Journal of Machine Learning Research 20 (2019) 1-50 Submitted 2/17; Revised 12/18; Published 2/19

NetSDM: Semantic Data Mining with Network Analysis

Jan Kralj JAN.KRALJ@IJS.SI
Jožef Stefan Institute, Department of Knowledge Technologies,
Jamova 39, 1000 Ljubljana, Slovenia

Marko Robnik-Šikonja MARKO.ROBNIK@FRI.UNI-LJ.SI
University of Ljubljana, Faculty of Computer and Information Science,
Večna pot 113, 1000 Ljubljana, Slovenia

Nada Lavrač NADA.LAVRAC@IJS.SI
Jožef Stefan Institute, Department of Knowledge Technologies,
Jamova 39, 1000 Ljubljana, Slovenia

Editor: Bert Huang

Abstract

Semantic data mining (SDM) is a form of relational data mining that uses annotated data together with complex semantic background knowledge to learn rules that can be easily interpreted. The drawback of SDM is a high computational complexity of existing SDM algorithms, resulting in long run times even when applied to relatively small data sets. This paper proposes an effective SDM approach, named NetSDM, which first transforms the available semantic background knowledge into a network format, followed by network analysis based node ranking and pruning to significantly reduce the size of the original background knowledge. The experimental evaluation of the NetSDM methodology on acute lymphoblastic leukemia and breast cancer data demonstrates that NetSDM achieves radical time efficiency improvements and that learned rules are comparable or better than the rules obtained by the original SDM algorithms.

Keywords: data mining, semantic data mining, ontologies, subgroup discovery, network analysis

1. Introduction

In data mining applications, relevant information can be scattered across disparate resources in heterogeneous data formats. Extraction of knowledge from large heterogeneous data is technically challenging and time consuming, and presents a significant obstacle to wider adoption of data mining in real-life applications. While most of the data mining algorithms (Witten and Frank, 2005) work only with tabular data, *relational data mining* (RDM) (Džeroski and Lavrač, 2001) and *relational learning* (De Raedt, 2008) algorithms can use additional relational information about the analyzed data to build richer and more accurate predictive models. One such form of additional information is domain knowledge (relational background knowledge) about instances of a data set. As shown by Page et al. (2012) and Peissig et al. (2014), relational data mining can significantly outperform classical machine learning approaches.

Previous work on RDM has shown that the use of *background knowledge* is vital for data analysis in many domains. However, the issue with using more complex data (i.e.,

data along with relations and background knowledge) is that it can no longer be easily represented in a tabular format (also referred to as a design matrix (Rendle, 2013)). Better data representations are needed when instances under examination are interconnected to a various (non-fixed) number of instances. Representing each connection from an instance as a separate column would result in a different number of columns for each row. Alternatively, if we encoded connections of an instance with a single column, columns would have to contain composite data structures such as lists. In RDM, this problem is addressed by representing data sets with *multi-relational databases* and by applying specialized relational data mining algorithms for data analysis (Džeroski and Lavrač, 2001). Examples of such instances are genes that are connected through mutual activation, or research papers connected through citations.

An alternative way to describe a data set containing inter-connected instances is to represent it as a *network* (Burt and Minor, 1983), a data structure containing nodes, information about the nodes, and connections between the nodes. In mathematical terms, such a structure is described as a *graph*, where the nodes are referred to as vertices, and the connections as edges. The field of *network analysis* deals with several types of networks. In social networks, nodes represent people and connections represent their social links. Biological networks include gene regulatory networks, protein interaction networks, drug interaction networks, etc. In information networks, directed connections encode information flow between the network nodes. While most researchers deal with *homogeneous information networks*, where all nodes and edges are of the same node/edge type, Sun and Han (2012) addressed the problem of *heterogeneous information network* analysis, where nodes and edges belong to different node or edge types. For example, we may have a network containing both the genes and the proteins they encode, which necessitates the use of two node types to represent the data. Sun and Han (2012) introduced the concept of *authority ranking* for heterogeneous information networks, where the impact of a node is transferred along the edges to simultaneously rank nodes of different types, while class labels can also be propagated through the network (Vanunu et al., 2010).

Network analysis can be considered as a form of relational learning, where instances are linked by relations and connected in a complex graph. Special form of relational data are *ontologies* (Guarino et al., 2009). The challenge of incorporating domain ontologies in the data mining process has been addressed in the work on *semantic data mining* (SDM) (Lavrač and Vavpetič, 2015). Semantic data mining can discover complex rules describing subgroups of data instances that are connected to terms (annotations) of an ontology, where the ontology is referred to as background knowledge used in the learning process. An example SDM problem is to find subgroups of enriched genes in a biological experiment, where background knowledge is the Gene Ontology (Ashburner et al., 2000).

Given that SDM algorithms are relatively slow, the size of the background knowledge used by SDM approaches is usually several orders of magnitude lower than the problems typically handled by network analysis approaches. Take for example SDM algorithm Hedwig (Vavpetič et al., 2013), which is one of the semantic data mining algorithm used in this work. Hedwig performs beam search to explore the space of all possible explanations to find the best rules explaining the data. The search space it explores is very large and even using an adequate heuristic to guide the search, the algorithm takes a long time to find relevant patterns in real-life data annotated by ontological background knowledge. As illustration,

take a SDM application where Hedwig was applied to a data set of 337 examples and a background knowledge of 21,062 interconnected ontology terms (Vavpetič et al., 2013), which is small compared to typical network analysis applications that deal with much larger data sets, possibly composed of e.g., hundred millions network nodes.

Despite large differences in the current sizes of data sets analyzed by SDM and network analysis approaches, the two research fields are aligned in terms of the research question of interest, which can be posed as follows: *Which part of the network structure is the most important for the analyst's current query?* The challenge addressed in this work is the reduction of the search space of SDM algorithms, which can be achieved by using network analysis approaches. To this end, we have developed an approach that is capable of utilizing network analysis algorithms Personalized PageRank (Page et al., 1999) and node2vec (Grover and Leskovec, 2016) to improve the efficiency of SDM. We show that the proposed approach, named NetSDM, can efficiently generate high quality rules by investigating only a fraction of the entire search space imposed by the background knowledge considered.

The rest of the paper is structured as follows. Section 2 presents the related work, as well as the technologies used; we first introduce the semantic data mining algorithms used, Hedwig and Aleph (Srinivasan, 1999), and then present the two network analysis algorithms, Personalized PageRank and node2vec. Section 3 presents the proposed NetSDM approach that exploits network analysis to reduce the size of the background knowledge used by SDM algorithms. Section 4 presents the experimental setup, and Section 5 presents the experimental results. Section 6 concludes the paper and presents plans for further work.

2. Related work and background technologies

This section presents the related work in the fields of semantic data mining and network analysis. It starts with the related work in the field of semantic data mining in Section 2.1, which presents also the two algorithms used in our experiments: Hedwig and Aleph. It continues with the related work in the field of network analysis in Section 2.2, and includes the two algorithms used in our experiments: Personalized PageRank and node2vec.

2.1. Semantic data mining

To find patterns in data annotated with ontologies, we rely on semantic data mining (SDM) (Dou et al., 2015; Lavrač and Vavpetič, 2015). In SDM, the input is composed of a set of class labeled instances and the background knowledge encoded in the form of ontologies, and the goal is to find descriptions of target class instances as a set of rules of the form `TargetClass` ← `Explanation`, where the explanation is a logical conjunction of terms from the ontology. Semantic data mining has its roots in symbolic rule learning, subgroup discovery and enrichment analysis research, briefly explained below.

2.1.1. Rule learning and subgroup discovery

One of the established techniques for data mining (Piatetsky-Shapiro, 1991) is symbolic rule learning (Fürnkranz et al., 2012). While rule learning was initially focused on learning predictive models in the form of classification rules, there is also a substantial research in descriptive rule learning, including association rule learning (Agrawal and Srikant, 1994), that

aims at finding interesting descriptive patterns in the unsupervised as well as in supervised learning settings (Liu et al., 1998).

Building on classification and association rule learning, subgroup discovery techniques aim at finding interesting patterns as sets of rules that best describe the target class (Klösgen, 1996; Wrobel, 1997). Typical output of subgroup discovery algorithms are rules where the rule condition (`Explanation`) is a conjunction of features (attribute values) that characterize the target class instances covered by the rule, and each rule describes a particular interesting subgroup of target class instances.

2.1.2. Using ontologies in enrichment analysis

Enrichment analysis (EA) techniques are statistical methods used to identify putative explanations for a set of entities based on over- or under-representation of their attribute values, which can be referred to as *differential expression*. In life sciences, EA is widely used with the Gene Ontology (GO) (Ashburner et al., 2000) to profile the biological role of genes, such as differentially expressed cancer genes in microarray experiments (Tipney and Hunter, 2010).

While standard EA provides explanations in terms of concepts from a single ontology, researchers are increasingly combining several ontologies and data sets to uncover novel associations. The ability to detect patterns in data sets that do not use only the GO can yield valuable insights into diseases and their treatment. For instance, it was shown that Werner's syndrome, Cockayne syndrome, Burkitt's lymphoma, and Rothmund-Thomson syndrome are all associated with aging related genes (Puzianowska-Kuznicka and Kuznicki, 2005; Cox and Faragher, 2007). On the clinical side, EA can be used to learn adverse events from Electronic Health Record (EHR) data such as increased comorbidities in rheumatoid arthritis patients (LePendu et al., 2013), or to identify phenotypic signatures of neuropsychiatric disorders (Lyalina et al., 2013). An ontology-based EA approach was used to identify genes linked to aging in worms (Callahan et al., 2015), aberrant pathways—the network drivers in HIV infection identifying HIV inhibitors strongly associated with mood disorders (Hoehndorf et al., 2012), or to learn a combination of molecular functions and chromosome positions from lymphoma gene expression (Jiline et al., 2011).

2.1.3. Using ontologies in rule learning

An abundance of taxonomies and ontologies that are readily available can provide higher-level descriptors and explanations of discovered subgroups. In the domain of systems biology the Gene Ontology (Ashburner et al., 2000), KEGG orthology (Ogata et al., 1999) and Entrez gene–gene interaction data (Maglott et al., 2005) are examples of structured domain knowledge. In rule learning, the terms in the nodes of these ontologies can take the role of additional high-level descriptors (generalizations of data instances) used in the induced rules, while the hierarchical relations among the terms can be used to guide the search for conjuncts of the induced rules.

The SEGS algorithm (Trajkovski et al., 2008a) was the first to combine enrichment analysis and machine learning research in the construction of rules explaining gene expression data. SEGS constructs rules where the explanation is a logical conjunction of terms from several ontologies, explaining sets of differentially expressed (target class) genes as

combinations of Gene Ontology (GO) terms, KEGG orthology terms, and terms describing gene–gene interactions obtained from the Entrez database.

The challenge of incorporating domain ontologies in data mining was addressed in SDM research by several other authors. Žáková et al. (2006) used an engineering ontology of Computer-Aided Design (CAD) elements and structures as a background knowledge to extract frequent product design patterns in CAD repositories and to discover predictive rules from CAD data. Using ontologies, the algorithm Fr–ONT for mining frequent concepts was introduced by Lawrynowicz and Potoniec (2011).

Vavpetič and Lavrač (2013) developed a SDM toolkit that includes two semantic data mining systems: SDM-SEGS and SDM-Aleph. SDM-SEGS is an extension of the earlier domain-specific algorithm SEGS (Trajkovski et al., 2008a), which supports semantic subgroup discovery in gene expression data. SDM-SEGS extends and generalizes this approach by allowing the user to input a set of ontologies in the OWL ontology specification language and an empirical data set annotated with domain ontology terms. SDM-SEGS employs a predefined selection of ontologies to constrain and guide a top-down search of the hierarchically structured hypothesis space. SDM-Aleph is an extension of the Inductive Logic Programming system Aleph (Srinivasan, 1999), which does not have the limitations of SDM-SEGS imposed by domain-specific properties of algorithm SEGS, and can accept any number of ontologies as domain background knowledge, including different relations.

In summary, semantic subgroup discovery algorithms SDM-SEGS and SDM-Aleph are either specialized for a specific domain (Trajkovski et al., 2008b) or adapted from systems that do not take into account the hierarchical structure of background knowledge (Vavpetič and Lavrač, 2013), respectively. Therefore in this work, we rather use the Hedwig algorithm (Vavpetič et al., 2013) and the original Aleph algorithm (Srinivasan, 1999), which can both accept any number of input ontologies, where Aleph can also use different types of relations.

2.1.4. Hedwig

To illustrate the rules induced by the Hedwig subgroup discovery algorithm (Vavpetič et al., 2013), take as an example a task of analyzing differential expression of genes in breast cancer patients, originally addressed by Sotiriou et al. (2006). In this task, the `TargetClass` of the generated rules is the class of genes that are differentially expressed in breast cancer patients as compared to the general population.

Rule rank	Explanation
1	chromosome ∧ cell cycle
2	cellular macromolecule metabolic process ∧ intracellular non-membrane-bounded organelle ∧ cell cycle
3	cell division ∧ nucleus ∧ cell cycle
4	regulation of mitotic cell cycle ∧ cytoskeletal part
5	regulation of mitotic cell cycle ∧ microtubule cytoskeleton
6	regulation of G2/M transition of mitotic cell cycle
7	regulation of cell cycle process ∧ chromosomal part
8	regulation of cell cycle process ∧ spindle
9	enzyme binding ∧ regulation of cell cycle process ∧ intracellular non-membrane-bounded organelle
10	ATP binding ∧ mitotic cell cycle ∧ nucleus

Figure 1: An example output of the Hedwig semantic subgroup discovery algorithm, where `Explanations` represent subgroups of genes, differentially expressed in patients with breast cancer.

The Hedwig algorithm generated ten rules (subgroup descriptions), each describing a subgroup of differentially expressed genes. In the rules, the `TargetClass` is the set of differentially expressed genes and the `Explanation` is a conjunction of Gene Ontology terms and covers the set of genes annotated by the terms in the conjunction. The resulting set of `Explanations` (rule conditions) is shown in Figure 1.

Take the first ranked rule, where the `Explanation` is a conjunction of biological concepts `chromosome` $\wedge$ `cell cycle`, which covers all the genes that are covered by both ontology terms. Take as another example the fourth best ranked rule, which explains that the regulation of the mitotic cell cycle and cytoskeletal formation explain the breast cancer based on gene expression.

The semantic subgroup discovery task addressed by Hedwig takes three types of inputs: the training examples, the domain knowledge, and a mapping between the two.

- Data set, composed of training examples expressed as RDF (Resource Description Framework) triples in the form subject-predicate-object, e.g., geneX–suppresses–geneY. Data set S is split into a set of positive examples S_+, i.e. 'interesting' target class instances (for example, genes enriched in a particular biological experiment), and a set of negative examples S_- of non-target class instances (e.g., non-enriched genes).
- Domain knowledge, composed of domain ontologies in RDF form.
- Object-to-ontology mapping, which associates each RDF triple with an appropriate ontological concept. We refer to these object-to-ontology mappings as *annotations*, meaning that object x is *annotated by* ontology term o if the pair (x, o) appears in the mapping.

For given inputs, the output of Hedwig is a set of descriptive patterns in the form of rules, where rule conditions are conjunctions of domain ontology terms that explain a group of target class instances. Ideally, Hedwig discovers explanations that best describe and cover as many target class instances and as few non-target instances as possible.

The Hedwig algorithm uses beam search, where the beam contains the best N rules. It starts with the default rule that covers all the training examples. In every search iteration, each rule from the beam is specialized via one of the four operations: (i) replace the predicate of a rule with a predicate that is a sub-class of the previous one, (ii) negate a predicate of a rule, (iii) append a new unary predicate to the rule, or (iv) append a new binary predicate, introducing a new existentially quantified variable, where the new variable has to be 'consumed' by a literal that has to be added as a conjunction to this clause in the next step of rule refinement.

Hedwig learns rules through a sequence of specialization steps. Each step either maintains or reduces the current number of covered examples. A rule will not be specialized once its coverage is zero or falls below some predetermined threshold. When adding a new conjunct, the algorithm checks if the specialized rule improves the probability of rule consequent—to this end the redundancy coefficient is used (Hämäläinen, 2010); if not, the specialized rule is not added to the list of specializations. After the specialization step is applied to each rule in the beam, a new set of best scoring N rules is selected. If no improvement is made to the rule collection, the search terminates. In principle, the procedure supports any rule scoring function. Numerous rule scoring functions for discrete targets are available: χ^2, Precision, WRAcc (Lavrač et al., 2004), Leverage, and Lift. Note that Lift is the default measure in Hedwig (see Section 2.1.6) that was used in our experiments.

In addition to an illustrative financial use case (Vavpetič et al., 2013), Hedwig was shown to perform well in breast cancer data analysis (Vavpetič et al., 2014), as well as in a biological setting when analyzing DNA aberration data for various cancer types (Adhikari

et al., 2016), where it was a part of three-step methodology, together with mixture models and banded matrices using several ontologies obtained from various sources: hierarchical structure of multiresolution data, chromosomal location of fragile sites, virus integration sites, cancer genes, and amplification hotspots.

2.1.5. Aleph

Aleph (A Learning Engine for Proposing Hypotheses) (Srinivasan, 1999) is a general purpose system for Inductive Logic Programming (ILP) that was conceived as a workbench for implementing and testing concepts and procedures from a variety of different ILP and relational learning systems and papers. The system can construct rules, trees, constraints and features; invent abnormality predicates; perform classification, regression, clustering, and association rule learning; allows for choosing between different search strategies (general-to-specific or specific-to-general, i.e. top-down or bottom-up, bidirectional, etc.), search algorithms (hill-climbing, exhaustive search, stochastic search, etc.) and evaluation functions. It allows users to specify their own search procedure, proof strategy, and visualization of hypotheses.

Similarly to Hedwig, Aleph accepts as input a set of positively and negatively labeled training examples and a file describing the background knowledge, including a mapping between the examples and the background knowledge. The training examples are expressed as Prolog facts, and the background knowledge is in the form of Prolog facts and clauses. In contrast to Hedwig that was used in our experiments for semantic subgroup discovery, we used Aleph in its default operation mode to learn a classifier (a Theory) composed of as a set of classification rules, which also explain the target class examples.

Aleph constructs rules in several steps. It first selects a positive example and builds the most specific rule explaining the example following the steps described by Muggleton (1995). Next, the algorithm searches for a more general rule, performed with a branch-and-bound search strategy, searching first through generalizations of the rule containing the fewest terms. Finally, the rule with the best score is added to the current theory, and all the examples covered by this rule are removed (this step is sometimes called the "cover removal" step). This procedure is repeated until all the positive examples are covered.

Aleph was incorporated into the SDM-toolkit (Vavpetič and Lavrač, 2013), which allows Aleph (named SDM-Aleph) to use the same input data as Hedwig. However, a drawback of using SDM-Aleph is that it can reason only over one type of background knowledge relations, while the original Aleph algorithm can reason over any type of relations. In the case of Gene Ontology, which was used as the background knowledge in our experiments, this generality allows Aleph to encode both the *is_a* and *is_part_of* relations, which are the most frequent relations in ontologies, as well as the relationships composed of these two relations.

2.1.6. Rule quality measures

We use several measures to evaluate the quality of rules discovered by Hedwig and Aleph. Take for example rule R with a condition that is true for 80 positive instances (*True Positives*, TP) and 20 negative instances (*False Positives*, FP) from the set S of 100 positive instances (*Positives*) and 100 negative instances (*Negatives*). Coverage(R) $= TP + FP =$

100 is the total number of instances covered by the rule. Support of the rule is calculated as follows: $\text{Support}(R) = \frac{TP+FP}{|S|} = \frac{\text{Coverage(R)}}{|S|} = 0.5$. $\text{Precision}(R) = 0.8$ is calculated as $\frac{TP}{TP+FP}$, and $\text{Accuracy}(R) = 0.8$ is calculated as $\frac{TP+TN}{|S|}$, where TN denotes the number of *True Negatives*.

In our experiments we use the Lift metric to evaluate the performance of SDM algorithms Hedwig and Aleph. $\text{Lift}(R)$ is defined as the ratio between the precision of a rule and the proportion of positive examples in the data set (i.e. the precision of the empty default rule that classifies all examples as positive). In our example $\text{Lift}(R) = \frac{0.8}{0.5} = 1.6$, which is calculated using the following formula:

$$\text{Lift(R)} = \frac{\text{Precision}(R)}{PR} \tag{1}$$

where $PR = \frac{|Positives|}{|S|}$ is the ratio of positive instances in the data set. The range of values Lift can take is from 0 to ∞ and larger values indicate better rules.

2.2. Network analysis

Network analysis uses concepts from graph theory to investigate characteristics of networked structures in terms of nodes (such as individual actors, people, or any other objects in the network) and edges or links that connect them (relationships, interactions or ties between the objects). This section focuses on the algorithms which we used or adapted to rank nodes in information networks.

2.2.1. Node ranking: Using Personalized PageRank algorithm

The objective of node ranking in information networks is to assess the relevance of a given node either globally (with regard to the whole graph) or locally (relative to some other node or group of nodes in the graph). A network node ranking algorithm assigns a *score* (or a *rank*) to each node in the network, with the goal of ranking the nodes in terms of their relevance.

A well known node ranking method is PageRank (Page et al., 1999), which was used in the Google search engine. Other methods for node ranking include Weighted PageRank method (Xing and Ghorbani, 2004), SimRank (Jeh and Widom, 2002), diffusion kernels (Kondor and Lafferty, 2002), hubs and authorities (Kleinberg, 1999), and spreading activation (Crestani, 1997). More recent network node ranking methods include PL-ranking (Zhang et al., 2016) and NCDawareRank (Nikolakopoulos and Garofalakis, 2013). Another way to rank nodes in a network is to use network centrality measures, such as Freeman's network centrality (Freeman, 1979), betweenness centrality (Freeman, 1977), closeness centrality (Bavelas, 1950), and Katz centrality (Katz, 1953).

In this paper, we are interested in network node ranking methods that can be 'localized', i.e. which do not compute the global importance/score of a node, but rather the score of a node in the context of a given subset of other nodes. An example of this type of network node ranking approaches is the Personalized PageRank (P-PR) algorithm (Page et al., 1999), sometimes referred to as random walk with restart (Tong et al., 2006). P-PR calculates the node score locally to a given network node—it is hence especially interesting for our work because it calculates the importance of network nodes relative to a given set of

starting nodes, in our case, those representing the positive examples. In the context of data set annotation with ontologies, this allows us to estimate the importance of background knowledge terms in relation to the positive examples in the data set.

Personalized PageRank uses a random walk approach to calculate the significance of nodes in an information network. Given a set of 'starting' nodes A, the Personalized PageRank calculated for A (denoted P-PR$_A$) is defined as the stationary distribution of random walker positions, where the walk starts in a randomly chosen member of A and then at each step either selects one of the outgoing connections or teleports back to a randomly selected member of A. Probability p of continuing the walk is a parameter of the Personalized PageRank algorithm and is usually set to 0.85. In our context, the P-PR algorithm is used to calculate the importance of network nodes *with respect to* a given set of starting nodes.

Remark 1 *P-PR$_A$ is a vector, and each value of the vector corresponds to some network node u. If u is the i-th node of the network, notation P-PR$_A(u)$ is used (rather than P-PR$_{Ai}$) to denote the i-th value of P-PR$_A$.*

2.2.2. Network embedding: Using node2vec algorithm

Network embedding is a mechanism for converting networks into a tabular/matrix representation format, where each network node is represented as a vector of some predefined fixed length k, and a set of nodes is represented as a table with k columns. The goal of such a conversion is to preserve structural properties of a network, which is achieved by preserving node similarity in both representations, i.e. node similarity is converted into vector similarity.

Network nodes vectorization can be effectively performed by the node2vec algorithm (Grover and Leskovec, 2016), which uses a random walk approach to calculate features that express the similarities between node pairs. The node2vec algorithm takes as input a network of n nodes, represented as a graph $G = (V, E)$, where V is the set of nodes and E is the set of edges in the network. For a user-defined number of columns k, the algorithm returns a matrix $f^* \in \mathbb{R}^{|V| \times k}$, defined as follows:

$$f^* = \text{node2vec}(G) = \underset{f \in \mathbb{R}^{|V| \times k}}{\text{argmax}} \sum_{u \in V} \left(-\log(Z_u) + \sum_{n \in N(u)} f(n) \cdot f(u) \right) \tag{2}$$

where $N(u)$ denotes the network neighborhood of node u (to be defined below).

Remark 2 *Each matrix f is a collection of k-dimensional feature vectors, with the i-th row of the matrix corresponding to the feature vector of the i-th node in the network. We write $f(u)$ to denote the row of matrix f corresponding to node u.*

The goal of the node2vec algorithm is to construct feature vectors $f(u)$ in such a way that feature vectors of nodes that share a certain neighborhood will be similar. The inner sum of the value, maximized in Equation 2, calculates the similarities between node u and all nodes in its neighborhood: it is large if the feature vectors of nodes in the same neighborhood are collinear, however it also increases if feature vectors of nodes have a large

norm. Next, value Z_u calculates the similarities between node u and all the nodes in the network as follows:

$$Z_u = \sum_{v \in V} e^{f(u) \cdot f(v)}$$

Note that value of $-\log(Z_u)$ decreases when the norms of feature vectors $f(v)$ increase, thereby penalizing collections of feature vectors with large norms.

Equation 2 has a probabilistic interpretation that models a process of randomly selecting nodes from the network (Grover and Leskovec, 2016). In this process, probability $P(n|u)$ of node n following node u in the selection process is proportional to $e^{f(n) \cdot f(u)}$. Assuming that selecting a node is independent from selecting any other node, we can calculate the probability of selecting all nodes from a given set N as $P(N|u) = \prod_{n \in N} P(n|u)$, and Equation 2 can then be rewritten as follows:

$$f^* = \text{node2vec}(G) = \underset{f \in \mathbb{R}^{|V| \times k}}{\text{argmax}} \sum_{u \in V} \log \left(P(N(u)|f(u)) \right) \tag{3}$$

Finally, let us explain neighborhood $N(u)$ in Equations 2 and 3, calculated by simulating a random walker traversing the network starting at node u. Unlike the PageRank random walker, the transition probabilities for traversing from node n_1 to node n_2 depend on node n_0 that the walker visited before node n_1, making the process of traversing the network a second order random walk. The non-normalized transition probabilities are set using two parameters, p and q, and are equal to:

$$P(n_2|\text{moved from node } n_0 \text{ to } n_1 \text{ in previous step}) = \begin{cases} \frac{1}{p} & \text{if } n_2 = n_0 \\ 1 & \text{if } n_2 \text{ can be reached from } n_1 \\ \frac{1}{q} & \text{otherwise} \end{cases}$$

The parameters of the expression are referred to as the *return* parameter p and the *in-out* parameter q. A low value of the return parameter p means that the random walker is more likely to backtrack its steps and the random walk will be closer to a breadth first search. On the other hand, a low value of parameter q encourages the walker to move away from the starting node and the random walk resembles a depth first search of the network. To calculate the maximizing matrix f^*, a set of random walks of limited size is simulated starting from each node in the network to generate several samples of sets $N(u)$.

The function maximizing Equation 2 is calculated using the stochastic gradient descent. The matrix of feature vectors node2vec(G) in Equation 2 is estimated at each generated sampling of neighborhoods $N(u)$ for all nodes in the network to discover matrix f^* that maximizes the expression for the simulated neighborhood set.

3. NetSDM methodology: Combining SDM with network analysis

SDM algorithms perform well on relatively small real world data sets, but even for small data sets the algorithms search through a very large space of possible patterns to find the 'best' pattern. The more conjuncts we allow in rule conditions and the larger the background knowledge, the larger the search space. This section presents the proposed

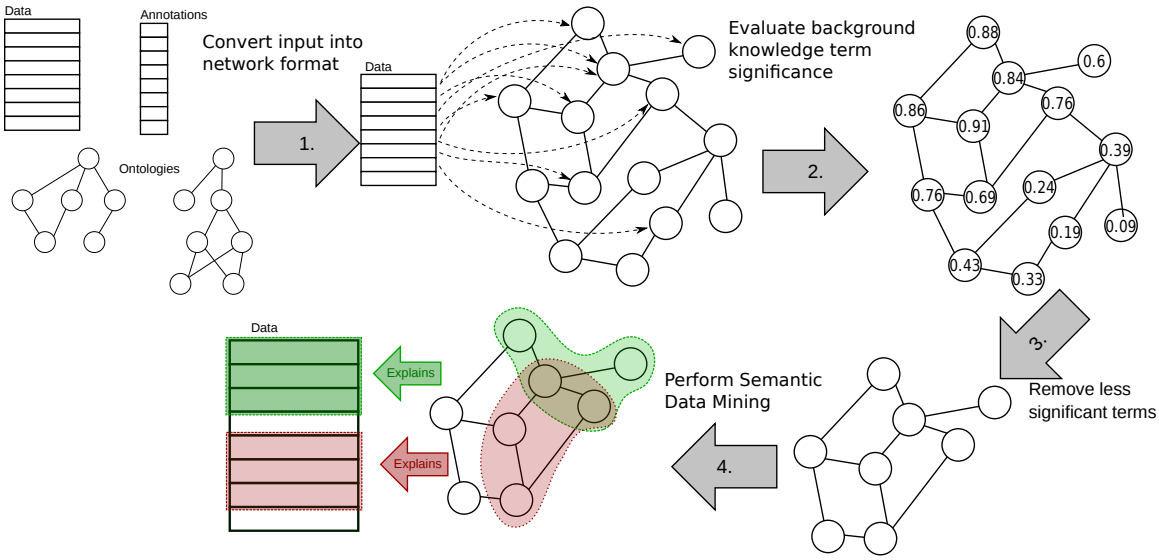

Figure 2: Illustrative outline of the proposed NetSDM methodology.

NetSDM methodology that combines semantic data mining with network analysis to reduce the search space of SDM algorithms. This section starts with an overview of the proposed methodology in Section 3.1, illustrated by an example in Section 3.2. Section 3.3 provides a detailed step-by-step description of the NetSDM methodology.

3.1. Methodology outline

SDM algorithms use heuristic search when mining for patterns in the data to limit the search to only the most promising parts of the search space. Constraining the search is necessary, as traversing the entire search space—consisting of conjuncts of logical expressions containing ontological expressions—is computationally unfeasible. However, heuristic search can produce poor results in some cases. For example, when searching for patterns with Hedwig using beam search, the beam may be too narrow, and consequently a branch that would lead to a high quality solution could be discarded early on; in large search spaces even wide beams can be quickly filled with terms that do not lead to good final solutions. Similarly, the search algorithm in Aleph can miss important patterns if all their sub-patterns exhibit low quality. In this paper we propose a new methodology, which aims to solve this problem. The proposed NetSDM methodology, which is illustrated in Figure 2, consists of the following four steps:

1. Convert a background knowledge ontology into a network format.
2. Estimate the importance of background knowledge terms using a scoring function.
3. Shrink the background knowledge, keeping only a proportion c of the top ranking terms.
4. Apply a semantic data mining algorithm on the original data set and the reduced background knowledge.

Input: SDM algorithm, network conversion method, scoring function, term removal method, background knowledge ontology O, set of examples S annotated with terms from O
Output: set of rules discovered by SDM algorithm
Parameters: shrinkage coefficient $c \in [0, 1]$

```
Convert ontology O into network format G_e (see Section 3.3.1)
Calculate significance scores score(t), for all terms t ∈ T(O) (see Section 3.3.2)
for term t ∈ T(O) do
    if RelativeRank(t) > c then
        mark term t for removal.
    end
end
form G_e' by removing marked terms from G_e (see Section 3.3.3)
convert G_e' to O'
Run SDM algorithm on S using reduced background knowledge O'
return Rules discovered by SDM algorithm
```

Algorithm 1: The NetSDM algorithm, implementing the proposed approach to semantic data mining with network node ranking and ontology shrinking.

The NetSDM methodology improves the efficiency of SDM algorithms by using a shrinking step in which we filter out background knowledge terms that are not likely to appear in significant rules. The methodology is implemented in Algorithm 1. A scoring function used in ontology shrinkage should (i) be able to evaluate the significance of terms based on the data, and (ii) be efficiently computed.
The input for the scoring function is the data used in SDM algorithms, consisting of:

- Set of instances S consisting of target (S_+) and non-target (S_-) instances,
- Ontology O represented as a set of RDF triples (*subject, predicate, object*) (to simplify the notation, for each term t appearing as either a subject or object in a triple, we will write $t \in O$ to denote that t is a term of the ontology O),
- Set of annotations $\mathcal{A}$ connecting instances in $S = S_+ \cup S_-$ with terms in O (annotations are presented as triples $(s, \textit{annotated-with}, t)$, where $s \in S$ and $t \in O$ denoting that data instance s is annotated by an ontology term t.

The output of the scoring function is a vector which, for each term in the background knowledge, contains a score estimating its significance. In other words, using $\mathcal{T}(O)$ to denote the set of all terms appearing (either as subjects or objects) in ontology O, the scoring function is defined as:

$$\text{score}_{S,O,\mathcal{A}} : \mathcal{T}(O) \to [0, \infty)$$

After defining a suitable scoring function *score* (in the subsequent subsections, we examine two possibilities), we use the computed scores to shrink the background knowledge ontology

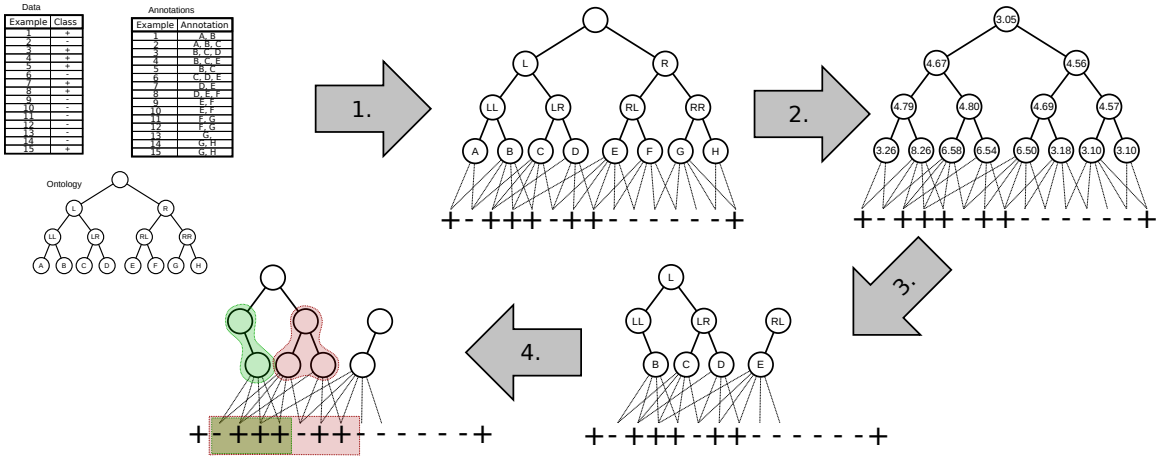

Figure 3: Illustration of using Algorithm 1 on example from Table 1.

O. Using a scoring function, a higher score of term $t \in O$ means that term t is more likely to be used in rules describing the positive examples. For every term $t \in \mathcal{T}(O)$, we use the results of the scoring function to assign *relative ranks* to ontology terms defined as

$$\text{RelativeRank}(t) = \frac{\text{Rank}(t)}{|\mathcal{T}(O)|} = \frac{|\{t' \in \mathcal{T}(O) : \text{score}(t') \geq \text{score}(t)\}|}{|\mathcal{T}(O)|}$$

which calculates the ratio of terms in the ontology that score higher than t, divided by the number of all terms. For example, a value $\text{RelativeRank}(t) = 0.05$ means that only 5% of all terms in the ontology score higher than t using a given scoring function.

3.2. Illustrative example

As an illustrative example consider the data set described in Table 1. The data set consists of 15 instances, 7 of which belong to the target class (positive examples, marked with + in Table 1). The instances are annotated by one or more terms in a hierarchy (i.e., simple ontology) consisting of 8 base nodes and 7 higher-level terms, shown on the left-hand side of Figure 3. Using the Personalized PageRank scoring function (described in Section 3.3.2), high scores are assigned to the background knowledge terms that are strongly related to the positive examples. The resulting scores are shown in the middle of Figure 3. We can see that the left-hand side terms have higher scores compared to the right-hand side terms as they annotate mostly the positive (target) examples.

Table 1: An illustrative data set consisting of 15 examples annotated by 8 possible annotations.

Example	Class	Annotated by
1	+	A,B
2	-	A,B,C
3	+	B,C,D
4	+	B,C,E
5	+	B,C
6	-	C,D,E
7	+	D,E
8	+	D,E,F
9	-	E,F
10	-	E,F
11	-	F,G
12	-	F,G
13	-	G
14	+	G,H
15	+	G,H

In the next step, the algorithm prunes the lower-ranked terms, leaving only the best 8 background terms in the reduced hierarchy of ontology terms. As most of the top scoring terms were in the left part of the hierarchy, the reduced hierarchy mostly contains these terms. In the final step, we use a SDM algorithm—in this case we use Hedwig—to find rules based on the instances from Table 1 and the reduced ontology. The best two rules discovered by Hedwig, which cover most of the positive examples, are:

`Positive` ← `LL` (shown in green in Figure 3), covering 3 positive and 1 negative instance (Coverage = 4, Precision = $\frac{3}{4}$).

`Positive` ← `LR` (shown in red in Figure 3), covering 5 positive and 2 negative instances (Coverage = 7, Precision = $\frac{5}{7}$).

If we run Hedwig using the original ontology for this data set, we get the same rules as on the reduced background knowledge. However, as demonstrated by the experiments on large real-life data sets in Section 5, removing unimportant parts of large ontologies benefits both the execution time and the quality of discovered rules.

3.3. Detailed description of NetSDM

In this section, we describe four steps of the NetSDM methodology outlined in Algorithm 1. We first describe the conversion of a background knowledge ontology into an information network, which we can then analyze using network analysis methods. Next, we describe two methods for term importance estimation in information networks. The section concludes with a discussion on the term deletion step of the algorithm in which low-scoring terms are removed from the network.

3.3.1. Conversion of background knowledge ontology into network format

The first step of the NetSDM methodology is conversion of the background knowledge ontologies into a network. We present two methods, a *direct* method and a *hypergraph* method of conversion.

Direct conversion. Input ontologies for the NetSDM methodology are collections of triples that represent relations between background knowledge terms. As any information network is also composed of a set of nodes and the connections between them, a natural conversion from the background knowledge into an information network is to convert each relation between two ontology terms into an edge between two corresponding nodes in an information network. This gives rise to the first function converting the input ontology into the information network, in which we view the ontology and the data instances as parts of one network. In the conversion, we merge the data set and the background knowledge into one network $G_e = (V_e, E_e)$ where

- $V_e = \mathcal{T}(O) \cup S_+$ are vertices of the new network consisting of all background knowledge terms $\mathcal{T}(O)$ and all target class instances (positive examples) S_+;
- $E_e = \{(t_1, t_2) | \exists r : (t_1, r, t_2) \in O\} \cup \{(s, t) | (s, \textit{annotated-with}, t) \in \mathcal{A}\}$ are edges of the new network consisting of edges that represent background knowledge relations, as well as all edges that represent the *annotated-with* relations between

data instances and background knowledge terms (the meaning of O, S, S_+, S_- and $\mathcal{A}$ is explained in Section 3.1).

Hypergraph conversion. Liu et al. (2013) proposed an alternative approach to converting an ontology into an information network format. They treat every relation in a semantic representation of data as a triple, consisting of a subject, predicate and object. They construct a network (which they call a *hypergraph*) in which every triple of the original background knowledge (along with the background knowledge terms) forms an additional node in the network with three connections: one connection with the subject of the relation, one with the object of the relation, and one with the predicate of the relation. In this conversion, each predicate that appears in the background knowledge is an additional node in the resulting network. This conversion results in a network $G_e = (V_e, E_e)$ where:

- $V_e = \mathcal{T}(O) \cup S_+ \cup \{n_r | r \in O \cup \mathcal{A}\} \cup \{p | \exists s, o : (s, p, o) \in O\} \cup \{\textit{annotated-with}\}$ are vertices of the new network consisting of (i) all background knowledge terms, (ii) all target class instances, (iii) one new node n_r for each relation r either between background knowledge terms or linking background knowledge terms to data instances and (iv) one node for each predicate appearing in the ontology O, as well as (v) one node representing the *annotated-with* predicate;
- $E_e = \bigcup_{r=(s,p,o)\in O\cup\mathcal{A}} \{(s, n_r), (p, n_r), (o, n_r)\}$ are edges of the new network, each relation r inducing three connections to node n_r that it induces (terms V, S, S_+, S_- and $\mathcal{A}$ are defined in Section 3.1).

While hypergraph conversion results in a slightly larger information network, less information is lost in the conversion process than in the direct conversion process presented above.

An important aspect of both conversion methods, presented above, is that they are defined simply on a set of triples that represent relations between background knowledge terms. This means that both methods can be used to transform an arbitrarily complex ontology with any number of predicates.

3.3.2. Term significance score calculation

After converting the background knowledge and the input data into a joined network, we use either Personalized PageRank or node2vec based scoring function to evaluate the significance of background knowledge terms. Term importance scoring function $\text{score}_{S,O,\mathcal{A}}$ is constructed by taking into account the data set S and background knowledge terms $\mathcal{T}(O)$.

P-PR based score computation. The first scoring function is computed using the Personalized PageRank algorithm (Page et al., 1999). While the basic PageRank algorithm evaluates the global importance of each node in a network (i.e. with respect to all other nodes), the Personalized PageRank (described in Section 2.2.1) evaluates the significance of nodes with respect to a given node (or a set of nodes). This fits well with our demand that a scoring function must evaluate the significance of a term

based on the actual data - in our case the importance of the nodes is not global but calculated in the context of the given positive examples. We can therefore reasonably expect that the highest scoring terms will not simply be the most prominent terms of the background knowledge ontology but rather those terms which are the most relevant to the input data set that a researcher is interested in. As each term $t \in \mathcal{T}(O)$ is a vertex in network G_e, we calculate the P-PR score of each term $t \in \mathcal{T}(O)$ as follows:

$$\text{score}_{\text{P-PR}}(t) = \text{P-PR}_{S_+}(t) \tag{4}$$

where the P-PR vector is calculated on the network G_e. A simple algebraic calculation (Grčar et al., 2013) shows that this value is equal to the average of all Personalized PageRank scores of v where the starting set for the random walker is a node from S_+:

$$\text{P-PR}_{S_+}(t) = \frac{1}{|S_+|} \sum_{w \in S_+} \text{P-PR}_{\{w\}}(t) \tag{5}$$

Following the definition of the Personalized PageRank score, the value of $\text{score}_{\text{P-PR}}$ is the stationary distribution of a random walker that starts its walk in one of the target data instances (elements of the set S_+) and follows the connections (either the is-annotated-by relation or a relation in the background knowledge). Another interpretation of $\text{score}_{\text{P-PR}}(t)$ is that it tells us how often we will reach node t in random walks, starting with positive (target class labeled) data instances. Note that the Personalized PageRank algorithm is defined on directed networks, allowing us to calculate the Personalized PageRank vectors of nodes by taking the direction of connections into account. In our experiments, we also tested the performance of the scoring function if network G_e is viewed as an undirected network. To calculate the PageRank vector on an undirected network, each edge between two nodes u and v is interpreted as a pair of directed edges, one going from u to v and another from v to u, allowing the random walker to traverse the original undirected edge in both directions.

node2vec based score computation. The second estimator of background knowledge terms significance is the node2vec algorithm (Grover and Leskovec, 2016). In our experiments, we used the default settings of $p = q = 1$ for the parameters of node2vec, meaning that the generated random walks are balanced between the depth-first and breadth-first search of the network. The maximum length of random walks was set to 15. The function node2vec (described in Section 2.2.2) calculates the feature matrix $f^* = \text{node2vec}(G_e)$, and each row of f^* represents a feature vector $f^*(u)$ for node u in G_e. The resulting feature vectors of nodes can be used to compute the similarity between any two nodes in the network. The approach uses the cosine similarity of the two feature vectors u and v, computed as follows:

$$\text{similarity}_{\text{node2vec}}(u, v) = \frac{f^*(u) \cdot f^*(v)}{|f^*(u)||f^*(v)|}$$

In our approach, we form feature vectors for nodes representing both background knowledge terms and data instances. We use these feature vectors to compute the

similarity between the background knowledge terms and the positive data instances (target class examples) using a formula inspired by Equation 5. With the Personalized PageRank, we evaluate the score of each node as P-PR$_{S_+}(v)$ where S_+ is the set of all target class instances. As the value P-PR$_{\{w\}}(t)$ measures the similarity between w and t, Equation 5 can also be used to construct a node2vec scoring function. We replace the individual P-PR similarities in Equation 5 with the individual node2vec similarities:

$$\text{score}_{\text{node2vec}}(t) = \frac{1}{|S_+|} \sum_{w \in S_+} \frac{f^*(w) \cdot f^*(t)}{|f^*(w)||f^*(t)|} \tag{6}$$

3.3.3. Network node removal

In the third step of Algorithm 1, low-scored nodes are removed from the network. We present and test two options for network node removal.

Naïve node removal. The first (naïve) option is to take every node that is marked for removal and delete both the node and any connection leading to or from it. This method is robust and can be applied to any background knowledge. However, when background knowledge is heavily pruned, this method may cause the resulting network to get decomposed into several disjoint connected components. If we convert the background knowledge into a hypergraph, the hypergraph has to contain relation nodes with *exactly* three neighbors (the subject, predicate and object of the relation) if we are to convert it back into a standard representation of background knowledge. Thus, naïvely removing nodes from the hypergraph may result in a network that we can no longer convert back to the standard background knowledge representation.

Advanced node removal. We tested an alternative approach that takes into account that the relations, encoded by the network edges, are often transitive. For example, in our work we use the *is-a* and *is-part-of* relations that are both transitive. Using transitivity, we design an algorithm for advanced removal of low scoring nodes from information networks, obtained by direct conversion of the background knowledge (Algorithm 2). This advanced node removal step can be used for ontologies with any number of relations. Just like the is-a relation in Algorithm 2, other transitive relation are added back to the network in row three of the algorithm. As the background knowledge is in direct correspondence with the information network obtained from it, removing the node from the information network corresponds to removing a matching term from the background knowledge. In this sense, Algorithm 2 can be used as the term-removal step of Algorithm 1. Algorithm 2 can also be used to remove low scoring nodes from hypergraphs constructed from the background knowledge if we first convert hypergraphs into a simple network form resulting from the direct conversion of background knowledge ontologies.

3.3.4. Applying SDM algorithms

SDM algorithms Hedwig and Aleph (described in Sections 2.1.4 and 2.1.5, respectively) were used in the last step of the NetSDM methodology to discover rules explaining the target class instances.

Input: Information network G_e (obtained by direct conversion of background knowledge into information network format with directed edges) and node $n \in G_e$ (a term of the original background knowledge) we wish to remove
Output: New background knowledge that does not contain node n

```
for b ∈ G_e : (b, n) is an edge in G_e do
    for a ∈ G_e : (n, a) is an edge in G_e do
        Add edge (b, a) into G_e
        Remove edge (n, a) from G_e
    end
    Remove edge (b, n) from G_e
end
Remove node n from G_e
Return G_e
```

Algorithm 2: The algorithm for removing a node from a network, obtained through direct conversion of the background knowledge into the information network format.

Applying Hedwig. Hedwig—which was previously successfully used in a real-life application for explaining biological data (Vavpetič et al., 2014)—can use only one relation type in the background knowledge, i.e. the *is-a* relation in the case of the Gene Ontology. Hedwig uses this relation and its transitivity property (if A *is-a* B and B *is-a* C, then A *is-a* C) to construct rules.

Applying Aleph. In contrast to Hedwig, the input to Aleph can contain several relations as well as interactions between them. Therefore applying Aleph enabled us to use both the *is-a* and *is-part-of* relations in the Gene Ontology. Aleph is capable of using not only these two relations and their transitivity, but also the interaction between the two relations in the form of two facts: (i) if A *is-a* B and B *is-part-of* C, then A *is-part-of* C and (ii) if A *is-part-of* B and B *is-a* C, then A *is-part-of* C.

Since Aleph can use several relations for constructing rules, the background knowledge network used in Aleph experiments contained the edges representing *is-a* relations as well as the edges representing *is-part-of* relations.

4. Experimental settings

This section presents the settings of the experiments with two semantic data mining algorithms (Hedwig and Aleph) and two search space reduction mechanisms (based on Personalized PageRank and node2vec). We first describe the data sets, followed by the outline of the three experimental setups.

4.1. Data sets

In the main experiments, presented in this paper, we used two data sets: the acute lymphoblastic leukemia (ALL) and the breast cancer data set.

ALL (acute lymphoblastic leukemia) data. The ALL data set, introduced by Chiaretti et al. (2004), is a typical dataset for medical research. In data preprocessing we followed the steps used by Podpečan et al. (2011) to obtain a set of 1,000 enriched genes (forming the target set of instances) from a set of 10,000 genes, annotated by concepts from the Gene Ontology (Ashburner et al., 2000), which was used as the background knowledge in our experiments. In total, the data set contained 167,339 annotations (connections between genes and Gene Ontology terms). In previous work on analyzing the ALL data set, the performance of the SegMine methodology (Podpečan et al., 2011) was compared to the performance of the DAVID algorithm (Huang et al., 2008). In this work, we use the same data set to assess if network node ranking and reduction can decrease the run time and improve the performance of the tested SDM algorithms.

Breast cancer data. The breast cancer data set, introduced by Sotiriou et al. (2006), contains gene expression data on patients suffering from breast cancer. In previous work, Vavpetič et al. (2014) first clustered the genes into several subsets and then used a semantic subgroup discovery algorithm to interpret the most important subset (as determined by domain experts). In this work, we constructed rules for the same subset of genes as by Vavpetič et al. (2014) and did not use the input of domain experts. The set contains 990 genes out of a total of 12,019 genes. In our experiments, we used SDM algorithms to describe the 990 interesting genes. The data instances are connected to the Gene Ontology terms by 195,124 annotations.

To test whether the results of NetSDM on the Gene-Ontology-annotated data sets described above can be generalized to other data sets, we ran the NetSDM algorithm also on a set of smaller data sets, annotated by hierarchical background knowledge, which were used in conjunction with the Hedwig algorithm in our previous work (Adhikari et al., 2016). The results of these experiments are shown in Appendix A of this paper.

4.2. Experimental setups

We constructed three experimental setups to test the NetSDM methodology. In all setups, both Hedwig and Aleph are used in their standard modes (subgroup discovery and classification rule learning, respectively) to return the set of rules explaining the target class examples. Hedwig returns a rule set containing all the rules in the search beam at the end of search (meaning that the size of the returned rule set matches the size of the beam). Aleph returns a rule set consisting of rules with a sufficient coverage and low enough noise (i.e. rules covering at least 10 positive examples and at most 100 negative examples). In the first setup, the entire set of rules was analyzed. In the second setup, we used the NetSDM methodology with the direct method of ontology conversion and the naïve version of node removal. Given their different operation modes (Hedwig was used for learning individual patterns and Aleph was used for predictive theory construction), we always analyzed only the top rule discovered by Hedwig, and the top 3 rules of the theory discovered by Aleph. We also list the metrics for the entire theory discovered by Aleph. This allows clearer comparison and is sufficient to demonstrate how shrinking the background knowledge affects the quality of discovered rules. In the third setup, we use the advanced (transitivity based)

node removal approach (described in Section 3.3.3). We compare the quality of the resulting rules by measuring the Lift values of each rule (defined in Equation 1 in Section 2.1.4). Details of each experimental setup are given below.

First experimental setup. We ran the two SDM algorithms on the entire data set to determine the baseline performance and examine the returned rules. Using Hedwig, we ran the algorithm with all combinations of depth (1 or 10), beam width (1 or 10) and support (0.1 or 0.01). For Aleph, we ran the algorithm using the settings recommended by the algorithm author—minimum number of positive examples covered by a rule was set to 10, and maximum number of negative examples covered by a rule was set to 100. These settings resulted in rules comparable to those discovered by Hedwig.

In both cases, our goal was to determine whether the terms used by the SDM algorithms are correlated with the scores of the terms returned by the network analysis algorithms, i.e., calculated by P-PR (Equation 4) or node2vec (Equation 6). We evaluated this relation by observing the ranks of terms used by both algorithms to construct the rules. For each term t appearing in the constructed rules, we computed the relative rank of the term t (as defined in Section 3.1). If the correlation between SDM algorithm's use of terms and returned scores is strong, the relative rank calculated for a term t and used by the algorithm will be low. Low percentages ensure that we can retain only a small percentage of the highest ranked scores.

Second experimental setup. In the second set of experiments, we exploited the correlations discovered in the first experimental setup. We used the scores of background knowledge terms to prune the background knowledge. Specifically, we ran the NetSDM algorithm by setting the shrinkage coefficient values of c to values between 0.005 and 1, thereby running the SDM algorithms on the gene ontology (GO) background knowledge containing only a subset of as little as 0.5% of the terms with the highest scores. We calculated the P-PR values of GO terms in two ways: (i) we viewed *is-a* relations as directed edges pointing from a more specific GO term to a more general term, and (ii) we viewed the relations as undirected edges. When running the experiments with Hedwig, we set the beam size, depth and minimum coverage parameters to the values that returned the best rules on each data set in the first set of experiments. For example, the results of the first experimental setup showed that the rules obtained by setting the beam size or rule depth to one are too general to be of any biological interest; we therefore decided to set both values to 10. In the case of rule depth, this value allows Hedwig to construct arbitrarily long rules (given that in our data sets Hedwig returned no rules of length greater than five). Setting the beam size to 10 allows Hedwig to discover important rules (as shown in the first set of experiments) in a reasonable amount of time; note that the run time of Hedwig increases drastically with increased beam size, and at size 10 the algorithm takes several hours to complete. Running the experiments with the Aleph algorithm, we again used the setting recommended by the algorithm author.

Third experimental setup. In the first two sets of experiments, we used the direct method for converting background knowledge into an information network and the naïve method for deleting low scoring nodes. In the third set of experiments, we

Table 2: The best rules discovered by the Hedwig algorithm on the ALL data set. The conjuncts (Gene Ontology terms) of each top ranking rule are separated by a horizontal line in the table with the 4 right-most columns denoting the settings used to discover the rule, and the Lift value of the entire rule. The second (third) column can be understood as the percentage of GO terms with the Personalized PageRank (node2vec) score higher than the corresponding term.

Term	RelativeRank (P-PR)[%]	RelativeRank (Node2vec)[%]	Beam	Depth	Support	Lift
GO:0003674	0.297	46.383	1	1	0.01	1.000
GO:0003674	0.297	46.383	1	10	0.01	1.000
GO:0003674	0.297	46.383	1	1	0.1	1.000
GO:0003674	0.297	46.383	1	10	0.1	1.000
GO:0050851	4.467	2.066	10	1	0.01	2.687
GO:0002376	0.603	62.137	10	10	0.01	3.420
GO:0002429	2.506	13.985				
GO:0005886	0.076	0.876				
GO:0005488	0.050	16.979				
GO:0002376	0.603	62.137	10	1	0.1	1.292
GO:0002376	0.603	62.137	10	10	0.1	1.414
GO:0005488	0.050	16.979				
GO:0048518	1.056	93.724				
GO:0003674	0.0046	46.38	1	10	0.01	1

tested the advanced versions of both steps: hypergraph conversion and advanced node deletion, respectively. As the node2vec function proved inferior to the Personalized PageRank scoring function in the second set of experiments, we ran the third round of experiments only using the P-PR scoring function. The results of this third set of experiments can then be compared to the results of the second set.

5. Experimental results

We present the results of algorithms Hedwig and Aleph using two scoring functions, two network conversion methods and two node deletion methods on the two data sets. We analyze the results obtained from the three experimental setups.

5.1. First setup: Scores of terms used in SDM algorithms

We begin by presenting the results of the first experimental setup where we analyze the scores and the ranks of terms appearing in rules constructed by Hedwig and Aleph, respectively.

5.1.1. Using Hedwig

Tables 2 and 3 present the best ranked rules discovered by Hedwig using different parameter settings in the first experimental setup using Personalized PageRank and node2vec node relevance scores for the ALL and breast cancer data sets, respectively. The results show for all the terms used by Hedwig that the relative ranks calculated by the P-PR scoring function are remarkably high. In all the cases, the terms used to construct rules are in top

Table 3: The best rules discovered by the Hedwig algorithm on the breast cancer data set. The meaning of columns is the same as explained in the caption of Table 2.

Term	RelativeRank (P-PR)[%]	RelativeRank (Node2vec)[%]	Beam	Depth	Support	Lift
GO:0043230	11.350	31.000	1	1	0.01	1.400
GO:0043230	11.350	31.000	1	10	0.01	1.400
GO:0043230	11.350	31.000	1	1	0.1	1.400
GO:0043230	11.350	31.000	1	10	0.1	1.400
GO:0000785	0.821	24.693	10	1	0.01	1.868
GO:0003674	0.202	36.568				
GO:0044427	0.409	20.538	10	10	0.01	3.743
GO:0000278	0.091	0.665				
GO:0022402	0.312	59.868				
GO:0043228	9.062	25.750	10	1	0.1	1.439
GO:0071840	29.454	58.599				
GO:0044428	0.051	13.580	10	10	0.1	1.556
GO:0003674	0.202	36.568				

percentiles of all term scores, giving credence to our hypothesis that high scoring terms are usually used in rules. On the other hand, use of the node2vec based scoring function was much less successful in this experiment.

This phenomenon is clearly shown in the top two charts of Figure 4 showing all the terms used by Hedwig and their ranks. Even taking into account all (not only the best) rules, we see that predominantly the used terms come from the top 5 percent of all the terms as scored by the Personalized PageRank scoring function. On the other hand, this phenomenon does not occur with the node2vec scoring function, as the results in Table 2 and Table 3 show for the ALL and breast cancer data sets. The rules used by Hedwig score quite low according to the node2vec function, however as seen in the bottom two charts of Figure 4, the node2vec score of the used terms is still above average, and it is possible that the node2vec scores are useful. We consider the node2vec scores to contain sufficient information to warrant testing both scores in the second experimental setup.

The rules discovered in the first experimental setup are biologically relevant, except when the search beam for the Hedwig algorithm was set to 1. In this case the only significant rule discovered contained in its condition a single gene ontology term GO:0003674 that denotes molecular function. This is a very general term that offers little insight and shows that a larger search beam is necessary in order for Hedwig to make significant discoveries. The most interesting results are uncovered when the beam size is set to 10 and the support is set to 0.01. When the depth is set to 1, the most important term GO:0050851 (antigen receptor-mediated signaling pathway) relates to the immune system related cell type. When searching with a depth of 10, we discovered a conjunct of four terms: immune system process (GO:0002376), immune response-activating cell surface receptor signaling pathway, (GO:0002429), plasma membrane (GO:0005886) and binding (GO:0005488). This conjunct provides additional insights about the action (binding), effect (immune response signaling pathway), and location (plasma membrane).

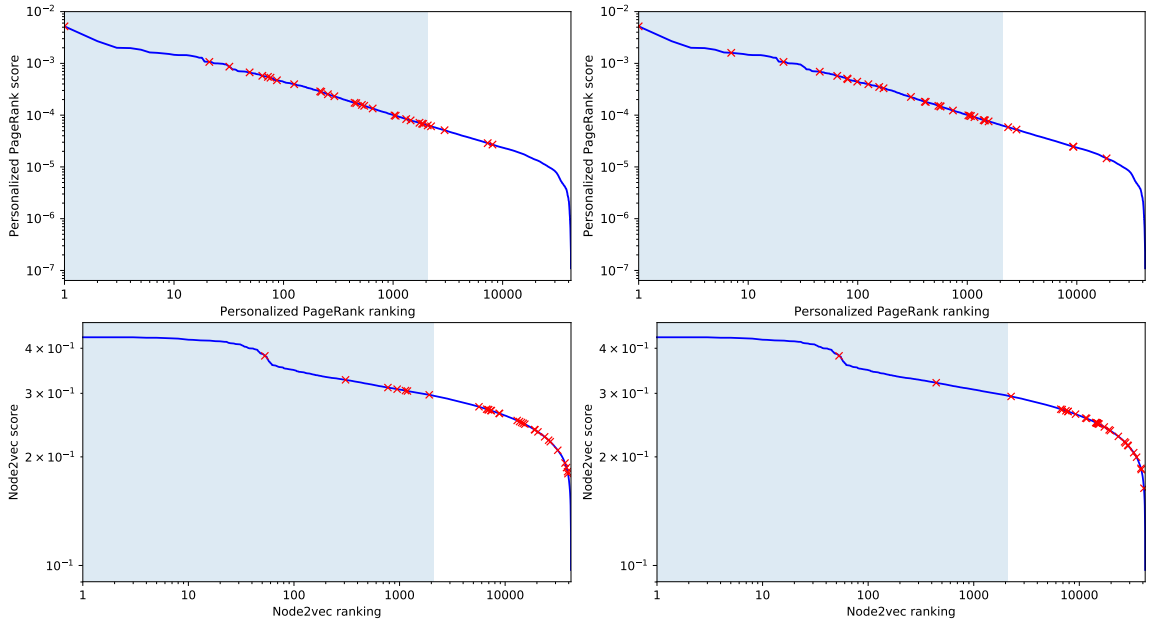

Figure 4: The ranks of the terms used by Hedwig to construct rules on the ALL (left) and breast cancer (right) data sets. The blue line shows the score for each of the terms in the background knowledge—the scores on the top two charts were calculated with Personalized PageRank and the scores on the bottom two charts were produced with node2vec. The terms in each chart are sorted by descending score. Each red cross highlights one of the terms used by Hedwig. The blue area on the left denotes the top 5 percent of all nodes (i.e. nodes with a relative rank of 0.05 or lower). Note the logarithmic scale of both axes.

5.1.2. Using Aleph

Tables 4 and 5 show the terms, used in the 10 best rules discovered by the Aleph algorithm that describe the ALL and breast cancer data sets, respectively, and the P-PR and node2vec scores of the terms in the rules. The scores were calculated taking both *is-a* and *is-part-of* relations into account.

The results again show that the Personalized PageRank scoring function ranks all the terms used by Aleph remarkably high. Moreover, similar to the experiments with Hedwig, the node2vec scoring function is much less successful in this experiment. The high Personalized PageRank ranking of terms appearing in rules discovered by the Aleph algorithm is also seen in Figure 5.

Similar to the results using the Hedwig algorithm, the discovered rules contain terms predominantly from the top 5 percent of all terms as scored by the Personalized PageRank scoring function. A high value of the node2vec scoring function is again a weaker predictor of whether a rule will be useful or not. Nevertheless, as seen in the bottom two charts of Figures 4 and 5, the node2vec score of the used terms is still above average, and it is

possible that the node2vec scores are useful. We consider the node2vec scores to contain sufficient information to warrant testing both scores in the second experimental setup.

5.2. Second setup: Shrinking the background knowledge ontology

This section presents the result of the second set of experiments with the NetSDM methodology. We present the results of shrinking the background knowledge for Hedwig and Aleph.

5.2.1. Ontology shrinking for Hedwig

The results of the first experimental setup show that the scores, calculated using the network analysis techniques, are relevant to determine whether background knowledge nodes will be used by SDM algorithms in the construction of rules describing the target instances. Here we present the results of using the computed scores to shrink the background knowledge used by Hedwig. We first present the results of both scoring functions on the ALL data set and then on the breast cancer data set. We set the beam and depth parameters of Hedwig to 10, and the minimum coverage parameter to 0.01; these are the settings that produced the best results for both data sets in the first experimental setup with Hedwig with no shrinking of the background knowledge (see Tables 2 and 3).

For Personalized PageRank-based network shrinkage we also test if taking into account edge directions makes a difference. Table 6 (undirected edges) and Table 7 (directed edges) show the results of semantic rule discovery using the P-PR function on the ALL data set. Both tables show a similar phenomenon: by decreasing the size of the background knowledge

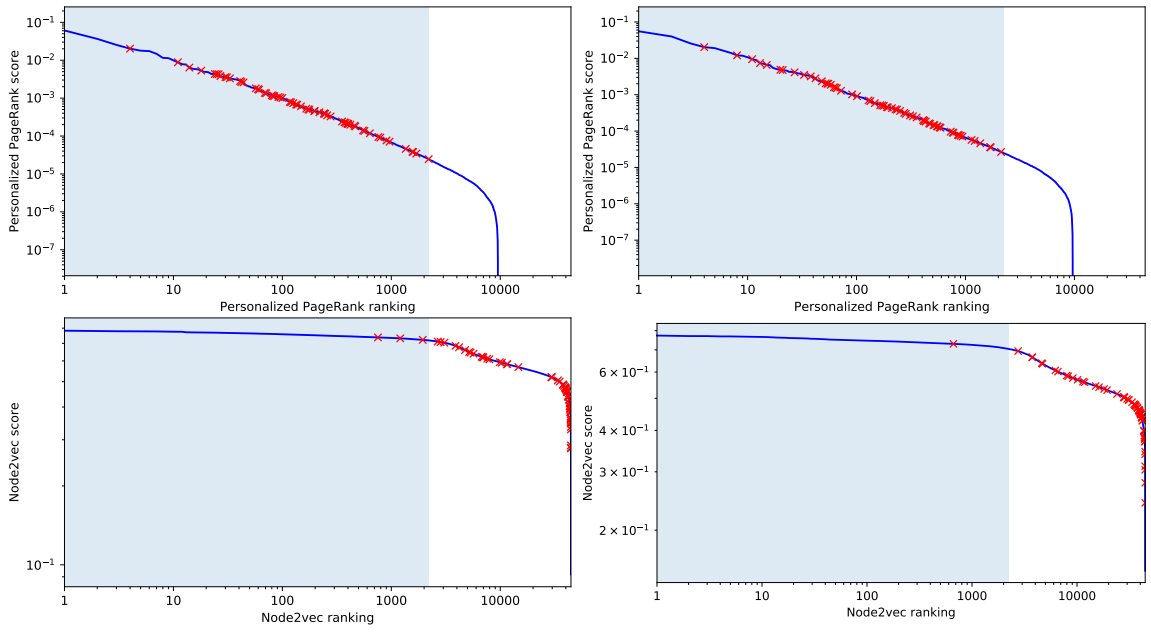

Figure 5: The ranks of the terms used by Aleph to construct rules on the ALL (left) and breast cancer (right) data sets. The chart interpretation is the same as explained in the caption of Figure 4. Note again the logarithmic scale of both axes.

Table 4: The 10 best rules discovered by the Aleph algorithm on the ALL data set. The meaning of the second and third column is the same as in Table 2. The scores were calculated by taking the *is-a* and *is-part-of* relations into account.

Term	RelativeRank (P-PR)[%] %	RelativeRank (Node2vec)[%]	Coverage	TP	Accuracy
GO:0016462	0.444	66.115	65	13	0.884
GO:0031224	0.092	11.634			
GO:0050839	2.161	12.756	69	15	0.884
GO:0031982	0.397	5.980			
GO:0098609	0.855	99.684	76	16	0.883
GO:0016021	0.094	11.982			
GO:0002703	4.948	97.078	86	21	0.883
GO:0030155	0.886	99.031	92	21	0.882
GO:0005634	0.056	6.891			
GO:0031224	0.092	11.634	98	24	0.882
GO:0055114	0.184	99.827			
GO:0050851	0.792	86.346	139	44	0.882
GO:0005488	0.009	94.384			
GO:0044255	0.283	97.785	117	33	0.882
GO:1901615	1.030	99.553			
GO:0006796	0.188	98.970	83	15	0.882
GO:0051129	3.045	97.949			
GO:0007165	0.058	99.764	117	32	0.882
GO:0045321	0.622	92.021			
GO:0044425	0.074	25.869			

Table 5: The 10 best rules discovered by the Aleph algorithm on the breast cancer data set. The meaning of the columns is the same as in Table 4.

Term	RelativeRank (P-PR)[%] %	RelativeRank (Node2vec)[%]	Coverage	TP	Accuracy
GO:0022610	0.399	98.582	39	11	0.914
GO:0005578	1.041	13.932			
GO:0050794	0.045	99.980	57	11	0.912
GO:0042592	0.435	93.017			
GO:0044428	0.085	10.539	77	20	0.912
GO:0000226	1.243	80.543			
GO:0043230	0.512	10.349	69	14	0.911
GO:0043087	0.969	80.674			
GO:0044424	0.018	25.560	93	21	0.911
GO:0044843	2.028	53.978			
GO:0044427	0.330	8.428	103	26	0.911
GO:0050790	0.227	99.147			
GO:0008509	3.826	74.934	76	12	0.910
GO:0044699	0.047	99.996			
GO:0005488	0.009	92.797			
GO:0016491	0.289	99.942	86	14	0.910
GO:0080090	0.426	99.760			
GO:0051186	1.227	97.193	84	13	0.910
GO:0043169	0.108	43.342			
GO:0043226	0.029	17.895	97	19	0.910
GO:0004842	0.615	79.657			

the rules discovered by the Hedwig algorithm either stay the same or change to rules with a higher lift value. When searching for longer rules, reducing the size of the network by

Table 6: Results of NetSDM using direct network conversion with *undirected* edges, Personalized PageRank scoring function, naïve node removal and the Hedwig SDM algorithm on the ALL data set.

Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
5	GO:0002376 GO:0002694 GO:0034110	immune system process regulation of leukocyte activation regulation of homotypic cell-cell adhesion	3.235	111	40
10	GO:0002376 GO:0002694 GO:0044459	immune system process regulation of leukocyte activation plasma membrane part	4.090	90	41
20	GO:0003824 GO:0044283 GO:0044444	catalytic activity small molecule biosynthetic process cytoplasmic part	4.257	116	55
50	GO:0002376 GO:0002429 GO:0005886 GO:0005488	immune system process immune response-activating cell surface receptor signaling pathway plasma membrane binding	3.420	105	40
100	GO:0002376 GO:0002429 GO:0005886 GO:0005488	immune system process immune response-activating cell surface receptor signaling pathway plasma membrane binding	3.420	105	40

50% still allows us to discover the same high quality conjunct of the terms GO:0002376, GO:0002429, GO:0005886 and GO:0005488 as before.

Comparing Table 6 and Table 7 we see that there is a slight difference in the resulting rules if we take the direction of the relations in the background knowledge into account. Better Lift values and more consistent results are obtained when we choose to ignore the direction of the relations. Taking edge directions into account, for example, leads to a discovery of a very low quality rule with Lift value of only 2.219 when the background knowledge is at 20% of its original size. When we do not ignore the direction of the relations, the score of each node is only allowed to propagate to the 'parent' nodes in the background knowledge, i.e. high scores will be given to those GO terms whose child terms (specializations) have a high score. On the other hand, when ignoring the direction, we allow nodes to pass their high scores also to sibling nodes, allowing Hedwig in the final step of our NetSDM methodology (outlined in Algorithm 1) to choose the correct specialization of the parent node. More experiments are needed to analyze this phenomenon.

Table 8 and Table 9 show the results of the experiments using the node2vec function on the ALL data set. The results show that node2vec based ontology shrinkage did not achieve the same performance as the shrinkage using the Personalized PageRank function. The results in both tables show that lower shrinkage coefficients consistently decrease the quality of the rules discovered by Hedwig, meaning that shrinking of the background knowledge with the node2vec scoring function removes several informative terms and connections. Without these terms Hedwig could not extract the relevant information from the data, causing decreased performance.

Another interesting phenomenon in the node2vec results is that as the background knowledge decreases in size, the resulting rules become shorter, with the lowest values

Table 7: Results of NetSDM using direct network conversion with *directed* edges, Personalized PageRank scoring function, naïve node removal and the Hedwig SDM algorithm on the ALL data set.

Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
5	GO:0003824 GO:0044283 GO:0044444 GO:0044238	catalytic activity small molecule biosynthetic process cytoplasmic part primary metabolic process	3.741	132	55
10	GO:0003824 GO:0044283 GO:0044444 GO:0044238	catalytic activity small molecule biosynthetic process cytoplasmic part primary metabolic process	3.769	131	55
20	GO:0045936 GO:0003824 GO:0009892	negative regulation of phosphate metabolic process catalytic activity negative regulation of metabolic process	2.219	89	22
50	GO:0002376 GO:0002429 GO:0005886 GO:0005488	immune system process immune response-activating cell surface receptor signaling pathway plasma membrane binding	3.420	105	40
100	GO:0002376 GO:0002429 GO:0005886 GO:0005488	immune system process immune response-activating cell surface receptor signaling pathway plasma membrane binding	3.420	105	40

Table 8: Results of NetSDM using direct network conversion with *undirected* edges, node2vec scoring function, naïve node removal and the Hedwig SDM algorithm on the ALL data set.

Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
5	GO:0050852	T cell receptor signaling pathway	2.586	125	36
10	GO:0050852	T cell receptor signaling pathway	2.586	125	36
20	GO:0050852	T cell receptor signaling pathway	2.586	125	36
50	GO:0050863	regulation of T cell activation	3.076	108	37
100	GO:0002376 GO:0002429 GO:0005886 GO:0005488	immune system process immune response-activating cell surface receptor signaling pathway plasma membrane binding	3.420	105	40

returning a single GO term as the only important node. This may be a consequence of so many terms pruned that the remaining terms describe non-intersecting sets of enriched genes (or sets with a very small intersection). In this case, term GO:0044281 covers the largest number of instances and is therefore selected as the best term. Upon selecting this term, Hedwig is not able to improve the rule by adding further conjuncts, as adding any other term to the conjunction drastically decreases the coverage of the resulting rule.

Tables 11 and 10 show the results of the Personalized PageRank scoring function used in reducing the background knowledge for the breast cancer data set. As in the ALL data set, we see that decreasing the background knowledge size does not drastically decrease the

Table 9: Results of NetSDM using direct network conversion with *directed* edges, node2vec scoring function, naïve node removal and the Hedwig SDM algorithm on the ALL data set.

Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
5	GO:0050852	T cell receptor signaling pathway	2.586	125	36
10	GO:0050852	T cell receptor signaling pathway	2.586	125	36
20	GO:0050852	T cell receptor signaling pathway	2.586	125	36
50	GO:0007507 GO:0032502	heart development developmental process	2.124	93	22
100	GO:0002376 GO:0002429 GO:0005886 GO:0005488	immune system process immune response-activating cell surface receptor signaling pathway plasma membrane binding	3.420	105	40

Table 10: Results of NetSDM using direct network conversion with *undirected* edges, Personalized PageRank scoring function, naïve node removal and the Hedwig SDM algorithm on the breast cancer data set.

Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
5	GO:0071840 GO:0000278 GO:0005515 GO:0044260 GO:0022402	cellular component organization or biogenesis mitotic cell cycle protein binding cellular macromolecule metabolic process cell cycle process	4.114	121	41
10	GO:0043232 GO:0000278 GO:0005515	intracellular non-membrane-bounded organelle mitotic cell cycle protein binding	3.055	151	38
20	GO:0071840 GO:0007049 GO:0003824 GO:0005515	cellular component organization or biogenesis cell cycle catalytic activity protein binding	3.781	122	38
50	GO:0003824 GO:0000278 GO:0022402 GO:0005515	catalytic activity mitotic cell cycle cell cycle process protein binding	3.469	126	36
100	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37

quality of rules discovered by Hedwig. In fact, the rules discovered from only 20% of the original background knowledge achieve almost the same Lift score as the rules produced from the entire data set. Further shrinking the data set allows Hedwig to discover an even better rule. Comparing the two tables we see—even more clearly than in the ALL data set—that ignoring the direction of edges resulted in better overall performance of the algorithm. Especially for small background knowledge sizes, NetSDM finds rules with substantially lower Lift values using scores calculated on directed edges compared to those calculated by ignoring the edge direction.

Table 11: Results of NetSDM using direct network conversion with *directed* edges, Personalized PageRank scoring function, naïve node removal and the Hedwig SDM algorithm on the breast cancer data set.

Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
5	GO:0003824	catalytic activity	3.035	136	34
	GO:0007049	cell cycle			
	GO:0044446	intracellular organelle part			
10	GO:0044421	extracellular region part	2.587	122	26
	GO:0003824	catalytic activity			
	GO:0070062	extracellular vesicular exosome			
	GO:0044707	single-multicellular organism process			
20	GO:0003674	molecular function	3.743	120	37
	GO:0044427	chromosomal part			
	GO:0000278	mitotic cell cycle			
	GO:0022402	cell cycle process			
50	GO:0003674	molecular function	3.743	120	37
	GO:0044427	chromosomal part			
	GO:0000278	mitotic cell cycle			
	GO:0022402	cell cycle process			
100	GO:0003674	molecular function	3.743	120	37
	GO:0044427	chromosomal part			
	GO:0000278	mitotic cell cycle			
	GO:0022402	cell cycle process			

Table 12: Results of NetSDM using direct network conversion with *undirected* edges, node2vec scoring function, naïve node removal and the Hedwig SDM algorithm on the breast cancer data set.

Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
5	GO:0000082	G1/S transition of mitotic cell cycle	2.589	136	29
10	GO:0000082	G1/S transition of mitotic cell cycle	2.589	136	29
20	GO:0000082	G1/S transition of mitotic cell cycle	2.589	136	29
50	GO:1903047	mitotic cell cycle process	2.916	204	49
	GO:0005515	protein binding			
100	GO:0003674	molecular function	3.743	120	37
	GO:0044427	chromosomal part			
	GO:0000278	mitotic cell cycle			
	GO:0022402	cell cycle process			

The results of NetSDM using the node2vec scoring funciton on the breast cancer data set, shown in Tables 12 and 13 are comparable to those on the ALL data set. The Hedwig algorithm does not discover important rules to explain the positive examples with the pruned background knowledge. As on the ALL data set, too many important nodes were pruned. Again we observe the phenomenon that as the background knowledge is shrunk, the length of the rules decreases to 1.

While the increased rule quality shows that using the Personalized PageRank as a filter before applying the Hedwig algorithm can improve the performance of the algorithm, the results are also significant considering the fact that with small background knowledge sizes

Table 13: Results of NetSDM using direct network conversion with *directed* edges, node2vec scoring function, naïve node removal and the Hedwig SDM algorithm on the breast cancer data set.

Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
5	GO:0000082	G1/S transition of mitotic cell cycle	2.589	136	29
10	GO:0044699 GO:0000278	single-organism process mitotic cell cycle	3.541	120	35
20	GO:0000082	G1/S transition of mitotic cell cycle	2.589	136	29
50	GO:0000082	G1/S transition of mitotic cell cycle	2.589	136	29
100	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37

the search space of Hedwig (and thus the computational complexity of the algorithm) is much smaller.

Figure 6 shows the relationship between the time taken by the Hedwig semantic rule learning algorithm (together with network preprocessing) to discover the relevant rules and the network shrinkage coefficient c, i.e. the relative size of the background knowledge left for Hedwig to analyze. As the network preprocessing steps were completed in seconds, the times reported in Figure 6 correspond almost entirely to the work of the Hedwig algorithm. The fact that shrinking takes orders of magnitude less time than rule discovery is a prerequisite for the SDM algorithms to be useful in real-life applications. We timed the algorithm on the ALL data set using different settings for the beam, depth and support on 8 core 2.60 GHz Intel Xeon(R)E5-2697 v3 machine with 64GB of RAM. The graph clearly shows that smaller background knowledge ontologies result in much shorter run times (note the logarithmic scale). This confirms our hypothesis that shrinking the background knowledge decreases the time it takes Hedwig to search the space of possible hypotheses. We can conclude that using the NetSDM background knowledge shrinking approach is beneficial both for increasing the quality of the rules and for speeding up the Hedwig algorithm.

5.2.2. Ontology shrinking for Aleph

This section examines the results of the second set of experiments with the Aleph algorithm used in the final step of the NetSDM methodology. Due to low quality of results achieved using node2vec in the first set of experiments with Aleph and Hedwig (presented in Section 5.1) as well as in the second set of experiments with Hedwig, we only tested Personalized PageRank based background knowledge shrinkage with the Aleph algorithm. Furthermore, due to low effect of taking directions of edges into account in the experiments with Hedwig, we only used the undirected network to calculate the scores for the terms in the background knowledge.

Table 14 shows the results of the Aleph algorithm on the ALL data set in the second set of experiments. The results of the algorithm on the breast cancer data set are shown in Table 24 of Appendix B. The results show that unlike with the Hedwig algorithm, using the Aleph algorithm on very heavily pruned data set does not result in the discovery of high quality rules. When the background knowledge is very small, the quality of the discovered rules, as

Table 14: Results of NetSDM on the ALL data set using the P-PR scoring function, direct network conversion and the naïve node removal, and using Aleph to learn a rule set (a theory, where only top 3 best rules are shown).

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
0.5	Theory	6,594	772	0.306	1.024
	GO:0005813 GO:0043167	92	19	0.880	1.807
	GO:0005829 GO:0055114	229	45	0.870	1.719
	GO:0044281 GO:0005829 GO:0044260	216	39	0.870	1.580
1	Theory	4,867	634	0.473	1.140
	GO:0000287 GO:0005575	118	26	0.878	1.928
	GO:0009897 GO:0031224	126	27	0.877	1.875
	GO:0055114 GO:0005829	229	45	0.870	1.719
2	Theory	4,180	601	0.544	1.258
	GO:0002250 GO:0005488	129	33	0.878	2.238
	GO:0005515 GO:0030027	100	24	0.880	2.100
	GO:0005886 GO:0001775 GO:0009987	238	53	0.871	1.949
3	Theory	4,333	599	0.526	1.210
	GO:0005515 GO:0050851	132	39	0.880	2.585
	GO:0030027 GO:0009987	103	25	0.880	2.124
	GO:0046649 GO:0071944	141	34	0.877	2.110
4	Theory	3,977	583	0.563	1.283
	GO:0002253 GO:0071944 GO:0008150	180	52	0.877	2.528
	GO:0030027 GO:0009987	103	25	0.880	2.124
	GO:0005634 GO:0048731 GO:0065007	144	33	0.877	2.005
5	Theory	4,136	589	0.546	1.246
	GO:0002253 GO:0071944	183	54	0.877	2.582
	GO:0030027 GO:0009987	103	25	0.880	2.124
	GO:0071944 GO:0005768	151	32	0.876	1.854
10	Theory	3,789	597	0.588	1.379
	GO:0005515 GO:0050851 GO:0065007	128	37	0.880	2.529
	GO:0009987 GO:0006066	171	45	0.876	2.303
	GO:0044763 GO:0008289 GO:0009653	65	16	0.882	2.154
20	Theory	3,214	548	0.643	1.492
	GO:0050851 GO:0071944	82	36	0.885	3.841
	GO:0080135 GO:0044249	138	35	0.878	2.219
	GO:0030027 GO:0009987	100	24	0.880	2.100
50	Theory	2,797	529	0.686	1.655
	GO:0008610 GO:0044283	127	41	0.881	2.825
	GO:0050851 GO:0005488	139	44	0.880	2.770
	GO:0080135 GO:0044249	138	35	0.878	2.219
100	Theory	2,765	528	0.690	1.671
	GO:0008610 GO:0044283	127	41	0.881	2.825
	GO:0002684 GO:0098552	82	23	0.882	2.454
	GO:0050852 GO:0005515	111	31	0.880	2.444

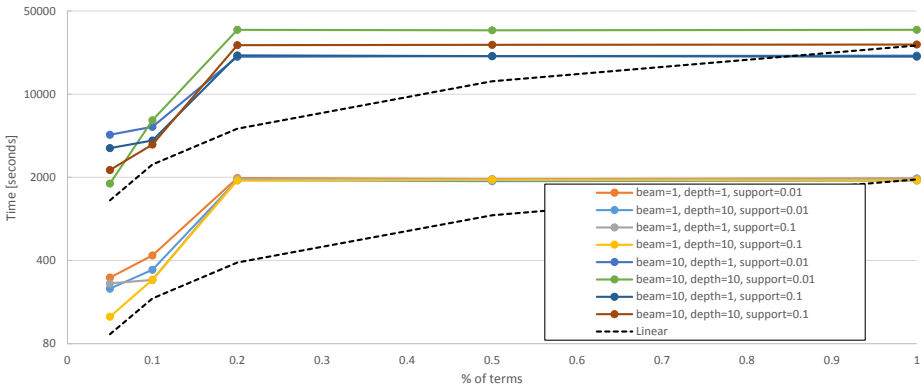

Figure 6: The times (in seconds) on logarithmic scale taken by Hedwig to discover rules on the ALL data set with different settings for beam, depth and support at varying sizes of the background knowledge set (different percentages of terms remaining in the background knowledge) in the second experimental setup. The times are plotted using a logarithmic scale on the y axis, and two linear trendlines (dashed) are drawn for comparison.

well as of the set of all rules, drops significantly. The benefits of network shrinkage for Aleph are visible when the background knowledge is pruned to 20% of its original size. At this background knowledge size, while the time taken to discover the rules was comparable to the time taken on the original background knowledge, the quality of the top two discovered rules is actually higher. Similar results were obtained using the Aleph algorithm on the breast cancer data set, shown in table 24 in Appendix B.

5.3. Third setup: Advanced network conversion and node deletion

This section presents the results of the final set of experiments using the advanced network conversion and node removal on both data sets using both learning algorithms.

5.3.1. Advanced network conversion for Hedwig

The results of the third set of experiments show somewhat different results from those achieved in the second experimental setup. Table 15 shows the results of both network conversion methods on the ALL data set, while Table 16 shows the results of the methods on the breast cancer data set. Unlike in the second set of experiments, the results of this setup show that the quality of rules, and also the discovered rules, remain the same when we reduce the size of background knowledge to one tenth of its original size. When further reducing the network size we see that the direct method of network conversion outperforms the advanced hypergraph based construction; while the hypergraph version of background knowledge conversion returns significantly worse results at 1% of its original size (in the breast cancer data set), the direct version still finds the same rule we find using the original background knowledge. When shrinking the network even further, the quality of the best

Table 15: Results of NetSDM on the ALL data set using the P-PR scoring function with advanced node removal, using Hedwig. We only list the smallest background knowledge size at which a network conversion method (direct or hypergraph) discovers the same rules as with no pruning, and one size below that.

Conversion method	Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
None	1	GO:0002376 GO:0002429 GO:0005886 GO:0005488	immune system process immune response-activating cell surface receptor signaling pathway plasma membrane binding	3.420	105	40
Direct	0.05	GO:0002376 GO:0002429 GO:0005886 GO:0005488	immune system process immune response-activating cell surface receptor signaling pathway plasma membrane binding	3.420	105	40
Direct	0.01	GO:0016020 GO:0050851 GO:0005488	membrane antigen receptor-mediated signaling pathway binding	3.413	89	34
Hypergraph	0.2	GO:0002376 GO:0002429 GO:0005886 GO:0005488	immune system process immune response-activating cell surface receptor signaling pathway plasma membrane binding	3.420	105	40
Hypergraph	0.1	GO:0002376 GO:0002694 GO:0005886 GO:0034110	immune system process regulation of leukocyte activation plasma membrane regulation of homotypic cell-cell adhesion	3.395	100	38

discovered rule decreases, but the decrease in quality occurs at a smaller size for the direct network conversion.

Figure 7 shows the run times of Hedwig on the pruned networks in the third set of experiments. The graph shows that unlike with the naïve network reduction methods in the second set of experiments, the time required to discover rules decreases linearly as we lower the shrinkage coefficient. When reducing the background knowledge network to 1% of its original size, the algorithm was able to discover the rules in two minutes, compared to over 11 hours on the unpruned network.

The result of this set of experiments can be compared with the results of the second set of experiments to draw two conclusions. First, the naïve version of network node removal causes Hedwig to discover rules of a slightly higher quality, but the results of shrinking the background knowledge are unpredictable both in terms of the quality of the rules discovered (which for some background knowledge sizes falls below the base value obtained on the unpruned network). Second, the times taken to discover the rules do not decrease significantly until we decrease the size of the background knowledge to 20% of its original size. On the other hand, the advanced version of node removal allows the algorithm to discover the same results as on the unpruned network even for very heavily pruned background knowledge sets. The results remain consistent, and the times taken to discover the rules decrease much more predictably with this node deletion method.

Table 16: Results of NetSDM on the breast cancer data set using the P-PR scoring function with advanced node removal, using Hedwig.

Conversion method	Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
None	1	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37
Direct	0.01	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37
Direct	0.005	GO:0022402 GO:0005654	cell cycle process nucleoplasm	2.942	120	30
Hypergraph	0.1	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37
Hypergraph	0.05	GO:0022411 GO:0005515 GO:0044422 GO:0032991	cellular component disassembly protein binding mitotic cell cycle macromolecular complex	3.743	120	37

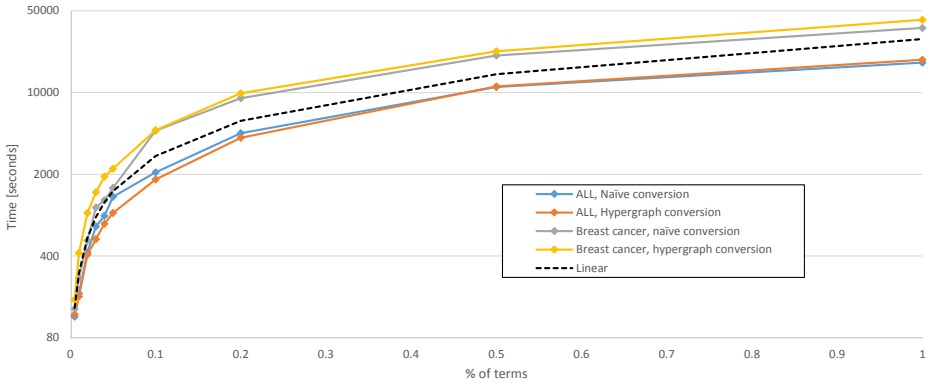

Figure 7: The times (in seconds) on logarithmic scale taken by Hedwig in the third experimental setup to discover rules with different conversion methods at varying sizes of the background knowledge (different percentages of terms remaining in the background knowledge). The times are plotted using a logarithmic scale on the y axis, and a linear trendline (dashed) is drawn for comparison.

5.3.2. Advanced network conversion for Aleph

Table 17 shows the results of the Aleph algorithm on the background knowledge pruned using advanced node reduction and direct network conversion, while Table 18 shows the results of background knowledge reduction using hypergraph network conversion. The results for all possible shrinkage coefficients are shown in Tables 22 and 23 in Appendix B. The results are comparable to the results obtained using the Hedwig algorithm: when the background knowledge is pruned to 10% (for the direct conversion) and 50% (for the hypergraph con-

version) of its original size, Aleph discovers the same best three rules. Furthermore, while shrinking the background knowledge further to only 3% of its original size, the quality of the discovered rules (measured by their Accuracy and Lift value) remains very close to the quality of the rules discovered on the entire background knowledge. The drop in quality is slightly smaller when using the hypergraph network conversion. Interestingly, for both network conversion methods, on a heavily pruned background knowledge (1% and 2% of its original size for the direct network conversion and for the hypergraph network conversion, respectively), the algorithms are able to discover rules of a higher quality than those on the original background knowledge. Similar results were obtained on the breast cancer data set and are shown in Appendix B in Tables 25 and 26.

Table 17: Results of NetSDM on the ALL data set using the P-PR scoring function, direct network conversion and advanced node removal. For each percentage of terms remaining in the reduced ontology, we show only the top 3 rules discovered by Aleph, and the properties of the entire set of rules (Theory).

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
0.5	Theory	2,561	489	0.704	1.671
	GO:0005783, GO:0005789, GO:0009058	63	19	0.883	2.639
	GO:0044459, GO:0002376, GO:0007275	137	38	0.879	2.427
	GO:0006950, GO:0015630	79	19	0.881	2.104
1	Theory	2,673	515	0.697	1.686
	GO:0050851, GO:0071944	82	36	0.885	3.841
	GO:0006066, GO:0044238, GO:0003824	122	40	0.881	2.869
	GO:0001775, GO:0006793	121	35	0.880	2.531
10	Theory	2,760	529	0.690	1.677
	GO:0008610, GO:0044283	127	41	0.881	2.825
	GO:0045321, GO:0005886	117	35	0.880	2.618
	GO:0002250, GO:0005623	119	33	0.880	2.426
100	Theory	2,568	510	0.708	1.738
	GO:0008610, GO:0044283	127	41	0.881	2.825
	GO:0045321, GO:0005886	116	34	0.880	2.565
	GO:0010008, GO:0032991	97	28	0.881	2.526

The run times of Aleph are shown in Figure 8. Compared to Hedwig, shrinking the background knowledge to a relatively small set does not have the same drastic effect on the run time. For example, shrinking the background knowledge to 10% of its original size for the direct network construction, which allows us to discover the same rules as with no ontology shrinkage, only decreases the run time by 14%, compared to the 90% time decrease for the Hedwig algorithm. Using the hypergraph network construction, a slightly larger speed-up was achieved in our experiments, however this is counter-balanced by the fact that for hypergraph network construction, the shrinkage coefficient must be set to a higher value in order to obtain the same rules.

The most promising finding of this series of experiments is that for very heavily pruned background knowledge (1% and 2% of the original size), Aleph (using the hypergraph conversion method) is capable of discovering high quality rules explaining the data. The times taken to discover the rules at this background knowledge size are halved when compared to times on the original background knowledge, and the resulting rules are of a higher quality

Table 18: Results of NetSDM on the ALL data set using the P-PR scoring function, hypergraph network conversion and advanced node removal. For each percentage of terms remaining in the reduced ontology, we show only the top 3 rules discovered by Aleph, and the properties of the entire set of rules (Theory).

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
0.5	Theory	2,577	492	0.703	1.671
	GO:1901362, GO:0044255, GO:0044281	84	26	0.882	2.708
	GO:0005874, GO:0005515, GO:0044707	48	14	0.883	2.552
	GO:0007166, GO:0050776, GO:0016021	138	40	0.879	2.536
2	Theory	2,534	516	0.713	1.782
	GO:0050870, GO:0005886	87	35	0.884	3.520
	GO:0007166, GO:0050778, GO:0044459	141	48	0.881	2.979
	GO:0007017, GO:0006950	48	15	0.884	2.734
50	Theory	2,648	527	0.703	1.741
	GO:0002376, GO:0044459, GO:0050865	140	48	0.881	3.000
	GO:0002429 GO:0071944	138	47	0.881	2.980
	GO:0006066, GO:0003824, GO:0044238	118	38	0.881	2.818
100	Theory	2,616	521	0.705	1.743
	GO:0002376, GO:0044459, GO:0050865	140	48	0.881	3.000
	GO:0002429, GO:0071944	138	47	0.881	2.980
	GO:0006066, GO:0003824, GO:0044238	112	38	0.882	2.969

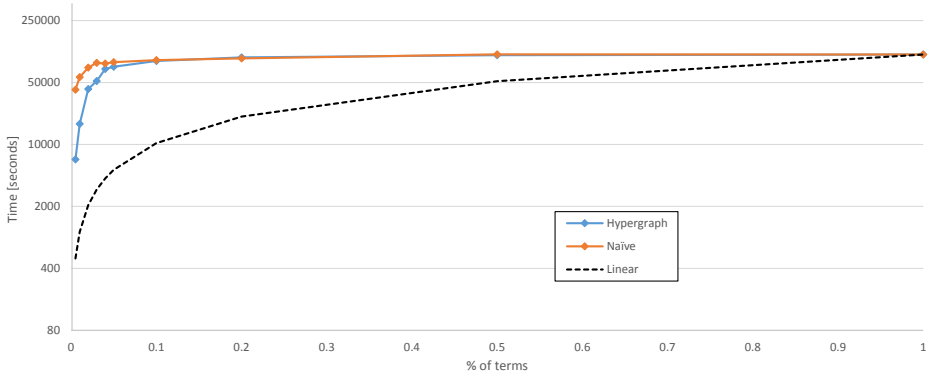

Figure 8: The times (in seconds) taken by the Aleph algorithm for different sizes of the background knowledge to discover rules in the third round of experiments on the ALL data set. Note the logarithmic scale of the y axis of the graph. A linear trendline (dashed) is drawn for comparison.

than even the best rule discovered without ontology shrinking. The main drawback of this result at this point is the fact that the ideal background knowledge size at which this occurs is different for different shrinkage methods, meaning that for a new data set we have to tune the background knowledge size.

6. Conclusions and further work

This paper presents an efficient approach to semantic data mining that uses network analysis for shrinking of background knowledge to significantly reduce the search space and make semantic data mining practically applicable. We tested several variants of combining network analysis with semantic data mining. The result is a four-step NetSDM framework for data analysis: background knowledge conversion, network node importance calculation, network shrinkage, and semantic data mining. We tested NetSDM using two rule learning algorithms: Hedwig and Aleph. In all the setups Hedwig and Aleph were used in their standard modes of operation: Hedwig was used for subgroup discovery and Aleph was used for classification rule learning. Nevertheless, as output, both algorithms return a set of rules explaining the target class examples.

The results of using Hedwig as the semantic data mining algorithm for rule learning in the final step of the NetSDM methodology show that using a direct network conversion, the Personalized PageRank function to estimate term importance, and an advanced version of network shrinkage (using transitivity of underlying relations) provides the most consistent results. Using this setup, the entire algorithm was able to discover the same rules as the original Hedwig algorithm, being faster by a factor of almost 100. If we replace the advanced node removal function with the naïve version, the time savings are less predictable, but the algorithm discovers different rules than if run on the unpruned data set. Consequently, when applying Hedwig the advanced node removal strategy is recommended. The results of comparing NetSDM by taking/not taking network edge directions into account were less conclusive and require further experiments for detailed analyisis.

Using Aleph as the SDM algorithm also yielded interesting results. Using any of the three shrinking methods, our results show that on largely reduced background knowledge, Aleph is capable of discovering high quality rules explaining the data set. Rule discovery in this case is faster by a factor of 2 which is a significant improvement. Using larger background knowledge, NetSDM is still capable of shrinking the background knowledge and (with direct network conversion and the Personalized PageRank scoring function) achieve a noticeable (15%) speed-up in the run time of the Aleph algorithm.

In summary, the purpose of this work is to examine the effect of using network analysis inspired network shrinkage on semantic data mining algorithms in the context of the proposed NetSDM methodology. Due to the different modes of operation of algorithms Hedwig and Aleph (performing subgroup discovery and classification rule learning, respectively), comparing their results does not provide additional insights regarding their operation. Nevertheless, comparing Aleph and Hedwig, our results show that the Hedwig algorithm is very consistent in that Hedwig can discover the same rules, but in a much shorter time, when applying NetSDM to reduce the background knowledge. In contrast, the speed-up—while still noticeable—was much less pronounced when using NetSDM with Aleph. The quality of the discovered rules also decreased more quickly with Aleph used as the final algorithm of NetSDM.

In future work, we plan a more comprehensive examination of how the performance of NetSDM compares to enrichment analysis based methods like SEGS (Trajkovski et al., 2008b) and SegMine (Podpečan et al., 2011). We plan further experiments with different methods for network reduction. For example, other network ranking methods or even other

network analysis methods, such as community detection, could be used to identify the most relevant parts of the background knowledge. The existing two methods, PageRank and node2vec, contain parameters (especially parameters p and q of node2vec) which may affect the resulting scoring function.

Additionally, a non-deterministic approach to background knowledge shrinking could be used instead of the hard shrinkage coefficient used in the current experiments. With a non-deterministic approach, the scores of background knowledge terms may represent (non-scaled) probabilities of sampling a given node. Such an approach has the benefit of not eliminating any background knowledge term outright. However, the probabilistic nature of the sampling would require multiple runs of the entire algorithm, including the SDM rule learner, causing additional computations that would negate the improvements in performance achieved by network shrinking. Alternatively, we could more tightly integrate network analysis scores and the SDM algorithms. Instead of using the relevance scores to prune the ontology and feed the pruned ontology to SDM algorithms, we could use the scores, calculated by the network analysis algorithms, to guide the search in SDM algorithms. The main drawback of this approach would be the lack of modularity: while it is now relatively simple to replace one SDM algorithm in NetSDM with another (for example, replacing Hedwig with Aleph), this would no longer be possible if the scores were an integral part of a SDM algorithm.

We plan applications of the NetSDM methodology to larger data sets. Our experiments so far have been limited to hierarchies (experiments described in Appendix A) and the Gene Ontology, which is a big ontology but relatively simple in structure and logic relationships (in the experiments, we used only two relationships out of five relationships available in GO). Without our shrinking approach, Hedwig required several hours to discover interesting rules using this ontology, and therefore using larger ontologies was not feasible without dramatically reducing the parameters guiding the search. However, using the NetSDM, the algorithm can now be applied to a much wider range of background knowledge sets, including more complex ontologies and large knowledge graphs, such as the BioMine graph (Eronen and Toivonen, 2012) of biological entities and connections between them, as well as the Bio2RDF knowledge graph (Callahan et al., 2013), two knowledge graphs we intend to examine in our further work.

Acknowledgments

We express our thanks to Michel Dumontier for joint work in defining this challenging research problem. We are grateful to Anže Vavpetič for useful discussions and support in using the Hedwig algorithm, to Ashwin Srinivasan for the support and advice in running the experiments with the Aleph algorithm, and to anonymous reviewers whose constructive feedback helped us to improve this manuscript. We acknowledge the financial support from the Slovenian Research Agency through core research programmes P2-0103 and P6-0411 and project *HinLife: Analysis of Heterogeneous Information Networks for Knowledge Discovery in Life Sciences* (J7-7303) and *Semantic Data Mining for Linked Open Data* (financed under the ERC Complementary Scheme, N2-0078).

References

Prem Raj Adhikari, Anže Vavpetič, Jan Kralj, Nada Lavrač, and Jaakko Hollmén. Explaining mixture models through semantic pattern mining and banded matrix visualization. *Machine Learning*, 105(1):3–39, 2016.

Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 487–499, San Francisco, California, USA, 1994.

Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, J. Michael Cherry, Allan P. Davis, Kara Dolinski, Selina S. Dwight, Janan T. Eppig, et al. Gene Ontology: Tool for the unification of biology. *Nature Genetics*, 25(1):25–29, 2000.

Alex Bavelas. Communication patterns in task-oriented groups. *Journal of the Acoustical Society of America*, 22:725–730, 1950.

Ronald S. Burt and Michael J. Minor. *Applied Network Analysis: A Methodological Introduction*. Sage Publications, 1983.

Alison Callahan, Jose Cruz-Toledo, Peter Ansell, and Michel Dumontier. Bio2RDF Release 2: Improved coverage, interoperability and provenance of life science linked data. In *ESWC*, volume 7882 of *Lecture Notes in Computer Science*, pages 200–212. Springer, 2013.

Alison Callahan, Juan José Cifuentes, and Michel Dumontier. An evidence-based approach to identify aging-related genes in Caenorhabditis elegans. *BMC Bioinformatics*, 16(1):1, 2015.

Sabina Chiaretti, Xiaochun Li, Robert Gentleman, Antonella Vitale, Marco Vignetti, Franco Mandelli, Jerome Ritz, and Robin Foa. Gene expression profile of adult T-cell acute lymphocytic leukemia identifies distinct subsets of patients with different response to therapy and survival. *Blood*, 103(7):2771–2778, 2004.

Lynne S. Cox and Richard Faragher. From old organisms to new molecules: Integrative biology and therapeutic targets in accelerated human ageing. *Cellular and Molecular Life Sciences*, 64(19-20):2620–2641, 2007.

F. Crestani. Application of spreading activation techniques in information retrieval. *Artificial Intelligence Review*, 11(6):453–482, December 1997.

L. De Raedt. *Logical and Relational Learning*. Springer, 2008.

Dejing Dou, Hao Wang, and Haishan Liu. Semantic data mining: A survey of ontology-based approaches. In *Semantic Computing (ICSC), 2015 IEEE International Conference on*, pages 244–251. IEEE, 2015.

Sašo Džeroski and Nada Lavrač, editors. *Relational Data Mining*. Springer, 2001.

Lauri Eronen and Hannu Toivonen. BioMine: Predicting links between biological entities using network models of heterogeneous databases. *BMC Bioinformatics*, 13:119, 2012.

Linton C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40: 35–41, 1977.

Linton C. Freeman. Centrality in social networks conceptual clarification. *Social Networks*, 1(3):215–239, 1979.

Johannes Fürnkranz, Dragan Gamberger, and Nada Lavrač. *Foundations of Rule Learning*. Springer, 2012.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, California, USA, 2016.

Miha Grčar, Nejc Trdin, and Nada Lavrač. A methodology for mining document-enriched heterogeneous information networks. *The Computer Journal*, 56(3):321–335, 2013.

Nicola Guarino, Daniel Oberle, and Steffen Staab. What Is an Ontology? In *Handbook on Ontologies*, pages 1–17. Springer, 2009.

Robert Hoehndorf, Michel Dumontier, and Georgios V. Gkoutos. Identifying aberrant pathways through integrated analysis of knowledge in pharmacogenomics. *Bioinformatics*, 28 (16):2169–2175, 2012.

Da Wei Huang, Brad T. Sherman, and Richard A. Lempicki. Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources. *Nature Protocols*, 4(1): 44–57, 2008.

Wilhelmiina Hämäläinen. *Efficient search for statistically significant dependency rules in binary data*. PhD thesis, Department of Computer Science, University of Helsinki, Finland, 2010.

Glen Jeh and Jennifer Widom. SimRank: A measure of structural-context similarity. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 538–543. ACM, 2002.

Mikhail Jiline, Stan Matwin, and Marcel Turcotte. Annotation concept synthesis and enrichment analysis: A logic-based approach to the interpretation of high-throughput experiments. *Bioinformatics*, 27(17):2391–2398, 2011.

Leo Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1): 39–43, 1953.

Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

Willi Klösgen. Explora: A multipattern and multistrategy discovery assistant. In *Advances in Knowledge Discovery and Data Mining*, pages 249–271. American Association for Artificial Intelligence, 1996.

Risi Imre Kondor and John D. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *Proceedings of the 19th International Conference on Machine Learning*, pages 315–322, 2002.

Nada Lavrač and Anže Vavpetič. Relational and semantic data mining. In *Proceedings of the Thirteenth International Conference on Logic Programming and Nonmonotonic Reasoning*, pages 20–31, Lexington, KY, USA, 2015.

Nada Lavrač, Branko Kavšek, Peter Flach, and Ljupčo Todorovski. Subgroup discovery with CN2-SD. *Journal of Machine Learning Research*, 5:153–188, 2004.

Agnieszka Lawrynowicz and Jedrzej Potoniec. Fr-ONT: An algorithm for frequent concept mining with formal ontologies. In *Foundations of Intelligent Systems, Proceedings of 19th International Symposium on Methodologies for Intelligent Systems (2011)*, volume 6804 of *Lecture Notes in Computer Science*, pages 428–437, 2011.

Paea LePendu, Srinivasan V. Iyer, Anna Bauer-Mehren, Rave Harpaz, Jonathan M. Mortensen, Tanya Podchiyska, Todd A. Ferris, and Nigam H Shah. Pharmacovigilance using clinical notes. *Clinical Pharmacology & Therapeutics*, 93(6):547–555, 2013.

Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD'98)*, pages 80–86, 1998.

Haishan Liu, Dejing Dou, Ruoming Jin, Paea LePendu, and Nigam Shah. Mining biomedical ontologies and data using RDF hypergraphs. In *Proceedings of the 12th International Conference on Machine Learning and Applications (ICMLA), 2013*, volume 1, pages 141–146. IEEE, 2013.

Svetlana Lyalina, Bethany Percha, Paea LePendu, Srinivasan V. Iyer, Russ B. Altman, and Nigam H. Shah. Identifying phenotypic signatures of neuropsychiatric disorders from electronic medical records. *Journal of the American Medical Informatics Association*, 20 (e2):e297–e305, 2013.

Donna Maglott, Jim Ostell, Kim D. Pruitt, and Tatiana Tatusova. Entrez Gene: Gene-centered information at NCBI. *Nucleic Acids Research*, 33(Database issue):D54–D58, 2005.

Stephen Muggleton. Inverse entailment and Progol. *New generation computing*, 13(3-4): 245–286, 1995.

Athanasios N Nikolakopoulos and John D Garofalakis. NCDawareRank: A novel ranking method that exploits the decomposable structure of the web. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, pages 143–152. ACM, 2013.

Hiroyuki Ogata, Susumu Goto, Kazushige Sato, Wataru Fujibuchi, Hidemasa Bono, and Minoru Kanehisa. KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Research*, 27(1):29–34, 1999.

David Page, Vítor Santos Costa, Sriraam Natarajan, Aubrey Barnard, Peggy Peissig, and Michael Caldwell. Identifying adverse drug events by relational learning. In *Proceedings of the Twenty-sixth AAAI Conference on Artificial Intelligence*, volume 2012, page 790, Toronto, Canada, 2012.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, November 1999.

Peggy L Peissig, Vitor Santos Costa, Michael D Caldwell, Carla Rottscheit, Richard L Berg, Eneida A Mendonca, and David Page. Relational machine learning for electronic health record-driven phenotyping. *Journal of Biomedical Informatics*, 52:260–270, 2014.

Gregory Piatetsky-Shapiro. Discovery, analysis, and presentation of strong rules. In *Knowledge Discovery in Databases*, pages 229–248. Menlo Park, CA: AAI/MIT, 1991.

Vid Podpečan, Nada Lavrač, Igor Mozetič, Petra Kralj Novak, Igor Trajkovski, Laura Langohr, Kimmo Kulovesi, Hannu Toivonen, Marko Petek, Helena Motaln, et al. SegMine workflows for semantic microarray data analysis in Orange4WS. *BMC Bioinformatics*, 12(1):416, 2011.

Monika Puzianowska-Kuznicka and Jacek Kuznicki. Genetic alterations in accelerated ageing syndromes: Do they play a role in natural ageing? *The International Journal of Biochemistry & Cell Biology*, 37(5):947–960, 2005.

Steffen Rendle. Scaling factorization machines to relational data. *Proceedings of the VLDB Endowment*, 6(5):337–348, March 2013. ISSN 2150-8097.

Christos Sotiriou, Pratyaksha Wirapati, Sherene Loi, Adrian Harris, Steve Fox, Johanna Smeds, Hans Nordgren, Pierre Farmer, Viviane Praz, Benjamin Haibe-Kains, et al. Gene expression profiling in breast cancer: Understanding the molecular basis of histologic grade to improve prognosis. *Journal of the National Cancer Institute*, 98(4):262–272, 2006.

Ashwin Srinivasan. The Aleph Manual, 1999. Available at http://www.cs.ox.ac.uk/activities/machinelearning/Aleph/aleph.

Yizhou Sun and Jiawei Han. *Mining Heterogeneous Information Networks: Principles and Methodologies*. Morgan & Claypool Publishers, 2012.

Hannah Tipney and Lawrence Hunter. An introduction to effective use of enrichment analysis software. *Human Genomics*, 4(3):1, 2010.

Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. Fast random walk with restart and its applications. In *Proceedings of the Sixth International Conference on Data Mining*, pages 613–622, Washington, DC, USA, 2006.

Igor Trajkovski, Nada Lavrač, and Jakub Tolar. SEGS: Search for enriched gene sets in microarray data. *Journal of Biomedical Informatics*, 41(4):588–601, 2008a.

Igor Trajkovski, Filip Železný, Nada Lavrač, and Jakub Tolar. Learning relational descriptions of differentially expressed gene groups. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 38(1):16–25, 2008b.

Oron Vanunu, Oded Magger, Eytan Ruppin, Tomer Shlomi, and Roded Sharan. Associating genes and protein complexes with disease via network propagation. *PLoS Computational Biology*, 6(1), 2010.

Anže Vavpetič, Vid Podpečan, and Nada Lavrač. Semantic subgroup explanations. *Journal of Intelligent Information Systems*, 42(2):233–254, 2014.

Anže Vavpetič and Nada Lavrač. Semantic subgroup discovery systems and workflows in the SDM–toolkit. *The Computer Journal*, 56(3):304–320, 2013.

Anže. Vavpetič, Petra Kralj Novak, Miha Grčar, Igor Mozetič, and Nada Lavrač. Semantic data mining of financial news articles. In *Proceedings of Sixteenth International Conference on Discovery Science (DS 2013)*, volume 8140 of *Lecture Notes in Computer Science*, pages 294–307, Singapore, 2013.

Monika Žáková, Filip Železný, Javier A. Sedano, Cyril Masia Tissot, Nada Lavrač, Petr Kremen, and Javier Molina. Relational data mining applied to virtual engineering of product designs. In *Proceedings of the 16th International Conference on Inductive Logic Programming (ILP'06)*, pages 439–453, Santiago de Compostela, Spain, 2006.

Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2005.

Stefan Wrobel. An algorithm for multi-relational discovery of subgroups. In *Proceedings of the First European Conference on Principles of Data Mining and Knowledge Discovery (PKDD '97)*, pages 78–87. Springer, 1997.

Wenpu Xing and Ali Ghorbani. Weighted PageRank algorithm. In *Proceedings of the 2nd Annual Conference on Communication Networks and Services Research*, pages 305–314. IEEE, 2004.

Liang Zhang, Bingpeng Ma, Guorong Li, Qingming Huang, and Qi Tian. PL-ranking: A novel ranking method for cross-modal retrieval. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 1355–1364. ACM, 2016.

Appendix A

The Hedwig algorithm was previously used on several publicly available data sets (Adhikari et al., 2016). To show generality of the proposed NetSDM approach, we tested the performance of the NetSDM algorithm on the same data sets. The results, shown in Tables 19, 21, and 20, show similar trends as for the biological data sets which comprise the main part of this paper. The use of NetSDM can drastically decrease the running time of the algorithm with little to no decrease in the quality of the discovered rules. The speed-ups, achieved on the reported data sets, are around the factor of 5, which is less than for the Gene Ontology background knowledge. This lower, though still very promising speed-up can be explained by the fact that (1) the background knowledge in this case is smaller, meaning that less pruning can take place, (2) because of smaller background knowledge, a larger portion of Hedwig's running time consists of pre-processing and post-processing of the data, the tasks that are not affected by background knowledge pruning.

Table 19: Results of NetSDM on the Cities data set using the P-PR scoring function, regular network conversion, and advanced node removal.

Shrinkage coefficient [%]	Time taken [s]	Rule	Coverage	TP	Precision	Lift
3	2.50	City, Municipality, Capital	50	44	0.880	3.054
		City, Municipality, Center	58	51	0.879	3.052
		City, Center	56	49	0.875	3.037
4	3.85	PopulatedPlace, City, Place, Center	49	49	1	3.47
		Underspecified, City, Place, Center	48	48	1	3.47
		PopulatedPlace, City, Place, Capital	42	42	1	3.47
100	17	PopulatedPlace, City, Place, Center	49	49	1	3.47
		Underspecified, City, Place, Center	48	48	1	3.47
		PopulatedPlace, City, Place, Capital	42	42	1	3.47

Table 20: Results of NetSDM on the NY Daily data set using the P-PR scoring function, regular network conversion, and advanced node removal.

Shrinkage coefficient [%]	Time taken [s]	Rule	Coverage	TP	Precision	Lift
5	57.44	Place, LivingPeople,Location	95	95	1	7.196
		Agent, District,CausalAgent,Place,Person	92	92	0.879	7.196
		Place, LivingPeople,Region	92	92	0.879	7.196
10	84.95	Agent, Location,CausalAgent,Place,Person	102	102	1	7.196
		Agebnt Location,CausalAgent,PopulatedPlace	100	100	1	7.196
		Agent, Region, CausalAgent, Place,Person	97	97	1	7.196
100	297.99	Agent, Location,CausalAgent,Place,Person	102	102	1	7.196
		Agebnt Location,CausalAgent,PopulatedPlace	100	100	1	7.196
		Agent, Region, CausalAgent, Place,Person	97	97	1	7.196

Table 21: Results of NetSDM on the Tweets data set using the P-PR scoring function, regular network conversion and advanced node removal.

Shrinkage coefficient [%]	Time taken [s]	Rule	Coverage	TP	Precision	Lift
0.5	13	Athlete, Agent	139	112	0.806	4.268
		Person, Athlete	147	117	0.796	4.215
		CausalAgent, Athlete	149	117	0.785	4.159
10	24	Athlete, Agent	139	112	0.806	4.268
		Athlete, Contestant	141	113	0.801	4.245
		Athlete, Person	147	117	0.796	4.215
100	113	Athlete, Agent	139	112	0.806	4.268
		Athlete, Contestant	141	113	0.801	4.245
		Athlete, Person	147	117	0.796	4.215

Appendix B

Table 22: Results of NetSDM on the ALL data set using the Personalized PageRank scoring function, direct network conversion, and advanced node removal. For each percentage of background knowledge terms left in the ontology, we list only the top 3 rules discovered and also report on the properties of the entire set of rules (Theory) discovered by Aleph.

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
0.5	Theory	2561	489	0.704	1.671
	GO:0005783 GO:0005789 GO:0009058	63	19	0.883	2.639
	GO:0044459 GO:0002376 GO:0007275	137	38	0.879	2.427
	GO:0006950 GO:0015630	79	19	0.881	2.104
1	Theory	2673	515	0.697	1.686
	GO:0050851 GO:0071944	82	36	0.885	3.841
	GO:0006066 GO:0044238 GO:0003824	122	40	0.881	2.869
	GO:0001775 GO:0006793	121	35	0.880	2.531
2	Theory	2814	547	0.688	1.701
	GO:0002250 GO:0044464	119	33	0.880	2.426
	GO:0045321 GO:0005829	105	28	0.880	2.333
	GO:1901360 GO:0008610	125	33	0.879	2.310
3	Theory	2879	549	0.681	1.669
	GO:0016021 GO:0002684 GO:0044459	137	43	0.880	2.746
	GO:0045321 GO:0044459	117	35	0.880	2.618
	GO:0002250 GO:0044464	119	33	0.880	2.426
4	Theory	2816	547	0.688	1.700
	GO:0016021 GO:0002684 GO:0044459	137	43	0.880	2.746
	GO:0045321 GO:0005886	117	35	0.880	2.618
	GO:0002250 GO:0044464	119	33	0.880	2.426

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
5	Theory	2852	543	0.683	1.666
	GO:0002684 GO:0016021 GO:0044459	137	43	0.880	2.746
	GO:0045321 GO:0005886	117	35	0.880	2.618
	GO:0002250 GO:0005623	119	33	0.880	2.426
10	Theory	2760	529	0.690	1.677
	GO:0008610 GO:0044283	127	41	0.881	2.825
	GO:0045321 GO:0005886	117	35	0.880	2.618
	GO:0002250 GO:0005623	119	33	0.880	2.426
20	Theory	2620	522	0.705	1.743
	GO:0008610 GO:0044283	127	41	0.881	2.825
	GO:0045321 GO:0005886	117	35	0.880	2.618
	GO:0010008 GO:0032991	97	28	0.881	2.526
50	Theory	2576	510	0.707	1.732
	GO:0008610 GO:0044283	127	41	0.881	2.825
	GO:0045321 GO:0005886	116	34	0.880	2.565
	GO:0010008 GO:0032991	97	28	0.881	2.526
100	Theory	2568	510	0.708	1.738
	GO:0008610 GO:0044283	127	41	0.881	2.825
	GO:0045321 GO:0005886	116	34	0.880	2.565
	GO:0010008 GO:0032991	97	28	0.881	2.526

Table 23: Results of NetSDM on the ALL data set using the Personalized PageRank scoring function, hypergraph network conversion, and advanced node removal. For each percentage of background knowledge terms left in the ontology, we list only the top 3 rules discovered and also report on the properties of the entire set of rules (Theory) discovered by Aleph.

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
0.5	Theory	2,577	492	0.703	1.671
	GO:1901362 GO:0044255 GO:0044281	84	26	0.882	2.708
	GO:0005874 GO:0005515 GO:0044707	48	14	0.883	2.552
	GO:0007166 GO:0050776 GO:0016021	138	40	0.879	2.536
1	Theory	2,627	515	0.702	1.715
	GO:1901362 GO:0044255 GO:0044281	84	26	0.882	2.708
	GO:0007166 GO:0050776 GO:0016021	138	40	0.879	2.536
	GO:0032403 GO:0005886 GO:0032991	114	30	0.880	2.303
2	Theory	2,534	516	0.713	1.782
	GO:0050870 GO:0005886	87	35	0.884	3.520
	GO:0007166 GO:0050778 GO:0044459	141	48	0.881	2.979
	GO:0007017 GO:0006950	48	15	0.884	2.734
3	Theory	2,728	539	0.696	1.729
	GO:0007166 GO:0050778 GO:0044459	141	48	0.881	2.979
	GO:0050863 GO:0016020	140	46	0.880	2.875
	GO:0006066 GO:0003824	129	40	0.880	2.713
4	Theory	2,548	513	0.711	1.762
	GO:0016021 GO:0002684 GO:0005886	137	43	0.880	2.746
	GO:0006066 GO:0003824	129	40	0.880	2.713
	GO:0001775 GO:0019899	132	35	0.879	2.320

% of terms	Rule	Cov	Pos	Acc	Lift
5	Theory	2,522	512	0.714	1.776
	GO:0016021 GO:0002684 GO:0005886	137	43	0.880	2.746
	GO:0006066 GO:0003824	129	40	0.880	2.713
	GO:0001775 GO:0019899	132	35	0.879	2.320
10	Theory	2,642	538	0.706	1.782
	GO:0050865 GO:0002376 GO:0005886	144	49	0.880	2.977
	GO:0006066 GO:0003824 GO:0044238	121	40	0.881	2.893
	GO:0051606 GO:0006950	46	14	0.884	2.663
20	Theory	2,675	527	0.700	1.724
	GO:0044459 GO:0002684 GO:0050865	139	47	0.881	2.959
	GO:0006066 GO:0003824 GO:0044238	120	39	0.881	2.844
	GO:0051606 GO:0006950	46	14	0.884	2.663
50	Theory	2,648	527	0.703	1.741
	GO:0002376 GO:0044459 GO:0050865	140	48	0.881	3.000
	GO:0002429 GO:0071944	138	47	0.881	2.980
	GO:0006066 GO:0003824 GO:0044238	118	38	0.881	2.818
100	Theory	2,616	521	0.705	1.743
	GO:0002376 GO:0044459 GO:0050865	140	48	0.881	3.000
	GO:0002429 GO:0071944	138	47	0.881	2.980
	GO:0006066 GO:0003824 GO:0044238	112	38	0.882	2.969

Table 24: Results of NetSDM on the breast cancer data set using the Personalized PageRank scoring function, direct network conversion, and naïve node removal. For each percentage of background knowledge terms left in the ontology, we list only the top 3 rules discovered and also report on the properties of the entire set of rules (Theory) discovered by Aleph.

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
0.5	Theory	6,439	659	0.463	1.177
	GO:0070062 GO:0005576 GO:0005634	84	19	0.909	2.602
	GO:0005794 GO:0044428	162	29	0.904	2.060
	GO:0005743 GO:1901363	109	18	0.907	1.900
1	Theory	5,196	575	0.558	1.273
	GO:0051301 GO:0050794	217	41	0.901	2.174
	GO:0005525 GO:0044421	110	20	0.907	2.092
	GO:0046983 GO:0035639 GO:0044424	118	21	0.906	2.048
2	Theory	3,902	522	0.662	1.539
	GO:0000082 GO:0005515	89	21	0.909	2.715
	GO:0008283 GO:0006139	185	37	0.903	2.301
	GO:0008201 GO:0044763 GO:0050896	70	14	0.909	2.301
3	Theory	5,078	603	0.573	1.366
	GO:1901363 GO:0032446	142	35	0.907	2.836
	GO:0005634 GO:0005794 GO:0044446	117	25	0.907	2.458
	GO:0003723 GO:0031967	101	21	0.908	2.392
4	Theory	4,605	582	0.611	1.454
	GO:0005634 GO:0005794 GO:0044446	117	25	0.907	2.458
	GO:0012501 GO:0043232	125	26	0.907	2.393
	GO:0044430 GO:0000278	195	40	0.903	2.360

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
5	Theory	5,137	611	0.569	1.368
	GO:1903047 GO:0000166	129	31	0.907	2.765
	GO:0005730 GO:0051179	94	22	0.909	2.693
	GO:0050896 GO:0000785	169	34	0.904	2.315
10	Theory	4,629	588	0.610	1.461
	GO:0006890 GO:0043229	60	15	0.910	2.876
	GO:1903047 GO:0000166	129	31	0.907	2.765
	GO:0032993	88	21	0.909	2.746
20	Theory	3,475	528	0.701	1.748
	GO:0000910 GO:0043226	50	12	0.911	2.761
	GO:0005654 GO:0008283	158	37	0.906	2.694
	GO:0032984 GO:0043232	87	19	0.909	2.513
50	Theory	3,006	526	0.742	2.013
	GO:0051301 GO:0032268	85	26	0.910	3.519
	GO:0005730 GO:0051179	94	22	0.909	2.693
	GO:1903561 GO:0048468	95	22	0.909	2.664
100	Theory	2,637	484	0.767	2.112
	GO:0032993 GO:0009987	76	21	0.910	3.179
	GO:0015931 GO:0031967	43	11	0.911	2.943
	GO:1903047 GO:0035639	120	30	0.908	2.876

Table 25: Results of NetSDM on the breast cancer data set using the Personalized PageRank scoring function, direct network conversion, and advanced node removal. For each percentage of background knowledge terms left in the ontology, we list only the top 3 rules discovered and also report on the properties of the entire set of rules (Theory) discovered by Aleph.

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
0.5	Theory	2,455	422	0.772	1.978
	GO:0000166 GO:0000278	129	31	0.907	2.765
	GO:0005730 GO:0051179	94	22	0.909	2.693
	GO:0051174 GO:0007049	68	15	0.910	2.538
1	Theory	2,724	474	0.757	2.002
	GO:1903047 GO:0035639	120	30	0.908	2.876
	GO:0005730 GO:0051179	94	22	0.909	2.693
	GO:0070647 GO:0033554	116	25	0.907	2.480
2	Theory	2,761	484	0.756	2.017
	GO:0005829 GO:0044106	31	10	0.912	3.711
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0044770 GO:0031981	126	34	0.908	3.105
3	Theory	2,749	497	0.759	2.080
	GO:0009308 GO:0005829	34	11	0.912	3.722
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0044770 GO:0031981	126	34	0.908	3.105
4	Theory	2,758	498	0.758	2.077
	GO:0005829 GO:0009308	34	11	0.912	3.722
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0044770 GO:0031981	126	34	0.908	3.105
5	Theory	2,765	492	0.757	2.047
	GO:0005829 GO:0009308	34	11	0.912	3.722
	GO:0015630 GO:0007059	70	21	0.911	3.452
	GO:0044770 GO:0031981	126	34	0.908	3.105
10	Theory	2,838	501	0.752	2.031
	GO:0005829 GO:0009308	34	11	0.912	3.722
	GO:0043232 GO:1903047 GO:0051276	75	21	0.910	3.221
	GO:0044770 GO:0031981	126	34	0.908	3.105
20	Theory	2,715	482	0.759	2.043
	GO:0005829 GO:0009308	34	11	0.912	3.722
	GO:0015630 GO:0007059	70	21	0.911	3.452
	GO:0044770 GO:0031981	126	34	0.908	3.105
50	Theory	2,880	510	0.750	2.037
	GO:0005829 GO:0009308	34	11	0.912	3.722
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0044770 GO:0031981	126	34	0.908	3.105
100	Theory	2,755	498	0.759	2.080
	GO:0005829 GO:0009308	34	11	0.912	3.722
	GO:0051276 GO:0000278	136	41	0.908	3.468
	GO:0044770 GO:0031981	126	34	0.908	3.105

Table 26: Results of NesSDM on the breast cancer data set using the Personalized PageRank scoring function, hypergraph network conversion, and advanced node removal. For each percentage of background knowledge terms left in the ontology, we list only the top 3 rules discovered and also report on the properties of the entire set of rules (Theory) discovered by Aleph.

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
5	Theory	2,392	430	0.779	2.068
	GO:0008283 GO:1901360	32	10	0.912	3.595
	GO:0005829 GO:1903047 GO:0051301	100	28	0.909	3.221
	GO:0031410 GO:0015031 GO:0005789	43	12	0.911	3.211
1	Theory	2,431	439	0.777	2.078
	GO:0005794 GO:0005634 GO:0044422	85	22	0.909	2.978
	GO:0006260 GO:0032991	82	21	0.910	2.946
	GO:0005654 GO:0003674 GO:1902589	96	24	0.909	2.876
2	Theory	2,721	492	0.761	2.080
	GO:0008283 GO:0006725	32	10	0.912	3.595
	GO:0051276 GO:0000278	136	41	0.908	3.468
	GO:0032993 GO:0009987 GO:0005634	73	21	0.910	3.310
3	Theory	2,791	491	0.754	2.024
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0010604 GO:0051726	86	21	0.909	2.809
	GO:0016567 GO:1901363	114	26	0.908	2.624
4	Theory	2,928	505	0.745	1.984
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0031323 GO:0051726	87	21	0.909	2.777
	GO:0016567 GO:1901363	114	26	0.908	2.624

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
5	Theory	2,920	499	0.744	1.966
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0005730 GO:0032879	81	20	0.909	2.841
	GO:0044237 GO:0051726	86	18	0.909	2.408
10	Theory	2,885	497	0.747	1.982
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0080090 GO:0051726	84	21	0.909	2.876
	GO:0005488 GO:0022411 GO:0043232	115	28	0.908	2.801
20	Theory	2,694	470	0.759	2.007
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0005730 GO:0051179	94	22	0.909	2.693
	GO:0043226 GO:0051726	82	19	0.909	2.666
50	Theory	2,912	503	0.746	1.987
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0000280 GO:0043170	78	22	0.910	3.245
	GO:0043226 GO:0051726	82	19	0.909	2.666
100	Theory	2,927	505	0.745	1.985
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0003677 GO:0070647 GO:0044424	89	21	0.909	2.715
	GO:0043226 GO:0051726	82	19	0.909	2.666