure deals with Hadoop, and other related software as:

- Apache Hadoop [3]: software for data-intensive distributed applications, based in the MapReduce programming model and a distributed file system called Hadoop Distributed Filesystem (HDFS). Hadoop allows writing applications that rapidly process large amounts of data in parallel on large clusters of compute nodes. A MapReduce job divides the input dataset into independent subsets that are processed by map tasks in parallel. This step of mapping is then followed by a step of reducing tasks. These reduce tasks use the output of the maps to obtain the final result of the job.
- Apache Pig [6]: software for analyzing large data sets that consists of a high-level language similar to SQL for expressing data analysis programs, coupled with infrastructure for evaluating these programs. It contains a compiler that produces sequences of MapReduce programs.
- Cascading [15]: software abstraction layer for Hadoop, intended to hide the underlying complexity of MapReduce jobs. Cascading allows users to create and execute data processing workflows on Hadoop clusters using any JVM-based language.
- Scribe [16]: server software developed by Facebook and released in 2008. It is intended for aggregating log data streamed in real time from a large number of servers.
- Apache HBase [4]: non-relational columnar distributed database designed to run on top of Hadoop Distributed Filesystem (HDFS). It is written in Java and modeled after Google's BigTable. HBase is an example if a NoSQL data store.
- Apache Cassandra [2]: another open source distributed database management system developed by Facebook. Cassandra is used by Netflix, which uses Cassandra as the back-end database for its streaming services.
- Apache S4 [29]: platform for processing continuous data streams. S4 is designed specifically for managing data streams. S4 apps are designed combining streams and processing elements in real time.
- Storm [34]: software for streaming data-intensive distributed applications, similar to S4, and developed by Nathan Marz at Twitter.

In Big Data Mining, there are many open source initiatives. The most popular are the following:

- Apache Mahout [5]: Scalable machine learning and data mining open source software based mainly in Hadoop. It has implementations of a wide range of machine learning and data mining algorithms: clustering, classification, collaborative filtering and frequent pattern mining.
- MOA [9]: Stream data mining open source software to perform data mining in real time. It has implementations of classification, regression, clustering and frequent item set mining and frequent graph mining. It started as a project of the Machine Learning group of University of Waikato, New Zealand, famous for the WEKA software. The `streams` framework [12] provides an environment for defining and running stream processes using simple XML based definitions and is able to use MOA.
- R [32]: open source programming language and software environment designed for statistical computing and visualization. R was designed by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand beginning in 1993 and is used for statistical analysis of very large data sets.
- Vowpal Wabbit [21]: open source project started at Yahoo! Research and continuing at Microsoft Research to design a fast, scalable, useful learning algorithm. VW is able to learn from terafeature datasets. It can exceed the throughput of any single machine network interface when doing linear learning, via parallel learning.
- PEGASUS [19]: big graph mining system built on top of MAPREDUCE. It allows to find patterns and anomalies in massive real-world graphs.
- GraphLab [25]: high-level graph-parallel system built without using MAPREDUCE. GraphLab computes over dependent records which are stored as vertices in a large distributed data-graph. Algorithms in GraphLab are expressed as vertex-programs which are executed in parallel on each vertex and can interact with neighboring vertices.

4 Big data mining academic and industrial research

There are many interesting ongoing academic and industrial research in the area, published in main conferences such as KDD, ICDM, ECML-PKDD, or journals such as “Data Mining and Knowledge Discovery", “Machine Learning" or “Journal of Machine Learning Research”. Some examples are the following, published in the SIGKDD Explorations Issue of December 2012:

- Mining Heterogeneous Information Networks: A Structural Analysis Approach by *Yizhou Sun and Jiawei Han (University of Illinois at Urbana-Champaign)* [35].

- Big Graph Mining: Algorithms and discoveries by *U Kang and Christos Faloutsos (Carnegie Mellon University)* [20].

- Scaling Big Data Mining Infrastructure: The Twitter Experience by *Jimmy Lin and Dmitriy Ryaboy (Twitter,Inc.)* [23].

- Mining Large Streams of User Data for Personalized Recommendations by *Xavier Amatriain (Netflix)* [1].

5 New problems: structured classification

A new important and challenging task may be the structured pattern classification problem. *Patterns* are elements of (possibly infinite) sets endowed with a partial order relation $\preceq$. Examples of patterns are itemsets, sequences, trees and graphs.

The structured pattern classification problem is defined as follows. A set of examples of the form (t, y) is given, where y is a discrete class label and t is a pattern. The goal is to produce from these examples a model $\hat{y} = f(t)$ that will predict the classes y of future pattern examples

Most standard classification methods can only deal with vector data, which is but one of many possible pattern structures. To apply them to other types of patterns, such as graphs, we can use the following approach: we convert the pattern classification problem into a vector classification learning task, transforming patterns into vectors of attributes. Each attribute denotes the presence or absence of particular subpatterns, and we create attributes for all frequent subpatterns, or for a subset of these.

As the number of frequent subpatterns may be very large, we may perform a feature selection process, selecting a subset of these frequent subpatterns, maintaining exactly or approximately the same information.

The structured output classification problem is even more challenging and is defined as follows. A set of examples of the form (t, y) is given, where t and y are patterns. The goal is to produce from these examples a pattern model $\hat{y} = f(t)$ that will predict the patterns y of future pattern examples. A way to deal with a structured output classification problem is to convert it to a multi-label classification problem, where the output pattern y is converted into a set of labels representing a subset of its frequents subpatterns.

Therefore, data stream multi-label classification methods may offer a solution to the structured output classification problem.

This problem has been studied in the non-streaming setting using relational learning techniques, and has been well developed within inductive logic programming and statistical relational learning [18].

6 New applications: social networks

A future trend in mining evolving data streams will be how to analyze data from social networks and micro-blogging applications such as Twitter. Micro-blogs and Twitter data follow the data stream model. Twitter data arrive at high speed, and algorithms that process them must do so under very strict constraints of space and time.

The main Twitter data stream that provides all messages from every user in real-time is called Firehose and was made available to developers in 2010. This streaming data opens new challenging knowledge discovery issues. In April 2010, Twitter had 106 million registered users, and 180 million unique visitors every month. New users were signing up at a rate of 300,000 per day. Twitter's search engine received around 600 million search queries per day, and Twitter received a total of 3 billion requests a day via its API. It could not be clearer in this application domain that to deal with this amount and rate of data, streaming techniques are needed.

Sentiment analysis can be cast as a classification problem where the task is to classify messages into two categories depending on whether they convey positive or negative feelings. See [31] for a survey of sentiment analysis, and [24] for opinion mining techniques.

To build classifiers for sentiment analysis, we need to collect training data so that we can apply appropriate learning algorithms. Labeling tweets manually as positive or negative is a laborious and expensive, if not impossible, task. However, a significant advantage of Twitter data is that many tweets have author-provided sentiment indicators: changing sentiment is implicit in the use of various types of emoticons. *Smileys* or *emoticons* are visual cues that are associated with emotional states. They are constructed using the characters available on a standard keyboard, representing a facial expression of emotion. Hence we may use these to label our training data.

When the author of a tweet uses an emoticon, they are annotating their own text with an emotional state. Such annotated tweets can be used to train a sentiment classifier [8, 10].

Another interesting application is the NELL (Never-Ending Language Learner) system developed by the group of Tom Mitchell [14, 36] at Carnegie Mellon University. The goal of this system is to build a never-ending machine learning system that acquires the ability to extract structured information from unstructured web pages. It involves text analysis of 500 million web pages and access to the remainder of the web through search engine APIs. NELL runs 24 hours per day, continuously, to perform two ongoing tasks: extract new instances of categories and relations, and learn to read better than yesterday.

7 New techniques: Hadoop, S4 or Storm

A way to speed up the mining of streaming learners is to distribute the training process onto several machines. Hadoop MapReduce is a programming model and software framework for writing applications that rapidly process vast amounts of data in parallel on large clusters of compute nodes.

A MapReduce job divides the input dataset into independent subsets that are processed by map tasks in parallel. This step of mapping is then followed by a step of reducing tasks. These reduce tasks use the output of the maps to obtain the final result of the job.

Apache S4 [29] is a platform for processing continuous data streams. S4 is designed specifically for managing data streams. S4 apps are designed combining streams and processing elements in real time. Storm [34] from Twitter uses a similar approach. Ensemble learning classifiers are easier to scale and parallelize than single classifier methods. They are the first, most obvious, candidate methods to implement using parallel techniques.

It is not clear yet how an optimal architecture for analytics systems may be to deal with historic data and with real-time data at the same time. An interesting proposal is the Lambda architecture of Nathan Marz [27]. The Lambda Architecture solves the problem of computing arbitrary functions on arbitrary data in realtime by decomposing the problem into three layers: the batch layer, the serving layer, and the speed layer. It combines in the same system Hadoop for the batch layer, and Storm for the speed layer. The properties of the system are: robust and fault tolerant, scalable, general, extensible, allows ad hoc queries, minimal maintenance, and debuggable.

8 Conclusions

We have discussed the challenges that in our opinion, mining evolving data streams will have to deal during the next years. We have outlined new areas for research. These include structured classification and associated application areas as social networks.

Our ability to handle many exabytes of data across many application areas in the future will be crucially dependent on the existence of a rich variety of datasets, techniques and software frameworks. There is no doubt that data stream mining offers many challenges and equally many opportunites as the quantity of data generated in real time is going to continue growing.

References

[1] X. Amatriain. Mining large streams of user data for personalized recommendations. *SIGKDD Explorations*, 14(2), 2012.

[2] Apache Cassandra, http://cassandra. apache.org.

[3] Apache Hadoop, http://hadoop.apache.org.

[4] Apache HBase, http://hbase.apache.org.

[5] Apache Mahout, http://mahout.apache.org.

[6] Apache Pig, http://www.pig.apache.org/.

[7] L. Backstrom, P. Boldi, M. Rosa, J. Ugander, and S. Vigna. Four Degrees of Separation. *CoRR*, abs/1111.4570, 2011.

[8] A. Bifet and E. Frank. Sentiment knowledge discovery in Twitter streaming data. In *Proc 13th International Conference on Discovery Science*, Canberra, Australia, pages 1–15. Springer, 2010.

[9] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer. MOA: Massive Online Analysis http://moa.cms.waikato.ac.nz/. *Journal of Machine Learning Research (JMLR)*, 2010.

[10] A. Bifet, G. Holmes, and B. Pfahringer. Moa-tweetreader: Real-time analysis in twitter streaming data. In *Discovery Science*, pages 46–60, 2011.

[11] A. Bifet, G. Holmes, B. Pfahringer, and E. Frank. Fast perceptron decision tree learning from evolving data streams. In *PAKDD*, 2010.

[12] C. Bockermann and H. Blom. The streams Framework. Technical Report 5, TU Dortmund University, 12 2012.

[13] P. Boldi, M. Rosa, and S. Vigna. HyperANF: approximating the neighbourhood function of very large graphs on a budget. In *Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28 - April 1, 2011*, pages 625–634, 2011.

[14] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. H. Jr., and T. M. Mitchell. Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI 2010)*, 2010.

[15] Cascading, http://www.cascading.org/.

[16] Facebook Scribe, https://github.com/ facebook/scribe.

[17] J. Gama. *Knowledge discovery from data streams*. Chapman & Hall/CRC, 2010.

[18] L. Getoor and B. Taskar. *Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2007.

[19] U. Kang, D. H. Chau, and C. Faloutsos. PEGASUS: Mining Billion-Scale Graphs in the Cloud. 2012.

[20] U. Kang and C. Faloutsos. Big graph mining: Algorithms and discoveries. *SIGKDD Explorations*, 14(2), 2012.

[21] J. Langford. Vowpal Wabbit, http://hunch.net/˜vw/, 2011.

[22] E. Letouzé. Big Data for Development: Opportunities & Challenges. May 2011.

[23] J. Lin and D. Ryaboy. Scaling big data mining infrastructure: The twitter experience. *SIGKDD Explorations*, 14(2), 2012.

[24] B. Liu. *Web data mining; Exploring hyperlinks, contents, and usage data*. Springer, 2006.

[25] Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. M. Hellerstein. Graphlab: A new parallel framework for machine learning. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, Catalina Island, California, July 2010.

[26] J. Manyika, M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh, and A. H. Byers. Big data: The next frontier for innovation, competition, and productivity. May 2011.

[27] N. Marz and J. Warren. *Big Data: Principles and best practices of scalable realtime data systems*. Manning Publications, 2013.

[28] S. Milgram. The small world problem. *Psychology Today*, 2:60–67, 1967.

[29] L. Neumeyer, B. Robbins, A. Nair, and A. Kesari. S4: Distributed Stream Computing Platform. In *ICDM Workshops*, pages 170–177, 2010.

[30] C. R. Palmer, P. B. Gibbons, and C. Faloutsos. ANF: a fast and scalable tool for data mining in massive graphs. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 23-26, 2002, Edmonton, Alberta, Canada*, pages 81–90, 2002.

[31] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2008.

[32] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2012. ISBN 3-900051-07-0.

[33] D. M. Scott. *Real-Time Marketing and PR, Revised: How to Instantly Engage Your Market, Connect with Customers, and Create Products that Grow Your Business Now*. Wiley Desktop Editions Series. John Wiley & Sons, 2011.

[34] Storm, http://storm-project.net.

[35] Y. Sun and J. Han. Mining heterogeneous information networks: A structural analysis approach. *SIGKDD Explorations*, 14(2), 2012.

[36] P. P. Talukdar, D. Wijaya, and T. Mitchell. Coupled temporal scoping of relational facts. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining (WSDM)*, Seattle, Washington, USA, February 2012. Association for Computing Machinery.

[37] United Nations Global Pulse, http://www.unglobalpulse.org.

Data Stream Mining: the Bounded Rationality

João Gama
LIAAD-INESC TEC, and FEP, University of Porto
R. Ceuta 118-6, 4050-190 Porto, Portugal
E-mail: jgama@fep.up.pt

Keywords: data stream mining, bounded rationality

Received: October 12, 2012

The developments of information and communication technologies dramatically change the data collection and processing methods. Data mining is now moving to the era of bounded rationality. In this work we discuss the implications of the resource constraints impose by the data stream computational model in the design of learning algorithms. We analyze the behavior of stream mining algorithms and present future research directions including ubiquitous stream mining and self-adaption models.

Povzetek: V prispevku so predstavljeni algoritmi rudarjenja pretakanja podatkov.

1 Introduction

Herbert Simon, one of the AI pioneers, developed the idea of *bounded rationality* [28]:

> *in decision-making, rationality of individuals is limited by the information they have, the cognitive limitations of their minds, and the finite amount of time they have to make a decision. In complex situations, individuals who intend to make rational choices are bound to make satisfactory (rather than maximizing) choices.*

The developments of information and communication technologies dramatically change the data collection and processing methods. What distinguish current data sets from earlier ones are automatic data feeds. We do not just have people entering information into a computer. We have computers entering data into each other [24]. Moreover, advances in miniaturization and sensor technology lead to sensor networks, collecting high detailed spatio-temporal data about the environment. These scenarios impose strong constraints in the way we think and design data mining algorithms. In short, data mining algorithms must take into account that computational resources are limited, and the design of learning algorithms must be thought in the context of *bounded rationality*.

The computational model of data streams [24] requires *resource-aware* algorithms. Hulten and Domingos [16] identify desirable properties of learning systems for efficient mining continuous, high-volume, open-ended data streams:

- Require small constant time per data example;
- Use fix amount of main memory, irrespective to the total number of examples;
- Built a decision model using a single scan over the training data;
- Generating a anytime model independent from the order of the examples;
- Ability to deal with concept drift;
- For stationary data, ability to produce decision models that are nearly identical to the ones we would obtain using a batch learner.

In this paper we review the techniques used in learning from data streams related to the bounded rationality. We also discuss ubiquitous data mining contexts where both data sources and processing devices are distributed.

2 Data streams

In the data stream computational model examples are processed once, using restricted computational resources and storage capabilities. The goal of data stream mining consists of learning a decision model, under these constraints, from sequences of observations generated from environments with unknown dynamics. Most of the stream mining works focus on centralized approaches. The phenomenal growth of mobile and embedded devices coupled with their ever-increasing computational and communications capacity presents exciting new opportunities for real-time, distributed intelligent data analysis in ubiquitous environments. In domains like sensor networks, smart grids, social cars, ambient intelligence, etc. centralized approaches have limitations due to communication constraints, power consumption, and privacy concerns. Distributed online algorithms are highly needed to address the above concerns. These applications require distributed stream algorithms, highly scalable, computationally efficient and resource-aware. Table 1 summarizes the main differences between data base processing and data stream processing.

	Data bases	Data Streams
Data access	Random	Sequential
Number of passes	Multiple	Single
Processing Time	Unlimited	Restricted
Available Memory	Unlimited	Fixed
Result	Accurate	Approximate
Distributed	No	Yes

Table 1: Summary of the main differences between data base processing and data stream processing.

2.1 Approximation and randomization

Bounded rationality appears in a fundamental aspect in stream processing: the tradeoff between time and space required to solve a query and the accuracy of the answer. Data stream management systems developed a set of techniques that store compact stream summaries enough to approximately solve queries. These approaches require a trade-off between accuracy and the amount of memory used to store the summaries, with an additional constrain of small time to process data items [24, 2]. The most common problems end up to compute quantiles, frequent item sets, and to store frequent counts along with error bounds on their true frequency. The techniques developed are used in very high dimensions both in the number of examples and in the cardinality of the variables.

Many fundamental questions, like counting, require space linear in the input to obtain exact answers. Within data stream framework, *approximation* techniques, that is, answers that are correct within some small fraction ϵ of error; and *randomization* [23], that allows a small probability δ of failure, are used to obtain answers that with probability $1-\delta$ are in an interval of radius ϵ. Algorithms that use both approximation and randomization are referred to as (ϵ, δ) approximations. The base idea consists of mapping a very large input space to a small synopsis of size $O(\frac{1}{\epsilon^2} log(\frac{1}{\delta}))$.

Approximation and randomization techniques are the most illustrative examples of bounded rationality in data stream processing. They are used to solve problems like measuring the entropy of a stream [6], association rule mining frequent items [21], k-means clustering for distributed data streams using only local information [9], etc.

2.2 Time windows

Most of the time, we are not interested in computing statistics over all the past, but only over the *recent* past. The assumption behind all these models is that most recent information is more relevant than historical data. The simplest situation uses *sliding windows* of fixed size. These types of windows are similar to *first in, first out* data structures. Whenever an element j is observed and inserted into the window, another element $j - w$, where w represents the window size, is forgotten. Computing statistics in the sliding window model requires storing all data inside the window. Exponential histograms [10] are used to approximately compute statistics over sliding windows requiring sublinear space in the size of the window.

Monitoring, analyzing and extracting knowledge from high speed streams might explore multiple levels of granularity, where the recent history is analyzed at fine levels of granularity and the need of precision decreases with the age of the data. As a consequence, the most recent data can be stored at the finest granularity, while more distant data at coarser granularity. This is called the *tilted* time window model. It might be implemented using exponential histograms [10].

2.3 Change detection

Sequence based windows is a general technique to deal with changes in the process that generates data. A reference algorithm is the `AdWin` – `AD`aptive sliding `WIN`dow presented by Bifet and Gavaldà [4]. `AdWin` keeps a variable-length window of recently seen items, with the property that the window has the maximal length statistically consistent with the hypothesis *there has been no change in the average value inside the window*. More precisely, an older fragment of the window is dropped if and only if there is enough evidence that its average value differs from that of the rest of the window. This has two consequences: first, that change is reliably declared whenever the window shrinks; and second, that at any time the average over the existing window can be reliably taken as an estimate of the current average in the stream (barring a very small or very recent change that is still not statistically visible). `AdWin` does not maintain the window explicitly, but compresses it using a variant of the exponential histogram technique. This means that it keeps a window of length W using only $O(logW)$ memory and $O(logW)$ processing time per item.

2.4 Sampling

Sampling is a common practice for selecting a subset of data to be analyzed. Instead of dealing with an entire data stream, we select instances at periodic intervals. Sampling is used to compute statistics (expected values) of the stream. While sampling methods reduce the amount of data to process, and, by consequence, the computational costs, they can also be a source of errors, namely in monitoring applications that require to detect anomalies or extreme values.

The main problem is to obtain a *representative* sample, that is, a subset of data that has approximately the same properties of the original data. In statistics, most techniques require to know the length of the stream. For data streams, we need to modify these techniques. The simplest form of sampling is *random sampling*, where each element has equal probability of being selected [1]. The *reservoir sampling* technique [31] is the classic algorithm to maintain an online random sample. The base idea consists of maintaining a sample of size k, called the reservoir. As the stream flows, every new element has a probability k/n, where n is the number of elements seen so far, of replacing an old element in the reservoir.

A similar technique, load shedding, drops sequences in the stream, when bursts cause bottlenecks in the processing capabilities. Tatbul et al. [29] discuss load shedding techniques in querying high-speed data streams.

2.5 Synopsis, sketches and summaries

Synopses are compact data structures that summarize data for further querying. Several methods have been used, including: wavelets [15], exponential histograms [10], frequency moments [3], etc. Data sketching via random projections is a tool for dimensionality reduction. Sketching uses random projections of data points with dimension d to a space of a subset of dimensions. It has been used for moment estimation [3], computing L-norms [24] and dot product of streams [1].

Cormode and Muthukrishnan [8] present a data stream summary, the so called *count-min* sketch, used for (ϵ, δ) approximations to solve *point queries*, *range queries*, and *inner product queries*. Consider an implicit vector $\mathbf{a}$ of dimension n that is incrementally updated over time. At each moment, the element a_i represents the counter associated with element i. A point-query is to estimate the value of an entry in the vector $\mathbf{a}$. The count-min sketch data structure, with parameters (ϵ, δ), is an array of $w \times d$ in size, where $d = log(1/\delta)$, and $w = 2/\epsilon$. For each incoming value of the stream, the algorithm use d hash functions to map entries to $[1, \ldots, w]$. The counters in each row are incremented to reflect the update. From this data structure, we can estimate at any time, the number of occurrences of any item j by taking $min_d CM[d, h_d(j)]$. Since the space used by the sketch CM is typically much smaller than that required to represent the vector $\mathbf{a}$ exactly, there is necessarily some approximation in the estimate. The estimate $\hat{a_j}$, has the following guarantees: $a_j \leq \hat{a_j}$, and, with probability at least $1 - \delta$, $\hat{a_i} \leq a_i + \epsilon||a||_1$. The error of the estimate is at most ϵ with probability at least $1 - \delta$ in space $O(\frac{1}{\epsilon}log(\frac{1}{\delta}))$.

3 Algorithms for learning from data streams

Data Mining studies the automated acquisition of domain knowledge looking for the improvement of systems performance as result of experience. Data stream mining systems address the problems of data processing, modeling, prediction, clustering, and control in changing and evolving environments. They self-evolve their structure and knowledge on the environment [12]. In this section we address two challenging problems in stream mining that enforce the idea of bounded rationality.

3.1 Mining infinite data

Hulten and Domingos [16] present a general method to learn from arbitrarily large databases. The method consists of deriving an upper bound for the learner's loss as a function of the number of examples used in each step of the algorithm. Then use this to minimize the number of examples required at each step, while guaranteeing that the model produced does not differ significantly from the one that would be obtained with infinite data. This methodology has been successfully applied in k-means clustering [16], hierarchical clustering of variables [26], decision trees [11, 17], regression trees [18], decision rules [14], etc.

The most representative algorithm in this line is the `Very Fast Decision Tree` [11]. `VFDT` is a decision-tree learning algorithm that dynamically adjusts its bias accordingly to the availability of data. In decision tree induction, the main issue is the decision of when to expand the tree, installing a splitting-test and generating new leaves. The basic idea consists of using a small set of examples to select the splitting-test to incorporate in a decision tree node. If after seeing a set of examples, the difference of the merit between the two best splitting-tests does not satisfy a statistical test (the Hoeffding bound), `VFDT` proceeds by examining more examples. It only makes a decision (i.e., adds a splitting-test in that node), when there is enough statistical evidence in favor of a particular test. This strategy guarantees model stability (low variance) and controls overfiting – examples are processed once without the need of model regularization (pruning). This profile is quite different from the standard decision trees model using greedy search and static data sets. Using static datasets, the decisions in deeper nodes of the tree are based on less and less examples. Statistical support decreases as the tree grows, and regularization is mandatory. In VFDT like algorithms the number of examples needed to grow a node is only defined by the statistical significance of the difference between the two best alternatives. Deeper nodes of the tree might require more examples than those used in the root of the tree!

3.2 Mining ubiquitous streams

The phenomenal growth of mobile and embedded devices coupled with their ever-increasing computational and communications capacity presents an exciting new opportunity for real-time, distributed intelligent data analysis in ubiquitous environments. Learning from distributed data, require efficient methods in minimizing the communication overheads between nodes [27]. The strong limitations of centralized solutions is discussed in depth in [20]. The authors point out *a mismatch between the architecture of most off-the-shelf data mining algorithms and the needs of mining systems for distributed applications*. Such mismatch may cause a bottleneck in many emerging applications, namely hardware limitations related to the limited bandwidth channels. Most important, in applications like monitoring, centralized solutions introduce delays in event detection and reaction, that can make mining systems useless.

Ubiquitous data stream mining implies new requirements to the bounded rationality to be considered [22]: *i)* the algorithms will process local information and *ii)* need to communicate with other agents in order to infer the global learning context; *iii)* the budget for communications (bandwidth, battery) are limited.

Typical applications include clustering sensor networks [25], autonomous vehicles [30], analysis of large social networks [19], multiple classification models exploring distributed processing [7], etc.

4 Concluding remarks

There is a fundamental difference between learning from small datasets and streaming data: mining data streams is a continuous process. This observation opens the ability to monitor the evolution of the learning process; to use change detection mechanisms to self-diagnosis the evolution of this process, and to react and repair decision models [13]. Continuous learning, forgetting, self-adaptation, and self-reaction are main characteristics of any intelligent system. They are characteristic properties of stream learning algorithms. From a bias-variance analysis there is a fundamental difference [5]: while learning from small data sets requires an emphasis in variance reduction, learning from large data sets is more effective when using algorithms that place greater emphasis on bias management.

Bounded rationality is in the core of next generation of data mining systems. Learning algorithms must be able to adapt continuously to changing environmental conditions (including their own condition) and evolving user needs. Learning must consider the real-time constrains of limited computing power and communication resources.

Acknowledgment

This work is funded by the ERDF through the Programme COMPETE PEst-C/SAU/UI0753/2011 and by FCT project PTDC/EIA/098355/2008, *Knowledge Discovery from Ubiquitous Data Streams*.

References

[1] Aggarwal, C. (2006). On biased reservoir sampling in the presence of stream evolution. In *Proceedings International Conference on Very Large Data Bases*, Seoul, Korea, pp. 607–618. ACM.

[2] Aggarwal, C. (Ed.) (2007). *Data Streams – Models and Algorithms*. Springer.

[3] Alon, N., Y. Matias, and M. Szegedy (1999). The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences 58*, 137–147.

[4] Bifet, A. and R. Gavaldà (2006). Kalman filters and adaptive windows for learning in data streams. In *Proceedings of the 9th Discovery Science*, Volume 4265 of *Lecture Notes Artificial Intelligence*, Barcelona, Spain, pp. 29–40. Springer.

[5] Brain, D. and G. Webb (2002). The need for low bias algorithms in classification learning from large data sets. In *Principles of Data Mining and Knowledge Discovery PKDD*, Volume 2431 of *Lecture Notes in Artificial Intelligence*, Helsinki, Finland, pp. 62–73. Springer.

[6] Chakrabarti, A., K. D. Ba, and S. Muthukrishnan (2006). Estimating entropy and entropy norm on data streams. In *STACS: 23rd Annual Symposium on Theoretical Aspects of Computer Science*, Marseille, France, pp. 196–205.

[7] Chen, R., K. Sivakumar, and H. Kargupta (2004). Collective mining of Bayesian networks from heterogeneous data. *Knowledge and Information Systems Journal 6*(2), 164–187.

[8] Cormode, G. and S. Muthukrishnan (2005). An improved data stream summary: the count-min sketch and its applications. *Journal of Algorithms 55*(1), 58–75.

[9] Cormode, G., S. Muthukrishnan, and W. Zhuang (2007). Conquering the divide: Continuous clustering of distributed data streams. In *ICDE: Proceedings of the International Conference on Data Engineering*, Istanbul, Turkey, pp. 1036–1045.

[10] Datar, M., A. Gionis, P. Indyk, and R. Motwani (2002). Maintaining stream statistics over sliding windows. In *Proceedings of Annual ACM-SIAM Symposium on Discrete Algorithms*, San Francisco, USA, pp. 635–644. Society for Industrial and Applied Mathematics.

[11] Domingos, P. and G. Hulten (2000). Mining High-Speed Data Streams. In *Proceedings of the ACM Sixth International Conference on Knowledge Discovery and Data Mining*, Boston, USA, pp. 71–80. ACM Press.

[12] Gama, J. (2010). *KnowledgeDiscovery from Data Streams*. Data Mining and Knowledge Discovery. Atlanta, US: Chapman & Hall CRC Press.

[13] Gama, J. and P. Kosina (2011a). Learning about the learning process. In *Advances in Intelligent Data Analysis*, Volume 7014 of *Lecture Notes in Computer Science*, pp. 162–172. Springer.

[14] Gama, J. and P. Kosina (2011b). Learning decision rules from data streams. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence, IJCAI*, pp. 1255–1260.

[15] Gilbert, A. C., Y. Kotidis, S. Muthukrishnan, and M. Strauss (2001). Surfing wavelets on streams: One-pass summaries for approximate aggregate queries. In *VLDB*, Rome, Italy, pp. 79–88.

[16] Hulten, G. and P. Domingos (2001). Catching up with the data: research issues in mining data streams. In *Proc. of Workshop on Research Issues in Data Mining and Knowledge Discovery*, Santa Barbara, USA.

[17] Hulten, G., L. Spencer, and P. Domingos (2001). Mining time-changing data streams. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, California, pp. 97–106. ACM Press.

[18] Ikonomovska, E., J. Gama, and S. Dzeroski (2011). Learning model trees from evolving data streams. *Data Mining and Knowledge Discovery 23*, 128–168.

[19] Kang, U., C. E. Tsourakakis, and C. Faloutsos (2011). Pegasus: mining peta-scale graphs. *Knowl. Inf. Syst. 27*(2), 303–325.

[20] Kargupta, H., A. Joshi, K. Sivakumar, and Y. Yesha (2004). *Data Mining: Next Generation Challenges and Future Directions*. AAAI Press and MIT Press.

[21] Manku, G. S. and R. Motwani (2002). Approximate frequency counts over data streams. In *Proceedings of 28th International Conference on Very Large Data Bases*, Hong Kong, pp. 346–357. Morgan Kaufmann.

[22] May, M. and L. Saitta (Eds.) (2010). *Ubiquitous Knowledge Discovery*. LNAI 6202, Springer.

[23] Motwani, R. and P. Raghavan (1997). *Randomized Algorithms*. Cambridge University Press.

[24] Muthukrishnan, S. (2005). *Data Streams: Algorithms and Applications*. Now Publishers.

[25] Rodrigues, P. P., J. Gama, and L. Lopes (2009). Knowledge discovery for sensor network comprehension. In *Intelligent Techniques for Warehousing and Mining Sensor Network Data*, pp. 118–134. Information Science.

[26] Rodrigues, P. P., J. Gama, and J. P. Pedroso (2008). Hierarchical clustering of time series data streams. *IEEE Transactions on Knowledge and Data Engineering 20*(5), 615–627.

[27] Sharfman, I., A. Schuster, and D. Keren (2007). A geometric approach to monitoring threshold functions over distributed data streams. *ACM Transactions Database Systems 32*(4), 301–312.

[28] Simon, H. (1957). *Models of Man*. John Wiley.

[29] Tatbul, N., U. Cetintemel, S. Zdonik, M. Cherniack, and M. Stonebraker (2003). Load shedding in a data stream manager. In *Proceedings of the International Conference on Very Large Data Bases*, Berlin, Germany, pp. 309–320. VLDB Endowment.

[30] Thrun, S. (2010). Toward robotic cars. *Communications ACM 53*(4), 99–106.

[31] Vitter, J. S. (1985). Random sampling with a reservoir. *ACM Transactions on Mathematical Software 11*(1), 37–57.

Automatic Text Analysis by Artificial Intelligence

Dunja Mladenić[2] and Marko Grobelnik
Jožef Stefan Institute, Artificial Intelligence Laboratory, Jamova 39, 1000 Ljubljana, Slovenia
E-mail: dunja.mladenic@ijs.si, marko.grobelnik@ijs.si, http://ailab.ijs.si/,

[2]Jožef Stefan International Postgraduate School, Jamova 39, 1000 Ljubljana, Slovenia

Keywords: text mining, natural language processing, machine learning, data visualization, big data

Received: November 7, 2012

Text is one of the traditional ways of communication between people. With the growing availability of text data in electronic form, handling and analysis of text by means of computers gained popularity. Handling text data with machine learning methods brought interesting challenges to the area that got further extended by incorporation of some natural language specifics. As the methods were capable of addressing more complex problems related to text data, the expectations become bigger calling for more sophisticated methods, in particular a combination of methods from different research areas including information retrieval, machine learning, statistical data analysis, data mining, natural language processing, semantic technologies. Automatic text analysis become an integral part of many systems, pushing boundaries of research capabilities towards what one can refer to as an artificial intelligence dream - never ending learning from text aiming at mimicking ways of human learning. The paper presents development of text analysis research in Slovenian that we have been personally involved in, pointing out interesting research problems that have been and are still addressed by the research, example tasks that have been addressed and some challenges on the way.

Povzetek: V članku je predstavljen razvoj raziskav analize besedil z metodami umetne inteligence v Sloveniji.

1 Introduction

Word expressed as a sound is known as a fundamental phenomena in creation of our world. “Every element of the universe is in a constant state of vibration manifested to us as light, sounds and energy. The human senses perceive only a fraction of the infinite range of vibration, so it is difficult to comprehend that the Word mentioned in the Bible is actually the totality of vibration which underlines and sustains the creation.” [1] . Written words are one of the traditional ways of communication over space and time. “Communication is a gift to know. Communication is a gift to understand. Communication is a gift to realize.” [2] . To understand what has been communicated by some text is not always easy. Automatic text analysis can often contribute to understand the text, to gaining knowledge from the data provided in textual form, to realize the underlying facts that have been communicated via the text.

As electronic media become widely used, the amount of texts in electronic form has grown rapidly and is still growing. While these texts are primarily aiming at human readers, it is not uncommon to use computer programs to manipulate texts. Text handling by computer programs has a wide range of usage from enabling text editing, storing and indexing text for searching and retrieval, ranking documents, classifying documents, extracting information and knowledge, question answering, etc.

In this paper we present development of artificial intelligence research in Slovenia related to handling of text data that we have been personally involved in. The paper points out some interesting research problems that have been and are still addressed, listing some example tasks that have been addressed in our group and some challenges on the way. We conclude by providing discussion and some direction for future research on automatic text analysis.

2 Handling text data

In the 1990s handling of text data by machine learning techniques was inspired mainly by information retrieval, where machine learning methods were used primarily for classification of documents regarding their relevance to a given query (as an alternative to the information retrieval ranking methods). At that time, machine learning was also applied for personalized information delivery on text, such as, learning to filter relevant Netnews, suggesting potentially relevant hyperlinks on Web documents [3] , [4] , browsing the Web [5] , powering intelligent agents [6] . As texts (documents, Web pages, news articles) are often manually labeled by some topic category (e.g., a news on acquisitions, a Web page on artificial intelligence), this is a natural area for applying machine learning methods to train a classifier for topic classification. The problem is far from being a trivial application of machine learning methods to a new domain. The number of classes may get much larger than what was usual at the time for machine learning methods to handle, requiring a careful handling of efficient classifier construction [7] and pruning the space of classifiers to be consulted at the classification of a new example [8] .

Using words as features is, in such a setting, a common way of representing text documents so that machine learning methods can be applied on them. As each word from the vocabulary is assigned a feature with its value being based on the frequency of the word in a document, the number of features easily got to several tens of thousands. Moreover, one can think about some more sophisticated features beyond single words, such as sequences of words [9] , additionally increasing the feature space. This requires careful handling of the problem including efficient feature selection [10] .

Even though many relevant problems can and have been addressed at the level of documents using machine learning methods and at the level of sentences and words using natural language processing methods, there is still a way to go towards automatically obtaining knowledge from text to be used for ontology extension and reasoning. Extracting knowledge form the text and representing it in logical forms means that a computer can reason on it, provide hopefully some interesting insights and propose new conclusions. One of the earlier attempts included information extraction from Web pages using manually constructed wrappers and forming rules connecting the extracted information [11] . A step towards extraction of knowledge for ontology generation is presented in [12] , where natural language processing is used in combination with semantic technologies. While there are a number of similar efforts in direction of knowledge extraction from text, the problem of obtaining logical statements corresponding to some text remains open.

In general different methods from the area of artificial intelligence can be used for obtaining knowledge from text [13] ranging from classification and clustering, to association rule construction and visualization.

3 Example tasks

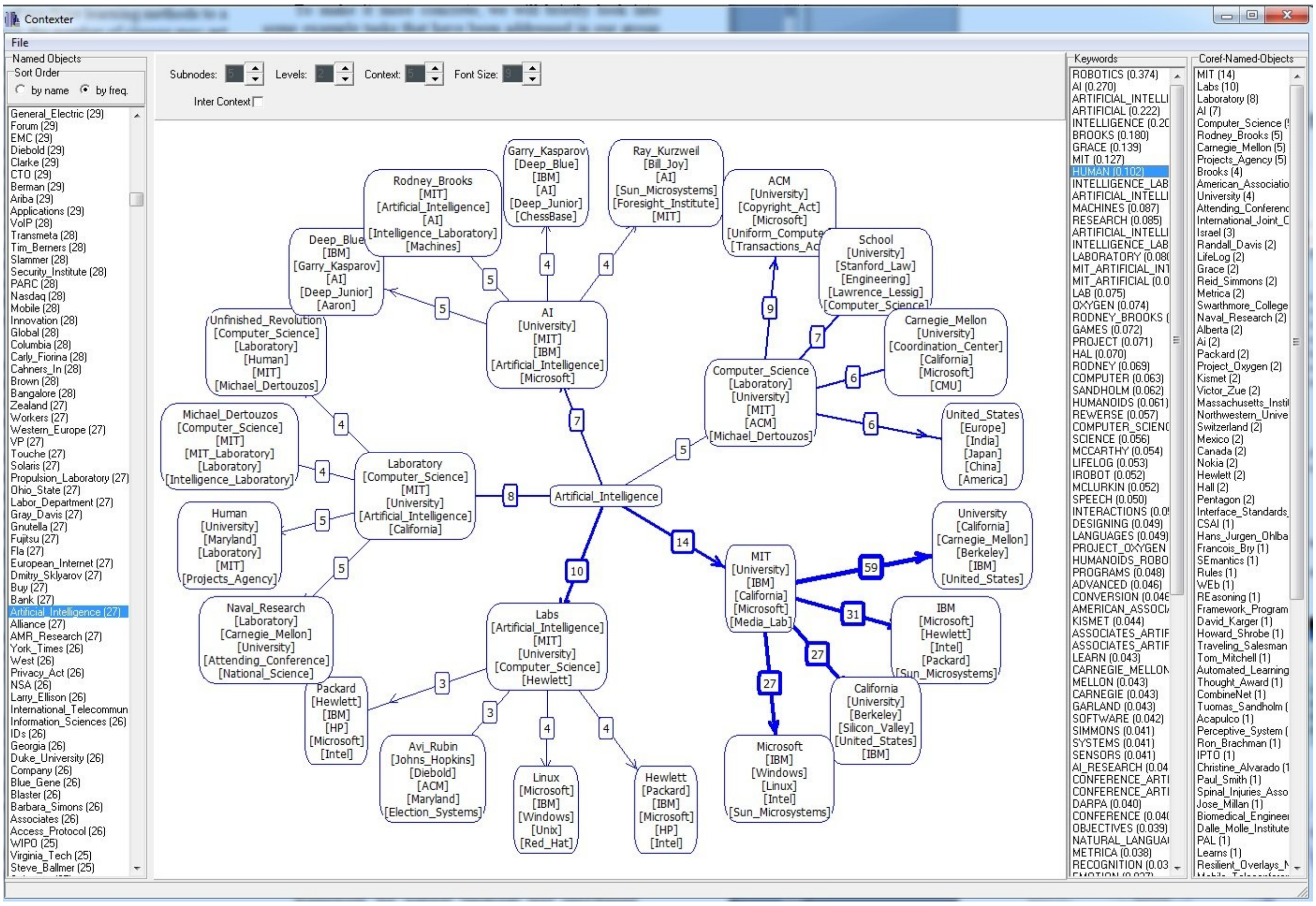
When we talk about applying artificial intelligence methods on text data, what we have in mind is a whole range of methods and problems that in some way involve analysis of text data. Many of these problems have been addressed in the area of Text Mining. For the definition of text mining we have adapted the definition of Data Mining, so we can say that text mining is about finding interesting regularities in large text data, where interesting means: non-trivial, hidden, previously unknown and potentially useful. Looking from the linguistic and semantic technologies perspective, text mining can be defined as finding semantic and abstract information from the surface form of text.

To be more concrete, we will briefly look into some example tasks that have been addressed in our group during the last twenty years by applying artificial intelligence methods on text data. These include:

- visualization of text available in news articles, visualization of named entities over time, visualization of document corpus, visualization of Web pages;
- triplet extraction from text, document representation using semantic graph, document summarization;
- text enrichment, contextual question answering;
- semi-automatic ontology construction from document corpus, ontology extension;
- knowledge extraction from text, text mining combined with social network analysis.

Visualization of text data available in news articles can be based on named entity extraction, as news are usually mentioning some named entities (e.g., people, countries, organizations) putting them in some relation. Visualization of news articles as proposed in [14] is based on extracting named entities from news articles and representing them in a graph (name entities being vertices connected if they appear in the same news article). The graph of entities is enriched with contextual information in the form of characteristic keywords and name entities related to the entity in the focus. Operations for browsing a graph are implemented to be efficient enabling interactive user experience with a quick capturing of large amounts of information present in the original text. Figure 1 shows the user interface on ACM Technology News consisting of 11000 article abstracts.

Named entities that have time information associated to them can be related to each other on a time scale. An approach relating people, places, organizations and events extracted from Wikipedia and linking them on a time scale is proposed in [15] . Relevant Wikipedia pages are identified by categorizing the articles as containing people, places or organizations. Then a timeline if generated linking the named entities and extracting events and their time frame.



Contexter

File

Named Objects

Sort Order

○ by name ● by freq.

General_Electric (29)
Forum (29)
EMC (29)
Diebold (29)
Clarke (29)
CTO (29)
Berman (29)
Ariba (29)
Applications (29)
VoIP (28)
Transmeta (28)
Tim_Berners (28)
Slammer (28)
Security_Institute (28)
PARC (28)
Nasdaq (28)
Mobile (28)
Innovation (28)
Global (28)
Columbia (28)
Carly_Fiorina (28)
Cahners_In (28)
Brown (28)
Bangalore (28)
Zealand (27)
Workers (27)
Western_Europe (27)
VP (27)
Touche (27)
Solaris (27)
Propulsion_Laboratory (27)
Ohio_State (27)
Labor_Department (27)
Gray_Davis (27)
Gnutella (27)
Fujitsu (27)
Fla (27)
European_Internet (27)
Dmitry_Sklyarov (27)
Buy (27)
Bank (27)
Artificial_Intelligence (27)
Alliance (27)
AMR_Research (27)
York_Times (26)
West (26)
Privacy_Act (26)
NSA (26)
Larry_Ellison (26)
International_Telecommun
Information_Sciences (26)
IDs (26)
Georgia (26)
Duke_University (26)
Company (26)
Blue_Gene (26)
Blaster (26)
Barbara_Simons (26)
Associates (26)
Access_Protocol (26)
WIPO (25)
Virginia_Tech (25)
Steve_Ballmer (25)

Subnodes: 5 Levels: 2 Context: 5 Font Size: 9

Inter Context □

Keywords

ROBOTICS (0.374)
AI (0.270)
ARTIFICIAL_INTELLI
ARTIFICIAL (0.222)
INTELLIGENCE (0.20
BROOKS (0.180)
GRACE (0.139)
MIT (0.127)
HUMAN (0.102)
INTELLIGENCE_LAB
ARTIFICIAL_INTELLI
MACHINES (0.087)
RESEARCH (0.085)
ARTIFICIAL_INTELLI
INTELLIGENCE_LAB
LABORATORY (0.08(
MIT_ARTIFICIAL_INT
MIT_ARTIFICIAL (0.0
LAB (0.075)
OXYGEN (0.074)
RODNEY_BROOKS (
GAMES (0.072)
PROJECT (0.071)
HAL (0.070)
RODNEY (0.069)
COMPUTER (0.063)
SANDHOLM (0.062)
HUMANOIDS (0.061)
REWERSE (0.057)
COMPUTER_SCIENC
SCIENCE (0.056)
MCCARTHY (0.054)
LIFELOG (0.053)
IROBOT (0.052)
MCLURKIN (0.052)
SPEECH (0.050)
INTERACTIONS (0.0!
DESIGNING (0.049)
LANGUAGES (0.049)
PROJECT_OXYGEN
HUMANOIDS_ROBO
PROGRAMS (0.048)
ADVANCED (0.046)
CONVERSION (0.04E
AMERICAN_ASSOCI
KISMET (0.044)
ASSOCIATES_ARTIF
ASSOCIATES_ARTIF
LEARN (0.043)
CARNEGIE_MELLON
MELLON (0.043)
CARNEGIE (0.043)
GARLAND (0.043)
SOFTWARE (0.042)
SIMMONS (0.041)
SYSTEMS (0.041)
SENSORS (0.041)
AI_RESEARCH (0.04
CONFERENCE_ARTI
CONFERENCE_ARTI
DARPA (0.040)
CONFERENCE (0.04(
OBJECTIVES (0.039)
NATURAL_LANGUA
METRICA (0.038)
RECOGNITION (0.03

Coref-Named-Objects

MIT (14)
Labs (10)
Laboratory (8)
AI (7)
Computer_Science (!
Rodney_Brooks (5)
Carnegie_Mellon (5)
Projects_Agency (5)
Brooks (4)
American_Associatio
University (4)
Attending_Conferenc
International_Joint_C
Israel (3)
Randall_Davis (2)
LifeLog (2)
Grace (2)
Reid_Simmons (2)
Metrica (2)
Swarthmore_College
Naval_Research (2)
Alberta (2)
Ai (2)
Packard (2)
Project_Oxygen (2)
Kismet (2)
Victor_Zue (2)
Massachusetts_Instil
Northwestern_Unive
Switzerland (2)
Mexico (2)
Canada (2)
Nokia (2)
Hewlett (2)
Hall (2)
Pentagon (2)
Interface_Standards_
CSAI (1)
Hans_Jurgen_Ohlba
Francois_Bry (1)
SEmantics (1)
Rules (1)
WEb (1)
REasoning (1)
Framework_Program
David_Karger (1)
Howard_Shrobe (1)
Traveling_Salesman
Tom_Mitchell (1)
Automated_Learning
Thought_Award (1)
CombineNet (1)
Tuomas_Sandholm (
Acapulco (1)
Perceptive_System (
Ron_Brachman (1)
IPTO (1)
Christine_Alvarado (1
Paul_Smith (1)
Spinal_Injuries_Asso
Jose_Millan (1)
Biomedical_Engineer
Dalle_Molle_Institute
PAL (1)
Learns (1)
Resilient_Overlays_N

Artificial_Intelligence
AI [University] [MIT] [IBM] [Artificial_Intelligence] [Microsoft]
Laboratory [Computer_Science] [MIT] [University] [Artificial_Intelligence] [California]
Labs [Artificial_Intelligence] [MIT] [University] [Computer_Science] [Hewlett]
MIT [University] [IBM] [California] [Microsoft] [Media_Lab]
Computer_Science [Laboratory] [University] [MIT] [ACM] [Michael_Dertouzos]
Rodney_Brooks [MIT] [Artificial_Intelligence] [AI] [Intelligence_Laboratory] [Machines]
Garry_Kasparov [Deep_Blue] [IBM] [AI] [Deep_Junior] [ChessBase]
Ray_Kurzweil [Bill_Joy] [AI] [Sun_Microsystems] [Foresight_Institute] [MIT]
Deep_Blue [IBM] [Garry_Kasparov] [AI] [Deep_Junior] [Aaron]
Unfinished_Revolution [Computer_Science] [Laboratory] [Human] [MIT] [Michael_Dertouzos]
Michael_Dertouzos [Computer_Science] [MIT] [MIT_Laboratory] [Laboratory] [Intelligence_Laboratory]
Human [University] [Maryland] [Laboratory] [MIT] [Projects_Agency]
Naval_Research [Laboratory] [Carnegie_Mellon] [University] [Attending_Conference] [National_Science]
Packard [Hewlett] [IBM] [HP] [Microsoft] [Intel]
Avi_Rubin [Johns_Hopkins] [Diebold] [ACM] [Maryland] [Election_Systems]
Linux [Microsoft] [IBM] [Windows] [Unix] [Red_Hat]
Hewlett [Packard] [IBM] [Microsoft] [HP] [Intel]
Microsoft [IBM] [Windows] [Linux] [Intel] [Sun_Microsystems]
California [University] [Berkeley] [Silicon_Valley] [United_States] [IBM]
IBM [Microsoft] [Hewlett] [Intel] [Packard] [Sun_Microsystems]
University [California] [Carnegie_Mellon] [Berkeley] [IBM] [United_States]
United_States [Europe] [India] [Japan] [China] [America]
Carnegie_Mellon [University] [Coordination_Center] [California] [Microsoft] [CMU]
School [University] [Stanford_Law] [Engineering] [Lawrence_Lessig] [Computer_Science]
ACM [University] [Copyright_Act] [Microsoft] [Uniform_Compute] [Transactions_Ac]
5 4 4 5 7 4 4 5 8 10 14 59 31 27 27 9 7 6 6 5 3 3 4 4

Figure 1: *Contexter* graphical interface for browsing/visualizing the name-entity network. Showing context of named entity *Artificial Intelligence* (eg., occurring 5 times in the news collection with *Computer Science* that occurs 6 times with *Carnegie Mellon*. Among the most important keywords for the news where *Artificial Intelligence* occurs, we can see *robotics, AI, human, machines* etc.

General document corpus can be also visualized using clustering methods on text data [16] . Document corpus visualization can be further used in **Semi-automatic construction of topic ontology** using machine learning to cluster document, to map documents onto some existing ontologies, to suggest concept naming [17] .

In addition to addressing problems that focus on handling documents as the main units, it is also relevant to split texts into smaller units, such as, paragraphs, sentences, words or even characters. In this way one can **annotate text** on different levels of granularity including topic category of the whole document, extraction of facts mentioned in the text, named entity extraction and resolution (into some ontology such as, DBPedia, OpenCyc). Figure 3 shows an example output for a homepage annotations produced by Enrycher [18] , where the identified named entities are linked to concepts in DBPedia, OpenCyc and GeoNames. In addition, text of the homepage is assigned several topic categories from Open directory (by machine learning methods an efficient text classifier is constructed from Open directory). Enrycher also produces a semantic graph of text and extracts interesting statements from it, such as Dunja Mladenic is Enwise expert. Text annotations has been also used for **enhancing visualization of web pages** [19] by transforming the page to semantic graph, using machine learning to rank the triplets enabling page visualization as a semantic graph of a chosen size (see Figure 2).

Extracting triplets from text [20] involves some more or less sophisticated natural language processing to extract what would be considered as subject – predicate – object triplets from sentences. Even though the original approach uses parsing of a sentence to get its logical form (extracting subject-predicate-object) [21] , reasonable results have been achieved by using predefined patterns, such as noun phrase – verb phrase – noun phrase to extract triplets [18] . The extracted triplets can be also generalized to a kind of templates [22] , such as, country – borders – country that can be further used to extend an ontology or to extract information from text.

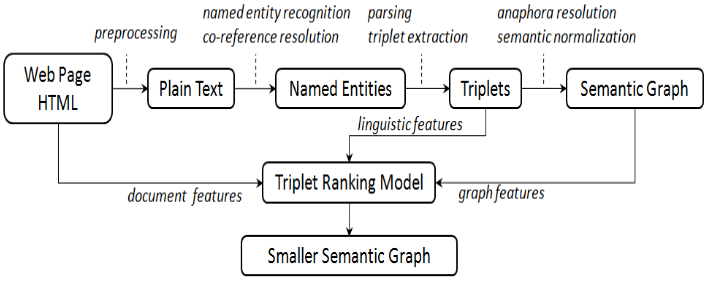
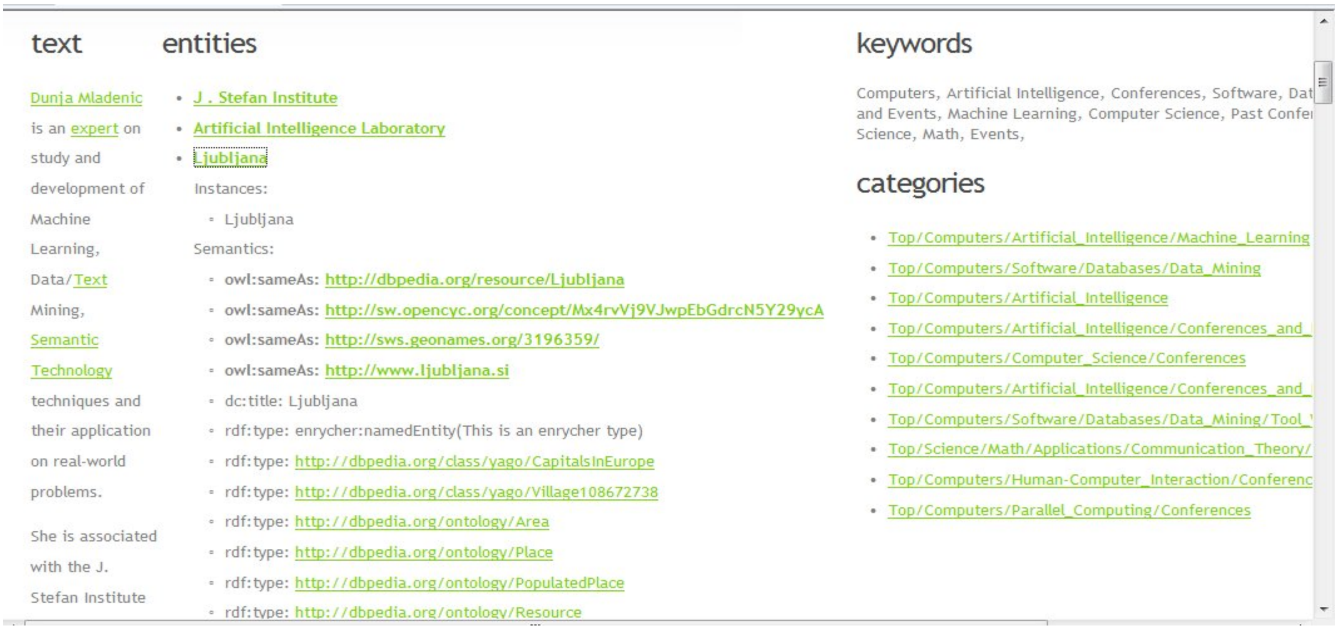


Figure 2: Steps in enhanced web page visualization.



text

Dunja Mladenic is an expert on study and development of Machine Learning, Data/Text Mining, Semantic Technology techniques and their application on real-world problems.

She is associated with the J. Stefan Institute

entities

- J . Stefan Institute
- Artificial Intelligence Laboratory
- Ljubljana

  Instances:
  - Ljubljana

  Semantics:
  - owl:sameAs: http://dbpedia.org/resource/Ljubljana
  - owl:sameAs: http://sw.opencyc.org/concept/Mx4rvVj9VJwpEbGdrcN5Y29ycA
  - owl:sameAs: http://sws.geonames.org/3196359/
  - owl:sameAs: http://www.ljubljana.si
  - dc:title: Ljubljana
  - rdf:type: enrycher:namedEntity(This is an enrycher type)
  - rdf:type: http://dbpedia.org/class/yago/CapitalsInEurope
  - rdf:type: http://dbpedia.org/class/yago/Village108672738
  - rdf:type: http://dbpedia.org/ontology/Area
  - rdf:type: http://dbpedia.org/ontology/Place
  - rdf:type: http://dbpedia.org/ontology/PopulatedPlace
  - rdf:type: http://dbpedia.org/ontology/Resource

keywords

Computers, Artificial Intelligence, Conferences, Software, Dat and Events, Machine Learning, Computer Science, Past Confe Science, Math, Events,

categories

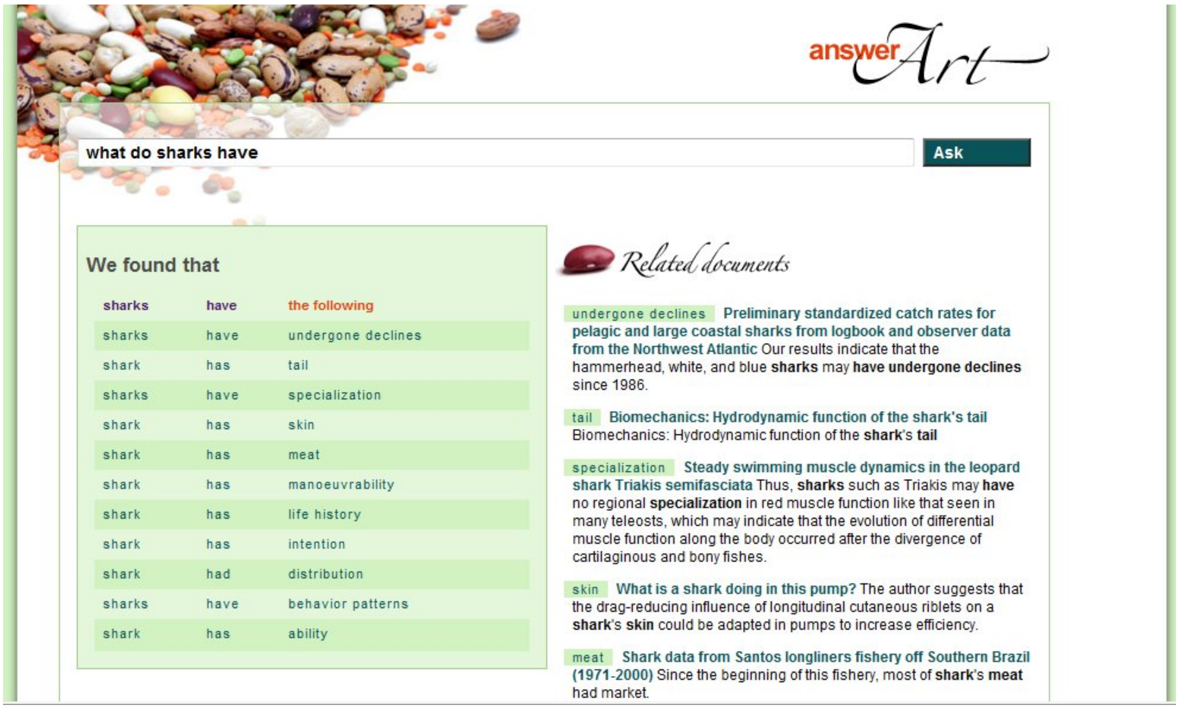
- Top/Computers/Artificial_Intelligence/Machine_Learning
- Top/Computers/Software/Databases/Data_Mining
- Top/Computers/Artificial_Intelligence
- Top/Computers/Artificial_Intelligence/Conferences_and_
- Top/Computers/Computer_Science/Conferences
- Top/Computers/Artificial_Intelligence/Conferences_and_
- Top/Computers/Software/Databases/Data_Mining/Tool_
- Top/Science/Math/Applications/Communication_Theory/
- Top/Computers/Human-Computer_Interaction/Conferenc
- Top/Computers/Parallel_Computing/Conferences

Figure 3: Enrycher providing text annotation on a text of homepage of Dunja Mladenić, linking entities to existing ontologies eg., Ljubljana is linked to concepts in DBPedia, Opencyc, GeoNames. The page is annotated by keywords (computers, artificial intelligence, etc.) and by topic categories from Open directory (eg., Top/Computer_Science/Artificial_Intelligence/Machine_Learning).

Document summarization aims at construction of a shorter version of the original document. It can be performed using different approaches, one of them as proposed in [21] is based on extracting triplets from text to obtain semantic structure of a document feeding features to a machine learning classifier trained to classify triplets for being included in the document summary or not.

Question answering can be also based on triplet extraction [23] , [24] . The whole document collection used for finding the answers is transformed into a collection of triplets and the question transformed into triplet is matched against the collection of triplets. Figure 4 shows interface of Answer Art system responding on the question "What do sharks have?" by listing answers in the form of triplets (eg., sharks have tail) and enabling the user to access the related documents that were used to obtain the listed answers.

Ontology construction and extension is usually performed entirely manually or semi-automatically by



We found that

sharks	have	the following
sharks	have	undergone declines
shark	has	tail
sharks	have	specialization
shark	has	skin
shark	has	meat
shark	has	manoeuvrability
shark	has	life history
shark	has	intention
shark	had	distribution
sharks	have	behavior patterns
shark	has	ability

Related documents

undergone declines **Preliminary standardized catch rates for pelagic and large coastal sharks from logbook and observer data from the Northwest Atlantic** Our results indicate that the hammerhead, white, and blue **sharks** may **have undergone declines** since 1986.

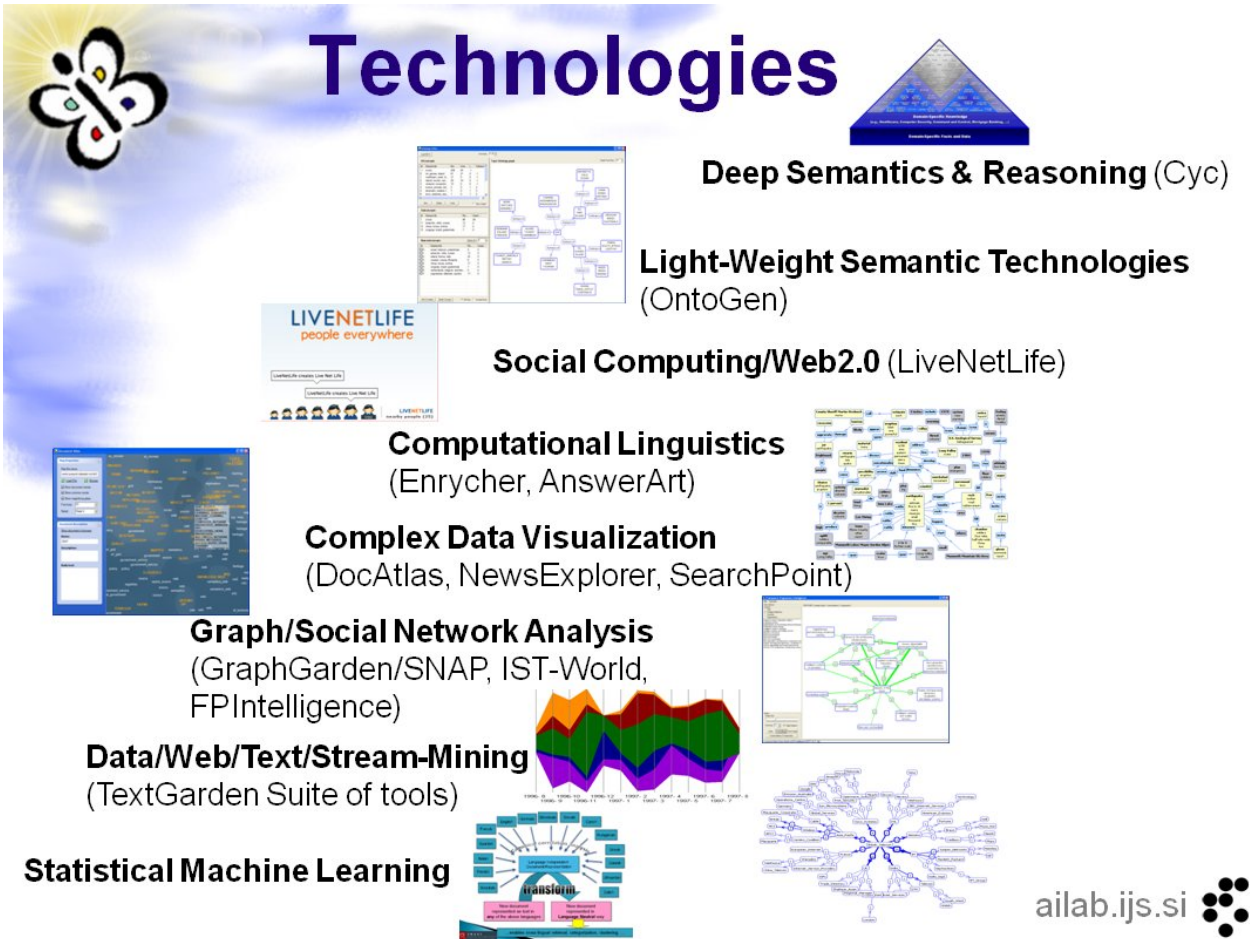
tail **Biomechanics: Hydrodynamic function of the shark's tail** Biomechanics: Hydrodynamic function of the **shark**'s **tail**

specialization **Steady swimming muscle dynamics in the leopard shark Triakis semifasciata** Thus, **sharks** such as Triakis may **have** no regional **specialization** in red muscle function like that seen in many teleosts, which may indicate that the evolution of differential muscle function along the body occurred after the divergence of cartilaginous and bony fishes.

skin **What is a shark doing in this pump?** The author suggests that the drag-reducing influence of longitudinal cutaneous riblets on a **shark**'s **skin** could be adapted in pumps to increase efficiency.

meat **Shark data from Santos longliners fishery off Southern Brazil (1971-2000)** Since the beginning of this fishery, most of **shark**'s **meat** had market.

Figure 4 *Answer Art* on question "*What do sharks have?*", based on documents on fisheries & aquaculture and ASFA ontology lists that *sharks undergo decline, have tail, have specialization, have skin, have meat* etc.



Technologies

Deep Semantics & Reasoning (Cyc)

Light-Weight Semantic Technologies (OntoGen)

Social Computing/Web2.0 (LiveNetLife)

Computational Linguistics (Enrycher, AnswerArt)

Complex Data Visualization (DocAtlas, NewsExplorer, SearchPoint)

Graph/Social Network Analysis (GraphGarden/SNAP, IST-World, FPIntelligence)

Data/Web/Text/Stream-Mining (TextGarden Suite of tools)

Statistical Machine Learning

ailab.ijs.si

Figure 5: Diagram showing different kind of technologies involving text data developed by Artificial Intelligence Laboratory at J. Stefan Institute.

applying some methods from artificial intelligence [25] . Annotation of text by the concepts of an existing ontology, as for instance used in Figure 3, is limited by the concepts that already exist in the ontology, unless we extend the existing ontology. Novel methodology for semi-automatic ontology extension aggregating the elements of text mining and user-dialog approaches is proposed in [26] . The domain of interest for ontology extension is defined by keywords and a glossary of relevant terms with term descriptions. Collaborative manual ontology extension can be supported by analysis of the dynamics of ontology changes over time. This can be especially useful when dealing with larger ontologies, where a number of editors having different expertise contribute to different parts of the ontology.

Methods from **social network analysis** can be used to gain some insights into editors' interaction with the ontology, their expertise and the ontology changes. An example approach proposed in [27] enables visualization of ontology concepts through the view of editors interacting with the concepts. Social network analysis in combination with analysis of text data can provide insights into research collaboration between institutions and countries, as proposed in [28] , where collaborations on European research projects is addressed.

Semantic technologies have been successfully applied to visualization of temporal data [30] and to support the users in dealing with information overload [31] . It was also recognized that by means of semantic technologies context of the data and the user may be used to support the user's personal productivity [32] . An approach to analysis of communication between individuals inside an organization using semantic technologies is proposed in [29] . The data on communication activity is first cleaned and transformed into a set of transactions reflecting the communication out of which a graph is constructed. The graph of transactions is represented as a matrix and fed into an tool for semi-automatic ontology construction. Out of the communication activity data an institutional ontology is constructed showing communities and important players inside the institution (eg., key people that are often involved in communication, isolated groups, well connected groups, etc.).

4 Discussion and future directions

Different Artificial Intelligence methods have been successfully applied on text data addressing a number of relevant problems. Figure 5 shows some of the technologies and the associated prototypes we have developed in our group at J. Stefan Institute ranging from statistical machine learning and data/web/text mining, to analysis of social networks and graphs, complex data visualization, computational linguistics, social computing, light-weighted semantic technologies and deep semantics with reasoning. As the methods in general become more sophisticated, the problems become more complex and researchers are constantly facing new challenges.

As an example, we can point out the fact that each text we have been handling is written is some natural language. The majority of artificial intelligence approaches focus on a single language, some handle multiple languages and other work in a cross-lingual setting adding to the complexity of the tasks and opening new challenges, as for instance in multilingual document retrieval [33] and multilingual sentiment analysis [34] . There are a number of open research challenges related to developing linguistic resources for different languages and covering multilingual and cross-lingual settings.

Another important direction of research is ensuring scalability of approaches as it is becoming common to deal with large data, also referred to as Big Data. Digging for knowledge is big data is very common goal, hoping that we will avoid traps of just noticing statistical artifacts instead of real, true phenomenon we are interested in revealing form the data. "…truth is simple, straight and with a smile. You don't have to remember it. You have to say it, you have to know it and then you have to live it."[2] .

Acknowledgements

This work was supported by the Slovenian Research Agency and the ICT Programme of the EC under a number of since 2000, the most recent being PASCAL2 (ICT-216886-NoE), RENDER (ICT-257790-STREP) and XLike (ICT-STREP-288342).

References

[1] Bhajan, Y. The Aquarian Teacher, pp.66, KRI, 2003.

[2] Bhajan, Y. Conscious Communication, KRI, 2006.

[3] Mladenić, D. Personal WebWatcher: Implementation and Design, *Technical Report IJS-DP-7472*, J. Stefan Institute, Slovenia, 1996.

[4] Joachims, T., Mladenić, D. Browsing-assistant, tour guides und adaptive WWW-server. *KI Journal, Künstl. Intell. (Oldenbourg)*, 1998, vol. 3, pp. 23-29.

[5] Mladenić, D. Web browsing using machine learning on text data. In *Intelligent exploration of the web*,111, New York; Heidelberg: Physica-Verlag, 2002, pp. 288-303.

[6] Mladenić, D. Text-learning and related intelligent agents : a survey. *IEEE intelligent systems and their applications*, 1999, vol. 14, pp. 44-54.

[7] Grobelnik, M, Mladenić, D. Simple classification into large topic ontology of web documents. *CIT. J. Comput. Inf. Technol.*, 2005, vol. 13, pp. 279-285.

[8] Mladenić, D. Turning Yahoo into an automatic Web-page classifier. In *Proceedings ECAI-1998*. Chichester [etc.]: John Wiley & Sons, 1998, pp. 473-474.

[9] Mladenić, D., Grobelnik, M. Word sequences as features in text-learning. In *Proceedings of the seventh Electrotechnik and Computer Science Conference ERK-1998, 1998*, Ljubljana: IEEE Region 8, Slovenian section IEEE, 1998, pp. 145-148.

[10] Mladenić, D., Grobelnik, M. Feature selection on hierarchy of web documents. *Decision support systems journal*, 2003, vol. 35, pp. 45-87.

[11] Ghani, R., Jones, R., Mladenić, D., Nigam, K., Slattery, S.. Data mining on symbolic knowledge extracted from the Web. In Proc. of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. *KDD-2000 working notes : workshop on text mining,* Boston, MA, USA., 2000, pp. 29-36.

[12] Baxter, D., Klimt, B., Grobelnik, M, Schneider, D.I., Witbrock, M.J., Mladenić, D. Capturing document semantics for ontology generation and document summarization. In *Semantic knowledge management : integrating ontology management, knowledge discovery, and human language technologies*. Berlin; Heidelberg: Springer, cop. 2009, pp. 141-154.

[13] Grobelnik, M, Mladenić, D. Automated knowledge discovery in advanced knowledge management. Journal of knowledge management, 2005a, vol. 9, pp. 132-149.

[14] Grobelnik, M, Mladenić, D. Visualization of news articles. *Informatica (Ljublj.)*, 2004, 28:4, pp. 375-380.

[15] Bhole, A., Fortuna, B., Grobelnik, M, Mladenić, D. Extracting named entities and relating them over time based on Wikipedia. *Informatica (Ljublj.)*, 2007, 31:4, pp. 463-468.

[16] Fortuna, B., Mladenić, D., Grobelnik, M. Visualization of text document corpus. *Informatica (Ljublj.)*, 2005, 29:4, pp. 497-502.

[17] Fortuna, B., Grobelnik, M, Mladenić, D. OntoGen: semi-automatic ontology editor. *Lect. notes comput. sci.*, 2007, vol. 4558, pp. 309-318.

[18] Štajner, T., Rusu, D., Dali, L., Fortuna, B., Mladenić, D., Grobelnik, M. A service oriented framework for natural language text enrichment. *Informatica (Ljublj.)*, 2010, 34:3, pp. 307-313.

[19] Dali, L., Rusu, D., Mladenić, D. Enhanced web page content visualization with Firefox. *Lect. notes comput. sci.*, 2009, lNAI 5782, pp. 718-721.

[20] Rusu, D., Fortuna, B., Grobelnik, M, Mladenić, D. Semantic graphs derived from triplets with application in document summarization. *Informatica (Ljublj.)*, 2009, 33:3, pp. 357-362.

[21] Leskovec, J., Grobelnik, M., Milic-Frayling, N., Learning Sub-structures of Document Semantic Graphs for Document Summarization, In *Proceedings of LinkKDD 2004* Workshop at KDD International conf.

[22] Sipoš, R., Mladenić, D., Grobelnik, M., Brank, J. Modeling common real-world relations using triples

extracted from n-grams, In the *Proc. of The Semantic Web Fourth Asian Conference, ASWC 2009*, Lecture Notes in Computer Science, 2009, 5926, pp. 16-30.

[23] Dali, L., Rusu, D., Fortuna, B., Mladenić, D., Grobelnik, M. *Question answering based on semantic graphs. WWW-2009 Workshop on Semantic Search 2009.*

[24] Bradeško, L., Dali, L., Fortuna, B., Grobelnik, M, Mladenić, D., Novalija, I., Pajntar, B. Contextualized question answering, In Proc. of ITI-2010.

[25] Novalija, I., Mladenić, D., Bradeško, L. OntoPlus : text-driven ontology extension using ontology content, structure and co-occurrence information. *Knowedge.-based systems*, 2011, 24:8, pp. 1261-1276.

[26] Novalija, I., Mladenić, D. Ontology extension towards analysis of business news. *Informatica (Ljublj.)*, 2010, 34:4, pp. 517-522.

[27] Tomašev, N., Mladenić, D. Social network analysis of ontology edit logs. *CIT. J. Comput. Inf. Technol.*, 2010, 18:2, pp. 191-200.

[28] Grobelnik, M, Mladenić, D. Analysis of a database of research projects using text mining and link analysis. In *Data mining and decision support : integration and collaboration*, Boston; Dordrecht; London: Kluwer Academic Publishers, 2003, pp. 157-166.

[29] Grobelnik, M, Mladenić, D., Fortuna, B.. Semantic technology for capturing communication inside an organisation. *IEEE internet computing*, 2009, 13:4, pp. 59-66.

[30] Fortuna, B., Mladenić, D., Grobelnik, M. Visualization of temporal semantic spaces. In *Semantic knowledge management : integrating ontology management, knowledge discovery, and human language technologies*. Berlin; Heidelberg: Springer, cop. 2009, pp. 155-169.

[31] Simperl, E., Thurlow, I., Warren, P., Dengler, F., Davies, J., Grobelnik, M, Mladenić, D., Gomez-Perez, J.M., Ruiz Moreno, C. Overcoming information overload in the enterprise : the active approach. *IEEE internet computing*, 2010, 14:6, pp. 39-46.

[32] Dolinšek, I., Grobelnik, M, Mladenić, D. Managing and understanding context. In *Context and semantics for knowledge management: technologies for personal productivity*. Heidelberg: Springer, 2011, pp. 91-106

[33] Rupnik, J., Muhič, A., Škraba, P. Multilingual Document Retrieval Through Hub Languages, In: *Proceedings of the Fifteenth International Multiconference Information Society 2012*. Ljubljana: Institut Jožef Stefan, 2012.

[34] Štajner, T., Novalija, I., Mladenić, D. Informal sentiment analysis in multiple domains for English and Spanish, In: *Proceedings of the Fifteenth International Multiconference Information Society 2012*. Ljubljana: Institut Jožef Stefan, 2012.

Relational and Semantic Data Mining for Biomedical Research

Nada Lavrač and Petra Kralj Novak
Jožef Stefan Institute and Jožef Stefan International Postgraduate School
Jamova 39, 1000 Ljubljana, Slovenia
Nada.Lavrač@ijs.si, http://kt.ijs.si/nada_lavrac/

Keywords: relational data mining, semantic data mining, biomedicine

Received: December 12, 2012

The paper presents a historical overview of data mining tools and applications in the field of biomedical research, developed at the Department of Knowledge Technologies, Jožef Stefan Institute, Ljubljana, Slovenia. It first outlines subgroup discovery and selected relational data mining approaches, with the emphasis on propositionalization and relational subgroup discovery, which prove to be effective for data analysis in biomedical applications. The core of this paper describes recently developed approaches to semantic data mining which enable the use of domain ontologies as background knowledge in data analysis. The use of the described tools is illustrated on selected biomedical applications.

Povzetek: Prispevek opisuje zgodovinski pregled razvoja orodij rudarjenja podatkov na področju biomedicine.

1 Introduction

Data analysis in biomedical applications aims at extracting potentially new relationships from data and providing insightful representations of detected relationships. Methods for symbolic data analysis are preferred since highly accurate but non-interpretable classifiers are frequently considered useless for medical practice. Subgroup discovery techniques [7, 20] are of interest to biomedical research, as they enable the discovery of patient subgroups from classified patient data, where the induced subgroup descriptions have the form of descriptive rules.

Let us illustrate the results of subgroup discovery in two biomedical applications. In the first application [4], the induced subgroup descriptions suggest how to select individuals for population screening, concerning high risk for coronary heart disease (CHD). One of the discovered rules describes a group of overweight female patients older than 63 years:

$$\begin{aligned}\text{High CHD Risk} \quad \leftarrow \quad & \text{gender = female \&} \\ & \text{age} > 63 \text{ years \&} \\ & \text{body mass index } > 25\mathrm{kg/m}^2\end{aligned}$$

In the second application [16], subgroup describing rules suggest genes that are characteristic for a given cancer type (leukemia), distinguishing it from other 13 cancer types (CNS, lung cancer, etc.):

$$\begin{aligned}\text{Leukemia} \leftarrow\ & \text{KIAA0128 } \textit{is diff\_expressed} \text{ \&} \\ & \text{prostaglandin d2 synthase } \textit{is not diff\_expressed}\end{aligned}$$

The following sections presents the evolution of tools and techniques from inductive logic programming and relational data mining through special purpose systems for bioinformatics to general purpose semantic data mining approaches which enable the use of domain ontologies as background knowledge for data analysis. We conclude by describing new challenges in the focus of our current and future research.

2 Relational data mining for biomedical applications

We first present selected approaches to inductive logic programming (ILP) [11, 9] and relational data mining (RDM) [1] which showed a great potential for biomedical research due to their capacity of using background knowledge in the learning process. From the available background knowledge (encoded as logical facts or rules) and a set of classified examples (encoded as a set of logical facts), an ILP/RDM algorithm derives a hypothesized logic program which explains the positive examples. While ILP focuses on data and background knowledge represented in a logical formalism, RDM assumes that the background knowledge and data are encoded in a unique relational database format. Compared to standard data mining techniques where the input data is typically stored in a single data table (e.g., in Excel), the input to an ILP/RDM algorithm is thus much more complex.

Propositionalization [8] is a RDM approach, which has been applied in several biomedical applications. Consider relational subgroup discovery, an approach effectively implemented in the RSD algorithm [2]. RSD generates descriptive rules as conjunctions of terms which encode background knowledge concepts. RSD performs example-weighting [10] (used in the so-called weighted covering algorithm) and uses the weighted relative accuracy (WRAcc) measure as a heuristic for rule selection. For

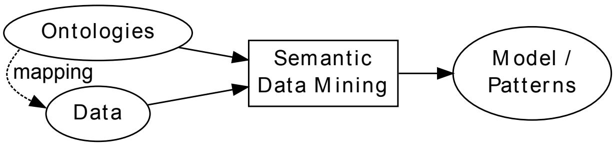


Figure 1: Semantic data mining schema

example, an induced description of gene group A, discovered by RSD for the CNS (central nervous system) cancer class in the problem of distinguishing between 14 cancer types determines group A of differentially expressed genes in CNS as a conjunction of two relational features [17]: $geneGroup(A) \leftarrow f_i(A)\&f_k(A)$, where the two features, $f_i(A)$ and $f_k(A)$, constructed in the propositionalization step of RSD, are:

$f_i(A)$: *interaction(A,B) &*
process(B,'phosphorylation')
$f_k(A)$: *interaction(A,B) &*
process(B,'negative regulation of apoptosis') &
component(B,'intracellular
membrane-bound organelle')

3 Semantic subgroup discovery

The RSD approach to relational subgroup discovery, which was successfully applied to mining microarray data [16], was the first step towards developing a novel data mining methodology, referred to as semantic subgroup discovery. The process of semantic data mining is illustrated in Figure 1.

The proposed semantic data mining methodology enables the generation of descriptive rules explaining the instances of a target class as conjunctions of ontology terms/concepts appearing in bioinformatics ontologies such as the well-known Gene Ontology (GO), KEGG and ENTREZ. An early approach to semantic subgroup discovery, named SEGS, is outlined below, followed by an outline of the SegMine methodology, which upgrades SEGS with a link discovery step.

3.1 Semantic subgroup discovery with SEGS

In many biomedical applications the goal of data analysis is gene set enrichment, i.e., finding groups of genes (gene sets) that are enriched, so that genes in the set are statistically significantly differentially expressed compared to the rest of the genes. Two well-known methods for testing the enrichment of gene sets include Gene Set Enrichment Analysis (GSEA, [15]) and Parametric Analysis of Gene Set Enrichment (PAGE, [6]). Originally, these methods use gene sets that are defined based on prior biological knowledge, e.g., published information about biochemical pathways, coexpression in previous experiments or Gene Ontology (GO) terms.

The RSD subgroup discovery approach combined with gene set enrichment analysis inspired the development of the SEGS algorithm (Searching for Enriched Gene Sets) [17], a specialized algorithm for semantic subgroup discovery for microarray data analysis. SEGS employs semantically annotated knowledge sources Gene Ontology (GO), the Kyoto Encyclopedia of Genes and Genomes (KEGG) and ENTREZ interactions, as background knowledge for semantic subgroup discovery. Based on this background knowledge, SEGS automatically formulates biological hypotheses: rules which define groups of differentially expressed genes. Finally, it estimates the relevance/significance of the formulated hypotheses on experimental microarray data. Compared to GSEA and PAGE, SEGS does not only test existing gene sets (defined by individual GO or KEGG terms), but constructs and tests also new gene sets, constructed by the combination of GO terms, KEGG terms, and also by taking into account the gene-gene interaction data from ENTREZ. The SEGS approach is outlined in Figure 2.

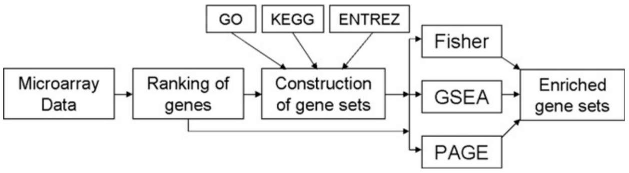


Figure 2: Schematic representation of SEGS.

As it is infeasible to generate all the possible gene set descriptions in the given hypothesis language and evaluate each rule separately in the next step of the procedure, SEGS uses the topology of GO and KEGG to search the hypothesis space in a general-to-specific fashion to be able to reduce the search. Moreover, SEGS includes the ranking of genes (according to their differential expression based on the input microarray experiment) into the gene set generation phase (as shown in Figure 2) and counts the number of differentially expressed genes covered by each generated rule. If the number of covered differentially expressed genes is lower than a predefined threshold, the rule is eliminated and not specialized further, thus pruning large parts of the hypothesis space.

SEGS uses three statistical tests to evaluate the significance of the newly generated gene sets: Fisher's exact test, the GSEA method [15] and the PAGE method [6]. It then uses weights to combine the results of the three statistical tests.

Consider the application domain described in [14, 5], where data instances are gene expression profiles of patients belonging to two cancer classes, AML (acute myeloid leukemia) and ALL (acute lymphoblastic leukemia). Our goal is to uncover interesting patterns that can help to better understand the dependencies between the classes (cancer types) and the attributes (gene expressions values). The rules, shown in Figure 3, were generated from data on gene expression profiles obtained by the Affymetrix HU6800 microarray chip, containing probes for 6,817 genes, for 73 instances of AML or ALL class

Gene Set description	ES
Enriched in ALL	
1. ALL ← int(Func('zinc ion binding')& Comp('chromosomal part')& Proc('interphase of mitotic cell cycle'))	0.60
2. ALL ← Proc('DNA metabolism')	0.59
3. ALL ← int(Func('ATP binding')& Comp('chromosomal part')& Proc('DNA replication'))	0.55
Enriched in AML	
1. AML ← int(Func('metal ion binding')& Comp('cell surface')&	0.54
2. AML ← int(Comp('lysosome'))	0.53
3. AML ← Proc('inflammatory response')	0.51

Figure 3: Enriched gene set descriptions in the AML-ALL domain, together with their enrichment score (ES) [17].

labeled expression vectors. The rules are ranked according to the enrichment score, measuring the enrichment of differential expression of a set of genes, defined by the given conjunction of GO, KEGG and/or ENTREZ interactions.

3.2 SegMine: Combining SEGS and BioMine

The SegMine methodology [12], developed for exploratory analysis of microarray data, is performed through semantic subgroup discovery by SEGS, followed by link discovery and visualization by Biomine [3], an integrated annotated bioinformatics information resource of interlinked data. The SegMine methodology, illustrated in Figure 4, consists of gene ranking, hypothesis/rule generation by the SEGS method for enriched gene set construction, rule clustering, linking of the discovered gene sets to related biomedical databases for link discovery with Biomine, and Biomine sub-graph visualization.

The Biomine service is a valuable addition to SEGS, complementing our semantic subgroup discovery technology by additional explanatory potential due to additional Biomine graph visualization. Biomine is used through its web interface which allows for querying via Biomine named entities, such as a set of GO terms, resulting in a Biomine (sub)-graph, which can be visualized for exploratory purposes. A sample Biomine graph is shown in Figure 5, while the SegMine implementation in the Orange4WS workflow construction and execution platform [13] is shown in Figure 6. In [12], the utility of the SegMine methodology was demonstrated in two microarray data analysis applications: a well-known dataset from a clinical trial in acute lymphoblastic leukemia (ALL), and a dataset about the senescence in human mesenchymal stem cells (MSC). In the analysis of senescence in human stem cells, the use of SegMine resulted in three novel research hypotheses that can improve the understanding of the underlying mechanisms of senescence and identification of candidate marker genes.

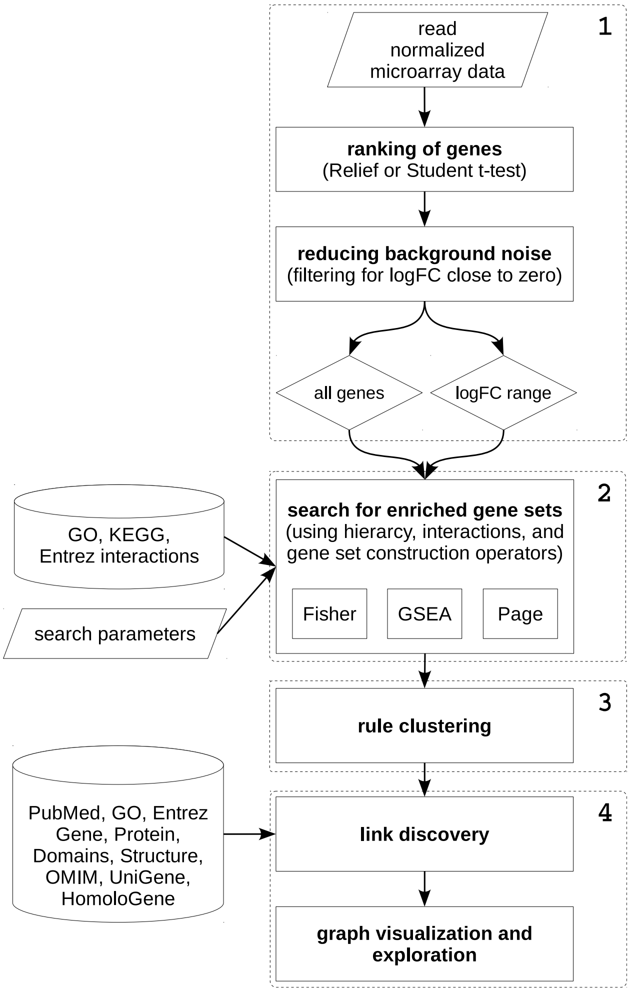

1
read normalized microarray data
ranking of genes
(Relief or Student t-test)
reducing background noise
(filtering for logFC close to zero)
all genes
logFC range
GO, KEGG, Entrez interactions
search parameters
2
search for enriched gene sets
(using hierarcy, interactions, and gene set construction operators)
Fisher
GSEA
Page
3
rule clustering
PubMed, GO, Entrez Gene, Protein, Domains, Structure, OMIM, UniGene, HomoloGene
4
link discovery
graph visualization and exploration

Figure 4: An overview of the SegMine methodology [12] emphasizing its four main steps: (1) data preprocessing, (2) search for differentially expressed gene sets, (3) clustering of rules describing differentially expressed gene sets, and (4) link discovery with graph visualization and exploration.

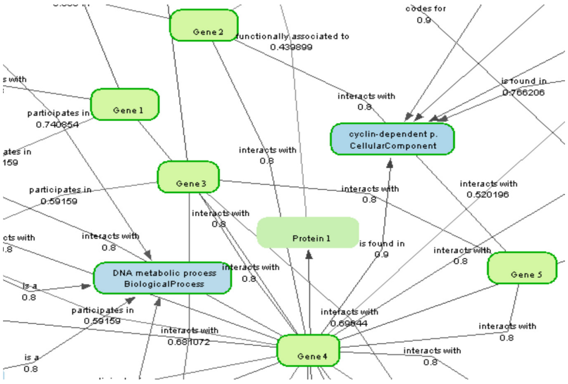


Figure 5: Biomine subgraph related to three genes from the enriched gene set constructed by SEGS.

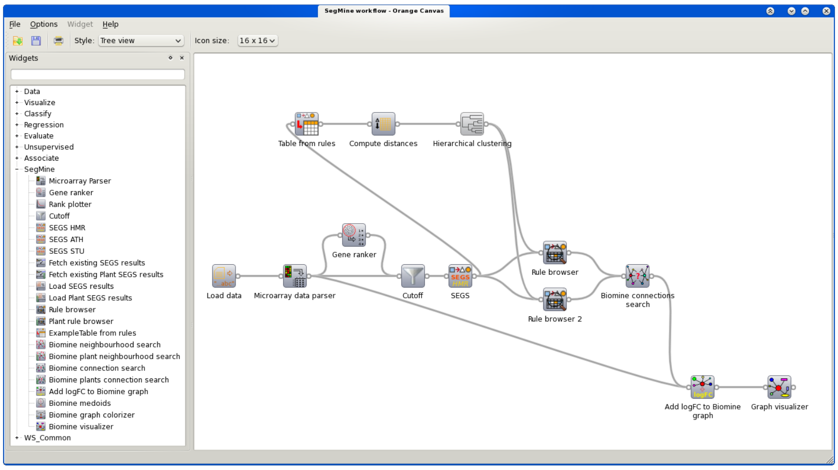


Figure 6: A screenshot of Orange4WS running a workflow of SegMine components [12].

4 General purpose semantic data mining

SEGS was the first special purpose semantic subgroup discovery algorithm developed. Recently, we developed two new general purpose semantic subgroup discovery systems: SDM-SEGS and SDM-Aleph [18]. SDM-SEGS is based on SEGS and can be used to discover subgroup descriptions from ranked data as well as from labeled data with the use of background knowledge in form of OWL ontologies. SDM-Aleph is based on the ILP system Aleph.[1] It was designed to be used in a similar way as SDM-SEGS. Unlike SDM-SEGS which is limited to four ontologies as input and only one additional *interacts* relationship, in SDM-Aleph any number of ontologies and additional relations between the input examples can be specified, which is due to the powerful underlying first-order logic formalism of the ILP system Aleph. SDM-SEGS and SDM-Aleph are implemented within a new semantic data mining toolkit, named SDM-Toolkit [18]. SDM-Toolkit has been made publicly available within the Orange4WS service-oriented data mining environment [13]. In [18], we illustrate the use of SDM-Toolkit tools for biomedical workflow construction and their execution in Orange4WS on the same two biomedical problem domains, ALL and hMSC, which were used in the evaluation of the utility of SegMine [12]. A qualitative evaluation of SDM-SEGS and SDM-Aleph, supported by experimental results and comparisons with SEGS, showed that SEGS and SDM-SEGS are more appropriate for data analysis in biomedical domains where rule specificity is desired, while SDM-Aleph is a more general purpose system, resulting in more general rules of lower precision.

Our recent work [19] also addresses semantic subgroup discovery, but focuses on a problem of explaining patient subgroups (e.g., similar patients, possibly all having a certain, yet unexplored cancer subtype) rather than explaining sets of differentially expressed genes characteristic for patients of a given class (cancer type) as a whole. This research is driven by a real-life problem of breast cancer patient analysis, motivated by the experts' assumption that there are several subtypes of breast cancer.

5 Conclusion

This paper presents a success story of three generations of data mining tools for biomedical research that use different forms of background knowledge. The paper presents the motivation and the evolution of ideas and techniques which were successfully applied in the field of biomedicine. A general-purpose semantic data mining toolkit is also presented, which offers numerous opportunities for applications where background knowledge in available in form of ontologies. All the presented tools are freely available online.

We envision further steps of development for semantic data mining. First, we foresee the usage of linked data as a general source of background knowledge used in semantic data mining. Second, we expect that the mining of knowledge encoded in ontologies will gain priority over mining the empirical data, which will, we believe, become a means of evaluation for the hypotheses generated from background knowledge.

Acknowledgement

We acknowledge numerous collaborators who have significantly contributed to this work: Dragan Gamberger, Filip Železny, Igor Trajkovski, Vid Podpečan, Igor Mozetič, Kristina Gruden, Hannu Toivonen and Anže Vavpetič.

The research presented in this paper was supported by the Slovenian Ministry of Higher Education, Science and Technology (grant no. P-103).

[1] http://www.cs.ox.ac.uk/activities/machinelearning/Aleph/

References

[1] S. Džeroski and N. Lavrač, editors. *Relational Data Mining*. Springer, New York, 2001.

[2] F. Železný and N. Lavrač. Propositionalization-based relational subgroup discovery with RSD. *Machine Learning*, 62(1-2):33–63, 2006.

[3] L. Eronen and H. Toivonen. Biomine: predicting links between biological entities using network models of heterogeneous databases. *BMC bioinformatics*, 13(1):119+, 2012.

[4] D. Gamberger and N. Lavrač. Expert-guided subgroup discovery: methodology and application. *J. Artif. Int. Res.*, 17(1):501–527, 2002.

[5] D. Gamberger, N. Lavrač, F. Železný, and J. Tolar. Induction of comprehensible models for gene expression datasets by subgroup discovery methodology. *Journal of Biomedical Informatics*, 37(4):269–284, 2004.

[6] S.Y. Kim and D. J. Volsky. PAGE: Parametric analysis of gene set enrichment. *BMC Bioinformatics*, 6:144, 2005.

[7] W. Klösgen. Explora: A multipattern and multistrategy discovery assistant. In *Advances in Knowledge Discovery and Data Mining*, pages 249–271. AAAI Press, Menlo Park, 1996.

[8] S. Kramer, N. Lavrač, and P. A. Flach. Propositionalization approaches to relational data mining. In N. Lavrač and S. Džeroski, editors, *Relational Data Mining*, pages 262–286. Springer, 2001.

[9] N. Lavrač and S. Džeroski. *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood, New York, 1994.

[10] N. Lavrač, B. Kavšek, P. Flach, L. Todorovski, and S. Wrobel. Subgroup discovery with CN2-SD. *Journal of Machine Learning Research*, 5:153–188, 2004.

[11] S. Muggleton, editor. *Inductive Logic Programming*. Academic Press, London, 1992.

[12] V. Podpečan, N. Lavrač, I. Mozetič, P. Kralj Novak, I. Trajkovski, L. Langohr, K. Kulovesi, H. Toivonen, M. Petek, H. Motaln, and K. Gruden. SegMine workflows for semantic microarray data analysis in Orange4WS. *BMC Bioinformatics*, 12:416, 2011.

[13] V. Podpečan, M. Zemenova, and N. Lavrač. Orange4WS environment for service-oriented data mining. *Comput. J.*, 55(1):82–98, 2012.

[14] S. Ramaswamy, P. Tamayo, R. Rifkin, S. Mukherjee, C.H. Yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, J. P. Mesirov, T. Poggio, W. Gerald, M. Loda, E. S. Lander, and T. R. Golub. Multiclass cancer diagnosis using tumor gene expression signatures. *Proceedings of the National Academy of Sciences of the United States of America*, 98(26):15149–15154, 2001.

[15] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, and J. P. Mesirov. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proc Natl Acad Sci U S A*, 102(43):15545–15550, 2005.

[16] I. Trajkovski, F. Železný, N. Lavrač, and J. Tolar. Learning relational descriptions of differentially expressed gene groups. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 38(1):16–25, 2008.

[17] I. Trajkovski, N. Lavrač, and J. Tolar. SEGS: Search for enriched gene sets in microarray data. *Journal of Biomedical Informatics*, 41(4):588–601, 2008.

[18] A. Vavpetič and N. Lavrač. Semantic subgroup discovery systems and workflows in the SDM-Toolkit. *The Computer Journal*, 2012.

[19] A. Vavpetič, V. Podpečan, S. Meganck, and N. Lavrač. Explaining subgroups through ontologies. In *PRICAI2012: Proceedings of the National Academy of Science*, volume 7458, pages 625–636, 2012.

[20] S. Wrobel. An algorithm for multi-relational discovery of subgroups. In *Proceedings of the First European Symposium on Principles of Data Mining and Knowledge Discovery*, PKDD '97, pages 78–87, London, UK, UK, 1997. Springer-Verlag.

Explanation and Reliability of Individual Predictions

Igor Kononenko, Erik Štrumbelj, Zoran Bosnić, Darko Pevec, Matjaž Kukar, Marko Robnik-Šikonja
Laboratory for Cognitive Modeling (LKM)
University of Ljubljana, Faculty of Computer and Information Science
Tržaška 25, 1000 Ljubljana, Slovenia
e-mail: lkm@fri.uni-lj.si (name.surname@fri.uni-lj.si); www: lkm.fri.uni-lj.si

Keywords: explanation, reliability, individual predictions, prediction intervals

Received: December 15, 2012

Classification and regression models, either automatically generated from data by machine learning algorithms, or manually encoded with the help of domain experts, are daily used to predict the labels of new instances. Each such individual prediction, in order to be accepted/trusted by users, should be accompanied by an explanation of the prediction as well as by an estimate of its reliability. We have recently developed a general methodology for explaining individual predictions as well as for estimating their reliability. Both, explanation and reliability estimation are general techniques, independent of the underlying model and provide on-line (effective and efficient) support to the users of prediction models.

Povzetek: Razvita je metodologija za razlago napovedi s pripadajočo verjetnostjo pri klasifikacijskih in regresijskih modelov strojnega učenja.

1 Introduction

In a typical machine learning scenario a machine learning algorithm is used to construct a model of the relationships between the input features and the target variable with the purpose to predict the target variable of new, yet unseen instances. Explaining the learned relationships is also an important part of machine learning. Some models, such as additive models or small decision trees, are inherently transparent and require little or no additional post processing [14, 32]. Other, more complex and often better performing models are non-transparent and require additional explanation. Therefore, model-specific explanation methods have been developed for models such as artificial neural networks and SVM. In practice, dealing with several different explanation methods requires undesirable additional effort and makes it difficult to compare models of different types. To address this issue, general explanation methods are used – methods, which treat each model as a black-box and can be used independent of the model's type. Most general explanation methods are based on marginalization of features [17, 35]. This approach is computationally efficient. It is also effective as long as the model is additive (that is, as long as the features do not interact). However, several widely-used machine learning models are not additive, which leads to misleading and incorrect explanations of the importance and influence of features [29]. Unlike existing general explanation methods, our method, described in Section 2, takes into account not only the marginal effect of single features but also the effect of subsets of features.

In supervised learning, one of the goals is to get the best possible prediction accuracy on new and unknown instances. As current prediction systems do not provide enough information about single predictions, experts find it hard to trust them. Common evaluation methods for classification and regression machine learning models give an averaged accuracy assessment of models, and in general, predictive models do not provide reliability estimates for their individual predictions. In many areas, appropriate reliability estimates may provide additional information about the prediction correctness and can enable the user (e.g. medical doctor) to differentiate between more and less reliable predictions. In Section 3 we describe our approaches to estimating the reliability of individual predictions. Finally, in Section 4 we overview directions of current research, carried out in LKM.

2 Explaining individual predictions

The idea behind our method for explaining individual predictions is to compute the contributions of individual features to the model's prediction for a particular instance by decomposing the difference between the model's prediction for the given instance and the model's expected prediction (i.e., the model's predictions if none of the features' values were known). We adopt, with minor modifications, the notation used in [30]. Let $A = A_1 \times A_2 \times \cdots \times A_n$ be our feature space, where each feature A_i is a set of values. Let p be the probability mass function defined on the sample space A. Let $f_c : A \to [0, 1]$ describe the classification model's prediction for class value c. Our goal is a general explanation method which can be used with any model, so no other assumptions are made about f_c. Therefore, we are limited to changing the inputs of the model and observing the outputs.

Let $S = A_1, \ldots, A_n$ be the set of all features. The influence of a certain subset Q of S for classification of a given instance is defined as:

$$\Delta(Q)(x) = E\left[f|Q(x)\right] - E\left[f\right] \tag{1}$$

where $Q(x)$ are values of features in Q for x. The value of the above function for the entire set of features S is exactly the difference between the model's prediction for a given instance and the model's expected prediction that we wish to decompose. Note that we omit the class value in the notation of f. Suppose that for every subset of features Q the value of $\Delta(Q)$ is known. The goal is to decompose $\Delta(S)$ in a way that assigns each feature a fair contribution with respect its influence on the model's prediction. In [29] a solution is proposed that is equivalent to the Shapley value [27] for the coalition game with the n features as players and Δ as the characteristic function. The contribution of the i-th feature is defined as

$$\phi_i(x) = \sum_{Q \subseteq S_{\{i\}}} \frac{|Q|!\,(|Q|-1)!}{|S|!} \delta(Q, x, i), \tag{2}$$

where

$$\delta(Q, x, i) = \Delta(Q \cup \{i\})(x) - \Delta(Q)(x). \tag{3}$$

These contributions have some desirable properties. Their sum for the given instance x equals $\Delta(S)$, which was our initial goal and ensures implicit normalization. A feature that does not influence the prediction will be assigned no contribution. And, features that influence the prediction in a symmetrical way will be assigned equal contributions.

The computation of Eq. 2 is infeasible for large n as the computation time grows exponentially with n. The approximation algorithm is proposed in [31, 30], where we show its efficiency and effectiveness. It is based on the assumption that p is such that individual features are mutually independent. With this assumption and using an alternative formulation of the Shapley value we get a formulation which facilitates random sampling. For a global view on features' contributions, we define the contribution of the feature's value as the expected value of that feature's contribution for a given value. Again, random sampling can be used to estimate the expected value [30]. Let us illustrate the use of the features' local and global contributions using a simple data set with 5 numerical features $A_1, \ldots, A_5$ with unit domains $[0, 1]$. The binary class value equals 1 if $A_1 > 0.5$ or $A_2 > 0.7$ or $A_3 > 0.5$. Otherwise, the class value is 0. Therefore, only the first three features are relevant for predicting the class value. This problem can be modeled with a decision tree. Figure 1 shows explanations for such a decision tree. The global contributions of each feature's values are plotted separately. The black line consists of points obtained by running the approximation algorithm for the corresponding feature and its value corresponding the value on the x-axis. The lighter line corresponds to the standard deviation of the samples across all values of that particular feature and can therefore be interpreted as the overall importance of the feature. The lighter lines reveal that only the first three features are important. The black lines reveal the areas where features contribute towards/against class value 1. For example, if the value of feature A_1 is higher than 0.5 it strongly contributes towards class value being 1. If it is lower, it contributes against class value being 1. For example, the instance $x = (0.47, 0.82, 0.53, 0.58, 0.59)$ belongs to class 1, which the decision tree correctly predicts. The visualization on Figure 2 shows the individual features' contributions for this instance. The last two features have a 0 contribution. The only feature value that contributes towards class = 1 is $A_2 = 0.82$, while the remaining two features' values have a negative contribution.

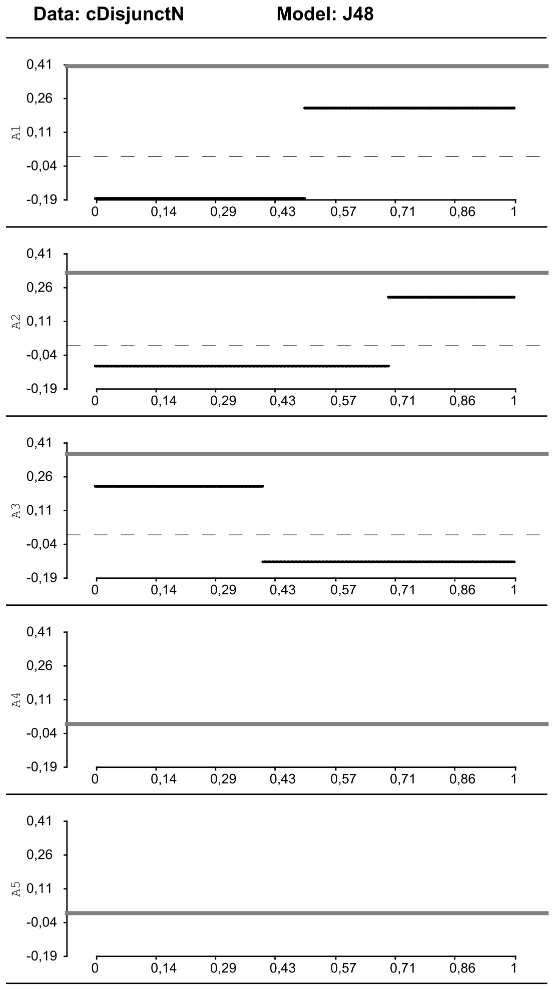


Figure 1: Explanation of a decision tree.

We have successfully applied this research to post-processing tools for breast cancer recurrence prediction [31], maximum shear stress prediction from hemodynamic simulations [6], and businesses' economic results precision [24].

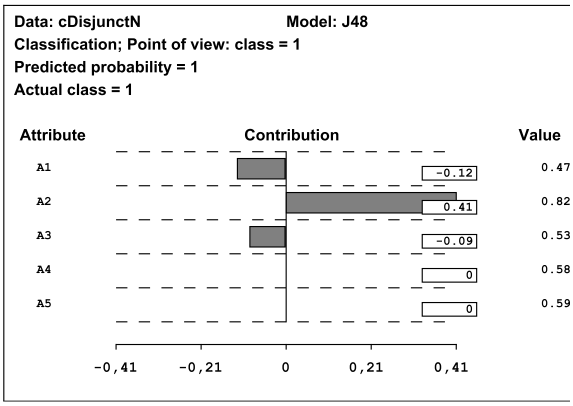


Figure 2: Visualization of the individual features' contributions for particular instance.

2.1 Efficient RBF network explanation

A lot of effort has been invested into increasing the interpretability of complex models such as artificial neural networks [2, 21]. Presented explanations method can be used with any type of classification model including neural networks. Below we show how for a special type of neural networks the approximations required to estimate Eq. 1 in [31, 30] can be especially efficient and require no relearning of the classification model [26]

The probabilistic radial basis function network classifier (PRBF) is an effective but non-transparent prediction model [11, 33]. The PRBF is a special case of the RBF network [3]. It adopts a cluster interpretation of the base functions, where each cluster can generate observations for any class. Therefore, it is a generalization of the Gaussian mixture model [3, 20]. We show how the PRBF can be efficiently explained with two presented explanation methods [25, 29].

Consider a classification problem with c classes y_k ($k = 1, \ldots, c$) and input instances $x = (A_1, \ldots, A_a)$. For this problem, the corresponding PRBF classifier has a inputs and c outputs, one for each class. Each output provides and estimate of the probability density $p(x|y_k)$ of the corresponding class y_k. Assume that we have M components (hidden units), each one computing a probability density value $f_j(x)$ of the input x. In the PRBF network all component density functions $f_j(x)$ are utilized for estimating the conditional densities of all classes by considering the components as a common pool [33]. Thus, for each class a conditional density function $p(x|y_k)$ is modeled as a mixture model of the form:

$$p(x|y_k) = \sum_{j=1}^{M} \pi_{jk} f_j(x), \quad k = 1, \ldots, c, \tag{4}$$

where the mixing coefficients π_{jk} are probability vectors; they take positive values and satisfy the following constraint:

$$\sum_{j=1}^{M} \pi_{jk} = 1, \quad k = 1, \ldots, c. \tag{5}$$

Once the outputs $p(x|y_k)$ have been computed, the class of data point x is determined using the Bayes rule, i.e. x is assigned to the class with maximum posterior $p(y_k|x)$ computed by

$$p(y_k|x) = \frac{p(x|y_k)P_k}{\sum_{\ell=1}^{c} p(x|y_\ell)P_\ell} \tag{6}$$

The class priors P_k are usually computed as the percentage of training instances belonging to class y_k.

In the following, we assume the Gaussian component densities of the general form:

$$\begin{aligned} f_j(x) &= \frac{1}{(2\pi)^{a/2}|\Sigma_j|^{1/2}} \cdot \\ & \exp\left\{-\frac{1}{2}(x-\mu_j)^T \Sigma_j^{-1}(x-\mu_j)\right\} \end{aligned} \tag{7}$$

where $\mu_j \in \Re^a$ represents the mean of component j, while Σ_j represents the corresponding $a \times a$ covariance matrix. The whole adjustable parameter vector of the model consists of the mixing coefficients π_{jk} and the component parameters (means μ_j and covariances Σ_j).

A notable convenient characteristic of the Gaussian distribution is the marginalization property: if the joint distribution of a set of random variables $S = \{A_1, \ldots, A_a\}$ is Gaussian with mean μ and covariance matrix Σ, then for any subset A of these variables, the joint distribution of the subset $Q = S - A$ of the remaining variables is also a Gaussian. The mean $\mu_{\setminus A}$ of this Gaussian is obtained by removing from μ the components corresponding to the variables in subset A and covariance matrix $\Sigma_{\setminus A}$ is obtained by removing the rows and columns of Σ corresponding to the variables in subset A. Therefore, if we know the mean and covariance of the joint distribution of a set of variables, we can immediately obtain the distribution of any subset of these variables.

The basis for the explanation is $\Delta(Q)(x)$ from Eq. 1, so for a given instance x we need the prediction using the set of all features $\{1, 2, ...a\}$, the prediction using only a subset of features Q, and the prediction using the empty set of features $\{\}$. Let these predictions be $h(x_{\{1,2,...a\}})$, $h(x_Q)$, and $h(x_{\{\}})$, respectively. Since an instance x is fixed in explanation, for readability sake we omit it from expressions below, but remain aware that the dependence exists.

For an input $x = (A_1 = v_1, \ldots, A_a = v_a)$ each output $p(x|y_k)$, $k = 1, \ldots, c$ of the PRBF is computed as a mixture of Gaussians:

$$p(x|y_k) = \sum_{j=1}^{M} \pi_{jk} \mathcal{N}(x; \mu_j, \Sigma_j) \tag{8}$$

Consequently, based on the marginalization property of the Gaussian distribution, for a subset of features Q we get

$$h(x_Q|y_k) = \sum_{j=1}^{M} \pi_{jk} \mathcal{N}(x_Q; \mu_{jQ}, \Sigma_{jQ}) \tag{9}$$

where μ_{jQ} and Σ_{jQ} are obtained by retaining only the elements from Q in μ_j and Σ_j. We obtain $h(y_k|x_Q)$ as

$$h(y_k|x_Q) = \frac{h(x_Q|y_k)P_k}{\sum_{\ell=1}^{c} h(x_Q|y_\ell)P_\ell} \tag{10}$$

and use it in the approximation via Eqs. (2) and (2).

As a consequence, with PRBF models approximations become more efficient and exact as no approximation of the conditional predictions is needed. Classification with a subset of features requires only a mask which selects appropriate elements from μ and appropriate rows and columns from Σ matrix.

3 Reliability of individual predictions

Because model independent approaches are general, they cannot exploit specific parameters of a given predictive model, but rather focus on influencing the parameters that are available in the standard supervised learning framework (e.g. a learning set and attributes). We expect from reliability estimators to give insight into the prediction error and we expect to find positive correlation between the two. There exist two distinct types of estimates – point estimators are covered in the first subsection and interval estimators are presented in the latter.

3.1 Point estimators

The first two algorithms described in this section are based on Reverse transduction and Local sensitivity analysis and follow the same transductive idea, though the first is applicable only to classification, and the second to regression models. Other algorithms are general and need only minor adaptations when converted from regression to classification models, or vice versa.

3.1.1 Reverse transduction and Local sensitivity analysis

Transduction can be used in the reverse direction, in the sense of observing the model's behavior when inserting modified learning instances of the unseen and unlabeled instance [15, 16]. Let x represent an instance and let y be its class label. Let us denote a learning instance with known label y as (x, y) and let $(x, \_)$ be an unseen and unlabeled instance, for which we wish to estimate the reliability of an initial prediction K. It is possible to create a modified learning instance by inserting the unseen instance $(x, \_)$ into the learning set and label it with the same (first) or different class (second best or last) as predicted by the initial model. The distance between the initial probability vector and that from the rebuilt model forms the reliability estimate. The three reliability estimators for classification models derived from three instances are labeled $TRANS_{first}$, $TRANS_{second}$ and $TRANS_{last}$.

In the regression the procedure is similar except that the predicted label is first slightly corrupted: $y = K + \delta$ and then we insert the newly generated instance (x, y) into the learning set and rebuild the predictive model. We define $\delta = \epsilon(l_{\max} - l_{\min})$, where ϵ expresses the proportion of the distance between largest ($l_{\max}$) and smallest ($l_{\min}$) prediction. In this way we obtain a sensitivity model, which computes a sensitivity estimate K_ϵ for the instance $(x, \_)$. To widen the observation window in local problem space and make the measures robust to local anomalies, the reliability measures use estimates from the sensitivity models, gained and averaged across different values of $\epsilon \in E$. For more details see [4]. Let us assume we have a set of nonnegative ϵ values $E = \epsilon_1, \epsilon_2, \ldots, \epsilon_{|E|}$. We define the estimates as follows:

- Estimate $SAvar$
  (Sensitivity Analysis local variance):

$$SAvar = \frac{\sum_{\epsilon \in E} (K_\epsilon - K_{-\epsilon})}{|E|} \tag{11}$$

- Estimate $SAbias$
  (Sensitivity Analysis local bias):

$$SAbias = \frac{\sum_{\epsilon \in E} (K_\epsilon - K) + (K_{-\epsilon} - K)}{2\,|E|} \tag{12}$$

3.1.2 Bagging variance

In related work, the variance of predictions in the bagged aggregate of artificial neural networks has been used to estimate the reliability of the aggregated prediction [13, 9]. The proposed reliability estimate is generalized to other models [5].

Let $K_i, i = 1 \ldots m$, be the predictor's class probability distribution for a given unlabeled example $(x, \_)$. Given a bagged aggregate of m predictive models, where each of the models yields a prediction $B_k, k = 1 \ldots m$, the reliability estimator $BAGV$ is defined as the variance of predictions' class probability distribution:

$$BAGV = \frac{1}{m} \sum_{k=1}^{m} \sum_{i} (B_{k,i} - K_i)^2 . \tag{13}$$

The algorithm uses a bagged aggregate of 50 predictive models as default.

3.1.3 Local cross-validation

The LCV (Local Cross-Validation) reliability estimate is computed using the local leave-one-out (LOO) procedure. Focusing on the subspace defined by k nearest neighbors, we generate k local models, each of them excluding one of the k nearest neighbors. Using the generated models, we compute the leave-one-out predictions $K_i, i = 1 \dots k$, for each of the k excluded nearest neighbors. Since the labels $C_i, i = 1 \dots k$, of the nearest neighbors are known, we are able to calculate the local leave-one-out prediction error as the average of the nearest neighbors' local errors:

$$LCV = \frac{1}{k} \sum_i |C_i - K_i| \,. \tag{14}$$

In experiment, the parameter k was assigned to one tenth of the size of the learning set.

3.1.4 Local error modeling

Given a set of k nearest neighbors, where C_i is the true label of the i-th nearest neighbor, the estimate CNK ($C_{\mathrm{N}eighbors} - K$) is defined as the difference between average label of the k nearest neighbors and the instance's prediction K:

$$CNK = \frac{\sum_i C_i}{k} - K. \tag{15}$$

CNK is not a suitable reliability estimate for the k-nearest neighbors algorithm, as they both work by the same principle. In our experiments we used $k = 5$. In regression tests, *CNK-a* denotes the absolute value of the estimate, whereas *CNK-s* denotes the signed value.

3.1.5 Density based estimation

This approach assumes that an error is lower for predictions in denser problem subspaces, and higher for predictions in sparser subspaces. Note that it does not consider the learning instances' labels. The reliability estimator DENS is a value of the estimated probability density function for a given unlabeled example.

3.1.6 Applications of point estimators

The proposed reliability estimation methodology has been implemented in several applications of machine learning and data mining in areas of medicine, financial applications, economy. In these application domains, the bare regression predictions have been supplemented with a suitable reliability estimator (the best performing among the proposed was chosen after the initial evaluation study), which helped the users of predictive systems gain greater insight into the trustworthiness of individual predictions. The most interesting of these applications are:

- breast cancer recurrence prediction problem for the Institute of Oncology, Ljubljana [31]. The collected dataset included data for 1023 patients, for whom the task was to predict potential cancer recurrence for the period of future 20 years. Given that a predictive timespan is so wide and that it reaches into the far future, the difficulty of the predictive problem was alleviated by implementing reliability estimators,
- electricity load forecast prediction problem for a particular European country [6]. Two regression models were implemented, the neural network and the k nearest neighbors algorithm and their predictions were corrected using the reliability estimator CNK. The results showed that the accuracy of corrected predictions using CNK is favorable in comparison to accuracy of predictions corrected with referential Kalman filter method.
- predicting maximum wall shear stress magnitude and its coordinates in the model of human carotid artery bifurcation [7]. Since one of the most common causes of human death is stroke, a medical expert system could significantly aid medical experts to detect hemodynamic abnormalities. Based on the acquired simulated data, we applied several prediction reliability estimators and the model explanation methodology that provided a useful tool for the given problem domain.

3.2 Interval estimators

Here we focus on standard regression problems where the data follows some continuous function and is somehow corrupted with additive noise – $y_i = f(\vec{x_i}) + \epsilon(\vec{x_i})$.

Confidence intervals are concerned with the accuracy of the model's estimate $\hat{y}_i = \hat{y}(\vec{x_i})$ of the true but unknown function $f(\vec{x_i})$. They strive to capture the distribution of the quantity $f(\vec{x_i}) - \hat{y}_i$, however in applications, it is more informative to quantify the accuracy of the model's output with respect to the realized observations y_i. Prediction intervals (PIs) should capture the distribution of individual future points and are concerned with the quantity $y_i - \hat{y}_i$. Expanding the first term, $y_i - \hat{y}_i = f(\vec{x_i}) + \varepsilon(\vec{x}) - \hat{y}_i = [f(\vec{x_i}) - \hat{y}_i] + \varepsilon(\vec{x})$, we see that the PI should enclose the confidence interval. In real world applications, PIs are more practical than confidence intervals because the former are concerned with the accuracy with which it is possible to predict the observed value itself, and not only with the accuracy of the estimate of the true conditional mean.

3.2.1 Bootstrap and maximum likelihood

The first family of methods is based on the idea of explaining the total prediction error as a sum of the model's error and the error caused by noise inherent to the data [13]. Noise in data and non-uniform distribution of examples represent a challenge for learning algorithms, leading to different prediction accuracies in different parts of the problem space. This component is called the data noise variance and is labeled as σ_d^2. Apart from the distribution of learning examples there are also other causes that influence the accuracy of prediction models and these factors form the component called the model uncertainty variance, σ_m^2. The two components are assumed to be independent of each other and their sum is the total prediction variance: $\sigma^2 = \sigma_m^2 + \sigma_d^2$.

The most straightforward approach was formed in [34]. Given a fixed model with available learning algorithm and training set, the reinterpretation of the method goes as follows. To estimate $\sigma_m^2(\vec{x})$, bagging is done with the training data, using the model at hand. Confidence intervals are formed by assuming the normal distribution and calculating $\hat{\sigma}_m^2(\vec{x})$, the variance of the bagged predictions. Then a radial basis function network (RBFN) is trained on the residuals of the bagging predictions on the training dataset and is used to provide the estimate $\hat{\sigma}_d^2(\vec{x})$. Assuming normal distribution, $\hat{\sigma}^2(\vec{x})$ equals to $\hat{\sigma}_m^2\ (\vec{x})+\hat{\sigma}_d^2\ (\vec{x})$, so the PI is $\hat{y}_{\text{bag}}(\vec{x}) \pm z_{\alpha/2}\hat{\sigma}(\vec{x})$ where $\hat{y}_{\text{bag}}$ is the bagged prediction of the model. This method is labeled as *BagMLa*.

Generalizing the method from [13], we label it *BagMLb*. Here, $\hat{\sigma}_m^2(\vec{x})$ is again obtained by calculating the variance of the bagged model. Estimation of $\sigma_d^2(\vec{x})$ is done by a RBFN trained on the out–of–sample bagged residuals. Assuming a locally normal distribution, the *BagMLb* PI is $\hat{y}(\vec{x}) \pm z_{\alpha/2}\hat{\sigma}(\vec{x})$ where $\hat{y}$ is the model's prediction. Keen readers would notice that *BagMLa* PIs are centered on $\hat{y}_{bag}(\vec{x})$ but with *BagMLb*, the PIs are centered on $\hat{y}(\vec{x})$. The idea is that $\hat{y}_{\text{bag}}(\vec{x})$ provides a more stable estimate of the true function $f(\vec{x})$ than $\hat{y}(\vec{x})$ does.

3.2.2 Local neighborhood

The second family assumes that samples, which are close in the attribute domain, will behave similarly. These approaches estimate the conditional prediction variance with use of the local neighborhoods for direct estimation of σ^2. As such, they can be applied even in cases where there is no access to the learning algorithm or the bootstrap procedure would be just too time consuming.

Adopting the idea from [28], we implemented k-means clustering on the training data. The number of clusters is defined with the common heuristic $k = \sqrt{n/2}$, where n is the size of the training set. *LNcl* PIs are constructed for each cluster directly from its empiric distribution of the residuals, by taking the appropriate percentiles, i.e. the 2.5 and 97.5 percentiles for 95% PIs. For an unseen example, the PI is defined by that of the nearest cluster.

The nearest neighbor algorithm can be used to construct PIs in the following way. First, signed residuals are obtained from the training set. From the nearest neighbors residuals, their mean $\bar{r}$ serves for bias correction and their standard deviation gives us $\hat{\sigma}^2(\vec{x})$. The number of used neighbors is relative to the size of the data set. With method *LN5*, the size of the neighborhood is 5% of the total population. Our second variant *LN100* is computationally even simpler, as it covers the whole (100%) population and is therefore equivalent to analytic methods that assume constant variance. Here we assume the Student's *t*-distribution with degrees of freedom equal to the number of neighbors, so the PIs take the form $\hat{y} + \bar{r} \pm t_{\alpha/2} \cdot \hat{\sigma}^2(\vec{x})$.

Natural progression would suggest use of adaptive neighborhood procedures. Random forests [8] were reinterpreted as a weighted mean of the observed response variables in [18] and generalized to Quantile Regression Forests in [19]. In this case, trees in these ensembles are not pruned and the values of all observations are kept in the leafs. This preserves information about the underlying distribution and makes estimation of conditional quantiles possible. The conditional distribution of residuals given $X = x$ can be written as $F(r|X = x) = P(R \leq r|X = x) = E(1_{R \leq r}|X = x)$, where $1_{R_i \leq r}$ is an indicator variable with value 1 if $R_i \leq r$ and 0 otherwise. This expression is approximated by the weighted mean over the indicator variables and the estimator is

$$\hat{F}(r|X = x) = \sum_{i=1}^{n} w_i(x) 1_{R_i \leq r}.$$

The weights w_i are averaged weights from the collection of trees and a weight in the individual tree is the inverse of the number of observations in the corresponding leaf. Weights sum to one, so they represent the distribution of possible values for the response variable. When a new sample is dropped trough the collection of trees, the weights are obtained. The corresponding r_i values are sorted in ascending order and for 95% PIs, we need to find r_l for which $\sum_{i=1}^{l} w_i \geq 0.025$ and r_u for which $\sum_{i=1}^{u} w_i \geq 0.975$ hold. The interval $[\hat{y} + r_l, \hat{y} + r_u]$ is our sought PI.

3.2.3 Evaluation of interval estimators

In Figure 3 we can see the results on a collection of 36 real-world and artificial datasets. It shows how the lowest achieved PICP values and those closest to the target PICP value are distributed among the used methods. In this regard, *QRF* seems best.

If two methods achieve the same PICP, the one with a lower RMPI value would have more narrow intervals and should be regarded as the better option. Small RMPI values are mostly achieved by *LN* methods due to their nature of producing optimal intervals and extremely low RMPI values are usually accompanied with zero prediction coverage. Largest RMPI values were produced by *BagML* methods, though the corresponding PICP values are 1.0. This means that all test examples enclosed on the account of very wide PIs. According to the figure, *QRF* did not produce any extreme RMPI values and is even in this respect the best method to use.

4 Current research directions

Our current research is focused on several topics. One research area is related to evaluation of ordinal features in the context of surveys and customer satisfaction in marketing, learning of imbalanced classification problems, and applying evolutionary computation to data mining (focused on using ant colony optimization for rule learning). Somewhat different area is spatial data mining of multi-level directed graphs with applications in oceanography [22].

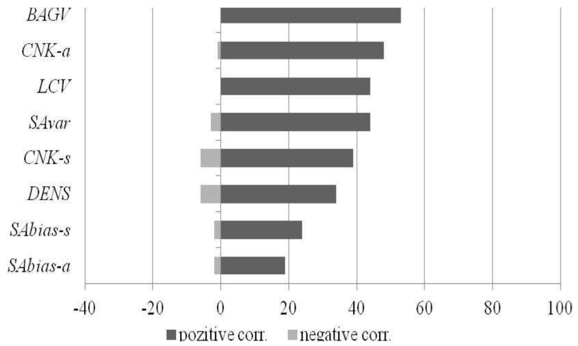


Figure 3: Distribution of experiments (their percentage) of achieved PICP values closest to the target (positive) and furthest (negative) compared to the distribution of the smallest (positive) and largest (negative) achieved RMPI values among the methods.

We recently restored our research in inductive logic programming (ILP) which is focused on employing background knowledge analysis for search space reduction in bottom-up ILP. Yet another branch of research is profiling of web users in an online advertising network together with heuristic search methods in clickstream mining [23], employing algebraic methods, particularly matrix factorization for text summarization, and modeling the progression of team sports matches and evaluation of the individual player's contributions. We also continue our long-term research in medical problems, particularly detection of (non)-ischaemic episodes in ECG signals [12]. The research, described in this paper, continues by adapting the reliability estimators and the explanation methodology for online learning (data streams).

Acknowledgement

Our research is supported by ARRS (Slovenian Research Agency). We thank all former and current members of Laboratory for Cognitive Modeling (LKM) at Faculty of Computer and Information Science, University of Ljubljana, for their contributions.

References

[1] A. Asunction, D. J. Newman (2007) *UCI ML repository*.

[2] A. S. d'Avila Garcez, K. Broda, D. M. Gabbay (2001) Symbolic knowledge extraction from trained neural networks: a sound approach, *Artificial Intelligence 125(1-2)*, pp. 155–207.

[3] C. M. Bishop (1995) *Neural Networks for Pattern Recognition*, Oxford University Press.

[4] Z. Bosnić, I. Kononenko (2007) Estimation of individual prediction reliability using the local sensitivity analysis, *Applied Intelligence, 29(3)*, pp. 187–203.

[5] Z. Bosnić, I. Kononenko (2008) Comparison of approaches for estimating reliability of individual regression predictions, *Data & Knowledge Engineering, 67(3)*, pp. 504–516.

[6] Z. Bosnić, P.P. Rodrigues, I. Kononenko, J. Gama (2011) Correcting streaming predictions of an electricity load forecast system using a prediction reliability estimate, *Man-machine interactions 2*, Springer, pp. 343–350.

[7] Z. Bosnić, P. Vračar, M.D. Radović, G. Devedžić, N. Filipović, I. Kononenko (2012) Mining data from hemodynamic simulations for generating prediction and explanation models, *IEEE trans. inf. technol. biomed., vol. 16, no. 2*, pp. 248–254.

[8] L. Breiman (2001) Random Forests, *Machine Learning – Volume 45*, pp. 5–32.

[9] J. Carney, P. Cunningham (1999) Confidence and prediction intervals for neural network ensembles, *Proceedings of IJCNN'99, The International Joint Conference on Neural Networks*, Washington, USA, pp. 1215–1218.

[10] Department of Statistics at Carnegie Mellon University (2005) *Statlib – Data, software and news from the statistics community* Retrieved from http://lib.stat.cmu.edu/

[11] C. Constantinopoulos, A. Likas (2006) An incremental training method for the probabilistic RBF network, *IEEE Trans. Neural Networks 17(4)*, pp. 966–974.

[12] J. Faganeli Pucer, J. Demšar, M. Kukar (2012) Classification of Ischaemic Episodes with ST/HR Diagrams, *Quality of Life through Quality of Information: Proceedings of MIE2012*, IOS Press, pp. 1108–1111.

[13] T. Heskes (1997) Practical confidence and prediction intervals, *Advances in Neural Information Processing Systems, 9*, The MIT Press, pp. 176–182.

[14] A. Jakulin, M. Možina, J. Demšar, I. Bratko, B. Zupan (2005) Nomograms for visualizing support vector machines, *KDD '05: ACM SIGKDD*, pp. 108–117.

[15] M. Kukar, I. Kononenko (2002) Reliable classifications with machine learning, *Proceedings of Machine Learning: ECML-2002*, Helsinki, Finland, Springer, pp. 219–231.

[16] M. Kukar (2006) Quality assessment of individual classifications in machine learning and data mining, *Knowledge and information systems, vol. 9, no. 3*, pp. 364–384.

[17] V. Lemaire, V. Feraud, N. Voisine (2008) Contact personalization using a score understanding method, *International Joint Conference on Neural Networks (IJCNN)*, pp. 649–654.

[18] Y. Lin, Y. Jeon (2006) Random Forests and Adaptive Nearest Neighbors, *Journal of the American Statistical Association – Volume 101*, pp. 578–590.

[19] N. Meinshausen (2006) Quantile Regression Forests, *Journal of Machine Learning Research – Volume 7*, pp. 983–999.

[20] G. McLachlan and D. Peel (2000) *Finite Mixture Models*, John Wiley & Sons.

[21] V. Palade, C.-D. Neagu, and R. J. Patton (2001) Interpretation of trained neural networks by rule extraction, *Proceedings of the International Conference, 7th Fuzzy Days on Computational Intelligence, Theory and Applications*, London, UK, Springer-Verlag, pp. 152–161.

[22] B. Petelin, V. Malačič, A. Malej, M. Kukar, I. Kononenko (2012) Multi-level association rules and directed graphs for the Lagrangian analysis of the Mediterranean ocean forecasting system (MFS), *Geophys. res. abstr., vol. 14*, pp. 4877.

[23] M. Poženel, V. Mahnič, M. Kukar (2011) Separation of interleaved Web sessions with heuristic search, *Proceedings of ICDM 2010*, IEEE Computer Society, pp. 411–420.

[24] M. Pregeljc, E. Štrumbelj, M. Mihelčič, I. Kononenko (2012) Learning and Explaining the Impact of Enterprises' Organizational Quality on their Economic Results, *Intelligent Data Analysis for Real-Life Applications: Theory and Practice*, pp. 228–248.

[25] M. Robnik Šikonja, I. Kononenko (2008). Explaining classifications for individual instances. *IEEE Transactions on Knowledge and Data Engineering*, 20(5):589-600.

[26] M. Robnik Šikonja, I. Kononenko, E. Štrumbelj (2012). Quality of classification explanations with PRBF. *Neurocomputing* 96:37–46

[27] L.S. Shapley (1953) A Value for n-person Games, *volume II of Contributions to the Theory of Games*, Princeton University Press, pp. 307–317.

[28] D. L. Shrestha, D. P. Solomatine (2006) Machine learning approaches for estimation of prediction interval for the model output, *Neural Networks 19(2)*, pp. 225–235.

[29] E. Štrumbelj, I. Kononenko (2010) An Efficient Explanation of Individual Classifications using Game Theory, *Journal of Machine Learning Research 11*, pp. 1–18.

[30] E. Štrumbelj, I. Kononenko (2011) A General Method for Visualizing and Explaining Black-Box Regression Models, *International Conference on Adaptive and Natural Computing Algorithms (ICANNGA) 2011*, pp. 21–30.

[31] E. Štrumbelj, Z. Bosnić, I. Kononenko, B. Zakotnik, C. Grašič (2010) Explanation and reliability of prediction models: the case of breast cancer recurrence, *Knowledge and information systems, vol. 24, no. 2*, pp. 305–324.

[32] B. Poulin, R. Eisner, D. Szafron, P. Lu, R. Greiner, D.S. Wishart, A. Fyshe, B. Pearcy, C. MacDonell, J. Anvik (2006) Visual explanation of evidence in additive classifiers, *Proceedings of Innovative Applications of Artificial Intelligence – Volume 2*, AAAI Press, pp. 1822–1829.

[33] M. K. Titsias and A. Likas (2001) Shared kernel models for class conditional density estimation, *IEEE Trans. Neural Networks 12(5)*, pp. 987–997.

[34] A. Zapranis, E. Livanis (2005) Prediction intervals for neural network models, *Proceedings of the 9th WSEAS International Conference on Computers*, pp. 76:1–76:7.

[35] A. Zien, N. Krämer, S. Sonnenburg, G. Rätsch (2009) The feature importance ranking measure, *ECML PKDD 2009, Part II*, Springer-Verlag, pp. 694–709.

DEX Methodology: Three Decades of Qualitative Multi-Attribute Modeling

Marko Bohanec and Martin Žnidaršič
Jožef Stefan Institute, Department of Knowledge Technologies, Jamova 39, 1000 Ljubljana, Slovenia
E-mail: marko.bohanec@ijs.si, http://kt.ijs.si/

Vladislav Rajkovič
University of Maribor, Faculty of Organisational Sciences, Kidričeva cesta 55a, 4000 Kranj, Slovenia

Ivan Bratko and Blaž Zupan
University of Ljubljana, Faculty of Computer and Information Science, Tržaška 25, 1000 Ljubljana, Slovenia

Keywords: decision support, multi-criteria decision making, qualitative multi-attribute modeling

Received: December 11, 2012

DEX is a qualitative multi-attribute decision modeling methodology that integrates multi-criteria decision modeling with rule-based expert systems. The method was conceived in 1979. Since, it has been continuously developed and implemented in a wide range of computer programs that have been applied in hundreds of practical decision-making studies. Here we present its main methodological concepts, contributions to the theory and practice of decision support, and outline a history of its development and evolution.

Povzetek: V prispevku predstavljamo najpomembnejše koncepte metodologije DEX, njene prispevke k teoriji in praksi podpore pri odločanju ter orišemo njen zgodovinski razvoj.

1 Introduction

DEX is a qualitative decision support methodology for the evaluation and analysis of decision alternatives. Conceived more than thirty years ago, the methodology has a long history of scientific, technical and practical contributions. It represents a pioneering approach of combining the "classical" numerical multi-criteria decision modeling with rule-based expert systems. This approach led to a development of new algorithms and techniques for acquisition and representation of decision knowledge and evaluation and analysis of decision alternatives. DEX was implemented in three generations of software – called DECMAK, DEX and DEXi – and embedded into many other computer programs and systems. It was used in hundreds of practical applications, nationally and internationally. Despite its age, DEX is still very much alive: it is actively used in international projects and cited in international scientific publications, it is taught in schools, there are ongoing new developments and strong plans for future work. Taking all this into account, DEX can be rightly considered an important long-term achievement of Slovenian research in artificial intelligence and decision support.

2 On origins and evolution of DEX

The foundations of what eventually became DEX were set up in Durham, UK, by Efstathiou and Rajkovič (1979). Influenced by fuzzy set theory, they proposed to use words rather than numbers in decision models. They proposed a tabular representation of utility relations, one of the key concepts of DEX methodology. Further development (Figure 1) continued in Slovenia, mainly through collaboration of Vladislav Rajkovič and Marko Bohanec. In the 1980's, the methodology was called DECMAK (Bohanec et al., 1983). The original idea was conceptually extended to cope with hierarchies of attributes (Rajkovič, Bohanec, 1980) and to facilitate the acquisition and explanation of decision knowledge (Rajkovič, Bohanec, 1988a; Rajkovič et al., 1988). The approach was successfully used in several important applications, such as evaluation of computer systems (Bohanec et al., 1983), personnel management (Rajkovič et al., 1988) and enrolment into nursery schools (Olave et al., 1989).

The name DEX (Decision EXpert) was coined in 1987 when the method was implemented as an expert system shell for decision making (Bohanec, Rajkovič, 1990). This was a state-of-the-art implementation of the complete methodology. In the 1990's, DEX contributed to solution of complex decision making problems in industry (Bohanec, Rajkovič, 1999), health-care (Bohanec et al., 2000a), project evaluation (Bohanec et al., 1995), housing (Bohanec et al., 2001), and sports (Bohanec et al., 2000b). An important related achievement was also HINT, a method for automatic problem decomposition (Zupan et al., 1999). Used as a machine learning algorithm, HINT is capable of developing DEX models from data.

The third distinctive period begun in year 2000 with the implementation of DEXi (Jereb et al., 2003), a stripped-down and user-friendly computer program aimed primarily at education. This paved the DEX's way into Slovenian secondary schools and universities

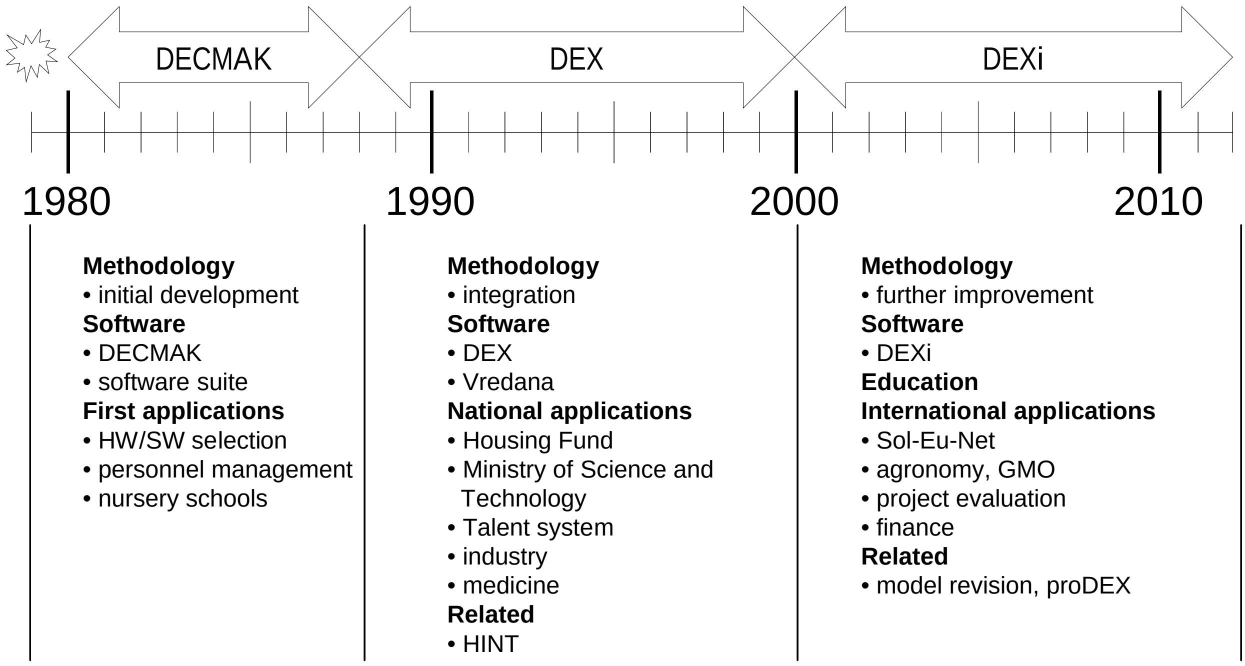


Figure 1: Timeline of DEX development and main achievements.

(Krapež, Rajkovič, 2003). In spite of its simplicity, DEXi turned out extremely useful even for most difficult decision-making tasks. Some outstanding international applications included European projects *Sol-Eu-Net* on data mining and decision support integration (Mladenić et al., 2003), *Healthreats* on health threats and crises management (Žnidaršič et al., 2009), *ECOGEN*, *SIGMEA* and *Co-Extra* on genetically modified crops (Bohanec et al., 2008; Žnidaršič et al., 2008, Bohanec et al., 2013), and *e-LICO* on data mining workflows (Žnidaršič et al., 2012). Other applications of DEX include studies in public administration (Leben et al., 2006), medicine (Šušteršič et al., 2009), agronomy (Griffiths et al., 2010; Pavlovič et al., 2011; Pelzer et al., 2012) and tourism (Stubelj Ars, Bohanec, 2010). In this period we also proposed a new method for automatic revision of DEX models (Žnidaršič, Bohanec, 2007), conceived a DSS tool for modeling uncertain knowledge called proDEX (Žnidaršič et al., 2006), and developed new methods for option ranking based on copulas (Mileva-Boshkoska, Bohanec, 2012).

3 Principles of DEX

The basic principles of DEX are intentionally kept very simple. The decision maker is requested to define a qualitative multi-attribute model, with which decision alternatives are evaluated and analyzed. In principle, the model represents a decomposition of the decision problem into smaller, less complex subproblems. The decomposition is represented by a hierarchy of attributes. The DEX model consists of:

- *Attributes*: variables that represent basic features and assessed values of decision alternatives.
- *Scales* of attributes: these are qualitative and consist of a set of words, such as: 'excellent', 'acceptable', 'inappropriate', etc. Usually, scales are ordered preferentially, i.e., from bad to good values.
- *Hierarchy* of attributes: represents the decomposition of the decision problem and relations between attributes; higher-level attributes depend on lower-level ones.
- *Decision rules*: tabular representation of a mapping from lower-level attributes to higher-level ones. In principle, a table should specify a value of the higher-level attribute for all combinations of values of its lower-level attributes.

Figures 2 and 3 illustrate these components on a simple model for the evaluation of cars (Bohanec, 2012).

The hierarchy in Figure 2 consists of ten attributes. There are six input attributes representing observed features of cars: BUY.PRICE, MAINT.PRICE, #PERS, #DOORS, LUGGAGE and SAFETY. These are aggregated through three intermediate attributes COMFORT, TECH.CHAR. and PRICE into the overall evaluation, which is represented by the root attribute CAR.

Figure 3 shows decision rules that correspond to the CAR attribute. The rules map all the combinations of values of PRICE and TECH.CHAR. into the values of CAR. The attributes PRICE and TECH.CHAR. have three and four values, respectively, so the number of rows in the table is 3×4=12. Each row provides a value of CAR for one combination of the values of PRICE and TECH.CHAR. Interpreted as an elementary decision rule, the fifth row, for example, means that

if PRICE=medium and TECH.CHAR.=bad
then CAR=unacc.

Decision rules, such as the ones in Figure 3, have to be defined for all aggregate attributes in the model. The CAR model thus contains three more rule sets that correspond to the intermediate attributes COMFORT, TECH.CHAR. and PRICE. See Bohanec (2012) for further details.

Attribute	Scale
CAR	**unacc**; acc; good; ***exc***
PRICE	**high**; medium; ***low***
BUY.PRICE	**high**; medium; ***low***
MAINT.PRICE	**high**; medium; ***low***
TECH.CHAR.	**bad**; acc; good; ***exc***
COMFORT	**small**; medium; ***high***
#PERS	**to_2**; 3-4; ***more***
#DOORS	**2**; 3; 4; ***more***
LUGGAGE	**small**; medium; ***big***
SAFETY	**small**; medium; ***high***

Figure 4: DEX model for the evaluation of cars: hierarchy and scales of attributes.

	PRICE	TECH.CHAR.	CAR
1	**high**	**bad**	**unacc**
2	**high**	acc	**unacc**
3	**high**	good	**unacc**
4	**high**	***exc***	**unacc**
5	medium	**bad**	**unacc**
6	medium	acc	acc
7	medium	good	good
8	medium	***exc***	***exc***
9	***low***	**bad**	**unacc**
10	***low***	acc	good
11	***low***	good	***exc***
12	***low***	***exc***	***exc***

Figure 5: Decision rules for an evaluation function PRICE×TECH.CHAR→CAR.

Attribute	Car1	Car2	Car3
CAR	***exc***	good	**unacc**; good; ***exc***
PRICE	***low***	medium	***low***
BUY.PRICE	medium	medium	***low***
MAINT.PRICE	***low***	medium	***low***
TECH.CHAR.	***exc***	good	**bad**; acc; good
COMFORT	***high***	***high***	medium
#PERS	***more***	***more***	3-4
#DOORS	4	4	3
LUGGAGE	***big***	***big***	medium
SAFETY	***high***	medium	*

Figure 2: Evaluation of three cars.

Decision alternatives (i.e., cars in this example) are evaluated by aggregation that is performed from input attributes towards the root of the DEX model. Figure 4 shows the evaluation of three cars: the values corresponding to the attributes COMFORT, TECH.CHAR, PRICE and CAR were evaluated by DEX according to input data provided by the decision maker (i.e., values corresponding to the leaves of the hierarchy) and corresponding decision rule sets. The evaluation of Car3 in Figure 4 illustrates the DEX's way of handling missing information: unknown SAFETY (denoted by '*') is handled by considering all possible values of this attribute, what results in a set of values (rather than a single value) assigned to attributes TECH.CHAR. and CAR.

Attribute	-1	Car2	+1
CAR		good	
BUY.PRICE	**unacc**	medium	***exc***
MAINT.PRICE	**unacc**	medium	***exc***
#PERS		***more***	]
#DOORS		4	
LUGGAGE		***big***	]
SAFETY	**unacc**	medium	***exc***

Figure 3: Plus-minus-1 analysis of Car2

In the final stage, DEX models are typically used for various analyses of alternatives, such as 'what-if' and sensitivity analysis. For example, a typical DEX's analysis is called "plus-minus-1", which investigates the effects of changing each input attribute by one step down (–1) or up (+1) in the attribute scale. Figure 5 shows the results for Car2, which has been originally evaluated as 'good'. Small changes of BUY.PRICE severely affect this evaluation, which becomes 'unacc' and 'exc' when buying price increases or decreases by one step, respectively. Two other attributes, MAINT.PRICE and SAFETY, have the same influence, while the evaluation is unaffected by changes of #PERS, #DOORS and LUGGAGE.

4 Important concepts

Conceptually, DEX is a combination of two approaches: multi-criteria decision analysis (MCDA) and expert systems. From MCDA (Figueira et al., 2005; Bouyssou et al., 2006), DEX borrows the idea of evaluation and analysis of decision alternatives using a hierarchically structured model. DEX departs from using numerical variables and weight-based utility functions by introducing concepts from expert systems: qualitative (symbolic, linguistic) variables, if-then rules, dealing with uncertainty, high emphasis on transparency of models and explanation of evaluation results. DEX has some similarities with two other independently developed approaches: DRSA (Greco et al., 2001) and Doctus (Baracskai, Dörfler, 2003).

Very early in DEX's history it became clear that working directly with model components was not practical and that additional tools were needed to acquire and validate model components, as well as to evaluate, analyze and explain the alternatives. The following concepts and principles were the most important for practical adoption of DEX.

Acquisition of decision rules: Direct definition of tables, such as the one in Figure 3, is tedious and error-prone, and computer-based assistance becomes vital, particularly when rule sets are large. In its early days, DECMAK offered an interactive command-line ASK/ANSWER dialogue. Now, DEXi supports three strategies for the definition of decision rules: direct, 'use scale orders', and 'use weights' (Bohanec, 2012, p. 35).

Validating rules: In comparison with common expert systems, DEXi rules are simple and restricted by the scales of the corresponding attributes, making them suitable for validation of completeness (to which extent they define the mapping) and consistency (are they in conflict with each other). This improves the overall quality of models.

"The user is always right" principle: In spite of consistency checking, DEX gives precedence to information provided by the decision maker. Thus, any decision rule, even if inconsistent, is taken literally and never modified by DEX. In case of inconsistency, the user is given a warning, though.

Dynamic aspects of model creation: The model as shown in Figures 2 and 3 is static. However, in practice, such models are continuously modified and improved: parts of the model are created, extended, moved around or deleted. There are many such operations, such as deleting or adding an attribute, reordering attributes, removing a scale value, etc. All these operations must be supported by appropriate algorithms so that the information already contained in the model is retained as much as possible after each operation. It is particularly important to properly handle decision rules. DEX does implement these operations and typically handles them transparently "behind the scenes".

Bridging the gap between qualitative and quantitative MCDA: The traditional MCDA heavily relies on weights to define the importance of attributes. Naturally, there are no weights in decision rules. However, it turned out to be practically important to deal with weights, so these were included into DEX, too. A partial transformation between attribute weights and rules is possible in both ways (Bohanec, 2012): (1) weights are estimated from defined rules by linear approximation, and (2) the values of undefined decision rules are determined on the basis of already defined rules and user-specified weights.

Handling uncertainty in alternatives and rules: By definition, an expert system must be able to deal with incomplete and uncertain knowledge. The early DECMAK was already able to evaluate incompletely defined alternatives using fuzzy and probabilistic aggregation (Bohanec et al., 1983). In most of the later implementations, the uncertainty in rules was only partly modeled by value intervals. Žnidaršič et al. (2008) extended this approach to using probabilistic distributions in decision rules.

Transparency and explanation: For practice, it is essential that DEX models appear transparent and comprehensible to the user. DEX always provided mechanisms for presenting decision rules in a user-friendly way, from ID3-based decision tree learning algorithms in the early software, to advanced rule generators in the modern DEXi.

Analyses of alternatives: In addition to the mere evaluation of alternatives, the decision support methodology has to provide advanced tools for the analysis of alternatives. For this purpose, DEX includes a number of methods, such as "what-if" analysis, "plus-minus-1" analysis (Figure 5) and selective explanation.

5 Software

Three main generations of qualitative modeling computer programs have been developed so far:

1. *DECMAK* was released in 1981 for operating systems RT-11, VAX/VMS and later for MS DOS. The program had an interactive command-line interface and facilitated the development of a tree of attributes, fuzzy evaluation of alternatives, ASK/ANSWER rule acquisition dialogue, and representing rule tables with complex rules and decision trees. Eventually, due to memory limitations of computers at that time, additional programs were developed separately to form a software suite, which supported functions such as analysis and ranking of alternatives, graphical presentation of decision rules and calculation of weights. In its final form, the DECMAK suite consisted of 19 programs (Bohanec, Rajkovič, 1988b).
2. *DEX* was released in 1987 as an integrated interactive computer program for MS DOS. DEX facilitated interactive model creation and editing, probabilistic and fuzzy evaluation of alternatives, report generation, and selective explanation of evaluation results. In 1995, a supplementary program for ranking of alternatives called *Vredana* was implemented for MS Windows (Šet et al., 1995).
3. *DEXi*, released in 2000, is an interactive educational program for MS Windows. It supports model creation and editing, tabular acquisition of rules, value-set-based evaluation of alternatives, "what-if" analysis, "plus-minus-1" analysis, selective explanation and comparison of options, textual and graphical reports. DEXi is publicly available (http://kt.ijs.si/MarkoBohanec/dexi.html) and its use is free for non-commercial applications.

DEXi is further extended with supporting tools, each related to a specific methodological aspect. They include proDEX (implementation of some DEX extensions, for example using probabilistic values in decision rules), JDEXi (an open-source Java library for evaluation of alternatives), DEXiEval (a command-line utility program for evaluation of alternatives), and DEXiTree (a program for pretty drawing of DEXi trees).

The evaluation part of DEX was often embedded into other software systems. Typical examples include:

- Talent, a system for advising children into sports (Bohanec et al., 2000b),
- a system for risk assessment of diabetic foot care (Bohanec et al., 2000a),
- ESQI: a web page on ECOGEN soil quality index (http://kt.ijs.si/MarkoBohanec/ESQI/ESQI.php),
- SMAC, an advisory system on maize co-existence (Bohanec et al., 2007), and
- a motorway traffic management system (Omerčević et al., 2008).

6 Applications

Ability to tackle complex, real-life problems, is one of DEX's strongest points. In its early days, we kept records of its applications and counted as many as thirty until 1988 (Bohanec, Rajkovič, 1988a). The number of applications continued to grow, but their recording became more and more difficult with the spread of the method and free use of the software. Today, we roughly estimate that DEX has been applied to several hundreds of real-life decision support projects. Considering prototypes and student work, the number of all developed DEX models likely exceeds several thousands.

The areas of DEX applications are very diverse. So far, DEX was used to evaluate technologies, companies, projects, and services. Important problem areas include health care, public administration, agronomy, food production, ecology, land use planning, tourism, housing, traffic control, sports and finance.

Practical experience indicates that DEX is particularly suitable for solving complex decision problems that require judgment and qualitative knowledge-based reasoning, dealing with inaccurate and/or missing data, as well as the analysis and justification of evaluation results. Typically, these problems require large models (with 15 or more attributes) and/or involve many alternatives (10 or more).

7 Future of DEX

Currently, the main software tool for developing DEX models is DEXi. Even after 12 years since its first release, it still seems suitable for education and typical decision making problems, and will – with proper maintenance – continue to serve for these purposes in the future. However, really difficult decision problems require a more powerful methodology and more advanced software (Žnidaršič et al., 2008). The advances in software engineering require new architectures, such as web-based, cloud-based and mobile. There is a need for a DEX library, DEX services and a set of tools for embedding DEX models into other systems, such as information systems, web portals and services, and mobile devices.

For these reasons, we plan to extend the DEX methodology and implement it in a new generation of software (Trdin, Bohanec, 2012). The most challenging methodological advances include:

- *Introduction of numeric attributes*, facilitating the use and interplay of both qualitative and quantitative attributes in an integrated model.
- *Full implementation of probabilistic and fuzzy distributions* for characterization of decision rules and alternatives.
- *Supporting attribute hierarchies*, that is, directed acyclic graphs rather than trees.
- *General aggregation functions* to facilitate the use of all types of aggregation functions known in MCDA.
- *Relational models* that extend the methodology from “flat” to relational alternatives, that is, alternatives composed of sets of subcomponents.

References

[1] Baracskai, Z., Dörfler, V. (2003): Automated fuzzy clustering for Doctus expert system. Proc. Int. Conf. on Computational Cybernetics, Siófok, Hungary.

[2] Bohanec, M., Bratko, I., Rajkovič, V. (1983): An expert system for decision making. *Processes and Tools for Decision Making* (ed. H.G. Sol), North-Holland, 235–248.

[3] Bohanec, M., Rajkovič, V. (1988a): Knowledge acquisition and explanation for multi-attribute decision making, *Proc. 8th Int Workshop "Expert Systems and Their Applications AVIGNON 88"*, Vol. 1, 59–78, Avignon.

[4] Bohanec, M., Rajkovič, V. (1988b): *DECMAK: An expert system shell for multi-attribute decision making*. Report DP-5031, Ljubljana: Jožef Stefan Institute.

[5] Bohanec, M., Rajkovič, V. (1990): DEX: An expert system shell for decision support, *Sistemica* 1(1), 145–157.

[6] Bohanec, M., Rajkovič, V., Semolič, B., Pogačnik, A. (1995): Knowledge-based portfolio analysis for project evaluation, *Information & Management* 28, 293–302.

[7] Bohanec, M., Rajkovič, V. (1999): Multi-attribute decision modeling: Industrial applications of DEX, *Informatica* 23, 487–491.

[8] Bohanec, M., Zupan, B., Rajkovič, V. (2000a): Applications of qualitative multi-attribute decision models in health care, *International Journal of Medical Informatics* 58–59, 191–205.

[9] Bohanec, M., Rajkovič, V., Leskošek, B., Kapus, V. (2000b): Expert knowledge management for sports talent identification and advising process. *Decision Support through Knowledge Management* (eds. S.A. Carlsson, P. Brezillon, P. Humphreys, B.G. Lundberg, A.M. McCosh, V. Rajkovič, V.), IFIP, 46–59.

[10] Bohanec, M., Cestnik, B., Rajkovič, V. (2001): Qualitative multi-attribute modeling and its application in housing. *Journal of Decision Systems* 10, 175–193.

[11] Bohanec, M., Messéan, A., Angevin, F., Žnidaršič, M. (2007): SMAC advisor: A decision-support tool on maize co-existence. *GMCC-07, Third International Conference on Coexistence between Genetically Modified (GM) an non-GM based Agricultural Supply Chains*. Seville, Spain, 20th-21st November 2007. Luxembourg: European Commission, 119–122.

[12] Bohanec, M., Messéan, A., Scatasta, S., Angevin, F., Griffiths, B., Krogh, P.H. (2008): A qualitative multi-attribute model for economic and ecological assessment of genetically modified crops. *Ecological Modelling* 215, 247–261.

[13] Bohanec, M. (2012): *DEXi: Program for Multi-Attribute Decision Making, User's Manual, Ver. 3.04*. IJS Report DP-11153, Ljubljana: Institut Jožef Stefan.

[14] Bohanec, M., Bertheau, Y., Brera, C., Gruden, K., Holst-Jensen, A., Kok, E.J., Lécroart, B., Messéan, A., Miraglia, M., Onori, R., Prins, T.W., Soler, L.-G., Žnidaršič, M. (2013): The Co-Extra decision support system: A model-based integration of project results. *Genetically modified and non-genetically modified food supply chains: Co-existence and traceability* (ed. Bertheau, Y.), 461–489, Wiley-Blackwell.

[15] Bouyssou, D., Marchant, T., Pirlot, M., Tsoukias, A., Vincke, P. (2006): *Evaluation and Decision Models with Multiple Criteria*. Springer.

[16] Efstathiou, J., Rajkovič, V. (1979): Multiattribute decisionmaking using a fuzzy heuristic approach. *IEEE Trans. on Systems, Man, and Cybernetics*, SMC-9, 326–333.

[17] Figueira, J., Greco, S., Ehrgott, M. (2005): *Multi criteria decision analysis: State of the art surveys*. Springer.

[18] Greco, S., Mattarazzo, B., Slowinski, R. (2001): Rough sets theory for multicriteria decision analysis. *European Journal of Operational Research* 129, 1–47.

[19] Griffiths, B.S., Ball, B.C., Daniell, T.J., Hallett, P.D., Neilson, R., Wheatley, R.E., Osler, G., Bohanec, M. (2010): Integrating soil quality changes to arable agricultural systems following organic matter addition, or adoption of a ley-arable rotation. *Applied Soil Ecology* 46(1), 43–53.

[20] Jereb, E., Bohanec, M., Rajkovič, V. (2003): *DEXi: Računalniški program za večparametrsko odločanje,* Moderna organizacija, Kranj.

[21] Krapež, A., Rajkovič, V. (2003): *Tehnologije znanja pri predmetu informatika*. Zavod RS za šolstvo.

[22] Leben, A., Kunstelj, M., Bohanec, M., Vintar, M. (2006): Evaluating public administration e-portals. *Information polity* 11(3/4), 207–225.

[23] Mileva-Boshkoska, B., Bohanec, M. (2012): Ranking of qualitative decision option using copulas. *Operations research proceedings 2011* (eds. D. Klatte, H.-J. Lüthi, K. Schmedders). Springer, 103–108.

[24] Mladenić, D., Lavrač, N., Bohanec, M., Moyle, S. (eds.) (2003): *Data mining and decision support: Integration and collaboration*. Kluwer Academic Publishers.

[25] Olave, M., Rajkovič, V., Bohanec, M. (1989): An application for admission in public school systems. *Expert Systems in Public Administration* (eds. I.Th.M. Snellen, W.B.H.J. van de Donk, J.-P. Baquiast), Elsevier, 145–160.

[26] Omerčević, D., Zupančič, M., Bohanec, M., Kastelic, T. (2008): Intelligent response to highway traffic situations and road incidents. *Proc. TRA 2008*, 21-24 April 2008, Ljubljana, 1–6.

[27] Pavlovič, M., Čerenak, A., Pavlovič, V., Rozman, Č., Pažek, K., Bohanec, M. (2011): Development of DEX-HOP multi-attribute decision model for preliminary hop hybrids assessment. *Computers and Electronics in Agriculture* 75, 181–189.

[28] Pelzer, E., Fortino, G., Bockstaller, C., Angevin, F., Lamine, C., Moonen, C., Vasileiadis, V., Guérin, D., Guichard, L., Reau, R., Messéan, A. (2012): Assessing innovative cropping systems with DEXiPM, a qualitative multi-criteria assessment tool derived from DEXi. *Ecological Indicators* 18, 171–182.

[29] Rajkovič, V., Bohanec, M. (1980): A cybernetic model of the computer aided decision making process. 9th Int. Congress on Cybernetics, Namur, 8–13th September, 185–199.

[30] Rajkovič, V., Bohanec, M., Batagelj, V. (1988): Knowledge engineering techniques for utility identification. *Acta Psychologica* 68(1–3), 271–286.

[31] Stubelj Ars, M., Bohanec, M. (2010): Towards the ecotourism: A decision support model for the assessment of sustainability of mountain huts in the Alps. *Journal of Environmental Management* 91(12), 2554–2564.

[32] Šet, A., Bohanec, M., Krisper, M. (1995): Vredana: Program za vrednotenje in analizo variant v večparametrskem odločanju [Vredana: A program for evaluation and analysis of alternatives in multi-attribute decision making]. *Proceedings of the Fourth International Electrotechnical and Computer Science Conference ERK-95*, Portorož, Slovenia. Ljubljana: Slovenska sekcija IEEE, 157–160.

[33] Šušteršič, O., Rajkovič, U., Dinevski, D., Jereb, E., Rajkovič, V. (2009): Evaluating patient's health using a hierarchical multi-attribute decision model. *Journal of International Medical Research* 37, 1646–1654.

[34] Trdin, N., Bohanec, M. (2012): Extending the multi-criteria decision making method DEX. Proc. 4th Jožef Stefan International Postgraduate School Students Conference (eds. D. Petelin, A. Tavčar, B. Kaluža), 25 May 2012, Ljubljana, Slovenija, 182–187.

[35] Zupan, B., Bohanec, M., Demšar, J., Bratko, I. (1999): Learning by discovering concept hierarchies, *Artificial Intelligence* 109, 211-242.

[36] Žnidaršič, M., Bohanec, M., Zupan, B. (2006): proDEX - A DSS tool for environmental decision-making. *Environmental Modelling & Software* 21, 1514–1516.

[37] Žnidaršič, M., Bohanec, M. (2007): Automatic revision of qualitative multi-attribute decision models. *Foundations of Computing and Decision Sciences* 32(4), 315–326.

[38] Žnidaršič, M., Bohanec, M., Zupan, B. (2008): Modelling impacts of cropping systems: Demands and solutions for DEX methodology. *European Journal of Operational Research* 189, 594–608.

[39] Žnidaršič, M., Bohanec, M., Lavrač, N., Cestnik, B. (2009): Project self-evaluation methodology: The Healthreats project case study. *Proc. Information Society 2009*, Ljubljana, 85–88.

[40] Žnidaršič, M., Bohanec, M., Trdin, N. (2012): Qualitative assessment of data-mining workflows. *Fusing decision support systems into the fabric of the context* (eds. A. Respício, F. Burstein). Amsterdam: IOS Press, 75–88.

Orange: Data Mining Fruitful and Fun - A Historical Perspective

Janez Demšar and Blaž Zupan
University of Ljubljana, Faculty of Computer and Information Science, Tržaška 25, Ljubljana, Slovenia
E-mail: {janez.demsar|blaz.zupan}@fri.uni-lj.si

Keywords: data mining, machine learning, applications

Received: November 15, 2012

Orange (http://orange.biolab.si) is a general-purpose machine learning and data mining tool. Its multilayer architecture is suitable for different kinds of users, from data mining beginners to programmers who prefer a scripting interface. In this paper we outline the history of Orange's development and present its achievements, current status, and future challenges.

Povzetek: Orange kot splošno orodje za stojno učenje in odkrivanje znanj iz podatkov, zgodovina, trenutna podoba in prihodnji izzivi.

1 Introduction

General-purpose machine learning software tools have had a short but eventful history. It began with utilities that implemented a specific induction method and reported on an inferred model in a textual form, to be then scrutinized and admired by the user. Such were implementations of tree and rule inducers C4.5 [5] and CN2 [3] in the 1980s. A contender to C4.5 was Assistant Professional [2]. Assistant Professional was expensive compared to C4.5, which came free with a book. As a result, C4.5 went on to become the most popular machine learning program of the last century and, in the spirit of free software, a precursor of open-source toolboxes.

In the 1990s the machine learning community grew substantially and so did the number of different approaches. Many researchers preferred a single toolbox to implement, apply, and test their methods on a standard collection of data sets. Packages like IND and MLC++[1] emerged, each with a set of command-line utilities that also implemented testing schemes and could report on evaluation scores.

To illustrate the challenges of those times, consider medical data modeled with a classification tree, and a medical doctor who wants to know which particular patients belong to a branch of the tree and what their average blood pressure is. With only a command-line utility at our disposal, this simple endeavor could (and did) prove rather impossible. Command-line data analytics is no fun. Systems with graphical user interfaces emerged as alternatives. The basic ones offered only data plotting capabilities such as R.[2] In those that were more advanced, the user could interact with the graphics, for instance by clicking on a graphical element to get the information about the data it represents. SGI's MineSet [1], for one, was a commercial product that was built on top of MLC++ and that implemented what was for the 1990s an advanced exploratory graphical interface. There was one problem, however. The data analysis pipeline was fixed: read the data, visualize the model, explore it, and interpret the results.

Data analysts are, by definition, inquisitive. When given data, we like to dissect it, build models, observe their parts, consider specific data subsets, and dissect it further. We like to construct data analysis pipelines, not just use them. The idea of visual programming, an interface in which pipelines are created by linking predefined or even user-designed components, was explored in the early 1990s by Sun, IBM, and SGI in packages for data visualization such as Data Explorer.[3] A similar idea took off in data mining with Clementine (later bought by SPSS and in 2009 renamed to SPSS Modeler). Open-source toolboxes followed: Weka[4], Knime[5], Yale (which is now a much redesigned RapidMiner[6]), and Orange[7], each built on their own favorite programming language, assembling a different set of core components and offering their own interface for explorative data analysis.

2 History of Orange

The development of Orange began in 1997 by the authors of this paper. Its development continued at the Artificial Intelligence Laboratory and is currently taking place at the Laboratory of Bioinformatics, both at the University of Ljubljana.

Orange as a library of C++ components and command-line utilities. Orange was first conceived as a C++ library of machine learning algorithms and related procedures, such as preprocessing, sampling, and other data ma-

[1] http://www.sgi.com/tech/mlc
[2] http://www.r-project.org
[3] http://www.research.ibm.com/dx
[4] http://www.cs.waikato.ac.nz/ml/weka
[5] http://www.knime.org
[6] http://rapid-i.com
[7] http://oranga.biolab.si

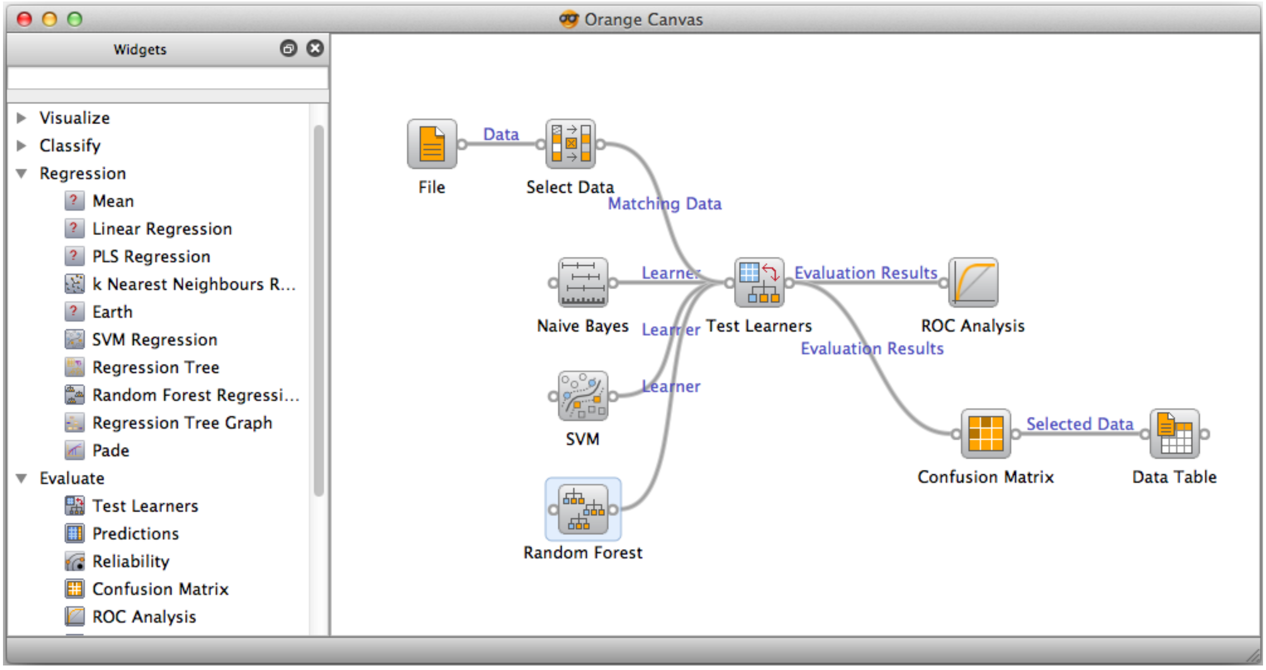


Figure 1: An Orange schema that explores misclassified data instances with a naive Bayesian classifier, support vector machines and random forest.

nipulation. We aimed at constructing a platform where advanced users would write their own components in C++. This turned out not to be the case. Instead, Orange was mostly used for data exploration in which different combinations of provided preprocessing and learning algorithms were tested and scored using cross-validation. The components were packed into programs with command line interfaces. As this was too limiting, we decided to provide a scripting interface to these components by exposing them to Python.

Orange as a Python module. Python, a modern scripting language, was chosen for a variety of reasons.

- It has a very clean and simple syntax that is easy to learn, not only for a programmer but also for a beginner. (Python is becoming the language of choice for teaching programming at many leading universities, including CMU, MIT, Berkeley, Rice and Caltech.)
- Despite its simplicity, Python is an industry-strength language. For instance, Python is behind many of Google's technologies, which is also why Google is one of the major sponsors of Python's development.
- Since programming in Python is fast, it is very suitable for prototyping of new methods.
- It is relatively easy to extend Python with modules written in C or C++. Python was even dubbed a glue language due to its use for gluing libraries in C or Fortran. Today, one seldom needs to implement specialized routines in low-level languages due to the availability of high quality libraries such as NumPy and SciPy.

Since 1999 Orange has been used almost exclusively as a Python module. Although the C++ core eventually rose to around 140,000 lines, most developers have been adding to its Python modules and rather than implementing their own classes in C++.

The transition to Python has enabled several important developments. More and more of Orange's functionality is implemented in pure Python or by combining the fast functions provided by the C++ Orange core using the glue code written in Python. Since the programs in Python are so readable, they enable collaboration of larger teams without the need to coordinate the development and establish a set of coding standards. The size of the group that develops the system has increased to 10-15 members, mostly from the Laboratory of Bioinformatics. Most importantly, migration to Python has simplified the development of the graphical user interface.

Orange Visual Programming. Our group has a tradition of collaborating with partners from other scientific and industrial areas, biomedicine in particular. We wished to provide them with a data exploration tool in which they could design their own data analysis pipelines without any scripting or Python programming [4]. Among a number of different Python libraries for implementing graphical user interfaces, we chose Qt, a strong cross-platform library which is available under both GPL and commercial licenses, and

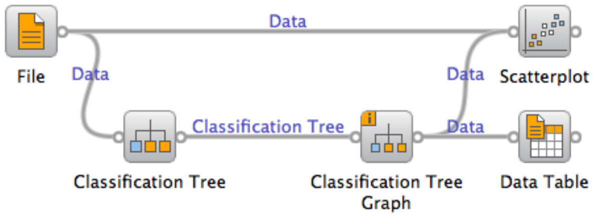


Figure 2: A schema where the user can observe data belonging to selected nodes of the decision tree. The selected examples are shown in a table and marked in a scatter plot (see Figure 3).

which is behind products as different as Skype and KDE desktop.

The majority of current users now use Orange only through its graphical interface (Figure 1). It consists of a canvas onto which users place pipeline components called widgets. Each widget offers some basic functionality, such as reading the data, showing a data table, selecting data features, either manually or based on some feature scoring, training predictors, cross-validating them, and so forth. The user connects the widgets by communication channels. The basic strength and flexibility of Orange is in the different ways in which the widgets can be combined into new schemata.

Special emphasis in the development and design of widgets was placed on data visualization and interactivity. For instance, a classification tree viewer allows the user to click on a node of the tree, which transmits the data samples that belong to the node to any widgets connected to the tree viewer widget (Figure 2). The user can thus construct a tree and then explore its content by observing, say, a data table with the data instances from interesting nodes, or, for example, by drawing scatter plots from the entire data set and marking the data points from different nodes of the tree (Figure 3).

Figure 1 shows a more complex schema. After loading some data, the user selects a subset of instances and induces a naive Bayesian classifier, support vector machine, and random forest. Results are compared using ROC curves and confusion matrices. The confusion matrix is connected to a data table widget with which the user can observe particular kinds of misclassifications.

While the general philosophy of Orange is that widgets should be simple and the power of the tool should stem from the different ways of connecting them, several widgets are rather powerful by themselves. For instance, in the widget for discretization of continuous attributes (Figure 4), we can set a general discretization method (equal interval width, equal frequency of data points, and entropy-based discretization), override it for individual attributes, or manually discretize an attribute by entering expert-defined thresholds or finding suitable ones from a graph that shows distributions and gains at different threshold values.

3 Orange in 2012

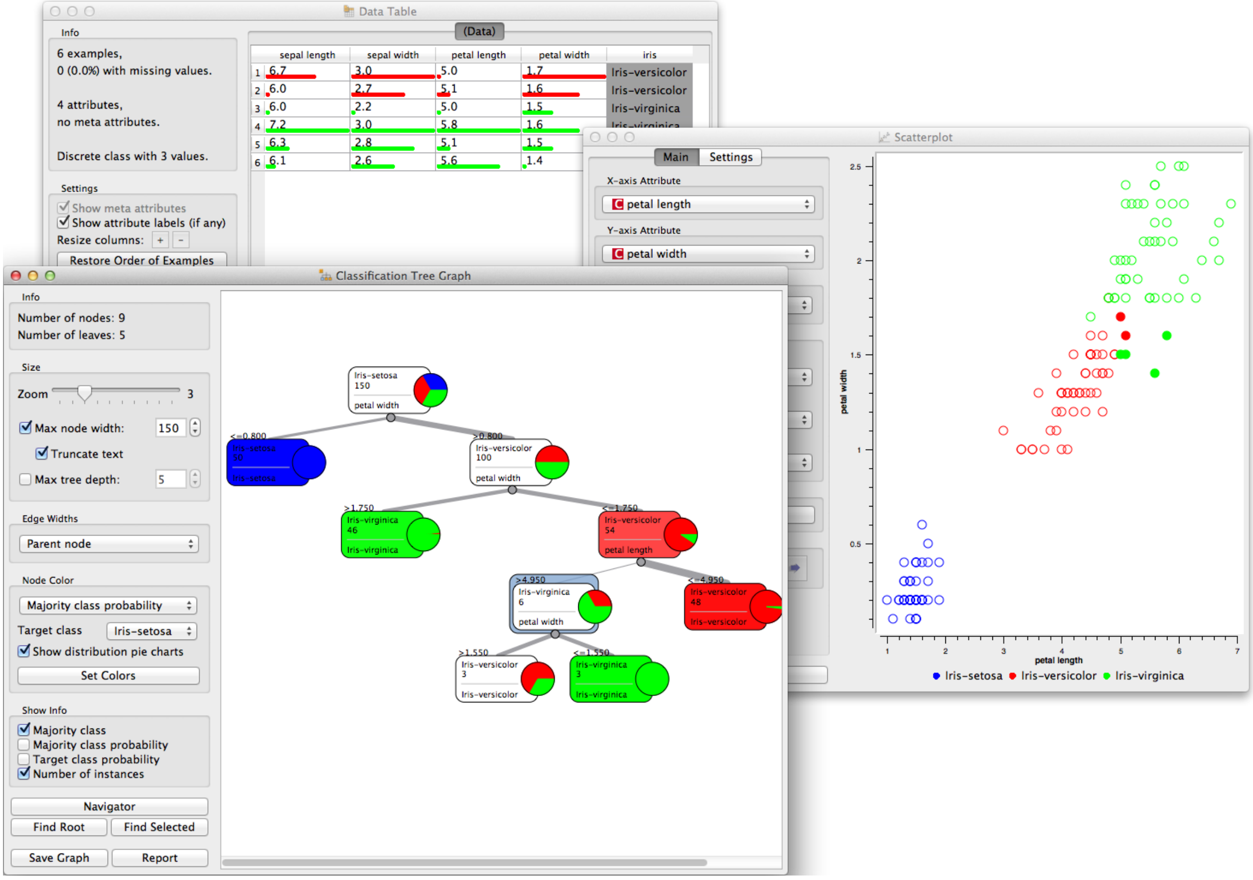
Orange, together with Knime, is among the easiest-to-use data mining tools available. It runs on OS X, Windows, and Linux. The default installation includes a number of machine learning, preprocessing, and data visualization algorithms. Unlike Weka, for instance, which offers everything there is in machine learning, the goal of Orange was to implement the most useful and commonly used techniques in a way that is flexible and user-friendly. The emphasis of the tool is data exploration.

The machine learning algorithms in the default installation are limited to a naive Bayesian classifier, k-nearest neighbors, induction of rules and trees, support vector machines, neural networks, linear and logistic regression, and ensemble methods. Most methods are, however, coupled with a visual representation that allows for exploration of the resulting model; the user can select a node in a classification tree or rule, and explore the training instances covered by them. A naive Bayesian classifier, logistic regression, and linear SVM can be explored through nomograms that offer insight into the importance of features and their individual values. Nomograms can also be used for explaining the model's predictions. This also applies to unsupervised methods, such as association rules, multidimensional scaling, self-organizing maps, and various types of clustering.

In contrast with the intentionally limited assortment of machine learning methods, Orange has a rich collection of visualization methods. Besides the common visualizations, such as box plots, histograms, and scatter plots, it contains a number of multivariate visualizations, including parallel coordinates, mosaic display, sieve diagram, survey plots, and a number of data projection techniques, such as multi-dimensional scaling, principal component analysis, RadViz, FreeViz, and others. The user can interactively explore the visualizations or connect them to other widgets that send or receive data from the visualization. Orange can also help the user in finding insightful visualizations by automatically ranking them by interestingness or by organizing them into a network of visualizations.

Orange can be extended with additional modules. We currently provide an extensive collection of methods for bioinformatics, as well as modules for text mining and multi-target learning and a set of powerful widgets for visualization and exploration of networks.

Orange has been used in science, industry, and teaching. Scientifically, it is used as a testing platform for new machine learning algorithms, as well as for implementing new computational approaches in molecular biology and bioinformatics. The most notable industrial partner is Astra-Zeneca, a pharmaceutical giant, which uses Orange in drug development and sponsors the development of several related parts of Orange [6]. At Jožef Stefan Institute, the visual programming interface has been upgraded in Orange4WS to support service-oriented architectures. Orange is also being used for teaching courses in machine learning



Data Table

Info

6 examples,
0 (0.0%) with missing values.

4 attributes,
no meta attributes.

Discrete class with 3 values.

Settings

☑ Show meta attributes
☑ Show attribute labels (if any)
Resize columns: + -
Restore Order of Examples

(Data)

	sepal length	sepal width	petal length	petal width	iris
1	6.7	3.0	5.0	1.7	Iris-versicolor
2	6.0	2.7	5.1	1.6	Iris-versicolor
3	6.0	2.2	5.0	1.5	Iris-virginica
4	7.2	3.0	5.8	1.6	Iris-virginica
5	6.3	2.8	5.1	1.5	
6	6.1	2.6	5.6	1.4	

Scatterplot

Main | Settings

X-axis Attribute: C petal length
Y-axis Attribute: C petal width

Classification Tree Graph

Info

Number of nodes: 9
Number of leaves: 5

Size

Zoom 3
☑ Max node width: 150
☑ Truncate text
☐ Max tree depth: 5

Edge Widths

Parent node

Node Color

Majority class probability
Target class: Iris-setosa
☑ Show distribution pie charts
Set Colors

Show Info

☑ Majority class
☐ Majority class probability
☐ Target class probability
☑ Number of instances

Navigator
Find Root | Find Selected
Save Graph | Report

Figure 3: Widgets from schema in Figure 2 showing the classification tree with selected node of six instances. These are displayed in the table and highlighted in the scatterplot. The user can change the selection, instructing Orange to propagate it through the schema and update table and scatterplot widget. By combining interaction and signal propagation, the collection of widgets thus becomes a tool for explorative data analysis.

and data mining in countries around the world, including the US, Italy, France, Japan, Turkey, Cuba, and Peru.

4 Future developments

The landscape of Python's libraries has been strongly affected by reorganizing the obsolete libraries Numeric and numarray into NumPy[8], which has become a standard library for scientific computation in Python. It provides arrays of arbitrary dimensions and linear algebra routines from BLAS and ATLAS. Another library, SciPy, adds many other common scientific routines, from statistical functions to fast Fourier transforms. The scikit-learn[9] library utilizes NumPy's fast vectorized operations to provide fast and high-quality implementations of most machine learning algorithms and is widely used by the community.

The power of Orange is not as much in its machine learning algorithms, but rather in the way in which they are packed and exposed to Python scripting in a simpler form. Beyond that, an even stronger feature of Orange is its graphical user interface and visual programming environment. We are intensely working on a new version of Orange in which we will replace the entire C++ core with routines in NumPy, SciPy, scikit-learn, and similar third-party open source libraries for Python. This should encourage contribution from outside the group and allow us to concentrate on the development of just those parts where Orange is unique. For early 2013, we plan a revamped user interface (Figure 5) and, in collaboration with company Ainda, we are designing MyFlow[10], a platform with a web-based interface to Orange.

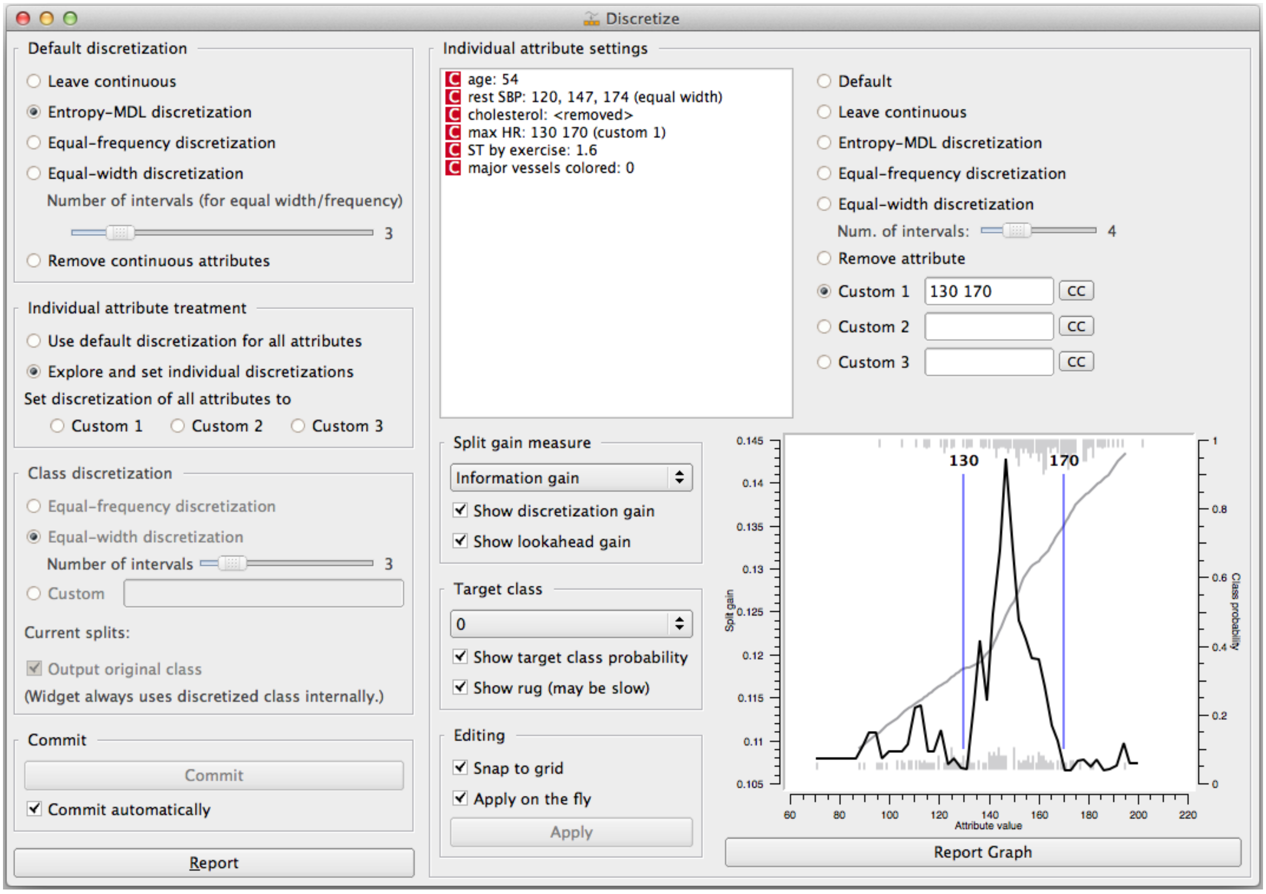
Acknowledgements

Development of Orange is a team effort of many developers, and we thank them all for their enthusiasm and support. Major contributions to the package have been made by Aleš Erjavec, Gregor Leban, Tomaž Curk, Marko Toplak, Miha

[8] http://NumPy.SciPy.org

[9] http://scikit-learn.org

[10] http://myflow.io



Discretize

Default discretization

○ Leave continuous
● Entropy-MDL discretization
○ Equal-frequency discretization
○ Equal-width discretization

Number of intervals (for equal width/frequency) 3

○ Remove continuous attributes

Individual attribute treatment

○ Use default discretization for all attributes
● Explore and set individual discretizations

Set discretization of all attributes to

○ Custom 1 ○ Custom 2 ○ Custom 3

Class discretization

○ Equal-frequency discretization
● Equal-width discretization

Number of intervals 3

○ Custom

Current splits:

☑ Output original class

(Widget always uses discretized class internally.)

Commit

☑ Commit automatically

Individual attribute settings

C age: 54
C rest SBP: 120, 147, 174 (equal width)
C cholesterol: <removed>
C max HR: 130 170 (custom 1)
C ST by exercise: 1.6
C major vessels colored: 0

○ Default
○ Leave continuous
○ Entropy-MDL discretization
○ Equal-frequency discretization
○ Equal-width discretization

Num. of intervals: 4

○ Remove attribute
● Custom 1 130 170
○ Custom 2
○ Custom 3

Split gain measure

Information gain

☑ Show discretization gain
☑ Show lookahead gain

Target class

0

☑ Show target class probability
☑ Show rug (may be slow)

Editing

☑ Snap to grid
☑ Apply on the fly

Figure 4: Discretization widget; the left part of the widget defines the general settings for all attributes and the right part allows for individual attribute customization.

Štajdohar, Anže Starič, Matija Polajnar, Lan Žagar, Jure Žbontar, Mitar Milutinović, Lan Umek, Črt Gorup, Martin Možina, Gregor Rot, Aleks Jakulin, and designers Peter Čuhalev and Roman Ražman. We thank the Slovenian Research Agency (P2-0209, J2-9699, L2-1112), National Institutes of Health (European Commission (FP7 CARE-MI 242038, FP7 AXLE 318633), Astra Zeneca, and Google through its Summer of Code program, for providing financial support.

References

[1] Clifford Brunk, James Kelly, and Ron Kohavi. MineSet: An Integrated System for Data Mining. In *KDD*, pages 135–138, 1997.

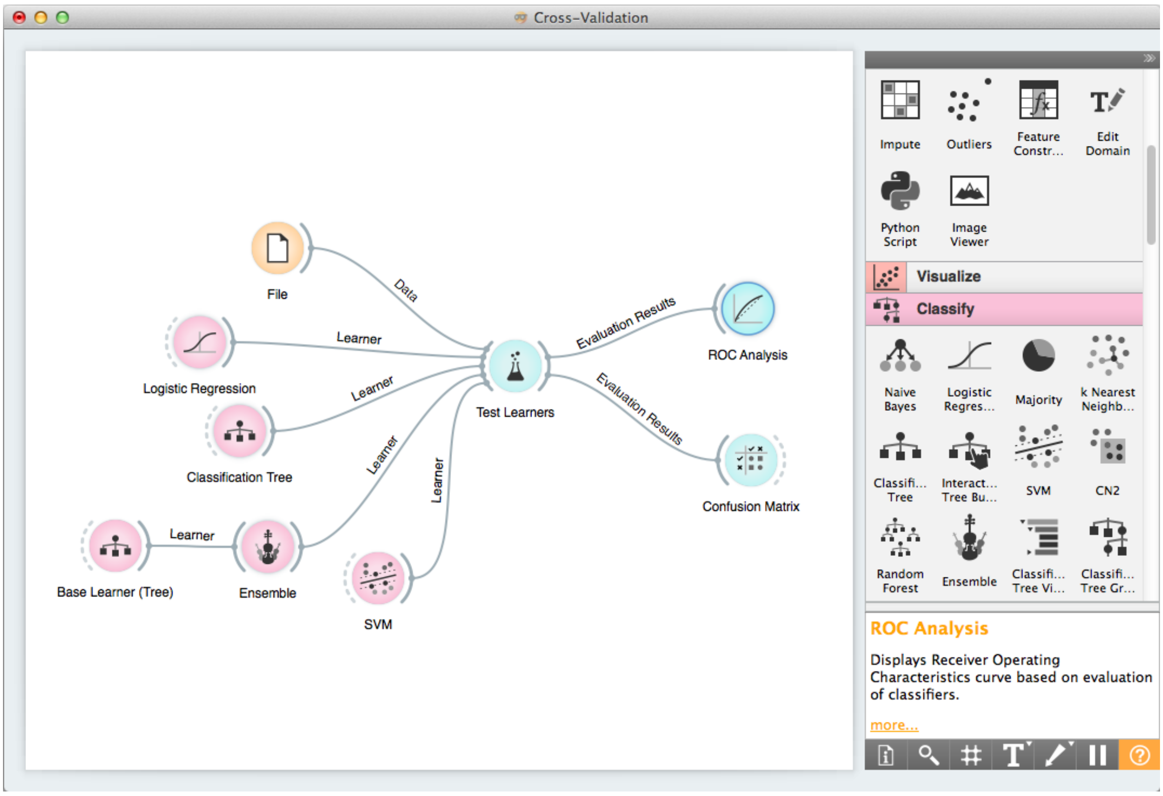
[2] Bojan Cestnik, Igor Kononenko, and Ivan Bratko. ASSISTANT 86: A Knowledge-Elicitation Tool for Sophisticated Users. In *EWSL*, pages 31–45, 1987.

[3] Peter Clark and Tim Niblett. The CN2 Induction Algorithm. *Machine Learning*, 3(4):261–283, 1989.

[4] Tomaz Curk, Janez Demsar, Qikai Xu, Gregor Leban, Uros Petrovic, Ivan Bratko, Gad Shaulsky, and Blaz Zupan. Microarray data mining with visual programming. *Bioinformatics*, 21(3):396–398, 2005.

[5] J R Quinlan. *C4.5: Programs for Machine Learning*, volume 1 of *Morgan Kaufmann series in Machine Learning*. Morgan Kaufmann, 1993.

[6] Jonna C Stålring, Lars A Carlsson, Pedro Almeida and Scott Boyer. AZOrange - High performance open source machine learning for QSAR modeling in a graphical programming environment. *Journal of Cheminformatics*, 3:28, 2011.



Displays Receiver Operating Characteristics curve based on evaluation of classifiers.

Figure 5: The design of a graphical interface for the upcoming version of Orange (by Peter Čuhalev).

The Advantage of Careful Imputation Sources in Sparse Data-Environment of Recommender Systems: Generating Improved SVD-based Recommendations

Mustansar Ali Ghazanfar and Adam Prugel-Bennett
School of Electronics and Computer Science, University of Southampton, Highfield Campus, SO17 1BJ, United Kingdom
E-mail: eng.musi@gmail.com, adp@ecs.soton.ac.uk,
Phone: +44 (023) 80594473; fax: +44 (023) 80594498

Overview paper

Keywords: SVD, recommender systems, imputation, collaborative filtering

Received: December 2, 2012

Recommender systems apply machine learning and data mining techniques for filtering unseen information and can predict whether a user would like a given item. The main types of recommender systems namely collaborative filtering and content-based filtering suffer from scalability, data sparsity, and cold-start problems resulting in poor quality recommendations and reduced coverage. There has been some work in the literature to increase the scalability by reducing the dimensions of the recommender system dataset using singular value decomposition (SVD); however, due to sparsity it results in inaccurate recommendations. In this paper, we show how a careful selection of an imputation source in singular value decomposition based recommender system can provide potential benefits ranging from cost saving, to performance enhancement. The proposed missing value imputation methods have the ability to exploit any underlying data correlation structures and hence have been proven to exhibit much superior accuracy and performance as compared to the traditional missing value imputation strategy—item average of the user-item rating matrix—that has been the preferred approach in the literature to resolve this problem. By extensive experimental results on three different dataset, we show that the proposed approaches outperform traditional one and moreover, they provide better recommendation under new user cold-start problem, new item cold-start problem, long tail problem, and sparse conditions.

Povzetek: Opisani so priporočilni sistemi, tj. sistemi, ki filtrirajo informacije s pomočjo metod strojnega učenja.

1 Introduction

1.1 Recommender systems

There has been an exponential increase in the volume of available digital information, electronic sources, and online services in recent years. This information overload has created a potential problem, which is how to filter and efficiently deliver relevant information to a user. Furthermore, information needs to be prioritised for a user rather than just filtering the right information, which can create information overload problems. Search engines help Internet users by filtering pages to match explicit queries, but it is very difficult to specify what a user wants by using simple keywords. The Semantic Web, also provides some help to find useful information by allowing intelligent search queries; however it depends on the degree to which the web pages are annotated. These problems highlight a need for information extraction systems that can filter unseen information and can predict whether a user would like a given item. Such systems are called *recommender systems* [1, 2], and they mitigate the aforementioned problems to a great extent. Given a new item, recommender systems can predict whether a user would like this item or not, based on the user preferences (likes—positive examples, and dislikes—negative examples), observed behaviour, and information (demographic or content information) about items/users.

An example of the recommender system is the *Amazon* recommender engine [3], which can filter through millions of available items based on the preferences or past browsing behaviour of a user and can make personal recommendations. Some other well-known examples are *Youtube* (www.youtube.com) video recommender service and *MovieLens* (www.movielens.com) movie recommender system, which recommend videos and movies based on the person's opinions. Recommender systems helps E-commerce sites in increasing their sales by making useful recommendation—items a customer/user would most likely to consume [4]. In these systems, the history of user's interactions with the system is stored, which shape user's preferences. The history of the user can be gathered by explicit feedback, where the user rates some items in some scale, or by implicit feedback, where the user's interaction with the system is observed—for instance, if a user purchases an item then this is a sign that they like that item, their browsing behaviour, etc.

There are two main types of recommender systems: collaborative filtering (CF) and content-based filtering recommender systems. Collaborative filtering recommender systems [5, 6, 7, 8, 9, 10] recommend items by taking into account the taste (in terms of preferences of items) of users, under the assumption that users will be interested in items that users similar to them have rated highly. Examples of these systems include the GroupLens system [9], and Ringo (www.ringo.com). Collaborative filtering can be classified into two sub-categories: memory-based CF and model-based CF. Memory-based approaches [6] make a prediction by taking into account the entire collection of previous rated items by a user. Examples of these systems include GroupLens recommender systems [9, 8]. Model-based approaches use rating patterns of users in the training set, group users into different classes, and use ratings of predefined classes to generate recommendation for an *active user*[1] on a *target item*[2]. Examples of these systems include item-based CF [11], Singular Value Decomposition (SVD) based models [12, 13, 14, 15], bayesian networks [16], clustering models [17, 18, 19, 20, 21, 22], and Kernel-mapping recommender [23].

Content-based filtering recommender systems [24, 25, 26] recommend items based on the content information of an item, under the assumption that users will like similar items to the ones they liked before. In these systems, an item of interest is defined by its associated features, for instance, NewsWeeder [24], a newsgroup filtering system uses the words of text as features. The textual description of items is used to build item profiles. User profiles can be constructed by building a model of the user's preferences using the descriptions and types of the items that a user is interested in, or a history of user's interactions with the system is stored (e.g. user purchase history, types of items they purchased together, etc.). The history of the user can be gathered by explicit feedback or implicit feedback. Explicit feedback is noise free but the user is unlikely to rate many items, whereas, implicit feedback is generally noisy (error prone), but can collect a lot of training data [27]. In general, a trade-off between implicit and explicit user feedback is used. Creating and learning user profiles is a form of classification problem, where training data can be divided into two categories: items liked by a user, and items disliked by a user. Furthermore, hybrid recommender systems have been proposed [28, 29, 30, 31, 32], which combine individual recommender systems to avoid certain limitations of individual recommender systems.

Recommendations can be presented to an active user in the followings two different ways: by predicting ratings of items, a user has not seen before and by constructing a list of items ordered by their preferences. The task of predicting an unknown rating is trivial, where an algorithm receives an unknown user-item pair and related users (user-based CF [8]), items (item-based CF [11]), item and user content features (content-based filtering recommender systems [33]), demographic features (demographic recommender systems [30]), or all of the aforementioned parameters (hybrid recommender systems [29]); and then predicts how much the user would like the given item in some numeric or binary scale. Different heuristics are used for producing an ordered list of items, sometimes termed as *top-N recommendations* [12]; for example, in collaborative filtering recommender system this list is produced by making the rating predictions of all items an active user has not yet rated, sorting the list, and then keeping the top-N items the active user would like the most.

1.2 Problem statement

A Recommender system (RS) consists of two basic entities: users and items, where users provide their opinions (ratings) about items. We denote these users by $\mathcal{U} = \{u_1, u_2, \cdots, u_M\}$, where the number of users using the system is $|\mathcal{U}| = M$, and denote the set of items being recommended by $\mathcal{I} = \{ i_1, i_2, \cdots, i_N\}$, with $|\mathcal{I}| = N$. The users will have given ratings of some but not all of the items. We denote these ratings by $(r_{i,u}|(i,u) \in \mathcal{D})$, where $\mathcal{D} \subset \mathcal{I} \times \mathcal{U}$ is the set of user-item pairs that have been rated. We denote the total number of ratings made by $|\mathcal{D}| = T$. Typically each user rates only a small number of the possible items, so that $|\mathcal{D}| = T \ll |\mathcal{I} \times \mathcal{U}| = N \times M$. It is not unusual in practical systems to have $T/(N \times M) \approxeq 0.01$. The set of possible ratings made by users can be thought of as elements of an $M \times N$ rating matrix R with elements $r_{i,j}$. The recommender system's task is to infer the elements in R for which we do not have any data. As prediction accuracy of a recommender system depends heavily on the available number of examples, hence it would suffer in case where the rating data is sparse.

The continuous increase of the users and items demands the following properties in a recommender system: (1) accuracy (2) scalability (3) maximum coverage (4) robustness with *sparsity* [34, 35, 36, 37, 38]. A number of approaches have been proposed to remedy the sparsity and scalability problems associated with the CF, ranging from supervised classification techniques [39] to unsupervised clustering techniques [17], to dimensionality reduction techniques spanning a number of algorithms such as singular value decomposition [12], low rank approximation using matrix factorisation techniques [40], and principal component analysis [41]. Remaining properties can be satisfied by a hybrid approach resulting in increased coverage while eliminating the sparsity problem.

Singular value decomposition methods have proved to be successful in increasing the scalability of the CF [12, 42]; however approximating missing values by the item average, which has been heavily used in the literature, is not a reasonable approach. Although, it reduces sparsity and increases coverage, it will lead to lower recommendation accuracy. To efficiently deal with the sparsity problem, we have to address the basic question: *"Why are the data*

[1] The user for whom the recommendations are computed.

[2] The item a system wants to recommend.

missing"? The following assumptions can be made about the missing data: *missing completely at random* (MCAR), *missing at random* (MAR), and *not missing at random* (NMAR) [43]. MCAR assumes that the data is missing for completely random reasons, and that the probability of observing the value of a rating does not depend on the observed or unobserved values of the dependent variable. Most of the CF algorithms assume that missing data is MAR [44], which means that probability of observing a rating does not depend on the value of the rating. NMAR occurs when the missingness is related to the unobserved dependent variable, i.e. if probability of observing the rating of an item depends on the value of the rating. NMAR assumes that there is a relationship between the missingness and what would have been observed.

To make a clear distinction between these categories, assume that $G_{i,j}$ is an indicator function, which takes a value of 1 if the subject i is observed at time j and 0 otherwise. If a study is conducted at n time-points, then the dependent variable and missing data indicator vectors can be represented by $y_i^{1\times n} = \{y_{i,1}, y_{i,2}, \cdots, y_{i,n}\}$ and $G_i^{1\times n} = \{G_{i,1}, G_{i,2}, \cdots, G_{i,n}\}$ respectively. The specific values in the missing data indicator vector, $G_{i,j}$, is equal to 1 when $y_{i,j}$ is observed and 0 otherwise. Based on the G, the dependent variable, y, can be divided into two classes: observed, y^o, and unobserved or missing, y^m. MCAR assumes that missing data indicator G_i are independent of both y_i^o and y_i^m. MAR assumes that missing data indicator G_i are independent of y_i^m; however, they are related to the observed dependent variable vector, i.e. y_i^o. An example of MAR is to drop some subjects from a study when their value falls below a certain threshold. For instance, in an election survey if a subject has age less than 18 then they are drop out of the survey. NMAR assumes that there is a relationship between the value of y_i^m and the missingness, $G_{i,j}$. MAR assumption is not true in recommender systems; for example, in MovieLens recommender system rating distribution is skewed towards the higher ratings, which indicates that users are much more likely to watch and rate movies they think they will like rather than entering rating for the movies they do not like, resulting in higher bias towards observing ratings with high value. Other researchers have claimed that this bias can results in erroneous recommendations [44].

The main claim of this work is that imputation methods to deal with the missing values, if effectively used prior to applying SVD in recommender systems, can provide potential benefits ranging from cost saving, to performance enhancement. The proposed missing value imputation methods have the ability to exploit any underlying data correlation structures and hence are proven to exhibit much superior accuracy and performance compared to the traditional missing value imputation strategy that has been the preferred approach in the literature to resolve this problem. This work, presents a comparative study of the missing values in the user-item matrix of a recommender system and their subsequent impact upon the accuracy and cost has been investigated. The empirical study has shown that the results are dataset dependent; however rather than using the traditional approach to fill the user-item rating matrix or merely ignoring the missing values, which have heavily been used in the literature, robust and advanced approaches can give considerable performance benefits in the (1) SVD based recommendations (2) iterative SVD (3) and CF applied over the reduced dataset. We evaluate our algorithms on the MovieLens (www.grouplens.org/node/73) ($100K$ ratings and $1M$ ratings) and FilmTrust (http://trust.mindswap.org/FilmTrust) datasets.

The rest of the paper has been organised as follows. Section 2 discusses the related work in detail. Section 3 presents the motivations and advantages of using the proposed algorithm. Section 4 sheds light on the background concepts related to the SVD. Section 5 outlines the proposed algorithms. Section 6 describes approaches used to approximate the missing values in the sparse user-item rating matrix. Section 7 describes the data set and metrics used in this work. Section 8 presents results comparing the performance of the proposed algorithms with the traditional one. Section 9 discusses when and how much imputation is sufficient to achieve good accuracy. Section 10 gives a detailed discussion of the work, and finally Section 11 concludes the work.

2 Related work

The Singular Value Decomposition (SVD)-based approach for solving the recommendation problem was first introduced by [39]. [12] presented a detailed analysis of the behaviour of SVD-based recommender systems. Various algorithms combining the SVD-based approach with the item-based CF have been advocated [45, 46, 47, 48]; for example, [46] combined SVD with the item-based CF and claimed that their approach outperformed the conventional item-based CF. An example of using the SVD-based approach with demographic data has been presented in [13], where the authors applied SVD over the user-item rating matrix and demographic data of users and items, and claimed that a system consisting of a linear combination of SVD-based demographic correlation and SVD-based (item-based) CF, increases the accuracy of the recommender system. [42] suggested an incremental SVD model building approach and claimed that it is more scalable than the conventional SVD-based recommender systems, while producing recommendations with same accuracy. All of the aforementioned approaches used the item average of the data matrix to approximate the missing values, which may destroy the covariance structure of the data, resulting in inaccurate recommendations.

Another way of applying SVD is presented in [49], where the authors applied Principal Component Analysis (PCA)[3] over a so-called 'gauge-set' of items—set of items

[3] PCA is a closely related concept to SVD, which reduces the dimen-

rated by every user in the system. Although it may reduce sparsity, getting this dataset is hard in real-life scenarios, and also it may lead to potential loss of information as we are ignoring ratings not in the gauge-set. Sometimes, case deletion strategy [41] is used for dealing with the missing values where all variables with missing values are omitted in the data matrix resulting in loss of information, which is not desirable. Some other well-known approaches to approximate the missing values are filling by zero and scaling the known entries as suggested by [50].

The missing values have been handled by the Expected Maximisation (EM) algorithm [51] by [52], [53], [54], [41], [55] and [14]. In this approach, the predictions generated by the current model are replaced by the previous one and the procedure is repeated until some stopping criteria are reached; for example, the error between two successive models becomes less than a threshold. The problem with this approach is that the final error and the convergence is highly dependent on the method used to approximate the initial values. [53] showed the convergence behaviour of the EM algorithm by approximating the missing entries by zeros [50] and using the gauge-set [49]. Furthermore, they proposed an approach by starting with a large rank approximation and gradually reducing the rank of SVD in each iteration of EM.

Netflix prize competition [56] has made extensive use of low rank approximation—matrix factorisation [57, 58, 59, 40, 60, 61] and SVD based scheme—for example in [62], the authors proposed a CF approach based on the global cost function. They proposed an accurate SVD based model that integrates implicit feedback. They also proposed a framework for integrating neighbourhood based and SVD based approaches and claimed that it is more accurate than other approaches. In [63], the authors removed global effect from the dataset by normalising it using different heuristics. Afterwards they simultaneously derived interpolation weights for all neighbours by solving an optimisation problem. They claimed that this approach is scalable and gives more accurate results than conventional neighbourhood based approach over Netflix dataset. In [54] the author proposed a model to maximise the log-likelihood of the available ratings using EM algorithm. These approaches often lead to *over-fitting* [64] and need extensive parameters tuning which may not be pragmatic or desired in certain cases.

Various SVD based approaches have been combined for improving the prediction accuracy; for example, in [58], the author proposed a solution for Netflix prize by blending 107 individual results, and won the Netflix 2007 prize. A similar approach is presented in [65], where the author proposed a linear combination of SVD based predictor, KMeans clustering, combining SVD with K-NN, post processing SVD with ridge regression, and others (total 72 predictors) and claimed that it gave 7.04% improvement in terms of Root Mean Square Error (RMSE) over Netflix's Cinematch[4] on Netflix prize competition. Another example is proposed in [66], where the authors used a linear combination of SVD, association rules, and other approaches for making accurate predictions. In [59], the authors combined (using ensemble methods) different variants of matrix factorisation, such as, regular matrix factorisation, and non-negative matrix factorisation, and claimed that combined approach gave 7% improvement, in terms of RMSE , over the Netflix CineMatch recommender system. Though, theoretically, we can increase accuracy of a recommender system by these methods, but practically it is not pragmatic [67]. Furthermore, these approaches completely miss other metrics, such as, coverage, and scalability proposed in [68]. In this work, we have focused on the SVD-based recommender systems [12, 13, 14, 47, 48] rather than matrix factorisation techniques.

The most similar work with ours is that undertaken by [14], where the authors used an item-item imputation technique in addition to the user-average over the Netflix dataset. Our work; however, differs from theirs in a number of areas, as follows: (1) they only used an item-item imputation while we are using 18 different approaches to analyse the behaviour of SVD and EM algorithms; (2) they only used one dataset; however, we are using three different datasets and furthermore, we find out that the results are highly dataset-dependent; (3) they claimed that the item-item imputation scheme is outperformed by the average, which is in contrast with our findings; and (4) we are applying CF over the reduced dataset; however, they did not apply it. In summary, their focus was on the efficient implementation of Lanczos, power iteration, and other algorithms rather than imputation; however, we are analysing the behaviour of the SVD-based algorithms under different recommender system conditions—cold-start, long tail, and sparsity problems.

The imputation has been used in collaborative filtering domain. The idea of using imputation in collaborative filtering domain was proposed by [16], where the authors used some default votes to decrease the sparsity of the user-item rating matrix. The author claimed that using the default votes in the user-based CF outperforms the conventional user-based CF in terms of accuracy. This idea has further been used by many researchers in various ways to approximate the missing values in the user-item rating matrix; for example, [28] used a Naive bayes classifier trained on the content profiles of users, [69] and [70] used information filtering agents or "Filterbot", [71] used a linear combination of user- and item-based CF, [72] used a recursive CF algorithm, and [73, 74] used several methods. The problem with these approaches is that they are not very scalable. Our approach is different from these because we are doing imputation in SVD domain and CF is applied over the dataset reduced by employing SVD. Furthermore, imputation has been used in other domains; for example, for Epistatic miniarray profiles [75].

Hybrid recommender systems have widely been used in

sionality by projecting high dimensional data along a smaller number of orthogonal dimensions.

[4] Cinematch is the Netflix proprietary recommender system.

the literature; for example, Pazzani [30] used a machine learning approach to build a classifier based on the demographic data about a user. The author used Winnow to extract features from user's home page to build the user model. Fab [76] employed a meta-level hybrid system [29], where first a content-based filtering using Rocchio's method [77] was applied to maintain a term vector model that describe the user's area of interest. This model was then used by CF to gather documents on basis of interest of community as a whole. LIBRA [33] a book recommender system, downloaded content information about book (meta data) from Amazon, and built user models using a Naive bayes classifier. In [31], the author proposed an on-line hybrid recommender system for news recommendation by dynamically adjusting weights to content-based filtering or CF.

Content-based recommender systems have also been combined with CF for reducing the sparsity of the dataset and producing accurate recommendations; for example in [36], the authors proposed a unique cascading hybrid recommendation approach by combining the rating, feature, and demographic information about items, and claimed that their approach outperforms other algorithms in terms of accuracy and coverage under sparse dataset. In [35], the authors combined Naive bayes classifier with CF using switching hybrid approach, and claimed that their algorithm provides better performance than other algorithms. Information filtering agents have been integrated with CF in [78, 69], where the author proposed a framework for combining the CF with content-based filtering agents. They used simple agents, such as spell-checking, which analyse a new document in the news domain and rate it to reduce the sparsity of the dataset. Again, the problem with these approaches is that, they are not very scalable.

Various hybrid recommender systems have been proposed using ontology and CF [79, 80, 81, 82, 83] to overcome the sparsity problem of the user-item rating matrix. For example in [79], the authors used domain specific ontologies to enhance the similarity between the items in the item-based CF. They linearly combine the similarities between items based on the user-item rating matrix and structure semantic knowledge about items to generate recommendations, and claimed that this semantically enhanced approach outperform item-based CF particularly given sparse dataset. The problems with these approaches in that they require laborsome knowledge engineering techniques to capture the domain specific knowledge and ontologies, which may not be pragmatic given millions of items that is common with E-commerce domains.

3 Motivation and advantages of the proposed algorithm

The motivation of our aim is to develop an efficient algorithm for producing accurate recommendations under sparse datasets at low cost. From this line of the research, we propose algorithms that satisfies the following properties:

3.1 Overcome sparsity problem

In many commercial recommender systems like Amazon; it is not unusual, even for active customer to provide ratings well under $< 1\%$ of all the available products. Furthermore, an increase in the number of items in the database will decrease the density of each user with these items. The performance—accuracy and coverage—of conventional collaborative filtering algorithms suffer the most under sparse conditions, because they rely on the similar users and items. In the worse case, we might find very few or no similar user (or item) as the correlation coefficient between two users (or items) can be defined if they have some ratings in common. If there are only a few common items, the correlation coefficient is poorly approximated, and thus less reliable recommendations are produced. The proposed algorithms do not suffer from sparsity because, they (1) use a suitable imputation source to approximate all the missing values (2) apply SVD over the dense user-item rating matrix, which captures the important latent relationship between users and items, leading to reduced sparsity and 100% coverage.

3.2 Accurate recommendations

An important task for a recommender system is to *find good items* and to ameliorate the *quality* of the recommendation for a customer. If a customer trusts and leverages a recommender system, and then discovers that they are unable to find what they want then it is unlikely that they will continue with the system. Consequently, the most important task for a recommender system is to accurately predict the rating of the non-rated user/item combination and recommend items based on these predictions. Our algorithms give more accurate recommendations as compared to the traditional SVD-based approaches and collaborative filtering.

3.3 Low on-line cost

The proposed SVD takes $O(1)$ time to generate a prediction which, is less than typical approaches. The complexity of the item-based and user-based CF, applied over the original user-item rating matrix, is $O(MN^2)$ and $O(NM^2)$ respectively[5]. When we reduce the user-item rating matrix to k dimensions using SVD, then the complexity reduces to $O(kN^2)$, and $O(kM^2)$, in case of user-based and item-based CF respectively. This reduced complexity will decrease the memory requirement and on-line processing time.

[5] We assume that SVD and the similarities between items and users are computed in off-line fashion.

3.4 Search efficiency: scalability

To engage visitors in a web-site or turn casual surfers into customer, a recommendation algorithm has to make recommendations quickly and accurately. In a database, with millions of ratings, it becomes very difficult to find the similar users or items using memory based approaches. By employing the imputed SVD, the dimension of the original user-item matrix can be reduced, and recommendations can be generated directly or CF can be applied over the reduced dimensions, making this approach suitable for high dimensional dataset.

3.5 Overcome cold-start problems

Generally, while testing recommender systems, a dataset is used where some sets of ratings are treated as unseen while the other ratings are used for learning. The unseen data are then used to test the performance of the algorithm. To obtain accurate results, datasets are usually selected with users that have made a relatively high number of ratings. However, in real applications, the datasets are often highly skewed; for example, a large number of users may have made only a small number of ratings and a large number of items may have received very few ratings. These are important scenarios in practical systems as making reasonable recommendations to new users can be crucial in attracting more users. There are two important cold-start scenarios as described below [84]:

– *New user cold-start problem:* When a new user enters the system, initially the system does not have enough data for that user, and hence the quality of the recommendations would suffer, a potential problem called the new user cold-start problem [34].

– *New item cold-start problem:* When a new item is added to a system, then initially it is not possible to get a rating for that item from a significant number of users, and consequently the CF recommender systems would not be able to recommend that item effectively. This problem is called the new item cold-start problem [34].

The Proposed imputation methods provide better recommendations than the conventional approach under the cold-start scenarios.

3.6 Overcome long tail problem

Newly introduced or unpopular items having only a few ratings can create a potential problem for a recommender system. Many recommender systems; for example, the CF ones, ignore these items or cannot produce reliable recommendations for these items. This problem is called the long tail problem [17]. As the majority of the items in a recommender system generally falls into this category [17], there is a need to develop algorithms which can filter, personalise, and accurately recommend from the huge amount of items available in the long tail. We show that the proposed imputation methods provide better recommendations than the conventional approach under the long tail scenario.

4 Background

Singular Value Decomposition (SVD) [85, 86] is a matrix factorisation technique that takes an $m \times n$ matrix A, with rank r and decomposes it into three component matrices as follows:

$$\mathrm{SVD}(A) = U \times S \times V^{\mathsf{T}}. \quad (1)$$

U and V (V^{T} is for the transpose of V) are orthogonal matrices with dimensions $m \times m$, and $n \times n$ respectively, and S, called the *singular matrix*, is a $m \times n$ diagonal matrix consisting of non-negative real numbers. These matrices reflect the decomposition of the original matrix into linearly independent vectors (factor values). The set of initial r diagonal values of S ($s_1, s_2, \cdots, s_r$) are all positive with $s_1 \geq s_2 \geq s_3, \cdots, \geq s_r$. The first r columns of U are eigenvectors of AA^{T} and represent the left singular vectors of A. Similarly, the first r columns of V are eigenvectors of $A^{\mathsf{T}}A$ and represent the right singular vectors of A. The best low-rank approximation of matrix A is obtained by retaining the first k diagonal values of S, by removing $r - k$ columns from U, and by removing $r - k$ rows from V, which can be represented as follows:

$$A_k = U_k \times S_k \times V_k^{\mathsf{T}}. \quad (2)$$

By keeping only the k largest singular values of S, the effective dimensions of the SVD matrices U, S, and V become $m \times k$, $k \times k$, and $k \times n$ respectively. The best-k rank approximation of matrix A with respect to the Frobenius norm can be represented by:

$$||A - A_k||_F^2 = \sum_{i,u}(a_{iu} - \sum_k U_{uk} \times S_k \times V^{\mathsf{T}}{}_{ki}) \quad (3)$$

SVD can be applied over the user-item rating matrix, of dimensions $M \times N$, generated by a recommender system. It assumes that there is some latent structure—overall structure that relates to all or most items (or users)—in the matrix that is partially obscured by variability in ratings assigned to items (or assigned by users). This latent structure can be captured by transforming the matrix in low dimensions. After transformation, users and items can be represented by a vector in the k-dimensional space. The matrix product $U_k.\sqrt{S_k}^{\mathsf{T}}$ represents M (pseudo) users and $\sqrt{S_k}.V_k^{\mathsf{T}}$ represents N (pseudo) items in the k-dimensional space. For example, in a movie domain, each element of $\sqrt{S_k}.V_k^{\mathsf{T}}(i)$ $(1 \leq i \leq N)$ can be a feature of movie i, such as whether it is a horror movie, whether it is rated PG-13 or not, etc. Similarly, the corresponding element of $U_k.\sqrt{S_k}^{\mathsf{T}}(u)$ $(1 \leq u \leq M)$ shows whether the user likes these feature in movies. A rating assigned by a pseudo-users u on item i is denoted by $r'_{i,u}$. The prediction $\hat{r}_{i,u}$

for the uth user on the ith item can be computed by the following equation:

$$\hat{r}_{i,u} = U_k.\sqrt{S_k}^{\mathsf{T}}(u).\sqrt{S_k}.V_k^{\mathsf{T}}(i). \tag{4}$$

If we normalise the user-item rating matrix by subtracting the respective user average ($\overline{r}_i$) from a rating, then a prediction is given by the equation:

$$\hat{r}_{i,u} = \overline{r}_i + U_k.\sqrt{S_k}^{\mathsf{T}}(u).\sqrt{S_k}.V_k^{\mathsf{T}}(i). \tag{5}$$

5 Proposed algorithms

5.1 Imputed SVD (Algorithm 1)

We used various imputation methods, F (discussed in the next section), for approximating the missing values in the user-item rating matrix R and then applied SVD for reducing the dimensions of the matrix. The pseudo code to generate improved recommendations is given in Algorithm 1. In step 7, which serves as a pre-processing step, we fill in the missing values in the initial sparse user-item rating matrix by an imputation source. In step 8, we normalise the filled rating matrix by subtracting the respective user average from the filled rating matrix. In step 9, we reduce the dimensions of the filled normalised rating matrix by applying SVD. In the IMPUTEDERROR procedure, from steps 12 to 26, we find the optimal number of dimensions (k) by changing the dimension from 1 to 50 and observing the corresponding MAE.

5.2 Collaborative filtering applied over the reduced dataset (Algorithm 2)

We can apply the user- and item-based CF over the matrix components generated by the IMPUTE procedure. Algorithm 2 outlines the steps required to apply CF over the reduced data matrix. The similarity between two items can be found by Adjusted cosine or cosine measure [34]. We used Adjusted cosine similarity because it gave us more accurate results. The similarity between two items i_x and i_y can be found by measuring the cosine of angle computed over k users as follows:

$$\mathrm{sim}(i_x, i_y) = \frac{\sum_{u=1}^{k} r'_{i_x,u}.r'_{i_y,u}}{\sqrt{\sum_{u=1}^{k} r'^{2}_{i_x,u} \sum_{u=1}^{k} r'^{2}_{i_y,u}}}, \tag{6}$$

where $r'_{i_x,u}$ and $r'_{i_y,u}$ are the ratings assigned by user u on items i_x and i_y respectively. The ratings shown by r' are obtained from the matrix product $\sqrt{S_k}^{\mathsf{T}}.V_k$, which represents the rating given by k (pseudo) users on N items[6].

Algorithm 1 : ImpSvd; Impute the matrix, compute SVD, and generate recommendations
Input: R, the user-item rating matrix; f, an imputation method
Output: $error^*$, the minimum MAE; k^*, the optimal number of dimensions for SVD

```
 1: procedure SVDRECOMMENDATION($R$, $f$)
 2:     $(U, S, V)$=IMPUTE($R$, $f$)
 3:     ($error^*$, $k^*$)=IMPUTEDERROR($U, S, V$)
 4:     return ($error^*$, $k^*$)
 5: end procedure

 6: procedure IMPUTE($R$, $f$)
 7:     Fill in the missing values in the user-item rating
        matrix $R$ by an imputation method $f$. Call the resulting
        dense matrix $R_f$.
 8:     Normalise the dense matrix ($R_f$) and call it $R_N$.
 9:     Apply SVD over the normalised matrix $R_N$ and
        find three components of the matrix as shown in equation 1. Call these matrices $U$, $S$, and $V$.
10:     return ($U, S, V$)
11: end procedure

12: procedure IMPUTEDERROR($U, S, V$)
13:     $error^* \leftarrow 10$
14:     $k^* \leftarrow 1$
15:     for $k \leftarrow 1, 50$ do
16:         ($U_k, S_k, V_k$)=DIMREDUCE($U, S, V, k$)
17:         Compute $U_k.\sqrt{S_k}^{\mathsf{T}}$ and $\sqrt{S_k}.V_k^{\mathsf{T}}$
18:         Make predictions using equation 5
19:         Compute MAE for all predictions, call this
        $error_{new}$
20:         if $error_{new} < error^*$ then
21:             $error^* \leftarrow error_{new}$
22:             $k^* \leftarrow k$
23:         end if
24:     end for
25:     return ($error^*$, $k^*$)
26: end procedure

27: procedure DIMREDUCE($U, S, V, k$)
    Perform dimensionality reduction step:
28:     Find $S_k$ by setting $S_{i,i} = 0$ for $i > k$
29:     Find $U_k$ by removing $r - k$ columns from $U$
30:     Find $V_k$ by removing $r - k$ rows from $V$
31:     return ($U_k, S_k, V_k$)
32: end procedure
```

[6] We do not need to subtract the respect user average while measuring the similarity as the matrix has already been normalised prior to applying SVD.

We used the significance weighting schemes as proposed by [87] while measuring the similarity between users (or items).

The similarity between two users can be found by the Pearson correlation or the cosine of angle [88]. We used the cosine of angle, which gave us more accurate results than the Pearson correlation. The similarity between two users can be found by the cosine of angle, computed over k items, as follows[7]:

$$\mathrm{sim}(u_a, u_b) = \frac{\displaystyle\sum_{i=1}^{k} {r'}_{i,u_a} . {r'}_{i,u_b}}{\sqrt{\displaystyle\sum_{i=1}^{k} {r'}^2_{i,u_a} \sum_{i=1}^{k} {r'}^2_{i,u_b}}}, \qquad (7)$$

where ${r'}_{i,u_a}$ and ${r'}_{i,u_b}$ are the ratings assigned on item i by users u_a and u_b respectively. The ratings shown by r' are obtained from the matrix product $U_k\sqrt{S_k}^{\mathsf{T}}$, which represents the ratings given by M users on k (pseudo) items.

In the case of item-based CF, the prediction for an active user u_a on target item i_t is made by using the adjusted weighted sum formula as follows:

$$\hat{r}_{i_t,u_a} = \bar{r}_{u_a} + \frac{\displaystyle\sum_{i=1}^{l} sim(i, i_t) \times {r'}_{i,u_a}}{\displaystyle\sum_{i=1}^{l} |sim(i, i_t)|}, \qquad (8)$$

where l represents the l most similar items against a target item, found after applying equation 6.

In the case of the user-based CF, the prediction for an active user u_a on target item i_t is made by using the adjusted weighted sum formula as follows:

$$\hat{r}_{i_t,u_a} = \bar{r}_{u_a} + \frac{\displaystyle\sum_{u=1}^{l} sim(u, u_a) \times ({r'}_{i_t,u} - \bar{r}_u)}{\displaystyle\sum_{i=u}^{l} |sim(u, u_a)|}, \qquad (9)$$

where l represents the l most similar users against an active user, found after applying equation 7.

Individual predictions made by the user- and item-based CF can be combined linearly. We expect that combining these two approaches will result in an increase in the accuracy, as both of them focus on different kinds of relationships. Let $\hat{r}^{ub}_{i,u}$ and $\hat{r}^{ib}_{i,u}$ represent the prediction generated by the user- and item-based CF respectively. The final prediction is a linear combination of these predictions as follows:

$$\hat{r}_{i,u} = \alpha \times \hat{r}^{ub}_{i,u} + \beta \times \hat{r}^{ib}_{i,u}, \qquad (10)$$

where parameters α and β can be found over the validation set. We call this algorithm $ImpSvd^{hybrid}_{CF}$.

Algorithm 2 : $\mathbf{ImpSvd_{CF}}$; Apply SVD over the reduced dataset
Input: R, the user-item rating matrix; f, an imputation method; $flag$, a variable to decide between the user- and item-based CF
Output: $error^*$, the minimum MAE; k^* and $neigh^*$, the optimal number of dimensions and neighbours for CF

```
procedure CFRecommendation($R$, $f$, flag)
    $(U, S, V)$=Impute($R$,$f$)
    Start grid search over dimensions, $k$ and neighbourhood size, $neigh$ to find the optimal number of dimensions, $k^*$ and neighbourhood size, $neigh^*$
    $(U_k, S_k, V_k)$=DimReduce($U, S, V, k$)
    if flag = 1 then
        $\hat{r}^{ib}_{i,u}$ = $ImpSvd^{ib}_{CF}$($U_k$, $S_k$, $V_k$, $neigh$)
    else
        $\hat{r}^{ub}_{i,u}$ = $ImpSvd^{ub}_{CF}$($U_k$, $S_k$, $V_k$, $neigh$)
    end if
    Store the minimum MAE, $error^*$; the optimal number of dimensions, $k^*$; and the optimal number of neighbours, $neigh^*$
    End grid search
    return ($error^*$, $k^*$, $neigh^*$)
end procedure

procedure $ImpSvd^{ib}_{CF}$($U_k$, $V_k$, $S_k$, $l$)
    Find the matrix product $\sqrt{S_k}.{V_k}^{\mathsf{T}}$
    Find the similarity between two items using equation 6
    Isolate $l$ most similar items to the target item (neighbours of the target item) found using equation 6
    Make a prediction, $\hat{r}^{ib}_{i,u}$, using equation 8
    return $\hat{r}^{ib}_{i,u}$
end procedure

procedure $ImpSvd^{ub}_{CF}$( $U_k$, $V_k$, $S_k$, $l$)
    Find the matrix product $U_k.\sqrt{S_k}^{\mathsf{T}}$
    Find the similarity between two users using equation 7
    Isolate $l$ most similar users to the active user (neighbours of the active user) found using equation 7
    Make a prediction, $\hat{r}^{ub}_{i,u}$, using equation 9
    return $\hat{r}^{ub}_{i,u}$
end procedure
```

[7] In this case, we do not normalise the user-item rating matrix prior to applying SVD.

5.3 Iterative SVD: Applying SVD combined with EM algorithm (Algorithm 3)

Algorithm 3 : $\mathtt{ItrSvd}$; Apply SVD in EM fashion
Input: R, the user-item rating matrix; f, an imputation method; ϑ, threshold value to terminate the EM algorithm
Output: t, the number of iterations in the EM algorithm; $error^*$, the MAE observed after the EM algorithm converges

```
procedure ITERATIVERECOMMENDATION($R$, $f$, $\vartheta$)
    $t \leftarrow 0$
    $error^{(t)} \leftarrow 0$
    repeat
        $(U, S, V)$ = IMPUTE($R$, $f$)
        ($U_{k^*}$,$V_{k^*}$,$S_{k^*}$)=DIMREDUCE($U, S, V, k^*$)    ## $k^*$ is the optimal number of dimensions learned through the validation set
        Compute $U_{k^*}\sqrt{S_{k^*}}^{\intercal}$, $\sqrt{S_{k^*}}V_{k^*}{}^{\intercal}$
        Call the current SVD model $\mathcal{M}_{k^*}$
        Make predictions using equation 5
        Compute the MAE for $\mathcal{M}_{k^*}$, call it $error_{new}$
        $t \leftarrow t + 1$
        $error^{(t)} \leftarrow error_{new}$
        $f \leftarrow \mathcal{M}_{k^*}$
    until $|error^{(t)} - error^{(t-1)}| < \vartheta$
    $error^* \leftarrow error^{(t)}$
    return $t$, $error^*$
end procedure
```

The $ItrSvd$ (Algorithm 3) uses the combination of SVD and Expected Maximisation (EM) [51] to estimate the missing values. As SVD calculations require the filled matrix, missing values are replaced by an imputation method prior to the k most effective eigenvalues being selected. In each iteration of the EM algorithm, the missing values are replaced by the corresponding values in the previous estimated model in the expectation step, i.e.

$$R_{iu}^{(t)} = \begin{cases} R_{iu} & if\ iu \in \mathcal{D}, \\ [\sum_k U_k \times S_k \times V^{\intercal}{}_k]_{iu}^{(t-1)} & otherwise, \end{cases} \tag{11}$$

and in the maximisation step the aim is to find the model ($\mathcal{M}^{(t)}$) parameters that minimises

$$\sum_{iu} (R_{iu}^{(t)} - \mathcal{M}_{iu})^2, \tag{12}$$

where $\mathcal{M}_{iu} = [\sum_k U_k \times S_k \times V^{\intercal}{}_k]_{iu}$. The algorithm keeps alternating between expectation and maximisation (SVD computation) steps, until it converges (the change in the MAE between two iterations becomes less than a pre-determined threshold (0.001)). This algorithm usually gives more accurate results after convergence; however, its drawback is that it is highly sensitive to the noise in the dataset and it only considers the global data correlation, which means that in a locally correlated dataset, it will lead to higher estimation error.

6 Proposed approaches to approximate the missing values in the user-item rating matrix

Our main focus is on careful selection of imputation sources for approximating the missing values in the user-item rating matrix that can lead to different results. Our claim is that, sensible approaches used for filling the missing values, prior applying the SVD, can lead to significant performance increase. The imputation approaches used are discussed below:

- *Filling by zero (zeros)*: We fill the missing values in the user-item rating matrix by zero. This approach is very simple, and computational efficient, which make it attractive. This approach does not take into account the underlying correlation structure of the data affecting the data variance that is generally high. Subsequently, if we have a large number of missing values, then this imputation approach can results in inaccurate recommendations.

- *Filling by random number (Rand)*: We fill the missing values in the user-item rating matrix by a random number generator function that generates a random number in the range of 1 to 5 in the case of MovieLens and 1 to 10 in the case of FilmTrust dataset. Its advantages and disadvantages are the same as those of Zero.

- *Filling by normal distribution (Nor_U, Nor_I)*: We fill the missing values in the user-item rating matrix by normal distribution $\mathcal{N}(\mu, \sigma^2)$. Here we denote Nor_U to represent the case where the corresponding user average and standard deviation of ratings are used as μ and σ respectively. Similarly, we denote Nor_I to represent the case where the corresponding item average and standard deviation of ratings (given by other users) are used as μ and σ respectively.

- *Filling by uniform distribution (UniformDist)*: We fill the missing values in the user-item rating matrix by uniform distribution $\mathcal{U}(a, b)$, where $a = 1, b = 5$ and $a = 1, b = 10$ for the MovieLens and FilmTrust dataset respectively.

- *Filling by item average (ItemAvg)*: In this approach an unknown rating is replaced by the average rating given by all the users in the training set. If no one has rated that item it is replaced by zero. This approach serves as a **baseline** for our experimental evaluation, as it has been the preferred approach to resolve this problem; for example it has been used in [12, 13, 46].

- *Filling by user average (UserAvg)*: In this approach an unknown rating for an active user is replaced by the average rating given by the active user in the training set. If the active user has rated no item, then it is replaced by zero. This approach is very simple however, it can distort the shape of the distribution and can reduce the variance of the data. We use the term **conventional methods** for the ItemAvg and UserAvg imputation sources.

- *Filling by the average of user and item averages (UserItemAvg)*: In this approach an unknown rating is replaced by averaging the user's average rating and item's average rating.
- *Filling by user-based CF (UBCF)*: In this approach an unknown rating is predicted by using user-based CF[8]. In user-based CF there are three main steps to make a prediction: (1) find all the users who have rated the target item, (2) find the similarity of the users with the active user, and isolate these users also called *neighbours* of the active user, (3) make prediction by adjusted weighted sum of the ratings provided by neighbours. Despite their simplicity they give accurate predictions.
- *Filling by item-based CF (IBCF)*: In this approach, an unknown rating is predicted by using item-based CF. In Item-based CF there are three main steps to make a prediction: (1) find all the items rated by the active user, (2) find the similarity of these items with the target item, and isolate the items also called *neighbours* of the target item, (3) make prediction by adjusted weighted sum of the ratings provided by the active user on the neighbouring items. It has been argued [11] that item-based CF outperforms user-based CF.
- *Filling by the average of user- and item-based CF (UBIBCF)*: In this approach, an unknown rating is replaced by averaging the predictions generated by user-based and item-based CF.
- *Filling by SVM classifier (SVMClass)*: In this approach an unknown rating is replaced by using the results obtained by applying the SVM classifier over the training set. We normalise the data in scale of $0-1$ and used LibSVM [89] for binary classification. We used linear kernel rather than radial basis function (RBF), as other researchers have found that if the number of features are very large compared to the number of instances, there is no significant benefit of using RBF over linear kernel [90]. Furthermore, tuning parameters in RBF and polynomial kernels is very computation intensive given a large feature size. For multi class problem, several methods have been proposed, such as one-verse-one (1v1), one-verse-all (1vR), Directed Acyclic Graph (DAG), and unbalanced decision tree (UDT) [91]. We did not found any significant difference among the results obtained by these methods, hence we show results in case of 1v1 only. More details to use SVM for recommender systems can be found in our previous work [92]. We have chosen SVM, as they give good performance for text categorisation problems; for example, in [93], the authors claimed that they outperform KNN, Naive bayes, and other classifiers in text categorisation tasks.
- *Filling by Naive bayes classifier (NBClass)*: In this approach an unknown rating is replaced by using the results obtained by applying the Naive bayes classifier over the training set. The details of building and using Naive bayes classifier for recommender system can be obtained from our previous work [35, 92].
- *K nearest neighbours (KNN)*: In this approach an unknown rating is replaced by using the results obtained by applying the K Nearest Neighbours (KNN) using the Weka collection of machine learning algorithms [94]. KNN estimates missing values by searching for the K nearest neighbours (users) and then taking the weighted average of these K neighbours' ratings. In our work, the proposed scheme is similar to the KNN; however, it differs in that the contribution of each neighbour is weighted by its similarity to the active user. As the degree of contribution will be determined by the choice of weighting system, hence we tested our scheme with two weighting systems. In the first approach (shown by KNN in the results) we weight neighbours by $1 - dist$, where $dist$ is the distance between two neighbours. In the second approach (shown by WKNN in the results), we weight neighbours according to the following scheme employed by [75]:

$$weight(i,j) = \left(\frac{dist^2}{1 - dist^2 + \epsilon}\right)^2,$$

where $\epsilon = 10^{-6}$ is added to avoid dividing by zero. This function is similar to the Gaussian kernel function, which gives more weight to closer neighbours than distant neighbours. WKNN has proven to give good results in [75].

- *Filling by decision tree (C4.5)*: In this approach, an unknown rating is replaced by using the results obtained by applying the Decision Tree (C4.5) using the Weka library. Although the process of constructing the tree tries to minimise the error rate using the training data for evaluation, it will probably not perform well while classifying the test data. The reason is that it can easily be *over-fitted* to the training data [64]. Therefore, in order to generalise its performance, we pruned the tree by learning the pruning confidence over the training set. We used Laplace smoothing for predicted probabilities and kept the minimum number of instances per leaf to 2.
- *Filling by SVM regression (SVMReg)*: In this approach, an unknown rating is replaced by using the results obtained by applying the SVM regression over the training set. We used linear kernel and trained the

[8] If algorithm fails to predict a rating, then it is replaced by the approach given in 6. The same is true for other algorithms

cost parameters. We used nu-SVR version of the SVM regression using the LibSVM [89] library.

- *Filling by linear regression (LinearReg)*: In this approach, an unknown rating is replaced by using the results obtained by applying the linear regression (with default parameters) using the Weka library. This method tries to lower the data variance of missing value estimates, by exploiting the underlying localised or global correlation structure of the data.
- *Filling by logistic regression (LogisticReg)*: In this approach, an unknown rating is replaced by using the results obtained by applying the logistic regression (with default parameters) using the Weka library.
- *AdaBoost (AdaBoost)*: In this approach, an unknown rating is replaced by using the results obtained by applying the Ada Boost over C4.5 decision tree approach, using the Weka library. It can be less susceptible to the over-fitting than most learning algorithms; however it is sensitive to noisy data.

7 Experimental evaluation

7.1 Datasets

We used the MovieLens (consisting of 100K ratings shown by SML and 1M ratings shown by ML) and FilmTrust (FT) datasets for evaluating our algorithm. The MovieLens data set has been used in many research projects [11, 13, 95, 92, 87]. We created the FilmTrust dataset by crawling (on 10th of March 2009) the FilmTrust website. The FilmTrust dataset is shown by FT1. We removed all movies and user from the FT1 dataset with less than 5 ratings and the resulting dataset is shown by FT5. The characteristics of the MovieLens and FilmTrust dataset are shown in Table 1. The sparsity of a dataset is calculated as follows: $\left(1 - \frac{\textit{non zero entries}}{\textit{all possible entries}}\right)$. Figure 1 and 2 show the distribution of ratings in datasets. We observe that in the MovieLens dataset, the rating distribution is skewed towards rating of 4, whereas, in the FilmTrust dataset it is skewed towards ratings between 9 to 10.

7.2 Feature extraction and selection

We downloaded information about each movie in the SML and FT datasets, from IMDB[9]. After stop word (frequently occurring words that carry little information [10]) removal [96] and stemming (removing the case and inflections information from a word and mapping it to the same stem[11]), we constructed a vector of ratings, keywords, tags, genres, directors, actors/actresses, producers, writers, and user reviews given to a movie in IMDB. We used TF-IDF (Term Frequency-Inverse Document Frequency) approach for determining the weights of words in a document (i.e. movie). In our case, we have 5 classes for the MovieLens and 10 classes for the FilmTrust dataset. These vector of features are used to train the classification and regression approaches discussed in Section 6. We used DF-Thresholding feature selection technique to reduce the feature space by eliminating useless noise words—words having little (or no) discriminating power in a classifier, or having low signal-to-noise ratio. We leverage WordNet using Java WordNet Interface (http://projects.csail.mit.edu/jwi/) for overcoming the *synonym* problem between features while finding the similarities among features. The details of training a classifier using these features can be found in our previous work [92].

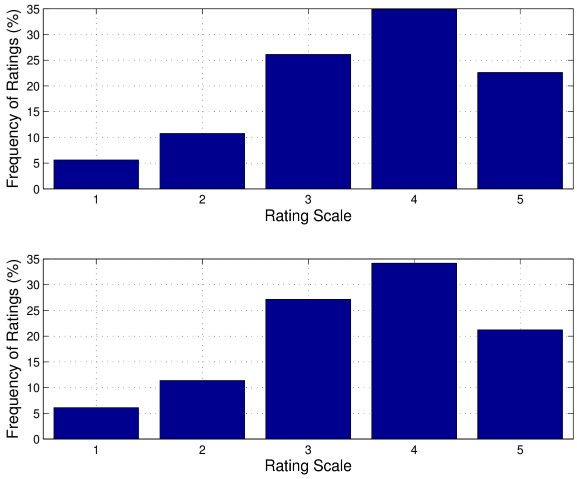


Figure 1: Rating distribution of the MovieLens datasets. The upper plot is for ML dataset and the lower plot is for SML dataset. We observe that, the rating distribution is skewed towards rating of 4.

It must be noted that the text categorisation and recommender system share a number of characteristics. A *Vector Space Model* is the most commonly used document representation technique, in which documents are represented by vectors of words. Each vector component represents a word feature and approaches, such as Boolean weights, $TF - IDF$, normalised $TF - IDF$ [97] etc. can be used for determining the weight of a word in document. The feature space in a typical attribute-value representation can be very large (e.g. 10 000 dimensions and more). A *word-by-document matrix* is used to represent a collection of documents, where each entry symbolises the occurrence of a word in a document. This matrix is typically very sparse, as not every word appears in every document. The recommender systems share the same characteristic. In [98] the authors argue that each user can be viewed as a document and each item rated by a user can be represented by a word

[9] We matched the movie titles, provided by the SML and FT datasets, against the titles in the IMDB (www.imdb.com). The details of the matching algorithm are beyond the scope of this paper.

[10] We used Google's stop word list www.ranks.nl/resources/stopwords.html.

[11] We used Porter Stemmer [27] algorithm for stemming.

Table 1: Characteristics of the datasets used in this work. The MovieLens dataset is shown by SML (100K ratings) and ML (1M ratings), and the FilmTrust dataset is shown by FT1 (original FilmTrust dataset) and FT5 (containing users and movies with at least 5 ratings). Average rating represents the average rating given by all users in the dataset.

Characteristics	**Dataset**			
	ML	**SML**	**FT**1	**TF**5
Number of users	6040	943	1214	1016
Number of movies	3706	1682	1922	314
Number of ratings	1000209	100000	28645	25730
Rating scale	1 (bad)-5 (excellent)	Same as ML	1.0 (bad)-10.0 (excellent)	−−
	(Integer scale)	(Integer scale)	(Floating point scale)	−−
Sparsity	0.955	0.934	0.988	0.919
Max number of ratings given by a user	2314	737	244	133
Max number of ratings given to a movie	3428	583	880	842
Average rating	3.581	3.529	7.607	7.601

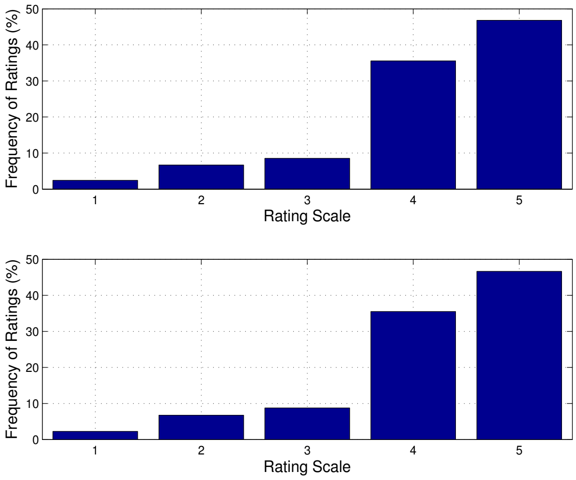


Figure 2: Rating distribution of the FilmTrust datasets. The upper plot is for FT1 dataset and the lower plot is for FT5 dataset. The rating scale is digitised as follows: a rating between 0 to 2.0 is represented by 1, a rating between 2.1 to 4.0 is represented by 2, a rating between 4.1 to 6.0 is represented by 3, a rating between 6.1 to 8.0 is represented by 4 , a rating between 8.1 to 10.0 is represented by 5. We observe that the rating distribution is skewed towards ratings between 8.1 to 10

appearing in a document. Our assumption is slightly different from the one used in [98], we view each user as a document; however we get (content) features against each item rated by a user. Each item is represented by a vector of bags of words and the user profile is represented by a big vector obtained by concatenating the vectors of bags of words of each item rated by a user. In this way, the user profile captured by a recommender system is very similar to the vector space model in text categorisation. Hence, our assumption is that the basic text categorisation algorithms can be applied to recommender system problem and that the results should be comparable.

7.3 Metrics

Several metrics have been used for evaluating recommender systems, which can broadly be categorised into *predictive accuracy metrics*, *classification accuracy metrics*, and *rank accuracy metrics* [68]. The predictive accuracy metrics measure how close is the recommender system's predicted value of a rating, with the true value of that rating assigned by the user. These metrics include mean absolute error, root mean squared error, and normalised mean absolute error, and have been used in research projects such as [19, 16, 18, 12, 11, 99]. The classification accuracy metrics determine the frequency of decisions made by a recommender system, for finding and recommending a good item to a user. These metrics include precision, recall, and $F1$ measure, and have been used in [12, 99]. The last category of metrics, rank accuracy metrics measure the proximity between the ordering predicted by a recommender system to the ordering given by the actual user, for the same set of items. These metrics include half-life utility metric proposed by Brease [16].

Our specific task in this paper is to predict scores for items that have already been rated by actual users, and to check how well this prediction helps users in selecting high quality items. considering this, we have used *Mean Absolute Error (MAE)*, *Receiver Operating Characteristic (ROC) sensitivity*, *precision*, *recall*, and $F1$ *measure*.

7.3.1 Mean absolute error (MAE)

The Mean Absolute Error (MAE) measures the average absolute deviation between the rating predicted by a recommendation algorithm and the true rating assigned by the user. It is computed as follows:

$$MAE = \frac{1}{|\mathcal{D}^{test}|} \sum_{r_{i,u} \in \mathcal{D}^{test}} |\hat{r}_{i,u} - r_{i,u}|,$$

where $r_{i,u}$ and $\hat{r}_{i,u}$ are the actual and predicted values of a rating respectively, and $\mathcal{D}^{test}$ is the set of rating records

Table 2: Classification of items in a document

	Selected	Not Selected	Total
Relevant	I_{rs}	I_{rn}	I_r
Irrelevant	I_{is}	I_{in}	I_i
Total	I_s	I_n	I

in the test set. A rating record is a tuple consisting user ID (Identifier), movie ID, and rating, $< uid, mid,$ where r is the rating a recommender system has to pre It has been used in [16], [12], [11], [42], [46], [71], [13], [36] and [35]. The aim of a recommender syste to minimise the MAE score.

7.3.2 Receiver operating characteristic (ROC) sensitivity

ROC is the extent to which an information filtering system can distinguish between good and bad items. *ROC sensitivity* measures the probability with which a system accept a good item. The ROC sensitivity ranges from 1 (perfect) to 0 (imperfect) with 0.5 for random. To use this metric for recommender systems, we must first determine which items are good (*signal*) and which are bad (*noise*). The guidelines of using this metric can be found in our previous work [92].

7.3.3 Precision, recall, and F1 measure

Precision, recall, and F1 measure evaluate the effectiveness of a recommender system by measuring the frequency with which it helps users selecting/recommending a good item [68]. The most appropriate way to measure the precision and recall in the context of recommender systems, is to predict the top-N items for the known ratings, which can be done by splitting each user's ratings into training and test set, training the model on the training set, and then predicting the top-N items from the test set. Here the underlying assumption is that, the distribution of relevant and irrelevant items in each user's test set, is the same as the true distribution for that user across all items. This has been used in [39].

Information retrieval [85] area, defines "objective" measure for precision, recall and related metric, where the relevance is independent to the user and is only associated with the query. However in context of recommender systems, the term "objective relevance" does not fit well—as every user have different taste, opinions, and reason to rate an item, hence, relevance is inherently "subjective" in recommender systems. The first step in computing the precision and recall is to divide items into two classes: relevant and irrelevant, which is the same as in ROC-sensitivity.

Precision gives us the probability that a selected item is relevant. A precision of 60% means that 6 of of every 10 recommendations for a user will be relevant. A user is more likely to understand the meaning of $x\%$ difference in precision rather comprehending 0.05%-point difference in the MAE [68]. Mathematically, it is defined as follows:

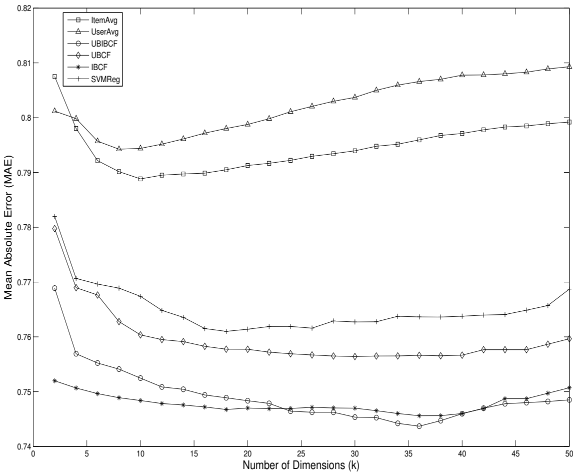


Figure 3: Determining the optimal number of dimensions in the imputed SVD over the training set (SML dataset). The error bars, lying between 0.001 and 0.004 for all approaches, are not shown for reasons of clarity.

$$Precision = \frac{I_{rs}}{I_s}.$$

Recall gives us the probability that a relevant item is selected [68]. Mathematically, it is defined as follows:

$$Recall = \frac{I_{rs}}{I_r}.$$

Precision and recall should be measure together, as it has been claimed that they are inversely proportional to each other, and furthermore, they depend on the size of the resultant vector returned to the user. $F1$ measure [68] combines the precision and recall into a single metric and has been used in many research projects [12, 99]. F1 is computed as follows:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}.$$

We calculated precision, recall, and F1 measures for each user, and reported the average results over all users.

7.4 Evaluation methodology

We performed the striated 5 fold cross validation and reported the average results with standard deviation. Each distinct fold contains 20% random ratings of each user as the test set and the remaining 80% as the training set. We further subdivided our training set into a validation set and training set for measuring the parameters sensitivity. For learning the parameters, we conducted 5-fold cross validation on the 80% training set, by selecting the different test and training set each time, and taking the average of results.

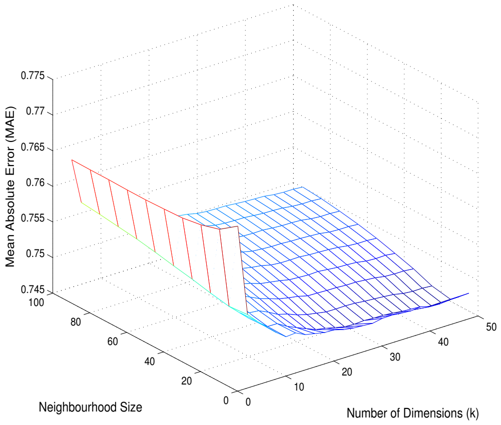


Figure 4: Determining the optimal parameters in user-based CF (for SML dataset with IBUBCF imputation source), through grid search over the training set. The "Number of Dimensions (k)" represents the number of dimensions in the reduced space (representing the k pseudo items) and "Neighbourhood Size" represents the number of most similar users against the active user.

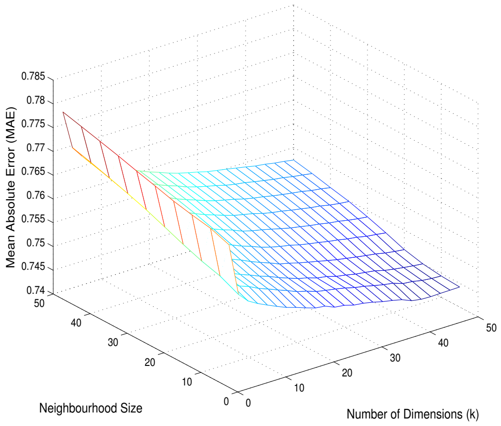


Figure 5: Determining the optimal parameters in item-based CF (for SML dataset with IBUBCF imputation source), through grid search over the training set. The "Number of Dimensions (k)" represents the number of dimensions in the reduced space (representing the k pseudo users) and "Neighbourhood Size" represents the number of most similar items against the target item.

8 Results and discussion

8.1 Learning the optimal parameters

The purpose of these experiments is to determine, which of the parameters affect the prediction quality of the proposed algorithms, and to determine their optimal values.

8.1.1 Finding the optimal number of dimensions

Two factors are important while finding the optimal number of dimensions. First, the number of dimensions must be small enough to make the resulting system scalable and second it must be big enough to capture the important latent information between the users or items. Figure 3 shows how the MAE changes as a function of the number of dimensions (k) in the case of SML dataset. We show results only for the conventional approaches and the ones giving us good results. We observe that, in the case of UBCF, IBCF, and UBIBCF, the MAE keeps on decreasing, reaches its minimum between $k = \{30 - 40\}$, and then starts increasing again. We choose $k = 36$ for these imputation methods. We further observe that the MAE is minimum at $k = 18$, $k = 8$, and $k = 10$ in the case of SVMReg, UserAvg, and ItemAvg respectively. Similarly, we tuned all approaches for the optimal dimensions for other datasets.

8.1.2 Finding the optimal number of neighbours and dimensions for user-based CF

The neighbourhood size is dataset dependent and furthermore change in the distribution and sparsity of the dataset will change the neighbourhood size. The work in [13] finds the optimal number of dimensions by keeping the neighbourhood size fixed to a value, changing the dimensions, and observing the corresponding MAE. The dimension that gives the minimum MAE is recorded to be the optimal one. Then the optimal neighbourhood size can be found by keeping the dimension parameter fixed to the optimal one. This sound a reasonable strategy; however, it does not show how the MAE changes with all possible combinations of both parameters—number of neighbours and dimension. We claim that *grid search* can effectively be used to find the optimal value of neighbourhood size and dimensions.

We performed a series of experiments by changing the dimension each time from 2 to 50 with a difference of 2. For each experiment, we changed the neighbourhood size from 5 to 100 with difference of 10, keeping the dimension parameter fixed, and observed the corresponding MAE. Figure 4 shows that the MAE is minimum at the neighbourhood size of 15. This is in contrast with the neighbourhood size in the classical user-based CF [11], where the MAE decreases with the increase in the neighbourhood size, reaches at its minimum for a specific neighbourhood size ranging from $\{50 - 70\}$, and then starts increasing again. The reason can be that filling the user-item rating matrix with an imputation source and then applying SVD may change the sparsity and distribution of the dataset. We observe in the dimension scale, keeping the neighbourhood size fixed to 15, that the MAE decreases with the increase in the rank of the lower dimension space, reaches at its peak at $k = 46$, and after that it either increases or stays constant. The grid coordinates, which gave the lowest MAE, are recorded to be the best parameter. In the case of UBIBCF imputation source, they found to be 15 for the neighbourhood size and 46 for the dimension. Similarly, we tuned the parameters for all approaches for other datasets.

Table 3: Learning parameter sets α and β over the training set through cross validation. α and β show the relative impact of user-based and item-based CF in a prediction respectively.

Params		**MAE**			
α	β	**ML**	**SML**	**FT1**	**FT5**
0.1	0.9	0.710	0.740	1.483	1.449
0.2	0.8	0.709	0.739	1.476	1.440
0.3	0.7	0.708	0.738	1.471	1.434
0.4	0.6	**0.706**	**0.736**	**1.470**	1.430
0.5	0.5	0.707	0.737	1.471	**1.429**
0.6	0.4	0.708	0.737	1.475	1.431
0.7	0.3	0.707	0.738	1.481	1.435
0.8	0.2	0.709	0.739	1.489	1.442
0.9	0.1	0.712	0.741	1.499	1.452

8.1.3 Finding the optimal number of neighbours and dimensions for item-based CF

We varied the number of dimensions from 2 to 50 with a difference of 2, and the number of neighbours from 5 to 50 with a difference of 5. Figure 5 shows that the MAE is minimum for the neighbourhood size of 5. After that, an increase in the neighbourhood size increases the MAE. In the dimension scale, keeping the neighbourhood size fixed to 5, we note that the MAE decreases with the increase in the rank of the lower dimension space, reaches at its peak at $k = 44$ and after that it either increases or stay constant. In the case of UBIBCF imputation source, the optimal parameters are found to be 5 for neighbourhood size and 44 for dimension. Similarly, we tuned the parameters for all approaches for other datasets.

8.1.4 Finding the optimal values of parameters α and β

Parameters α and β (refer to Section 5.2) determine the relative weights of user-based and item-based CF in the final prediction. The 9 parameter sets were generated by producing all possible combination of parameters values, ranging from 0.1 to 1.0 with differences of 0.1[12]. Table 3 present the parameter sets learned. The parameters sets $\alpha = 0.6, \beta = 0.4$; $\alpha = 0.6, \beta = 0.4$; $\alpha = 0.6, \beta = 0.4$; and $\alpha = 0.5, \beta = 0.5$ gave the lowest *MAE* in the case of ML, SML, FT1 and FT5 dataset respectively. It is worth noting that the values of parameters are found different for the MovieLens and FilmTrust dataset. We note that the item-based CF has more weight in the final prediction.

8.2 Performance evaluation of different imputation sources (Algorithm 1, $ImpSvd$)

The results obtained by *ImputedSVD* (Algorithm 1) under different imputation sources are shown in Table 4. Note that, we only show the best results obtained by varying k from 1 to 50. The table shows that the imputation methods SVM regression, UBCF, IBCF, and UBIBCF give more accurate results than others. The % decrease in MAE over the baseline method ItemAvg is found to be: (1) 4.79%, 5.61%, and 6.57% in case of UBCF, IBCF, and UBIBCF respectively for ML dataset (2) 5.16%, 5.55%, 5.94%, and 7.23% in case of SVM regression, UBCF, IBCF, and UBIBCF respectively for SML dataset (3) 17.0%, 14.70%, 14.52%, and 15.52% in case of SVM regression, UBCF, IBCF, and UBIBCF respectively for FT1 dataset (4) 5.86%, 2.70%, 2.56%, and 4.58% in case of SVM regression, UBCF, IBCF, and UBIBCF respectively for FT5 dataset. The ranking of the algorithms (with respect to the MAE) with the respective *p*-value in case of pair t test is found to be: (1) **UBIBCF** ($p < 0.001$) > **IBCF** ($p < 0.05$) > **UBCF** ($p < 0.05$) for ML dataset (2) **UBIBCF** ($p < 0.001$) > **IBCF** ($p < 0.001$) > **UBCF** ($p < 0.001$) >**SVMReg** ($p < 0.05$) for SML dataset (3) **SVMReg** ($p < 0.001$) > **UBIBCF** ($p < 0.001$) > **IBCF** ($p < 0.001$) >**UBCF** ($p < 0.001$) for FT1 dataset (4) **SVMReg** ($p < 0.001$) > **UBIBCF** ($p < 0.005$) > **IBCF** ($p < 0.001$) >**UBCF** ($p < 0.005$) for FT5 dataset. Furthermore, the proposed imputation sources give 5% to 10% improvement over the baseline approach, in terms of ROC-sensitivity, precision, recall, and F1 scores (results not shown due to space limits). These results indicate that proposed approaches accurately (1) predict an unknown rating for a user, (2) recommend the top-N items a user would like the most.

The FilmTrust dataset is a good example of the real world recommender system's characteristics. It has imbalanced data, i.e. a user may have 1 rating and other may have more than 100 ratings and the same is true for items as well. It captures well the *new user cold-start* and *new item cold-start* problems. What is evident from table 4 is that approximating missing values in the user-item rating matrix with the baseline approach gives the worst results. We observe that the SVMReg approach outperform others in FilmTrust dataset. The reason is that, the FilmTrust dataset is well suited to regression algorithms, as it has floating point scale (refer to Section 7.1). It is worth noting that, in the case of FT1 dataset, the UserAvg imputation approach gives accurate (or comparable) results as compared to the other approaches. We believe that it is due to the distribution of the dataset—in the FilmTrsut dataset, majority of the users have rated the popular set of movies and their rating tends to match the average user rating. In the case of FT5 dataset, the performance of approaches, even conventional ones, improves simply because we have removed users and items with less clear profiles. We note that, in FT5 case, again the baseline approach gives the worse re-

[12] We assume that $\alpha + \beta = 1$ without the loss of generalisation.

Table 4: Best MAE observed in different imputation sources. The best results have been shown in bold. K represents the number of dimensions, which gave the most accurate results.

Imp. Sr.	**Best *MAE***				**Dimension (k)**		
	ML	**SML**	**FT**1	**FT**5	**ML**	**SML**	**FT**
Zeros	2.425 ± 0.002	2.321 ± 0.002	4.354 ± 0.020	3.898 ± 0.031	26	12	2
Rand	1.092 ± 0.002	1.072 ± 0.005	2.214 ± 0.019	2.064 ± 0.021	14	4	2
ItemAvg	0.730 ± 0.002	0.774 ± 0.002	1.700 ± 0.011	1.483 ± 0.012	22	10	10
UserAvg	0.759 ± 0.002	0.778 ± 0.002	1.452 ± 0.016	1.433 ± 0.005	22	8	4
UserItem Avg	0.724 ± 0.002	0.754 ± 0.003	1.527 ± 0.016	1.442 ± 0.014	30	12	14
Uniform Dist	0.911 ± 0.002	0.905 ± 0.002	2.061 ± 0.031	1.933 ± 0.023	10	4	2
Nor_U	0.790 ± 0.002	0.810 ± 0.002	1.505 ± 0.018	1.491 ± 0.010	4	2	2
Nor_I	0.766 ± 0.002	0.800 ± 0.002	1.796 ± 0.019	1.562 ± 0.020	2	2	2
UBCF	0.695 ± 0.002	0.731 ± 0.001	1.450 ± 0.016	1.442 ± 0.015	40	36	4
IBCF	0.689 ± 0.002	0.728 ± 0.003	1.453 ± 0.010	1.445 ± 0.013	40	36	6
UBIBCF	**0.682 ± 0.002**	**0.718 ± 0.002**	1.436 ± 0.014	1.415 ± 0.017	40	36	6
KNN	−−	0.804 ± 0.005	1.485 ± 0.017	1.479 ± 0.019	−−	18	4
WKNN	−−	0.793 ± 0.002	1.481 ± 0.007	1.474 ± 0.008	−−	18	4
NBClass	−−	0.775 ± 0.005	1.475 ± 0.016	1.468 ± 0.017	−−	26	8
SVMClass	−−	0.763 ± 0.004	1.455 ± 0.014	1.445 ± 0.016	−−	18	6
C4.5	−−	0.781 ± 0.003	1.495 ± 0.012	1.485 ± 0.015	−−	22	10
SVMReg	−−	0.734 ± 0.004	**1.411 ± 0.015**	**1.396 ± 0.019**	−−	18	6
Linear Reg	−−	0.783 ± 0.003	1.447 ± 0.014	1.437 ± 0.018	−−	16	4
Logistic Reg	−−	0.781 ± 0.004	1.443 ± 0.015	1.434 ± 0.017	−−	14	4
AdaBoost	−−	0.772 ± 0.006	1.476 ± 0.014	1.468 ± 0.018	−−	26	8

Table 5: The best MAE observed in different imputation sources in the case of item-based CF applied over the reduced dataset. The best results have been shown in bold.

Imputation Source	**Best *MAE***			
	ML	**SML**	**FT**1	**FT**5
ItemAvg	0.741 ± 0.002	0.781 ± 0.0018	1.702 ± 0.017	1.475 ± 0.018
UserAvg	0.767 ± 0.002	0.788 ± 0.004	1.496 ± 0.015	1.442 ± 0.016
UBCF	0.721 ± 0.002	0.739 ± 0.003	1.483 ± 0.018	1.434 ± 0.011
IBCF	0.701 ± 0.002	0.738 ± 0.002	1.459 ± 0.013	1.462 ± 0.014
UBIBCF	**0.691 ± 0.002**	**0.723 ± 0.004**	1.432 ± 0.017	1.418 ± 0.018
SVMReg	−−	0.744 ± 0.002	**1.417 ± 0.019**	**1.404 ± 0.019**

Table 6: The best MAE observed in different imputation sources in the case of user-based CF applied over the reduced dataset. The best results have been shown in bold.

Imputation Source	**Best *MAE***			
	ML	**SML**	**FT**1	**FT**5
ItemAvg	0.742 ± 0.002	0.776 ± 0.002	1.731 ± 0.017	1.465 ± 0.018
UserAvg	0.773 ± 0.002	0.786 ± 0.002	1.483 ± 0.014	1.439 ± 0.015
UBCF	0.709 ± 0.002	0.734 ± 0.003	1.465 ± 0.015	1.422 ± 0.017
IBCF	0.706 ± 0.002	0.732 ± 0.002	1.446 ± 0.017	1.445 ± 0.018
UBIBCF	**0.692 ± 0.002**	**0.722 ± 0.003**	1.445 ± 0.019	1.419 ± 0.019
SVMReg	−−	0.743 ± 0.002	**1.416 ± 0.018**	**1.401 ± 0.019**

Table 7: Best MAE, ROC-Sensitivity, Precision, Recall, and F1 observed in the case of hybrid recommender systems proposed in algorithm 3. The SMVReg is used for FilmTrust dataset and the UBIBCF is used for the remaining datasets as imputation source, prior applying the SVD. The optimal parameters are learned over the training set using grid search. Precision, Recall, and F1 have been measured over top 20 recommendations.

DataSet	MAE	ROC	Precision	Recall	F1
ML	0.684 ± 0.002	0.790 ± 0.002	0.518 ± 0.005	0.595 ± 0.003	0.524 ± 0.004
SML	0.717 ± 0.002	0.695 ± 0.006	0.543 ± 0.005	0.555 ± 0.003	0.513 ± 0.005
FT1	1.409 ± 0.013	0.566 ± 0.008	0.591 ± 0.008	0.568 ± 0.007	0.549 ± 0.007
FT5	1.394 ± 0.016	0.578 ± 0.008	0.598 ± 0.011	0.574 ± 0.008	0.556 ± 0.012

sults.

8.3 Performance evaluation of CF applied over the reduced dataset (Algorithm 2, $ImpSvd_{CF}$)

Table 5 and 6 show that the proposed approaches give more accurate results than the conventional ones, when we apply CF over the reduced dataset.

It is worth noting that the results (in general) obtained by applying CF over the reduced dataset do not give any advantage over the results obtained by applying the SVD. However, in the case of FilmTrust dataset, some of the proposed approaches (UBCF, IBCF, UBIBCF) give (insignificantly) better results when CF is applied over the reduced dataset. It might be due to the reason that the FilmTrust dataset is very sparse, which implies the the latent structure between movies and users might not be captured by applying the SVD, and can be found by applying CF over the reduced dataset. Another thing to note is that, the results obtained in the case of proposed approaches are (almost) equivalent to the ones obtained in the proposed Imputed SVD. Furthermore, in general, user-based CF performs better than the item-based CF in case of FT1, FT5, and SML dataset, whereas, item-based CF performs better than user-based CF in case of ML dataset.

8.4 Performance evaluation of hybrid recommender system (Algorithm 2, $ImpSvd_{CF}^{hybrid}$)

User-based and item-based CF can be combined linearly. Table 7 shows that by linearly combining the UBCF and IBCF, in case of UBIBCF imputation source, gives the improved results with MAE less than 0.684, 0.717, 1.409, and 1.394 in case of SML, ML, FT1, and FT5 datasets respectively. The reason of improvements in the results is that user-based and item-based CF focus on different kind of relationship in the dataset. In certain cases, user-based CF may be useful in identifying different kind of relationship that item-based CF will fail to recognize; for example, if none of the items rated by an active user are closely related to the target item i_t, then it is beneficiary to switch to user-oriented perspective that may find set of users very similar to the active user, who rated i_t.

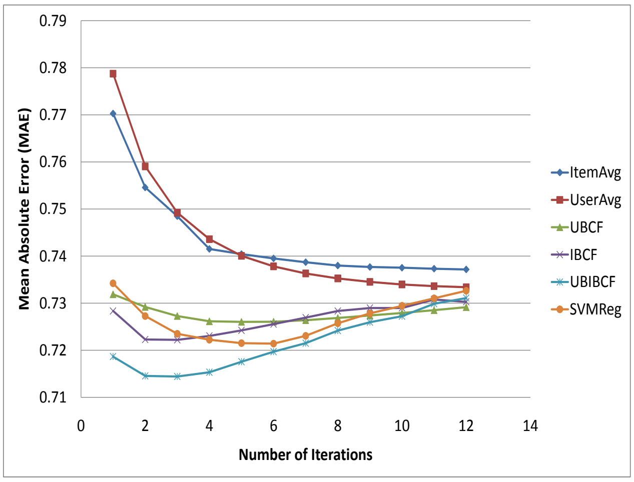


Figure 6: Comparing the proposed approaches with others in the case of iterative SVD (fixed dimension case), over SML dataset. X-axis shows the number of iterations and y-axis shows the corresponding MAE observed. The proposed approaches converge much quicker as compared to the conventional ones. The error bars (< 0.001 for all approaches) are not shown for reasons of clarity.

8.5 Performance evaluation of iterative SVD (Algorithm 3, $ItrSvd$)

There are two options to find the optimal number of dimensions in the $ItrSvd$ algorithm; (1) learning the optimal number of dimensions in the first iteration using the validation set and keeping them fixed for all the iterations, and (2) learning the optimal number of dimensions in each iteration using the validation. In the following, we represent the former case with *fixed dimension* and the latter one with *variable dimension*. We first show results for the fixed dimension and then proceed to the variable dimension case.

Figure 6 shows how the MAE changes with the number of iterations in SML dataset. We observe that the conventional approaches converge much slower as compared to the proposed ones. Figure 6 shows that in case of the baseline approach the MAE keeps on decreasing until it converges after 10 iterations. The minimum MAE observed after 10 iterations is 0.738. The MAE in case of the proposed approaches is shown at the lower plot of the figure. We observe that the MAE is much lower as com-

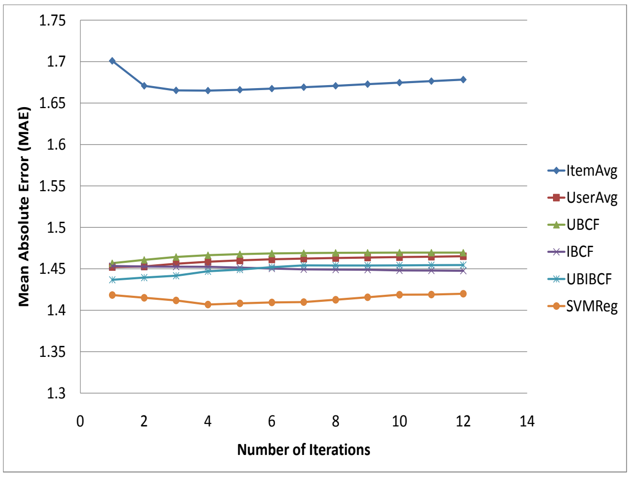


Figure 7: Comparing the proposed approaches with others in the case of iterative SVD (fixed dimension case), over FT1 dataset. X-axis shows the number of iterations and y-axis shows the corresponding MAE observed. The conventional approach converges much quicker than the others; however, the MAE observed after convergence is much higher than the proposed ones. The error bars (lying between 0.001 and 0.004 for all approaches) are not shown for reasons of clarity.

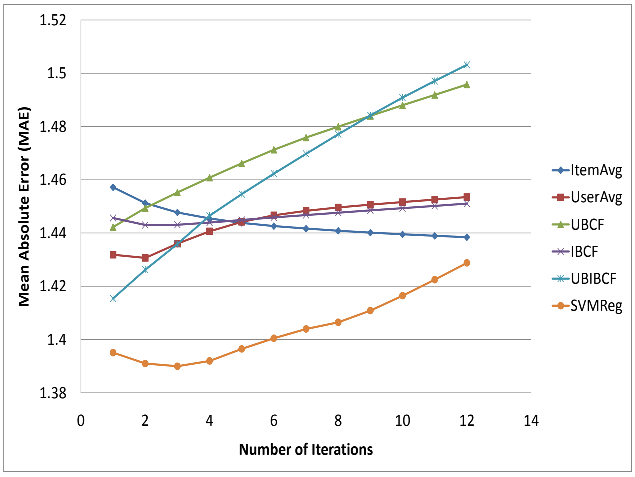


Figure 8: Comparing the proposed approaches with others in the case of iterative SVD (fixed dimension case), over FT5 dataset. X-axis shows the number of iterations and y-axis shows the corresponding MAE observed. The proposed approaches converge much quicker as compared to the conventional ones. The error bars (lying between 0.001 and 0.004 for all approaches) are not shown for reasons of clarity.

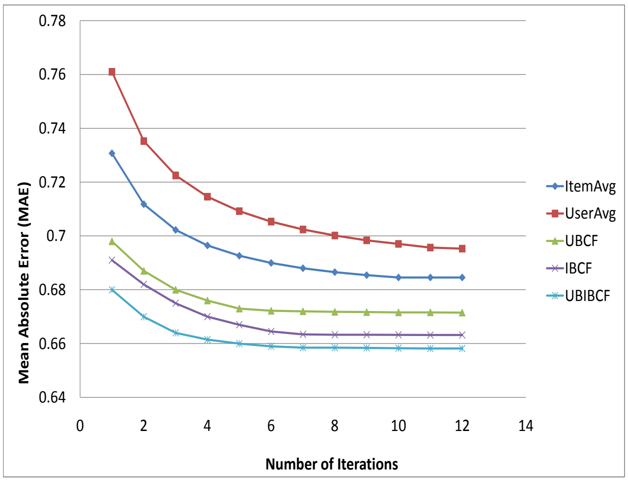


Figure 9: Comparing the proposed approaches with others in the case of iterative SVD (fixed dimension case), over ML dataset. X-axis shows the number of iterations and y-axis shows the corresponding MAE observed. The proposed approaches converges much quicker as compared to the conventional ones. The error bars (< 0.001 for all approaches) are not shown for reasons of clarity.

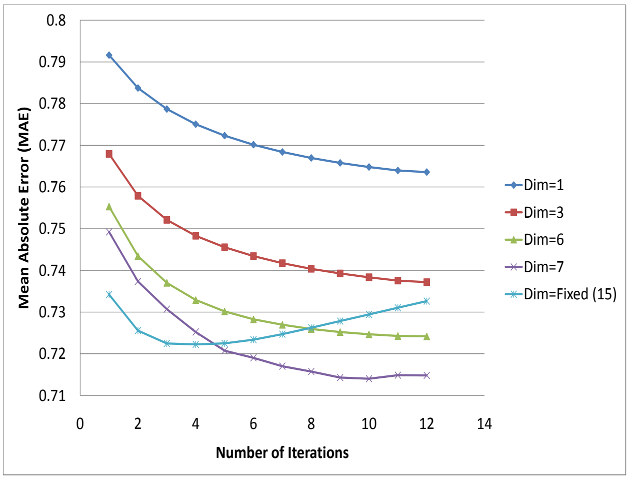


Figure 10: how the MAE changes with the increase in the number of dimensions for SVM regression approach, over SML dataset. X-axis shows the number of iterations and y-axis shows the corresponding MAE observed. Fixed dimensions represent the case, where the optimal numbers of dimensions are learned in the first iteration through the training set and kept fixed for all iterations. We observe that the results are highly dependent on the dimension parameter. The error bars (< 0.001 for all approaches) are not shown for reasons of clarity.

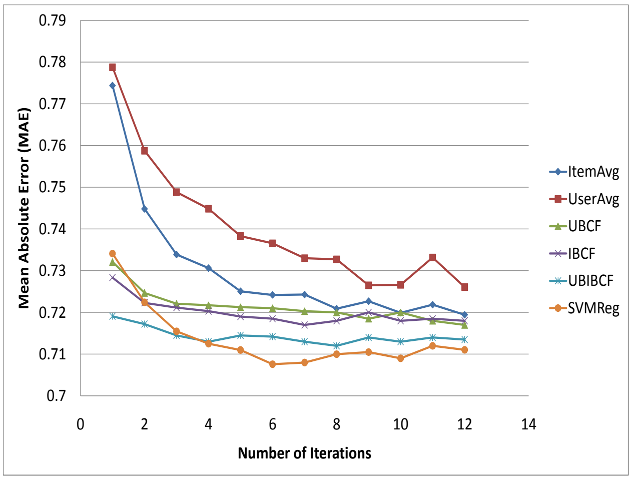


Figure 11: Comparing the proposed approaches with others in the case of iterative SVD (variable dimension case) over SML dataset. X-axis shows the number of iterations and y-axis shows the corresponding MAE observed. The proposed approaches converge much quicker as compared to the conventional ones. The error bars (< 0.001 for all approaches) are not shown for reasons of clarity.

pared to the conventional approaches, even in the first iteration and it converges much faster than conventional approaches. The IBCF and UBIBCF converges after $2-3$ whereas the UBCF and SVMReg converges after $5-6$ iterations, and then the MAE starts increasing, which may be due to the over-fitting. We further observe that approximating the missing values using CF gives better results as compared to the SVM regression.

The performance of the baseline approach is even worse for FT1 dataset. Figure 7 shows that in the case of ItemAvg, the MAE is 1.7 at first iteration, keeps on decreasing until it reaches at its minimum to 1.66 after $3-4$ iterations. The algorithm converges after 3 to 4 iterations and then the MAE starts increasing again. IBCF and SVMReg approaches show the similar behaviour, where the MAE reaches at its minimum after $4-5$ iterations, and then starts increasing again. The remaining approaches do not show any improvement in the MAE with an increase in the number of iterations. We note that the SVMReg imputation approach gives more accurate results with MAE = 1.40.

The results in the case of FT5 dataset are shown in Figure 8. The results of the baseline approach are surprisingly good where the MAE keeps on decreasing, until it converges after $10-12$ iterations. The lowest MAE observed after 12 iterations is still higher than the ones obtained in first iteration of the proposed approaches. The MAE in the case of UBCF and UBIBCF imputation approaches, increases with the increase in the number of iterations, which might be due to over-fitting. The MAE in the case of remaining imputation approaches decreases with the increase in number of iterations, reaches at its minimum at $2-3$ iterations and then starts increasing. Again, the SVMReg imputation approach gives more accurate results.

The results in case of ML dataset are shown in Figure 9. We observe that they show the similar behaviour as in the case of SML dataset. Based on the experimental results, we can conclude that the proposed approaches produce anytime [100] recommendations and converge much faster than the conventional ones. It is worth noting that computing SVD is very computation expensive (regardless it is done off-line), which implies finding the solution in the iterative SVD using the baseline approach is not pragmatic, and hence proposed approaches should be used to save time and memory.

The optimal number of dimensions can be learned at each iteration using the training set, though it is very expensive however, it may increase accuracy. To check how the MAE changes with the dimension parameter, we show results in the case of SVMReg imputation approach over SML dataset for different number of dimensions. *"Dim=Fixed (15)"* represents the case, where we learn the optimal number of dimensions through the training set and then these dimensions are kept fixed for all iterations. We also consider other cases, where we (randomly) choose dimension parameter to be 1, 3, 5, and 7. Figure 10 shows that the MAE is highly dependent on the dimension parameter. To further investigate the results, we perform experiments where the optimal numbers of dimensions are learned at each iteration. We only did experiments with SML and FT datasets, as we found it very expensive for ML dataset, both in terms of memory requirements and computation cost.

There were no clear improvement (and patterns) in results to be discussed in the case of FT5 dataset, may be due to the reason that we do not have enough data to learn the optimal parameters. We discuss the results in the case of SML data set, though they show the similar behaviour for FT1 dataset. We choose SML dataset as it has heavily been used in the literature and it is easy to reproduce the results. The results[13] are shown in Figure 11.

Figure 11 shows that learning the optimal number of dimensions at each iteration decreases the MAE of all approaches in general. Furthermore, all approaches except the SVMReg show the same behaviour as shown by the fixed iteration case. The MAE in the case of SVMReg approach keeps on decreasing with the increase in the number of iterations, reaches at its minimum at iteration 6, and then either stays stable or increases again. We further observe that the SVMReg outperform others, which is not true in fixed iteration case.

Table 8 compares the performance—in terms of MAE—of different approaches under the iterative SVD. The optimal number of dimensions are learned in the first iterations and kept fixed. We observe that the proposed approaches produce better results than the baseline approach. We further observe that, in the case of MovieLens dataset, the

[13] Note that we used the training data to estimate the best parameters and used an independent test set to give the unbiased estimate of the generalisation error.

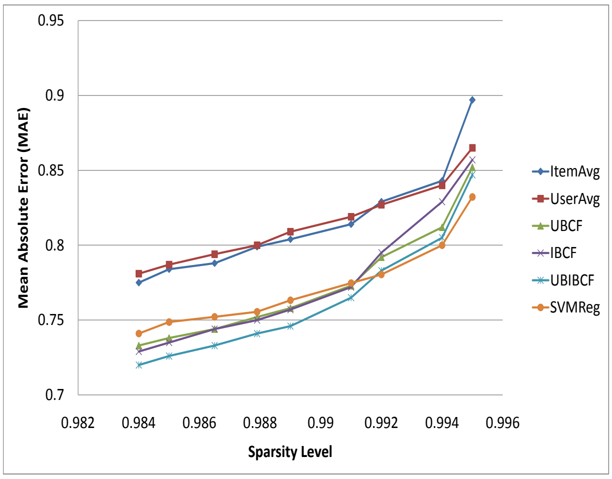


Figure 12: How sparsity affects the performance of different approaches, for SML dataset. Algorithm 1 was used to make recommendations.

UBIBCF and IBCF approaches outperform others; whereas in the case of FilmTrust dataset, the SVMReg and UBCF perform the best. The performance comparisons in terms of ROC-sensitivity and F1 measure are given in Appendix.

8.6 Performance evaluation under different sparsity levels

To check the performance of the proposed approaches under sparsity, we increased the sparsity level of the training set by dropping some randomly selected rating records. Whereas, we kept the test set same for each sparse training set. We used algorithm 1 to make recommendations. Figure 12 shows how different approaches perform under sparse conditions. The figure shows that the performance of the conventional approaches suffer more than the proposed ones. The reason is that under sparse conditions, the item and user averages might be misleading resulting in erroneous recommendations.

It must be noted that under very sparse conditions (sparsity ≥ 0.994), SVMReg outperforms the rest. It is because, with an increase in the sparsity, we do not have comprehensive user/item rating profile that can be used to make predictions for other unknown items. However, we can capture user profile in terms of the important features in which a user is interested, resulting in improved user profile and predictions. We also note that, the remaining approaches give the equivalent results. Hence, under very sparse condition, the SVMReg can be used provided enough resources are available, and the conventional approaches can be used otherwise.

8.7 Performance evaluation under cold-start scenarios

8.7.1 New user cold-start scenario

For testing the performance of approaches under new user cold-start scenario, we selected 50 random users, and kept their number of ratings in the training set to 2, 5, 10, 15, and 20. The corresponding MAE; represented by $MAE2$, $MAE5$, $MAE10$, $MAE15$, and $MAE20$ is shown in Table 9. Table 9 shows that the conventional approaches suffer the most under this scenario. It is worth noting that, when a user has rated less than (or equal to) 10 movies, then UserItemAvg gives the best results; however, as a user rates more items, the CF and SVMReg give reliable recommendation as shown by the table.

8.7.2 New item cold-start scenario

For testing the performance of approaches under new item cold-start scenario, we selected 50 random items, and kept the number of users in the training set who have rated the these item to 2, 5, 10, 15, and 20. The corresponding MAE; represented by $MAE2$, $MAE5$, $MAE10$, $MAE15$, and $MAE20$ is shown in Table 10. Table 10 shows that the SVMReg gives the best performance when an item has been rated by less than (or equal) to 10 items, and UBCF gives the best performance otherwise. We note that the IBCF does not perform very well as compared to the UBCF, the reason is we do not have comprehensive item rating profiles for finding other similar items. The reason for the good results in the case of SVMReg is the same as discussed in Section 8.6.

8.8 Performance evaluation under long tail scenario

To test the performance of the proposed algorithms under long tail scenario, we created the artificial long tail scenario by randomly selecting the 80% of items in the tail. The number of ratings given in the tail part were varied between 2, 4, 6, 8, and 10. The results, shown in Table 11, demonstrated the similar behaviour as in the case of new item case.

8.9 Performance evaluation under different training and test sizes

We performed experiments with different sizes of the test and train set by randomly dividing the rating records into $X\%$ training set and a $(100-X)\%$ test set. A value of $X = 20\%$ for SML dataset indicates that $100\,000$ ratings have been divided into $20\,000$ train cases and $80\,000$ test cases. Table 12 shows that the proposed approaches outperform others at each value of X. We note that the SVMReg gives the best performance for smaller training set sizes. The reason is the same as discussed in Section 8.6.

Table 8: Comparing the MAE observed in different imputation approaches under the iterative SVD (fixed dimension case). Only the conventional approaches and the approaches which gave the best results are compared. The best results have been shown in bold.

Imputation Source	**Best MAE**			
	ML	**SML**	**FT**1	**FT**5
ItemAvg	0.685 ± 0.002	0.738 ± 0.002	1.661 ± 0.003	1.438 ± 0.012
UserAvg	0.697 ± 0.002	0.734 ± 0.002	1.451 ± 0.003	1.430 ± 0.005
UBCF	0.672 ± 0.002	0.723 ± 0.002	1.450 ± 0.003	1.442 ± 0.013
IBCF	0.664 ± 0.002	0.722 ± 0.003	1.448 ± 0.003	1.442 ± 0.017
UBIBCF	**0.659 ± 0.002**	**0.715 ± 0.002**	1.436 ± 0.003	1.418 ± 0.013
SVMReg	−−	0.721 ± 0.003	**1.401 ± 0.003**	**1.390 ± 0.015**

Table 9: Comparing MAE observed in different imputation approaches under **new-user cold start scenario**, for SML dataset. Only the conventional approaches and the approaches, which gave the best results are compared. The best results have been shown in bold.

Imp. Sr.	**Best MAE**				
	MAE2	**MAE**5	**MAE**10	**MAE**15	**MAE**20
ItemAvg	0.908 ± 0.041	0.887 ± 0.042	0.885 ± 0.041	0.883 ± 0.041	0.882 ± 0.041
UserAvg	1.087 ± 0.049	0.928 ± 0.044	0.903 ± 0.043	0.878 ± 0.043	0.877 ± 0.042
UserItemAvg	**0.901 ± 0.041**	**0.855 ± 0.040**	**0.850 ± 0.040**	0.843 ± 0.040	0.839 ± 0.040
UBCF	1.080 ± 0.049	0.886 ± 0.043	0.865 ± 0.043	0.841 ± 0.042	0.825 ± 0.041
IBCF	1.082 ± 0.050	0.896 ± 0.043	0.868 ± 0.043	0.844 ± 0.043	0.817 ± 0.042
UBIBCF	1.071 ± 0.049	0.891 ± 0.043	0.862 ± 0.042	**0.837 ± 0.043**	**0.816 ± 0.041**
SVMReg	0.962 ± 0.050	0.912 ± 0.044	0.873 ± 0.043	0.841 ± 0.042	0.836 ± 0.042

Table 10: Comparing MAE observed in different imputation approaches under **new-item cold start scenario**, for SML dataset. Only the conventional approaches and the approaches which gave the best results are compared. The best results have been shown in bold.

Imp. Sr.	**Best MAE**				
	MAE2	**MAE**5	**MAE**10	**MAE**15	**MAE**20
ItemAvg	1.010 ± 0.064	0.876 ± 0.052	0.854 ± 0.054	0.840 ± 0.056	0.838 ± 0.056
UserAvg	0.876 ± 0.059	0.874 ± 0.057	0.872 ± 0.058	0.870 ± 0.058	0.867 ± 0.057
UserItemAvg	0.865 ± 0.056	0.833 ± 0.054	0.832 ± 0.052	0.824 ± 0.055	0.822 ± 0.054
UBCF	0.911 ± 0.061	0.829 ± 0.052	0.810 ± 0.053	**0.800 ± 0.059**	**0.790 ± 0.053**
IBCF	0.840 ± 0.059	0.834 ± 0.059	0.829 ± 0.060	0.818 ± 0.056	0.813 ± 0.058
UBIBCF	0.858 ± 0.060	0.824 ± 0.053	0.812 ± 0.057	0.802 ± 0.0056	0.795 ± 0.055
SVMReg	**0.822 ± 0.062**	**0.815 ± 0.052**	**0.809 ± 0.056**	0.804 ± 0.057	0.802 ± 0.057

Table 11: Comparing the MAE observed in different imputation methods under the **long tail scenario**, for the SML dataset. The best results are shown in bold font.

Imp. Sr.	**Best MAE**				
	MAE2	**MAE**4	**MAE**6	**MAE**8	**MAE**10
ItemAvg	1.090 ± 0.003	0.891 ± 0.003	0.879 ± 0.003	0.867 ± 0.003	0.861 ± 0.003
UserAvg	0.881 ± 0.003	0.878 ± 0.003	0.869 ± 0.003	0.866 ± 0.003	0.865 ± 0.002
UserItemAvg	0.884 ± 0.003	0.882 ± 0.003	0.871 ± 0.003	0.863 ± 0.003	0.861 ± 0.003
UBCF	0.881 ± 0.003	0.874 ± 0.003	0.847 ± 0.003	0.838 ± 0.003	0.819 ± 0.002
IBCF	0.886 ± 0.003	0.875 ± 0.003	0.861 ± 0.003	0.860 ± 0.003	0.856 ± 0.003
UBIBCF	0.882 ± 0.003	0.869 ± 0.003	0.844 ± 0.003	0.836 ± 0.003	0.824 ± 0.002
SVMReg	**0.879 ± 0.002**	**0.865 ± 0.002**	**0.842 ± 0.002**	**0.833 ± 0.002**	**0.817 ± 0.002**

Table 12: Comparing MAE observed in different imputation approaches under **varying training set sizes**, for SML dataset. Only the conventional approaches and the approaches which gave the best results are compared. The best results have been shown in bold.

Imp. Sr.	**Best MAE**			
	$X = 20\%$	$X = 40\%$	$X = 60\%$	$X = 80\%$
ItemAvg	0.838 ± 0.004	0.809 ± 0.005	0.788 ± 0.005	0.774 ± 0.002
UserAvg	0.839 ± 0.003	0.818 ± 0.005	0.792 ± 0.005	0.778 ± 0.002
UserItemAvg	0.798 ± 0.003	0.784 ± 0.005	0.767 ± 0.005	0.754 ± 0.002
UBCF	0.807 ± 0.004	0.766 ± 0.005	0.746 ± 0.006	0.732 ± 0.003
IBCF	0.804 ± 0.004	0.762 ± 0.005	0.740 ± 0.006	0.730 ± 0.003
UBIBCF	0.802 ± 0.005	0.760 ± 0.005	$\mathbf{0.733 \pm 0.006}$	$\mathbf{0.721 \pm 0.003}$
SVMReg	$\mathbf{0.796 \pm 0.005}$	$\mathbf{0.756 \pm 0.006}$	0.748 ± 0.007	0.736 ± 0.003

Table 13: A comparison of the proposed algorithm with existing in terms of cost (based on [13]) and accuracy metrics. The SMVReg is used for FilmTrust dataset and the UBIBCF is used for the remaining datasets as imputation source, prior applying the SVD. t represents the number of iterations in the EM algorithm.

Algorithm	**Off-line Cost**	**On-line Cost**	**Best *MAE***			
			ML	**SML**	**FT**1	**FT**5
User-based CF with DV	$O(NM^2)$	$O(NM)$	0.706	0.746	1.442	1.416
Item-based CF	$O(N^2M)$	$O(N^2)$	0.705	0.744	1.433	0.418
Baseline SVD	$O(M)$	$O(1)$	0.730	0.774	1.700	1.483
Baseline Item -Based SVD	$O(M)$	$O(N^2)$	0.741	0.781	1.702	1.522
$ImpSvd$	$O(M^2N)+$ $O(N^2M)+O(N^3)$	$O(1)$	0.682	0.718	1.411	1.396
$ImpSvd^{ib}_{CF}$	$O(M^2N)+$ $O(N^2M)+O(N^3)$	$O(N^2)$	0.691	0.723	1.417	0.404
$ImpSvd^{ub}_{CF}$	$O(M^2N)+$ $O(N^2M)+O(N^3)$	$O(NM)$	0.692	0.722	1.416	0.401
$ImpSvd^{hybrid}_{CF}$	$O(M^2N)+$ $O(N^2M)+O(N^3)$	$O(NM)$	0.684	0.717	1.409	1.394
$ItrSvd$	$t * O(M^2N)+$ $O(N^2M)+O(N^3)$	$O(1)$	$\mathbf{0.659}$	$\mathbf{0.715}$	$\mathbf{1.401}$	$\mathbf{1.390}$

8.10 A comparison of the proposed algorithms with others

8.10.1 Direct comparison

We compared our algorithms with four different algorithms: user-based CF with default voting propose in [16] (shown by *User-Based CF with DV* in Table 13), item-based CF propose in [11] (shown by *Item-Based CF* in Table 13), a simple SVD based approach proposed in [12] (shown by *Baseline SVD* in Table 13), an item-based CF approach applied over the reduced user-item rating matrix, proposed in [46] (shown by Baseline Item-Based SVD in Table 13). Furthermore, we tuned all algorithms for the best mentioning parameters. For the proposed algorithms, we used UBIBCF and SVMReg as imputation sources in MovieLens and FilmTrust dataset respectively.

Table 13 shows the cost of the proposed algorithms and others with the corresponding lowest MAE. The table shows that the proposed algorithms are scalable and practical as they have on-line cost less than or equal to the cost of other algorithms; however they give much lower MAE. It must be noted that, the baseline algorithms do not perform very well as compared to the user-based and item-based CF applied over the original user-item rating matrix, which is in contrast with the work proposed in [13][14]. The proposed $ItrSvd$ algorithm performs the best out of all of them; however, it would incur the biggest off-line cost (depending on the number of iterations required to converge), and must be used given the availability of sufficient resources. The same is true for the $ImpSvd_{CF}^{hybrid}$, which gives more accurate results compared to baseline or simple CF; however, it would incur the greater cost. The $ImpSvd$ algorithm comes the next, and can be used if we want the lowest off-line cost (as SVD is applied only once), fast on-line performance, and prefer (good) accuracy.

8.10.2 Indirect comparison

In this section, we compare our results with other algorithms indirectly, i.e. we take the results[15] from the respective papers without re-implementing them, which might make the comparison less than ideal. The test procedures used in these papers are different from ours. We used a standard approach of testing using 5 fold-cross validation.

A comparison in terms of Normalised MAE (NMAE)[16] of the algorithms is given in Table 14. In Table 14, the URP represents the algorithm proposed in [101], Attitude represents the algorithm proposed in [102], MatchBox is proposed in [103], MMMF represents the maximum margin matrix factorisation algorithm proposed in [104], Item has been proposed in [105], E-MMF represents the ensemble maximum margin matrix factorisation technique proposed in [106], and NLMF represents the non-linear matrix factorisation technique (with linear and RBF versions) as proposed in [107].

Table 14: Comparing the NMAE (Normalized MAE) observed in different algorithms for the ML dataset. The proposed algorithms outperforms URP [101], Attitude [102], MatchBox [103], and MMMF [104]. They give the comparable results to Item [105], E-MMF [106], and NLMF [107]. Our results and the best results have been shown in bold.

Algorithm	NMAE
URP	0.4341 ± 0.0023
Attitude	0.4320 ± 0.0055
MatchBox	0.4206 ± 0.0055
MMMF	0.4156 ± 0.0037
$ItrSvd$	$\mathbf{0.4118 \pm 0.0025}$
Item	0.4096 ± 0.0029
E-MMF	0.4029 ± 0.0027
NLMF Linear	0.4052 ± 0.0011
NLMF RBF	$\mathbf{0.4026 \pm 0.0020}$

Table 14 shows that the NLMF, E-MMF, and Item perform better than the rest. The proposed hybrid algorithm gives comparable results to them with NMAE = 0.4118. It is worth mentioning that Item [105], E-MMF [106], and NLMF [107] employ extensive parameters learning, for instance the E-MMF is an ensemble of about 100 predictors, which makes this algorithms unattractive. From this table, we may conclude that the proposed algorithm is comparable to the state-of-the-art algorithm for the ML dataset.

9 When and how much imputation is required

As it is costly to do imputation by the proposed approaches, hence we investigate when it is beneficial to switch to the conventional approaches, which are cheap to compute. Next, we shed light on the following two questions: (1) when is imputation required? and (2) how much imputation is required?

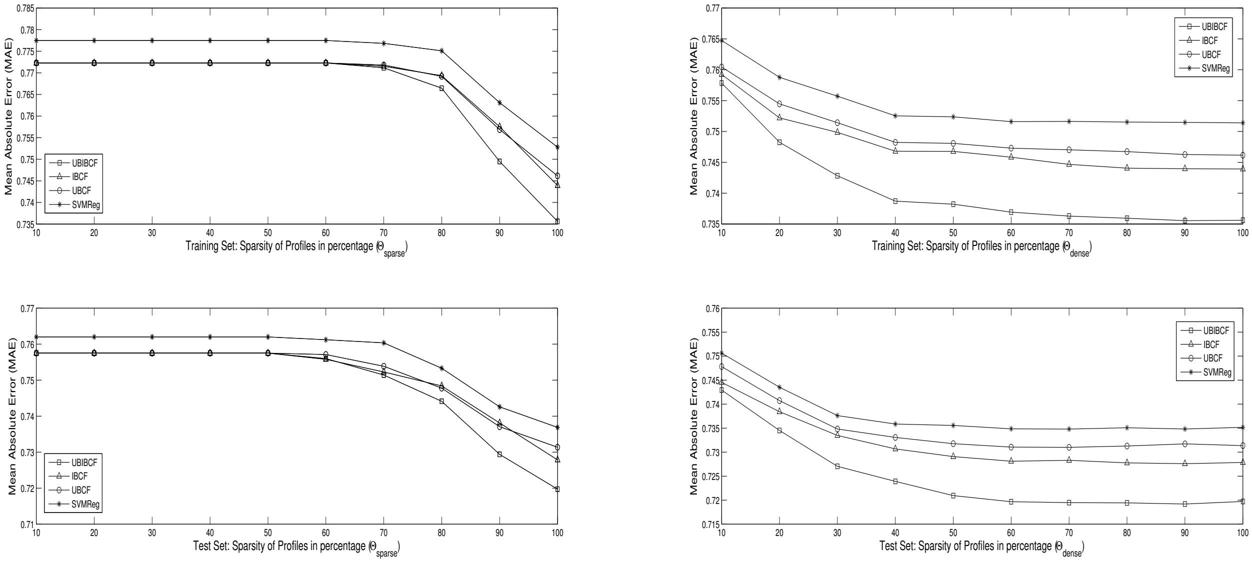
9.1 When to do imputation by the proposed approaches

To answer this question, we look into the sparsity of users' and items' profiles. We only do imputation by the proposed approaches when a user's (or item's) profile is $\Theta_{sparse}\%$ sparse, where $\Theta_{sparse} = \{10, 20, \cdots, 100\}$. A value of $\Theta_{Sparse} = 10$ shows that the proposed approaches are used to fill in the missing values if the sparsity of a profile is less than $10\% = 0.1$, and the UserItemAvg approach is used

[14] It might be due to the reason that, the author in [13] did not use any significance weighting schemes, and used weighted sum prediction formula [87] in the item-based CF.

[15] The results for the *weak generalisation* case were taken, as this procedure resembles with our test procedure.

[16] NMAE has been used in [102, 107], and is computed by normalizing the MAE by a factor. The value of the factor depends on the range of the ratings; for example for the MovieLens dataset, it is 1.6. For further information, refer to [107].



(a) When is imputation required?

(b) How much imputation is required?

Figure 13: Figures showing when and how much imputation is required for the SML dataset. Θ_{sparse} shows the sparsity of users' (or items') profiles in percentage. Θ_{dense} shows the percentage up to which users' (or items') profiles are filled using the proposed approaches. The optimal number of dimensions have been kept the same as shown in Table 4.

otherwise. Figure 13(a) shows that the MAE is minimum at $\Theta_{sparse} = 100$. For the subsequent experiments, we choose to do imputation when $\Theta_{sparse} = 100$.

9.2 How much imputation is required

Users' (or items') profiles can be filled up to $\Theta_{dense}\%$ of the missing values in their profiles. To investigate how much imputation is necessary, we performed experiments with different values of Θ_{dense} and observed the corresponding MAE. A value of $\Theta_{dense} = 10$ shows that 10% missing values of a profile are filled using the proposed approaches and UserItemAvg is used for the remaining 90% missing values. Figure 13(b) shows that after $\Theta_{desne} = 60$, the change in the MAE becomes very small. Hence, 60% imputation is sufficient to achieve good accuracy.

10 Discussion

What is evident from the experimental results is that the approximation of missing values in the sparse user-item rating matrix has an important role in SVD based recommendations. The literature proposes using item average to approximate the missing values in the sparse user-item rating matrix. We show that this is not a feasible solution in terms of accuracy. Moreover, the convergence is very slow in the case of conventional iterative SVD. Approaches such as CF or SVM should be used for this purpose.

We note that the imputation approaches based on the content-based filtering are not very accurate as compared to the collaborative filtering ones in the recommender system domain, though content-based filtering has successfully been applied to text categorisation and it gives accurate results as well. The reason is that the text categorisation and recommender system problems are quite different from each other. First, a user rates the same item differently under different context [108] and the reason of rating might be complex. Similarly, the positive feedback [109] given by a user, e.g. *purchased an item* is dependent on the context; for example, a user might purchase an item as a gift, hence we cannot predict that they will purchase other similar items. second, the user feedback [26] in a recommender system is noisy, the observations, *did not buy an item*, or *did not watch a movie* do not necessarily mean that the user is not interested in that item or movie. It can be the case, that user like that item or movie but has not purchased or watched it. Third, the evaluation criteria for both is different, recommender system usually provides a list of top items a user would like to consume, whereas text categorisation classify a given document to set of pre-defined categorisation. Furthermore, in text categorisation a document belongs to a single or a very few categories, whereas a user in recommender system might be interested in a large number of different items. Fourth, a user might change their taste over time and this temporal change in profile is not shared by the text categorisation tasks. Making accurate recommendation given the noisy input is different and more difficult as compared to the text categorisation task.

We captured the user profiles in terms of the important features, i.e. a user likes a certain types of movies such as horror, romantic etc. However, this assumption was not very correct as well. As most of the users love to watch movies that makes them simply feel good rather than strictly following the the same types of movies. Furthermore, taking into account the temporal property of datasets might improve the results.

It is worth noting that the SVM regression gives much better performance than other classification and regression approaches, though we have imbalanced dataset. Schemes dealing with the imbalanced data may increase the performance of SVM regression even further. In the case of SVM classification, we overcome this problem by assigning different penalties to classes according to the prior knowledge of users. The prior knowledge of a user is the fraction of the total number of ratings belonging to a class to the total number of ratings provided by the user in the training set. Over sampling and under sampling [64] can be used to check the performance of SVM regressions, which is a subject of future research.

Based on the experimental results, we can underline five interesting points: (1) the results of different approaches are dataset dependent and no approach is a panacea. Due to dataset characteristic—data distribution, scale, and sparsity—one approach might be very good for one dataset while might fail to produce good results for the second dataset, (2) collaborative filtering and SVM provide more accurate and much more computationally tractable results under all experiments: the iterative SVD, simple imputed SVD, CF applied over reduced dataset, and in SVD under sparse settings (3) although, conventional approaches are straight forward to implement, they do not provide good results. The same is true for many classification and regression approaches, (4) the hybrid recommender system algorithms provide more accurate recommendations than the individual ones. Different recommendation algorithms, if combined in a systemic way, have complementary role for recommendation generation, (5) different imputation schemes can be chosen depending the different circumstances and priorities—time and frequency of running the off-line computation, required accuracy, required recommendation time, and available resources (for example, the content features and memory).

11 Conclusion and future work

Recommender systems play an important role in identifying the interesting items for users and try to solve the problem of information overload. This paper makes significant contributions to the state-of-the-art in two areas of recommender systems, namely, SVD based recommendation algorithms, and the hybrid recommendation algorithms.

There has been some work, in the literature to overcome the scalability problem of recommender system using SVD; however due to sparsity it leads to poor quality recommendation. We show how both scalability and accuracy problems can be eliminated by using a suitable imputation source, as a pre-processing step, with SVD. We have shown by empirical study that rather than merely using the item average of user-item rating matrix as imputation, or ignoring the missing values, which have been the preferred approaches in the literature, flexible and robust imputation approaches gives considerable benefits ranging from cost saving to performance enhancement, and therefore, should be used prior to applying SVD over user-item rating matrix. We further show how the results of CF, when applied over the dataset reduced by SVD, change with the imputation source.

An important research issue in recommender system is that the recommendations should be tailored to the user's current information seeking task [110]. In this paper, we consider the two-dimensional Users × Items space, by recommending items to users based on the information only about users and items. It has been claimed that taking the additional context information (such as time, place, and the company of a user) into account, either by extending the user-item rating matrix into multiple dimensions or using reduction-based recommendation approach, might increase the performance of the recommender systems [111]. This multi-criteria ratings data would be very sparse as compared to the traditional user-item rating matrix, and drawing supplementary information from the content or demographic data of user/item might help reducing the sparsity of data matrix, which makes our imputation sources even more attractive in these scenarios. Keeping these promising results as starting point, we are focusing on the multi-dimensional dataset, where clustering algorithms can be used for partitioning the data and imputed SVD can be applied to reduce the sparsity and dimensional of the resulting partition.

Another avenue for future work would be to incorporate external sources of information, such as ontology of items, which may improve the results, particularly under sparse conditions. Furthermore, we would like to explore in detail the questions discussed in Section 9.

Acknowledgment

The work reported in this paper has formed part of the Instant Knowledge Research project which is jointly funded by Mobile VCE, (the Virtual Centre of Excellence in Mobile & Personal Communications, www.mobilevce.com), UK Technology Strategy Board (TSB), and EPSRC (Engineering and Physical Sciences Research Council).

12 Performance comparison of different imputation approaches

Tables 15 and 16 compare the performance (in terms of ROC-sensitivity and F1 measure respectively) of different approaches under the iterative SVD. We observe that

Table 15: Comparing the ROC observed in different imputation approaches under the iterative SVD (fixed dimension case). Only the conventional approaches and the approaches which gave the best results are compared. The best results have been shown in bold.

Imputation Source	**Best ROC-sensitivity**			
	ML	**SML**	**FT**1	**FT**5
ItemAvg	0.685 ± 0.003	0.651 ± 0.005	0.504 ± 0.012	0.569 ± 0.011
UserAvg	0.721 ± 0.002	0.683 ± 0.009	0.572 ± 0.011	0.571 ± 0.013
UBCF	0.724 ± 0.002	0.691 ± 0.005	0.546 ± 0.011	0.530 ± 0.012
IBCF	**0.759 ± 0.002**	**0.724 ± 0.012**	0.534 ± 0.011	0.544 ± 0.016
UBIBCF	0.747 ± 0.002	0.711 ± 0.004	0.517 ± 0.011	0.563 ± 0.009
SVMReg	−−	0.695 ± 0.007	**0.574 ± 0.013**	**0.583 ± 0.014**

Table 16: Comparing the Top-N F1 (computed over top-20 recommendations) observed in different imputation approaches under the iterative SVD (fixed dimension case). Only the conventional approaches and the approaches which gave the best results are compared. The best results have been shown in bold.

Imputation Source	**Best F**1			
	ML	**SML**	**FT**1	**FT**5
ItemAvg	0.445 ± 0.004	0.481 ± 0.005	0.486 ± 0.012	0.547 ± 0.008
UserAvg	0.463 ± 0.004	0.503 ± 0.007	0.540 ± 0.011	0.538 ± 0.014
UBCF	0.468 ± 0.004	0.514 ± 0.004	0.531 ± 0.012	0.520 ± 0.010
IBCF	**0.487 ± 0.004**	**0.531 ± 0.009**	0.505 ± 0.012	0.519 ± 0.012
UBIBCF	0.481 ± 0.004	0.528 ± 0.002	0.507 ± 0.012	0.534 ± 0.010
SVMReg	−−	0.508 ± 0.005	**0.556 ± 0.014**	**0.563 ± 0.014**

the proposed approaches give better results than traditional ones.

References

[1] P. Resnick and H. R. Varian, "Recommender systems," *Commun. ACM*, vol. 40, no. 3, pp. 56–58, 1997.

[2] B. Mobasher, "Recommender systems," *KI*, vol. 21, no. 3, pp. 41–43, 2007.

[3] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Computing*, vol. 7, pp. 76–80, January 2003.

[4] J. B. Schafer, J. Konstan, and J. Riedi, "Recommender systems in e-commerce," in *Proceedings of the 1st ACM conference on Electronic commerce*, ser. EC '99. New York, NY, USA: ACM, 1999, pp. 158–166. [Online]. Available: http://doi.acm.org/10.1145/336992.337035

[5] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry," *Commun. ACM*, vol. 35, pp. 61–70, December 1992. [Online]. Available: http://doi.acm.org/10.1145/138859.138867

[6] U. Shardanand and P. Maes, "Social information filtering: algorithms for automating word of mouth," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, ser. CHI '95. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1995, pp. 210–217. [Online]. Available: http://dx.doi.org/10.1145/223904.223931

[7] L. Terveen, W. Hill, B. Amento, D. McDonald, and J. Creter, "Phoaks: a system for sharing recommendations," *Commun. ACM*, vol. 40, pp. 59–62, March 1997. [Online]. Available: http://doi.acm.org/10.1145/245108.245122

[8] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: an open architecture for collaborative filtering of netnews," in *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, ser. CSCW '94. New York, NY, USA: ACM, 1994, pp. 175–186. [Online]. Available: http://doi.acm.org/10.1145/192844.192905

[9] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl, "Grouplens: applying collaborative filtering to usenet news," *Commun. ACM*, vol. 40, pp. 77–87, March 1997. [Online]. Available: http://doi.acm.org/10.1145/245108.245126

[10] D. M. Pennock, E. Horvitz, S. Lawrence, and C. L. Giles, "Collaborative filtering by personality diagnosis: A hybrid memory and model-based approach," in *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, ser. UAI '00.

San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, pp. 473–480.

[11] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, “Item-based collaborative filtering recommendation algorithms,” in *Proceedings of the 10th international conference on World Wide Web*, ser. WWW ’01. New York, NY, USA: ACM, 2001, pp. 285–295. [Online]. Available: http://doi.acm.org/10.1145/371920.372071

[12] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Application of dimensionality reduction in recommender system–a case study,” in *IN ACM WEBKDD WORKSHOP*. Citeseer, 2000.

[13] M. Vozalis and K. G. Margaritis, “Using svd and demographic data for the enhancement of generalized collaborative filtering,” *Information Sciences*, vol. 177, pp. 3017–3037, August 2007.

[14] M. Kurucz, A. Benczúr, and K. Csalogány, “Methods for large scale SVD with missing values,” in *Proceedings of KDD Cup and Workshop*. Citeseer, 2007.

[15] M. A. Ghazanfar and A. Prügel-Bennett, “The advantage of careful imputation sources in sparse data-environment of recommender systems: Generating improved svd-based recommendations,” in *IADIS European Conference on Data Mining*, July 2011.

[16] J. S. Breese, D. Heckerman, and C. Kadie, “Empirical analysis of predictive algorithms for collaborative filtering,” in *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, ser. UAI’98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 43–52.

[17] Y.-J. Park and A. Tuzhilin, “The long tail of recommender systems and how to leverage it,” in *Proceedings of the 2008 ACM conference on Recommender systems*, ser. RecSys ’08. New York, NY, USA: ACM, 2008, pp. 11–18. [Online]. Available: http://doi.acm.org/10.1145/1454008.1454012

[18] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering,” in *Proceedings of the Fifth International Conference on Computer and Information Technology*, 2002.

[19] G.-R. Xue, C. Lin, Q. Yang, W. Xi, H.-J. Zeng, Y. Yu, and Z. Chen, “Scalable collaborative filtering using cluster-based smoothing,” in *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, ser. SIGIR ’05. New York, NY, USA: ACM, 2005, pp. 114–121. [Online]. Available: http://doi.acm.org/10.1145/1076034.1076056

[20] A. M. Rashid, S. K. Lam, G. Karypis, and J. Riedl, “Clustknn: a highly scalable hybrid model-& memory-based cf algorithm,” in *Proc. of WebKDD 2006: KDD Workshop on Web Mining and Web Usage Analysis, in conjunction with the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006), August 20-23 2006, Philadelphia, PA*. Citeseer, 2006.

[21] F. Wang, T. Li, and C. Zhang, “Semi-supervised clustering via matrix factorization,” in *SDM*, 2008, pp. 1–12.

[22] M. A. Ghazanfar and A. Prügel-Bennett, “Fulfilling the needs of gray-sheep users in recommender systems, a clustering solution,” in *2011 International Conference on Information Systems and Computational Intelligence*, January 2011. [Online]. Available: http://eprints.ecs.soton.ac.uk/21770/

[23] M. A. Ghazanfar, S. Szedmák, and A. Prügel-Bennett, “Incremental kernel mapping algorithms for scalable recommender systems,” in *IEEE ICTAI*, 2011, pp. 1077–1084.

[24] K. Lang, “NewsWeeder: learning to filter netnews,” in *Proceedings of the 12th International Conference on Machine Learning*. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA, 1995, pp. 331–339.

[25] R. van Meteren and M. van Someren, “Using content-based filtering for recommendation,” in *Proceedings of the Machine Learning in the New Information Age: MLnet/ECML2000 Workshop*. Citeseer, 2000.

[26] M. J. Pazzani and D. Billsus, “The adaptive web,” P. Brusilovsky, A. Kobsa, and W. Nejdl, Eds. Berlin, Heidelberg: Springer-Verlag, 2007, ch. Content-based recommendation systems, pp. 325–341. [Online]. Available: http://dl.acm.org/citation.cfm?id=1768197.1768209

[27] S. Alag, *Collective Intelligence in Action*. Manning Publications, October, 2008.

[28] P. Melville, R. J. Mooney, and R. Nagarajan, “Content-boosted collaborative filtering for improved recommendations,” in *Eighteenth national conference on Artificial intelligence*. Menlo Park, CA, USA: American Association for Artificial Intelligence, 2002, pp. 187–192.

[29] R. Burke, “Hybrid recommender systems: Survey and experiments,” *User Modeling and User-Adapted Interaction*, vol. 12, no. 4, pp. 331–370, November 2002.

[30] M. J. Pazzani, “A framework for collaborative, content-based and demographic filtering,” *Artif. Intell. Rev.*, vol. 13, pp. 393–408, December 1999.

[Online]. Available: http://dx.doi.org/10.1023/A:1006544522159

[31] M. Claypool, A. Gokhale, T. Mir, P. Murnikov, D. Netes, and M. Sartin, "Combining content-based and collaborative filters in an online newspaper," in *In Proceedings of ACM SIGIR Workshop on Recommender Systems*. Berkeley, California: ACM, 1999.

[32] R. Burke, "Integrating knowledge-based and collaborative-filtering recommender systems," in *In AAAI Workshop on AI in Electronic Commerce*. AAAI, 1999, pp. 69–72.

[33] R. J. Mooney and L. Roy, "Content-based book recommending using learning for text categorization," in *Proceedings of the fifth ACM conference on Digital libraries*, ser. DL '00. New York, NY, USA: ACM, 2000, pp. 195–204. [Online]. Available: http://doi.acm.org/10.1145/336597.336662

[34] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. on Knowl. and Data Eng.*, vol. 17, pp. 734–749, June 2005. [Online]. Available: http://dx.doi.org/10.1109/TKDE.2005.99

[35] M. A. Ghazanfar and A. Prügel-Bennett, "An Improved Switching Hybrid Recommender System Using Naive Bayes Classifier and Collaborative Filtering," in *Lecture Notes in Engineering and Computer Science: Proceedings of The International Multi Conference of Engineers and Computer Scientists 2010*. IMECS 2010, 17–19 March, 2010, Hong Kong, 2010, pp. 493–502. [Online]. Available: http://eprints.ecs.soton.ac.uk/18483/

[36] ——, "A scalable, accurate hybrid recommender system," in *Proceedings of the 2010 Third International Conference on Knowledge Discovery and Data Mining*, ser. WKDD '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 94–98. [Online]. Available: http://dx.doi.org/10.1109/WKDD.2010.117

[37] M. A. Ghazanfar, S. Szedmak, and A. Prugel-Bennett, "Incremental kernel mapping algorithms for scalable recommender systems," in *Proceedings of the 2011 IEEE 23rd International Conference on Tools with Artificial Intelligence*, ser. ICTAI '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 1077–1084. [Online]. Available: http://dx.doi.org/10.1109/ICTAI.2011.183

[38] M. A. Ghazanfar, A. PrüGel-Bennett, and S. Szedmak, "Kernel-mapping recommender system algorithms," *Inf. Sci.*, vol. 208, pp. 81–104, Nov. 2012. [Online]. Available: http://dx.doi.org/10.1016/j.ins.2012.04.012

[39] D. Billsus and M. J. Pazzani, "Learning collaborative information filters," in *Proceedings of the Fifteenth International Conference on Machine Learning*, ser. ICML '98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 46–54.

[40] G. Takács, I. Pilászy, B. Németh, and D. Tikk, "Investigation of various matrix factorization methods for large recommender systems," in *Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition*, ser. NETFLIX '08. New York, NY, USA: ACM, 2008, pp. 6:1–6:8. [Online]. Available: http://doi.acm.org/10.1145/1722149.1722155

[41] D. Kim and B.-J. Yum, "Collaborative filtering based on iterative principal component analysis," *Expert Syst. Appl.*, vol. 28, pp. 823–830, May 2005. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2004.12.037

[42] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Incremental singular value decomposition algorithms for highly scalable recommender systems," in *Proceedings of the 5th International Conference in Computers and Information Technology*. Citeseer, 2002, pp. 27–28.

[43] R. J. A. Little and D. B. Rubin, "Statistical analysis with missing data," 1987.

[44] B. Marlin, R. Zemel, S. Roweis, and M. Slaney, "Collaborative filtering and the missing at random assumption," in *Uncertainty in Artificial Intelligence: Proceedings of the 23rd Conference (Submitted)*, vol. 47. Citeseer, 2007, pp. 50–54.

[45] M. G. Vozalis and K. G. Margaritis, "Applying svd on item-based filtering," in *Proceedings of the 5th International Conference on Intelligent Systems Design and Applications*, ser. ISDA '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 464–469. [Online]. Available: http://dx.doi.org/10.1109/ISDA.2005.25

[46] M. Vozalis and K. G. Margaritis, "Applying SVD on generalized item-based filtering," *International Journal of Computer Science and Applications*, vol. 3, no. 3, pp. 27–51, 2006.

[47] A. Martinez, J. Arias, A. Vilas, J. Garcia Duque, and M. Lopez Nores, "What's on tv tonight? an efficient and effective personalized recommender system of tv programs," *Consumer Electronics, IEEE Transactions on*, vol. 55, no. 1, pp. 286–294, 2009.

[48] A. B. Barragáns-Martínez, E. Costa-Montenegro, J. C. Burguillo, M. Rey-López, F. A. Mikic-Fonte, and A. Peleteiro, "A hybrid content-based and item-based collaborative filtering approach to

recommend tv programs enhanced with singular value decomposition," *Inf. Sci.*, vol. 180, pp. 4290–4311, November 2010. [Online]. Available: http://dx.doi.org/10.1016/j.ins.2010.07.024

[49] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: A constant time collaborative filtering algorithm," *Information Retrieval*, vol. 4, pp. 133–151, July 2001.

[50] Y. Azar, A. Fiat, A. Karlin, F. McSherry, and J. Saia, "Spectral analysis of data," in *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, ser. STOC '01. New York, NY, USA: ACM, 2001, pp. 619–626. [Online]. Available: http://doi.acm.org/10.1145/380752.380859

[51] C. Do and S. Batzoglou, "What is the expectation maximization algorithm?" *Nature biotechnology*, vol. 26, no. 8, pp. 897–899, 2008.

[52] J. Canny, "Collaborative filtering with privacy via factor analysis," in *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, ser. SIGIR '02. New York, NY, USA: ACM, 2002, pp. 238–245. [Online]. Available: http://doi.acm.org/10.1145/564376.564419

[53] N. Srebro and T. Jaakkola, "Weighted low-rank approximations," in *ICML*, vol. 20, no. 2, 2003, p. 720.

[54] S. Zhang, W. Wang, J. Ford, F. Makedon, and J. Pearlman, "Using singular value decomposition approximation for collaborative filtering," in *Proceedings of the Seventh IEEE International Conference on E-Commerce Technology*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 257–264.

[55] S. Zhang, W. Wang, J. Ford, and F. Makedon, "Learning from incomplete ratings using non-negative matrix factorization," in *6th SIAM Conference on Data Mining (SDM)*. Citeseer, 2006, pp. 548–552.

[56] J. Bennett and S. Lanning, "The netflix prize," in *Proceedings of KDD Cup and Workshop*, vol. 2007. Citeseer, 2007.

[57] N. Srebro, J. D. M. Rennie, and T. Jaakkola, "Maximum-margin matrix factorization," *Advances in neural information processing systems*, vol. 17, pp. 1329–1336, 2005.

[58] R. Bell, Y. Koren, and C. Volinsky, "The BellKor solution to the Netflix prize, in: AT&T Labs–Research: Technical report November," 2007.

[59] M. Wu, "Collaborative filtering via ensembles of matrix factorizations," in *Proceedings of KDD Cup and Workshop*. Citeseer, 2007.

[60] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," *Advances in Neural Information Processing Systems*, vol. 20, pp. 1257–1264, 2008.

[61] G. Takács, I. Pilászy, B. Németh, and D. Tikk, "Scalable collaborative filtering approaches for large recommender systems," *J. Mach. Learn. Res.*, vol. 10, pp. 623–656, June 2009.

[62] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '08. New York, NY, USA: ACM, 2008, pp. 426–434. [Online]. Available: http://doi.acm.org/10.1145/1401890.1401944

[63] R. M. Bell and Y. Koren, "Scalable collaborative filtering with jointly derived neighborhood interpolation weights," in *Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 43–52.

[64] I. H. W. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, October 1999.

[65] A. Paterek, "Improving regularized singular value decomposition for collaborative filtering," in *Proc. KDD Cup and Workshop*. Citeseer, 2007.

[66] M. Kurucz, A. Benczúr, T. Kiss, I. Nagy, A. Szabó, and B. Torma, "Who rated what: a combination of SVD, correlation and frequent sequence mining," in *Proc. KDD Cup and Workshop*, vol. 23. Citeseer, 2007, pp. 720–727.

[67] M. Piotte and M. Chabbert, "The pragmatic theory solution to the netflix grand prize, in: Netflix prize documentation," 2009.

[68] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Trans. Inf. Syst.*, vol. 22, pp. 5–53, January 2004. [Online]. Available: http://doi.acm.org/10.1145/963770.963772

[69] N. Good, J. B. Schafer, J. A. Konstan, A. Borchers, B. Sarwar, J. Herlocker, and J. Riedl, "Combining collaborative filtering with personal agents for better recommendations," in *Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence*, ser. AAAI '99/IAAI '99. Menlo Park, CA, USA: American Association for Artificial Intelligence, 1999, pp. 439–446.

[70] S.-T. Park, D. Pennock, O. Madani, N. Good, and D. DeCoste, "Naive filterbots for robust cold-start recommendations," ser. KDD '06. New York, NY, USA: ACM, 2006, pp. 699–705. [Online]. Available: http://doi.acm.org/10.1145/1150402.1150490

[71] H. Ma, I. King, and M. R. Lyu, "Effective missing data prediction for collaborative filtering," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, ser. SIGIR '07. New York, NY, USA: ACM, 2007, pp. 39–46. [Online]. Available: http://doi.acm.org/10.1145/1277741.1277751

[72] J. Zhang and P. Pu, "A recursive prediction algorithm for collaborative filtering recommender systems," in *Proceedings of the 2007 ACM conference on Recommender systems*, ser. RecSys '07. New York, NY, USA: ACM, 2007, pp. 57–64. [Online]. Available: http://doi.acm.org/10.1145/1297231.1297241

[73] X. Su, T. M. Khoshgoftaar, X. Zhu, and R. Greiner, "Imputation-boosted collaborative filtering using machine learning classifiers," in *Proceedings of the 2008 ACM symposium on Applied computing*, ser. SAC '08. New York, NY, USA: ACM, 2008, pp. 949–950. [Online]. Available: http://doi.acm.org/10.1145/1363686.1363903

[74] X. Su, T. M. Khoshgoftaar, and R. Greiner, "A mixture imputation-boosted collaborative filter," in *Proceedings of the 21th International Florida Artificial Intelligence Research Society Conference (FLAIRS'08)*, 2008, pp. 312–317.

[75] C. Ryan, D. Greene, G. Cagney, and P. Cunningham, "Missing value imputation for epistatic MAPs," *BMC Bioinformatics*, vol. 11, no. 1, pp. 197+, 2010.

[76] M. Balabanović and Y. Shoham, "Fab: content-based, collaborative recommendation," *Commun. ACM*, vol. 40, pp. 66–72, March 1997. [Online]. Available: http://doi.acm.org/10.1145/245108.245124

[77] T. Joachims, "A probabilistic analysis of the rocchio algorithm with tfidf for text categorization," in *Proceedings of the Fourteenth International Conference on Machine Learning*, ser. ICML '97. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997, pp. 143–151.

[78] B. M. Sarwar, J. A. Konstan, A. Borchers, J. Herlocker, B. Miller, and J. Riedl, "Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system," in *Proceedings of the 1998 ACM conference on Computer supported cooperative work*, ser. CSCW '98. New York, NY, USA: ACM, 1998, pp. 345–354. [Online]. Available: http://doi.acm.org/10.1145/289444.289509

[79] B. Mobasher, X. Jin, and Y. Zhou, "Semantically Enhanced Collaborative Filtering on the Web," vol. 3209, pp. 57–76, Sep. 2003.

[80] S. M. David, D. C. D. Roure, and N. R. Shadbolt, "Capturing knowledge of user preferences: Ontologies in recommender systems," in *In Proceedings of the First International Conference on Knowledge Capture (K-CAP 2001), Oct 2001.* ACM Press, pp. 100–107.

[81] M. Szomszor, C. Cattuto, H. Alani, K. O'äôhara, A. Baldassarri, V. Loreto, and V. D. P. Servedio, "Folksonomies, the semantic web, and movie recommendation," in *Bridging the Gep between Semantic Web and Web 2.0 (SemNet 2007)*, 2007, pp. 71–84.

[82] I. Cantador, A. Bellogín, and P. Castells, "A multilayer ontology-based hybrid recommendation model," *AI Commun.*, vol. 21, pp. 203–210, April 2008.

[83] S.-S. Weng and H.-L. Chang, "Using ontology network analysis for research document recommendation," *Expert Syst. Appl.*, vol. 34, pp. 1857–1869, April 2008.

[84] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Methods and metrics for cold-start recommendations," in *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, ser. SIGIR '02. New York, NY, USA: ACM, 2002, pp. 253–260. [Online]. Available: http://doi.acm.org/10.1145/564376.564421

[85] M. W. Berry, S. T. Dumais, and G. W. O'Brien, "Using linear algebra for intelligent information retrieval," *SIAM Rev.*, vol. 37, pp. 573–595, December 1995.

[86] D. Scott C., D. Susan T., L. Thomas K., F. George W., and H. Richard A., "Indexing by latent semantic analysis," *Journal of the American society for information science*, vol. 41, no. 6, pp. 391–407, 1990.

[87] M. A. Ghazanfar and A. Prügel-Bennett, "Novel significance weighting schemes for collaborative filtering: Generating improved recommendations in sparse environments." in *DMIN*. CSREA Press, 2010, pp. 334–342.

[88] J. Herlocker, J. A. Konstan, and J. Riedl, "An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms," *Inf. Retr.*, vol. 5, pp. 287–310, October 2002.

[89] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, pp. 27:1–27:27, May 2011. [Online]. Available: http://doi.acm.org/10.1145/1961189.1961199

[90] C. Hsu, C. Chang, C. Lin *et al.*, "A practical guide to support vector classification," 2003.

[91] A. Ramanan, S. Suppharangsan, and M. Niranjan, "Unbalanced decision trees for multi-class classification," in *IEEE - Second International Conference on Industrial and Information Systems, ICIIS 2007*. IEEE, August 2007, pp. 291–294. [Online]. Available: http://eprints.ecs.soton.ac.uk/21490/

[92] M. A. Ghazanfar and A. Prügel-Bennett, "Building Switching Hybrid Recommender System Using Machine Learning Classifiers and Collaborative Filtering," *IAENG International Journal of Computer Science*, vol. 37, no. 3, pp. 272–287, 2010.

[93] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *Proceedings of the 10th European Conference on Machine Learning*. London, UK: Springer-Verlag, 1998, pp. 137–142.

[94] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *SIGKDD Explor. Newsl.*, vol. 11, pp. 10–18, November 2009. [Online]. Available: http://doi.acm.org/10.1145/1656274.1656278

[95] M. Vozalis and K. G. Margaritis, "On the enhancement of collaborative filtering by demographic data," *Web Intelli. and Agent Sys.*, vol. 4, pp. 117–138, April 2006.

[96] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning, "Kea: practical automatic keyphrase extraction," in *Proceedings of the fourth ACM conference on Digital libraries*, ser. DL '99. New York, NY, USA: ACM, 1999, pp. 254–255. [Online]. Available: http://doi.acm.org/10.1145/313238.313437

[97] K. Aas and L. Eikvil, "Text categorisation: A survey." 1999.

[98] T. Zhang and V. S. Iyengar, "Recommender systems using linear classifiers," *J. Mach. Learn. Res.*, vol. 2, pp. 313–334, March 2002.

[99] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Analysis of recommendation algorithms for e-commerce," in *Proceedings of the 2nd ACM conference on Electronic commerce*, ser. EC '00. New York, NY, USA: ACM, 2000, pp. 158–167. [Online]. Available: http://doi.acm.org/10.1145/352871.352887

[100] M. A. Ghazanfar and A. Prügel-Bennett, "Novel heuristics for coalition structure generation in multi-agent systems," in *The 2010 International Conference of Computational Intelligence and Intelligent Systems*. ICCIIS'10, 30 June–2 July 2010, London, U.K., 2010. [Online]. Available: http://eprints.ecs.soton.ac.uk/18788/

[101] B. Marlin, "Modeling user rating profiles for collaborative filtering," *Advances in neural information processing systems*, vol. 16, pp. 627–634, 2004.

[102] ——, "Collaborative Filtering: A Machine Learning Perspective," Master's thesis, University of Toronto, 2004.

[103] D. H. Stern, R. Herbrich, and T. Graepel, "Matchbox: large scale online bayesian recommendations," in *Proceedings of the 18th international conference on World wide web*, ser. WWW '09. New York, NY, USA: ACM, 2009, pp. 111–120. [Online]. Available: http://doi.acm.org/10.1145/1526709.1526725

[104] J. D. M. Rennie and N. Srebro, "Fast maximum margin matrix factorization for collaborative prediction," in *Proceedings of the 22nd international conference on Machine learning*, ser. ICML '05. New York, NY, USA: ACM, 2005, pp. 713–719. [Online]. Available: http://doi.acm.org/10.1145/1102351.1102441

[105] S. Park and D. Pennock, "Applying collaborative filtering techniques to movie search for better ranking and browsing," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2007, pp. 550–559.

[106] D. DeCoste, "Collaborative prediction using ensembles of maximum margin matrix factorizations," in *Proceedings of the 23rd international conference on Machine learning*, ser. ICML '06. New York, NY, USA: ACM, 2006, pp. 249–256. [Online]. Available: http://doi.acm.org/10.1145/1143844.1143876

[107] N. D. Lawrence and R. Urtasun, "Non-linear matrix factorization with gaussian processes," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. New York, NY, USA: ACM, 2009, pp. 601–608. [Online]. Available: http://doi.acm.org/10.1145/1553374.1553452

[108] L. Baltrunas, "Exploiting contextual information in recommender systems," in *Proceedings of the 2008 ACM conference on Recommender systems*, ser. RecSys '08. New York, NY, USA: ACM, 2008, pp. 295–298. [Online]. Available: http://doi.acm.org/10.1145/1454008.1454056

[109] D. Oard and J. Kim, "Implicit feedback for recommender systems," in *Proceedings of the AAAI Workshop on Recommender Systems*, 1998, pp. 81–83.

[110] S. M. McNee, "Meeting user information needs in recommender systems," Ph.D. dissertation, UNIVERSITY OF MINNESOTA, USA, 2006.

[111] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin, "Incorporating contextual information in recommender systems using a multidimensional approach," *ACM Trans. Inf. Syst.*, vol. 23, pp. 103–145, January 2005. [Online]. Available: http://doi.acm.org/10.1145/1055709.1055714

Network Anomaly Identification using Supervised Classifier

Prasanta Gogoi, B. Borah and D. K. Bhattacharyya
Department of Computer Science and Engineering
Tezpur University, Napam, Tezpur-784028, India
{prasant,bgb, dkb}@tezu.ernet.in and www.tezu.ernet.in

Keywords: supervise, incremental, clustering, anomaly, DoS, classification, PCC

Received: July 31, 2012

In this paper we present a clustering based classification method and apply it in network anomaly detection. A set of labeled training data consisting of normal and attack instances are divided into clusters which are represented by their representative profiles consisting of attribute-value pairs for selected subset of attributes. Each category of attack and normal instances are broken down into a set of clusters using a training algorithm based on supervised classification algorithm. The cluster profiles together with their class label form rules for labeling unseen test instances. Methods for clustering, training and prediction are provided. The proposed method is evaluated using real life TUIDS Intrusion datasets. Evaluation results on KDD 1999 datasets showed good performance in comparison to results produced by decision tree counterparts. The method presented can be utilized for classification jobs in any other domain.

Povzetek: Predstavljena je metoda za identifikacijo anomalij v omrežju.

1 Introduction

A network intrusion can be any exploit of a network that compromises its stability or the security of information stored on computers connected to it. An intrusion detection system gathers relevant data from computers or the network and analyzes them for signs of intrusion. Different methods such as statistical, pattern matching, machine learning, and data mining are used for intrusion detection. Some approaches are online, that is, they detect attacks in progress in real time, while offline approaches such as data mining provide after-the-fact clues about the attacks to help reduce the possibilities of future attacks of the same type. In general there are two types of approaches [1] for network intrusion detection: misuse detection and anomaly detection. Misuse detection searches for specific signatures to match, signaling previously known attacks without generating a large number of false alarms. Such methods fail to detect new types of attacks as their signatures are not known. Anomaly detection builds models for normal behavior and significant deviations from it are flagged as attack. Supervised or unsupervised learning can be used for anomaly detection. In a supervised approach, the model is developed based on labeled training data. Unsupervised approaches work without any training data or they may use training with unlabeled data. The main advantage of anomaly detection is that it can detect previously unknown attacks if their behavior is significantly different from what is considered to be normal. False alarm rates tend to be higher for anomaly detection methods.

Data mining methods including classification, association analysis, clustering and outlier detection are being used in network intrusion detection. Anomaly detection often tries to cluster test datasets into groups of similar instances which may be either attacks or normal data. Intrusion detection problem is then reduced to the problem of labeling the clusters as intrusive or normal traffic. When doing unsupervised anomaly detection a model based on clusters of data is trained using unlabeled data, normal as well as attacks. Supervised anomaly detection methods such as classification algorithms need to be presented with both normal and known attack data for training.

In this paper we present a supervised classification method and use it for network anomaly detection. Labeled training dataset is broken down into clusters belonging to normal and attack categories and the clusters are represented by their representative profiles, which together with category labels form the classification rules. During detection, the cluster profiles collect similar testing instances which are then labeled as belonging to the same category as the label of the profiles.

Rest of the paper is organized as follows. In Section 2, some related work regarding supervised anomaly detection methods are presented. The proposed classification algorithm including a training method and a prediction method is presented in Section 3. Application of the method to network anomaly detection is presented in Section 4. Experimental results for datasets TUIDS intrusion, KDD 1999, and NSL-KDD datasets are reported in Section 5. Finally, Section 6 concludes providing a few possible future extensions.

2 Related work

Classification is an important supervised learning method that has been applied to anomaly detection. Lee and Stolfo [1] proposed a systematic framework employing data mining methods for intrusion detection. This framework consists of classification, association rules and frequent episodes algorithms that can be used to construct detection models. The authors in [2] presented PNrule, for multi-class classification problem. The key idea used in PNrule is learning a rule-based model in two stages: first find P-rules to predict presence of a class and then find N-rules to predict absence of a class. The scoring mechanism used in PNrule allows one to tune selectively the effect of each N-rule on a given P-rule. ADWICE [3] uses extended BIRCH [4] clustering algorithm to implement a fast, scalable and adaptive anomaly detection scheme. They apply clustering as a method for training of the normality model. Several soft computing paradigms, viz., fuzzy rule-based classifiers, support vector machines, linear genetic programming and an ensemble method to model fast and efficient intrusion detection systems were investigated in [5]. Empirical results clearly show that soft computing approach could play a major role for intrusion detection. Yang et al. [6] present an anomaly detection approach based on clustering and classification for intrusion detection. They perform clustering to group training data points into clusters, from which they select some clusters as normal and create known-attack profiles according to certain criteria. The training data excluded from the profile are used to build a specific classifier. During the testing stage, they use an influence-based classification algorithm to classify network behaviors. A comparative study of several supervised probabilistic and predictive machine learning methods for intrusion detection is reported in [7]. Two probabilistic methods, Naive Bayes and Gaussian and two predictive methods, Decision Tree [8] and Random Forests [9], are used. The ability of each method to detect four attack categories (DoS, Probe, R2L and U2R) has been compared. A new supervised intrusion detection method based on TCM-KNN (Transductive Confidence Machine for K-Nearest Neighbors) algorithm has been presented in [10]. This algorithm works well even if sufficient attack data are not available for training. A review of the most well known anomaly-based intrusion detection methods is provided by [11, 12, 13]. Available platforms, systems under development and research projects in the area are also presented. This paper also outlines the main challenges to be dealt with for the wide scale deployment of anomaly-based intrusion detectors, with special emphasis on assessment issues. Beghdad [14] presents a study of the use of some supervised learning methods to predict intrusions. The performances of six machine learning algorithms involving C4.5, ID3, Classification and Regression Tree (CART), Multinomial Logistic Regression (MLR), Bayesian Networks (BN), and CN2 rule-based algorithm are investigated. KDD 1999 datasets were used to evaluate the considered algorithms. An SVM-based intrusion detection system is found in [15], one which combines a hierarchical clustering algorithm (BIRCH [4]), a simple feature selection procedure, and the SVM (support vector machines) method. This method is also evaluated using on the KDD 1999 datasets. For these evaluations, three cases were considered: the whole attacks case, the five behaviors classes case, and the two behaviors classes case. The performances of each method were compared.

3 Proposed supervised method

The proposed supervised method uses a training algorithm to create a set of representative clusters from the available labeled training objects. Unlabeled test objects are then inserted in these representative clusters based on similarity calculations and thus get labels of the clusters in which they are inserted. We first present the basics of the clustering algorithm, and then present the training and prediction algorithms.

3.1 Background

The dataset to be clustered contains n objects, each described by d attributes $A_1,\ A_2,\ \cdots,\ A_d$ having finite discrete valued domains $D_1,\ D_2,\ \cdots,\ D_d$, respectively. A data object can be represented as $X = \{x_1,\ x_2,\ \cdots,\ x_d\}$. The j-th component of object X is x_j and it takes one of the possible values defined in domain D_j of attribute A_j. Referring to each object by its serial number, the dataset can be represented by the set $N = \{1, 2, \cdots, n\}$. Similarly, the attributes are represented by the set $M = \{1,\ 2,\ \cdots,\ d\}$.

3.1.1 Similarity function between two objects

The similarity between two data objects X and Y is the sum of per attribute similarity for all the attributes. It is computed as $sim(X, Y)$,

$$sim(X,\ Y) = \sum_{j=1}^{d} s(x_j,\ y_j), \tag{1}$$

where $s(x_j, y_j)$ is the similarity of the j-th attribute defined as

$$s(x_j,\ y_j) = \begin{cases} 1 & \text{if } |x_j - y_j| \le \delta_j \\ 0 & \text{otherwise ,} \end{cases} \tag{2}$$

where δ_j is the similarity threshold for the j-th attribute. For categorical attributes δ_j=0 and for numeric attributes $\delta_j \ge 0$.

3.1.2 Similarity between cluster and object

A subspace based incremental clustering method is used here. A cluster is a set of objects which are similar considering a subset of attributes only. The minimum size of the subset of attributes required to form a cluster is defined by the threshold $MinAtt$. Let the subset of defining attributes be represented by $Dattributes = \{a_1,\ a_2,\ \cdots,\ a_{noAttributes}\}$ such that $Dattributes \subseteq M$ and $noAttributes$ is the size of $Dattributes$. A cluster is represented by its profile, somewhat like an object. All objects in a cluster are similar to the profile. The cluster profile is defined by a set of values, $Values = \{v_1\ , v_2,\ \cdots,\ v_{noAttributes}\}$ considering the attributes in $Dattributes$. That is $v_1 \in D_{a_1}$ is the value for attribute $a_1 \in M$, $v_2 \in D_{a_2}$ is the value for attribute $a_2 \in M$, and so on. Thus, the cluster profile is defined by

$$Profile = \{noAttributes,\ Dattributes,\ Values\}. \quad (3)$$

Let $Olist \subseteq N$ be the list of data objects in the cluster. A cluster C is completely defined by its $Profile$ and $Olist$

$$C = \{Olist,\ Profile\}. \quad (4)$$

Our incremental clustering algorithm inserts an object in any one of the clusters existing at a particular moment. So, the similarity between a cluster and a data object needs to be computed. Obviously, the cluster profile is used for computing this similarity. As the similarity is computed over the set of attributes in $Dattributes$ only, the similarity function between a cluster C and an object X becomes $sim(C, X)$

$$sim(C,\ X) = \sum_{j=1}^{noAttributes} s(v_j,\ x_{a_j}), \quad (5)$$

where $s(v_j,\ x_{a_j})$ is the similarity of the j-th attribute defined as

$$s(v_j,\ x_{a_j}) = \begin{cases} 1 & \text{if } |v_j - x_{a_j}| \leq \delta_{a_j} \\ 0 & \text{otherwise .} \end{cases} \quad (6)$$

Example: Consider a small dataset shown in Table 1 with seven objects defined using five attributes A_1, A_2, A_3, A_4 and A_5. The domains for the attributes are respectively,
$D_1 = \{a1,\ a2,\ a3\}$, $D_2 = \{b1,\ b2\}$,
$D_3 = \{c1, c2, c4\}$, $D_4 = \{d1, d2, d3\}$ and $D_5 = \{e1, e2\}$.

Clusters C_1 and C_2 can be identified in the dataset with $MinAtt = 3$ and $\delta_{a_j} = 0$ for $a_j \in Dattributes$:
$C_1 = \{Olist = \{1,\ 2,\ 4\},\ noAttributes = 3,$
$Dattributes = \{2,\ 3,\ 5\},\ Values = \{b2,\ c4,\ e2\}\}$, and
$C_2 = \{Olist = \{3,\ 5,\ 7\},\ noAttributes = 4,$
$Dattributes = \{1,\ 2,\ 3,\ 5\}, Values = \{a3,\ b1,\ c2,\ e1\}\}$.

3.1.3 Our supervised clustering algorithm

The clustering algorithm starts with an initially empty set of clusters. It reads each object X_i sequentially, inserts it in an existing cluster based upon the similarity between X_i and the clusters or a new cluster is created with X_i if it is not similar enough, as defined by the threshold $MinAtt$ for insertion in an existing cluster. The search for a cluster for inserting the present object is started with the last cluster created and moves towards the first cluster until the search is successful. If successful, the object is inserted in the cluster found and the search is terminated. At the time of inserting the object in the found cluster C, the number of defining attributes of the cluster ($C.noAttributes$) is set according to the computed similarity measure between the cluster and the object and the sets of $C.Dattributes$ along with $C.Values$ are updated. If the search is not successful a new cluster is created and the object itself made the representative object of the cluster, i.e., the full set of attributes becomes the $Dattributes$ while full set of values of the object becomes corresponding $Values$ of the new cluster profile.

Table 1: A sample dataset

$Serial\ no.$	A_1	A_2	A_3	A_4	A_5
1	a3	b2	c4	d1	e2
2	a2	b2	c4	d3	e2
3	a3	b1	c2	d1	e1
4	a2	b2	c4	d1	e2
5	a3	b1	c2	d3	e1
6	a1	b2	c1	d2	e2
7	a3	b1	c2	d2	e1

Table 2: Algorithm Notations

$Symbol$	$Description$
$MinAtt$	threshold value for minimum attribute.
$minSize$	minimum objects in a cluster threshold value.
$sim(X, Y)$	similarity value between two objects.
$sim(C, X)$	similarity value between a cluster and an object.
$Olist$	object list in a cluster.
$Profile$	profile of a cluster.
$noAttribtes$	total number of attributes in a cluster profile.
$Dattributes$	distribution of attributes in a cluster profile.
$Values$	values of attributes in a cluster profile.

The classification method begins with a fixed number of existing clusters (as defined by their profiles) with no objects present in them. Objects are incrementally inserted in any of the clusters and no new clusters are created. The description of notations used in the algorithm is given in *Table 2*

3.2 Training method

Given a set of labeled training data, a combination of unsupervised incremental clustering and supervised classification methods are applied to create and refine a set of clusters from which profiles are extracted for use in the test object labeling process. The unsupervised clustering algorithm decides cluster membership immediately as the objects arrive sequentially without considering subsequent objects that are yet to be seen. Therefore, refinement of the created clusters is performed using a subsequent supervised classification step that allow possible realignment of the data objects in the clusters.

In the beginning of the training algorithm, all objects are marked unprocessed. Similarity thresholds $minAtt$ (the minimum number of attributes) and $minSize$ (the minimum number of objects in a cluster) are set high and they are gradually decreased in steps. In each iteration, the remaining unprocessed objects are clustered using a combination of supervised classification and unsupervised clustering. If the supervised classification process fails to insert an object in any of the preexisting (created in the previous iteration) clusters (SCs), then the unsupervised clustering process inserts it in any of the unsupervised clusters (UCs) created in the present iteration or a new unsupervised cluster is created with the object. When the clustering process ends in the present iteration, cluster profiles are extracted from each of the SCs with at least $minSize$ objects in it and the objects in such a cluster are marked processed. All SCs are then deleted. The UCs may also include some insignificant clusters whose sizes are less than $minSize$. Such clusters are deleted. Remaining UCs are made SCs for next iteration after making them empty by deleting their object lists. Then the threshold values $minSize$ and $minAtt$ are reduced so that the next iteration can create larger clusters instead of fragmented clusters. Reducing the thresholds allows more generalization. The algorithm iterates so long as there are unprocessed objects. To ensure termination of the algorithm $minSize$ is reduced to $minSize$/2 so that the ultimate value of $minSize$ becomes 1, beyond which no objects remain unprocessed. The threshold $minAtt$ is loosened by setting $minAtt$=$minAtt$ $-$ α, where α is a small integral constant such as 1 or 2. Reduction of $minAtt$ below a certain level (MIN) is not allowed, after which remains constant at MIN. Generalization beyond MIN makes data objects from two different classes indistinguishable. When training terminates, the set of profiles found in the profile file becomes the final cluster profiles for use in the prediction process. The training algorithm is given as Algorithm 2.

Algorithm 1 Training algorithm

```
Input: Training Dataset;
Output: $file1$=TrainProfile;
read ($n$, $d$, $minAtt$, $minSize$) ;
for ($i = 0; i < n; i = i + 1$) do
    read ($X[i]$, $recordLabel[i]$) ;
    processed[i]=0 ;
end for
$totalProfile = 0$ ;
$top = 0$ ;
while ($minSize > 0$) do
    for ($i = 0; i < n; i = i + 1$) do
        $found = FALSE$; //supervised clustering
        for ($j = 0; j < totTrain; j = j + 1$) do
            if ($recordLabel[i] == clusterLabel[j]$) then
                if ($sim(SC[j],\ X[i]) == SC[j].noAttributes$) then
                    $append(i,\ SC[j].oList)$ ;
                    $found = TRUE$ ;
                    break ;
                end if
            end if
        end for //end supervised clustering
        if ($found == TRUE$) then
            continue;
        end if
        for ($j = top; j >= 0; j = j - 1$) do
            if ($recordLabel[i] == clusterLabel[j]$) then
                if ($sim(UC[j],\ X[i]) >= minAtt$) then
                    $found = TRUE$ ;
                    $update(UC[j],\ i,\ X[i])$ ;
                    $break$ ;
                end if
            end if
        end for
        if ($found == FALSE$) then
            $top = top + 1$ ; //create new cluster
            $UC[top]$ =new $cluster$ ;
            $append(i,\ UC[top].oList)$ ;
            $UC[top].noAttributes = d$ ;
            for ($j = 0; j < d; + + j$) do
                $UC[top].a[j] = j$ ;
                $UC[top].v[j] = X[i][j]$ ;
            end for
            $clusterLabel[top] = recordLabel[i]$ ;
        end if
    end for//end new cluster
    for ($i = 0; i < totTrain; i = i + 1$) do
        $m$=sizeof($SC[i].oList$)
        if ($m >= minSize$) then
            $k = SC[i].noAttributes$ ;
            for ($j = 0; j < k; j = j + 1$) do
                write($file1$, $SC[i].a[j]$) ;
            end for
            for ($j = 0; j < k; j = j + 1$) do
                write($file1$, $SC[i].v[j]$) ;
            end for
            $p = SC[i].oList$ ;
            for ($j = 0; j < m; j = j + 1$) do
                $k = p.getdata(j)$ ;
                processed[k]=1 ;
                p=p.next;
            end for
            delete SC[i] ;
        end if
        $totTrain = 0$ ;
        for ($i = 0; i < top; i = i + 1$) do
            $m$=sizeof($UC[i].oList$)
            if ($m >= minSize$) then
                $SC[totTrain] = UC.[i]$ ;
                delete $UC[i].oList$ ;
                $UC[i].oList = NULL$ ;
```

3.2.1 Effective profile searching

Two-dimensional link list (2-DLL) structure (as found in [16]) is used to store the profiles in memory. It enables to update the profile base dynamically. The 2-DLL structure updates row wise with insertion of new parameter in a profile and column wise with the insertion of a new protocol type. It enables to search the profile base protocol wise, viz., TCP, ICMP, UDP, "other" (which profiles are not included the protocols viz., TCP, ICMP or UDP. Once the respective protocol is identified in this 2-DLL structure, further traversal is supported for other parameters matching by following the link list structure.

3.3 Prediction method

Once the set of cluster profiles is ready, labeling test objects becomes simple. Supervised clustering is performed with the objects. The label of the cluster profile becomes the label of each object inserted in the cluster. The objects

2 Training algorithm (continued)

```
                $totTrain = totTrain + 1$ ;
            else
                delete UC[i] ;
            end if
        end for
        $minSize = minSize/2$ ;
        $MinAtt = minAtt - \alpha$ ;
        if ($MinAtt < MIN$) then
            $minAtt = MIN$ ;
        end if
    end for
end while
int function sim($cluster\ C,\ object\ X$)
int $scount = 0$ ;
$k = C.noAttributes$ ;
for ($j = 0; j < k; j = j + 1$) do
    $l = C.a[j]$ ;
    if ($abs(C.v[j] - x[l]) <= \delta[l]$) then
        $scount = scount + 1$ ;
    end if
end for
return $scount$ ;
end function
function $update(cluster\ C,\ int\ r,\ object\ X)$
$append(r,\ C.Olist)$ ;
int $count = 1$;
$m = C.noAttributes$;
for ($j = 0; j < m; + + j$) do
    $l = C.a[j]$ ;
    if (abs($s(C.v[j] - x[l])$) $<= \delta[l]$) then
        $count = count + 1$ ;
        $C.v[count] = C.v[j]$ ;
        $C.a[count] = C.a[j]$ ;
    end if
    $C.noAttributes = count$ ;
end for
end function
```

Algorithm 3 Prediction algorithm

```
Input: Dataset $file1$=TrainProfile, $file2$=DataFile;
Output: Object cluster label of $file2$ ;
read($file1$, $totTrain$);
for $i = 0; i < totTrain; i = i + 1$ do
    read($file1$, $k$);
    $C[i].noAttributes = k$;
    for ($q = 0; q < k; q = q + 1$) do
        read($file1$, $C[i].a[q]$);
        for ($j = 0; j < k; j = j + 1$) do
            read($file1$, $C[i].v[j]$);
            if $count == C[j].noAttributes$ then
                print(“Object" $i$ “ is of category"
                $clusterLabel[j]$);
                $found = 1$; break;
            else
                if $count > max$ then
                    $max = count; position = j$;
                end if
            end if
        end for
        if $found == 0$ then
            print(“Object" $i$ “is of category";
            $clusterLabel[position]$);
        end if
    end for
end for
```

not inserted in any of the clusters defined by the profiles need special attention. They may be flagged suspicious for anomalies. Another method for dealing with such objects is to insert such an object in a cluster which is most similar to the object and label accordingly. The algorithm for prediction method is given as Algorithm 4.

3.3.1 Complexity analysis

The clustering algorithms require one pass through the set of training examples that currently remain unprocessed. Each training example needs to be compared with existing clusters one after another until it gets inserted in one of the clusters. The similarity computation involves a subset of attributes. Therefore, the clustering process has a complexity $O(ncd)$, where n is the number of training examples, c is the number of clusters and d is the number of attributes. Each of the created clusters needs to be visited to extract its size and profile. Hence, maximum time complexity of one iteration of the training algorithm becomes $O(ncd) + O(c)$. The algorithm performs at most k iterations, where $k = log_2(minSize)$. As $minSize$ is the minimum number of objects for a cluster to be considered significant, it is not large. Overall maximum time complexity of the algorithm is $O(kncd) + O(kc)$.

4 Classification of network anomalies

In this section we apply our classification method to network anomaly detection.

4.1 Dataset Description

The algorithm was evaluated with three real life TUIDS [17] intrusion datasets and two benchmark intrusion datasets, viz., KDD Cup 1999 [18] and NSL-KDD [19] datasets.

4.1.1 TUIDS dataset

The real life TUIDS Intrusion datasets consist of three datasets *Packet level*, *Flow level* and *Portscan* where each dataset consists of attributes (numerical/categorical) of 50(11/39), 24(9/15) and 23(9/14), respectively as given in Table 3.

4.1.2 Benchmark intrusion datasets

The description of two benchmark intrusion datasets, KDD Cup 1999 [18] and NSL-KDD [19] are given in Table 4 and Table 5, respectively. Each record of the datasets represents a connection between two network hosts according to some well defined network protocol and is described by 41 attributes (38 continuous or discrete numerical attributes and 3 categorical attributes). Each record is labeled as either normal or one specific kind of attack. The attacks fall

Table 3: TUIDS Intrusion Datasets

Data sets	*Connection Type*	*Data set type*	
		Training	*Testing*
Packet Level	**Normal**	35043	27895
	Attack	397832	138370
	Total	432875	166265
Flow Level	**Normal**	36402	16770
	Attack	363729	123955
	Total	400131	140725
Portscan	**Normal**	2445	1300
	Attack	39215	28615
	Total	41660	29915

Table 4: KDD Cup 1999 datasets

Data sets	*DoS*	*U2R*	*R2L*	*Probe*	*Normal*	*Total*
Corrected KDD	229853	70	16347	4166	60593	311029
10-percent KDD	391458	52	1126	4107	97278	494021

Table 5: NSL-KDD datasets

Data sets	*DoS*	*U2R*	*R2L*	*Probe*	*Normal*	*Total*
$KDDTrain^+$	45927	52	995	11656	67343	125973
$KDDTest^+$	7458	67	2887	2422	9710	22544

in one of the four categories: User to Root ($U2R$), Remote to local ($R2L$), Denial of Service (DoS) and $Probe$.

- Denial of Service(DoS): Attacker tries to prevent legitimate users from using a service. For example, SYN flood, smurf, teardrop etc.
- Remote to Local ($R2L$): Attackers try to gain access to victim machine without having an account on it. For example, guessing password.
- User to Root ($U2R$): Attackers have local access to the victim machine and tries to gain super user privilege. For example, buffer overflow attacks.
- $Probe$: Attacker tries to gain information about the target host. For example, Port-scan, ping-sweep etc.

Number of samples of each category of attack in $Corrected\ KDD$ dataset and 10 $percent\ KDD$ dataset of KDD 1999 are shown in Table 4. The attack distribution in $KDDTrain^+$ and $KDDTest^+$ of NSL-KDD are shown in Table 5.

4.2 Data preprocessing

We have discretized continuous valued attributes by taking logarithm to the base 2 and then converting to integer. This is done for each attribute value z using the computation: $if\ (z > 2)\ z = int(log_2(z)) + 1$. Before taking logarithm, the attributes which take fractional values in the range [0, 1] are multiplied by 100 so that they take values in the range [0, 100]. Nominal valued attributes are mapped to discrete numeric codes which are nothing but serial numbers beginning with zero for the unique attribute values in the order in which they appear in the dataset. The class label attribute is removed from the dataset and stored separately in a different file. The class labels are used for training and evaluating the detection performance of the algorithm.

4.3 Feature selection

Information gain [20] is computed for each of the discretized attributes. Six attributes (attribute numbers 7, 9, 15, 18, 20 and 21 in KDD Cup 1999 and NSL-KDD datasets) corresponding to very low information gain are removed from the dataset. It reduces computation time.

4.4 Parameter selection

The training algorithm has 3 parameters to be given as inputs: $minAtt$, $minSize$ and MIN. The parameters $minAtt$ takes values in the range [1,d], where d is the number of attributes. Higher values of $minAtt$ corresponds to more specialization leading to a larger number of rules and a lower value of $minAtt$ corresponds to more generalization leading to fewer number of rules. But over generalization can make normal and attack data indistinguishable resulting in more false positives or false negatives. So, a lower limit for $minAtt$ is specified using the parameter MIN. Now, the possible range of values for $minAtt$ becomes [MIN,d]. To determine the value of MIN, information gain [20] is calculated for each attribute in the training dataset. Attributes with very low information gain can be ignored and the number of remaining attributes is set as the value for MIN. The parameter $minSize$ is the minimum number of data elements received to form a cluster. In some cases, at least one training example for an attack should be considered significant. So, the minimum value for $minSize$ is 1. Its value should increase with increase in $minAtt$ value. The maximum value for $minSize$ can be set to be any multiple of $log_2(n)$, where n is the number of training examples.

In our experiments, MIN = 27. The value for $MinAtt$ is set to 33, which is gradually reduced to MIN = 27 in steps of -1. The starting value for $MinSize$ is set to $2log_2(n)$. The similarity thresholds for all attributes are set to zero, $\delta_i = 0, i = 1, \cdots, d$.

4.5 Performance measures

Evaluation of performance of a classification model is based on the counts of test records correctly and incorrectly predicted by the model. These counts are tabulated in a confusion matrix [21]. A confusion matrix that summarizes the number of instances predicted correctly or incorrectly by a classification model is shown in Table 6.

Table 6: Confusion matrix for binary classification

		Predicted Class	
		+	-
Actual Class	+	$f_{++}\ (TP)$	$f_{+-}\ (FN)$
	-	$f_{-+}\ (FP)$	$f_{--}\ (TN)$

Recall and Precision are two widely used metrics employed in applications where successful detection of one of the classes is considered more significant than detection of the other classes. A formal definition of the metrics is given below.

$$Precision,\ p = \frac{TP}{TP + FP} \tag{7}$$

$$Recall,\ r = \frac{TP}{TP + FN} \tag{8}$$

Building a model that maximizes both precision and recall is the key challenge for classification algorithms. Another measure of accuracy is PCC (percentage of correct classification). A formal definition of PCC is given below.

$$PCC = \frac{TP + TN}{TP + FN + FP + TN} \tag{9}$$

A high value of PCC ensures reasonably correct classification.

5 Experimental results

The experiments were performed on a 3 GHz HP dc 7000 series desktop with 2 GB RAM, 250 GB HDD. C++ programs were used in a LINUX environment. We provide the cross evaluation result performing training and testing of the method with corresponding training and testing datasets.

5.1 Results on TUIDS Intrusion Datasets

Two categories of experiments 2-class ($normal$ and $attack$) and all-attacks behaviors are carried out.

5.1.1 2-class prediction results

The confusion matrices for the 2-class behavioral categories on the *Packet Level*, *Flow Level* and *Portscan* datasets are shown in Table 7. Classification rates with *PCC* 99.72%, 99.70 and 98.31% for *Packet Level*, *Flow Level* and *Portscan* datasets, respectively indicate good performance for our method.

Table 7: 2-class confusion matrix for TUIDS datasets

Data set	Actual Class	Predicted Class				
		$Normal$	$Attack$	Sum	Recall	1-Prc*
Packet level	$Normal$	34790	253	35043	0.9928	0.0267
	$Attack$	955	396877	397832	0.9976	0.0006
	Sum	35745	397130	432875		
		Re-substitution error =0.0028			PCC=99.72%	
		Predicted Class				
		$Normal$	$Attack$	Sum	Recall	1-Prc*
Flow level	$Normal$	36107	295	36402	0.9919	0.0246
	$Attack$	910	362819	363729	0.9975	0.0008
	Sum	37017	363114	400131		
		Re-substitution error =0.0030			PCC=99.70%	
		Predicted Class				
		$Normal$	$Attack$	Sum	Recall	1-Prc*
Portscan	$Normal$	2414	31	2445	0.9876	0.2175
	$Attack$	671	38544	39215	0.9829	0.0009
	Sum	3085	38575	41660		
		Re-substitution error =0.0169			PCC=98.31%	

Note- *1-Precision

5.1.2 All-attacks prediction results

The confusion matrices for all-attacks categories on the *Packet level*, *Flow level* and*Portscan* datasets are shown in Table 8. Classification rates of *PCC* are 99.42%, 99.01% and 98.31% for *Packet level*, *Flow level* and*Portscan* datasets, respectively. The average execution time of classification for *Packet level*, *Flow level* and *Portscan* datasets are 0.39 minute, 0.17 minute and 0.14 minute, respectively.

5.2 Results on KDD Cup 1999 dataset

We perform three different experiments which are categorized into 2-class ($normal$ and $attack$), 5-class ($normal$, $R2L$, DoS, $Probe$ and $U2R$) and all-attacks behaviors.

5.2.1 2-class prediction results

The confusion matrix for the 2-class behavioral categories on the $Corrected\ KDD$ is shown in Table 9. Classification model is trained with 10-$percentKDD$ dataset. The classification rate is $PCC = 93.40\%$. The performance degrades due to the fact that no examples are present in the training set corresponding to the 15 categories of attacks that are present in the testing set. Most of these attacks are misclassified as normal by our algorithm.

5.2.2 5-class prediction results

The confusion matrix for the 5-class behavioral categories on the $Corrected\ \ KDD$ datasets is shown in Table 10. The classification model is trained with the 10-$percentKDD$ dataset. In this case most of the unseen attacks are misclassified as normal category and PCC reduced to 92.39%.

5.2.3 All-attacks prediction results

The confusion matrix for the all-attacks categories on the $Corrected\ KDD$ is shown in Table 11. The classification model is trained with $10\%KDD$ dataset that includes

Table 10: 5-class confusion matrix results of $Corrected\ KDD$ dataset training with $10\%KDD$ dataset

		Predicted Class							
		$Normal$	$R2L$	DoS	$Probe$	$U2R$	Sum	Recall	1-Prc*
Actual class	$Normal$	60211	21	73	284	4	60593	0.9937	0.7386
	$R2L$	13868	1115	3	1356	5	16347	0.0682	0.9378
	DoS	6360	42	223349	102	0	229853	0.9717	0.9977
	$Probe$	1036	0	443	2687	0	4166	0.6450	0.6067
	$U2R$	49	11	0	0	10	70	0.1429	0.5263
	Sum	81524	1189	223868	4429	19	311029		
		Re-substitution error=0.0761						PCC=92.39%	

Note- *1-Precision

22 attack classes. The testing dataset $Corrected\ KDD$ include 37 attack classes. Since no examples are present in the training set, objects belonging to the 15 categories of attacks cannot be classified correctly and hence we exclude them (18729 objects) from the testing dataset. We see from the results that DoS and $Probe$ attacks are well detected but $R2L$ and $U2R$ attacks are detected very poorly. It is apparent from that there are very few examples in the training set corresponding to attacks belonging to $R2L$ and $U2R$ categories compared to DoS, $Probe$ and normal categories. The average execution time of classification for Corrected KDD dataset is 1 minute.

5.3 Results on NSL-KDD dataset

NSL-KDD [19] is a data set for network-based intrusion detection systems. It is the new version of KDD Cup 1999 intrusion detection benchmark dataset. In the KDD Cup dataset, there are huge number of redundant records, which can cause the learning algorithms to be biased towards the frequent records. To solve this issue, one copy of each record is kept in the NSL-KDD data set. Though, this dataset is not the perfect representative of real networks, still, it can be applied as an effective benchmark dataset to compare different intrusion detection methods. NSL-KDD included two datasets $KDDTrain^+$ and $KDDTest^+$ are shown in Table 5.

5.3.1 2-class prediction results

The confusion matrix for the 2-class behavior category on the $KDDTest^+$ datasets is shown in Table 12. The classification model is trained with $KDDTrain^+$ dataset. Classification rate of $PCC = 98.34\%$ in Table 12 indicate good performance for our method. The performance for the datasets is better compared to the KDD Cup dataset for 2-class classification.

5.3.2 5-class prediction results

The confusion matrix for the 5-class behavioral category of $KDDTest^+$ dataset is shown in Table 16. The classification model is trained with $KDDTrain^+$ dataset. Classification rate of $PCC = 98.39\%$ in Table 16 indicate good performance for our method datasets.

5.3.3 All-attacks prediction results

The confusion matrix for the all-attacks categories on $KDDTest^+$ dataset is shown in Table 13. Here, the classification model is trained with $KDDTrain^+$ dataset. The testing dataset $KDDTest^+$ includes 37 attack classes and training dataset $KDDTtrain^+$ includes 15 attack classes. Objects belonging to the 15 categories of attacks cannot be classified correctly since no examples are present in the training set and hence they are excluded (3751 objects) from the testing dataset. Classification rate of $PCC = 98.94\%$ in Table 13 indicates good performance for our method. The classification rates still remain better compared to the KDD Cup dataset. The average execution time of classification for KDDTest+ dataset is 0.07 minute.

5.3.4 Performance comparisons

The algorithms $C4.5$ [22], $CART$ [23], $BayesianNetwork(BN)$ [24] and $CN2$ [25] rule-based algorithm are executed for TUIDS dataset using Weka [26]. Their results are compared with method as given in Table 17 for TUIDS datasets. The performance results of our method outperforms the other methods.

The performance results of our method for *Corrected KDD* dataset is compared with the experiment results provided in [14] for methods $C4.5$ [14, 22], $CART$ [14, 23], $BayesianNetwork(BN)$ [14, 24] and $CN2$ [14, 25] rule-based algorithm as given in Tables 18-20. We see that in terms of PCC our algorithm performs better than these algorithms in most of the cases.

The performance rates of SVM-based IDS [15], a combined method of hierarchical clustering and SVM method, are shown in Tables 14 and 15 using *Corrected KDD* dataset. The evaluation result using the $Corrected\ KDD$ dataset, shows that our algorithm performance rates are distinctly higher in the 5-class attack behavior and in new attack detection. The accuracy rate of detection for DoS attacks is 99.99% as shown in Table 14. In the detection of $snmpgetattack$ and $snmpguess$ attacks, our method detects 6457 and 2360 records respectively, whereas the other method detected 0 and 1 record respectively as shown in Table 15.

Table 16: 5-class confusion matrix of $KDDTest^+$ dataset trined with $KDDTrain^+$ dataset

		Predicted Class							
		$Normal$	$R2L$	DoS	$Probe$	$U2R$	Sum	Recall	1-Prc*
Actual class	$Normal$	9556	125	5	24	0	9710	0.9841	0.0195
	$R2L$	157	2727	0	0	3	2887	0.9446	0.0482
	DoS	14	1	7443	0	0	7458	0.9980	0.0009
	$Probe$	16	5	2	2399	0	2422	0.9905	0.0099
	$U2R$	3	7	0	0	57	67	0.8507	0.0500
	Sum	9746	2865	7450	2423	60	22544		
		Re-substitution error=0.0161						PCC=98.39%	

Note- *1-Precision

Table 17: Comparison among CART, C4.5, CN2, BN for all Attacks of TUIDS datasets

Data set	**Attack values**	**CART**		**C4.5**		**CN2**		**BN**		**Our Algorithm**	
		Recall	**1-Prc***	**Recall**	**1-Prc***	**Recall**	**1-Prc***	**Recall**	**1-Prc***	**Recall**	**1-Prc***
Packet level	$normal$	0.9430	0.0540	0.9436	0.0507	0.9822	0.1442	0.7855	0.0215	**0.9928**	0.0014
	$smurf$	0.6360	0.3862	0.6384	0.3828	0.0026	0.0000	0.9503	0.6135	**0.9981**	0.0403
	1234	0.7289	0.1721	0.2353	0.5000	0.7900	0.2410	0.8400	0.2510	**0.9883**	0.0550
	$bonk$	0.8600	0.3156	0.7500	0.2100	0.8300	1.0000	0.8950	0.2400	0.5714	0.2000
	$fraggle$	1.0000	0.0000	1.0000	0.0000	1.0000	0.0011	0.9995	0.0012	**1.0000**	0.0000
	$jolt$	0.9641	0.0199	0.9902	0.0000	0.9248	0.0752	0.9739	0.2905	**0.9998**	0.0424
	$nestea$	0.7694	0.2310	0.0556	0.0000	6830	0.3100	0.8450	0.2010	**0.9993**	0.0027
	$newtear$	0.6829	0.2390	0.6300	0.2190	0.5910	0.2110	0.9300	0.3400	**0.9900**	0.0000
	$oshare$	0.7590	0.3270	0.8200	0.1040	0.6703	0.1401	0.8700	0.1031	**1.0000**	0.0000
	$saihyousen$	0.8900	0.1590	0.2353	0.5000	0.7810	0.2110	0.8505	0.2030	**0.9500**	0.0952
	$syndrop$	0.5910	0.3410	0.4550	0.1060	0.6900	0.1030	0.5450	0.1300	**1.0000**	0.0331
	syn	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0011	0.9995	0.0012	0.9931	0.0014
	$teardrop$	0.9641	0.0199	**0.9902**	0.0000	0.9248	0.0752	0.9739	0.2905	0.9759	0.0714
	$window$	0.7910	0.3200	0.0556	0.0000	0.7500	0.2100	0.7480	0.2041	**0.9881**	0.2727
	$winnuke$	0.6830	0.1520	0.5800	0.2170	0.5950	0.2140	0.6200	0.2100	**1.0000**	0.0000
	$xmas$	0.7420	0.1320	0.7300	0.2380	0.6350	0.2100	0.7900	0.2100	**0.9911**	0.0632
		PCC	98.30%		98.65%		95.67%		94.15%		99.42%
Flow level	$normal$	0.9500	0.2310	0.8850	0.2200	0.8350	0.2101	0.7950	0.0067	**0.9918**	0.0018
	$smurf$	0.8507	0.2030	0.7900	0.1220	0.8900	0.2020	0.8105	0.0821	0.6000	0.0103
	1234	0.9863	0.0137	0.9944	0.0293	0.8546	0.1428	0.9850	0.4125	**0.9999**	0.0150
	$bonk$	0.9800	0.1040	0.8700	0.0098	0.9300	0.2055	0.8600	0.0025	**0.9961**	0.0340
	$fraggle$	0.9724	0.2917	0.8598	0.0628	0.8898	0.1113	0.8677	0.1160	**0.9987**	0.0029
	$jolt$	0.8500	0.2010	0.2308	0.0000	0.8250	0.0712	0.9100	0.1090	0.8486	0.0474
	$land$	0.9706	0.0404	0.8927	0.0389	0.8965	0.0268	0.9924	0.0905	**0.9998**	0.0000
	$nestea$	**0.9750**	0.0250	0.9789	0.0000	0.8762	0.1086	0.9657	0.0639	0.5217	0.0021
	$newtear$	0.7809	0.3105	0.8700	0.2110	0.9700	0.2510	0.6950	0.0910	0.7956	0.0062
	$saihyousen$	0.8510	0.1700	0.9881	0.3197	**1.0000**	0.0000	**1.0000**	0.6147	0.7817	0.0059
	$syndrop$	0.9205	0.1850	0.9769	0.0000	0.8900	0.0072	0.6900	0.2900	0.6515	0.0371
	syn	**0.9990**	0.0016	0.9978	0.0011	0.9994	0.0011	0.9923	0.0000	0.9775	0.0214
	$teardrop$	0.4500	0.0230	0.5000	0.0000	0.4900	0.0087	0.7800	0.0426	0.5769	0.0314
	$window$	0.8950	0.0089	0.9400	0.2330	0.7800	0.2600	0.8100	0.1100	**0.9832**	0.3728
	$winnuke$	**1.0000**	0.4583	0.9545	0.0132	0.7951	0.1164	0.9791	0.0835	0.9740	0.0000
	$xmas$	0.7025	0.5088	0.8038	0.3553	0.8987	0.1744	0.9494	0.4700	**0.9720**	0.0538
		PCC	98.69%		97.19%		95.26%		95.15%		99.01%
Portscan	$Normal$	0.9342	0.3527	0.5000	0.0078	0.7263	0.4219	0.8621	0.0069	**0.9876**	0.0024
	SYN	0.8458	0.3910	0.8600	0.7610	0.6500	0.4510	0.5500	0.3290	**0.9859**	0.0040
	ACK	**1.0000**	0.4583	0.9545	0.0132	0.8252	0.1164	0.9791	0.0835	0.9930	0.0252
	FIN	0.7536	0.5088	0.8038	0.3553	0.9393	0.1744	0.9494	0.4700	**0.9766**	0.0153
	$xmas$	0.8813	0.4510	0.6800	0.24100	0.7502	0.3160	0.6578	0.2715	**0.9761**	0.0143
		PCC	87.45%		82.37%		78.76%		78.99%		98.31%

Note- *1-Precision

Table 18: Comparison among CART, C4.5, CN2, BN for two-class on $Corrected\ KDD$ dataset

	CART		**C4.5**		**CN2**		**BN**		**Our Algorithm**	
Values	**Recall**	**1-Prc***	**Recall**	**1-Prc***	**Recall**	**1-Prc***	**Recall**	**1-Prc***	**Recall**	**1-Prc***
$Normal$	0.9297	0.0511	0.9297	0.0511	0.8917	0.0640	0.9869	0.1835	0.9001	0.0269
$Attack$	0.9879	0.0169	0.9879	0.0145	0.9852	0.0259	0.9463	0.0033	**0.9940**	0.0237
PCC		97.65%		97.85%		96.69%		95.42%		97.57%

Note- *1-Precision

Table 19: Comparison among CART, C4.5, CN2, BN for five-class on $Corrected\ KDD$ dataset

	CART		**C4.5**		**CN2**		**BN**		**Our Algorithm**	
Values	**Recall**	**1-Prc***	**Recall**	**1-Prc***	**Recall**	**1-Prc***	**Recall**	**1-Prc***	**Recall**	**1-Prc***
$Normal$	0.9379	0.0526	0.9442	0.0526	0.8708	0.0490	0.7946	0.0624	0.9007	0.0273
$R2L$	0.7813	0.1919	0.8153	0.1927	0.8451	0.3561	0.8871	0.4554	**0.9110**	0.2873
DoS	0.9984	0.0027	0.9997	0.0011	0.9993	0.0015	0.9811	0.0022	**0.9999**	0.0001
$Probe$	0.9081	0.2525	0.9482	0.0403	0.9585	0.0230	0.8778	0.3036	**0.9875**	0.0044
$U2R$	0.5526	0.4545	0.6711	0.0192	0.6754	0.12	0.7281	0.9195	**0.7857**	0.0517
PCC		97.37%		97.83%		96.54%		93.83%		97.57%

Note- *1-Precision

Table 20: Comparison among CART, C4.5, CN2, BN for all Attacks on $Corrected\ KDD$ dataset

Attack values	**CART**		**C4.5**		**CN2**		**BN**		**Our Algorithm**	
	Recall	**1-Prc***	**Recall**	**1-Prc***	**Recall**	**1-Prc***	**Recall**	**1-Prc***	**Recall**	**1-Prc***
$normal$	0.9430	0.0540	**0.9436**	0.0507	0.9822	0.1442	0.7855	0.0215	0.9016	0.0257
$snmpgetattack$	0.6360	0.3862	0.6384	0.3828	0.0026	0.0000	**0.9503**	0.6135	0.8341	0.4792
$named$	0.0000	1.0000	0.2353	0.5000	0.0000	1.0000	0.0000	1.0000	**0.8824**	0.3750
$xlock$	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	**0.8889**	0.1111
$smurf$	1.0000	0.0000	1.0000	0.0000	1.0000	0.0011	0.9995	0.0012	**1.0000**	0.0000
$ipsweep$	0.9641	0.0199	0.9902	0.0000	0.9248	0.0752	0.9739	0.2905	**0.9869**	0.0033
$multihop$	0.0000	1.0000	0.0556	0.0000	0.0000	1.0000	0.0000	1.0000	**0.7222**	0.0714
$xsnoop$	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	**0.7500**	0.0000
$sendmail$	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	**0.9412**	0.2000
guess_passwd	0.9968	0.0566	0.9863	0.0242	0.9725	0.0270	0.9679	0.0404	**0.9979**	0.0002
$saint$	0.1236	0.0000	0.8302	0.2382	0.8003	0.2209	0.7024	0.3277	**0.9402**	0.6121
buffer_overflow	0.0000	1.0000	0.4545	0.4118	0.0000	1.0000	0.0000	1.0000	**0.9545**	0.0455
$portsweep$	0.8362	0.1111	0.9463	0.1604	0.7260	0.1376	**0.9915**	0.2252	0.9746	0.0000
pod	0.8391	0.0000	**1.0000**	0.4082	0.8391	0.0000	0.7701	0.4071	0.9655	0.0118
$apache2$	0.0000	1.0000	0.9937	0.1841	0.8476	0.0399	0.9786	0.0152	**0.9987**	0.0000
phf	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	**1.0000**	0.0000
$udpstorm$	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	**1.0000**	0.0000
$warezmaster$	0.9863	0.0137	**0.9944**	0.0293	0.8546	0.1428	0.9850	0.4125	0.9744	0.0013
$perl$	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	**1.0000**	0.0000
$satan$	**0.9724**	0.2917	0.8598	0.0628	0.8898	0.1113	0.8677	0.1160	0.3270	0.0582
$xterm$	0.0000	1.0000	0.2308	0.0000	0.0000	1.0000	0.0000	1.0000	**0.7692**	0.0000
$mscan$	0.9706	0.0404	0.8927	0.0389	0.8965	0.0268	0.9924	0.0905	**1.0000**	0.0000
$processtable$	0.9750	0.0250	0.9789	0.0000	0.8762	0.1086	0.9657	0.0639	**1.0000**	0.0000
ps	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	**0.8125**	0.0000
$nmap$	0.0000	1.0000	0.9881	0.3197	1.0000	0.0000	1.0000	0.6147	**1.0000**	0.0000
$rootkit$	0.0000	1.0000	0.0769	0.0000	0.0000	1.0000	0.0000	1.0000	**0.6923**	0.1818
$neptune$	0.9990	0.0016	0.9978	0.0011	0.9994	0.0011	0.9923	0.0000	**1.0000**	0.0000
$loadmodule$	0.0000	1.0000	0.5000	0.0000	0.0000	1.0000	0.0000	1.0000	**1.0000**	0.0000
$imap$	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	**1.0000**	0.0000
$back$	1.0000	0.4583	0.9545	0.0132	0.7951	0.1164	0.9791	0.0835	**1.0000**	0.0000
$httptunnel$	0.7025	0.5088	0.8038	0.3553	0.8987	0.1744	0.9494	0.4700	**0.9873**	0.0000
$worm$	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000
$mailbomb$	0.9516	0.0000	0.9982	0.0062	**0.9998**	0.0161	0.9992	0.0046	**0.9998**	0.0000
ftp_write	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	**1.0000**	**1.0000**	0.0000
$teardrop$	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	**0.4167**	0.1667
$land$	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	**1.0000**	0.0000
$sqlattack$	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	**1.0000**	0.0000
$snmpguess$	0.9909	0.0004	**0.9983**	0.0144	0.3603	0.3812	0.3624	0.4718	0.9809	0.0000
PCC		97.25%		97.69%		96.16%		94.75%		97.25%

Note- *1-Precision

Table 8: All-attacks confusion matrix for TUIDS datasets

Data set	Actual Class	Predicted Class			
		Detected	Present	Recall	1-Prc*
Packet level	$normal$	34790	35043	0.9928	0.0014
	$smurf$	38652	38726	0.9981	0.0403
	1234	19679	19902	0.9883	0.0550
	$bonk$	387	678	0.5714	0.2000
	$fraggle$	8622	8624	1.0000	0.0000
	$jolt$	27659	27661	0.9998	0.0424
	$nestea$	23758	23775	0.9993	0.0027
	$newtear$	23537	23775	0.9900	0.0000
	$oshare$	20000	20000	1.0000	0.0000
	$saihyousen$	4338	4567	0.9500	0.0952
	$syndrop$	53623	53627	1.0000	0.0331
	syn	20230	20371	0.9931	0.0014
	$teardrop$	8171	8373	0.9759	0.0714
	$window$	39908	40386	0.9881	0.2727
	$winnuke$	68248	68249	1.0000	0.0000
	$xmas$	38771	39118	0.9911	0.0632
	Sum	430373	432875		
	Re-substitution error=0.0058			PCC=99.42%	
	Actual Class	Predicted Class			
		Detected	Present	Recall	1-Prc*
Flow level	$normal$	36103	36402	0.9918	0.0018
	$smurf$	18	30	0.6000	0.0103
	1234	163408	163412	0.9999	0.0150
	$bonk$	27196	27303	0.9961	0.0340
	$fraggle$	38238	38288	0.9987	0.0029
	$jolt$	1183	1394	0.8486	0.0474
	$land$	2	2	0.9998	0.0000
	$nestea$	48	92	0.5217	0.0021
	$newtear$	109	137	0.7956	0.0062
	$saihyousen$	197	252	0.7817	0.0059
	$syndrop$	43	66	0.6515	0.0371
	syn	20330	20797	0.9775	0.0214
	$teardrop$	75	130	0.5769	0.0314
	$window$	40947	41646	0.9832	0.3728
	$winnuke$	29915	30714	0.9740	0.0000
	$xmas$	38361	39465	0.9720	0.0538
	Sum	396173	400131		
	Re-substitution error=0.0099			PCC=99.01%	
	Actual Class	Predicted Class			
		Detected	Present	Recall	1-Prc*
Portscan	$Normal$	2414	2445	0.9876	0.0024
	SYN	9612	9750	0.9859	0.0040
	ACK	9875	9945	0.9930	0.0252
	FIN	9551	9780	0.9766	0.0153
	$xmas$	9505	9740	0.9761	0.0143
	Sum	40957	41660		
	Re-substitution error=0.0169			PCC=98.31%	

Note- *1-Precision

Table 9: 2-class Confusion matrix of $Corrected\ KDD$ with training $10\%KDD$ dataset

		Predicted Class				
		$Normal$	$Attack$	Sum	Recall	1-Prc*
Actual class	$Normal$	60215	378	60593	0.9938	0.2508
	$Attack$	20155	230281	250436	0.9195	0.0016
	Sum	80370	230659	311029		
	Re-substitution error=0.0660				PCC=93.40%	

Note- *1-Precision

Table 11: All-attacks confusion matrix results of $Corrected\ KDD$ dataset training with $10\%KDD$

		Predicted Class			
		Detected	Present	Recall	1-Prc*
Actual class	$normal$	60207	60593	0.9936	0.0790
	$smurf$	164089	164091	1.0000	0.0002
	$ipsweep$	298	306	0.9739	0.0418
	$multihop$	0	18	0.0000	1.0000
	$guess\_passwd$	3	4367	0.0007	0.2500
	$buffer\_overflow$	2	22	0.0909	0.5000
	$portsweep$	340	354	0.9605	0.2075
	pod	82	87	0.9425	0.1546
	phf	1	2	0.5000	0.0000
	$warezmaster$	2	1602	0.0012	0.0000
	$perl$	0	2	0.0000	1.0000
	$satan$	1280	1633	0.7838	0.1551
	$nmap$	84	84	1.0000	0.0455
	$rootkit$	2	13	0.1538	0.9962
	$neptune$	57971	58001	0.9995	0.0004
	$loadmodule$	0	2	0.0000	1.0000
	$imap$	0	1	0.0000	1.0000
	$back$	1068	1098	0.9727	0.0000
	ftp_write	0	3	0.0000	1.0000
	$teardrop$	12	12	1.0000	0.7857
	$land$	6	9	1.6667	0.6667
	$warezclient$	0	0	0.0000	1.0000
	spy	0	0	0.0000	1.0000
	Sum	285447	292300		
	Re-substitution error=0.0234			PCC=97.66%	

Note- *1-Precision

Table 12: 2-class confusion matrix of $KDDTest^{+}$ dataset training with $KDDTrain^{+}$

		Predicted Class				
		$Normal$	$Attack$	Sum	Recall	1-Prc*
Actual class	$Normal$	9531	179	9710	0.9816	0.0200
	$Attack$	195	12639	12834	0.9848	0.0140
	Sum	9726	12818	22544		
	Re-substitution error =0.0166				PCC=98.34%	

Note- *1-Precision

Table 13: All attacks confusion matrix of $KDDTest^+$ dataset

		Predicted Class			
		Detected	Present	Recall	1-Prc*
Actual class	$normal$	9609	9710	0.9895	0.0125
	$smurf$	605	665	0.9098	0.0529
	$ipsweep$	136	141	0.9845	0.0178
	$multihop$	17	18	0.9444	0.2273
	$guess\_passwd$	1130	1231	0.9180	0.0238
	$buffer\_overflow$	20	20	1.0000	0.0000
	$portsweep$	149	157	0.9490	0.0688
	pod	39	41	0.9512	0.0000
	phf	2	2	1.0000	0.0000
	$warezmaster$	902	944	0.9555	0.0075
	$perl$	2	2	1.0000	0.0000
	$satan$	716	735	0.9741	0.1836
	$nmap$	73	73	1.0000	0.0000
	$rootkit$	12	13	0.9231	0.2500
	$neptune$	4626	4657	0.9933	0.0008
	$loadmodule$	2	2	1.0000	0.0000
	$imap$	1	1	1.0000	0.0000
	$back$	339	359	0.9443	0.3416
	ftp_write	3	3	1.0000	0.0000
	$teardrop$	5	12	0.4167	0.2857
	$land$	7	7	1.0000	0.0000
	Sum	18593	18793		
	Re-substitution error=0.0181			PCC=98.94%	

Note- *1-Precision

Table 14: Comparison with SVM-based IDS for 5-class over $Corrected\ KDD$ dataset

Type of Traffic	**SVM-based IDS**			**Accuracy of Our Algorithm (%)**
	Correctly Detected	**Miss Detected**	**Accuracy (%)**	
$Normal$	60166	427	99.29	90.07
DoS	228769	1084	99.53	**99.99**
$Probe$	4064	102	97.55	**98.75**
$U2R$	45	183	19.73	78.57
$R2L$	4664	11525	28.81	**91.10**
Overall	297708	13321	95.72	**97.57**

Table 15: Comparison with SVM-based IDS for attack detection over $Corrected\ KDD$ dataset

Attack Name	**Attack present**	**Detection**	
		SVM-based IDS	**Our Algorithm**
$apache2$	794	536	793
$mailbomb$	5000	4459	4999
$processtable$	759	578	759
$mscan$	1053	981	1053
$saint$	736	724	692
$httptunnel$	158	15	156
ps	16	3	13
$sqlattack$	2	2	2
$xterm$	13	5	10
$sendmail$	17	2	**16**
$named$	17	3	**15**
$snmpgetattack$	7741	0	**6457**
$snmpguess$	2406	1	**2360**
$xlock$	9	2	**8**
$xsnoop$	4	0	**3**
$worm$	2	0	0
Total	18729	39.04%	**92.56%**

6 Conclusion and future works

In this paper we provide a clustering based classification method and applied it in network anomaly detection. We have developed a subspace based incremental clustering method which forms the basis for the classification method. A training algorithm with a combination of unsupervised incremental clustering and supervised classification algorithm clusters a labeled training dataset into different clusters which are then represented by their profiles. These profiles together with the class label behave as classification rules. Prediction is done using a supervised classification algorithm that matches testing objects with the cluster profiles for labeling them. The simple classification method provided is effective in network anomaly detection as indicated by the evaluation results on real life TUIDS intrusion dataset and the benchmark intrusion datasets. The method can be applied for other classification jobs as well.

At present, rules (profiles) are stored in a flat file. A decision tree can be constructed based on the derived rules to reduce search time. Investigation for dealing with test instances that fit poorly with the supervised profiles may increase the performance of the algorithm. Another possible area is rule refinement that may cause reduction in number of attributes in each rule and also reduction of some rules themselves. For instance, KDD 1999 data set contains many similar training examples having different labels. In such situation, similar rules may be derived to detect more than one category of attacks.

Acknowledgement

This work is an outcome of a research project funded by MCIT, New Delhi.

References

[1] W. Lee and S. J. Stolfo, "Data mining approaches for intrusion detection," in *Proc. of the 7th conf. on USENIX Security Symposium - Volume 7*. San Antonio, Texas, USA: USENIX, 26-29, January 1998, pp. 1–16.

[2] R. Agarwal and M. V. Joshi, "PNrule: A New Framework for Learning Classifier Models in Data Mining (A Case-Study in Network Intrusion Detection)," Department of Computer Science, University of Minnesota, Tech. Rep. TR 00-015, 2000.

[3] K. Burbeck and S. Nadjm-Tehrani, "ADWICE - Anomaly Detection with Real-Time Incremental Clustering," in *Proc. of Information Security and Cryptology -ICISC 2004*, vol. 3506/2005. Berlin, Germany: Springer, May 2005, pp. 407–424.

[4] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: an effective data clustering method for very large databases," *SIGMOID Record 1996 ACM SIGMOID*

Int'nl Conf. on Management of Data, vol. 25, pp. 103–114, 1996.

[5] A. Abraham and R. Jain, "Soft Computing Models for Network Intrusion Detection Systems," *Studies in Computational Intelligence*, vol. 16, pp. 191–211, 2005.

[6] H. Yang, F. Xie, and Y. Lu, "Clustering and Classification Based Anomaly Detection," *Fuzzy Systems and Knowledge Discovery*, vol. 4223/2006, pp. 1082–1091, 2006.

[7] A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur, and J. Srivastava, "A Comparative Study of Supervised Machine Learning Techniques for Intrusion Detection," in *Proc. of the 3rd SIAM Int'nl Conf. on Data Mining*. San Francisco, CA: SIAM, May 1-3 2003, pp. 25–36.

[8] G. Stein, B. Chen, A. S. Wu, and K. A. Hua, "Decision tree classifier for network intrusion detection with GA-based feature selection," in *Proc. of the 43rd Annual Southeast Regional Conference, ACM-SE 43*, 2005, pp. 136–141.

[9] J. Zhang and M. Zulkernine, "Network intrusion detection using random forest," in *Proc. of the 3rd Annual Conf. on Privacy, Security and Trust*, 2005, pp. 53–61.

[10] Y. Li, B.-X. Fang, L. Guo, and Y. Chen, "TCM-KNN Algorithm for Supervised Network Intrusion Detection," *LNCS*, vol. 4430/2007, pp. 141–151, 2007.

[11] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Macia-Fernandez, and E. Vazquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Computers & Security*, vol. 28, pp. 18–28, 2009.

[12] P. Gogoi, B. Borah, and D. K. Bhattacharyya, "Anomaly Detection Analysis of Intrusion Data using Supervised & Unsupervised Approach," *Journal of Convergence Information Technology*, vol. 5(1), pp. 95–110, Feb. 2010.

[13] P. Gogoi, D. K. Bhattacharyya, B. Borah, and J. K. Kalita, "A Survey of Outlier Detection Methods in Network Anomaly Identification," *The Computer Journal*, vol. 54(4), pp. 570–588, 2011.

[14] R. Beghdad, "Critical Study of Supervised Learning Techniques in Predicting Attacks," *Information Security Journal: A Global Perspective*, vol. 19, pp. 22–35, 2010.

[15] S.-J. Horng, M.-Y. Su, Y.-H. Chen, T.-W. Kao, R.-J. Chen, J.-L. Lai, and C. D. Perkasa, "A novel intrusion detection system based on hierarchical clustering and support vector machines," *Expert Systems with Applications*, vol. 38, pp. 306–313, 2011.

[16] M. Roesch, "Snort-lightweight intrusion detection for networks," in *Proc. of the 13th USENIX conf. on System administration*. Seattle, Washington: USENIX, Nov. 1999, pp. 229–238.

[17] P. Gogoi, M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Packet and Flow Based Network Intrusion Dataset," in *LNCS Proc. of IC3 2012*, ser. CCIS, vol. 306. Berlin Heidelberg: Springer, August 6-8 2012, pp. 322–334.

[18] University of California, "KDD Cup 1999 Data," http://kdd.ics.uci.edu/∼/kddcup99.html, Irvine, 1999.

[19] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," in *Proc. of the 2009 IEEE Symposium on Computational Intelligence in Security and Defence Applications (CISDA 2009)*. Ottawa, Canada: IEEEXplore, 8-10 July 2009, pp. 1–6.

[20] H. G. Kayacik, A. N. Z. Heywood, and M. I. Heywood, "Selecting Features for Intrusion Detection: A Feature Relevance Analysis on KDD 99 Intrusion Detection Datasets," in *Proc. of the 3rd Annual Conf. on Privacy, Security and Trust*. Halifax, Dalhousie University, October 2005.

[21] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Pearson Education, Inc., 2009.

[22] J. R. Quinlan, "C4.5: Programs for Machine Learning," Morgan Kaufman, New York, USA, 1993.

[23] L. Brieman, J. H. Friedman, R. A. Olshen, and C. Stone, "Classification and reregression trees," Wadsworth & Brooks, Monterey, CA, 1984.

[24] F. V. Jesen, "Introduction to Bayesien networks," UCL Press, 1996.

[25] P. Clark and T. Niblett, "The CN2 induction algorithm," *Machine Learning*, vol. 3, pp. 261–283, 1989.

[26] University of Waikato, "Weka 3: Data Mining Software in Java," http://www.cs.waikato.ac.nz/ml/weka, New Zealand, 1999.

Considering Autocorrelation in Predictive Models

Daniela Stojanova
Department of Knowledge Technologies, Jožef Stefan Institute, Jamova cesta 39 Ljubljana Slovenia
E-mail: Daniela.Stojanova@ijs.si, http://kt.ijs.si/daniela_stojanova

Thesis Summary

Keywords: autocorrelation, predictive models, predictive clustering trees

Received: February 21, 2013

This article presents a summary of the doctoral dissertation of the author, which addresses the task of considering autocorrelation in predictive models.

Povzetek: Članek predstavlja povzetek doktorske disertacije avtorice, ki obravnava nalogo upoštevanja autokorelacije v napovednih modelih.

1 Introduction

Most machine learning, data mining and statistical methods rely on the assumption that the analyzed data points are independent and identically distributed (i.i.d.). More specifically, the individual examples included in the training data are assumed to be drawn independently from each other from the same probability distribution. However, cases where this assumption is violated can be easily found: For example, species are distributed non-randomly across a wide range of spatial scales. The i.i.d. assumption is often violated because of the phenomenon of autocorrelation.

The cross-correlation of an attribute with itself is typically referred to as autocorrelation: This is the most general definition found in the literature. Specifically, in spatial analysis, spatial autocorrelation has been defined as the correlation among data values, which is strictly due to the relative location proximity of the objects that the data refer to. It is justified by Tobler's first law of geography [1] according to which "everything is related to everything else, but near things are more related than distant things". In network studies, autocorrelation is defined by the homophily principle [2] as the tendency of nodes with similar values to be linked with each other.

2 Methods and evaluation

In this thesis, we first give a general definition of the autocorrelation phenomenon, which includes spatial and network autocorrelation for continuous and discrete responses. We then present a broad overview of the existing autocorrelation measures for the different types of autocorrelation and data analysis methods that consider them. Focusing on spatial and network autocorrelation, we propose three algorithms that handle non-stationary autocorrelation within the tasks of classification, regression and structured output prediction. The algorithms are based on the concept of predictive clustering trees (PCTs) [3], according to which hierarchies of clusters of similar data are identified and a predictive model is associated to each cluster.

The first algorithm, SCLUS (Spatial Predictive Clustering System) [4] , explicitly considers spatial autocorrelation when learning predictive clustering models. The method is able to learn predictive models for both a continuous response (regression task) and a discrete response (classification task). It can deal with autocorrelation in the data and provide a multi-level insight into the spatial autocorrelation phenomenon. The predictive models adapt to the local properties of the data, providing at the same time spatially smoothed predictions. We evaluate SCLUS on several real world problems of spatial regression and spatial classification and show that it performs better than CLUS, which completely ignores the spatial context and CLUS*, which takes the spatial coordinates of the data points into account, but not their autocorrelation.

The second algorithm, NCLUS (Network Predictive Clustering System) [5] , explicitly considers autocorrelation when building single and multi-target predictive models from network data. We evaluate our approach on several real world problems of network regression, coming from the areas of social and spatial networks. Empirical results show that our algorithm performs better than a variety of other mainstream existing regression approaches (SVR, *k*-NN and M5').

The third algorithm, NHMC (Network Hierarchical Multi-label Classification) [6] , has been motivated by recent algorithms for gene function prediction where instances may belong to multiple classes and the classes are organized into a hierarchy. Thus, NHMC builds hierarchial multi-label classification models, but besides relationships among classes, it also considers the relationships among examples. Although such relationships have been identified and extensively studied in the literature, in particular as defined by protein-to-protein interaction (PPI) networks, they have not received much attention in hierarchical and multi-class gene function prediction. The results of the evaluation show that our method, which uses the hierarchical structure of classes gene/protein properties and network information, yields better performance than the methods, using each of these sources separately.

3 Conclusion

The research presented in this thesis extends the PCT framework towards learning in the presence of autocorrelation. We consider four different types of autocorrelation (spatial, temporal, spatio-temporal and network (relational)) in the development of the proposed algorithms we focus on spatial and network autocorrelation. We address the problem of learning predictive models in the case when the examples in the data are not i.i.d, such as the definition of autocorrelation measures for a variety of learning tasks, the definition of autocorrelation-based heuristics, the development of algorithms that use such heuristics for learning predictive models, as well as their experimental evaluation on real-world data.

The major contributions of this dissertation are three extensions (SCLUS, NCLUS and NHMC) of the predictive clustering approach for handling non-stationary (spatial and network) autocorrelation for different predictive modeling tasks. The algorithms are heuristic: we define new heuristic functions that take into account both the variance of the target variables and its spatial/network autocorrelation. Different combinations of these two components enable us to investigate their influence in the heuristic function and on the final predictions. We have performed extensive empirical evaluation of the newly developed methods on single target classification and regression problems, on multi-target classification and regression problems as well as tasks of hierarchial multi-label classification.

Our approaches compare very well to mainstream methods that do not consider autocorrelation, as well as to well-known methods that consider autocorrelation. Furthermore, our approaches can more successfully remove the autocorrelation of the errors of the obtained models. Finally, the obtained predictions are more coherent in space (or in the network context). We also apply the proposed predictive models to real-word problems, such as the prediction of outcrossing rates from genetically modified crops to conventional crops in ecology, prediction of the number of views of online lectures, and protein function prediction in functional genomics.

References

[1] W. Tobler (1970) A computer movie simulating urban growth in the Detroit region. *Economic Geography*, 46:234–240.

[2] J. Neville, O. Simsek, D. Jensen (2004) Autocorrelation and relational learning: Challenges and opportunities. *Proc. Wshp. Statistical Relational Learning*, ICML, Montreal, 2004, 33–50.

[3] H. Blockeel (1998) *Top-down induction of first order logical decision trees*, PhD Thesis, Katholieke Universiteit Leuven, Belgium.

[4] D. Stojanova, M. Ceci, A. Appice, D. Malerba, S. Džeroski (2013) Dealing with spatial autocorrelation in gene flow modeling, *Ecological informatics*, 13:22–39.

[5] D. Stojanova, M. Ceci, A. Appice, S. Džeroski (2012) Network regression with predictive clustering trees, *Data Mining and Knowledge Discovery*, 25(2):378–413.

[6] D. Stojanova, M. Ceci, A. Appice, D. Malerba, S. Džeroski (2012) Using PPI networks in hierarchical multi-label classification trees for gene function prediction, *Proc. 6th International Workshop on Machine Learning in Systems Biology*, Basel, Switzerland, 10.

Information Visualization using Machine Learning

Gregor Leban
Institut Jožef Stefan, Jamova cesta 39, Slovenia
E-mail: gregor.leban@ijs.si

Thesis Summary

Keywords: information visualization, machine learning, radviz, parallel coordinates, mosaic plot

Received: February 20, 2013

Data visualization is an important tool for discovering patterns in the data. Finding interesting visualizations can be however a difficult task if there are many possible ways to visualize the data. In this paper we present the VizRank method that can estimate visualization interestingness. The method can be applied on a number of visualization techniques and can automatically identify the most interesting data visualizations.

Povzetek: Predstavljena metoda VizRank omogoča avtomatsko ocenjevanje zanimivosti različnih vizualizacij podatkov in posledično identifikacijo najzanimivejših prikazov. Metodo je mogoče uporabiti na poljubni metodi s točkovnim prikazom podatkov, na metodi paralelnih koordinat ter na mozaičnih diagramih.

1 Introduction

Data visualization has a great potential for extracting knowledge from data. Visualizing the right set of features can clearly identify interesting patterns. However, not all data projections are equally interesting and the task of the data analyst is to find the most insightful ones. In case of supervised learning, we are looking for those visualizations that show clear class separation. Finding such visualizations (if they exist), can be very challenging especially when there are many possible ways to visualize the data.

In order to make the task easier we developed a method called VizRank that can be used to automatically identify the most interesting visualizations of a dataset. The method was developed to be used on all point-based visualization methods such as scatterplot, radviz, polyviz and linear projections. We later extended it also for use on parallel coordinates and mosaic plots.

In the paper we will mention two less commonly known visualization methods - radviz[1] and parallel coordinates[2]. In radviz the visualized features are represented as dots distributed along the circle. For each data example, each dot (feature) is attracting the example with a force corresponding to the value of that feature - the greater the value, the greater the attraction. The example is displayed where the sum of forces equals 0. In parallel coordinates, the axes for the visualized features are displayed parallel to each other. Each example is shown as a series of lines that intersect each axis at the point that corresponds to the value of the feature.

2 VizRank

VizRank[4] identifies the most interesting visualizations by repeating the following steps. First, a method for generating different feature subsets is used to select a set of features to be visualized and evaluated. Given the features, the positions of data points in the projection are then determined based on the chosen visualization method. A new dataset is then constructed consisting of only the x and y data point positions and their labels. A machine learning algorithm is then used on this dataset to evaluate the quality of class separation. The computed accuracy of the algorithm is used as the score of the interestingness of the projection.

Method for generating different feature subsets. The space of possible visualizations is commonly too large to evaluate all possible visualizations. To identify interesting projections fast and by checking only a small subset of possible projections we developed different heuristic methods. The one that performs best starts by ranking individual features using a feature selection method such as ReliefF[3]. From the ranked list of features we then choose the desired number of features using the gamma probability distribution. With this sampling method, the higher ranked features are selected more often. The intuition for this approach is that the features that are better at class separation will more likely generate interesting visualizations and should be tested more often than features that are worse.

Learning algorithm. Humans are able to detect arbitrarily shaped class boundaries in the visualizations. In order to best mimic humans we decided to use k-NN as the learning method. We experimented with different scoring functions such as classification accuracy and Brier score. At the end

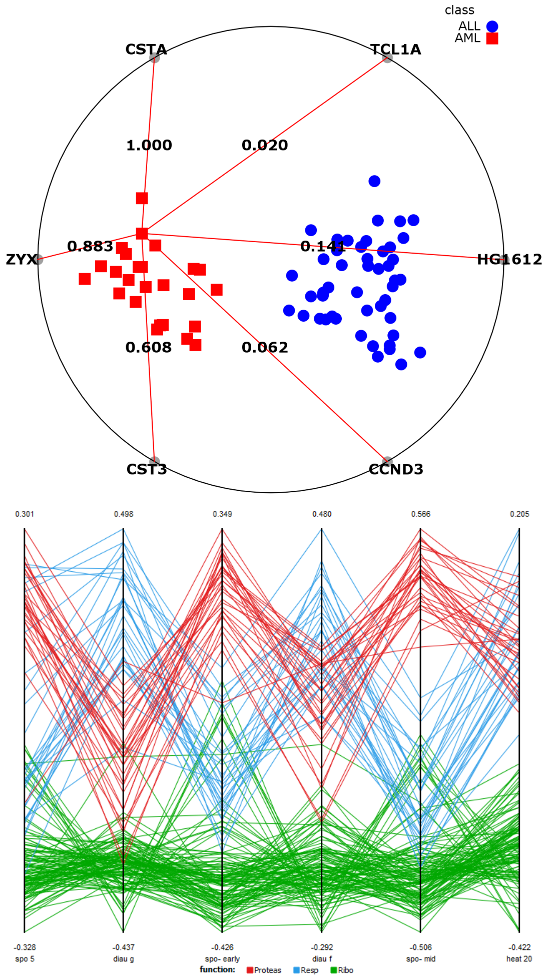


Figure 1: Two visualizations identified by VizRank - radviz visualization of *leukemia* dataset (top) and parallel coordinates plot of *yeast* dataset (bottom).

used the average probability assigned to the correct class which is defined as

$$\frac{1}{N}\sum_{i=1}^{N} P(y_i|x_i)$$

We chose this method since our experiments indicated that it produces very refined and human-like ranking of projections.

Some uses of best ranked projections. VizRank produces as a result a ranked list of projections. One possible use of this list is to perform feature scoring. In this case, the features are scored based on how often they appear in the top ranked projections. Instead of myopic measures that score each feature independently of the others, this measure can also identify features that are important when combined with other features. The list of projections can also be used for outlier detection. Frequently in top ranked projections some points lie outside of their main cluster of points. To understand if the example is really an outlier we can visualize the class prediction of the point in several top ranked projections.

Agreement with human ranking. Our base assumption in VizRank is that projections with high prediction accuracy are most insteresting for the data analysts. To evaluate how well does ranking obtained using VizRank actually correspond to ranking done by humans we performed an experiment in which 30 people ranked 20 pairs of projections. The obtained correlation between VizRank and human ranking was 0.78. To test the influence of the learning algorithm we also ranked projections using SVM and decision trees instead of k-NN. Using SVM the correlation fell to 0.58, while using decision trees it dropped to only 0.28. Results confirm that k-NN is the most appropriate of the tested methods and that ranking results highly correlate with human ranking.

Use on other visualization methods. The method, as presented, can be applied on any point based visualization method. We extended VizRank also to parallel coordinates and mosaic plots by identifying the desired properties that interesting visualizations have. In case of parallel coordinates, for example, examples from each class should be drawn under similar angle. This reduces clutter caused by intersecting lines and allows the detection of a regular pattern. By defining a corresponding optimization function we were then able to identify visualizations as the one in Figure 1.

3 Conclusion

We developed and presented the VizRank method that can automatically evaluate interestingness of different visualizations of labeled data. It is most valuable when the analysed data contains hundreds or thousands of features which makes manual search for interesting visualizations impractical. Empirical results confirm that k-NN is the most appropriate of the learning algorithms and that ranking of projections obtained with VizRank highly correlates with human ranking. The method can be applied on any point-based method as well as on parallel coordinates and mosaic plots.

References

[1] G. Grinstein, M. Trutschl, and U. Cvek. High-dimensional visualizations. *Proceedings of the Visual Data Mining Workshop, KDD*, 2001.

[2] Alfred Inselberg. *Parallel coordinates: visual multidimensional geometry and its applications*. Springer, 2009.

[3] I. Kononenko and E. Simec. Induction of decision trees using relieff. In *Mathematical and statistical methods in artificial intelligence*. Springer Verlag, 1995.

[4] G. Leban, I. Bratko, U. Petrovic, T. Curk, and B. Zupan. Vizrank: finding informative data projections in functional genomics by machine learning. *Bioinformatics*, 21(3):413–414, 2005.

JOŽEF STEFAN INSTITUTE

Jožef Stefan (1835-1893) was one of the most prominent physicists of the 19th century. Born to Slovene parents, he obtained his Ph.D. at Vienna University, where he was later Director of the Physics Institute, Vice-President of the Vienna Academy of Sciences and a member of several scientific institutions in Europe. Stefan explored many areas in hydrodynamics, optics, acoustics, electricity, magnetism and the kinetic theory of gases. Among other things, he originated the law that the total radiation from a black body is proportional to the 4th power of its absolute temperature, known as the Stefan–Boltzmann law.

The Jožef Stefan Institute (JSI) is the leading independent scientific research institution in Slovenia, covering a broad spectrum of fundamental and applied research in the fields of physics, chemistry and biochemistry, electronics and information science, nuclear science technology, energy research and environmental science.

The Jožef Stefan Institute (JSI) is a research organisation for pure and applied research in the natural sciences and technology. Both are closely interconnected in research departments composed of different task teams. Emphasis in basic research is given to the development and education of young scientists, while applied research and development serve for the transfer of advanced knowledge, contributing to the development of the national economy and society in general.

At present the Institute, with a total of about 900 staff, has 700 researchers, about 250 of whom are postgraduates, around 500 of whom have doctorates (Ph.D.), and around 200 of whom have permanent professorships or temporary teaching assignments at the Universities.

In view of its activities and status, the JSI plays the role of a national institute, complementing the role of the universities and bridging the gap between basic science and applications.

Research at the JSI includes the following major fields: physics; chemistry; electronics, informatics and computer sciences; biochemistry; ecology; reactor technology; applied mathematics. Most of the activities are more or less closely connected to information sciences, in particular computer sciences, artificial intelligence, language and speech technologies, computer-aided design, computer architectures, biocybernetics and robotics, computer automation and control, professional electronics, digital communications and networks, and applied mathematics.

The Institute is located in Ljubljana, the capital of the independent state of **Slove**nia (or S♡nia). The capital today is considered a crossroad between East, West and Mediterranean Europe, offering excellent productive capabilities and solid business opportunities, with strong international connections. Ljubljana is connected to important centers such as Prague, Budapest, Vienna, Zagreb, Milan, Rome, Monaco, Nice, Bern and Munich, all within a radius of 600 km.

From the Jožef Stefan Institute, the Technology park “Ljubljana” has been proposed as part of the national strategy for technological development to foster synergies between research and industry, to promote joint ventures between university bodies, research institutes and innovative industry, to act as an incubator for high-tech initiatives and to accelerate the development cycle of innovative products.

Part of the Institute was reorganized into several high-tech units supported by and connected within the Technology park at the Jožef Stefan Institute, established as the beginning of a regional Technology park "Ljubljana". The project was developed at a particularly historical moment, characterized by the process of state reorganisation, privatisation and private initiative. The national Technology Park is a shareholding company hosting an independent venture-capital institution.

The promoters and operational entities of the project are the Republic of Slovenia, Ministry of Higher Education, Science and Technology and the Jožef Stefan Institute. The framework of the operation also includes the University of Ljubljana, the National Institute of Chemistry, the Institute for Electronics and Vacuum Technology and the Institute for Materials and Construction Research among others. In addition, the project is supported by the Ministry of the Economy, the National Chamber of Economy and the City of Ljubljana.

Jožef Stefan Institute
Jamova 39, 1000 Ljubljana, Slovenia
Tel.:+386 1 4773 900, Fax.:+386 1 251 93 85
WWW: http://www.ijs.si
E-mail: matjaz.gams@ijs.si
Public relations: Polona Strnad

INFORMATICA

AN INTERNATIONAL JOURNAL OF COMPUTING AND INFORMATICS

INVITATION, COOPERATION

Submissions and Refereeing

Please submit a manuscript at: http://www.informatica.si/Editors/PaperUpload.asp. At least two referees outside the author's country will examine it, and they are invited to make as many remarks as possible from typing errors to global philosophical disagreements. The chosen editor will send the author the obtained reviews. If the paper is accepted, the editor will also send an email to the managing editor. The executive board will inform the author that the paper has been accepted, and the author will send the paper to the managing editor. The paper will be published within one year of receipt of email with the text in Informatica MS Word format or Informatica LaTeX format and figures in .eps format. Style and examples of papers can be obtained from http://www.informatica.si. Opinions, news, calls for conferences, calls for papers, etc. should be sent directly to the managing editor.

QUESTIONNAIRE

□ Send Informatica free of charge

□ Yes, we subscribe

Please, complete the order form and send it to Dr. Drago Torkar, Informatica, Institut Jožef Stefan, Jamova 39, 1000 Ljubljana, Slovenia. E-mail: drago.torkar@ijs.si

Since 1977, Informatica has been a major Slovenian scientific journal of computing and informatics, including telecommunications, automation and other related areas. In its 16th year (more than nineteen years ago) it became truly international, although it still remains connected to Central Europe. The basic aim of Informatica is to impose intellectual values (science, engineering) in a distributed organisation.

Informatica is a journal primarily covering intelligent systems in the European computer science, informatics and cognitive community; scientific and educational as well as technical, commercial and industrial. Its basic aim is to enhance communications between different European structures on the basis of equal rights and international refereeing. It publishes scientific papers accepted by at least two referees outside the author's country. In addition, it contains information about conferences, opinions, critical examinations of existing publications and news. Finally, major practical achievements and innovations in the computer and information industry are presented through commercial publications as well as through independent evaluations.

Editing and refereeing are distributed. Each editor can conduct the refereeing process by appointing two new referees or referees from the Board of Referees or Editorial Board. Referees should not be from the author's country. If new referees are appointed, their names will appear in the Refereeing Board.

Informatica is free of charge for major scientific, educational and governmental institutions. Others should subscribe (see the last page of Informatica).

ORDER FORM – INFORMATICA

Name: ....

Title and Profession (optional): ....

....

Home Address and Telephone (optional): ....

....

Office Address and Telephone (optional): ....

....

E-mail Address (optional): ....

Signature and Date: ....

Informatica WWW:

http://www.informatica.si/

Referees from 2008 on:

Informatica

An International Journal of Computing and Informatics

Web edition of Informatica may be accessed at: http://www.informatica.si.

Subscription Information Informatica (ISSN 0350-5596) is published four times a year in Spring, Summer, Autumn, and Winter (4 issues per year) by the Slovene Society Informatika, Litostrojska cesta 54, 1000 Ljubljana, Slovenia.
The subscription rate for 2013 (Volume 37) is
– 60 EUR for institutions,
– 30 EUR for individuals, and
– 15 EUR for students
Claims for missing issues will be honored free of charge within six months after the publication date of the issue.

Typesetting: Borut Žnidar.
Printing: ABO grafika d.o.o., Ob železnici 16, 1000 Ljubljana.

Orders may be placed by email (drago.torkar@ijs.si), telephone (+386 1 477 3900) or fax (+386 1 251 93 85). The payment should be made to our bank account no.: 02083-0013014662 at NLB d.d., 1520 Ljubljana, Trg republike 2, Slovenija, IBAN no.: SI56020830013014662, SWIFT Code: LJBASI2X.

Informatica is published by Slovene Society Informatika (president Niko Schlamberger) in cooperation with the following societies (and contact persons):
Robotics Society of Slovenia (Jadran Lenarčič)
Slovene Society for Pattern Recognition (Janez Perš)
Slovenian Artificial Intelligence Society (Dunja Mladenić)
Cognitive Science Society (Urban Kordeš)
Slovenian Society of Mathematicians, Physicists and Astronomers (Andrej Likar)
Automatic Control Society of Slovenia (Sašo Blažič)
Slovenian Association of Technical and Natural Sciences / Engineering Academy of Slovenia (Vojteh Leskovšek)
ACM Slovenia (Andrej Brodnik)

Informatica is surveyed by: ACM Digital Library, Citeseer, COBISS, Compendex, Computer & Information Systems Abstracts, Computer Database, Computer Science Index, Current Mathematical Publications, DBLP Computer Science Bibliography, Directory of Open Access Journals, InfoTrac OneFile, Inspec, Linguistic and Language Behaviour Abstracts, Mathematical Reviews, MatSciNet, MatSci on SilverPlatter, Scopus, Zentralblatt Math

Volume 37 Number 1 March 2013 **ISSN 0350-5596**

Informatica

An International Journal of Computing and Informatics

Informatica **37** (2013) Number 1, pp. 1–111