HETEROGENEOUS INFORMATION NETWORK ANALYSIS FOR SEMANTIC DATA MINING

Jan Kralj

Doctoral Dissertation
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL

Jan Kralj

HETEROGENEOUS INFORMATION NETWORK ANALYSIS FOR SEMANTIC DATA MINING

Doctoral Dissertation

ANALIZA HETEROGENIH INFORMACIJSKIH OMREŽIJ ZA SEMANTIČNO RUDARJENJE PODATKOV

Doktorska disertacija

Supervisor: Prof. Dr. Nada Lavrač

Co-Supervisor: Prof. Dr. Marko Robnik-Šikonja

Ljubljana, Slovenia, October 2017

To Teja

Acknowledgments

No man is an island entire of itself; every man is a piece of the continent, a part of the main. Similarly, no man's work is ever his and his alone. Without the help of many people, this work would never have been finished or even conceived of. Their support and help cannot be understated.

First of all, I want to thank my supervisor Nada Lavrač and co-supervisor Marko Robnik-Šikonja for all their knowledge, patience, research ideas, support and energy in the endless reviews, rewrites and corrections that were needed to direct not only this work, but my research path in general.

I also wish to thank the funding bodies that financially supported my research: The Slovenian Research Agency, the Department of Knowledge Technologies at the Jožef Stefan Institute, the Jožef Stefan International Postgraduate School, and the European Commission for funding the research projects ConCreTe, WHIM and HBP in which I was involved.

I am grateful to all the people I worked with during these years, especially Blaž Škrlj, Jaakko Hollmén, Prem Raj Adhikari and Anita Valmarska, and also to my work colleagues at the Department of Knowledge Technologies for providing an encouraging and friendly working environment. Thank you to Mili Bauer and Tina Anžič for the help in the endless battle with paperwork, and thank you to Vid Podpečan and Miha Grčar for being the shoulders of giants and providing me with all the knowledge about the technologies used in this work as a starting point. Also, a special thanks goes out to Janez Kranjc, Anže Vavpetič and Matic Perovšek, not only for providing all the help needed with the Hedwig and ClowdFlows technologies, but also for the fun moments shared at lunch breaks, conference trips, and other events, and a nickname that will probably stay with me for the rest of my life.

Finally, I wish to thank my family and friends for the emotional support and encouragement through the years, and to apologize to them for the not-too-frequent times my work made me cranky, nervous or just distant. My biggest thanks goes to Teja, who had to carry the brunt of my mood swings and nevertheless gave me all the caring and support I needed throughout my studies.

Abstract

Most data mining approaches focus on the analysis of data represented in a table with a fixed number of columns. More complex means of storing data include relational databases and information networks, for which the standard data mining algorithms are not well suited. The thesis presents several contributions to the area of mining complex data.

The first part of the thesis addresses the problem of node classification in heterogeneous networks (i.e. networks containing nodes and edges of different types). We focus on a network decomposition algorithm that constructs one or more homogeneous networks out of a heterogeneous network. We propose several heuristics designed to evaluate the significance of so called intermediary nodes of the heterogeneous network. The motivation behind the developed heuristics originated in heuristics for bag-of-words vector construction in text mining. The heuristics were designed to penalize the effect of intermediary nodes, connecting too many nodes, and intermediary nodes, connecting many nodes with different class labels. We tested the new heuristics on several data sets and showed that using them improves classification of network nodes. The new approach is implemented in the newly developed HinMine methodology.

Second, the thesis presents our work on combining network analysis methods with semantic data mining (SDM) algorithms for explaining data sets annotated by ontology terms. The proposed methodology, named NetSDM, uses network node ranking methods to evaluate the significance of the terms in the background knowledge ontology. We tested the methodology on several data sets, including a biological data set containing gene expression data for potato plants, infected with a virus. Our results show that using NetSDM can speed up the existing state of the art semantic data mining algorithm Hedwig by a factor of one hundred, discovering high quality rules describing the data set in minutes instead of hours.

Third, we present a methodology for data visualization using banded matrices. The methodology improves the comprehensibility of semantic data mining algorithm outputs by using banded matrix algorithms to uncover a hidden structure in the data. As we discover this structure independently of the SDM algorithm, we can use the results in an easy-to-understand informative visual overlay of both results.

Finally, the algorithms presented in the thesis were made publicly available. The HinMine and NetSDM methodologies are available as workflows in the ClowdFlows online data mining platform, and the banded matrix algorithms are available as Python packages on GitHub.

Povzetek

Večina pristopov podatkovnega rudarjenja je namenjena analizi podatkov, predstavljenih v obliki tabele z vnaprej določenim številom stolpcev. Alternativo predstavljajo kompleksnejši načini shranjevanja podatkov kot so relacijske podatkovne baze in informacijska omrežja, za katere standardni pristopi podatkovnega rudarjenja niso primerni. Disertacija predstavi več prispevkov k področju rudarjenja kompleksnih podatkov.

V prvem delu disertacija obravnava problem klasifikacije vozlišč v večvrstnih omrežjih (t.j. omrežjih z različnimi tipi vozlišč in povezav). Osredotočimo se na pristop dekompozicije večvrstnih omrežij v enovrstna omrežja. Predlagamo več hevristik, s katerimi lahko pri postopku dekompozicije ocenimo pomembnost tako imenovanih povezovalnih vozlišč heterogenega omrežja. K razvoju teh hevristik so nas motivirale metode za uteževanje besed pri tvorjenju vektorjev besed v rudarjenju tekstovnih podatkov. Doprinos teh hevristik je zmanjševanje učinka tistih povezovalnih vozlišč, ki so povezana z veliko drugimi vozlišči, in povezovalnih vozlišč, ki povezujejo vozlišča z različnimi oznakami razredov. Hevristike smo testirali na več podatkovnih množicah in pokazali, da njihova uporaba lahko izboljša klasifikacijo vozlišč večvrstnih omrežij. Pristop, opisan v tem delu disertacije, imenujemo HinMine.

V drugem delu disertacije predstavimo kombinacijo metod za analizo omrežij in algoritmov za semantično rudarjenje podatkov (semantic data mining - SDM) z namenom iskanja vzorcev v podatkih, anotiranih z ontološkimi termi. Razvita metodologija NetSDM uporabi metode za rangiranje vozlišč v omrežjih in z njimi oceni pomembnost termov dane ontologije. Metodologijo smo testirali na več podatkovnih množicah, med drugim na množici podatkov izraženosti genov v odzivu na virusno okužbo. Eksperimentalni rezultati kažejo, da z uporabo NetSDM lahko pohitrimo obstoječi SDM algoritem Hedwig tako, da za odkrivanje opisnih pravil uporabi le nekaj minut namesto nekaj ur računalniškega časa.

Tretji del disertacije predstavi metodologijo za vizualizacijo podatkov s pasovnimi matrikami. Metodologija izboljša razumljivost rezultatov algoritma Hedwig tako, da s pomočjo pasovnih matrik najprej odkrije skrito strukturo v analiziranih podatkih. To strukturo odkrijemo neodvisno od algoritma Hedwig, zato lahko s prikazom obeh rezultatov na isti sliki končnemu uporabniku prikažemo nov vpogled na naučena pravila skupaj z zgradbo podatkovne množice.

Disertacijo zaključimo s predstavitvijo razvitih algoritmov, ki so javno dostopni. Metodologiji HinMine in NetSDM sta dostopni v okviru spletne platforme za rudarjenje podatkov ClowdFlows, metodologija za vizualizacijo s pasovnimi matrikami pa je na voljo kot paket programskega jezika Python na spletni platformi GitHub.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

This thesis contributes to the improvement of data analysis methods by combining machine learning algorithms with network analysis approaches. The main contributions include improving network node classification using text-mining-inspired heuristics, improving semantic data mining algorithms using network analysis based preprocessing, and improving the interpretability of semantic data mining algorithms using banded matrix visualization. This introductory chapter briefly motivates the problems addressed, introduces the relevant background technologies and presents an overview of the proposed solutions to the addressed problems. This is followed by outlining the purpose of the dissertation, its goals and scientific contributions. The chapter concludes with a structural overview of the rest of the thesis.

1.1 Background and Motivation

For many research questions, the relevant information is scattered across disparate public resources in heterogeneous data formats. Collecting, storing and processing of these large amounts of data is technically challenging and time-consuming, and presents a significant obstacle for a wide application of data analysis techniques.

Most data mining algorithms work only with tabular data, where the data we want to analyze is presented in a single spreadsheet. However, most of the data collected does not exist in isolation. When researchers collect new data about a given phenomenon, they are adding to an already existing pool of knowledge about the phenomenon. For example, when examining the activity of a particular gene in response to a particular disease, the researchers add information to an already existing set of known facts about the gene: which protein it encodes, which characteristics of an organism it is responsible for, which genes it activates (up-regulates), which genes it suppresses (down-regulates), and which biological processes and cell functions it is a part of. It can be shown that taking this knowledge into account improves the results of classical machine learning approaches (D. Page et al., 2012).

Like in the work of D. Page et al. (2012), previous work in the area of Inductive Logic Programming (ILP) and Relational Data Mining (RDM) (Muggleton, 1992; De Raedt, 2008; Džeroski & Lavrač, 2001) has shown that existing knowledge (referred to as *background knowledge* in this thesis) is vital for data analysis, and that a simple tabular representation is not a feasible way of describing the background knowledge. Two problems arise that necessitate the use of better data representations.

The first problem is that in the case where instances under examination are connected to each other (like genes that are connected through mutual activation, or research papers connected through paper citations), but where the instances are not connected to the same number of other instances. In this case, representing all knowledge about a given instance in a single row would be difficult. If we represented each instance connected to the described instance as a separate column, this would result in the number of columns in the resulting data set to vary depending on the row. Alternatively, if we encode the connections of one instance with a single column of the data set, this would require that the columns of the data set contain complex data structures such as lists. In RDM, this problem is addressed by representing the data set in a *multi-relational data base.* An alternative way of describing a data set containing inter-connected instances is to represent it as a *network* (Burt & Minor, 1983), a data structure containing nodes, information about the nodes, and connections between the nodes. In mathematical terms, such a structure is a represented by a *graph* – the nodes are also referred to as vertices, and the connections as edges. The field of network analysis deals with several different types of networks based on their origin. Social networks are networks in which nodes represent people and connections represent their social connections. Biological networks are networks that arise from biological data and include among others gene regulatory networks, protein interaction networks and drug interaction networks. The term *information* network is used to described networks where connections are directed and encode some sort of information flow from one node to another. When collected data becomes even more complex, it is appropriate to represent such structures as a *heterogeneous information network* (Sun & Han, 2012) with each node and edge belonging to a given type. For example, we may have a network containing both genes and the proteins they encode, necessitating the use of two node types to represent the data.

The second problem requiring better data representation arises when instances already belong to some known groups of connected instances (in the case of genes, we may already know that a certain set of genes is responsible for cell division). While each group can be represented in a tabular data set as a zero-one column, this does not solve the problem if the groups of instances themselves are further organized. For example, a group of genes can be involved in mitosis (cellular division in a non-reproductive cells) and another group can be involved in meiosis (cellular division in reproductive cells), while a third group describes all genes that are involved in any sort of cellular division. In such a case, no gene belongs to only one of the three described groups, and this information is lost if we encode each group as a simple zero-one column in a tabular representation of data. In this case, a better way to represent knowledge is through the use of *ontologies* (Guarino, Oberle, & Staab, 2009), such as the Gene Ontology (Ashburner et al., 2000). Ontologies are a formal way of representing background knowledge using *terms* connected by directed edges to form an acyclic graph in which connections represent relationships (for example, the *is-a* relationship) between the terms. This special form of background knowledge, which has not been exploited in the original ILP and RDM literature, has been recently addressed in Semantic Data Mining (SDM) research (Lavrač, Vavpetič, Soldatova, Trajkovski, & Kralj Novak, 2011).

1.1.1 Advancing heterogeneous information network analysis

In our work, we are particularly interested in the task of *classification* in a network setting. The task receives as input an information network in which some of the nodes are labeled as belonging to a certain class. In the case of genes, the classes may denote active and inactive genes, or if we are analyzing scientific papers connected through a network of citations, the classes denote the research areas investigated by the paper. The goal is to correctly predict the labels of the remaining nodes in the network. A well established way of labeling nodes in a network is the label propagation approach presented by D. Zhou, Bousquet, Lal, Weston, and Schölkopf (2004) in which class labels are propagated from nodes with known labels via connections in a network to the nodes with unknown labels.

In a heterogeneous information network, the task of classification may be limited to only one node type (for example, in a paper-and-author network, we may only wish to classify the papers as belonging to a certain class). One way of approaching the task of node classification in heterogeneous information networks is through network decomposition, a method first described by Grčar, Trdin, and Lavrač (2013). Network decomposition (also refered to as network projection (T. Zhou, Ren, Medo, & Zhang, 2007)) of a heterogeneous network results in the construction of homogeneous (i.e. single type) networks in which connections are inferred through intermediary nodes. Either label propagation or propositionalization can then be used to infer the missing labels. This thesis further develops the idea of classification via heterogeneous information network decomposition by examining the impact of intermediary nodes. The approach, named HINMine, is presented in Chapter 3.

1.1.2 Advancing semantic data mining

To find patterns in data annotated with ontologies, we rely on Semantic Data Mining (SDM), which extends the research on symbolic rule learning (Fürnkranz, Gamberger, & Lavrač, 2012). In SDM, the input is composed of a set of instances, each belonging to a class, and background knowledge encoded in the form of an ontology. Each instance is annotated by one or more terms from the ontology, and the goal is to find class descriptions composed of rules of the form `TargetClass` ← `Explanation`, where the explanation is a logical conjunction of terms from the ontology. For example, when analyzing the expression of genes in breast cancer patients, the `TargetClass` of the generated rules is the class of all genes that are differentially expressed in the patients compared to the general population. An example of an explanation would then be the conjunction of biological concepts `chromosome AND cell cycle`, meaning the rule covers all genes that are covered by both concepts. A list of other possible rules for this example is shown in Figure 2.2 of Chapter 2.

The starting point for the research in this thesis is the Hedwig algorithm (Vavpetič, Novak, & Lavrač, 2013; Vavpetič, 2016), which is able to discover complex rules describing subgroups of data instances annotated by ontology terms. The background knowledge in this case is represented in the Resource Description Framework (RDF) format[1]. The format encodes relationships in the form of *subject-predicate-object* triples and can be used to encode both *is-a* and other relationships by using different predicate relationships. The algorithm uses a heuristic-guided beam search to traverse the potentially infinite space of possible logical expressions that could be used as descriptions. A potentially large size of the search space in Hedwig indicates that the algorithm will be able to discover informative rules describing the data set, however it also means that the search can take a long time. This issue is addressed by this thesis in Chapter 4, where we present a method for pruning the background knowledge before using it as an input for Hedwig. A large search space may also cause overfitting and hence prevent the finding of (truly) informative rules, an issue addressed by algorithms by requiring a minimum number of examples that a rule needs to describe (this number is called the coverage of a rule). A second problem with Hedwig is that its output is a list of rules that may be difficult to understand. This motivated us to develop a visualization algorithm for the data sets that can visualize Hedwig's rules and make them more comprehensible. The algorithm is described in Chapter 5.

[1]https://www.w3.org/TR/REC-rdf-syntax/

1.2 Purpose of the Dissertation

The purpose of this dissertation is to improve data analysis methods by combining network analysis algorithms with machine learning with a specific focus semantic data mining. The dissertation covers three main topics: improving network classification using text-mining inspired heuristics, improving semantic data mining algorithms using network-analysis-based preprocessing, and improving interpretability of semantic data mining algorithms using banded matrix algorithms.

- *Improving node classification in heterogeneous networks using text mining inspired heuristics.* The first part of the dissertation (covered in Chapter 3) examines the improvements made to the algorithms performing network classification through propositionalization. In classifying network nodes through network decomposition, the current algorithm (Grčar et al., 2013) considers all connecting nodes as equals, which may be inappropriate. For example, in a network of papers and their authors, highly prolific senior authors who publish many papers in several research fields may induce a lot of connections, however these connections can often bridge different research fields (due to the authors seniority and thus familiarity with different research fields). This problem is similar to a bag-of-words vector contruction problem in text mining. The problem of common words appearing in many documents is tackled in text mining by using several weighting heuristics such as the `tf-idf` (term frequency-inverse document frequency) heuristic to penalize the impact of common words. A second problem arises when classifying nodes using label propagation in a highly imbalanced data set. In such a case, unlabeled nodes are likely to be surrounded mostly by neighbors belonging to the majority class compared to the minority class. The dissertation examines how this over-voting problem can be addressed through appropriate weighting schemes applied to the weighting of network nodes.

- *Improving semantic data mining algorithms using network-analysis-based preprocessing.* We examine how network analysis algorithms can be used to explore and prune the search space of semantic data mining algorithms. The performance of semantic data mining algorithms is currently bottlenecked by the size of the search space these algorithms must examine. Even with the state-of-the-art heuristics guiding the search, the construction of high quality rules is a time-consuming process. This problem is addressed by using network analysis methods, such as the famous PageRank method (L. Page, Brin, Motwani, & Winograd, 1999), with which one can estimate the importance of a given node in a network. We evaluate three methods for node ranking and explore how they can be used to improve the quality of rules, obtained by semantic data mining.

- *Improving the interpretability of semantic data mining algorithms using banded matrix algorithms.* While semantic data mining algorithms produce complex and highly informative results, the results are relatively inaccessible even to domain experts as the output of an SDM algorithm is a potentially long list of rules that characterize a given class of data. Improved readability of the results can be achieved by better visualization of semantic data mining results using banded matrix algorithms. Banded matrix algorithms (Garriga, Junttila, & Mannila, 2011) can discover underlying structures in the data independently of SDM algorithms, and overlaying them with SDM results produces an informative and easier-to-read output.

The purpose is also to make these algorithms publicly accessible through reusable implementations of the proposed algorithms to allow simple comparison with competing algorithms. The algorithms are implemented in ClowdFlows (Kranjc, Podpečan, & Lavrač, 2012), an open-source cloud-based platform for composition, execution, and sharing of interactive machine learning and data mining workflows.

1.3 Hypotheses and Goals

Aligned with the three problems presented above, the hypotheses of the work are as follows.

We hypothesize that heuristics, developed in text mining for weighting words in documents for bag-of-words vector construction, can be adapted for weighting connecting nodes during network decomposition. The adapted weights can penalize connecting nodes that do not induce informative connections, and in this way the heuristics should improve the accuracy of classification algorithms for heterogeneous networks.

As the data and background knowledge can be viewed as a heterogeneous network, we hypothesize that it is possible to use network analysis algorithms to estimate the importance of background knowledge terms before running semantic data mining algorithms. In particular, we hypothesize that it is possible to prune the search space by eliminating the nodes that are found to be non-informative by network analysis algorithms. Such pruning should increase the quality of the resulting rules while at the same time reduce the time complexity of the SDM algorithms.

The results of semantic data mining algorithms are currently only represented in a textual format. Our hypothesis is that the interpretability of semantic data mining algorithms can be enhanced by rule visualization using banded matrix algorithms. This shall improve readability and comprehensibility of results.

1.4 Scientific Contributions

The scientific contributions of the thesis are as follows.

Contribution 1 We developed an improved methodology for classification of nodes in a heterogeneous network via network decomposition and propositionalization, using text mining inspired heuristics. This allows us to better assess the importance of intermediary nodes and prevent uninformative intermediary nodes to induce a vast majority of connections. We also improved the final step of classification through decomposition and label propagation by improving initial weights of classes when the class distribution is unbalanced. We evaluated the effects of different heuristics and different initial weights for class propagation on several data sets with varying characteristics.

Publications related to this contribution

Journal Paper

Kralj, J., Robnik-Šikonja, M., & Lavrač, N. (2017). HINMine: Heterogeneous Information Network Mining with Information Retrieval Heuristics. *Journal of Intelligent Information Systems*, 1–33.

Conference Paper

Kralj, J., Valmarska, A., Robnik-Šikonja, M., & Lavrač, N. (2015). Mining Text Enriched Heterogeneous Citation Networks. In *Proceedings of the 19th Pacific-Asia Conference on Knowledge Discovery and Data Mining* (pp. 672–683).

Book Chapter

Kralj, J., Valmarska, A., Grčar, M., Robnik-Šikonja, M., & Lavrač, N. (2016). Analysis of Text-enriched Heterogeneous Information Networks. In N. Japkowicz & J. Stefanowski (Eds.), *Big Data Analysis: New Algorithms for a New Society* (pp. 115–139). Springer.

Kralj, J., Robnik-Šikonja, M., & Lavrač, N. (2015b). Heterogeneous Network Decomposition and Weighting with Text Mining Heuristics. In *International Workshop on New Frontiers in Mining Complex Patterns* (pp. 194–208). Springer.

Contribution 2 Our research resulted in a methodology for network analysis assisted semantic data mining (SDM) which uses network analysis methods to pre-prune the background knowledge used by SDM. This methodology is capable of extracting knowledge from much larger background knowledge collections than pure SDM algorithms. We evaluated different variations of network conversion, network node scoring and network pruning methods to pre-process the background knowledge. We also applied the algorithms to a new biological data set containing gene expression data for potato plants, infected with a virus. We report the positive effects of background knowledge pruning on the quality of the rules and the efficiency of SDM algorithms.

Publications related to this contribution

Journal paper

Kralj, J., Robnik-Šikonja, M., & Lavrač, N. (n.d.). NetSDM: Network Filtering for Semantic Data Mining. *(submitted)*.

Conference paper

Kralj, J., Vavpetič, A., Dumontier, M., & Lavrač, N. (2016). Network Ranking Assisted Semantic Data Mining. In *International Conference on Bioinformatics and Biomedical Engineering* (pp. 752–764). Springer.

Contribution 3 We have developed a methodology merging a semantic data mining algorithm with banded matrix visualization to display the rules, discovered using the SDM algorithm. The methodology discovers a hidden structure in data sets independently from semantic data mining, meaning that overlaying its results with semantic data mining rules offers a new view of the underlying data set. We tested the methodology on several publicly available data sets.

Publications related to this contribution

Journal Paper

Adhikari, P. R., Vavpetič, A., Kralj, J., Lavrač, N., & Hollmén, J. (2016). Explaining Mixture Models through Semantic Pattern Mining and Banded Matrix Visualization. *Machine Learning*, *105*(1), 3–39.

Conference Paper

Adhikari, P. R., Vavpetič, A., Kralj, J., Lavrač, N., & Hollmén, J. (2014). Explaining Mixture Models through Semantic Pattern Mining and Banded Matrix Visualization. In *International Conference on Discovery Science* (pp. 1–12). Springer.

Contribution 4 We have implemented the functions, developed in contributions 1 and 2, publicly available on the ClowdFlows website. The source code for the functions is also available on GitHub. The descriptions of the software, along with the links to the code repositories, is available in Chapter 6.

1.5 Structure

The rest of the thesis is structured as follows. Chapter 2 presents the related work. The chapter is split into sections roughly corresponding to the related work of Chapters 3, 4 and 5 that present the main body of our work.

In Chapter 3, we describe our approach to improving the network decomposition and label propagation steps of the methodology developed by Grčar et al. (2013). We begin the chapter with a detailed description of the network decomposition and label propagation algorithms and the problems arising with using each of them. We then present the improved HinMine methodology that improves on both the network decomposition approach (by using text mining inspired heuristics) and label propagation (by modifying the starting weights of the nodes) and fuses both approaches. We present the experimental setting in which the new algorithm was tested — first we provide a description of the data sets that were used, and then the results on each of the data sets. We conclude the chapter with a summary of experimental results.

In Chapter 4, we describe the NetSDM methodology that uses network analysis methods to pre-process input data for semantic data mining algorithms. We first present the background technologies relevant for the chapter, specifically the Hedwig semantic data mining algorithm and the network scoring function used in our experiments. We then present the new methodology that converts a background knowledge network into an information network, evaluates the importance of the terms in the network, and then prunes the network, leaving only the most relevant terms in the background knowledge. Experiments show that the pruned background knowledge allows for much quicker analysis with SDM technologies and yields better results than the existing solutions.

Chapter 5 presents the results of using banded matrix algorithms to visualize the outputs of semantic data mining algorithms. We present a methodology that performs clustering, rule discovery using an SDM algorithm, and data visualization using banded matrix methods. The methodology is tested on several publicly available data sets.

The work continues with Chapter 6 in which we present the publicly available workflows, developed and implemented in the course of this thesis, and concludes with Chapter 7 which summarizes of the presented work and presents the ideas for future work.

Chapter 2

Related Work

This chapter provides the related work in three research topics covered by the thesis: network analysis, semantic data mining and data visualization.

2.1 Network Analysis

In network analysis, concepts and results from graph theory are used to investigate characteristics of networked structures in terms of nodes (individual actors, people, or objects/things within the network) and their ties, edges, or links (relationships or interactions) that connect them. This section presents the related work on node classification and ranking in information networks, for homogeneous and heterogeneous information networks. As the work presented in Chapter 3 is inspired by the heterogeneous information network propositionalization using bag-of-words vector construction, we also present several heuristics used in text mining.

2.1.1 Node classification in networks

In network classification, the task is to find class labels for some of the nodes in the network using known class labels of the remaining network nodes. As the network structure is known in advance even for the unclassified nodes, this is a semi-supervised learning task. Several classification methods are proposed in the literature. Sen et al. (2008) presented four network classification algorithms (in their work, they refer to the problem of classifying nodes in a network as *collective classification*), while de Sousa, Rezende, and Batista (2013) describe five algorithms used for semi-supervised learning which are based on constructing a network of nodes and using it to classify the nodes. One of the approaches used is the propagation of labels in the network (referred to as *label propagation*), a concept used by D. Zhou et al. (2004) and Vanunu, Magger, Ruppin, Shlomi, and Sharan (2010). Other approaches include Gaussian Random Fields (Zhu, Ghahramani, Lafferty, et al., 2003), its modification Robust Multi-Class Graph Transduction (W. Liu & Chang, 2009), Laplacean support vector machines, and Laplacean Regularized Least Squares (Belkin, Niyogi, & Sindhwani, 2006).

The limitation of collective classification and label propagation algorithms is that the algorithms do not take the types of nodes in a network into account and only work well on homogeneous information networks (where all nodes are of the same type). Another limitation of these algorithms is that they are not designed for imbalanced data. For example, in label propagation, if one class has a large majority in the initially labeled set, then this class casts a much stronger vote along the edges of the network, potentially over-voting all other classes.

2.1.2 Node ranking in information networks

The objective of ranking in information networks is to assess the relevance of a given object either globally (with regard to the whole graph) or locally (relative to some object in the graph). A network node ranking algorithm assigns a *score* (or *rank*) to each node in the network, with the goal of ranking the nodes in terms of their relevance. A well known ranking method is PageRank (L. Page et al., 1999), which was used in the Google search engine. Other methods for ranking include Weighted PageRank method (Xing & Ghorbani, 2004), SimRank (Jeh & Widom, 2002), diffusion kernels (Kondor & Lafferty, 2002), hubs and authorities (Kleinberg, 1999), and spreading activation (Crestani, 1997). Another approach to rank nodes in a network is to use network centrality measures, e.g., Freeman's network centrality (Freeman, 1979), betweenness centrality (Freeman, 1977), closeness centrality (Bavelas, 1950), and Katz centrality (Katz, 1953). For our work, the Personalized-PageRank (L. Page et al., 1999)—frequently abbreviated as P-PR—that calculates the node score locally to a given network node is especially interesting because it calculates the importance of network nodes in relation to a given set of starting nodes. In the context of a data set annotated by using ontologies, this allows us to estimate the importance of background knowledge terms in relation to the examples in the data set.

In this work, we used a network node ranking approach (1) to calculate feature vectors for nodes in a network and (2) to limit the search space of semantic data mining. In both use cases, we are interested in network ranking methods which can be 'localized', i.e. which does not compute the global importance/score of a node but rather calculate the score of a node in the context of a given subset of nodes. An example of this type of network node ranking approaches is the Personalized PageRank algorithm (L. Page et al., 1999), sometimes referred to as random walk with restarts (Tong, Faloutsos, & Pan, 2006). For example, this algorithm is used by Grčar et al. (2013) in a classification setting to construct feature vectors from network nodes based on their relevance to a chosen node.

The idea of PageRank—frequently abbreviated as PR—is motivated by two different views. The first is the random walker approach: a random walker starts walking from a random node v of the network and in each step walks to one of the neighboring nodes with a probability proportional to the weight of the edge traversed. The PageRank of a node is then the expected proportion of time the walker spends in the node, or, equivalently, the probability that the walker is in the particular node after a long time. The second view of PageRank is the view of score propagation. The PageRank of a node is its score, which it passes to the neighboring nodes. A node v_i with a score $PR(i)$ transfers its score to all its neighbors. Each neighbor receives a share of the score proportional to the strength of the edge between it and v_i. This view explains the PageRank with a principle that in order for a node to be highly ranked, it must be pointed to by many highly ranked nodes.

Given a set of 'starting' nodes A, the Personalized PageRank vector, calculated for A (denoted P-PR$_A$) in a network is defined as the stationary distribution of positions of a random walker that starts the walk 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. The probability (denoted p) of continuing the walk is a parameter of the Personalized PageRank algorithm and is usually set to 0.85. To calculate the vector P-PR$_A$ for a given network, it is best to represent the network with an *adjacency matrix* M. For a directed network containing n nodes $v_1, v_2, \ldots, v_n$, the matrix M is a $n \times n$ matrix for which M_{ij} is equal to the weight on the edge going from node v_i to node v_j. Using the adjacency matrix to represent the network, the vector P-PR$_A$ is calculated in three steps.

1. In the first step, only the all-zero row of matrix M are modified. For every j for which $M_{ij} = 0$ for all j, the value of M_{ij} is set to 1 if the node v_i is in the set A, and

to 0 otherwise. In effect, this modifies the original network by adding edges nodes which do not have any outgoing edges. In the modified network, each such node is connected to all starting nodes in the set A. This first step modifies the starting matrix M into matrix M'.

2. In the second step, the adjacency matrix M is transformed into a stochastic matrix. A stochastic matrix is a matrix that satisfies two conditions: (1) all its values are positive and the sum of each row of the matrix is equal to 1. To transform the matrix M into a stochastic matrix P, each row is divided by the sum of its values. Mathematically, we calculate P as

$$P = D^{-1}M$$

where D is a diagonal matrix for which $D_{ii} = \sum_{j=1}^{n} M_{ij}$.

3. In the third step, the matrix P is used to iteratively calculate the vector P-PR$_A$. The iteration starts with the vector x_0 whose i-th component is equal to 1 if $v_i \in A$ and 0 otherwise. In each iterative step $s \geq 1$, the vector x_s is calculated as

$$x_t = p \cdot Px_{t-1} + (1-p) \cdot x_0.$$

The parameter p is a parameter of the Personalized PageRank algorithm and is usually set to $p = 0.85$. It represents the probability of a random walker to continue its walk instead of returning back to a randomly selected node from A. While setting p to 0.85 has no theoretical background, empirical results show that this is a good setting. The algorithm terminates once the difference between x_s and x_{s-1} falls below a threshold, and the resulting vector x_s is the vector P-PR$_A$. It can be shown (Bourchtein & Bourchtein, 2013) that iterating this step converges as the procedure is effectively calculating the eigenvector corresponding to the eigenvalue 1 of an irreducible non-periodic stochastic matrix whose second largest eigenvalue is p.

In our context, the Personalized PageRank algorithm is used to calculate the importance of network nodes *with respect to* a given starting set of nodes.

Remark 2.1.1. While P-PR$_A$ is technically a vector, each value of the vector corresponds to a particular network node v. To shorten the notation, we write P-PR$_A(v)$ instead of using the i-th value of P-PR$_A$, where v is the i-th node.

2.1.3 Node embedding using node2vec

A recently developed approach to vectorize network nodes is the node2vec algorithm (Grover & Leskovec, 2016), which uses the random walk approach to calculate features that express similarities between pairs of nodes. 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 in the network and E is the set of connections, or edges, in the network. The algorithm returns a matrix $f \in \mathbb{R}^{|V| \times k}$ with a pre-defined number of columns k. The matrix f is interpreted as 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. As with the P-PR$_A$ vector, we write $f(u)$ to mean the row of matrix f, corresponding to the node u. The goal of the algorithm is to construct the feature vectors $f(u)$ in such a way that the feature vectors of all nodes that share a certain neighborhood will be similar. The matrix f is calculated as as

$$\text{node2vec}(G) = \underset{f \in \mathbb{R}^{|V| \times k}}{\text{argmax}} \sum_{u \in V} \left(-\log(Z_u) + \sum_{n_i \in N_S(u)} f(n_i) \cdot f(u) \right). \tag{2.1}$$

where $N(u)$ denotes the network neighborhood of node u given a *sampling strategy* and

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

In the above expression the inner sum calculates the similarities between a node and all nodes in its neighborhood. This sum 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. The first value of each summand, $-\log(Z_u)$, decreases when the norms of feature vectors increase, thereby penalizing collections of feature vectors with large norms.

The expression (2.1) has a probabilistic interpretation which models a process of randomly selecting nodes from the network. The 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 one node is independent from selecting any other node, we can calculate the probability of selecting all nodes from a given set A as $P(A|u) = \prod_{n \in A} P(n|u)$, and Equation 2.1 can then be rewritten as

$$\text{node2vec} = \underset{f \in \mathbb{R}^{|V| \times k}}{\text{argmax}} \sum_{u \in V} \log \left(P(N_S(u) | f(u)) \right). \tag{2.2}$$

The term $N_S(u)$ in Equations (2.1) and (2.2) denotes the neighborhood of u given a sampling strategy S and is 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 also depends on the node n_0 the walker visited before node n_1, making the process of traversing the network a second order random walk. The unnormalized transition probabilities are set using two parameters, p and q, and are equal to

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

The parameters p and q are referred to as the *return* parameter and the *in-out* parameter, respectively. A low value of the return parameter p means that the random walker is more likely to backtrack its steps, meaning the random walk will be closer to a breadth first search. On the other hand, a low value of the 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 vector f, a set of random walks of limited size is simulated starting from each node in the network to generate several samples of the set $N_S(u)$.

The function maximizing expression (2.1) is calculated using stochastic gradient descent. The value of (2.1) is estimated at each generated sampling of the neighborhoods $N_S(u)$ for all nodes in the network to discover the vector f that maximizes the expression for the simulated neighborhood set.

2.1.4 Analysis of heterogeneous information networks

Analagous to the algorithms developed for node classification in homogeneous networks, adapted methods have been proposed for classification of nodes in heterogeneous networks. Hwang and Kuang (2010) expand the idea of label propagation used by D. Zhou et al. (2004) to include multiple dampening parameters (one for each node type pair) in place of a single parameter. A similar approach is taken by Sun and Han (2012). Ji, Sun, Danilevsky, Han, and Gao (2010) propose the GNetMine algorithm which uses the idea of knowledge propagation through a heterogeneous information network to find probability estimates for labels of the unlabeled data. A strong point of this approach is that it has no limitations on how different node types in a network are connected (this is sometimes referred to as a network schema), meaning that it can be applied to both heterogeneous and homogeneous networks. Building on the idea of GNetMine, Sun and Han (2012) propose a classification algorithm that relies on within-class ranking functions to achieve better classification results. The idea is that nodes, connected to high ranked entities of a given class, most likely belong to the same class. This idea is implemented in the RankClass framework for classification in heterogeneous information networks.

2.1.5 Propositionalization of heterogeneous information networks

An alternative approach to classification in heterogeneous networks, is classification of network nodes through network decomposition and propositionalization. In this section, we present the approach, developed by Grčar et al. (2013), as it represents the starting point for the work, presented in in Chapter 3. The approach analyzes a network in which a certain type of nodes (called the target type) is partially labeled. The task is to discover the labels of target-type nodes that are unknown. The adopted approach consists of two main steps. In the first step, a heterogeneous network is decomposed into a set of homogeneous networks. In the second step, the homogeneous networks are used to predict the labels of target nodes.

In the first step of the methodology, the original heterogeneous information network is decomposed into a set of homogeneous networks. Each homogeneous network contains only the nodes of the original network of a given target node type (in this chapter, we will refer to these nodes as *base nodes*). In each homogeneous network two nodes are connected if they are directly or indirectly linked in the original heterogeneous network. The nodes are linked indirectly of they are connected to the same intermediary node. Take as an example the network originally presented by Grčar et al. (2013). Let the network contain two types of nodes, *Papers* and *Authors*, and two edge types, *Cites* (linking papers to papers) and *Written_ by* (linking papers to authors). From this network we can construct two homogeneous networks of papers: the first, in which two papers are connected if one paper cites another, and the second, in which they are connected if they share a common author (shown in Figure 2.1). This construction is similar to the construction of author collaboration networks (C. Chen & Paul, 2001), however instead of using papers as intermediate nodes inducing links between authors, we use authors to induce links between papers in this decomposition. In general, the choice of links used in the network decomposition step requires an expert who takes the meaning of links into account and chooses only the decompositions relevant for a given task (any link between two papers in the heterogeneous network can be used to construct either a directed or undirected edge in the homogeneous network).

Note that in the network in which two papers are connected if they share a common author, each common author induces one connection between the two papers. This can be expressed mathematically as the weight of the link between the two papers being set to the number of common authors (number of common neighboring nodes of the type *Author*).

In the second step of the methodology, class labels for the unlabeled nodes are calculated using a process of *network propositionalization*, a method for classifying the target nodes in the original heterogeneous network introduced by Grčar et al. (2013). The propositionalization step calculates feature vectors for each target node in the network using the Personalized PageRank (P-PR) algorithm (L. Page et al., 1999), described in 2.1.2. The Personalized PageRank of node v corresponds to the Personalized PageRank vector, as described in Section 2.1.2, if we set the starting set A to $A = \{v\}$. Once calculated, the resulting PageRank vectors for each node in the network are normalized according to the Euclidean norm. The resulting vector contains information about the proximity of node v to each of the nodes of the network. We consider the P-PR vectors of a node as a propositionalized feature vector of the node. Two nodes with similar P-PR vectors will be in proximity of similar nodes, and classifiers will consider them as similar instances. We use the vectors to classify the nodes from which they were calculated.

For a single homogeneous network, the propositionalization results in one feature vector per node. For classifying a heterogeneous network decomposed into k homogeneous networks Grčar et al. (2013) propose to concatenate and assign weights to the k vectors, obtained from the k homogeneous networks. The weights of the vectors are optimized using the computationally expensive differential evolution (Storn & Price, 1997). A simpler alternative is to use equal weights and postpone weighting to the learning phase. Due to the size of feature vectors in our experiments, we to followed the latter approach. After the weights are set, many classifiers, for example SVM (Manevitz & Yousef, 2001; Kwok, 1998; D'Orazio, Landis, Palmer, & Schrodt, 2014), k-NN (Tan, 2006) or a centroid classifier (Han & Karypis, 2000) can be used. It was shown by Grčar et al. (2013) that the calculating feature vectors and using them for the classification of nodes in a network yields good results.

2.1.5.1 Improvement opportunities

A drawback of the decomposition and propositionalization approach by Grčar et al. (2013) is that in the decomposition step—where individual homogeneous networks are constructed—the homogeneous networks are composed of all the nodes of the given type that are con-

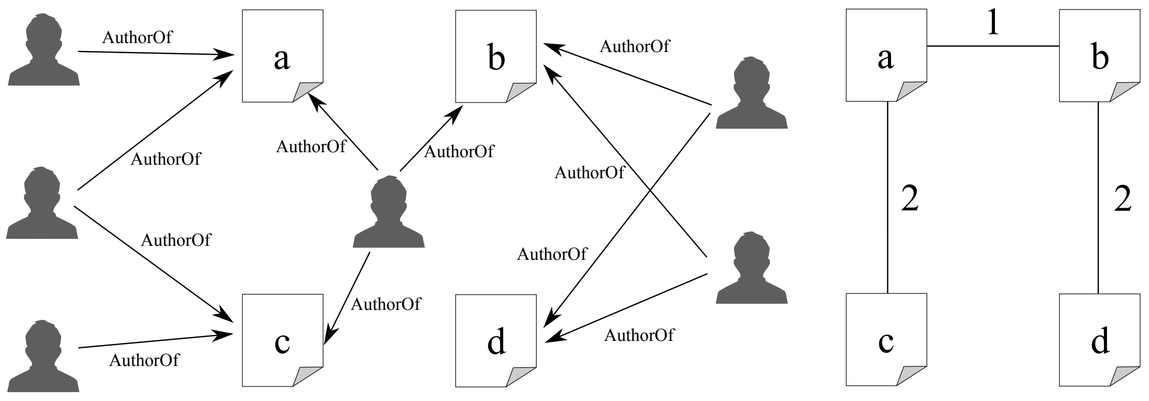


Figure 2.1: An example (first used in (Grčar, Trdin, & Lavrač, 2013)) of a heterogeneous network (on the left-hand side) and a homogeneous network extracted from it (on the right-hand side). In the homogeneous network, papers are connected if they share a common author. Weights of the edges are equal to the number of authors that contributed to both papers.

nected through an intermediary node of any other type. For example, from a network consisting of papers and their authors, a homogeneous network of papers is constructed in which two papers are connected if they have the same author (in this example, author nodes are the intermediate nodes). This construction may cause highly prolific authors to induce most of the connections in the resulting network, which is not optimal. In reality, links induced by the most prolific authors are not the most important as they are not informative about the papers related to other papers; on the other hand, if an author only writes a small number of papers, those papers are much more likely to be closely connected to each other. A similar problem was addressed in the construction of co-authorship networks of authors where papers are the intermediate nodes. If each publication two writers co-author induces a link with a weight of 1 between the authors, then publications with k authors induce $\binom{k}{2}\frac{k(k-1)}{2}$ links in the induced network. Perianes-Rodriguez, Waltman, and van Eck (2016) propose to add a weight of $\frac{1}{k}$ to each of the induced connections. This means a publication with k authors increases the sum of all weights in the induced network by $\frac{k}{2}$ which is proportional to k, the number of authors.

In our work, we based our research on the fact that a similar problem to that of the exaggerated impact of prolific authors in a paper-author network is also encountered in the field of text mining when constructing bag-of-words (BOW) vectors from text documents. The simplest way to construct a BOW vector of a document is to count appearances of a word or term in the document. This term frequency (tf) weighting is used by Grčar et al. (2013) for BOW vector construction in heterogeneous network propositionalization. In text mining, however, tf-based weighting is rarely used, as the term-frequency inverse-document-frequency (tf-idf) weighting introduced by Jones (1972) contains more information for document classification. The tf-idf weighting schema was designed to penalize terms that appear in many documents of the corpus, as their appearance in a particular document is not informative. A number of other weighting heuristics, which also take labels of documents into account, have been proposed. These include the χ^2, information gain and gain ratio (Debole & Sebastiani, 2004), Δ-idf (Martineau & Finin, 2009), and relevance frequency (Lan, Tan, Su, & Lu, 2009). These weighting schemes are usable in a classification setting where the objective is to determine the class of a document. In such a setting, the weights are designed to penalize terms that appear in the documents of all/most of the target class examples and therefore poorly discriminate between the classes. While the above-mentioned term weighting schemes evaluate the importance of a given term in the entire corpus of documents, the Okapi BM25 function (Robertson & Walker, 1994) evaluates the importance of a term specifically for each (class labeled) document. It should be noted that this function gives smaller weights to terms that appear in long (compared to the average document length) documents. In Chapter 3, we adapt the text retrieval heuristics to weigh intermediate nodes in network decomposition.

2.1.5.2 Label propagation

An alternative method of classifying nodes in a homogeneous network is the label propagation (D. Zhou et al., 2004) algorithm. The algorithm starts with a network adjacency matrix $M \in \mathbb{R}_{+}^{n,n}$ and a class matrix $Y \in \{0,1\}^{n,|C|}$, where $C = \{c_1, \ldots, c_m\}$ is the set of classes, with which the network nodes are labeled. The j-th column of Y represents the j-th label of C, meaning that Y_{ij} is equal to 1 if the i-th node belongs to the j-th class and 0 otherwise. A node with no labels is represented in Y by a all-zero row. The algorithm constructs the matrix $S = D^{-\frac{1}{2}}MD^{-\frac{1}{2}}$, where D is a diagonal matrix and the value of each diagonal element is the sum of the corresponding row of M (i.e. the degree of the corresponding node). Matrix S is sometimes referred to as the symmetric normalized Laplacian matrix of the network (Newman, 2010).

The goal of the algorithm is to calculate a matrix $F^* \in \mathbb{R}^{n,|C|}$ which satisfies two conditions. First, for labeled nodes in the network, the corresponding rows of F must be similar to the rows of Y. Second, if two nodes are connected in the network, then their corresponding rows must also be similar. The label propagation algorithm calculates the matrix F iteratively. Initially, the matrix $F(0)$ is defined as $F(0) = Y$. In each step of the iteration, the matrix $F(t)$ is calculated as $F(t) = \alpha S F(t-1) + (1-\alpha)Y$, and the iteration concludes when there are no changes in matrix $F(t)$. D. Zhou et al. (2004) show that the iterative process converges to the matrix F regardless of the starting matrix $F(0)$. In their work, D. Zhou et al. (2004) also show that matrix F can also be calculated by solving a system of linear equations, as

$$F^* = (I - \alpha S)^{-1} Y. \tag{2.3}$$

More importantly, the resulting matrix F minimizes the quadratic expression

$$\sum_{i,j=1}^{n} M_{ij} \left\| \frac{1}{\sqrt{D_{ii}}} F_i - \frac{1}{\sqrt{D_{jj}}} F_j \right\|^2 + \left(\frac{1}{\alpha} - 1 \right) \sum_{i=1}^{n} \|F_i - Y_i\|^2$$

where F_i and Y_i denotes the i-th row of matrix F and Y, respectively. The first summand of the quadratic expression characterizes the requirement that connected nodes must have similar corresponding rows of matrix F, and the second summand characterizes the requirement that the matrix F is similar to Y. The term α is a parameter of the algorithm that was set to 0.99 in our experiments following D. Zhou et al. (2004).

The resulting matrix F is used to predict the class labels of all unlabeled nodes in the network by labeling the i-th node in the network with the class, associated with the column at which the i-th row of matrix F acchieves the highest value.

2.2 Semantic Data Mining

Semantic Data Mining (SDM) can be viewed as an extension of rule learning and subgroup discovery. In SDM, structured data and background knowledge are used to uncover explanations composed of multiple attributes and relations. SDM forms explanatory if-then rules that characterize the instances of the target class, i.e. the class of interest to the domain expert. This section presents the related work on semantic data mining and forms an introduction to the work presented in Chapter 4.

2.2.1 Rule learning and subgroup discovery

One of the established knowledge discovery and data mining techniques (Piatetsky-Shapiro, 1991) is rule learning (Fürnkranz et al., 2012). Rule learning was initially focused on learning predictive models in the form of classification rules and descriptive pattern mining. For example, association rule learning (Agrawal & Srikant, 1994) can find interesting patterns in both unsupervised and supervised setting (B. Liu, Hsu, & Ma, 1998). Building on classification and association rule learning, subgroup discovery techniques find interesting patterns as sets of rules that best describe the target variable (Klösgen, 1996; Wrobel, 1997). Measures of rule quality are further described in Chapter 4. Typically, in the learned rules of the form `TargetClass` $\leftarrow$ `Explanation`, the rule condition (`Explanation`) is a conjunction of features (attribute values) that characterize the examples of the target class covered by the rule. For example, a rule describing a subgroup of genes differentially expressed in breast cancer is the rule containing a conjunction of two terms, `chromosome` and `cell cycle`. The rule covers all genes that are connected to both biological terms. A table of other rules describing the same set of genes is shown in Figure 2.2.

2.2.2 enrichment analysis and ontologies

Semantic data mining can be viewed as an extension of enrichment analysis (Trajkovski, Lavrač, & Tolar, 2008). Enrichment analysis (EA) techniques (Trajkovski, Lavrač, & Tolar, 2008) are statistical methods used to identify putative explanations for a set of entities based on the over- or under-representation of their attribute values. In life sciences, EA is widely used with the Gene Ontology (GO) (Consortium, 2008) to profile the biological role of genes, such as differentially expressed cancer genes in microarray experiments (Tipney & Hunter, 2010).

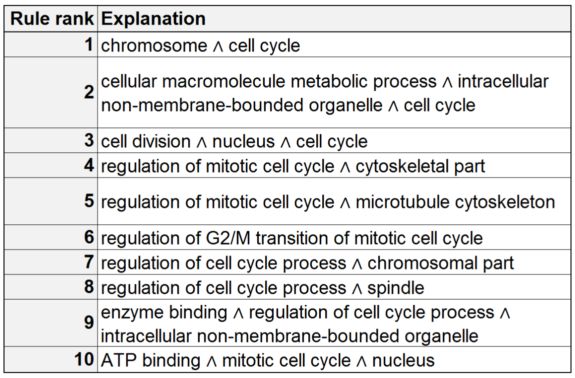
While standard EA provides explanations in terms of concepts from a single ontology, researchers are increasingly using it with combinations of several ontologies and data sets to uncover novel associations. The ability to detect patterns in data sets which do not use only GO can yield valuable insights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

2.2.3 Using ontologies in rule learning

Given an abundance of taxonomies and ontologies that are readily available, these can be used to provide higher-level descriptors and explanations of discovered subgroups. For example, in the domain of systems biology the Gene Ontology (Consortium, 2008), KEGG orthology (Ogata et al., 1999) and Entrez gene–gene interaction data (Maglott, Ostell, Pruitt, & Tatusova, 2005) are examples of structured domain knowledge that can be used as additional higher-level descriptors in induced rules describing interesting subgroups. The SEGS algorithm (Trajkovski, Lavrač, & Tolar, 2008) was the first to combine enrichment analysis and machine learning research in the construction of rules explaining gene expression data. SEGS constructs gene sets as combinations of GO ontology (Consortium, 2008) terms, KEGG orthology (Ogata et al., 1999) terms, and terms describing gene–gene interactions obtained from the Entrez database (Maglott et al., 2005).

The challenge of incorporating domain ontologies in data mining was addressed in SDM research by several authors (Žáková et al., 2006; Lawrynowicz & Potoniec, 2011; Vavpetič & Lavrač, 2013). In the work of Žáková et al. (2006) an engineering ontology of Computer-Aided Design (CAD) elements and structures was used 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) describe and evaluate the 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, Lavrač, & Tolar, 2008) which supports semantic subgroup discovery in gene expression data. SDM-SEGS extends and generalizes this approach by allowing users 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 ontologies to constrain and guide a top-down search of hierarchically structured hypotheses space. SDM-Aleph, which is built using the inductive logic programming system Aleph (Srinivasan, 2007) does not have the limitations of SDM-SEGS, imposed by the domain-specific algorithm SEGS. Additionally, in the learning process SDM-Aleph can accept any number of OWL ontologies as a background knowledge. Like SDM-Aleph, the Hedwig semantic subgroup discovery algorithm (Vavpetič, Podpečan, & Lavrač, 2014), which is a generalization of the SDM-SEGS algorithm, also accepts any number of owl ontologies as input.

As an example, let us take the analysis of breast cancer data set, originally presented in Sotiriou et al., 2006. Use of the Hedwig algorithm resulted in the discovery of 10 rules, describing a subgroup of breast cancer patients. The resulting set of ranked rules (the rules are ranked based on their lift value, a metric described in Chapter 4) is shown in Figure 2.2, where the conjuncts explain different subgroups of breast cancer patients in terms of ontology concepts. This work showed that cytoskeletal formation and the regulation of the mitotic cell cycle best explain breast cancer gene expression (Vavpetič et al., 2014).



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 2.2: An example output of a semantic rule learning, where rules are learned from the breast cancer data set. All the rules represent subgroups of genes, differentially expressed in patients with breast cancer.

2.2.4 The Hedwig semantic data mining algorithm

We describe the semantic subgroup discovery system Hedwig (Vavpetič, Novak, Grčar, Mozetič, & Lavrač, 2013; Vavpetič, 2016). Compared to standard subgroup discovery algorithms, Hedwig uses domain ontologies to structure the search space and to formulate generalized hypotheses using ontology terms as conjuncts of if-then rules. Other semantic subgroup discovery algorithms are either specialized for a specific domain (Trajkovski, Železný, Lavrač, & Tolar, 2008) or adapted from systems that do not take into account the hierarchical structure of background knowledge (Vavpetič & Lavrač, 2013). Hedwig overcomes these limitations as it is designed to be a general purpose semantic subgroup discovery system. As Hedwig was shown to be a state-of-the-art semantic data algorithm by Vavpetič (2016), we used Hedwig in the experiments, described in this chapter.

In Hedwig, the semantic data mining task addressed takes three types of inputs: the training examples, domain knowledge, and a mapping between the two. The training examples are expressed as RDF triplets (Resource Description Framework i.e. subject-predicate-object; e.g., gene X suppresses gene Y) and represent the data instances of our data set. Domain knowledge is in the form of ontologies and represents the background knowledge associated with the data set. The third input is the object-to-ontology mapping which associates each RDF triplet with an appropriate ontological concept. We refer to these object-to-ontology mappings as 'annotations' and say that a particular object x is *annotated by* an ontology term o if the pair (x, o) appears in the mapping. As the annotations represent the existing knowledge about the data instances, they are known in advance. The set S of objects in our data set is split into a set of 'interesting' target class objects (for example, genes enriched in a particular biological experiment) and a set of non-target objects (non-enriched genes). Our approach finds a hypothesis (a predictive model or a set of descriptive patterns), expressed by domain ontology terms explaining the given empirical data. This hypothesis is expressed as a logical expression of ontology terms that best describe as many target objects and as few non-target objects as possible.

Semantic subgroup discovery by Hedwig results in logical descriptive rules. Take as example the rule, induced from a data set with 200 examples, 100 of which are positive:

$$\texttt{Class(X)} \leftarrow \texttt{C1(X), R(X,Y), C2(Y)}\ [80,20].$$

The rule is a first-order logical expression, where variables such as `X` and `Y` represent sets of input instances, `R` is a binary relation between the examples, and `C1` and `C2` are ontological concepts. This particular rule correctly covers 80 positive (target class) examples (True Positives, TP) and 20 negative examples (False Positives, FP) in training set S. The particular rule can be interpreted as follows. If an example x from example set X is annotated with concept C_1 and is related with another example (y from Y) via relation R, and y is annotated with concept C_2, then the conclusion $Class(X)$ holds. Note that in this rule X is a global (universally quantified) variable, and Y is a local (existentially quantified) variable introduced in the rule body. The rule condition is true for 100 input instances (also called *coverage*), and the *support* of the rule is 0.5, computed as $\frac{TP+FP}{|S|}$. The *precision* of the rule, calculated as $\frac{TP}{TP+FP}$, is 0.8. In our experiment, we use the *lift* metric to evaluate the performance of the proposed algorithm. The lift of a rule 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 an empty rule that classifies all examples as positive) and is calculated as $\frac{\text{precision}}{|S_+|/|S|}$. In our example, the lift of the rule is $\frac{0.8}{0.5} = 1.6$.

Like many ILP and RDM algorithms, the Hedwig algorithm uses beam search, where the beam contains the best N rules found so far. It starts with the default rule which covers all the input examples. In every search iteration, each rule from the beam is specialized via one of the four operations: (1) replace the predicate of a rule with a predicate that is a sub-class of the previous one, (2) negate predicate of a rule, (3) append a new unary predicate to the rule, or (4) append a new binary predicate, introducing a new existentially quantified variable, where the new variable has to be 'consumed' by a literal, which has to be added as a conjunction to the body of this clause in the next step of rule refinement.

Hedwig learns rules via 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 extended rule improves the probability of rule consequence, calculated as $p(\text{rule}) = \frac{|\{\text{positive examples covered by the rule}\}|}{|\{\text{all examples covered by the rule}\}|}$. To perform this check, we use the redundancy coefficient first introduced in the work of Hämäläinen (2010), which estimates whether the coefficient $\frac{p(\text{new rule})}{p(\text{old rule})}$ is greater than 1. If it is not, then the new 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 set, 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, Kavšek, Flach, & Todorovski, 2004), leverage and lift. The latter is the default choice in Hedwig and was used in our experiments.

In addition to a financial use case (Vavpetič, Novak, Grčar, et al., 2013), Hedwig was shown to perform well in a biological setting, analyzing DNA aberration data for various cancer types (Adhikari, Vavpetič, Kralj, Lavrač, & Hollmén, 2016), where it was a part of a three-step methodology, together with mixture models and banded matrices. In that analysis, additional background knowledge in the form of several ontologies obtained from various sources was used: hierarchical structure of multiresolution data, chromosomal location of fragile sites, virus integration sites, cancer genes, and amplification hotspots. A detailed examination of the methodology is provided in the next chapter.

2.3 Banded Matrix Visualization

This section presents the related work on banded matrix algorithms, which—along with the related work on SDM presented in Section 2.2—composes the related work for the research presented in Chapter 5.

Visualization is an important ingredient of data mining because it presents insights into complex data sets by communicating their key aspects (Tufte, 1986). Furthermore, providing information in the visual format is one of the fastest and most comprehensible methods for application area specialists. Data is often represented in a matrix form and the research community has developed numerous methods for matrix visualization (C.-H. Chen et al., 2004; H.-M. Wu, Tien, & Chen, 2010).

While binary matrices (matrices with entries equal to either 0 or 1) are frequently used in data mining (perhaps the most notable example of binary matrices being market basket data analysis), the concept of banded matrices has its origins in numerical analysis. The computational effort of multiplying matrices is much smaller when matrices are banded (i.e. the non-zero elements of a matrix only appear on positions that form a single band going from the top left to the bottom right corner of the matrix). The interest of the numerical community is usually to reduce the total bandwidth of a matrix, where the bandwidth of a matrix with entries a_{ij} is defined as the smallest value k such that if $|i - j| > k$, then $a_{ij} = 0$. This differs from data mining, where the goal is to find a matrix structure as close to a banded one with the underlying assumption that the data analyzed is noisy and contains outliers. Banded matrices and their relation to data analysis were initially studied by (Garriga et al., 2011). Garriga et al. (2011) presented an overview of several algorithms to find optimal permutations of rows (and sometimes columns) that best expose the banded structure of a matrix.

In Chapter 5 we conducted experiments with three algorithms: minimal banded augmentation (MBA), bidirectional MBA (biMBA), and a barycentric approach (first proposed by (Sugiyama, Tagawa, & Toda, 1981), and first used in the context of discovering a banded structure of binary matrix by Garriga et al. (2011)). Given that the performance of the biMBA method was superior to both MBA and the barycentric method, we used this method for visualizing the results of semantic rules and clusters in Chapter 5.

Chapter 3

HinMine: Heterogeneous Information Network Mining with Information Retrieval Heuristics

This chapter presents an advanced approach to mining heterogeneous information networks by decomposing them into homogeneous networks. The proposed HinMine methodology is based on previous work that classifies nodes in a heterogeneous network in two steps. In the first step the heterogeneous network is decomposed into one or more homogeneous networks using different connecting nodes. We improve this step by using new methods inspired by weighting of bag-of-words vectors mostly used in text mining and information retrieval, where larger weights are assigned to nodes which are more informative and characteristic for a given class of nodes. In the second step, the resulting homogeneous networks are used to classify data either by network propositionalization or label propagation. We propose an adaptation of the label propagation algorithm to handle imbalanced data and test several classification algorithms for propositionalization. The new methodology is tested on three data sets with different properties. For each data set, we perform a series of experiments and compare different heuristics used in the first step of the methodology. We also use different classifiers which can be used in the second step of the methodology when performing network propositionalization.

This chapter is structured as follows. Section 3.1 provides the motivation for the work described in this chapter. We also analyze the drawbacks of the existing methods, described in Section 2.1.5 in dealing with prolific intermediary nodes and with imbalanced data sets, and provide a framework that can be used to modify the algorithm. In Section 3.2, we propose improvements to the basic methodology by introducing text-mining inspired weight heuristics on intermediary nodes and by modifying starting weights for the label propagation algorithm. In Section 3.3, we present the results of the proposed methodology on three data sets.

3.1 Motivation

The methodology proposed in this chapter presents an extension of existing methods for heterogeneous network mining using various text mining inspired heuristics, described in Section 2.1.5.1. We made two significant improvements to the original methodology proposed by Grčar et al. (2013): we improved the network decomposition and the label propagation approach. In this section, we present the motivation for our improvements, while a more extensive overview of the related work is provided in Section 2.1 and the actual approach to mining heterogeneous networks is described in Section 3.2.

3.1.1 Improved network decomposition

In the field of text mining, most formulas for calculating the bag-of-words vector (a complete list of formulas used is provided in Section 3.2.2.1) share the same basic structure for weighting a term t in a document d:

$$w(t,d) = f(t,d) \cdot w(t), \tag{3.1}$$

where $f(t,d)$ is the frequency of term t in document d and $w(t)$ is a function that depends on the given term (and, implicitly, on the entire document corpus we are analyzing), and not on the particular document. The function w can be interpreted as a weighting function that determines the importance of a given term. For example, in the standard term-frequency (`tf`) scheme, function w is identically equal to 1, meaning all terms in a document receive equal weight. The term-frequency inverse-document (`tf-idf`) frequency scheme is designed to lower the weights for the terms that appear in a large number of documents. Other weighting schemes aim to lower the weights of terms that appear in all target classes with a similar frequency. By transferring the same rationale from text mining to network analysis, we aim to improve the network decomposition step of the described methodology by decreasing the link weights (1) when intermediate nodes connect to a large number of nodes (links induced by authors who wrote a lot of papers are less likely to be informative) and (2) when intermediate nodes connect to nodes from different classes (links induced by authors that write in many different fields are less informative). We therefore use network-analogues of function w used in text mining schemes and translate function f of Equation (3.1) to a network setting.

To transform the function f, we first rewrite Equation (3.1). The value $f(t,d)$ is the number of times t appears in d, meaning that we can understand Equation (3.1) as the sum of $w(t,d)$ over all appearances of t in document d, i.e. if we consider a document to be an ordered tuple $(t_1, t_2, \ldots, t_N)$ of terms, Equation 3.1 equals to

$$w(t,d) = \sum_{\substack{1 \le i \le N \\ t_i = t}} w(t). \tag{3.2}$$

This form of equation clearly emphasizes that the full weight of a term in a document is calculated by *summing* the weights of appearances of the term in the document.

In the network decomposition step of the methodology described in Section 2.1.5, we explained that the weight of the link between two base nodes (e.g., two papers) is equal to the number of intermediate nodes (e.g., authors) that are linked to (e.g., are authors of) both papers. In other words, we can say that the weight of the link between nodes v and u is

$$w(v,u) = \sum_{\substack{m \in M \\ m \text{ is linked to } v \text{ and } u}} 1, \tag{3.3}$$

where M is the set of all intermediate nodes.

Equation (3.3) can now be adapted and extended in a similar way as the term-frequency weights are adapted in text mining. In the text mining case, we replace 1 with $w(t)$, where w is a weighting function that penalizes the terms appearing in many documents or in documents from different classes. Similarly, in the network setting, we replace value 1 used in the calculation of $w(v,u)$ with a function $w(m)$, where w penalizes intermediate nodes that link to many base nodes or those that link the base nodes from different classes. The

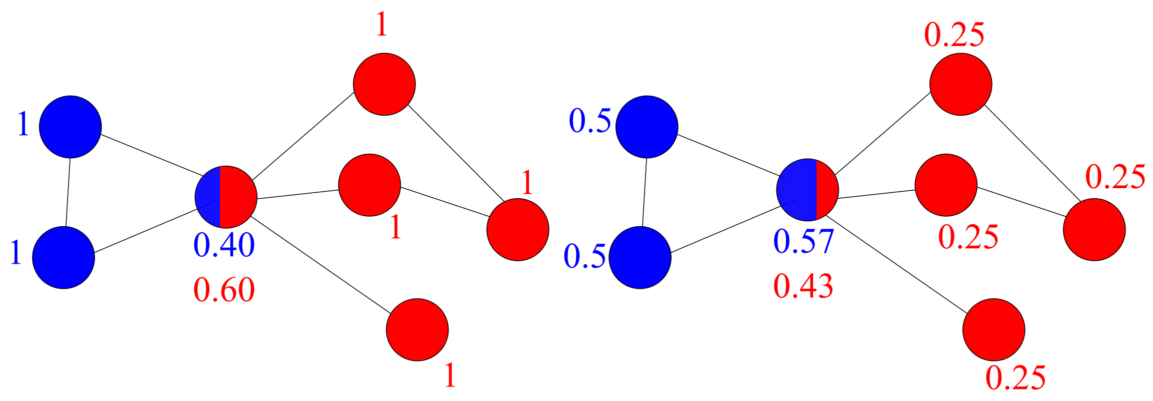


Figure 3.1: Results of a label propagation algorithm on an imbalanced data set.

result is the following formula for computing the weight of the link between u and v

$$w(v,u) = \sum_{\substack{m \in M \\ m \text{ is linked to } v \text{ and } u}} w(m). \tag{3.4}$$

The novelty of our work concerns the following extension of the existing methodology for classification in heterogeneous networks via network decomposition: by using different functions w in Equation (3.4) we are able to provide more informative weights used in the decomposition step. Using functions analogous to the inverse document frequency function, we improve the methodology by constructing a network in which links induced by highly connected intermediate nodes will not dominate in the homogeneous networks. For example, in the original methodology, adding an intermediate node that connects to all base nodes would cause the homogeneous network to be fully connected and would therefore result in nodes with similar personal PageRank values. Adding such a node in our scenario would still induce a fully connected network, however the additionally added links would have low weights (because the added intermediate node would have a weight) and would not affect the PageRank calculation. As the base nodes of a network are labeled, we can also take the class labels into account when calculating the weights of intermediate nodes. In particular, we introduce functions w (described in detail in Section 3.2.2.2) that give low weights to links induced by nodes that connect several classes, resulting in improved classification accuracy when such nodes cause authority to spread over a large portion of a network.

3.1.2 Label propagation and improvement

In our work, we also improved on the second step of the decomposition-based methodology presented by Grčar et al. (2013). Along with classification through propositionalization, we also use the label propagation to classify the nodes of homogeneous networks that arise from decomposing the heterogeneous network. To classify the original heterogeneous network, decomposed into k homogeneous networks, *all* available connections from all k homogeneous networks can be used. We construct a new network with the same set of nodes as in the original network. The weight of a link between two nodes is calculated as the sum of link weights in all homogeneous networks. In effect, if the decomposed homogeneous networks are represented by adjacency matrices $M_1, M_2, \ldots, M_k$, the new network's adjacency matrix equals $M_1 + M_2 + \cdots + M_k$.

The label propagation algorithm, as defined in (D. Zhou et al., 2004), works by propagating class labels from all members belonging to a certain class. By doing so, the algorithm

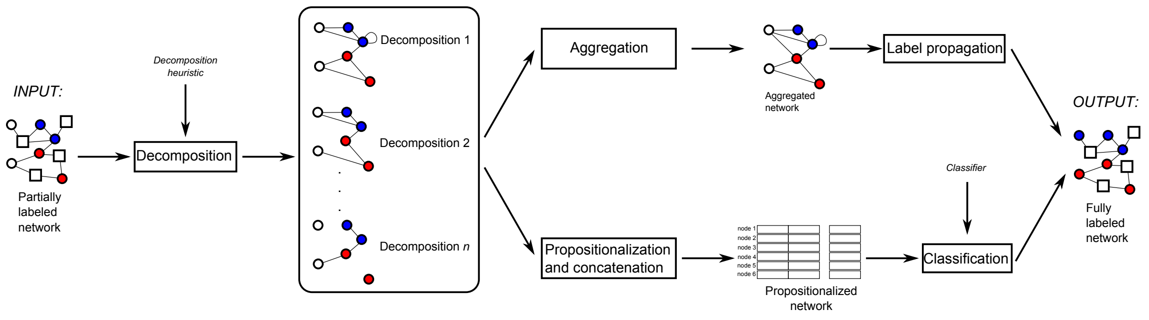


Figure 3.2: Overview of the proposed methodology. In the first step, an input partially labeled heterogeneous network is decomposed into one or more homogeneous networks. In the second step, these decompositions are merged and used to classify the data using label propagation (top branch) or each decomposition is used to calculate feature vectors for each base node in the network (bottom branch). In the latter case, the feature vectors are concatenated and the result can be used by any classifier (we use the k-NN, centroid based and the SVM classifiers) to classify the unlabeled nodes.

may overestimate the importance of larger classes (those with more instances) when data is imbalanced. We propose a modification to the label propagation algorithm which addresses this issue. For example, the left-hand side of Figure 3.1 shows an example in which the label propagation algorithm will classify the central node as belonging to the larger class simply because it has three neighbors of the red and only two neighbors of the blue class. This, in some cases, may not be a desired outcome. It could be argued that the node we wish to classify is actually adjacent to *all* elements of the blue class, but only *some* elements of the red class. Therefore, in the relative sense, blue nodes cast a stronger vote than red nodes. Consider Figure 3.1 to inspect the results of label propagation on an imbalanced data set. If we run the original label propagation algorithm, each labeled node begins the iteration with a weight of 1 for the class they belong to. In each step of the iteration, every node collects votes from its neighboring nodes and adds a portion (defined by α which was set to 0.6 in this example) of its original weight. In this case (on the left hand side), the central node receives a proportional vote of $\frac{2}{5} = 0.40$ from the blue class and a vote of $\frac{3}{5} = 0.60$ from the red class. However, using our modified weights (on the right hand side), the labeled nodes start with a weight of $\frac{1}{2}$ for the blue class with two nodes and $\frac{1}{4}$ for the red class with four nodes. Because of this, the proportion of votes for the blue class increases to 0.57. This is justified by proportional voting and the fact that the central node receives the highest possible vote from the blue class consisting of only two nodes, while it does not receive the highest possible vote from the red class.

3.2 The HinMine Methodology

In compliance with the motivation presented in Section 3.1, this section presents two improvements to the existing methodology. It first addresses the issue of handling of imbalanced data sets, followed by the presentation of a novel edge weighting approach used in the construction of homogeneous networks from the original heterogeneous network. The proposed approach named HinMine (Heterogeneous information network Mining methodology) is composed of a heterogeneous network decomposition step, followed by a classification step in which classification is performed either by propositionalization or label propagation. Figure 3.2 shows an overview of the proposed approach.

3.2.1 Imbalanced data sets and label propagation

The reasoning, described in Section 3.1.2, has made us believe that the label propagation approach may not perform well if the data is highly imbalanced, i.e. if the frequencies of class labels are noticeably different. In Section 2.1.5.2 we described that in each step of the iteration, matrix $F(t)$ is calculated as $F(t) = \alpha SF(t-1) + (1-\alpha)Y$, where S is a matrix calculated from the adjacency matrix of the network and each column of the zero-one matrix Y represents one of the possible labels in the network. We propose an adjustment of the label propagation algorithm by changing the initial label matrix Y so that larger classes will have less effect in the iterative process. The value of label matrix Y in this case is no longer binary (i.e. 0 or 1), but it is set to $\frac{1}{|c_j|}$ if node i belongs to class c_j and 0 otherwise.

If the data set is balanced (all class values are equally represented), then the modified matrix Y is equal to the original binary matrix multiplied by the inverse of the number of class values. This, along with Equation (2.3), means that the resulting prediction matrix only changes by a constant and the final predictions remain unchanged. However, if the data set is imbalanced, smaller classes now have a larger effect in the iterative calculation of F^*. This prevents votes from more frequent classes to outweigh votes from less frequent classes.

3.2.2 Text mining inspired weights

We first present weighting of terms in the construction of bag-of-words (BOW) vectors. We then explain how we can draw an analogy between BOW construction and extraction of homogeneous networks from heterogeneous networks. This can be done by replacing *terms* with *intermediate nodes* and *documents* with *network base nodes*, thus also implicitly replacing the relation "term-*appears-in*-text" with the relation "intermediate node-*is-linked-to*-base node".

3.2.2.1 Term weighting in text mining

In the bag-of-words vector construction one feature vector represents one document in a corpus of documents. In this vector, the i-th component corresponds to the i-th term (a word or a n-gram) that appears in the corpus. The value of the feature depends primarily on the frequency of the term in the particular document. We describe several methods for assigning feature values. We use the following notations: $f(t,d)$ denotes the number of times that term t appears in document d. The term D denotes the corpus (a set of all documents). We assume that the documents in the set are labeled, each document belonging to a class c from a set of all classes C. The number of all documents belonging to class c is denoted as $|c|$, while the number of all training documents that do not belong to class c is denoted as $|\neg c|$. We use the notation $t \in d$ to describe that a term t appears in document d. Where used, $P(t)$ is the probability that a randomly selected document contains term t, and $P(c)$ is the probability that a randomly selected document belongs to class c. We use $|d|$ to denote the length (in words) of a document, and *avgdl* denotes the average document length in the corpus. The constants b and k, used in the `bm25` weighting, are free variables of the weighting scheme and were set to their default values of 0.75 and 1.5, respectively, in our experiments.

Table 3.1 shows different methods for term weighting. Term frequency (`tf`) weights each term with its frequency in the document. Term frequency–inverse document frequency (`tf-idf`) (Jones, 1972) addresses the drawback of the `tf` scheme, which tends to assign high values to common words that appear frequently in the corpus – the weights of terms are

Table 3.1: Term weighing schemes and their formulas in text mining.

Scheme	Formula
`tf`	$f(t,d)$
`tf-idf`	$f(t,d) \cdot \log\left(\frac{\lvert D\rvert}{\lvert\{d' \in D : t \in d'\}\rvert}\right)$
`chi`^{2}	$f(t,d) \cdot \sum_{c \in C} \frac{(P(t \wedge c)P(\neg t \wedge \neg c) - P(t \wedge \neg c)P(\neg t \wedge c))^2}{P(t)P(\neg t)P(c)P(\neg c)}$
`ig`	$f(t,d) \cdot \sum_{c \in C, c' \in \{c, \neg c\}, t' \in \{t, \neg t\}} \left(P(t',c') \cdot \log \frac{P(t' \wedge c')}{P(t')P(c')}\right)$
`gr`	$f(t,d) \cdot \sum_{c \in C} \frac{\sum_{c' \in \{c,\neg c\}} \sum_{t' \in \{t, \neg t\}} \left(P(t',c') \cdot \log \frac{P(t' \wedge c')}{P(t')P(c')}\right)}{-\sum_{c' \in \{c, \neg c\}} P(c') \cdot \log P(c')}$
`Δ-idf`	$f(t,d) \cdot \sum_{c \in C} \left(\log \frac{\lvert c\rvert}{\lvert\{d' \in D : d' \in c \wedge t \in d'\}\rvert} - \log \frac{\lvert\neg c\rvert}{\lvert\{d' \in D : d' \notin c \wedge t \notin d'\}\rvert}\right)$
`rf`	$f(t,d) \cdot \sum_{c \in C} \log\left(2 + \frac{\lvert\{d' \in D : d' \in c \wedge t \in d'\}\rvert}{\lvert\{d' \in D : d' \notin c \wedge t \notin d'\}\rvert}\right)$
`bm25`	$f(t,d) \cdot \log\left(\frac{\lvert D\rvert}{\lvert\{d' \in D : t \in d'\}\rvert}\right) \cdot \frac{k+1}{f(t,d) + k \cdot \left(1 - b + b \cdot \frac{\lvert d\rvert}{\text{avgdl}}\right)}$

multiplied by the logarithm of the inverse of the number of documents the term appears in, thus decreasing the importance of terms which are common in all documents of the corpus. The same drawback is addressed by the `bm25` weighting scheme (also known as the Okapi BM25), first proposed in (Robertson & Walker, 1994). When the Okapi BM25 measure was first introduced, it was used to evaluate how well a query of several words matches up to a given document. The score is calculated as the sum of simple scoring functions, each of the scoring functions evaluating only one of the query words, and we use this simple scoring function as the basis for the bag-of-words construction.

The χ^2 (`chi`^{2}) weighting scheme (Debole & Sebastiani, 2004) attempts to correct another drawback of the `tf` scheme (one which is not addressed by the `tf-idf` scheme) by additionally taking class value of processed documents into consideration. Mimicking the χ^2 distribution used in statistics, this scheme measures the dependency of a given term and a given class label in the data set. The scheme penalizes terms that appear in documents of all classes (i.e. when the term is not dependent on any particular class), and favors terms which are specific to some classes. Information gain (`ig`) (Debole & Sebastiani, 2004) also uses class labels to improve term weights, however it uses an information-theoretic approach by measuring the amount of information about one random variable (the class of a document) gained by knowledge of another random variable (the appearance of a given term). The gain ratio scheme (`gr`) is similar to the information gain, but is normalized by the total entropy of the class labels in the data set. The Δ-idf (`Δ-idf`) (Martineau & Finin, 2009) and relevance frequency (`rf`) (Lan et al., 2009) attempt to merge ideas of `tf-idf` and both above class-based schemes by penalizing both common and non-class-informative terms.

3.2.2.2 Intermediate node weighting in homogeneous network construction

In this section, we present the translation of the text mining weighting schemes for bag-of-words vector construction into the weighting schemes for node weighting in network analysis.

Example 3.2.1. Let us revisit the example from Section 2.1.5 in which two papers are

Table 3.2: Heuristics for weighting intermediate nodes in the decomposition step of constructing homogeneous networks from the original heterogeneous network.

Scheme	Formula
`tf`	1
`tf-idf`	$\log\left(\frac{\lvert B\rvert}{\lvert\{b \in B : (b,m) \in E\}\rvert}\right)$
`chi`^{2}	$\sum_{c\in C} \frac{(P(m \wedge c)P(\neg m \wedge \neg c) - P(m,\neg c)P(\neg m, c))^2}{P(m)P(c)P(\neg m)P(\neg c)}$
`ig`	$\sum_{c\in C, c'\in\{c,\neg c\}, m'\in\{m,\neg m\}} \left(P(m' \wedge c') \log \frac{P(m' \wedge c')}{P(c')P(m')}\right)$
`gr`	$\sum_{c\in C} \frac{\sum_{c'\in\{c,\neg c\}} \left(\sum_{m'\in\{m,\neg m\}} P(m' \wedge c') \log\left(\frac{P(m'\wedge c')}{P(c')P(m')}\right)\right)}{-\sum_{c'\in\{c,\neg c\}} P(c') \log P(c')}$
`Δ-idf`	$\sum_{c\in C} \left\lvert \log \frac{\lvert c\rvert}{\lvert\{b \in B : n \in c \wedge (b,m) \in E\}\rvert} - \log \frac{\lvert\neg c\rvert}{\lvert\{b \in B : b \notin c \wedge (b,m) \notin E\}\rvert} \right\rvert .$
`rf`	$\sum_{c\in C} \left(\log\left(2 + \frac{\lvert\{n \in B : n \in c \wedge (n,m) \in E\}\rvert}{\lvert\{b \in B : n \notin c \wedge (b,m) \notin E\}\rvert}\right)\right)$
`bm25`	$\log\left(\frac{\lvert B\rvert}{\lvert\{b\in B:(b,m)\in E\}\rvert}\right) \cdot \frac{k+1}{1+k\cdot\left(0.25+0.75\cdot\frac{\deg(u)}{\mathrm{avgdeg}}\right)}$

connected by one link for each author they share. The resulting network is equivalent to a network in which two papers are connected by a link with a weight equal to the number of authors that wrote both papers (Figure 2.1). The method treats all authors equally, which may not be correct from an information content point of view. For example, if two papers share an author that only co-authored a small number of papers, it is more likely that these two papers are similar than if the two papers share an author that co-authored tens or even hundreds of papers. The first pair of papers should therefore be connected by a stronger weight than the second. Moreover, if papers are labeled by the research field, then two papers, sharing an author publishing in only one research field, are more likely to be similar than if they share an author who has co-authored papers in several research fields. Again, the first pair of papers should be connected by an edge with a larger weight.

We alter the term weighting schemes from text mining in such a way that they can be used to set weights to intermediate nodes in heterogeneous graphs (such as authors in our example). We propose that the weight of a link between two base nodes is calculated by summing the weights of all the intermediate nodes they share. In particular, if we construct a homogeneous network in which nodes are connected if they share a connection to a node of type T in the original heterogeneous network, then the weight of the link between nodes v and u should be equal to

$$\sum_{\substack{m\in T:\\(m,v)\in E\\(m,u)\in E}} w(m), \tag{3.5}$$

where $w(m)$ is the weight assigned to the intermediate node m. The value of $w(m)$ can be calculated in several ways. Table 3.2 shows the proposed intermediate node weighting heuristics corresponding to term weighting heuristics used in document retrieval (Table 3.1). The notation used is as follows. We denote with B the set of all nodes of the base node type, and with E the set of all edges of the heterogeneous network. When m is a node, $P(m)$ denotes the probability that a randomly selected labeled base node is connected to

Table 3.3: Comparison of weighting heuristics in text mining and network analysis.

Scheme	Property (text mining)	Property (network analysis)
`tf`	Counts each appearance of each word in a document equally.	Counts each intermediate node connecting two base nodes equally.
`tf-idf`	Gives a greater weight to a word if it appears only in a small number of documents.	Gives a greater weight to a intermediate node if it is connected to only a small number of base nodes.
`chi`^{2}	For a given term, measures the dependence of the class label of a document with respect to the given term in the document.	For an intermediate node, measures the dependence of the class label of a base node with respect to the link between the base node and the given intermediate node.
`ig`	Measures how much information about the class label of a document is gained by the appearance of a given term in the document.	Measures how much information about the class of a base node is gained by the existence of a link between a given intermediate node and the base node.
`gr`	A modification of the `ig` scheme, normalized by the total entropy of each class.	A modification of the `ig` scheme, normalized by the total entropy of each class.
`Δ-idf`	Measures the difference between the `tf-idf` value of a term when observing only documents of one class compared to the `tf-idf` value when observing the documents of other classes.	Measures the difference between the `tf-idf` value of an intermediate node when only observing the base nodes of one class compared to observing the nodes of other classes.
`rf`	Measures the ratio between the `tf-idf` value of a term when observing only documents of one class, compared to the `tf-idf` value when observing the documents of other classes.	Measures the ratio between the `tf-idf` value of an intermediate node when observing only the base nodes of one class compared to observing the nodes of other classes.
`bm25`	Gives greater weights to words that appear in short documents and words that appear in a small number of documents.	Gives greater weights to intermediate nodes that are connected to a small number of base nodes and to nodes of low degree.

the intermediate node m (in other words, $P(m) = \frac{|\{\text{base nodes, connected to } m\}|}{|\{alllabeledbasenodes\}|}$. A subset of the base nodes of the network is labeled, and each labeled node in the network belongs to one or more classes c from the set of all classes C. We use $P(c)$ to denote the probability that a random base node is in class c. The term $P(c \wedge m)$ denotes the probability that a random base node is both in the class c and linked to the intermediate node m.

Table 3.3 shows a comparison of weighting schemes in text mining and their newly developed analogs in network analysis. The `tf` weight is effectively used in (Grčar et al., 2013), where all base nodes weighed equally. Out of the remaining weights, the `ig`, `gr`, `chi`, `rf` and `tf-idf` weights transform naturally from the text mining setting to the network decomposition setting, and only some extra modifications were required on the last two schemes.

The `Δ-idf` weighting scheme, unlike other term weighting schemes, can assign negative weights to certain terms when constructing bag-of-words vectors. Since link weights in graphs are assumed to be positive both by the PageRank and the label propagation

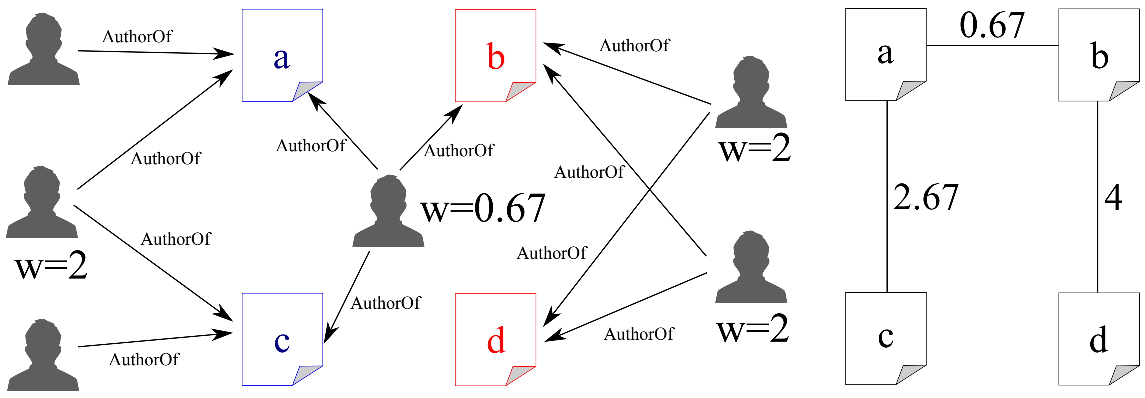


Figure 3.3: The construction of a homogeneous network from the toy network in Figure 2.1 using the χ^2 heuristic. The blue color denotes that the paper belongs to class 1 and the red color denotes class 2.

algorithm, we must change the weighting scheme before it can be used to construct homogeneous networks. We interpret the original weighting scheme in such a way that terms which receive negative values are informative about the term *not* being typical of a certain class. Therefore it is reasonable to take the absolute values of the weights in network construction, which explains the use of the absolute value in row 6 of Table 3.2 compared to the corresponding row of Table 3.1.

We also change the `bm25` scheme. The other weighting schemes are in the form $f(t,d) \cdot g(t,D)$, where $f(t,d)$ is the frequency of the term t in document d, and $g(t,D)$ is a function that depends on the term and the entire corpus. The Okapi BM 25 scheme cannot be decomposed in such a way, as the second function includes the average length of a given document. In our adaptation of the text mining functions to network weighting, each term translates to a connecting node and each document translates to a basic node, meaning that the document length translates into the degree of the basic node (the number of connecting nodes it is linked to). This means that the `bm25` weighting scheme must not return a weight for each connecting node (term), but a weight for each pair of intermediate node and basic node (term-document). For this reason, the formula in the last row of Table 3.2 depends on the choice of the base node u.

Example 3.2.2. Figure 3.3 shows the construction of a heterogeneous network from Figure 2.1, using the χ^2 heuristic. The weight of the central author m is calculated as the sum over both classes as

$$\frac{(P(m \wedge c)P(\neg m \wedge \neg c) - P(m, \neg c)P(\neg m, c))^2}{P(m)P(c)P(\neg m)P(\neg c)} \tag{3.6}$$

When c is the first (blue) class, we calculate the required values as $P(m \wedge c) = \frac{2}{4}, P(\neg m \wedge \neg c) = \frac{1}{4}, P(m, \neg c) = \frac{1}{4}$, $P(\neg m \wedge c) = 0$; $P(m) = \frac{3}{4}, P(\neg m) = \frac{1}{4}$, $P(c) = P(\neg c) = \frac{1}{2}$, yielding the value of expression (3.6) as

$$\frac{(\frac{2}{4} \cdot \frac{1}{4} - \frac{1}{4} \cdot 0)^2}{\frac{3}{4} \cdot \frac{1}{4} \cdot \frac{1}{2} \cdot \frac{1}{2}} = \frac{1}{3}.$$

When c is the red class, after calculating $P(m \wedge c) = P(\neg M \wedge c) = \frac{1}{4}, P(\neg m \wedge \neg c) = 0, P(m \wedge \neg c) = \frac{2}{4}$, we see that the second summand of expression (3.6) is $\frac{1}{3}$ and the total weight of author m is $\frac{2}{3}$.

The weights of the remaining authors are calculated in the same way. In our case, none of the other authors wrote papers from both classes, so their weights are all equal to 2.

The homogeneous network on the right hand side is constructed by summing the weights of all common authors for each pair of papers. We can see that the connection between papers a and b is weaker than the other connections because the common author was assigned a smaller weight.

3.3 Experimental Setting and Results

This section describes the experiments used to evaluate the performance of the proposed weighting schemes. We first describe the data sets and then present the experimental setup and results.

3.3.1 Data sets descriptions

We evaluated the proposed weighting heuristics on three separate data sets. The first data set (E-commerce) contains 30,000 base nodes split into two target classes. The target classes are highly imbalanced, allowing us to test the imbalance sensitivity of the modified label propagation algorithm. The second data set contains over 10 times more nodes than the first one, split into 11 target classes. It was used to test the scalability of our approach. The third data set contains 12,000 base nodes in 20 target classes and was used to test the performance of the algorithm on many target classes, each with a small number of nodes.

E-commerce data set The first data set contains data about customer purchases used in the PAKDD 2015 mining competition *Gender prediction based on e-commerce data* where we acchieved 22. place. The data set, available on the competition website[1] describes 30,000 customers. The data for each customer consists of the gender (the target variable), the start and end time of the purchase, and the list of products purchased. A typical product is described by a 4-part string (for example: `A3/B5/C2/D8`). The strings describe a 4-level hierarchy of products, meaning that the example product is the product $D8$ (or D-level category) which belongs to (A-level) category $A3$, sub-category (or B-level category) $B5$ and sub-sub-category (or C-level category) $C3$. The category levels are consistent, meaning that if two products belong to the same B-level category, they also belong to the same A-level category. The data set is highly imbalanced: 23,375 customers are women and 6,625 are men. Along with the purchase data, the original data also contains a timestamp when the purchase occurred. For the purpose of our experiments, we ignored this data which explains the superior performance of other approaches which took the time information into account.

For the purpose of our experiments, we ignored the temporal aspects of the data and only focused on the products purchased by the customers. This allowed us to view the data set as an implicitly defined heterogeneous network. The network consists of five node types: customers (the base node type) and four hierarchy levels. In this heterogeneous network, every purchase by a customer defines four edges in the heterogeneous network: one edge between the customer and each (sub)category to which the product belongs.

We constructed four homogeneous networks from the original heterogeneous network. In the first network, two customers are connected if they purchased the same product (same D-level item), i.e. if they are connected by a path in the original network

[1]https://knowledgepit.fedcsis.org/contest/view.php?id=107

that goes through a D-level item. In the second network, they are connected if they purchased a product in the same sub-subcategory (C-level item), in the third network they are linked if they purchased the same B-level item and in the fourth network they are connected if they purchased the same A-level item. The constructed networks are referred to as A-, B-, C- and D-level networks in the following sections. Our objective is to use the constructed networks to predict the gender of customers.

ACM papers data set The second data set is a collection of papers first used by Tang et al. (2008). The data set is freely available online[2] along with several other citation networks. For our experiments, we used a part of the ACM citation network. We chose this network because, unlike the DBLP network also available on the same website, the papers in the ACM data set are classified into one of 11 categories. The data set contains 2,381,688 papers and 10,476,564 citation relationships. The papers are divided into three major groups: book chapters, proceedings papers and journal papers.

For the purposes of our experiments, we reduced the data set to only include conference papers with authors which wrote at least two papers in order to eliminate isolated nodes. We did not perform any name disambiguation as this would be a demanding task that is not directly connected to our topic of research. We constructed a citation network containing two types of nodes–papers and their authors–and constructed a homogeneous network by converting each paper-author-paper path in the network into a link between the two papers. The resulting network contained 320,339 nodes and 3,531,263 links between them. In our experiments, the objective was to correctly predict the category (research field, for example Data Mining, Theoretical Computer Science, Neural Networks) of papers in the network. With this large network, we tested the scalability and limitations of our approach.

iMDB data set The third data set was first published by the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems (Cantador, Brusilovsky, & Kuflik, 2011) and is freely available[3]. The data set contains information extracted from the internet movie database[4] about 10,198 movies (the actors, directors, locations and tags, associated with each movie). Each movie is labeled with one or more genres it belongs to. Altogether there are 20 genres.

We focused on the movies and actors in the data set and constructed a heterogeneous network composed of actors and movies. Based on this network we constructed a homogeneous network of movies in which two movies are connected if they share a common actor. Using this movie-actor-movie network, our objective was to correctly predict at least one genre label assigned to each movie.

3.3.2 Experimental setting

In summary, note that we have two situations. The E-commerce data set is the most complex, having several types of nodes and relations. Therefore, the entire HinMine methodology needs to be applied, resulting in several possible decompositions of the heterogeneous network. On the other hand, the ACM and iMDB data sets are simpler, containing only one relation type and two node types. This means only one decomposition can be calculated, which simplifies the methodology described in Section 3.2. In the simplified case,

[2]https://aminer.org/citation
[3]http://grouplens.org/datasets/hetrec-2011/
[4]https://www.imdb.com

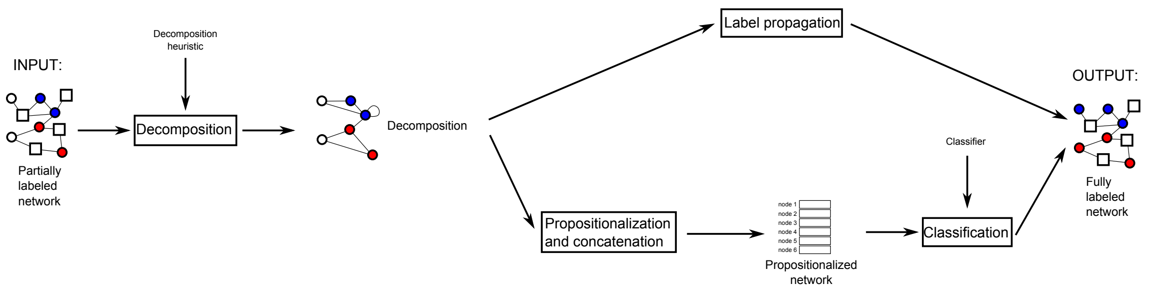


Figure 3.4: An overview of the methodology described by Figure 3.2 when we only construct one homogeneous network from the heterogeneous network. In contrast to Figure 3.2, only one homogeneous network is constructed in the decomposition step. Consequently, the aggregation step before label propagation is skipped as we only have one matrix to sum, and the concatenation step before classification is skipped as we only produce one feature vector per node during propositionalization.

presented in Figure 3.4, we construct one homogeneous network from the original heterogeneous network. However, in the following step, if we perform classification using label propagation, we do not need to aggregate several networks and can directly perform label propagation to classify the nodes. If we use the propositionalization approach, the propositionalization produces one feature vector per base node and no concatenation is required.

By analyzing the E-commerce data set, we were able to fully analyze the effects of using different decompositions and classifiers on classification performance. The results on this data set allowed us to demonstrate that the centroid-based classifier proposed by Grčar et al. (2013) was not always optimal and that a k-NN and SVM classifier can out-perform it. With these conclusions, we then performed the simplified experiments on the ACM and iMDB data sets because our goal is to analyze the effects of using different decomposition heuristics for homogeneous network construction.

3.3.3 Experiments on E-commerce data set

We performed four sets of experiments on the E-commerce data sets. We first describe the experiments and then discuss the results.

3.3.3.1 Experiment description

We performed four sets of experiments on the E-commerce data set. In this section, we present the experimental setup for the sets of experiments.

First experimental setup The first set of experiments was designed to verify the results of Grčar et al. (2013), which show that a centroid classifier, trained on Personalized PageRank feature vectors, performs as well as the more complex SVM classifier. The centroid classifier is a simple classifier that calculates a centroid vector for each target class as the sum of all vectors belonging to the class, divided by the number of instances in the class. It then classifies a new instance into the class whose centroid vector is the closest to the feature vector of the new instance. Grčar et al. (2013) showed that centroid P-PR vectors are easy to calculate from networks, making the centroid classifier highly scalable. However, the simplicity of the classifier means that it is not always the best performing classifier. We tested the performance of the centroid classifier, the k-nearest neighbors classifier (with k set to $1, 2, 5$ and 10), and the SVM classifier. Because the data set is imbalanced, we tested the SVM classifier both with uniform instance weights as well as weights proportional to the class frequencies. We only tested the k-nearest neighbors classifier with uniform instance weights. The tests were performed on feature vectors extracted from all four homogeneous networks. We randomly sampled 3,000 network nodes to train the classifiers and tested their performance on the remaining 27,000 nodes. The small size of the training set ensured that the training phase was fast, but the training set was still large enough to allow us to compare the classification algorithms we used.

Second experimental setup In the second set of experiments, we tested the heuristics used in the construction of the homogeneous networks. We tested three classifiers. We first used the SVM classifier using solely the Personalized PageRank vectors extracted from the network. As the results of the first experiment show that weights, proportional to the class frequencies, improve the classification accuracy of the SVM classifier, we used the same weights for this set of experiments. The second tested classifier is the label propagation classifier, defined by D. Zhou et al. (2004), which classifies the network nodes using the graph itself. The third classifier is the label propagation classifier with the starting matrix Y adjusted for the class frequencies, as proposed in Section 2.1.5.2. The goal of this round of experiments was to compare the label propagation classifier with the SVM classifier and evaluate whether the adjusted starting matrix Y has any effect on classifier performance. As in the first experiment, we trained the classifiers on a randomly sampled set of 3,000 nodes and tested their performance on the remaining 27,000 nodes.

Third experimental setup In the third round of experiments, we tested the performance of the label propagation and propositionalization-based classifiers on all four homogeneous networks. Based on the results of the first two sets of experiments, we used the SVM classifier for the propositionalization approach and the label propagation method with the modified starting matrix Y. As explained in Section 2.1.5, we constructed feature vectors for SVM classifiers by concatenating feature vectors of individual homogeneous networks. We constructed the adjacency matrix for the label propagation algorithm by summing the four adjacency matrices. One of the goals of the third round of experiments was to test the performance of the classifiers when trained on a large data set; for this reason we trained the classifiers on 90% of the data set and tested their performance on the remaining 10%. Compared to the first two sets of experiments, this setup was computationally much more demanding, however the accuracy of the SVM classifier was increased by approximatelly 5%.

Fourth experimental setup In the fourth round of experiments, we used the E-commerce data set to observe the effect of class imbalance on classifier performance. The full data set contains more female than male customers. We used the original data set to construct several data sets with different levels of class imbalances by using only a subsample of female customers and all of the male customers. We used a sample of $28, 30, 40, 50, 60, 70, 80, 90$ and 100% of all females in the data set (we chose the 28% as this is the ratio at which the data set becomes perfectly balanced). For each of the constructed data sets, we ran the original and modified label propagation algorithms.

In all the experiments we evaluated the accuracy of the classifiers using the *balanced accuracy* metric. This metric, which was used in the PAKDD'15 Data Mining Competition, is defined as follows:

$$\frac{\frac{|\{\text{Correctly classified male customers}\}|}{|\{\text{All male customers}\}|} + \frac{|\{\text{Correctly classified female customers}\}|}{|\{\text{All female customers}\}|}}{2}. \tag{3.7}$$

Table 3.4: Results of the first set of experiments on the E-commerce data set comparing the performance of different classifiers on the propositionalized network. SVM_1 marks the standard SVM classifier, while SVM_2 marks the classifier using balanced instance weights.

Classifier:	Centroid	1-nn	2-nn	5-nn	10-nn	SVM_1	SVM_2
A-level network	74.19%	63.61%	71.93	72.74%	74.36%	74.03%	**74.62%**
B-level network	70.78%	56.42%	59.17%	65.30%	67.73%	63.51%	**72.61%**
C-level network	64.71%	63.62%	67.21%	68.26%	71.65%	70.15%	**75.18%**
D-level network	60.08%	67.36%	70.39%	66.72%	66.06%	65.61%	**71.17%**

Table 3.5: Performance of the SVM classifier in the second round of experiments on the E-commerce data set comparing the effect of different weighting schemes on the classifier performance.

Scheme	A-level	B-level	C-level	D-level
`tf`	76.61%	74.00%	**77.34%**	**73.65%**
`chi`^{2}	77.80%	74.17%	76.86%	68.76%
`tf-idf`	**77.80%**	74.22%	77.23%	72.25%
`Δ-idf`	**77.80%**	74.14%	77.23%	72.52%
`rf`	**77.80%**	74.11%	76.81%	70.54%
`ig`	**77.80%**	74.12%	76.87%	68.72%
`gr`	**77.80%**	**74.32%**	77.12%	69.44%
`bm25`	76.61%	74.16%	76.26%	69.35%

Table 3.6: Performance of the label propagation classifier in the second round of experiments on the E-commerce data set comparing the effect of different weighting schemes on classifier performance.

Scheme	A-level	B-level	C-level	D-level
`tf`	75.52%	64.28%	63.60%	72.44%
`chi`^{2}	**76.02%**	65.15%	71.95%	72.75%
`tf-idf`	74.90%	63.83%	61.02%	72.48%
`Δ-idf`	74.90%	63.76%	61.05%	72.48%
`rf`	75.52%	64.28%	67.59%	72.55%
`ig`	**76.02%**	65.15%	**72.41%**	**72.96%**
`gr`	75.79%	**65.32%**	68.64%	72.52%
`bm25`	74.95%	63.98%	60.83%	72.48%

Table 3.7: Performance of the modified label propagation classifier in the second round of experiments on the E-commerce data set comparing the effect of different weighting schemes on the classifier performance.

Scheme	A-level	B-level	C-level	D-level
`tf`	77.16%	**74.75%**	77.28%	73.91%
`chi`^{2}	77.16%	74.44%	**77.61%**	73.82%
`tf-idf`	**77.20%**	74.70%	77.74%	73.76%
`Δ-idf`	**77.20%**	74.71%	77.74%	73.76%
`rf`	77.16%	74.59%	77.21%	**74.03%**
`ig`	77.16%	74.49%	77.59%	73.79%
`gr`	77.16%	74.70%	77.23%	73.80%
`bm25`	77.16%	74.70%	77.44%	73.76%

Table 3.8: The results of the third set of experiments on the E-commerce data set showing the balanced accuracies of the SVM and label propagation classifiers on the entire data set.

Scheme	SVM	Label propagation
`tf`	81.35%	77.06%
`chi`^{2}	81.78%	77.10%
`tf-idf`	**82.09%**	79.03%
`Δ-idf`	81.94%	**79.08%**
`rf`	81.49%	77.16%
`ig`	81.56%	77.12%
`gr`	81.74%	77.12%
`bm25`	81.53%	78.16%

3.3.3.2 Experimental results

The result of the first set of experiments, shown in Table 3.4, demonstrates a large difference in the performance of different classifiers. Similarly to Grčar et al. (2013), the simple centroid classifier performs well on feature vectors extracted from several different homogeneous networks. However, the classifier is consistently outperformed by the SVM classifier if the instance weights of the classifiers are set according to the class sizes. In contrast with the findings by Grčar et al. (2013) which opted for an efficient centroid classifier as a default classifier in all settings, our experiments show that the optimal classifier for the methodology are data set specific.

The results of the second set of experiments are shown in Tables 3.5, 3.6 and 3.7. When comparing the results of the two label propagation approaches the results demonstrate that label propagation with adjusted starting matrix has a large impact on the performance of the classifier, as the balanced accuracy increases by 1–2% in the case of the A- and D-level network and even more in the case of B- and C-level networks. This result confirms the intuition explained in Section 2.1.5.2 which motivated the construction of the adjusted starting matrix.

Different heuristics used in the construction of homogeneous networks also affect the final performance of all three classifiers. No heuristic consistently outperforms the others, meaning that the choice of the heuristic is data-dependent. The last conclusion of the second round of experiments is that the computationally demanding propositionalization method does not outperform the label propagation method. In all four networks choosing the appropriate heuristic and appropriate weights for the starting matrix allows the label propagation method to perform comparably to the SVM classifier.

Table 3.8 shows the results of the third set of experiments. In this experiment, the propositionalization-based approach clearly outperforms the label propagation algorithm. It is possible that this effect occurs because the network propositionalization approach, in particular the SVM classifier, requires more training instances (compared to the label propagation classifier) to perform well. A second explanation may come from the way the four networks were combined in our experiments (i.e. the concatenation approach in the case of network propositionalization and the matrix sum in the case of label propagation). By summing the adjacency matrices before performing label propagation, we implicitly assumed that the connections between the customers that purchased the same D-level product, are equally important as connections between customers that purchased the same A-level product. This may cause the amount of A-level edges to overwhelm the effects of the D-level edges, causing the resulting network to be very similar to the original A-level network. The propositionalization-based approach is less prone to errors of this type, as the idea of the SVM algorithm is to learn correct weights for elements of feature vectors. The SVM algorithm is therefore flexible enough to assign larger weights to the features, produced by the D-level network, if it estimates that these features are more important for classification. The second conclusion we can draw from the third set of experiments is that using weighting heuristics in the construction of the homogeneous networks is highly beneficial. With both classification methods the adjusted `Δ-idf` and `tf-idf` heuristics perform best.

Figure 3.5 shows the results of the fourth set of experiments. The chart compares the standard label propagation classifier with the modified imbalance-sensitive classifier on data subsets of varying levels of imbalances. We see that when we sample 28% of females into the data set, i.e. when the data set is completely balanced, both classifiers perform equally; this was expected, as the modified label propagation classifier returns predictions that are in this case merely a constant factor different from the predictions of standard label propagation. The difference in performances of the two classifiers then steadily increases as the data set becomes more and more imbalanced, confirming our hypothesis that the modified label propagation algorithm will perform better on imbalanced data sets. Note however that the difference between the classifiers does not arise from decreased performance of the standard label propagation classifier: while by increasing ratio of female customers the training data sets grew larger, enabling the label propagation classifier to improve its performance, the modified imbalance-sensitive label propagation classifier achieved a significantly better performance on the increased data sets. Moreover, the chart also shows a significant drop in classifier performance even when most of the data is used in the classification. As this phenomenon occurred in more than one run of our experiments, we believe it is not an

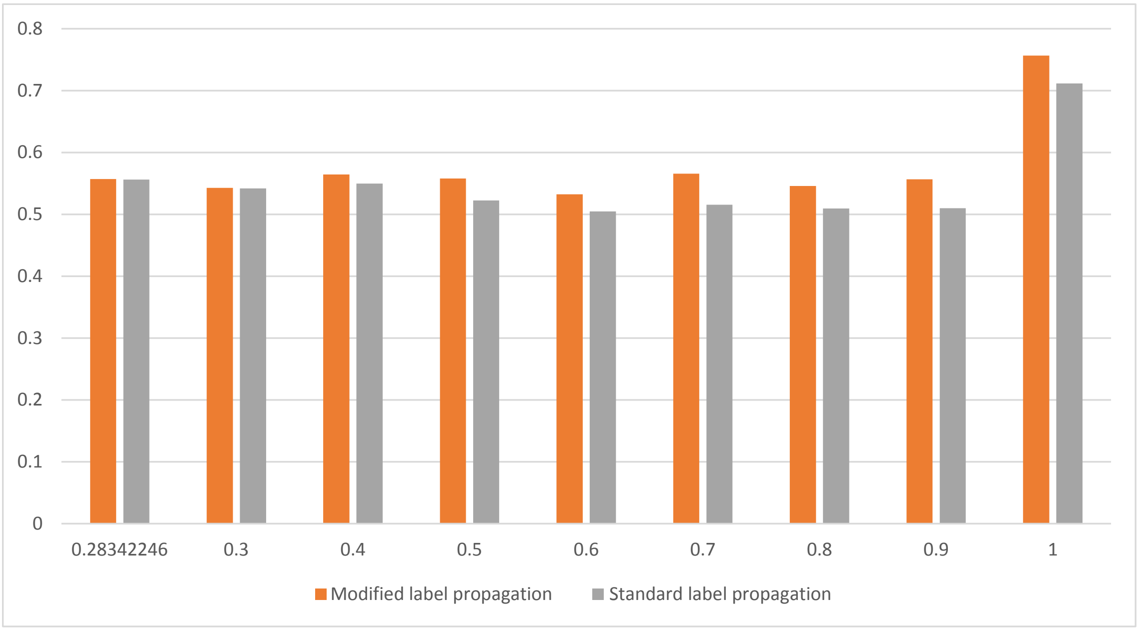


Figure 3.5: Performance of the standard and modified label propagation classifiers at different levels of class imbalance. The x axis shows the proportion of female customers chosen (at 28%, the data set is perfectly balanced) and the y axis shows the classification accuracy. The height of each column is the balanced accuracy score achieved by the modified or standard label propagation classifier.

anomaly, but rather that it is the result of the characteristics of the data set used. It shows that even when removing only a small portion of the nodes from the network, the structure of the network changes significantly and results in much poorer performance by the label propagation classifiers. In future, more advanced methods of sampling network data must be used (such as those described by Y. Wu et al. (2017)) as our experiments show that a simple random sample selection leads to significant drops in performance.

3.3.4 Experiments on ACM papers data set

Because the ACM data set is much larger than the E-commerce data set, it is computationally unfeasible to exactly repeat the experiments from the first data set. In this section, we describe the experiments on the ACM data set and their results.

3.3.4.1 Experiment description

Calculating the P-PR vectors of all base nodes in the network is both time consuming and memory intensive. Using our efficient speed-adapted algorithms (Kralj, Valmarska, Robnik-Šikonja, & Lavrač, 2015) that compute approximately one P-PR vector per second we would need 3–4 days to calculate the vectors for each weighting scheme, and as each P-PR vector has 320,339 dimensions and takes 30 megabytes of memory, 10,000 vectors occupy approximately 30 gigabytes of memory.

The second problem is that training times for most machine learning algorithms are too long. Given that our main objective is to evaluate the performance of different intermediate node weighting schemes (not classifiers), we decided to only use k-nearest neighbors classifiers to evaluate the weighting schemes. However, even this simple classifier is too

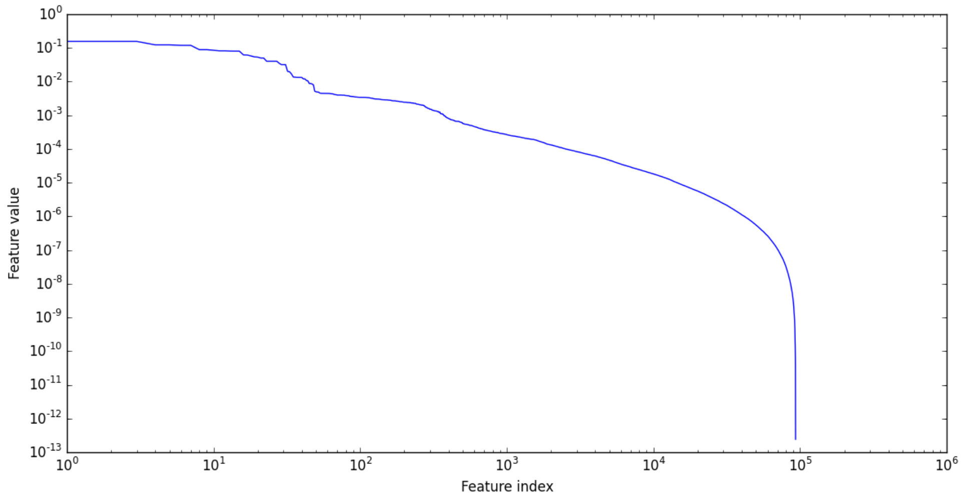


Figure 3.6: Sorted feature values of an example feature vector from the propositionalized ACM data set. The x axis shows the index of the features sorted by size, and the y axis shows the feature values.

slow to run on the full data set. To execute the algorithms in a reasonable time we used the following scenario:

- We sparsified the P-PR vectors by cutting off low ranking nodes. The motivation behind this can be seen in Figure 3.6 which shows that only a small number of papers in the network has P-PR values above 10^{-5}, so discarding the rest of the values should have little impact on the results while greatly decreasing the run time of the algorithm. We tested three different cutoff values to sparsify the data (0.001, 0.0001 and 0.00001) and re-normalized the data set for each cutoff value.

- We randomly selected 12,000 papers in the network and calculated their P-PR vectors for each homogeneous network. We split the data set into a training set (80%) and test set (20%) and tested the performance of the classifier. In performing the split, we followed the approach of Sechidis, Tsoumakas, and Vlahavas (2011) to account for the fact that some instances in the data set can be labeled with more than one category. To evaluate statistical significance, we repeated this step 5 times, testing all the decomposition heuristics on 5 different sets of 12,000 papers.

As the ACM papers data set contains papers that are labeled with 11 categories and each paper can be labeled with more than one category, we use the same performance metric as Grčar et al. (2013) and evaluate how well the classifier predicts the category of a paper when we consider its top 1, 2, 3 or 5 predictions. We did not examine the top 10 predictions as in Grčar et al. (2013) because, with 11 categories, the performance of a classifier would be nearly perfect.

3.3.4.2 Experimental results

Tables 3.9, 3.10 and 3.11 show the results of the experiments on the ACM data set. We show the accuracy and its standard deviation across five random resamples. The results indicate that our decision to cut off the near-zero values of PageRank vectors was well justified as there is no significant difference in performance due to cutoff values.

Table 3.9: Classification accuracy (in %) of the k-nearest neighbor algorithms on the ACM data set. The classifier was trained on P-PR vectors of nodes in homogeneous networks, obtained from the original heterogeneous network using eight weighting schemes. All values of the vectors, smaller than 0.001, were set to 0.

$k = 1$

top n	`gr`	`tf`	`ig`	χ^2	`rf`	`Δ-idf`	`tf-idf`	`bm25`
1	28.25 ± 2.23	26.84 ± 0.34	20.85 ± 3.45	18.95 ± 0.73	26.67 ± 0.55	19.94 ± 0.30	19.46 ± 0.65	$\mathbf{32.36 \pm 0.34}$
2	39.30 ± 5.29	$\mathbf{51.43 \pm 0.23}$	39.27 ± 7.47	34.84 ± 0.78	51.48 ± 0.31	35.10 ± 0.98	31.58 ± 0.58	41.18 ± 0.56
3	42.48 ± 0.85	52.78 ± 0.24	34.39 ± 10.70	30.48 ± 1.10	$\mathbf{53.65 \pm 0.86}$	32.63 ± 0.43	32.62 ± 0.51	42.22 ± 0.60
5	52.34 ± 0.49	62.14 ± 0.23	47.46 ± 8.83	45.35 ± 1.06	$\mathbf{62.27 \pm 0.51}$	46.52 ± 0.50	46.68 ± 0.41	52.61 ± 0.33

$k = 3$

top n	`gr`	`tf`	`ig`	χ^2	`rf`	`Δ-idf`	`tf-idf`	`bm25`
1	29.70 ± 2.39	$\mathbf{34.57 \pm 0.46}$	18.02 ± 2.73	19.14 ± 1.09	28.10 ± 1.11	19.75 ± 0.65	18.10 ± 2.38	32.62 ± 1.86
2	53.93 ± 0.66	$\mathbf{54.77 \pm 0.14}$	35.00 ± 7.24	49.26 ± 0.79	$\mathbf{54.93 \pm 0.58}$	36.64 ± 7.43	32.99 ± 3.30	46.02 ± 4.92
3	$\mathbf{59.64 \pm 1.29}$	58.57 ± 0.76	37.97 ± 11.43	53.10 ± 0.75	55.14 ± 7.36	43.24 ± 7.13	45.03 ± 6.48	49.34 ± 5.06
5	$\mathbf{69.87 \pm 1.18}$	67.62 ± 0.49	52.42 ± 12.97	62.45 ± 0.26	67.84 ± 0.57	53.33 ± 6.22	66.89 ± 7.56	58.81 ± 4.75

$k = 5$

top n	`gr`	`tf`	`ig`	χ^2	`rf`	`Δ-idf`	`tf-idf`	`bm25`
1	29.92 ± 2.70	$\mathbf{34.07 \pm 2.11}$	21.50 ± 3.60	26.12 ± 7.53	28.46 ± 1.10	19.43 ± 0.74	20.62 ± 4.77	28.48 ± 0.93
2	53.91 ± 1.39	$\mathbf{56.23 \pm 0.55}$	39.87 ± 8.58	50.48 ± 1.09	$\mathbf{55.64 \pm 0.91}$	45.27 ± 7.50	36.82 ± 4.95	54.71 ± 1.19
3	$\mathbf{61.06 \pm 1.47}$	$\mathbf{61.02 \pm 0.28}$	48.84 ± 10.40	60.41 ± 6.18	58.48 ± 6.00	53.37 ± 7.66	52.69 ± 3.99	58.86 ± 0.84
5	73.69 ± 1.90	69.47 ± 0.41	64.36 ± 12.48	$\mathbf{76.81 \pm 4.26}$	72.02 ± 0.68	65.14 ± 9.31	76.62 ± 1.49	68.30 ± 0.99

$k = 10$

top n	`gr`	`tf`	`ig`	χ^2	`rf`	`Δ-idf`	`tf-idf`	`bm25`
1	31.20 ± 2.81	$\mathbf{36.28 \pm 0.68}$	19.87 ± 1.00	34.77 ± 1.47	29.26 ± 0.93	26.27 ± 4.19	22.38 ± 4.54	33.47 ± 2.50
2	55.79 ± 0.50	$\mathbf{56.78 \pm 0.64}$	39.44 ± 7.17	51.50 ± 0.97	$\mathbf{56.40 \pm 0.69}$	44.67 ± 1.56	45.77 ± 10.03	49.99 ± 5.75
3	62.46 ± 1.22	65.03 ± 1.03	57.58 ± 8.58	64.44 ± 4.57	62.99 ± 1.31	$\mathbf{65.16 \pm 1.94}$	52.92 ± 8.49	58.94 ± 6.81
5	75.47 ± 1.75	76.82 ± 1.74	77.96 ± 5.85	$\mathbf{79.64 \pm 4.56}$	76.35 ± 2.91	79.36 ± 4.68	75.90 ± 4.04	69.64 ± 2.11

Table 3.12 shows the results of the label propagation experiments on the ACM data set. We can conclude that the performance of the label propagation algorithm is significantly worse than the performance of the k-nearest neighbor methods. There is no significant difference in performance when comparing the standard label propagation algorithm with the modified version introduced in Section 3.2.1. This result is not surprising. In Section 3.2.1, we explained that if the data set is perfectly balanced, the modified matrix Y is equal to the original starting matrix, multiplied by a constant, and thus the final predictions do not change. As the ACM data set is more balanced than the E-commerce data set, Table 3.12 confirms our expectation that in a balanced data set, the modified label propagation algorithm will perform identically to the original algorithm.

The weighting schemes perform differently depending on the data set. The `gr` and `ig` schemes show similar performance overall. As the two schemes are calculated using a similar formula (the `ig` weight is a normalized version of the `gr` weight) this is an expected result. There is a distinction between well performing weighting schemes, namely the `gr`, `tf`, `ig` and `chi` weighting schemes, and other schemes, however the difference is hard to evaluate. In order to provide a better visualization of performance, we use the methodology first described by Demšar (2006) to compare weighting schemes on multiple classifier/data set pairs. On each of the five random resamples of the data (as described in Section 3.3.4.1), we ran the classification methodology using each of the 8 decomposition heuristics. We then use the Friedman test to evaluate the null hypothesis that all weighting schemes perform equally. We rejected the hypothesis at a p-value of 10^{-8} and performed a post-hoc Nemenyi test to assess different algorithms. The test applies the Bonferroni correction to account for multiple hypothesis testing and evaluates the average ranking of each weighting scheme. The results are shown in Figure 3.7. The best weighting scheme for the ACM data set is the `gr`, followed by the `chi` and `tf` weighting schemes. Based on this result and the results in Tables 3.9, 3.10, and 3.11, we recommend using the gain ratio (`gr`) weighting scheme to classify papers in the ACM paper-author-paper network.

Table 3.10: Classification accuracy (in %) of the k-nearest neighbor algorithms on the ACM data set. The classifier was trained on P-PR vectors of nodes in homogeneous networks, obtained from the original heterogeneous network using eight weighting schemes. All values of the vectors, smaller than 0.0001, were set to 0.

				$k=1$				
top n	gr	tf	ig	χ^2	rf	Δ-idf	tf-idf	bm25
1	$\mathbf{31.52 \pm 2.16}$	19.07 ± 0.91	19.17 ± 0.52	19.20 ± 0.65	22.11 ± 6.00	32.66 ± 0.49	14.76 ± 0.66	26.72 ± 1.07
2	$\mathbf{42.74 \pm 2.77}$	35.40 ± 1.22	26.40 ± 0.66	35.50 ± 1.00	38.53 ± 5.84	41.87 ± 0.22	31.62 ± 0.50	36.14 ± 1.79
3	$\mathbf{46.59 \pm 1.82}$	28.10 ± 0.67	31.19 ± 0.96	28.07 ± 0.81	31.52 ± 6.12	42.89 ± 0.37	27.49 ± 0.57	42.90 ± 2.33
5	$\mathbf{55.82 \pm 1.11}$	41.88 ± 0.61	42.57 ± 1.15	41.93 ± 0.67	44.53 ± 4.58	52.80 ± 0.25	41.53 ± 0.31	52.55 ± 1.56
				$k=3$				
top n	gr	tf	ig	χ^2	rf	Δ-idf	tf-idf	bm25
1	$\mathbf{30.11 \pm 2.30}$	19.58 ± 0.53	27.11 ± 0.86	29.30 ± 2.17	20.07 ± 0.84	24.14 ± 5.60	27.24 ± 0.82	20.22 ± 1.12
2	$\mathbf{54.37 \pm 0.57}$	39.67 ± 3.66	43.98 ± 0.92	51.66 ± 3.61	36.78 ± 1.18	49.01 ± 3.54	40.30 ± 0.85	44.78 ± 0.92
3	$\mathbf{61.62 \pm 1.65}$	42.30 ± 8.00	43.40 ± 2.23	55.72 ± 4.38	60.03 ± 0.92	54.52 ± 4.56	43.79 ± 0.72	35.18 ± 1.19
5	69.62 ± 1.56	55.61 ± 5.95	$\mathbf{70.04 \pm 3.76}$	64.55 ± 2.81	68.69 ± 0.95	63.84 ± 4.25	56.50 ± 0.55	74.09 ± 1.11
				$k=5$				
top n	gr	tf	ig	χ^2	rf	Δ-idf	tf-idf	bm25
1	$\mathbf{34.76 \pm 0.37}$	19.62 ± 0.60	28.44 ± 1.07	26.78 ± 10.36	19.56 ± 0.61	20.71 ± 1.08	19.63 ± 1.50	19.91 ± 0.96
2	$\mathbf{55.02 \pm 0.50}$	38.02 ± 6.22	54.60 ± 0.61	51.35 ± 5.87	36.30 ± 0.92	41.35 ± 5.90	44.36 ± 1.38	46.79 ± 3.05
3	$\mathbf{63.27 \pm 1.73}$	46.84 ± 6.09	59.28 ± 0.83	51.93 ± 12.93	53.96 ± 7.74	58.33 ± 1.49	58.00 ± 2.47	54.85 ± 7.35
5	$\mathbf{71.48 \pm 1.20}$	58.24 ± 5.45	72.23 ± 2.70	69.06 ± 4.96	67.55 ± 9.84	72.29 ± 4.32	73.75 ± 2.44	64.67 ± 6.72
				$k=10$				
top n	gr	tf	ig	χ^2	rf	Δ-idf	tf-idf	bm25
1	$\mathbf{35.81 \pm 0.58}$	20.01 ± 0.61	35.69 ± 0.86	33.96 ± 2.39	24.86 ± 8.20	28.45 ± 7.50	31.54 ± 3.19	30.01 ± 2.75
2	$\mathbf{56.64 \pm 0.79}$	39.11 ± 5.74	$\mathbf{56.64 \pm 0.65}$	53.25 ± 4.95	39.20 ± 8.33	52.03 ± 0.47	52.54 ± 4.08	47.94 ± 4.73
3	$\mathbf{63.20 \pm 1.40}$	52.66 ± 11.58	62.52 ± 0.76	53.85 ± 5.35	56.86 ± 1.46	58.99 ± 1.30	61.65 ± 1.25	67.62 ± 2.41
5	71.44 ± 0.93	72.70 ± 3.43	68.70 ± 0.92	66.74 ± 5.24	72.41 ± 1.37	$\mathbf{75.63 \pm 0.46}$	77.23 ± 3.17	77.51 ± 1.12

Table 3.11: Classification accuracy (in %) of the k-nearest neighbor algorithms on the ACM data set. The classifier was trained on P-PR vectors of nodes in homogeneous networks, obtained from the original heterogeneous network using eight weighting schemes. All values of the vectors, smaller than 0.00001, were set to 0.

				$k=1$				
top n	gr	tf	ig	χ^2	rf	Δ-idf	tf-idf	bm25
1	15.58 ± 2.57	21.83 ± 3.27	19.39 ± 0.72	$\mathbf{22.93 \pm 5.53}$	13.33 ± 0.82	19.50 ± 0.57	19.33 ± 0.69	18.85 ± 0.61
2	31.99 ± 2.49	36.99 ± 1.16	35.50 ± 1.07	$\mathbf{38.88 \pm 6.05}$	29.72 ± 0.19	33.04 ± 3.40	35.13 ± 0.85	35.12 ± 0.71
3	31.43 ± 1.44	32.66 ± 7.03	28.45 ± 0.57	$\mathbf{34.34 \pm 4.84}$	30.70 ± 1.29	28.54 ± 0.54	28.21 ± 0.84	27.99 ± 0.39
5	42.73 ± 0.54	$\mathbf{45.48 \pm 4.97}$	42.41 ± 0.65	47.91 ± 3.01	41.78 ± 0.64	42.64 ± 0.88	42.32 ± 0.81	42.16 ± 0.46
				$k=3$				
top n	gr	tf	ig	χ^2	rf	Δ-idf	tf-idf	bm25
1	17.65 ± 5.70	$\mathbf{23.43 \pm 4.09}$	22.57 ± 3.53	23.30 ± 5.84	16.22 ± 3.07	21.02 ± 0.61	19.49 ± 0.43	19.74 ± 0.74
2	36.94 ± 10.16	48.44 ± 3.92	40.95 ± 7.55	$\mathbf{50.37 \pm 0.59}$	33.56 ± 2.96	33.96 ± 0.73	36.59 ± 0.97	31.42 ± 0.79
3	40.87 ± 9.50	$\mathbf{54.21 \pm 3.70}$	39.31 ± 10.71	51.61 ± 2.94	42.31 ± 3.92	37.06 ± 0.58	37.62 ± 5.20	34.55 ± 0.56
5	50.94 ± 8.63	$\mathbf{68.14 \pm 5.47}$	51.98 ± 8.58	61.30 ± 2.36	53.66 ± 2.68	51.09 ± 0.89	51.77 ± 5.32	49.37 ± 0.59
				$k=5$				
top n	gr	tf	ig	χ^2	rf	Δ-idf	tf-idf	bm25
1	22.67 ± 8.15	23.43 ± 3.96	24.99 ± 3.92	$\mathbf{33.50 \pm 0.83}$	16.17 ± 2.87	20.61 ± 0.77	19.20 ± 0.63	19.43 ± 0.70
2	46.55 ± 5.45	47.33 ± 3.51	49.58 ± 6.89	$\mathbf{50.80 \pm 1.07}$	31.96 ± 2.26	41.88 ± 5.88	36.08 ± 0.95	31.78 ± 0.81
3	$\mathbf{59.62 \pm 1.87}$	50.47 ± 1.21	50.84 ± 8.57	58.20 ± 2.02	43.39 ± 7.24	53.53 ± 7.58	42.27 ± 1.86	35.85 ± 1.04
5	71.82 ± 4.32	$\mathbf{73.64 \pm 5.31}$	65.36 ± 7.77	66.86 ± 2.84	60.35 ± 5.11	71.75 ± 10.30	58.55 ± 0.81	51.44 ± 0.94
				$k=10$				
top n	gr	tf	ig	χ^2	rf	Δ-idf	tf-idf	bm25
1	29.04 ± 1.26	22.69 ± 3.72	30.44 ± 1.91	28.15 ± 6.69	18.52 ± 0.93	$\mathbf{32.95 \pm 2.38}$	21.35 ± 3.35	22.91 ± 5.61
2	$\mathbf{55.92 \pm 1.40}$	46.08 ± 6.68	55.76 ± 0.96	52.06 ± 0.32	32.01 ± 3.57	50.05 ± 4.70	37.94 ± 3.93	41.01 ± 9.53
3	$\mathbf{66.45 \pm 2.65}$	62.95 ± 5.93	63.84 ± 3.50	64.63 ± 3.99	43.96 ± 8.23	53.59 ± 4.26	49.72 ± 4.89	54.13 ± 10.42
5	$\mathbf{78.68 \pm 2.83}$	77.72 ± 4.02	76.52 ± 1.17	76.14 ± 6.76	61.12 ± 9.89	68.32 ± 2.27	64.37 ± 4.93	67.98 ± 9.34

3.3.5 Experiments on iMDB data set

We repeated the experiments from the ACM data set on the iMDB data set, and also analyzed other semi-supervised learning methods for classification on the data set. In this section, we describe the experiments and results for the iMDB data set.

Table 3.12: Classification accuracy (in %) for the label propagation algorithm on the ACM data set. The label propagation was run in homogeneous networks, obtained from the heterogeneous network using eight weighting schemes.

Modified label propagation								
top n	gr	tf	ig	χ^2	rf	Δ-idf	tf-idf	bm25
1	29.26	29.66	29.61	28.82	29.81	29.80	**30.16**	29.36
2	58.03	57.46	58.14	57.22	58.12	58.27	57.29	**58.35**
3	54.98	55.21	**55.71**	55.20	55.41	55.51	54.72	54.76
5	70.68	71.74	70.64	70.64	70.82	71.34	70.76	**71.80**
Standard label propagation								
top n	gr	tf	ig	χ^2	rf	Δ-idf	tf-idf	bm25
1	29.26	29.66	29.61	28.82	29.81	29.80	**30.16**	29.36
2	58.03	57.46	58.14	57.22	58.12	58.27	57.29	**58.35**
3	54.98	55.21	**55.71**	55.20	55.41	55.51	54.72	54.76
5	70.68	71.74	70.64	70.64	70.82	71.34	70.76	**71.80**

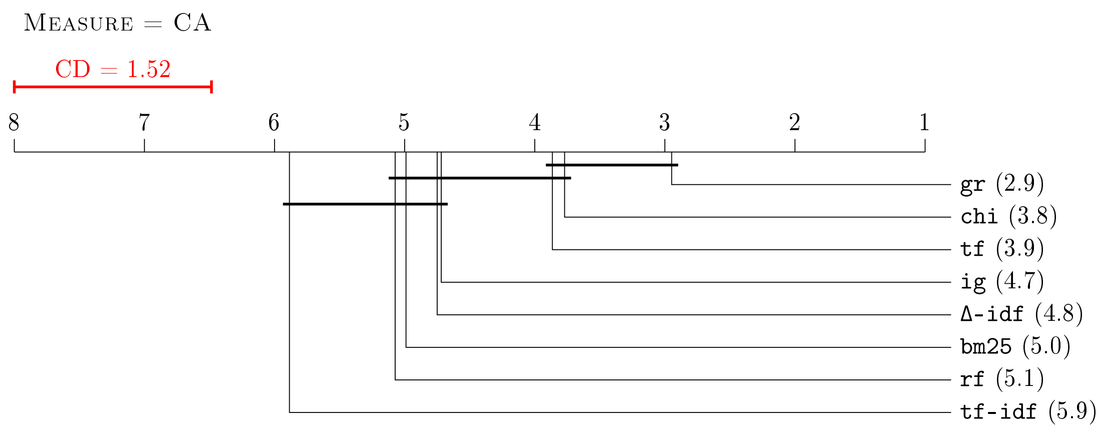


Figure 3.7: Critical distance diagram of the weighting schemes and their performance on the ACM data set. The critical distance is calculated at a p value of 0.05.

3.3.5.1 Experiment description

We used the iMDB data set to compare several alternative semi-supervised learning approaches described in Section 2.1.1. Five different semi-supervised classification algorithms are described in (de Sousa et al., 2013). One of these is label propagation, which we used on all our data sets. We tested two other methods, Gaussian Random Fields (GRF) and Robust Multi-Class Graph Transduction (RMGT) on the iMDB data set. We chose the RMGT and GRF methods over Laplacian Support Vector Machines and Laplacian Regularized Least Squares because of implementation difficulties of the latter two methods which are in MATLAB, while the former two were easy to use in our experiments. We performed the same set of experiments as on the ACM data set. However, as the network size is much smaller, we used 5-fold cross validation using the same multi-label stratification approach as in the ACM data set (Sechidis et al., 2011). As in the ACM data set, each instance is labeled with one or more labels and we again evaluate the top-n predictions. Because of a larger number of class labels (20 genres), we also tested the top-10 classifier.

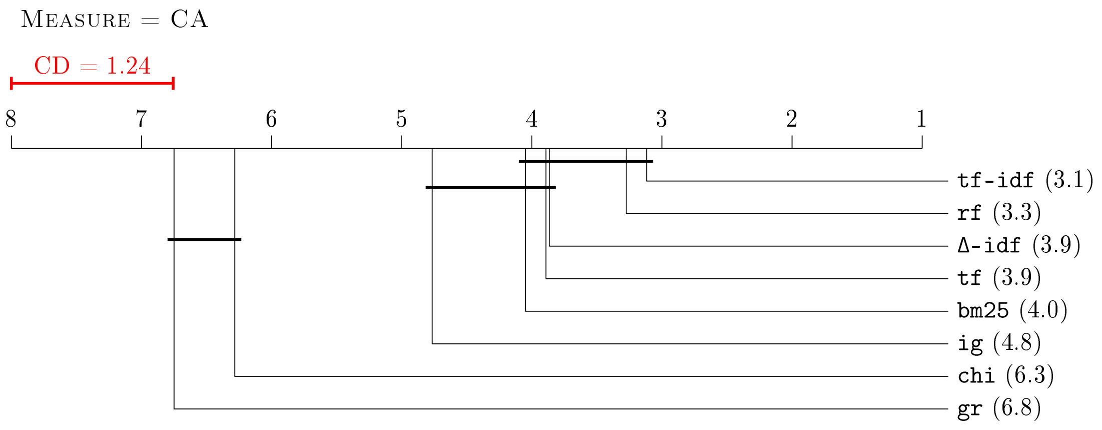


Figure 3.8: Critical distance diagram of the weighting schemes and their performance on the IMDB data set. The critical distance is calculated at a p value of 0.05.

3.3.5.2 Experimental results

Table 3.13 shows the results of different methods for semi-supervised learning on the iMDB data set. We see that on this data set, the RMGT method does not perform as well as the standard label propagation and the GRF methods, which both perform comparably.

Table 3.14 shows the results of the experiments on the iMDB data set. We repeated the same experiments as in the ACM data set. Because of the larger number of possible labels (20 as opposed to 11 in the ACM data set), we also calculated the accuracy of the top-10 classifier. Interestingly, the results show a different picture from the ACM data set. The label propagation with standard weights substantially outperforms the label propagation with imbalance-sensitive weights. There are several factors that can contribute to this result. Compared to the other data sets, the iMDB data set is smaller and has the largest number of different labels. The most frequent label in the data set, "Drama", contains 5,076 representatives, the second most frequent, "Comedy", contains 3,566, while the least frequent label, "IMAX", only contains 25 instances. This represents an extreme imbalance ratio between large and small classes (up to 1:203), and means that the number of rarely represented classes is probably too small for reliable setting of imbalance correction coefficient.

A second explanation for poor result of the imbalance-sensitive label propagation classifier is the imbalance-agnostic performance measure used. We used the same method for calculating accuracy as Grčar et al. (2013) in which an example is correctly classified if one of its labels appears among the top k suggestions by the classifier (with k being set to 1, 2, 3 and 5) and the accuracy is then calculated as the proportion of correctly classified examples. This method does not take class sizes into account and penalizes a misclassified "IMAX" movie just as heavily as a misclassified "Drama" movie. While increasing the accuracy on the smaller classes (even at the cost of miss-classifying larger classes) may be beneficial in some cases, the accuracy measure we used did not take this into account.

Another observation is that the standard label propagation algorithm performs almost as well as the 10-nearest neighbors algorithm, but on average, it is still beneficial to use the computationally more demanding step of P-PR vector calculation.

In Figure 3.8 we see the full comparison of weighting schemes. On average, the `tf-idf` weighting scheme performs best, while the `gr` scheme performs worst. This result stands in contrast to the results on the ACM data set and further reinforces our hypothesis that the best weighting scheme must be selected for each data set individually.

Table 3.13: Semi-supervised classifier accuracies (in %) for the iMDB data set. The classifiers were trained on P-PR vectors of nodes in homogeneous networks, obtained from the original heterogeneous network using eight weighting schemes.

Robust Multi-Class Graph Transduction								
top n	gr	tf	ig	χ^2	rf	Δ-idf	tf-idf	bm25
1	**49.62 ± 1.56**	40.66 ± 0.96	49.43 ± 1.56	48.66 ± 1.52	40.98 ± 0.84	19.51 ± 0.99	37.03 ± 0.71	30.25 ± 1.07
2	**63.03 ± 1.28**	50.24 ± 1.27	61.45 ± 1.33	61.18 ± 1.41	50.74 ± 1.32	23.84 ± 0.81	45.55 ± 1.10	37.64 ± 1.63
3	**68.68 ± 1.11**	55.06 ± 1.62	66.75 ± 0.86	66.79 ± 1.32	55.61 ± 1.62	27.11 ± 0.75	50.27 ± 1.35	41.58 ± 1.64
5	**74.90 ± 0.83**	61.08 ± 1.66	73.05 ± 0.69	73.05 ± 1.09	61.67 ± 1.68	33.41 ± 0.39	56.60 ± 1.84	47.78 ± 1.66
10	**82.09 ± 0.69**	70.77 ± 1.02	80.42 ± 0.84	80.21 ± 0.57	71.22 ± 0.98	47.08 ± 0.92	66.96 ± 0.99	59.78 ± 1.23
Gaussian Random Fields								
top n	gr	tf	ig	χ^2	rf	Δ-idf	tf-idf	bm25
1	61.57 ± 0.80	60.87 ± 0.92	**62.77 ± 0.63**	62.74 ± 0.63	61.00 ± 0.85	61.19 ± 0.58	60.87 ± 0.92	60.80 ± 0.81
2	77.83 ± 1.20	76.46 ± 1.01	**78.02 ± 0.89**	78.71 ± 1.07	76.45 ± 1.00	76.98 ± 0.89	76.46 ± 1.01	76.58 ± 0.98
3	85.04 ± 0.84	84.85 ± 0.88	85.29 ± 0.79	**85.53 ± 0.79**	84.87 ± 0.90	85.08 ± 0.84	84.85 ± 0.88	84.86 ± 0.85
5	91.30 ± 0.70	91.45 ± 0.73	91.60 ± 0.67	91.65 ± 0.62	91.44 ± 0.74	**91.87 ± 0.71**	91.45 ± 0.73	91.53 ± 0.71
10	96.03 ± 0.48	96.19 ± 0.53	96.05 ± 0.52	96.01 ± 0.46	96.17 ± 0.54	**96.43 ± 0.56**	96.19 ± 0.53	96.22 ± 0.58
Modified label propagation								
top n	gr	tf	ig	χ^2	rf	Δ-idf	tf-idf	bm25
1	19.03 ± 1.09	27.03 ± 0.93	23.95 ± 0.73	19.60 ± 0.92	26.99 ± 0.85	**28.37 ± 0.79**	27.61 ± 0.94	25.84 ± 0.59
2	33.66 ± 1.12	43.84 ± 0.88	39.66 ± 0.79	32.99 ± 1.17	43.64 ± 0.94	45.69 ± 0.97	**44.26 ± 0.91**	42.14 ± 0.75
3	46.40 ± 1.60	55.53 ± 1.19	52.09 ± 1.19	44.95 ± 1.24	55.46 ± 1.12	57.06 ± 0.69	**56.04 ± 1.25**	53.57 ± 0.97
5	63.89 ± 1.21	70.80 ± 0.82	68.24 ± 0.91	62.18 ± 1.17	70.68 ± 0.87	**72.56 ± 0.61**	71.20 ± 0.88	69.42 ± 0.97
10	83.86 ± 1.36	90.06 ± 0.93	86.22 ± 0.66	82.11 ± 1.06	89.96 ± 0.96	**90.61 ± 0.84**	90.08 ± 0.82	89.14 ± 1.02
Standard label propagation								
1	53.97 ± 1.08	51.74 ± 1.08	55.67 ± 1.22	**56.11 ± 1.11**	51.81 ± 1.05	52.64 ± 1.27	51.74 ± 1.10	52.26 ± 1.08
2	74.20 ± 0.69	74.15 ± 0.62	**74.29 ± 0.70**	**74.29 ± 0.70**	74.16 ± 0.61	74.17 ± 0.73	74.12 ± 0.69	74.01 ± 0.69
3	**83.74 ± 0.54**	83.23 ± 0.45	83.84 ± 0.52	84.27 ± 0.46	83.27 ± 0.51	83.33 ± 0.81	83.22 ± 0.55	83.22 ± 0.69
5	89.67 ± 0.64	89.06 ± 0.75	89.69 ± 0.60	**90.03 ± 0.55**	89.05 ± 0.74	89.30 ± 0.72	89.08 ± 0.74	88.76 ± 0.70
10	95.98 ± 0.51	96.14 ± 0.57	96.03 ± 0.60	96.09 ± 0.57	96.15 ± 0.58	**96.38 ± 0.47**	96.25 ± 0.57	96.06 ± 0.55

Table 3.14: k-NN classifiers accuracies (in %) for the iMDB data set. The classifiers were using P-PR vectors of nodes in homogeneous networks, obtained from the original heterogeneous network using eight weighting schemes.

$k = 1$								
top n	gr	tf	ig	χ^2	rf	Δ-idf	tf-idf	bm25
1	40.74 ± 1.12	41.84 ± 1.74	41.71 ± 1.70	41.09 ± 1.87	41.87 ± 1.65	**42.47 ± 1.86**	42.33 ± 1.58	41.78 ± 1.74
2	53.87 ± 0.72	54.40 ± 1.58	54.80 ± 1.57	54.16 ± 1.36	54.62 ± 1.59	**54.87 ± 1.44**	54.76 ± 1.43	54.35 ± 1.62
3	57.75 ± 0.61	58.10 ± 1.52	58.66 ± 1.86	58.21 ± 1.43	58.59 ± 1.42	**58.91 ± 1.13**	58.75 ± 1.28	58.02 ± 1.56
5	62.25 ± 0.95	62.62 ± 1.46	**63.04 ± 1.92**	62.77 ± 1.50	63.04 ± 1.47	63.26 ± 1.45	63.03 ± 1.44	62.57 ± 1.51
10	74.27 ± 0.85	74.50 ± 1.00	**74.87 ± 1.36**	74.41 ± 1.34	74.67 ± 1.22	74.99 ± 1.16	74.64 ± 1.21	74.49 ± 1.01
$k = 5$								
top n	gr	tf	ig	χ^2	rf	Δ-idf	tf-idf	bm25
1	54.79 ± 1.01	56.27 ± 1.39	56.32 ± 1.46	54.75 ± 1.68	**56.38 ± 1.28**	55.17 ± 1.48	56.80 ± 1.51	56.28 ± 1.46
2	71.51 ± 0.93	73.36 ± 0.82	72.28 ± 1.10	71.09 ± 1.03	73.57 ± 0.81	72.69 ± 1.78	**73.72 ± 1.44**	73.37 ± 0.88
3	78.10 ± 1.27	79.83 ± 0.43	78.77 ± 0.79	77.82 ± 0.27	80.02 ± 0.17	80.07 ± 1.02	**80.35 ± 0.55**	79.81 ± 0.43
5	86.18 ± 0.85	87.65 ± 0.50	86.91 ± 0.57	85.80 ± 0.79	87.86 ± 0.10	**88.03 ± 0.46**	87.86 ± 0.55	87.67 ± 0.55
10	93.87 ± 0.88	95.02 ± 0.82	94.09 ± 0.72	93.80 ± 0.68	**95.17 ± 0.63**	94.94 ± 0.50	94.90 ± 0.64	95.02 ± 0.82
$k = 10$								
top n	gr	tf	ig	χ^2	rf	Δ-idf	tf-idf	bm25
1	59.46 ± 1.44	61.26 ± 1.06	60.73 ± 1.54	59.25 ± 1.30	61.35 ± 1.45	59.04 ± 1.11	**61.53 ± 0.92**	61.19 ± 0.99
2	75.89 ± 0.62	78.14 ± 0.71	76.02 ± 0.71	75.37 ± 0.67	78.12 ± 0.85	77.34 ± 0.53	**78.30 ± 1.08**	78.10 ± 0.66
3	82.09 ± 0.62	84.43 ± 0.45	82.54 ± 1.12	82.10 ± 1.20	84.44 ± 0.48	84.25 ± 0.44	**84.54 ± 0.49**	84.46 ± 0.44
5	89.46 ± 0.60	91.19 ± 0.41	90.09 ± 0.83	89.75 ± 0.89	91.17 ± 0.42	**91.37 ± 0.43**	91.25 ± 0.45	91.22 ± 0.40
10	96.06 ± 0.47	97.00 ± 0.23	96.24 ± 0.39	96.14 ± 0.70	97.00 ± 0.26	96.81 ± 0.17	96.69 ± 0.16	**97.01 ± 0.25**

3.3.6 Summary of experimental findings

Our experiments show that using different weighting schemes can significantly alter the propositionalization vectors, obtained by following the methodology proposed by Grčar et al. (2013). The selection of the best weighting scheme is data set and classifier specific, as different data sets and different classifiers applied to the propositionalized networks can yield significantly different results. The behavior of the gain ratio and inverse document frequency weighting schemes illustrates this, as the `gr` scheme performs very well on the ACM data set, but does not perform well on the iMDB data set, and the roles are reversed for the `tf-idf` weighting scheme.

Overall, such performance is expected in the light of weighting schemes origins. None of the tested methods is shown to consistently outperform others in the field of text mining, and the selection of the best scheme to use in bag-of-vector construction is still a vital part of a text mining process. Our experiments show that a similar step should be used in the network decomposition methodology.

Furthermore, our experiments show that when the number of classes to predict is small, and the data set is heavily imbalanced, the label propagation algorithm should be modified by changing the initial propagation matrix Y. This produces much better results than the standard initial matrix defined by D. Zhou et al. (2004). The modified matrix, however, yields no improvement when the classes are uniformly distributed in the data set (as in the ACM data set), and can even lead to unreliable classifications when the number of classes to predict is large and each class has a small number of representatives (as in the iMDB data set).

3.4 Conclusions

We have proposed and implemented several improvements to existing network classification algorithms into an improved heterogeneous network mining methodology we call HinMine. We used the HinMine methodology to analyze the effect of starting weights for node classification using label propagation and the effects of different heuristics to asses the importance of intermediary nodes in network decomposition. The contributions of this chapter are as follows. First, by setting the weights of the initial class matrix proportionally to the class value frequency, we improved the performance of the label propagation algorithm when applied to the highly imbalanced data set. Second, we adapted heuristics, developed primarily for use in text mining, for the construction of homogeneous networks from heterogeneous networks. We used 8 different heuristics – some were designed to penalize terms common to all documents in a corpus (`tf-idf`, `bm25`), some penalized terms appearing in documents of all classes (`chi`, `gr` and `ig`) and some combined the two properties (`Δ-idf` and `rf`). Third, we tested the newly developed algorithms on three data sets with different properties. The first data set was of medium size with only two target classes, the second was a smaller data set with 20 classes, and the third was very large with 11 classes.

Our results show that when using label propagation to classify nodes in a network, using modified starting weights can have a large impact on the accuracy of the resulting classifier. We have shown that on an imbalanced data set, the modified weights can decrease the effect of the larger classes and prevent some cases of over-voting. A second significant contribution of this chapter is the analysis of how the choice of heuristics impacts the performance of both label propagation classifier and classifiers based on the propositionalization approach of Grčar et al. (2013). The introduction of the heuristics from text mining into network decomposition is a relatively general one, meaning that in future, possible new weighting heuristics, developed for bag-of-word vector construction can quickly be adapted into the network decomposition step of HinMine. We have also shown that the choice of the correct heuristic cannot be made in advance, as it depends on the structure of the network. This is evident comparing results on the ACM and iMDB data sets: the `gr` weighting scheme performs well on the ACM data set and poorly on the iMDB data set, while the situation is reversed for the `tf-idf` weighting.

Chapter 4

NetSDM: Network Filtering for Semantic Data Mining

Semantic data mining (SDM) uses annotated data and complex interconnected background knowledge to generate rules that are easily interpreted by the end user. However, computational complexity of SDM algorithms is high, resulting in long running times even when applied to relatively small data sets. On the other hand, network analysis algorithms are among the most scalable data mining algorithms. In this section, we propose and test an effective SDM approach that combines semantic data mining and network analysis.

4.1 Combining Semantic Pattern Mining with Network Analysis

While the Hedwig algorithm, described in Section 2.2.4 performs well on small real-world data sets, the algorithm searches through a very large space of possible patterns to find the 'best' if-then rule explaining the data. The more conjuncts we allow in rules, the larger the search space. This section presents the proposed methodology that combines semantic data mining with network analysis to reduce the search space of semantic data mining algorithms.

4.1.1 Proposed methodology NetSDM for reducing the search of SDM

The Hedwig algorithm uses a beam search, consequently only searching the locally most promising parts of the search space. However, this strategy is heuristic and can lead to poor results in some cases. For example, if the search beam is too narrow, the branch that leads to good solutions could be discarded early on. In large search spaces even wide beams can be quickly filled with terms that will eventually not lead to a good solution. The proposed methodology, named NetSDM, is outlined in Figure 4.1.

The goal of the proposed methodology is to improve the efficiency of the Hedwig algorithm by using a pre-pruning step in which we filter out background knowledge terms that are not likely to appear in significant rules. A scoring function used in pruning should (i) be able to evaluate the significance of terms based on data, and (ii) be efficiently computed. The scoring function receives as input all the data that is later used in the Hedwig algorithm: data set S consisting of target (S_+) and non-target (S_-) instances, background knowledge ontology G (consisting of a set of terms V and a set of relations E), and a set of annotations $\mathcal{A}$ connecting instances in $S = S_+ \cup S_-$ with nodes in G. Just like the Hedwig algorithm, the NetSDM can accept several ontologies as input — in that case, the input ontologies are joined together by adding a dummy root node connected to all the

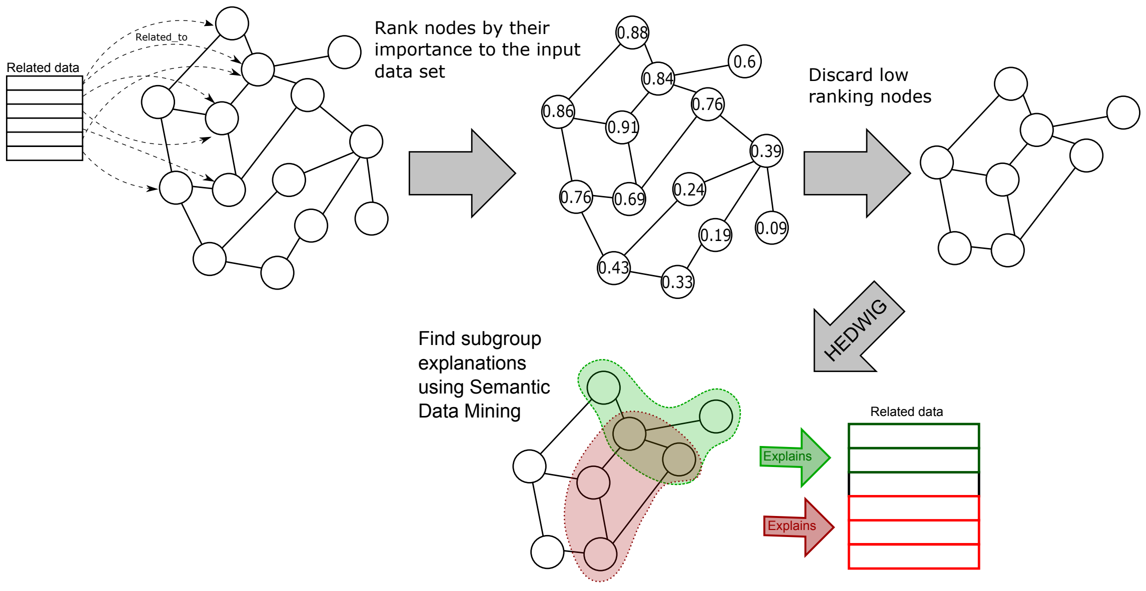

Related data
Related_to
Rank nodes by their importance to the input data set
0.88
0.6
0.84
0.76
0.86
0.91
0.39
0.24
0.76
0.69
0.19
0.09
0.43
0.33
Discard low ranking nodes
HEDWIG
Find subgroup explanations using Semantic Data Mining
Related data
Explains
Explains

Figure 4.1: An illustrative example, outlining of the proposed NetSDM methodology.

Algorithm 4.1: The proposed NetSDM algorithm, combining semantic data mining with network ranking.

```
Input: Background knowledge $G$ and a set of examples $S$ annotated with nodes
       from $G$
Output: A set of rules discovered by semantic data mining algorithm.
Parameters: threshold value $c \in [0, 1]$
Calculate importance scores score($g$) for all $g \in G$
for node $g \in G$ do
    if $|\{g' \in G : \text{score}(g') > \text{score}(g)\}| > c \cdot |G|$ then
        mark node $g$ for removal.
    end
end
form $G'$ by removing all marked nodes from $G$
Run semantic data mining on $S$ using the pruned $G'$ as background knowledge
return Rules discovered by the SDM algorithm
```

root nodes of the input ontologies. Annotations are presented as pairs (s, g), where $s \in S$ and $g \in G$, which denote that data instance s is annotated by background knowledge term g. The output of the scoring function is a vector which, for each term in the background knowledge, calculates a score for the term–in other words, the scoring function is a function

$$\text{score}_{S,G,\mathcal{A}} : V \to [0, \infty).$$

For our application, we are interested in score functions for which a higher score of term $t \in V$ means that term t is more likely to be used in rules describing the original data. After defining a suitable scoring function *score* (in the subsequent subsections we will examine two possibilities), we use the computed scores to prune the background knowledge G and get a smaller background knowledge G' using a selected threshold value $c \in [0, 1]$. We construct G' from G by keeping only the proportion c of nodes with the highest scores, i.e.

we keep only terms t' for which it holds that

$$\frac{|\{t' \in V : \text{score}(t') > \text{score}(t)\}|}{|V|} \leq c.$$

For example, if $c = 0.4$, the new network G' consists only of terms in G which are in the top 40 percent of all nodes according to the scoring function.

The proposed NetSDM methodology, illustrated in Figure 4.1, is presented in Algorithm 4.1. It consists of three steps that are explained in more detail in Section 4.1.3:

1. estimate the importance of background knowledge using the scoring function *score*,
2. prune the background knowledge, keeping only the proportion c of top ranking terms,
3. use semantic data mining algorithm Hedwig on the original data set and pruned background knowledge.

4.1.2 Illustrative example

As a toy example we use the data set described in Table 4.1. The data set consists of 15 instances, 7 of which belong to the target class (marked with + in Table 4.1). The instances are annotated by one or more nodes in a hierarchy (simple ontology) consisting of 8 base nodes and 7 higher-level nodes, shown on the left-hand side of Figure 4.2. Using the Personalized PageRank scoring function described in Section 4.1.3, we assign high scores to those background knowledge nodes that are strongly related to the positive examples. Resulting scores are depicted in the middle of Figure 4.2. We can see that the nodes on the left-hand side have higher scores compared to the right-hand nodes as they annotate mostly positive (target) examples. In the next step, we prune the lower-ranked nodes, leaving only the best 8 background nodes in the new hierarchy. As most of the top scoring nodes were in the left part of the hierarchy, the pruned hierarchy mostly contains these nodes. In the final step, we use Hedwig to find rules based on instances from Table 4.1 and the reduced ontology. The best two rules discovered by Hedwig are $\boxed{\texttt{Positive(X)} \leftarrow \texttt{LL(X)}}$ (shown in yellow in Figure 4.2) covering 3 positive and 1 negative instance (coverage = 4, precision = $\frac{3}{4}$), and $\boxed{\texttt{Positive(X)} \leftarrow \texttt{LR(X)}}$ (shown in red in Figure 4.2) covering 5 positive and 2 negative instance (coverage = 7, precision = $\frac{5}{7}$). These two rules indeed cover most of the positive examples.

Table 4.1: A toy 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

We could run Hedwig on the original data set for this toy data set and get the same rules as on a pruned data set. However (as shown by our experiments depicted in Section 4.2.3), in actual real-world data sets, removing unimportant parts of large ontologies benefits both the execution time and the quality of the discovered rules.

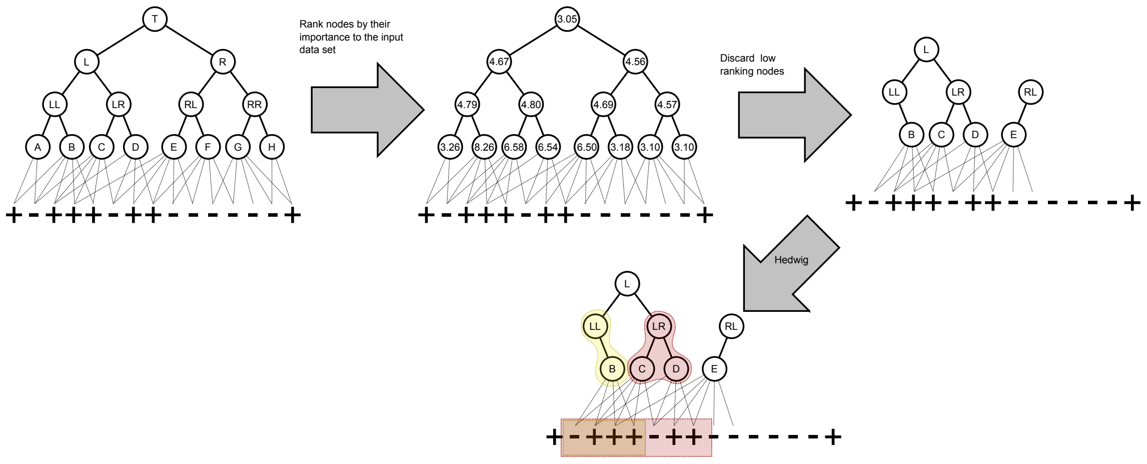

Rank nodes by their importance to the input data set
Discard low ranking nodes
Hedwig
+-+++-++------+

Figure 4.2: An illustration of Algorithm 4.1 on toy example from Table 4.1.

4.1.3 Using network analysis to evaluate node significance

In this section, we describe in detail three vital steps of Algorithm 4.1. We first describe how it is possible to convert a background knowledge into an information network which we can then analyze using network analysis methods. Then, we describe two methods that can be used to estimate the importance of nodes in an information network. The section concludes with a discussion on the node deletion step of the algorithm in which low-scoring nodes are removed from the network.

4.1.3.1 Conversion to a network

Input ontologies for the NetSDM algorithm are collections of terms (background knowledge terms) and relations between them. As an information network is also composed of a set of nodes and the connections between the nodes, a natural conversion from a background knowledge into an information network arises in which we convert each relation connecting two ontology terms into an edge between two corresponding nodes in an information network. This gives rise to the first function converting an input ontology into an information network, in which we view both as parts of one network and score the background terms by their proximity to the data instances in the expanded network. Therefore, we merge the data set and the background knowledge into one expanded network network $G_e = (V_e, E_e)$ where

- $V_e = V \cup S_+$ are vertices of the new network consisting of all background knowledge terms V and all target data instances S_+;
- $E_e = E \cup \mathcal{A}$ are edges of the new network consisting of all background knowledge relations E, as well as all relations from $\mathcal{A}$ of the form s is-annotated-by t (the terms V, S, S_+, S_- and $\mathcal{A}$, used in this definition, are defined in Section 4.1.1).

An alternative approach to converting an ontology into an information network was proposed in (H. Liu, Dou, Jin, LePendu, & Shah, 2013). In their work, H. Liu et al. (2013) use the view that every relation in a semantic representation of data is in fact a triple, consisting of subject, predicate and object. Authors propose to construct a network (which they call a *hypergraph*) in which every relation of the original background knowledge (along with the background knowledge terms) forms an additional node in the network with exactly three connections: one connection to the subject of the relation, one to the object of the relation, and one to the predicate of the relation. Thus, in this alternative information network conversion, each predicate that appears in the background knowledge is an additional node in the resulting network. While this results in a slightly larger information network, the benefit of losing less information in the process of conversion may mean that this network conversion method is better than the naïve version presented in the beginning of this section.

4.1.3.2 Calculating node significance scores from the converted network

We use the Personalized PageRank and node2vec functions to evaluate the significance of background terms. Our goal is to construct the node importance scoring function $\text{score}_{S,G,\mathcal{A}}$ by taking into account both the data set S and the background knowledge terms V.

The first scoring function we used to evaluate the significance of background knowledge terms is the score computed by the Personalized PageRank algorithm (L. Page et al., 1999). The basic PageRank algorithm is used to evaluate the global importance of each node in a network, while the Personalized PageRank (described in Section 2.1.2) evaluates the significance of a given node with respect to the starting set S. This fits well with our demand that a scoring function must evaluate the significance of a term based on the actual data; as each term $g \in G$ is an element of the network G_e, we calculate the P-PR score of each term $g \in G$ as

$$\text{score}_{\text{P-PR}}(g) = \text{P-PR}_{S_+}(g), \tag{4.1}$$

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 one node from S:

$$\text{P-PR}_{S_+}(g) = \frac{1}{|S_+|} \sum_{w \in S_+} \text{P-PR}_{\{w\}}(g). \tag{4.2}$$

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 set S_+) and follows the relations (either the is-annotated-by or the is-a) in the background knowledge. Another interpretation of $\text{score}_{\text{P-PR}}(v)$ is that it tells us how often we will reach node v in random walks, starting with positive (target) 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 the 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 one from v to u, allowing the random walker to traverse the original undirected edge in both directions.

The second estimator of background knowledge significance is the node2vec algorithm (Grover & Leskovec, 2016). In our experiments, we used the default settings of $p = q = 1$ for the parameters of the node2vec function, meaning the random walks generated were a balance between depth-first and breadth-first searches of the network. The maximum length of the random walks was set to 15. The function node2vec, described in Section 2.1.3, calculates a 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 is 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 data instances using a formula inspired by Equation 4.2. With the Personalized PageRank, we evaluate the score of each node as P-PR$_{S_+}(v)$ where S_+ is the set of all target data instances. As the value P-PR$_{\{w\}}(v)$ measures the similarity between w and v, Equation (4.2) can also be used to construct a node2vec scoring function. We replace the individual P-PR similarities in Equation (4.2) with the individual node2vec similarities:

$$\text{score}_{\text{node2vec}}(v) = \frac{1}{|S|} \sum_{w \in S} \frac{f(w) \cdot f(v)}{|f(w)||f(v)|}, \tag{4.3}$$

where feature vectors $f(u)$ are calculated on the network G_e.

4.1.3.3 Node removal step

In the third step of Algorithm 4.1 we remove low-scoring nodes from the network. We present and test two options of 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, it may, at low cutoff thresholds, cause the resulting network to decompose into several disjoint connected components. Also, if we converted the background knowledge into a hypergraph, the hypergraph must always 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 a background knowledge. Thus, blindly removing nodes from the hypergraph will result in a network we can no longer convert back into a background knowledge. To counteract this, we also tested an approach that takes into account the fact that relations, encoded by the network edges, are often transitive – in our example, the gene ontology encodes the is-a relation which is indeed transitive, meaning that if g_1 is-a g_2 and g_2 is-a g_3, then we also know that g_1 is-a g_3.

Using transitivity, we can design an algorithm for advanced removal of low scoring nodes from an information network, obtained by naïve conversion of the background knowledge (Algorithm 4.2). As the background knowledge is in direct correspondence with the information network, obtained from it, we can also view removing the node from the information network as removing a corresponding term from the background knowledge, and therefore Algorithm 4.2 can be used as the term-removal step of Algorithm 4.1.Algorithm 4.2 can also be used to remove low scoring nodes from hypergraphs, constructed from background knowledge, if we first convert hypergraphs into a simpler representation without nodes, representing relations.

4.2 Experimental Setting and Results

In this section, we present the experiments with semantic data mining using both the Personalized PageRank and node2vec search space reduction mechanisms. We first explain the data sets, followed by the experimental setting outline and presentation of the results.

4.2.1 Data sets

In the experiments we used two data sets: the acute lymphoblastic leukemia data set and the breast cancer data set.

Algorithm 4.2: The algorithm for removing a node from a network, obtained through naïve conversion of the background knowledge into an information network.

```
Input: An information network $G$ (obtained by naïvely converting a
       background knowledge into an information network) and a node $n \in G$
       (a term of the original background knowledge) we wish to remove
Output: A new background knowledge that does not contain the node $n$
for $b \in G : (b, n)$ is an edge in $G$ do
    for $a \in G : (n, a)$ is an edge in $G$ do
        Add the edge $(b, a)$ into $G$
        Remove the edge $(n, a)$ from $G$
    end
    Remove the edge $(b, n)$ from $G$
end
Remove the node $n$ from $G$
Return $G$
```

ALL data set We analyzed the acute lymphoblastic leukemia (ALL) data set using semantic data mining. We followed the steps used in the work of 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. The genes were 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, Sherman, & Lempicki, 2008). In this work, we use the same data set to assess if we can improve the performance of the Hedwig algorithm which was already shown to perform well in a biological setting. Our research question is if network node ranking can decrease the running time and improve the performance of the Hedwig algorithm.

Breast cancer data set We also analyzed the breast cancer data set (Vavpetič et al., 2014). In previous work, the authors first used a subgroup discovery algorithm and analyzed the results. After this step, they used the approach to explain the most important subgroup (as determined by domain experts). In this work, we only constructed rules for the same subgroup as in (Vavpetič et al., 2014) and did not use the input of the domain expert. The group contained 11,029 negatively labeled and 990 positively labeled data instances. The data instances were connected to the Gene Ontology terms by 195,124 annotations.

4.2.2 Experimental setting

For each data set and each network scoring function, the experiment consisted of three experimental setups. In both setups, Hedwig was used in its standard mode to return the set of rules explaining the data. The set consists of the last rules that are left on the search beam after the search is concluded (meaning that the size of the set matches the size of the search beam). In the first setup, the entire set of rules was analyzed, and in the second setup only the best rule was analyzed, which allows clearer comparison and is sufficient to demonstrate how pruning the background knowledge affects the quality of the discovered rules. We compared the quality of the resulting rules by measuring the lift values of each rule.

First experimental setup In this set of experiments we ran the Hedwig algorithm on the entire data set to determine the baseline performance of the algorithm and examined all the returned rules. We ran the algorithm with all combinations of depth (1 or 10), beam width (1 or 10) and support (0.1 or 0.01). The goal of this set of experiments was to determine whether terms used by Hedwig are correlated with the score of the terms returned by network analysis algorithms, as calculated by P-PR (Equation 4.1) and node2vec (Equation 4.3). We evaluated this relation by observing the ranks of the terms used by Hedwig to construct the rules. For each term t used by Hedwig we computed the percentage of all terms t' that scored higher by a given scoring function.

$$100 \cdot \frac{|\{t' \in V : \text{score}(t') > \text{score}(t)\}|}{|V|}$$

If the relation between Hedwig's use of terms and returned scores is strong, then the percentage calculated for a given term t will be low.

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 Algorithm 4.1, setting the threshold values of c to $0.5, 0.2, 0.1$ and 0.05, thereby running the Hedwig algorithm on the gene ontology (GO) background knowledge containing only a subset of $5\%, 10\%, 20\%$ and 50% of nodes with the highest score, respectively. We calculated Personalized PageRank values of GO nodes 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. We set beam size, depth and minimum coverage parameters of Hedwig to 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 ten. In the case of rule depth, this value ensures that Hedwig returns longer conjuncts (as on our data sets Hedwig returns no rules of length greater than five, setting the rule depth to ten effectively means that we allow Hedwig to construct arbitrarily long rules). Setting the beam size to ten allows us to discover important rules (as shown in the first set of experiments) in a reasonable amount of time—the runtime of Hedwig increases drastically with increased beam size, and at size ten the algorithm takes several hours to complete).

Third experimental setup In the first two sets of experiments we always used both the naïve 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 tested the advanced versions of both the conversion step (converting the a into a hypergraph) and the node deletion step (using transitivity to maintain connectedness in a network). As the node2vec function proved inferior to the Personalized PageRank scoring function in the second set of experiments, we only ran the third round of experiments with the P-PR scoring function. We ran the algorithm with advanced node deletion on both the information network, obtained through naïve conversion of the background knowledge, and the hypergraph, obtained from the same background knowledge. The results of this third set of experiments can then be compared to the results of the second set.

Table 4.2: The best rules discovered by the Hedwig algorithm in the ALL data set. Each row presents the conjuncts (Gene Ontology terms) of the top ranking rule. The second (third) column shows the percentage of GO terms with the Personalized PageRank (node2vec) score higher than the corresponding term.

Term	Personalized PageRank %	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

Table 4.3: The best rules discovered by the Hedwig algorithm in the breast cancer data set. Each row presents the conjuncts (Gene Ontology terms) of the top ranking rule. The second (third) column shows the percentage of GO terms with the Personalized PageRank (node2vec) score higher than the corresponding term.

Term	Personalized PageRank %	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	10	10	0.01	3.743
GO:0044427	0.409	20.538				
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	10	10	0.1	1.556
GO:0044428	0.051	13.580				
GO:0003674	0.202	36.568				

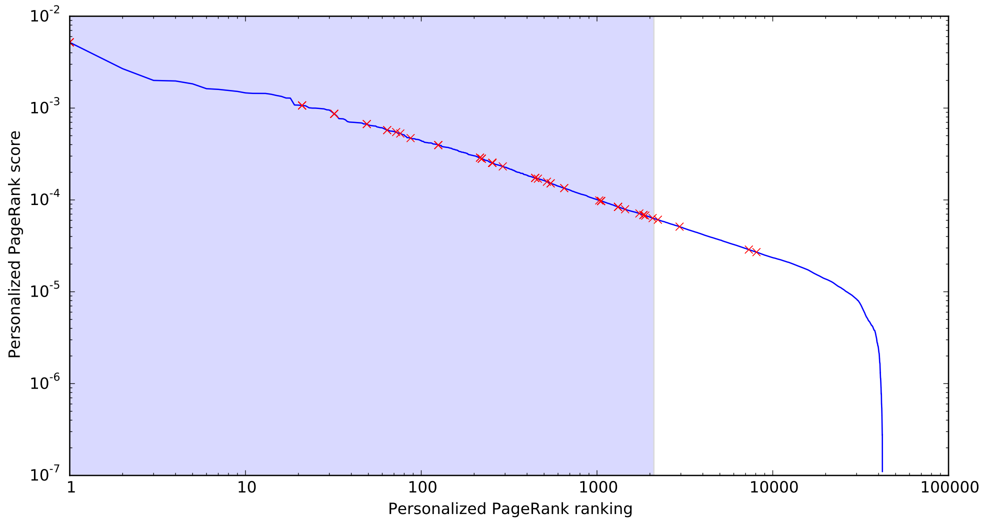
4.2.3 Experimental results

We present the results of semantic data mining algorithm (using two scoring functions, two network conversion methods and two node deletion methods) on the two data sets. We analyze the results of the three experimental setups.

4.2.3.1 First setup: Relation between terms used in Hedwig and term scores

The best rules discovered by Hedwig using different parameter settings in the first experimental setup using the Personalized PageRank function to evaluate node relevance are shown in Tables 4.2 and 4.3 for the ALL and breast cancer data sets, respectively. The results show that the Personalized PageRank scoring function ranks all the terms, used by Hedwig very high with a remarkably low percentage of terms scoring higher. In all cases, the terms used to construct rules were in top percentiles of all rule scores, confirming our hypothesis that high scoring terms are usually used in rules.

This phenomenon is shown even more clearly in Figure 4.3 showing all the terms used by Hedwig and their ranks. Even taking into account all (not only the best) rules, we see



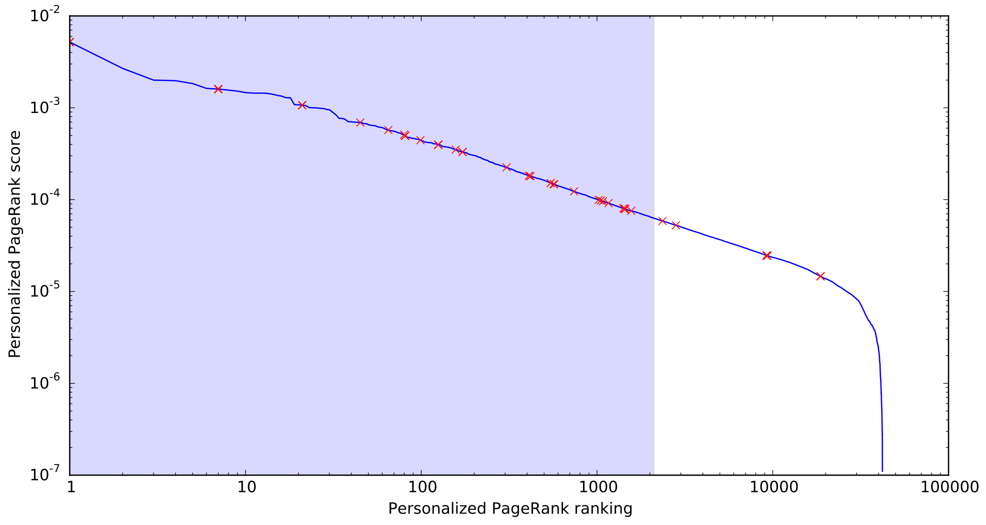
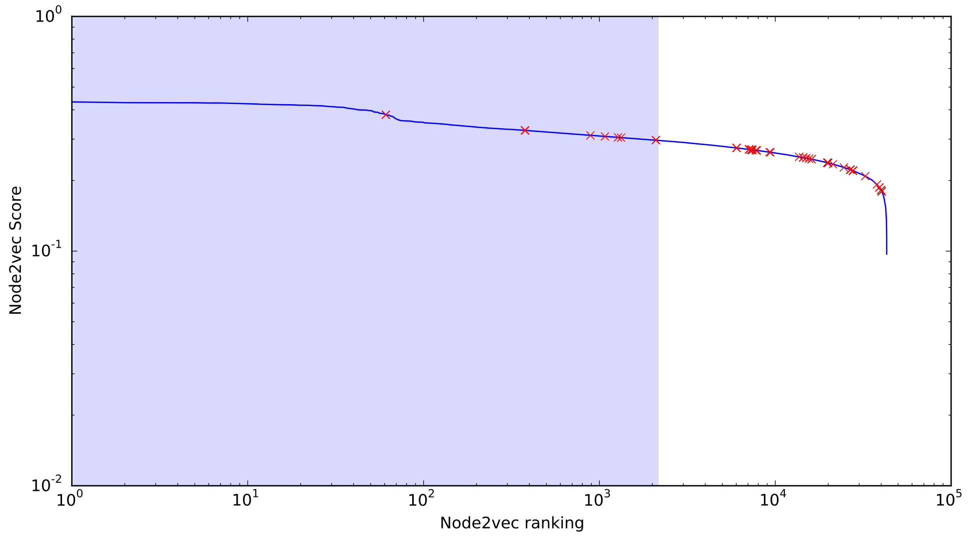


Figure 4.3: The score ranks of the terms used by Hedwig to construct rules on the ALL (top) and breast cancer (bottom) data set. The blue line shows the Personalized PageRank score (the value of P-PR(v)) for each of the terms in the background knowledge. The terms are sorted by descending score. Each red cross highlights one of the terms used by Hedwig. The blue area on the left denotes the top 5 percent of all nodes (according to the ranking).

that predominantly the used terms come from the top 5 percent of all 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 Tables 4.2 and 4.3 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 Figure 4.4, the node2vec score of the used terms is still above average, and it is possible that the node2vec score ranks 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 this experimental setup are also biologically relevant. When the search beam for the Hedwig algorithm was set to 1, the only significant rule discovered contained in its condition a single gene ontology term GO:0003674, a term denoting



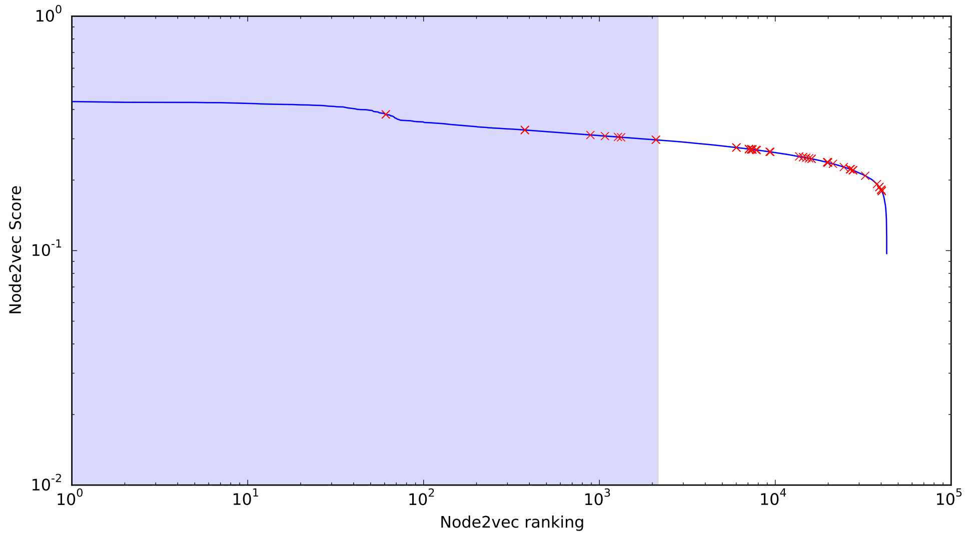


Figure 4.4: The score ranks of the terms used by Hedwig to construct rules on the ALL (top) and breast cancer (bottom) data set. The blue line shows the node2vec score for each of the terms in the background knowledge. The terms are sorted by descending score. Each red cross highlights one of the terms used by Hedwig. The blue area on the left denotes the top 5 percent of all nodes (according to the ranking).

molecular function. This is a very general term which 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).

Table 4.4: Results of Algorithm 4.1 on the ALL data set using the Personalized PageRank scoring function. A smaller cutoff value means that a smaller part of the background knowledge was kept and used by the Hedwig rule discovery algorithm. The direction of the relations in the background knowledge was not taken into account.

Cutoff threshold	GO-term		Lift	Coverage	Positives
0.05	GO:0003824	catalytic activity	3.741	132	55
	GO:0044283	small molecule biosynthetic process			
	GO:0044444	cytoplasmic part			
	GO:0044238	primary metabolic process			
0.1	GO:0003824	catalytic activity	3.769	131	55
	GO:0044283	small molecule biosynthetic process			
	GO:0044444	cytoplasmic part			
	GO:0044238	primary metabolic process			
0.2	GO:0045936	negative regulation of phosphate metabolic process	2.219	89	22
	GO:0003824	catalytic activity			
	GO:0009892	negative regulation of metabolic process			
0.5	GO:0002376	immune system process	3.420	105	40
	GO:0002429	immune response-activating cell surface receptor signaling pathway			
	GO:0005886	plasma membrane			
	GO:0005488	binding			
1	GO:0002376	immune system process	3.420	105	40
	GO:0002429	immune response-activating cell surface receptor signaling pathway			
	GO:0005886	plasma membrane			
	GO:0005488	binding			

Table 4.5: Results of Algorithm 4.1 on the ALL data set using the Personalized PageRank scoring function. A smaller cutoff value means that a smaller part of the background knowledge was kept and used by the Hedwig rule discovery algorithm. The direction of the relations in the background knowledge was taken into account.

Cutoff threshold	GO-term		Lift	Coverage	Positives
0.05	GO:0002376	immune system process	3.235	111	40
	GO:0002694	regulation of leukocyte activation			
	GO:0034110	regulation of homotypic cell-cell adhesion			
0.1	GO:0002376	immune system process	4.090	131	55
	GO:0002694	regulation of leukocyte activation			
	GO:0044459	plasma membrane part			
0.2	GO:0003824	catalytic activity	4.257	116	57
	GO:0044283	small molecule biosynthetic process			
	GO:0044444	cytoplasmic part			
0.5	GO:0002376	immune system process	3.420	105	40
	GO:0002429	immune response-activating cell surface receptor signaling pathway			
	GO:0005886	plasma membrane			
	GO:0005488	binding			
1	GO:0002376	immune system process	3.420	105	40
	GO:0002429	immune response-activating cell surface receptor signaling pathway			
	GO:0005886	plasma membrane			
	GO:0005488	binding			

4.2.3.2 Second setup: Pruning the background knowledge

The results of the first experimental setup show that scores, calculated using the network analysis techniques, are relevant to determine whether background knowledge nodes will be used by Hedwig in the construction of rules describing the target instances. Here we present the results of using the computed scores to pre-prune 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 parameter of Hedwig to 10, and set the minimum coverage parameter 0.01; these are the settings that—when running Hedwig without network analysis—actually produced the best results for both data sets (see Tables 4.2 and 4.3).

For Personalized PageRank-based network pruning we test also if taking into account edge directions makes a difference. Table 4.4 (undirected edges) and Table 4.5 (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 cutoff threshold 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, decreasing the size of the network by 50% still allows us to discover the same high quality conjunct of GO:0002376, GO:0002429, GO:0005886 and GO:0005488 as before, however further decreasing the size of the ontology produces a conjunct GO:0003824, GO:0044283, GO:0044444, GO:0044238 which is more general and hence less interesting.

Comparing Table 4.4 and Table 4.5 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 directions into account, for example, leads to a discovery of a very low quality rule with a lift value of only 2.219 at the cutoff threshold of 0.2. 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 4.1) to choose the correct specialization of the parent node. More experiments are needed to analyze this phenomenon.

Table 4.6 and Table 4.7 show the results of the experiments using the node2vec function on the same data set. The results show that the node2vec pruning did not achieve the same performance as the pruning using the Personalized PageRank function. The results in both tables show that lower threshold values strictly decrease the quality of the rules discovered by Hedwig, meaning that pruning 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 threshold value for pruning background knowledge decreases, the resulting rules become shorter, with the lowest values 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 4.8 and 4.9 show the results of the Personalized PageRank scoring function in pruning the background knowledge for the breast cancer data set. As in the ALL data set, we see that decreasing the threshold value does not drastically decrease the 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

Table 4.6: Results of Algorithm 4.1 on the ALL data set using the node2vec scoring function. A smaller cutoff value means that a smaller part of the background knowledge was kept and used by the Hedwig rule discovery algorithm. The direction of the relations in the background knowledge was not taken into account.

Cutoff threshold	GO-term		Lift	Coverage	Positives
0.05	GO:0050852	T cell receptor signaling pathway	2.586	125	36
0.1	GO:0050852	T cell receptor signaling pathway	2.586	125	36
0.2	GO:0050852	T cell receptor signaling pathway	2.586	125	36
0.5	GO:0050863	regulation of T cell activation	3.076	108	37
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

Table 4.7: Results of Algorithm 4.1 on the ALL data set using the node2vec scoring function. A smaller cutoff value means that a smaller part of the background knowledge was kept and used by the Hedwig rule discovery algorithm. The direction of the relations in the background knowledge was taken into account.

Cutoff threshold	GO-term		Lift	Coverage	Positives
0.05	GO:0050852	T cell receptor signaling pathway	2.586	125	36
0.1	GO:0050852	T cell receptor signaling pathway	2.586	125	36
0.2	GO:0050852	T cell receptor signaling pathway	2.586	125	36
0.5	GO:0007507 GO:0032502	heart development developmental process	2.214	93	22
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

entire data set. Further pruning 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 low threshold values, Algorithm 4.1 finds rules with substantially lower lift values using scores calculated on directed edges compared to those calculated by ignoring the edge direction.

The results of Algorithm 4.1 using the node2vec scoring funciton on the breast cancer data set 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 threshold is lowered, 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 low threshold values the search space of Hedwig (and thus the computational complexity of the algorithm) is much smaller.

Figure 4.5 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 pruning threshold c, i.e. the size of the background knowledge left for Hedwig to analyze. The network preprocessing steps were completed in seconds, meaning that the times, reported in Figure 4.5, correspond almost entirely to the times that the

Table 4.8: Results of Algorithm 4.1 on the breast cancer data set using the Personalized PageRank scoring function. The direction of the relations in the background knowledge was not taken into account.

Cutoff threshold	GO-term		Lift	Coverage	Positives
0.05	GO:0071840	cellular component organization or biogenesis	4.114	121	41
	GO:0000278	mitotic cell cycle			
	GO:0005515	protein binding			
	GO:0044260	cellular macromolecule metabolic process			
	GO:0022402	cell cycle process			
0.1	GO:0043232	intracellular non-membrane-bounded organelle	3.055	151	38
	GO:0000278	mitotic cell cycle			
	GO:0005515	protein binding			
0.2	GO:0071840	cellular component organization or biogenesis	3.781	122	38
	GO:0007049	cell cycle			
	GO:0003824	catalytic activity			
	GO:0005515	protein binding			
0.5	GO:0003824	catalytic activity	3.469	126	36
	GO:0000278	mitotic cell cycle			
	GO:0022402	cell cycle process			
	GO:0005515	protein binding			
1	GO:0003674	molecular function	3.743	120	37
	GO:0044427	chromosomal part			
	GO:0000278	mitotic cell cycle			
	GO:0022402	cell cycle process			

Table 4.9: Results of Algorithm 4.1 on the breast cancer data set using the Personalized PageRank scoring function. The direction of the relations in the background knowledge was taken into account.

Cutoff threshold	GO-term		Lift	Coverage	Positives
0.05	GO:0003824	catalytic activity	3.035	136	34
	GO:0007049	cell cycle			
	GO:0044446	intracellular organelle part			
0.1	GO:0044421	extracellular region part	2.587	122	26
	GO:0003824	catalytic activity			
	GO:0070062	extracellular vesicular exosome			
	GO:0044707	single-multicellular organism process			
0.2	GO:0003674	molecular function	3.743	120	37
	GO:0044427	chromosomal part			
	GO:0000278	mitotic cell cycle			
	GO:0022402	cell cycle process			
0.5	GO:0003674	molecular function	3.743	120	37
	GO:0044427	chromosomal part			
	GO:0000278	mitotic cell cycle			
	GO:0022402	cell cycle process			
1	GO:0003674	molecular function	3.743	120	37
	GO:0044427	chromosomal part			
	GO:0000278	mitotic cell cycle			
	GO:0022402	cell cycle process			

Hedwig algorithm took to complete. The fact that pruning takes orders of magnitude less time than rule discovery is a pre-requisite for our pruning algorithms to be useful in real-life applications. We timed the algorithm on the ALL data set using different settings for beam, depth and support on a 8 core 2.60 GHz Intel Xeon(R)E5-2697 v3 machine with 64GB of RAM. The graph clearly shows that smaller threshold values result in much shorter running times (note the logarithmic scale). This confirms our hypothesis that

Table 4.10: Results of Algorithm 4.1 on the breast cancer data set using the node2vec scoring function. The direction of the relations in the background knowledge was not taken into account.

Cutoff threshold	GO-term		Lift	Coverage	Positives
0.05	GO:0000082	G1/S transition of mitotic cell cycle	2.589	136	29
0.1	GO:0000082	G1/S transition of mitotic cell cycle	2.589	136	29
0.2	GO:0000082	G1/S transition of mitotic cell cycle	2.589	136	29
0.5	GO:1903047 GO:0005515	mitotic cell cycle process protein binding	2.916	204	49
1	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37

Table 4.11: Results of Algorithm 4.1 on the breast cancer data set using the node2vec scoring function. The direction of the relations in the background knowledge was taken into account.

Cutoff threshold	GO-term		Lift	Coverage	Positives
0.05	GO:0000082	G1/S transition of mitotic cell cy- cle	2.589	136	29
0.1	GO:0044699 GO:0000278	single-organism process mitotic cell cycle	3.541	120	35
0.2	GO:0000082	G1/S transition of mitotic cell cy- cle	2.589	136	29
0.5	GO:0000082	G1/S transition of mitotic cell cy- cle	2.589	136	29
1	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37

pruning the background knowledge decreases the time it takes Hedwig to search the space of possible hypotheses. Taking into account also previous results we can conclude that using our background knowledge pre-pruning approach is beneficial both in increasing the quality of the rules and in speeding up the Hedwig algorithm.

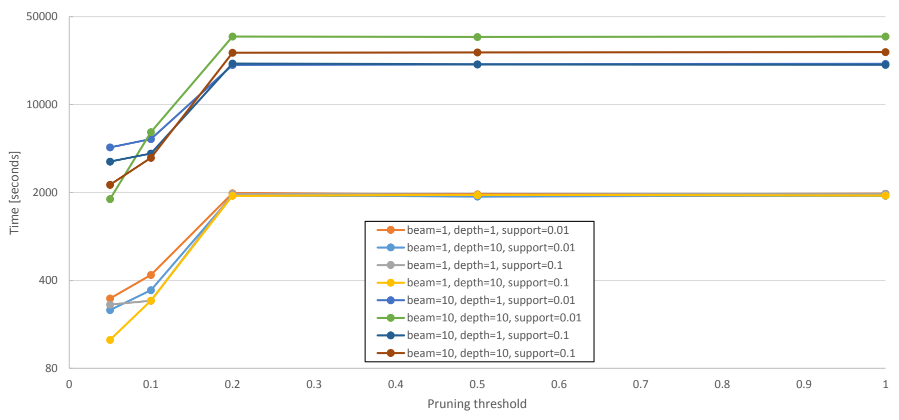


Figure 4.5: The times (in seconds) on logarithmic scale (y axis) taken by Hedwig to discover rules with different settings for beam, depth and support at varying settings for the node pruning threshold c (x axis).

Table 4.12: Results of Algorithm 4.1 on the breast cancer data set using the Personalized PageRank scoring function with advanced node removal. We only list the smallest cutoff value at which each network conversion method (naïve and Hypergraph, respectively) discovers the same rules as for cutoff value 1, and one cutoff value below that threshold.

Conversion method	Cutoff threshold	GO-term		Lift	Coverage	Positives
None	1	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37
Naïve	0.01	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37
Naïve	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

Table 4.13: Results of Algorithm 4.1 on the ALL data set using the Personalized PageRank scoring function with advanced node removal. We only list the smallest cutoff value at which each network conversion method (naïve and Hypergraph, respectively) discovers the same rules as for cutoff value 1, and one cutoff value below that threshold.

Conversion method	Cutoff threshold	GO-term		Lift	Coverage	Positives
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
naïve	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
naïve	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

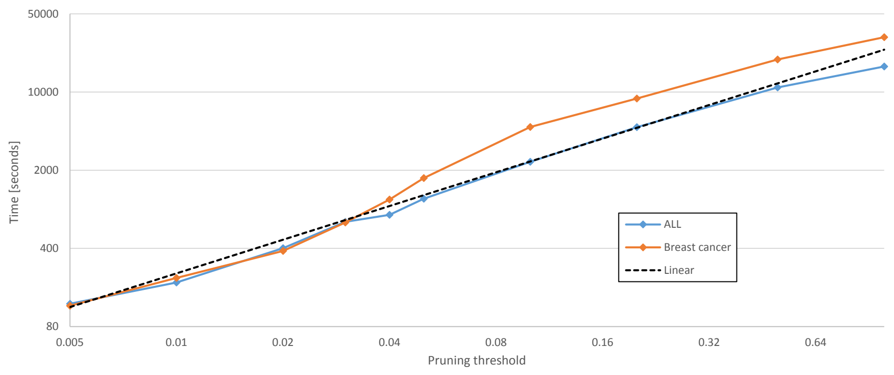


Figure 4.6: The times (in seconds) on logarithmic scale (y axis) taken by Hedwig to discover rules in the third round of experiments. Both axes are drawn at a logarithmic scale, and a linear trendline (dashed) is drawn for comparison.

4.2.3.3 Third setup: advanced network conversion and node deletion

The results of the third set of experiments show a different pattern from those in the second round of experiments. Table 4.12 shows the results of both network conversion methods on the breast cancer data set, while Table 4.13 shows the results of the methods on the ALL data set. Unlike in the second set of experiments, the results of this setup show that the quality of the rules, and in fact entire rules discovered, remains the same when we decrease the size of the background knowledge to one tenth its original size. When decreasing the network further, we see that the naïve method of network conversion outperforms the hypergraph construction; while the hypergraph version of the background knowledge returns significantly worse results at 1% of its original size (in the breast cancer data set), the naïve version still finds the same rule as we find without network pruning. When pruning the network even further, the quality of the resulting decreases, but the decrease in quality happens at a smaller size for the naïve network conversion.

Figure 4.6 shows the times, required for Hedwig to discover the rules on the pruned networks in the third round of experiments. The graph shows that unlike the case with naïve network deletion methods in the second set of experiments, the time required to discover rules decreases linearly as we lower the pruning threshold. When reducing the network to 1 percent of its original size, the algorithm was able to discover the rules in two minutes, compared to the runtime of over 11 hours on the non-pruned network.

The result of this set of experiments can be compared with the results of the first set of experiments to draw two conclusions. The naïve version of network node removal causes Hedwig to discover rules of a slightly higher quality, but the results of decreasing the cutoff threshold are unpredictable both in the quality of the rules discovered (which for some cutoff values falls below the base value obtained on an unpruned network), and the times taken to discover the rules do not decrease significantly until we lower the cutoff value to below 0.2. On the other hand, the advanced version of node removal allows the algorithm to discover the same results as on an unpruned network even for very small cutoff values. The results remain consistent, and the times taken to discover the rules decrease much more predictably with this node deletion method.

4.3 Application in Explaining Gene Expression in Potato Virus Infection

In this section, we present an application of the netSDM methodology on a biological data set. We applied the NetSDM methodology on a real-life data set containing gene expression data for potato plants, infected with the PVY (Potato Virus Y) virus. The data was collected by collaborating biologists of the National Institute for Biology (NIB) in Ljubljana, Slovenia.

4.3.1 Gene expression data set

The data set was the result of a biological experiment, performed at NIB, examining gene expression after a viral infection with the PVY (Potato Virus Y) virus (Baebler et al., 2014). Potato plants were grown in a controlled environment for 4 weeks. After this period, the leaves of the plants were dusted in carborundum powder and rubbed with cheesecloth dipped in a sap prepared from the leaves of PVY-infected tobacco plants. Control plants were treated with mock inoculations where water was used instead of the sap. The procedure was repeated on two strains of potato plants: Rywal and PW363.

On day 1, 3 and 6 after infection, RNA samples were collected from plants and analyzed using microarrays. The result was a collection of 42,034 continuous gene expression values for each treatment (mock or virus), each genotype (Rywal and PW363) and each day post infection (1, 3 and 6). The data was analyzed in the R environment (Team, 2012) with the Bioconductor Limma package (Smyth, 2005). The background signal on all of the microarrays was uniform and low and was hence ignored in further calculations. Differentially expressed genes (Benjamini and Hochberg corrected P-value ≤ 0.05, fold change ≥ 2) between virus and mock-inoculated plants at each time point and for each genotype were identified using the empirical Bayes method (Smyth, 2005).

The input of the NetSDM workflow was a set of 42,034 data instances (microarray probes) belonging to one of three groups: positively expressed, negatively expressed, or non-expressed. The instances were annotated by Gene Ontology terms.

4.3.2 GO-slim

Previous experiments with the Hedwig algorithm, presented in Sections 4.2.2 and 4.2.3 have already shown that using SDM on gene expression data sets results in informative rules explaining experimental results. However, similarly to the data sets used in Section 4.2, the analysis takes several hours to complete with large beam and depth settings for Hedwig. So far, the best option of reducing time complexity without severely reducing either beam or depth settings was to use a smaller version of the GO ontology, called a GO-slim. GO-slims are cut-down versions of the GO ontologies that contain a subset of the terms in the entire ontology.

Just like the Gene Ontology, GO-slims are freely available on the GO website[1] which also provides tools to map a data set, annotated with the entire ontology, into one annotated with a selected GO-slim. As our experiments were conducted on (potato) plants, we used the Plant slim ontology developed by The Arabidopsis Information Resource[2] which is a cut-down version of the GO ontology focusing on terms, more closely related to plants. The current version of the GO-slim ontology, for example, contains only 99 terms (only 0.22% of the terms in the Gene Ontology).

[1]http://geneontology.org/page/go-slim-and-subset-guide
[2]http://www.arabidopsis.org/

Table 4.14: Results of NetSDM on positively expressed genes 1 day after infection for the PW363 genotype.

%	GO-term		Lift	Cov	Pos	Time[seconds]
1	GO:0044464 GO:0009753 GO:0003824 GO:0044237 GO:0009605	cell part response to jasmonic acid catalytic activity cellular metabolic process response to external stimulus	4.378	14	8	**35,576**
0.01	GO:0044464 GO:0009753 GO:0003824 GO:0044237 GO:0009605	cell part response to jasmonic acid catalytic activity cellular metabolic process response to external stimulus	4.378	14	8	**223**
0.005	GO:0060089 GO:0005783	molecular transducer activity endoplasmic reticulum	3.283	14	6	**97**
GO-slim	GO:0009719 GO:0006464 GO:0005829 GO:0016740	response to endogenous stimulus cellular protein modification process cytosol transferase activity	2.775	21	8	**111**

Table 4.15: Results of NetSDM on positively expressed genes 3 days after infection for the PW363 genotype.

%	GO-term		Lift	Pokritost	Pozitivnih	Čas[sekunde]
1	GO:0060089 GO:0001101 GO:0016301	molecular transducer activity response to acid chemical kinase activity	4.650	14	6	**37,238**
0.01	GO:0060089 GO:0001101 GO:0016301	molecular transducer activity response to acid chemical kinase activity	4.650	14	6	**259**
0.005	GO:0048046 GO:0005773 GO:0043167 GO:0003824	apoplast vacuole ion binding catalytic activity	4.468	17	7	**141**
GO-slim	GO:0016020 GO:0007154 GO:0005215	membrane cell communication transporter activity	4.088	24	9	**89**

4.3.3 Experiments and results

We ran the NetSDM methodology on the data sets. We used the naïve network conversion (simply transforming semantic relations into edges of an information network) and advanced node deletion methods (taking relation transitivity into account) that proved most efficient in Section 4.2. Like in Section 4.2, we ran the methodology with different cutoff thresholds ranging from 0.005 on to discover the threshold at which we discovered the same rules as when running Hedwig alone on the entire ontology. To compare the results, we also ran the Hedwig algorithm on the plant GO-slim. In this section, we report the results of our experiments.

4.3.4 PW363 genotype

Tables 4.14, 4.15 and 4.16 show the results of the experiments when searching for explanations of the positively expressed instances for the three time points (1, 3 and 6 days after infection). The first row of every table shows the rules discovered with no pruning using the Hedwig algorithm with Gene Ontology as the background knowledge. The second row shows the smallest discovered cutoff value to which we can prune the background knowledge using NetSDM and still discover the same rules — the cutoff percentage for that

Table 4.16: Results of NetSDM on positively expressed genes 6 days after infection for the PW363 genotype.

%	GO-term		Lift	Cov	Pos	
1	GO:0005575 GO:0009873 GO:0005634 GO:0034641	cellular_component ethylene-activated signaling pathway nucleus cellular nitrogen compound metabolic process	7.100	18	8	**26,849**
0.03	GO:0005575 GO:0009873 GO:0005634 GO:0034641	cellular_component ethylene-activated signaling pathway nucleus cellular nitrogen compound metabolic process	7.100	18	8	**595**
0.005	GO:0001071 GO:0009755 GO:0003006	nucleic acid binding transcription factor activity hormone-mediated signaling pathway developmental process involved in reproduction	6.846	14	6	**88**
GO-slim	GO:0003700 GO:0007165 GO:0003676 GO:0005623	transcription factor activity, sequence-specific DNA binding signal transduction nucleic acid binding cell	3.412	37	8	**105**

Table 4.17: Results of NetSDM on negatively expressed genes 1 day after infection for the PW363 genotype.

%	GO-term		Lift	Cov	Pos	Time[seconds]
1	GO:0001906 GO:0005575	cell killing cellular_component	3.648	14	7	**38,853**
0.01	GO:0001906 GO:0005575	cell killing cellular_component	3.648	14	7	**226**
0.005	GO:0022892 GO:0006082 GO:0005575	substrate-specific transporter activity organic acid metabolic process cellular_component	3.474	21	10	**155**
GO-slim	GO:0007165 GO:0005794 GO:0005488 GO:0005886	signal transduction Golgi apparatus binding plasma membrane	2.500	25	9	**135**

rows therefore varies depending on the particular experiment. The third row shows the rules discovered by pruning the background knowledge to 0.5% of its original size. Finally, we report the rules discovered by Hedwig by using the plant GO-slim as the background knowledge. For each cutoff value, we also report the times taken by Hedwig to complete the rule discovery process.

Table 4.14 shows that the rule, discovered by Hedwig taking the entire Gene Ontology as its background knowledge is the highest quality discovered rule. However, this same rule can also be discovered using NetSDM by removing all but 1% of the background knowledge with the time required to discover the rules reducing from slightly less than 10 hours to less than 4 minutes. Reducing the background knowledge further, to 0.5% its original size reduces the time further to approximately 100 seconds, however this setting results in a lower quality of the discovered rule. Finally, running the Hedwig algorithm on the GO-slim ontology results in a rule that is discovered relatively quickly (100 seconds). However, this rule is always of a much lower quality than the rule discovered using the entire Gene Ontology. The rule is also worse when compared to the rule discovered on the

Table 4.18: Results of NetSDM on negatively expressed genes 3 days after infection for the PW363 genotype.

%	GO-term		Lift	Cov	Pos	Time[seconds]
1	GO:0043226 GO:0015662 GO:0000166 GO:0006164	organelle ATPase activity, coupled to transmembrane movement of ions, phosphorylative mechanism nucleotide binding purine nucleotide biosynthetic process	3.475	15	5	**28,525**
0.03	GO:0043226 GO:0015662 GO:0000166 GO:0006164	organelle ATPase activity, coupled to transmembrane movement of ions, phosphorylative mechanism nucleotide binding purine nucleotide biosynthetic process	3.475	15	5	**790**
0.005	GO:0051704 GO:0005829 GO:0043169 GO:0003824	multi-organism process cytosol cation binding catalytic activity	5.957	14	8	**128**
GO-slim	GO:0005794 GO:0005515 GO:0016740	Golgi apparatus protein binding transferase activity	3.608	23	8	**144**

Table 4.19: Results of NetSDM on negatively expressed genes 6 days after infection for the PW363 genotype.

%	GO-term		Lift	Cov	Pos	Time[seconds]
1	GO:0009987 GO:0051169 GO:0003674 GO:0044424	cellular process nuclear transport molecular_function intracellular part	5.064	14	6	**37,943**
0.03	GO:0009987 GO:0051169 GO:0003674 GO:0044424	cellular process nuclear transport molecular_function intracellular part	5.064	14	6	**910**
0.005	GO:0051704 GO:0003006 GO:0005737 GO:0005488	multi-organism process developmental process involved in reproduction cytoplasm binding	5.064	14	6	**154**
GO-slim	GO:0009856 GO:0005575 GO:0006950	pollination cellular_component response to stress	4.223	20	7	**153**

most severely pruned ontology. A very similar result is shown in Table 4.15: the highest quality rule is discovered in slighly more than 10 hours, and the same rule was discovered in 4 minutes on the pruned background knowledge (with the cutoff set at 1%). Setting the cutoff at 0.5% lowers the quality of the discovered rule, using the GO-slim instead results in an even lower quality rule. For the data set collected 6 days after infection, the cutoff value was set at 3% to discover the same rule as on the entire Gene Ontology. This meant that the time saving was smaller, but still very significant (8 hours reduced to 10 minutes). Pruning the network further again caused a slight decrease in rule quality, also resulted in a rule of a higher quality than the one discovered using GO-slim as a background knowledge.

Tables 4.17, 4.18 and 4.19 show the results of the experiments when searching for explanations of the negatively expressed genes for the three time points. The pattern, described previously for positively expressed instances, is also evident here, with NetSDM discovering in minutes what Hedwig alone can only find after hours of search. In all

Table 4.20: Results of NetSDM on positively expressed genes 1 day after infection for the Rywal genotype.

%	GO-term		Lift	Cov	Pos	Time[seconds]
1	GO:0071840	cellular component organization or biogenesis	2.366	14	9	**30,592**
	GO:0048581	negative regulation of post-embryonic development				
	GO:2000241	regulation of reproductive process				
0.01	GO:0071840	cellular component organization or biogenesis	2.366	14	9	**304**
	GO:0048581	negative regulation of post-embryonic development				
	GO:2000241	regulation of reproductive process				
0.005	GO:0022414	reproductive process	2.629	14	10	**164**
	GO:0005829	cytosol				
	GO:0016787	hydrolase activity				
	GO:0005488	binding				
GO-slim	GO:0009653	anatomical structure morphogenesis	2.317	20	12	**135**
	GO:0007154	cell communication				

experiments, the cutoff value of 3% was sufficient for NetSDM to return the same rule as Hedwig on an un-pruned Gene Ontology. In half the cases, reducing the cutoff value to 1% was also possible with no decrease in rule quality. Table 4.24 shows the only outlier from there rules. In the case of searching for rules, explaining negatively expressed instances 3 days post infection, the GO-slim background knowledge actually slightly outperformed the entire Gene Ontology. In this case, however, pruning the Gene Ontology to 0.5% of its original size, NetSDM discovered a rule that outperforms both un-pruned ontologies, and the rule was discovered in only two minutes.

Figure 4.7 shows the time required to discover the rules for both the PW363 genotype and the Rywal genotype described in the following section. Similarly to Figure 4.6, it shows time savings, proportional to the amount of ontology terms pruned.

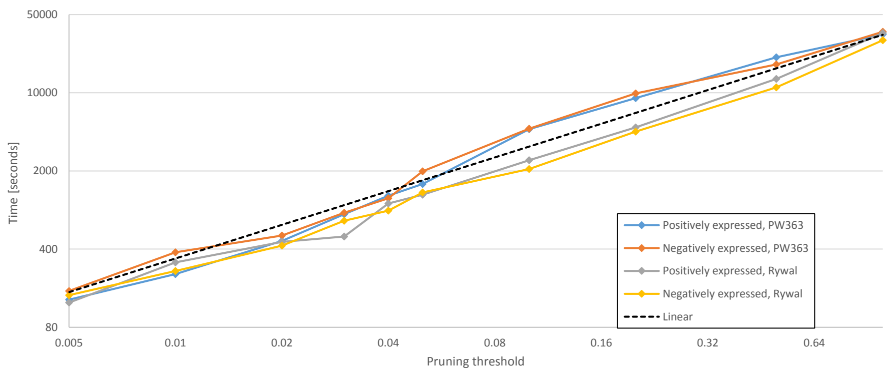


Figure 4.7: The times (in seconds) on logarithmic scale (y axis) taken by Hedwig to discover rules for the potato virus infection data set. Both axes are drawn at a logarithmic scale, and a linear trendline (dashed) is drawn for comparison.

4.3.5 Rywal genotype

Tables 4.20, 4.21 and 4.22 show the rules explaining positively expressed genes on the Rywal genotype on the three time points, respectively. The results are comparable to those presented in Section 4.3.4 and confirm the phenomena described there. Using NetSDM, we

Table 4.21: Results of NetSDM on positively expressed genes 3 days after infection for the Rywal genotype.

%	GO-term		Lift	Cov	Pos	Time[seconds]
1	GO:0030054	cell junction	2.562	19	17	**47,257**
	GO:0003006	developmental process involved in reproduction				
	GO:0016020	membrane				
	GO:0009987	cellular process				
	GO:0003674	molecular_function				
0.005	GO:0030054	cell junction	2.562	19	17	**164**
	GO:0003006	developmental process involved in reproduction				
	GO:0016020	membrane				
	GO:0009987	cellular process				
	GO:0003674	molecular_function				
GO-slim	GO:0009791	post-embryonic development	2.222	29	21	**140**
	GO:0003676	nucleic acid binding				
	GO:0000166	nucleotide binding				
	GO:0000003	reproduction				

Table 4.22: Results of NetSDM on positively expressed genes 6 days after infection for the Rywal genotype.

%	GO-term		Lift	Cov	Pos	Time[seconds]
1	GO:0005057	signal transducer activity, downstream of receptor	4.779	14	6	**24,667**
	GO:0000165	MAPK cascade				
0.03	GO:0005057	signal transducer activity, downstream of receptor	4.779	14	6	**518**
	GO:0000165	MAPK cascade				
0.005	GO:0001071	nucleic acid binding transcription factor activity	4.646	24	10	**114**
	GO:0009755	hormone-mediated signaling pathway				
	GO:0003676	nucleic acid binding				
	GO:0006351	transcription, DNA-templated				
GO-slim	GO:0009791	post-embryonic development	3.562	23	7	**141**
	GO:0016301	kinase activity				
	GO:0000003	reproduction				
	GO:0006464	cellular protein modification process				
	GO:0005575	cellular_component				
	GO:0000166	nucleotide binding				

are able to discover the same rules as Hedwig on the entire data set, but in a fraction of the time required by Hedwig. The rules are of much higher quality than those discovered by using GO-slim as background knowledge and take only slightly longer to discover. Using very low values cutoff for background knowledge pruning results in rules that are better than those from using GO-slim, and are discovered in a comparable time frame. Rules for the set of positively expressed genes 3 days after infection stayed the same even when pruning the ontology to 0.5% its original size.

The same phenomenon observed on the positively expressed genes is also exhibited in results on the negatively expressed instances, shown in Tables 4.23 and 4.25. Just like for the PW363 genotype, the only outlier is the set of negatively expressed genes 6 days after infection. For this set, the GO-slim ontology outperformed the entire Gene Ontology. The highest quality rules were discovered by pruning the ontology to 0.5% its original size.

Table 4.23: Results of NetSDM on negatively expressed genes 1 day after infection for the Rywal genotype.

%	GO-term		Lift	Cov	Pos	Time[seconds]
1	GO:0060089 GO:0005773	molecular transducer activity vacuole	2.672	14	7	**34,541**
0.01	GO:0060089 GO:0005773	molecular transducer activity vacuole	2.672	14	7	**210**
0.005	GO:0044702 GO:0005975 GO:0016043 GO:0044464	single organism reproductive process carbohydrate metabolic process cellular component organization cell part	2.672	14	7	**170**
GO-slim	GO:0040029 GO:0006464 GO:0005515	regulation of gene expression, epigenetic cellular protein modification process protein binding	2.264	22	9	**110**

Table 4.24: Results of NetSDM on negatively expressed genes 3 days after infection for the Rywal genotype.

%	GO-term		Lift	Cov	Pos	Time[seconds]
1	GO:0004871 GO:0004714 GO:0005575	signal transducer activity transmembrane receptor protein tyrosine kinase activity cellular_component	2.738	15	10	**36,426**
0.03	GO:0004871 GO:0004714 GO:0005575	signal transducer activity transmembrane receptor protein tyrosine kinase activity cellular_component	2.738	15	10	**864**
0.005	GO:0032502 GO:0004674 GO:0005515 GO:0043229 GO:0044763	developmental process protein serine/threonine kinase activity protein binding intracellular organelle single-organism cellular process	3.559	15	13	**161**
GO-slim	GO:0009628 GO:0016787 GO:0005975 GO:0006950	response to abiotic stimulus hydrolase activity carbohydrate metabolic process response to stress	3.054	27	19	**122**

Table 4.25: Results of NetSDM on negatively expressed genes 6 days after infection for the Rywal genotype.

%	GO-term		Lift	Cov	Pos	Time[seconds]
1	GO:0060089 GO:0004714	molecular transducer activity transmembrane receptor protein tyrosine kinase activity	3.531	18	7	**17,490**
0.03	GO:0060089 GO:0004714	molecular transducer activity transmembrane receptor protein tyrosine kinase activity	3.531	18	7	**155**
0.005	GO:0060089 GO:0016020 GO:0032502 GO:0016021	molecular transducer activity membrane developmental process integral component of membrane	3.026	15	5	**134**
GO-slim	GO:0007154 GO:0016787 GO:0005575 GO:0006950	cell communication hydrolase activity cellular_component response to stress	3.982	22	9	**135**

4.4 Conclusion

We presented and tested several variations of combining network analysis with semantic data mining. The result is an overall four-step framework (background knowledge conversion, network importance calculation, network pruning, and semantic data mining) for data analysis. Results show that using a naïve network conversion, the Personalized PageRank function to estimate node importance, and an advanced version of network pruning (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, but after the NetSDM data processing, Hedwig rule discovery is faster by a factor of almost 100. If we replace the advanced node removal function with a naïve version, the time saving is less predictable, but the algorithm discovers different rules than if run on a non-pruned data set.

We also tested the algorithms on a new biological data set containing gene expression data after an infection with the PVY virus. Our experiments show that if we annotate the data set with the Gene Ontology, NetSDM can discover high quality rules in a fraction of the time Hedwig requires to discover the same rules. Previously, it was only possible to discover rules in such a short amount of time if we used the GO-slim ontology instead of the GO ontology, however resorting to GO-slim significantly reduced the quality of the resulting rules. Using NetSDM, we can discover rules of much higher quality in only slightly more time than using GO-slim. Alternatively, using extremely low cutoff values, NetSDM can discover high quality rules (but worse than on an un-pruned data set) on GO in the same time as Hedwig can discover lower quality rules on GO-slim.

Chapter 5

Banded Matrix Visualization

In the previous chapter, we focused on improving the scalability and speed of semantic data mining, however the results of the algorithms were output in the same format as before our proposed methods. One problem of SDM algorithms is that their outputs are relatively hard to read and understand, even by field specialists. In this chapter, we present our work on improving the visualization of outputs of semantic data mining algorithms by using banded matrix algorithms. Our work was guided by the idea that SDM algorithms discover patterns in the data they are applied to. These patterns can also be exposed in some other way by manipulating the input data directly – if some structure can be found that exposes the SDM-discovered patterns and makes them more easily readable, then this will both reinforce the results of the SDM algorithms and make them more comprehensible.

5.1 Description of the Methodology

In this section, we describe a three-step methodology for data clustering, explanation and visualization. We first describe the banded matrix algorithm used, and then show how it fits into the proposed three-step methodology, developed by Adhikari et al. (2016).

5.1.1 Banded matrix algorithm in detail

Consider a binary matrix M with N rows and d columns and two permutations, κ and π of the first N and d integers. Matrix M_κ^π, defined as $(M_\kappa^\pi)_{i,j} = M_{\kappa(i),\pi(j)}$, is constructed by applying the permutations π and κ on the rows and columns of M. If, for some pair of permutations π and κ, matrix M_κ^π has the following properties:

- each row i of the matrix has the *consecutive ones property*. This means that the column indices for which the value in the matrix is 1 appear consecutively. In other words, for each i, there exist $a_i, b_i \in \mathbb{N}$ such that

$$(M_\kappa^\pi)_{i,j} = \begin{cases} 1 & \text{if } a_i \leq x \leq b_i \\ 0 & \text{otherwise} \end{cases}$$

- for each i, we have $a_i \leq a_{i+1}$ and $b_i \leq b_{i+1}$ (here, a_i and b_i are the same values as in the point above)

then the matrix M is *fully banded*. Furthermore, if matrix M is fully banded, then its transposition $M^\top$ is also fully banded.

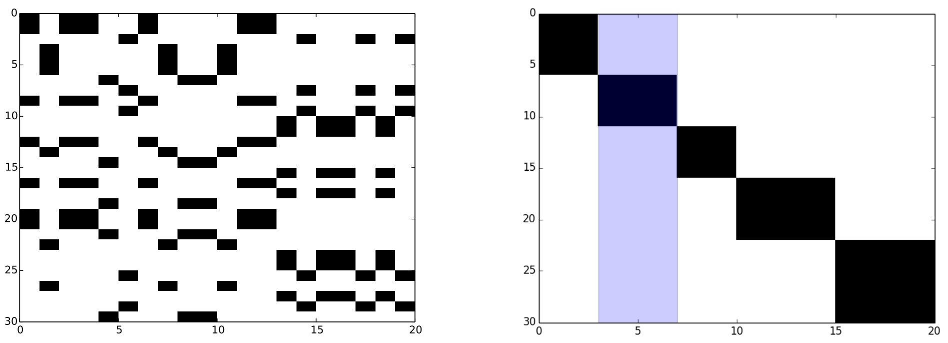


Figure 5.1: An example of a binary matrix before and after row and column permutations exposing a banded structure.

Figure 5.1 demonstrates the motivation behind banded matrices as it shows that finding the banded structure of a matrix also exposes the clustered structure of the underlying data. For a fully banded matrix, it can be shown that a banded structure can be found in polynomial time (Garriga et al., 2011). We cannot expect, however, that real-world matrices will be fully banded. The problem is that, for a noisy matrix, finding the correct row and column permutations that show a structure, close to a banded one, may be computationally infeasible. We therefore need algorithms that are designed not only to find column and row permutations that are close to banded, but also find these 'almost banded' structures in a reasonable time.

The method used to find the banded structure of a matrix in this work, called the bidirectional minimum matrix augmentation (biMBA) method, was first proposed in (Sugiyama et al., 1981) and was first used as a method of banded matrix extraction in (Garriga et al., 2011). One step of the method consists of three substeps:

1. Each row of the binary matrix is transformed to have the consecutive ones property. This is done by finding the smallest number of matrix elements which have to be changed (either from 1 to 0 or from 0 to 1) in order for the row to have the consecutive ones property. The elements which have to be changed can be found in linear time by transforming the problem into a *maximum subarray problem.* This transforms matrix M into matrix M'.

2. The algorithm finds each pair of rows M_1, M_2 such that $a_1 < a_2$ and $b_1 > b_2$, where the column indices for which values 1 appear in row M_i range from a_i to b_i for $i = 1, 2$. In other words, the algorithm finds all pairs of rows where one row's indices of ones completely envelops the other row's. The motivation behind this is that a matrix with the consecutive ones property is fully banded if and only if no such pair of indices can be found. For each such pair, the algorithm performs the minimum number of changes required that either $a_1 = a_2$ or $b_1 = b_2$. This step transforms matrix M' into matrix M''. As shown by Garriga et al. (2011), after this step, the resulting matrix M'' is fully banded.

3. In the final step, the algorithm finds the permutation π that exposes the banded structure of M'' (meaning that (1) matrix $M^{(\pi)}$ has the consecutive ones property and (2) the intervals of ones of matrix $M^{(\pi)}$ move strictly to the right).

In this way, the method calculates the best possible (according to some criterion) permutations of rows that will best expose the banded structure of the input matrix. The

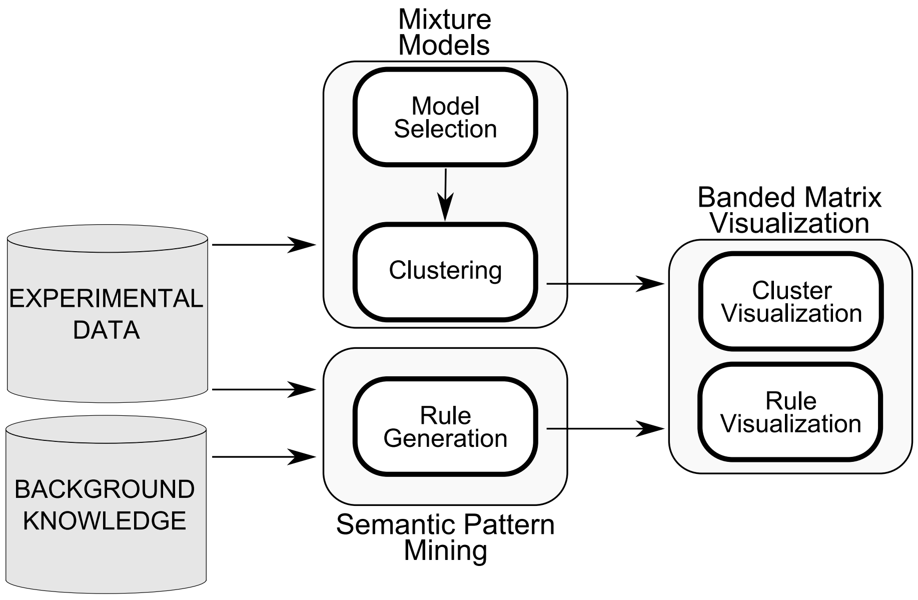


Figure 5.2: Overview of the proposed three-part methodology used in the analysis of high dimensional multiresolution data.

result of the method is the original matrix M, on which we apply the permutation π. However, the biMBA algorithm is non–optimal, heuristic, and does not find any permutation of columns (Garriga et al., 2011). The *alternating* biMBA method (Garriga et al., 2011) transposes the resulting matrix and iteratively repeats the described method on the transposed matrix until convergence or until reaching a predetermined number of steps. This second iterative step, however comes with a price in when applied to the data set described in Section 5.2.1: as neighboring columns of a matrix represent chromosome bands that are in physical proximity to one another, the goal may be to only find the optimal row permutation while not permuting the matrix columns.

As motivated by Figure 5.1, finding a banded structure of a matrix will expose the cluster structure of the underlying data. The image of the banded structure can be overlaid with a cluster visualization. Because the rows of the matrix represent instances, highlighting one set of instances (one cluster) means highlighting several matrix rows. If the discovered clusters are exposed by the matrix structure, we can expect that several adjacent matrix rows will be highlighted, forming a wide band. Highlighting clusters need not be limited to only one cluster: because each instance belongs to exactly one cluster, we can highlight all of them at once. The only limitation is the number of clusters: because each cluster is colored with its own color, too many clusters may cause colors to be too similar for the human eye.

5.1.2 Data clustering and visualization methodology

We used the banded matrix algorithm described in the previous section in a joint methodology for knowledge discovery from data proposed by Adhikari et al. (2016). The methodology merged three data analysis methods into a single pipeline, outlined in Figure 5.2.

The methodology starts with a set of experimental data and background knowledge facts as shown in Figure 5.2. Next, both a mixture model and a banded matrix are induced independently from the data. The mixture model results in a set of k probabilities $(\pi_1, \ldots, \pi_k)$ for each data instance, where k is manually selected to the value at which an objective function estimating the fit of the data stops increasing noticeably. The objective function used is the log-likelihood of the data given the probabilities $(\pi_i)_i$, estimated using cross validation. The model is then applied to the original data, to obtain a clustering of the data in which each data instance is associated with the index of the largest probability π_i. A detailed description of the methods used to calculate the values of π_i is available in Adhikari et al. (2016), however for the purposes of this work, the resulting index assignment is considered as an input to the banded matrix methodology which presents the novel work, presented in this thesis. In our contribution, we implemented the banded matrix step of the methodology which enables the visualization of the resulting clusters. The step is described in detail in Section 5.1.1. A third part of the methodology, semantic pattern mining describes the clusters as conjunctions of terms in the background knowledge. In the final step, all three models (the mixture model, the banded matrix and the patterns) are joined to produce the final visualization.

5.2 Data Set Description

This section describes the application of the visualization methodology to several real-world data sets. We first describe the data sets and then the results.

5.2.1 Multiresolution chromosomal amplification data set

A wide range of genetic mutations and molecular mechanisms known as chromosomal aberrations have been identified as the hallmarks of various disorders such as cancer, schizophrenia, and infertility (Albertson, 2006). In cancer research, identification and characterization of chromosomal aberrations are crucial to study and understand pathogenesis of cancer. Furthermore, study of chromosomal aberrations provides necessary information to select the optimal target for cancer therapy on individual level (Kirsch, 1993). The study of chromosomal aberrations also has several clinical applications such as studying multiple congenital abnormalities and identifying the family history of Down syndrome (Obe & Vijayalaxmi, 2007).

The data set under study describes DNA copy number amplifications in 4,590 cancer patients. The data describes 4,590 patients as data instances, with attributes being chromosomal locations indicating aberrations in the genome. These aberrations are described as ones (amplification) and zeros (no amplification). Myllykangas et al. (2006) describe the amplification data set in detail. Amplification data is further described at two different resolution levels (312 and 393 locations, for 24 different chromosomes).

Given the complexity of the multiresolution data, we were forced to reduce the complexity of the learning setting to a simpler setting, allowing us to develop and test the proposed methodology. To this end, we have reduced the size of the data set: from the initial set of instances describing 4,590 patients, each belonging to one of the 73 different cancer types, we have focused on the 34 most frequent cancer types only, as there were small numbers of instances available for many of the rare cancer types, thus reducing the data set from 4,590 instances to a 4,104 instances data set. The choice of the 34 most frequent cancers is motivated by the fact that it covers 90% of entire data. The original data with 393 genomic locations are high dimensional and could result in a curse of dimensionality (Bellman, 1961) greatly increasing the runtime of the algorithm. Therefore, we focused on the data from a single chromosome in our experiment, using as input to step 2 of the proposed methodology the data clusters obtained at coarse resolution using a mixture modeling approach (Myllykangas, Tikka, Böhling, Knuutila, & Hollmén, 2008).

When chromosomes are extracted from the data, some cancer patients show no amplifications in any regions of the chromosome 1. We have removed such samples without amplifications (zero vectors) because we are interested in the amplifications and their relation to cancers, not their absence. We considered negation cases unsuitable because we are only investigating one chromosome at a time. A negation result could infer that if a region is not aberrated, it is likely to be a specific cancer which will be misleading as information from other chromosomes is missing. This reduces the sample size, for example sample size of chromosome 1 is reduced from 4,104 to 407. While this data reduction may be an over-simplification, finding relevant patterns in this data set is a huge challenge, given the fact that even individual cancer types are known to consist of cancer sub-types which have not yet been explained in the medical literature. If we consider the entire data, our initial experiments have shown that inference and density estimation will produce degenerate results because of the curse of dimensionality (Bellman, 1961). Additionally, the experiments performed on chromosome 1 can be seamlessly extended to all the other chromosomes, thus efficiently using each sample present in the data. Furthermore, chromosome-wise analysis can generate chromosome-specific patterns for certain cancer types. The proposed methodology may prove, in future work, to become a cornerstone in developing means through which such sub-types could be discovered, using automated pattern construction and innovative pattern visualization using banded matrices visualization.

In addition to the DNA amplifications data set, we used supplementary background knowledge in the form of an ontology to enhance the analysis of the data set. The supplementary background knowledge consists of a hierarchical structure of multiresolution amplification data, chromosomal locations of fragile sites, virus integration sites, cancer genes, and amplification hotspots. The hierarchical structure of multiresolution data is due to International System of Cytogenetic Nomenclature (ISCN) which allows an exact description of all numeric and structural amplifications in genomes (Shaffer & Tommerup, 2005). A fragile site is a chromosomal region that tends to show a constriction or a gap and may tend to break on metaphase chromosomes when subjected to partial replication stress, i.e. following partial inhibition of DNA synthesis (Durkin & Glover, 2007). A metaphase chromosome is a chromosome in the stage of the cell cycle (the sequence of events in the life of a cell) when a chromosome is most condensed, highly coiled, and aligned in the equator of the cell before being separated into each of the two daughter cells. At this stage of the cell cycle, it is easiest to distinguish and study the chromosome. Virus integration sites are also the chromosomal locations where viral DNA inserts into host-cell DNA (zur Hausen, 2009). Approximately, 12% of cancers are caused by viruses (zur Hausen, 2009). Cancer genes are also the chromosome locations of known cancer causing genes. The list of virus integration sites was obtained from (Futreal et al., 2004). Amplification hotspots are frequently amplified chromosomal loci identified using computational modeling (Myllykangas et al., 2006). Chromosome fragile sites and virus integration sites were used as additional desriptions of the data sets (forming additional columns in the data matrix indicating an abberation occurs in a particular site).

5.2.2 Publicly available data sets

In addition to the chromosomal aberrations data, we experimented with our proposed three-part visualization methodology on four additional data sets used by Ristoski and Paulheim (2014):

Cities data set describes the most and least liveable cities in the world according to the Mercer rankings[1].

NY Daily data set describes the crawled news items along with their sentiment scores.

[1]https://mobilityexchange.mercer.com/Insights/quality-of-living-rankings

Tweets data set is a collection of tweets with different features and the original task was to identify different sports related tweets. In this scenario we use the 'DBpedia Direct Types' ontology from (Ristoski & Paulheim, 2014).

StumbleUpon data set consists a of training data set used in a Kaggle competition.

Since the publicly available data sets were highly sparse, we preprocessed them to remove highly sparse variables. In the Cities data set, we selected only those features which were positive in at least 25 different samples. Similarly, in the Cities data set we also eliminated features that were dense, i.e. those variables that were positive in more than 170 rows. In the NY Daily data set, we selected only those features that were positive in more than 200 samples but less than 450 samples. Furthermore, we selected only those features that were positive in more than 400 samples in the Stumble Upon data set. Similarly, we selected only those features that were positive in more than 100 samples of the tweets data set. The motivation behind removing those features was the fact that features that are sparse or too dense carry little information with regard to different types of classes within the data. Furthermore, removing those sets of features mitigates negative effects of the curse of dimensionality (Bellman, 1961).

5.3 Results

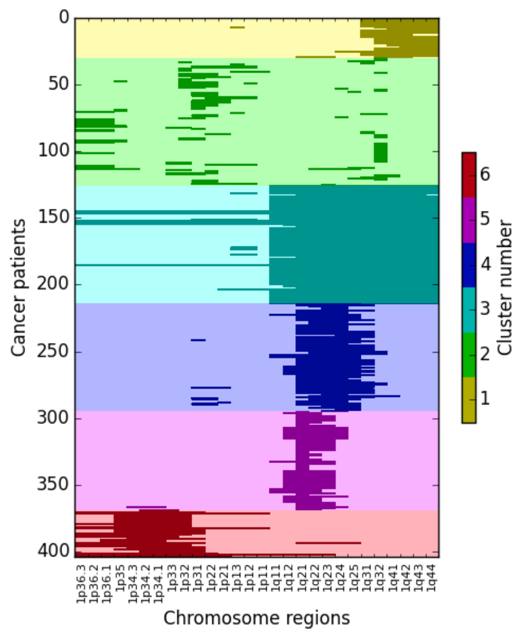
We first describe the results of cluster visualization using banded matrices on each of the data sets. This is followed by the results of semantic rule visualization.

5.3.1 Cluster visualization using banded matrices

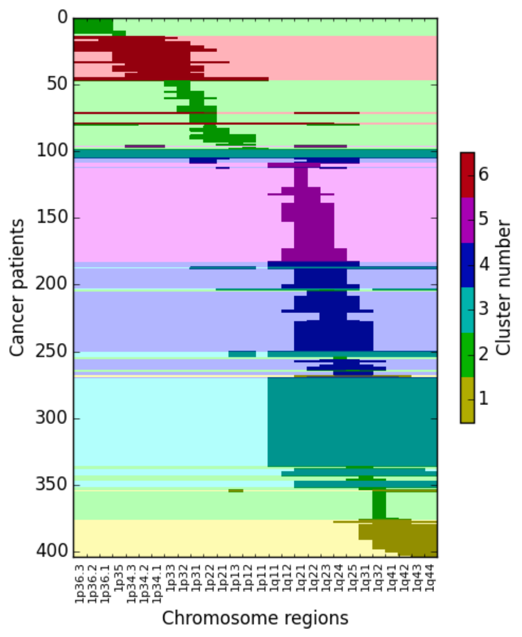
We used the bidirectional minimal banded augmentation method, described in Section 5.1.1, to extract the banded structure in the data. As explained in Section 5.1.1, we decided to only allow permutations of rows of the data matrix. The dark color indicates ones in the data and light color denotes zeros in the data. The resulting data was then overlaid with the 6 clusters, discovered using mixture models, as shown in Figure 5.3.

Figure 5.3 shows two visualizations of the clusters discovered in the data. In Figure 5.3a, the data instances are sorted so that all instances in the same cluster appear together as consequtive rows in the visualization. Figure 5.3b, shows the same data set with the instances sorted to expose the banded structure of the data set. Comparing the two it is clear that Figure 5.3a shows a more homogeneous coloring, which is expected as a this was the only criterion for sorting the data instances before visualizing them. On the other hand, the colors in Figure 5.3b are mostly, but not entirely grouped together. Unlike Figure 5.3a, Figure 5.3b does not group instances from cluster 2 together, but rather splits them into several subsets. These subsets, clearly visible in Figure 5.3b, are not appparent in 5.3a. Clusters $1, 5$ and 6 are very well isolated by the banded matrix algorithms. Overall, by exposing the banded structure of a matrix, Figure 5.3b allows a clear visualization of not only the clusters discovered in the data, but also how the inner structure of each cluster. Examination of the figure shows that each cluster captures amplifications in some specific regions of the genome. Also evident simply from the figure is the phenomenon that the p-arm of chromosome 1 (left part of the figure) shows a comparatively smaller number of amplifications whereas the q-arm shows a higher number of amplifications. As this is also true for the mixture model results obtained in the first step of the visualization methodology, we see that finding the banded matrix structure of the data has enabled more insightful visual inspection of the data.

During the mixture modeling phase of the methodology, Cluster 1 was characterized by pronounced amplifications in the end of the q-arm (regions 1q32–q44) of chromosome



(a) Cluster-based sorting



(b) Banded matrix visualization

Figure 5.3: Visualization of the clusters discovered in the data. The data in Figure 5.3a is sorted so that data instances in the same cluster appear together. Figure 5.3b shows an alternative visualization of the clusters in the data where the cluster information is shown over a banded structure, independendly discovered from the data.

1. Samples in the second cluster contained sporadic amplifications spread across both p and q-arms in different regions of chromosome 1. This cluster does not carry much information and contains cancer samples that do not show discriminating amplifications in chromosomes as the values of random variables (values of π_i for instances in this cluster) were near 0.5. This is consistent with the fact that cluster 2 is the only cluster that was split into many separate matrix regions. In contrast, Cluster 3 portrayed marked amplifications in regions 1q11–44. Cluster 4 showed amplifications in regions 1q21–25. Similarly, Cluster 5 is denoted by amplifications in 1q21–25. Cluster 6 is defined by pronounced amplifications in the p-arm of chromosome 1.

The results of the mixture modeling step matched very well with the banded matrix visualization shown in Figure 5.3b. All of the clusters, except for the uninformative Cluster 2, were almost completely isolated by the banded matrix algorithms. Especially interesting is the fact that left image in Figure 5.3b also draws a distinction between Clusters 4 and 5 which at first show no obvious difference to the human eye.

5.3.1.1 Public data sets

On the publicly available data sets, we ran the alternating biMBA method to expose the banded structure of the matrices. The choice of alternating method was motivated by the fact that the ordering of the columns in the publicly available data sets was arbitrary. This is unlike the amplification data set which had fixed ordering of regions in the genome.

Cities The biMBA algorithm converged after 7 iterations exposing the banded structure of the matrix. The banded structure in Figure 5.4 clearly visualizes the four clusters found by the presented methodology. Clusters 2 and 3 are almost completely separated from Clusters 1 and 4. The visualization shows that Cluster 1 and Cluster 2 are both composed of two parts which are hard to distinguish. This phenomenon was also captured during model selection in Cities data set because increase in the validation likelihood was minimal when the number of components were increased from 3 to 4. When we selected four components, a relatively homogeneous cluster is broken down into two.

NY Daily The biMBA algorithm converged after 11 iterations for the NY Daily data set. The banded structure in Figure 5.5 clearly highlights Clusters 1, 2 and 6 and shows that Clusters 4 and 3 are more similar to each other. Interestingly, even though Cluster 3 is split into several parts, it can still be seen that the annotations, drawn on the left side of the visualization, are more important for Cluster 3 (meaning that splitting the two clusters was a good choice). As in the Cluster 2 of the amplification data set, the algorithm also highlights the Cluster 5 which does not capture a specific pattern but patterns scattered across different columns in the data set.

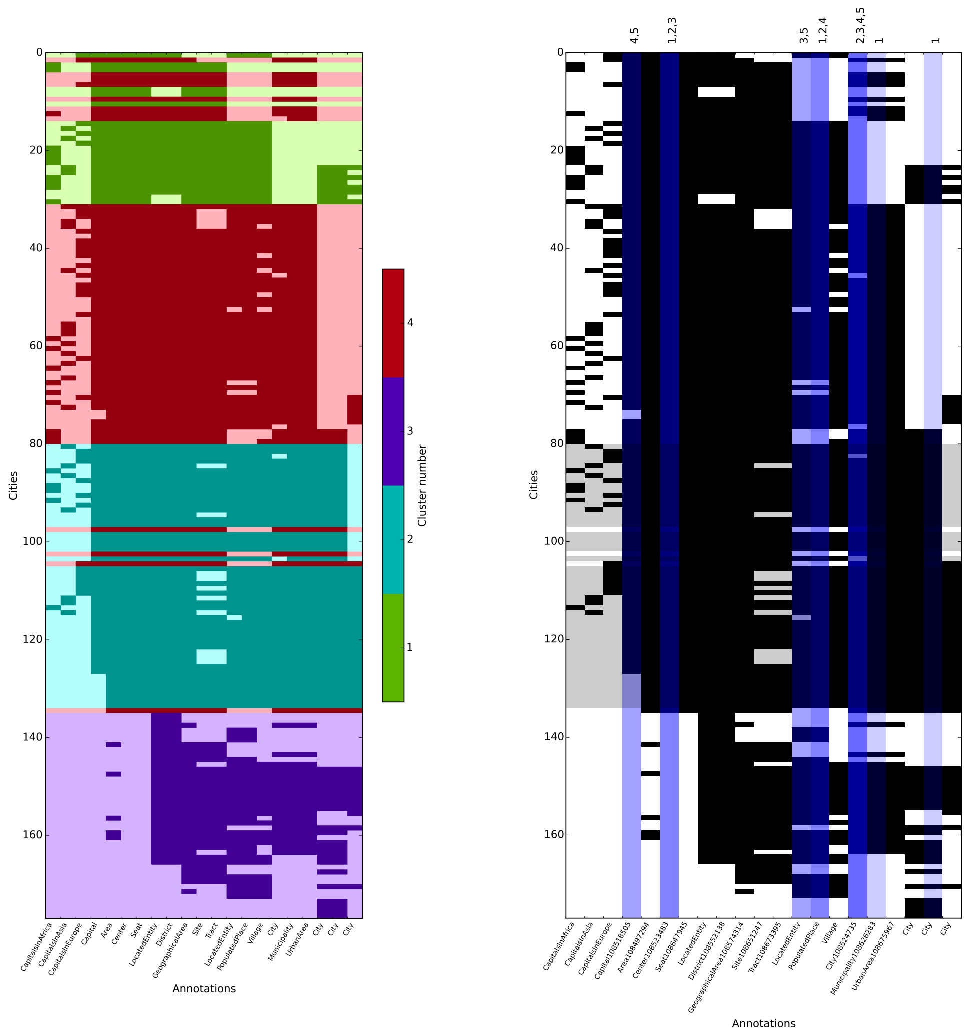
Tweets The biMBA algorithm converged after 33 iterations for the Twitter data set with credible results confirming the cluster allocation from step 1 of the methodology. The visualization provided in Figure 5.6 shows that Clusters 1, 2 and 3 are clearly separable from the rest of the data set. Cluster 4, the largest of the clusters, is split into two large parts, both of which are fairly homogeneous. However, Clusters 5, 6, and 7 were relatively small with the value mixture components equal to 0.07, 0.05, and 0.03. Hence, these clusters are not fully exposed in the visualization.

Stumble Upon The Stumble Upon data set was the only data set on which our methodology did not achieve credible results, as shown in Figure 5.7. The model selection procedure shows that both training and validation likelihood smoothly increases until the number of components is 20. Even with the number of components greater than 20 the validation likelihood did not decrease showing that there is no apparent structure in the data. Similarly, the biMBA algorithm converged much more slowly than in the other data sets, taking 521 iterations to reach the optimal banded structure. Visualizing the structure shows that the data is fractured into several small chunks. Some clusters, like Clusters 8 and 10, are separated from the rest, but the remaining clusters are sporadically scattered across all rows. This negative result, however, still shows us that the overall methodology is sound, as the banded matrix visualization comfirmed the result of mixture modelling – there is no underlying structure to discover in the Stumble Upon data set.

5.3.2 Rules induced through semantic pattern mining

Using the Hedwig algorithm, described in Chapter 4, we induced subgroup descriptions for each cluster as the target class. For a selected cluster, all the other clusters represent the negative training examples, which resembles one-versus-all approach in multiclass classification. In this section, we focus our discussion on the results pertaining to Clusters 1 and 3 (see Tables 5.1 and 5.2), however we provide the results of the other clusters as well. In our experiments we have considered only rules without negations in the rule conditions, as we are interested in the existence of amplifications characterizing the clusters (and thereby the specific cancers). In contrast, the absence of amplifications normally characterizes the absence of cancers not their presence.

Tables 5.1 and 5.2 show the rules induced for Clusters 1 and 3, together with their relevant statistics. The rules presented in Table 5.2 quantify the clustering results obtained



#	Rule			TP	FP	Precision	Lift

1	`Cluster2(X)`	←	`City(X)`
			`PopulatedPlace(X)`
			`Center(X)`
			`Municipality(X)`
				51	0	1.00	3.47

2	`Cluster2(X)`	←	`PopulatedPlace(X)`
			`City(X)`
			`Place(X)`
			`Center(X)`
				49	0	1.00	3.47

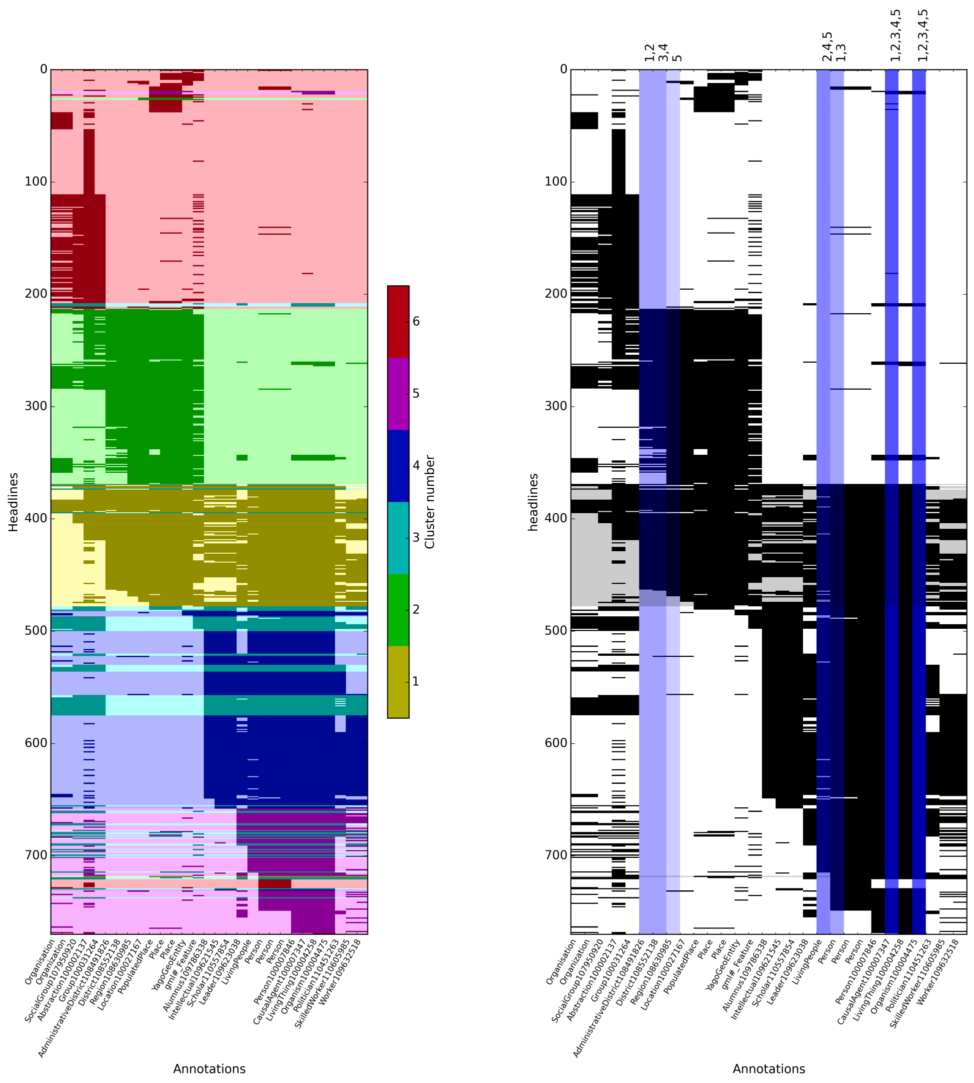
3	`Cluster2(X)`	←	`Location_Underspecified(X)`
			`City(X)`
			`Place(X)`
			`Center(X)`
				48	0	1.00	3.47

4	`Cluster2(X)`	←	`PopulatedPlace(X)`
			`City(X)`
			`Place(X)`
			`Capital(X)`
				42	0	1.00	3.47

5	`Cluster2(X)`	←	`Location_Underspecified(X)`
			`City(X)`
			`Place(X)`
			`Capital(X)`
				41	0	1.00	3.47

Figure 5.4: Results of the methodology on the Cities data set.

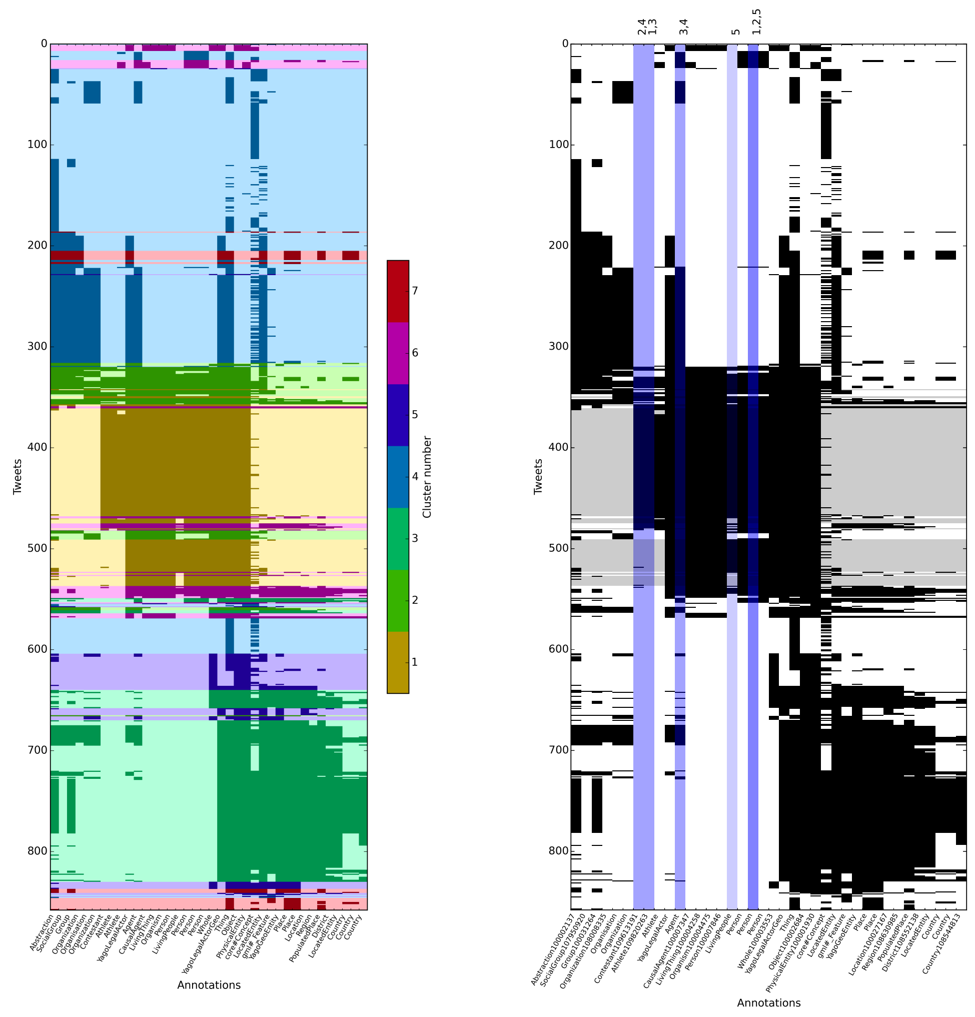
through mixture modeling and confirmed by banded matrix visualization in Section 5.3.1. The banded matrix visualizations are depicted in Figure 5.3 and show that Cluster 3



#	Rule			TP	FP	Precision	Lift
	`Cluster1(X)`	←	`Agent(X)`				
1			`AdministrativeDistrict(X)`	90	0	1.00	7.20
			`Organism(X)`				
	`Cluster1(X)`	←	`Organism(X)`				
2			`AdministrativeDistrict(X)`	87	0	1.00	7.20
			`LivingPeople(X)`				
	`Cluster1(X)`	←	`Agent(X)`				
3			`District(X)`	92	1	0.99	7.12
			`Organism(X)`				
	`Cluster1(X)`	←	`Organism(X)`				
4			`District(X)`	89	1	0.99	7.12
			`LivingPeople(X)`				
	`Cluster1(X)`	←	`Organism(X)`				
5			`Region(X)`	92	2	0.98	7.04
			`LivingPeople(X)`				

Figure 5.5: Results of the methodology on the NY Daily data set.

is marked by the amplifications in the regions 1q11–44. However, the rules obtained in Table 5.2 show that amplifications in all the regions 1q11–44 do not equally discriminate Cluster 3. For example, the rule $\boxed{\text{Rule 1: } \texttt{Cluster3(X)} \leftarrow \texttt{1q43-44(X)} \wedge \texttt{1q12(X)}}$ characterizes Cluster 3 best with a precision of 1. This means that amplifications in regions 1q43–44 and 1q12 denote Cluster 3. The rule covers 81 of the 88 samples in Cluster 3.



#	Rule			TP	FP	Precision	Lift
1	`Cluster1(X)`	←	`Person(X)` `Athlete(X)`	117	30	0.80	4.21
2	`Cluster1(X)`	←	`Person(X)` `Contestant(X)`	120	32	0.79	4.18
3	`Cluster1(X)`	←	`Agent(X)` `Athlete(X)`	117	32	0.79	4.16
4	`Cluster1(X)`	←	`Agent(X)` `Contestant(X)`	120	34	0.78	4.13
5	`Cluster1(X)`	←	`Person(X)` `LivingPeople(X)`	145	46	0.76	4.02

Figure 5.6: Results of the methodology on the Tweets data set.

Clinically, the aberration in these regions characterize Ependymoma (Myllykangas et al., 2008).

Nevertheless, amplifications in regions 1q11–44 that were identified through mixture modeling as discriminating regions appear in at least one of the rules in Table 5.2 with a varying degree of precision. The first part of the rule (i.e. amplifications in region 1q43–44) is the most discriminating for Cluster 1 as shown in Table 5.1. However, precision and lift are considerably reduced.

Although the rule $\boxed{\text{Rule 2: } \texttt{Cluster1(X)} \leftarrow \texttt{1q43-44(X)}}$ appears in semantic de-

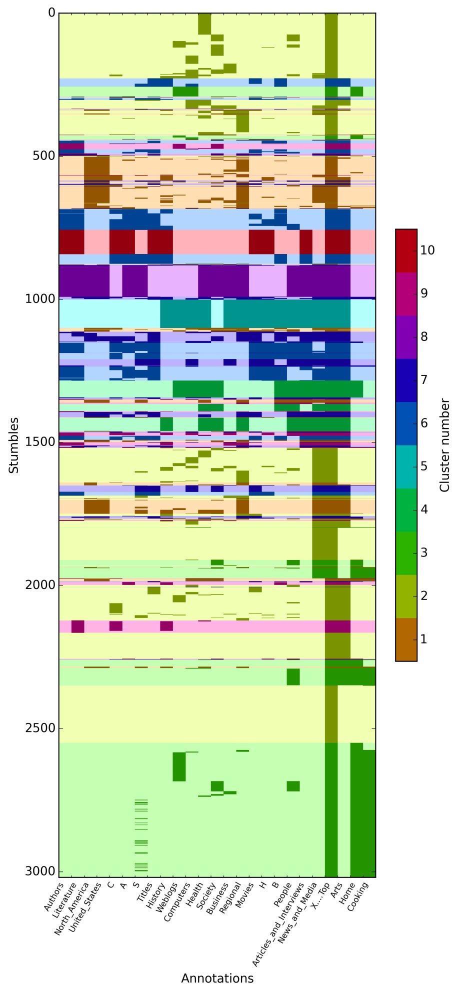


Figure 5.7: Results of the banded matrix visualization on the StumbleUpon data set. The data is more fragmented compared to the NY daily, Tweets and Cities data set. As Hedwig discovered no significant rules, we only show the banded matrix visualization.

#	Rules for Cluster 1	TP	FP	Precision	Lift
1	`Cluster1(X)` ← `1q43-44(X)`	26	88	0.23	3.09
2	`Cluster1(X)` ← `1q41(X)`	26	90	0.22	3.04
3	`Cluster1(X)` ← `1q32(X)`	24	116	0.17	2.33
4	`Cluster1(X)` ← `HotspotSite(X)`	30	280	0.10	1.31
5	`Cluster1(X)` ← `FragileSite(X)`	30	317	0.09	1.17

Table 5.1: Rules induced for Cluster 1 of the chromosomal amplification data set.

scriptions of both Cluster 1 and 3, addition of a conjunct 1q12 to the rule improves the discriminating power for Cluster 3. Rule 2 covers all 88 samples of Cluster 3 with precision of 0.77 whereas it covers 26 out of 30 samples in Cluster 1 with the precision of 0.23. This shows that amplifications in region 1q43–44 characterize both Clusters 1 and 3. If the negation rules are considered, amplifications only in regions 1q43-44 would more

#	Rules for Cluster 3	TP	FP	Precision	Lift
1	`Cluster3(X)` ← `1q43-44(X)` ∧ `1q12(X)`	81	0	1.00	4.62
2	`Cluster3(X)` ← `1q11(X)`	78	9	0.90	4.15
3	`Cluster3(X)` ← `1q43-44(X)`	88	26	0.77	3.57
4	`Cluster3(X)` ← `1q41(X)`	88	28	0.76	3.51
5	`Cluster3(X)` ← `1q12(X)`	81	43	0.65	3.02
6	`Cluster3(X)` ← `1q32(X)`	88	52	0.63	2.91
7	`Cluster3(X)` ← `1q31(X)`	87	54	0.62	2.85
8	`Cluster3(X)` ← `1q25(X)`	88	64	0.58	2.68
9	`Cluster3(X)` ← `1q24(X)`	88	97	0.48	2.20
10	`Cluster3(X)` ← `1q21(X)`	88	134	0.40	1.83
11	`Cluster3(X)` ← `1q22-24(X)`	88	149	0.37	1.72
12	`Cluster3(X)` ← `HotspotSite(X)`	88	222	0.28	1.31
13	`Cluster3(X)` ← `CancerSite(X)`	88	245	0.26	1.22
14	`Cluster3(X)` ← `FragileSite(X)`	88	259	0.25	1.17

Table 5.2: Rules induced for Cluster 3 of the chromosomal amplification data set.

likely make it a candidate for Cluster 1. Similarly, the second most discriminating rule for Cluster 3 is $\boxed{\text{Rule 2: }\texttt{Cluster3(X)} \leftarrow \texttt{1q11(X)}}$ which covers 78 positive samples and 9 negative samples.

The rules listed in Table 5.2 also capture the multiresolution phenomenon of the data. We input only one resolution of data to the algorithm but the hierarchy of different resolutions is made available to the algorithm as background knowledge. For example, the literal 1q43–44 denotes a joint region in coarse resolution thus showing that the algorithm produces results at different resolutions. The results at different resolutions improve the understandability and interpretability of the rules (Hollmén & Tikka, 2007).

Furthermore, other information added to the background knowledge are amplification hotspots, fragile sites, cancer genes, which are discriminating features of cancers but do not show to discriminate any specific clusters present in the data. Therefore, such additional information can be better utilized in situations where the data set contains not only cancer samples but also control samples. Unfortunately, this is not the situation here as our data set has only cancer patients.

5.3.2.1 Public data sets

We ran the same semantic subgroup discovery procedure (with the same parameters) on the publicly available data sets. Due to the large amount of experimental results, we chose to describe one cluster and the top five rules for that cluster for each data set (Figures 5.4, 5.5 and 5.6). For the Stumble Upon data set we report only the (lacking) banded matrix visualization to illustrate an instance where our methodology is not applicable (as described in Section 5.3.1.1).

Cities Cluster 2 was chosen as an example of a very well characterized cluster (Figure 5.4). We report the top five rules, all with 100% precision. The first rule actually perfectly describes the cluster, since it covers all examples from Cluster 2.

By investigating the rule conjuncts it follows that this cluster contains cities that are at the same time annotated as centers, municipalities and populated places. Furthermore, the cities data set comes with a label describing its livability: low, medium, and high (Ristoski & Paulheim, 2014). Although clustering, rule extraction, and visualization were performed independent of the labels, the rules and clusters mostly describe cities with medium and high livability.

In the table, we omit the full concept URIs of concepts for visual clarity. Nevertheless, the exact semantics of each concept can be verified by visiting the corresponding DBpedia pages, e.g., full URI of Center is http://dbpedia.org/class/yago/Center108523483.

NY Daily For this data set, we report the top five rules for Cluster 1 (Figure 5.5). Similar to the previous data set, the found rules are of high precision and each covers a relatively large portion of examples from this cluster (a total of 107 examples).

Compared to the subgroup descriptions found for the other five clusters, this cluster contains mainly headlines annotated with the District and Region concepts, together with Agent and Organism concepts.

Tweets For this data set we feature the top five rules for Cluster 1 (Figure 5.6). The rules found were of lower precision (76%-80%), which indicates that this cluster is harder to describe compared to the clusters mentioned in the previous two data sets. Nevertheless, the subgroup descriptions indicate that this cluster contains mainly tweets mentioning specific athletes (i.e. annotated with Person and Athlete concepts) and not for example teams or organizations, which do appear in rules for the other clusters (e.g., Organization concept). Furthermore, the tweets data set consists of associated class labels which denotes sports related and unrelated tweets (Ristoski & Paulheim, 2014). Although clustering, rule extraction, and visualization were performed independently of the label, this cluster mostly contains the sports related tweets.

5.3.3 Visualizing semantic rules and clusters with banded matrices

The second way we can use the exposed banded structure of the data is to display columns that were found to be important due to appearing in rules from Section 5.3.2. We achieve this by highlighting the chromosomal regions which appear in the rules. The Figure 5.8 depicts colored overlays of the rules on the ordered/serialized patient-chromosome matrix. A highlighted column in the figure denotes that an amplification in the corresponding region characterizes the instances of the particular cluster. A darker hue means that the region appears in more rules. The numbers on top right of the figures correspond to rule numbers. For example, $1, 3$ above rightmost column of cluster 3 indicates that the chromosome region appears in rules 1 and 3 tabulated in Table 5.2. As shown in Figure 5.8, the highlighted band for Cluster 1 spans chromosome regions 1q32–44. For Cluster 3, the entire q-arm of the chromosome is highlighted, as indeed the instances in Cluster 3 have amplifications throughout the entire arm. We can see that the regions 1q11–12 and 1q43–44 appear in rules with higher lift, in contrast to the other regions. This tells us that the amplifications on the edges of the region are more important for the characterization of the cluster.

#	Rule	TP	FP	Precision	Lift
1	`Cluster2(X)` $\leftarrow$ `1p31(X)`	28	26	0.52	2.20
2	`Cluster2(X)` $\leftarrow$ `1p32(X)`	19	35	0.35	1.49

Table 5.3: Rules for Cluster 2.

As shown in the left panel of Figure 5.9, this cluster captures the heterogeneity in data. Since we are using only chromosome 1, this cluster is more likely to capture those mainly cancers that are characterized by aberrations in other chromosomes rather than chromosome 1. The samples from clusters are distributed in different parts by the banded

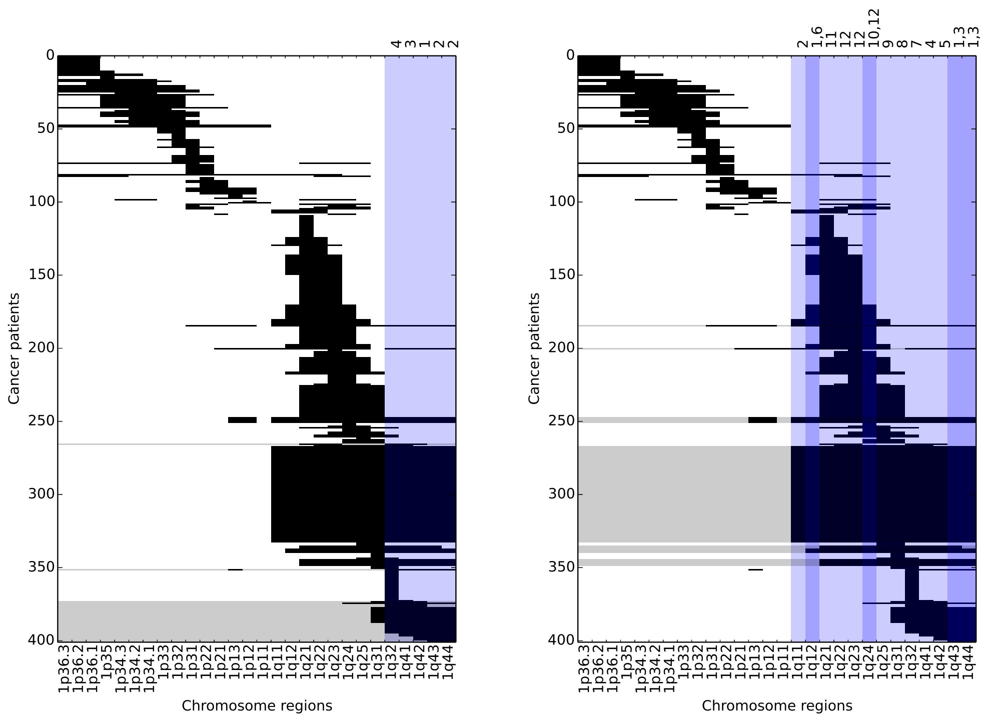


Figure 5.8: Clusters 1 (left) and 3 (right). The highlighted columns in each image correspond to the attributes that appear in the rules, tabulated in Tables 5.1 and 5.2.

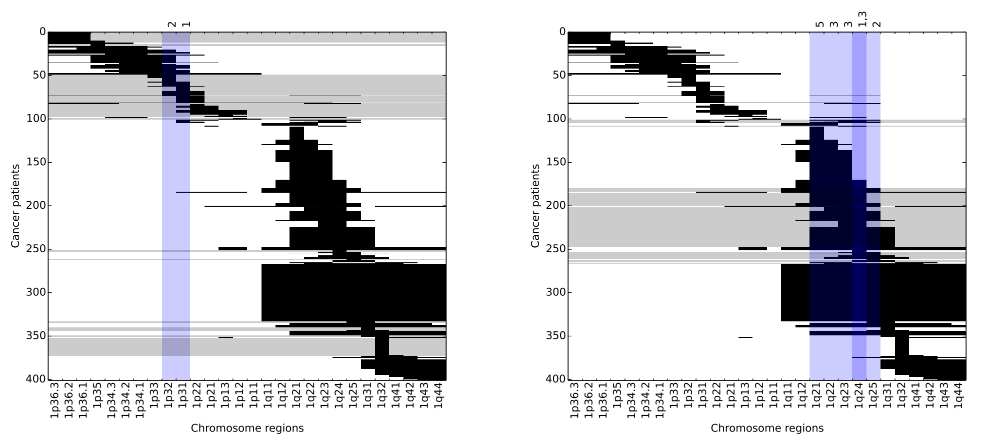


Figure 5.9: Clusters 2 (left) and 4 (right) with relevant columns highlighted.

matrix visualization. Nevertheless, the aberrations captured by these clusters are miscellaneous samples, i.e. those cancers that do not show prominent aberrations in chromosome 1. Nevertheless, aberrations captured by this cluster characterize glioblastoma multiforme (Myllykangas et al., 2008).

As shown in the right panel of Figure 5.9, this cluster captures the aberrations near the beginning of the q arm of chromosome 1. The rules tabulated in Table 5.4 show

#	Rule	TP	FP	Precision	Lift
1	`Cluster4(X)` ← `1q24(X)`	81	104	0.44	2.20
2	`Cluster4(X)` ← `1q25(X)`	57	95	0.38	1.88
3	`Cluster4(X)` ← `1q22-24(X)`	81	156	0.34	1.72
4	`Cluster4(X)` ← `HotspotSite(X)`	81	229	0.26	1.31
5	`Cluster4(X)` ← `1q21(X)`	56	166	0.25	1.27
6	`Cluster4(X)` ← `CancerSite(X)`	81	252	0.24	1.22
7	`Cluster4(X)` ← `FragileSite(X)`	71	276	0.20	1.03

Table 5.4: Rules for Cluster 4.

aberrations in regions 1q21–1q25. Clinically, the aberrations in these regions of Cluster 4 mark liposarcoma (Myllykangas et al., 2008).

#	Rule	TP	FP	Precision	Lift
1	`Cluster5(X)` ← `1q21(X)`	75	147	0.34	1.83
2	`Cluster5(X)` ← `1q12(X)`	33	91	0.27	1.44
3	`Cluster5(X)` ← `1q22-24(X)`	60	177	0.25	1.37
4	`Cluster5(X)` ← `HotspotSite(X)`	75	235	0.24	1.31
4	`Cluster5(X)` ← `HotspotSite(X)`	75	235	0.24	1.31
5	`Cluster5(X)` ← `CancerSite(X)`	75	258	0.23	1.22
6	`Cluster5(X)` ← `FragileSite(X)`	75	272	0.22	1.17

Table 5.5: Rules for cluster 5.

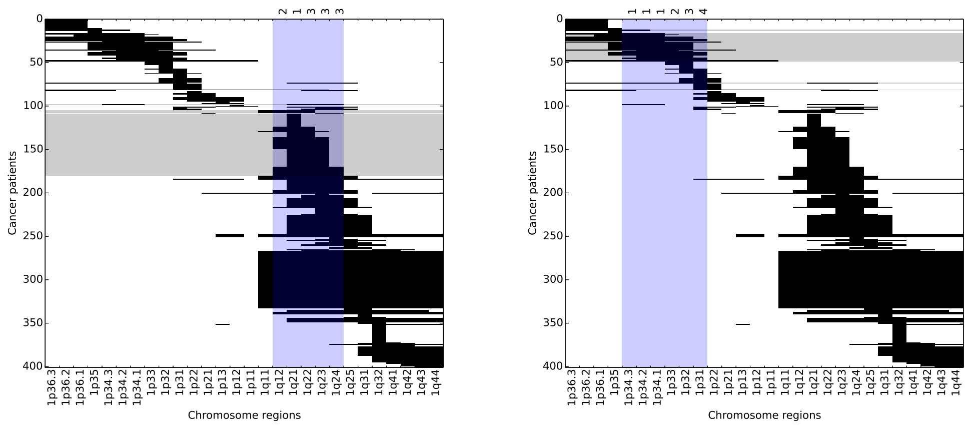


Figure 5.10: Clusters 5 (left) and 6 (right) with relevant columns highlighted.

The regions and rules in Cluster 5 depicted in the left panel of Figure 5.10 overlap with the rules describing Cluster 4. However, the rules describing these clusters have higher precision than those describing Cluster 4. These two clusters are the prime candidates if any two clusters need to be merged. In terms of clinical relevance, the aberrations in the regions captured by this cluster denote malignant fibrous histiocytoma of bone (Myllykangas et al., 2008).

The amplifications in the P-arm of Chromosome 1 captured by this cluster are depicted in the right panel of Figure 5.10. This is clearly distinguishable from other clusters because other clusters mainly capture the aberrations in Q arm of chromosome 1. The aberration

#	Rule	TP	FP	Precision	Lift	p-value
1	`Cluster6(X)` ← `1p34(X)`	37	8	0.82	9.04	0.000
2	`Cluster6(X)` ← `1p33(X)`	31	12	0.72	7.93	0.000
3	`Cluster6(X)` ← `1p32(X)`	29	25	0.54	5.91	0.000
4	`Cluster6(X)` ← `1p31(X)`	15	39	0.28	3.06	0.000
5	`Cluster6(X)` ← `CancerSite(X)`	36	297	0.11	1.19	0.000

Table 5.6: Rules for cluster 6.

in these regions characterizes the phenomenon of small cell lung cancer (Myllykangas et al., 2008).

In summary, figures 5.3 and 5.8 offer an improved view of the structure of the underlying data. The Figures show that most samples in the same cluster also come together in the banded matrix visualization although the permutations are constrained and restricted to only rows in the data set. This has been achieved by reordering the matrix rows by placing similar items closer together to form a banded structure, which allows easier visualization of the clusters and rules. It is important to reorder the rows independently of the clustering process. This is because the reordering does not depend on the cluster structure discovered. Therefore, the resulting figures offer new insight into both the data and the clustering.

5.3.3.1 Public data sets

Similar to the chromosomal aberrations data set, we also highlighted the relevant variables captured by the rules describing each cluster on the public data sets. Due to the large amount of experimental results, we chose to limit the discussions on one cluster and the top five rules for that cluster for each data set (Figures 5.4, 5.5, and 5.6). For the StumbleUpon data set we report only the (lacking) banded matrix visualization to illustrate an instance where our methodology is not applicable.

Cities Cluster 2 in the data set was perfectly described by the rules. This cluster was chosen as an example of a very well characterized cluster (Figure 5.4). The visualization shows a clear band of features, with the top instances annotated by features on the left side of the chart and the bottom instances annotated by features on the right. Cluster 2 in the middle is characterized by containing instances that are annotated by features on both sides of the band, as instances above it are not annotated by the rightmost features and instances above are not annotated by the leftmost features. The visualization shows that all five top rules cover features on both sides of the band.

NY Daily For this data set, we report the top five rules for Cluster 1 as shown in Figure 5.5. The visualization clearly identifies the banded structure of the data, with three distinct vertical bands. The cluster is characterized as the cluster which contains instances, annotated by the features in the (unlike Clusters 3 and 4) second and (unlike Clusters 2 and 6) third band. The visualisation shows that all rules take this into account as all rules explain Cluster 1 with at least one conjunct covering features on the second band and one conjunct in the third band.

Tweets For this data set we feature the top five rules for Cluster 1 (Figure 5.6). Despite lower precision of rules extracted by our methodology, the visualization still clearly shows the most important features for Cluster 1. The banded structure visualization shows us two sets of features that are important to Cluster 1. The first is the block of tweets, annotated with the terms `Athlete` and `Contestant`. One of these two annotations occurs in all top four rules, found for this cluster. The second, larger block of features is used in all top five rules we present. Additionally, the visualization of all clusters can also tell us why the precision of rules, found for this data set, was lower: Cluster 2 contains several instances which are annotated by all features that also annotate features in Cluster 1.

5.4 Conclusion

We have applied algorithms for banded matrix discovery to improve the visualization of data, in particular results of semantic data mining on the data.

The contributions of this chapter are as follows. We designed an algorithm that uses a banded-matrix finding algorithm to visualize data sets that can be presented as $0-1$ matrices. We have tested the algorithm on several data sets. On each data set, we have shown that the algorithm improves the visualization of the data set by permuting its rows and columns. By independently running a clustering algorithm on the data set, we have shown that clusters, discovered using the clustering algorithm, were easily identified after we permute the data set to discover its banded structure. The fact that the clusters and the banded structure were discovered independently reinforces each component of the described methodology. We also ran a semantic data mining algorithm to uncover rules that describe the clusters of the data set. As before, we have shown that the independently discovered banded structure allows for easier visualization of the rules, discovered by Hedwig.

The three steps of the visualization methodology were all able to discover the underlying structure of the data, with each having its benefits and all working together. The synergy between the three steps was also confirmed on the Stumble Upon data set. Although the results of the three steps were not good on this data set, it is important to note that all three steps of the methodology failed to uncover any hidden structure – it was difficult to select the number of clusters, the banded structure of the data was hard to discover and barely visible, and the discovered clusters were hard to describe using Hedwig. These three negative results show that the data set has no underlying structure to uncover.

Chapter 6

Publicly Available Software

The HinMine and NetSDM methodologies, described in Chapters 3 and 4, are publicly available. We implemented the algorithms, described in this thesis, in the ClowdFlows data mining platform. ClowdFlows is an open-source cloud-based platform for composition, execution, and sharing of interactive machine learning and data mining workflows (Kranjc et al., 2012) using a visual programming paradigm. As ClowdFlows allows users to develop their own widgets and contribute to the roster of data analysis methods available, we developed several widgets implementing the HinMine and NetSDM workflows. In this chapter, we present the implementation of both workflows. The algorithms are also joined in a GitHub repository[1].

6.1 HinMine Workflow in ClowdFlows

We implemented all the functions, used in our experiments with the HinMine methodology, in the ClowdFlows platform. The resulting workflow is shown in Figure 6.1. The workflow begins by loading a data set encoded as a `.gml` file. The GML (Graph Modeling Language) (Himsolt, 1997) is a text format that allows for easy representation of network data. For the HinMine methodology, the input requires that each node in the network is of a given type. In the online example[2], the methodology is run on a subset of the IMDB data set containing nodes of type `person` and nodes of type `movie`. One node type (the base node type for the HinMine methodology) must be labeled, and if more than one label is applicable for a node, the labels must be separated by a `---` separator. An example of a node looks as follows:

```
node [
  id 6336
  label "movie_303"
  labels "Action---Adventure---Western"
  type "movie"
  name "movie_the-quick-and-the-dead"
]
```

The methodology loads the GML file in the widget *Read GML from file*, where it identifies the base node type (the node type that is labeled) and training instances (all instances that are labeled). The loaded network is passed as the variable `net` to the *HinMine-Decompose* widget where network decomposition is applied. This is an interactive

[1]https://github.com/JanKralj/cfnetSDM
[2]http://clowdflows.org/workflow/11019/

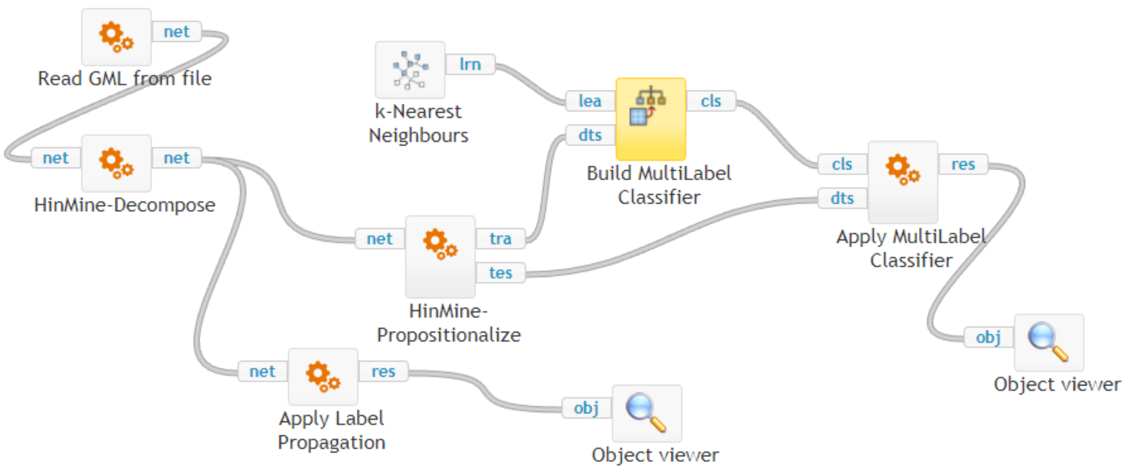
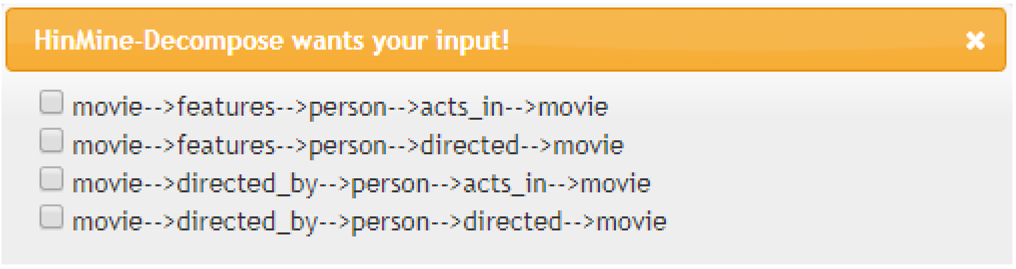


Figure 6.1: Overview of the HinMine methodology as a workflow in the Clowdflows platform.


HinMine-Decompose wants your input!

- ☐ movie-->features-->person-->acts_in-->movie
- ☐ movie-->features-->person-->directed-->movie
- ☐ movie-->directed_by-->person-->acts_in-->movie
- ☐ movie-->directed_by-->person-->directed-->movie

Figure 6.2: The decomposition selection in HinMine.

widget that, when first run, provides all possible decompositions of the input network. The widget discovers all possible decomposition paths and allows the user to choose which decompositions to perform. After performing the decompositions, the widget returns the decomposed network in the variable `net`. Figure 6.2 shows the possible decompositions of the online example. After decomposition, the methodology has two options:

1. If we classify the data set with label propagation, we can use the *Apply Label Propagation* widget. This widget performs label propagation on the network and returns the results as a numpy (Walt, Colbert, & Varoquaux, 2011) array (variable `res`).

2. If we classify the data using propositionalization, the *HinMine-Propositionalize* widget performs the network propositionalization described in Section 2.1.5. This widget constructs the feature vectors for the labeled (training, variable `tra`) and unlabeled (test, variable `tes`) nodes separately. In this way, the classifier can be trained using the *Build MultiLabel Classifier* widget on the training set. In classifier construction using the *Build MultiLabel Classifier*, any learner capable of predicting labels on data sets containing numeric values can be used as the inuput variable `lea`. In the online example, we use the k-nearest neighbours classifier. Finally, the induced classifier is applied to the test set and a numpy array is returned as a result in the variable `res`.

Both options above result in the workflow returning a numpy array. The array's columns represent the labels of the data set and the rows represent the unlabeled nodes. Each row contains results of label propagation applied to the given unlabeled node. The

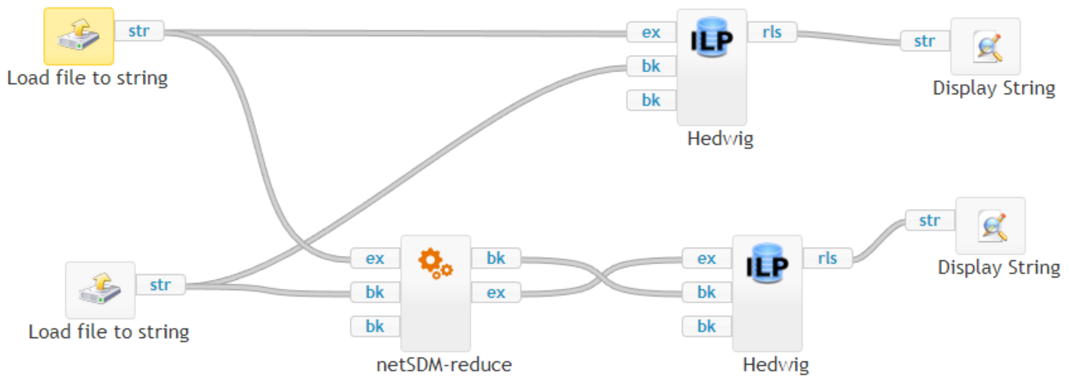


Figure 6.3: Overview of the NetSDM methodology as a workflow in the Clowdflows platform.

result is a vector of values between 0 and 1, and the higher the value, the more likely it is that the label is applicable to the given node.

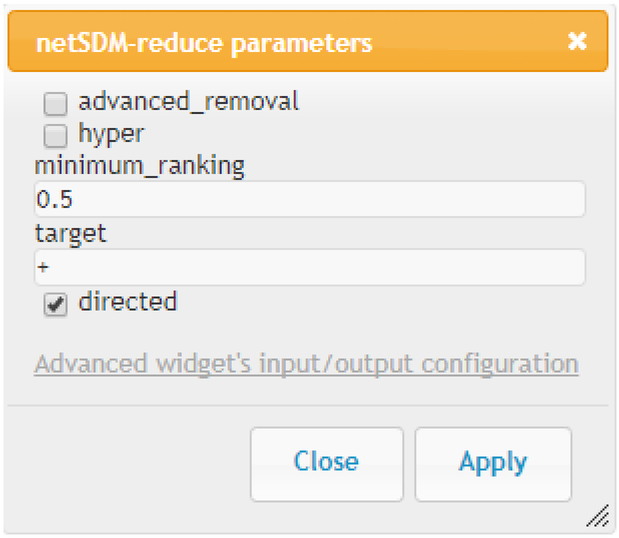
6.2 NetSDM Workflow in ClowdFlows

The workflow, implementing the NetSDM methodology, shown in Figure 6.3, is available online[3]. The workflow begins by loading the background knowledge (denoted as the input variable `bk`) and the set of examples (denoted as the variable `ex`). This step is the same as the first step of the Hedwig methodology which is also available in the ClowdFlows platform[4]. The background knowledge file is then loaded into the *netSDM-reduce* widget which prunes the background knowledge network. Double-clicking on the widget allows the user to change the parameters of the NetSDM algrithm:

- the *advaced_removal* checkbox determines whether the algorithm will use the advanced or naive node removal method,
- checking the *hyper* checkbox causes the algorithm to construct a hypergraph out of the background ontologies instead of using the naive network conversion,
- the *directed* checkbox tells the algorithm whether to take directions of network edges into account when calculating network scores,
- *minimum ranking* determinines how much of the background knowledge should remain in the pruned data set.

The widget returns two objects: the pruned background knowledge set (`bk`) and the newly annotated set of examples (`ex`. These two objects can be used to discover rules in the widget *Hedwig* that runs the Hedwig SDM algorithm.

[3]http://clowdflows.org/workflow/11015/
[4]http://clowdflows.org/workflow/7031/



☐ advanced_removal
☐ hyper
minimum_ranking
0.5
target
+
☑ directed

Figure 6.4: Selecting the parameters for the NetSDM widget.

6.3 Banded Matrix Algorithm

The banded matrix algorithm is available as a command line script from GitHub[5]. After installing the package with the command `python setup.py install`, the user can apply the banded matrix algorithm to a data set and display the data set and clusters by running

```
python -m banded_visualization clusters path-to-folder
```

The folder at location `path-to-folder` must contain at least three files (an example is also provided on GitHub in the folder `tweets`:

- `data.txt` – the text file containing the binary matrix of the data (contains zeroes and ones, separated by spaces).
- `column_names.txt` – a text file containing column names of the matrix – one name per row.
- `row_labels.txt` – a text file containing row indices and the number of a cluster each instance belongs to, separated by a tab. For example, the line `1 4` means that the first row of the data belongs to the cluster 4.

To visualize the rules explaining the data set, the folder must contain two additional files:

- `rules.txt` – the output of the Hedwig algorithm on the data.
- `hierarchy.txt` – the hierarchy of labels used by Hedwig to construct the rules. Each row of this file contains first the parent term of the background ontology and then (separated by a tab) a list of child-terms, separated by the semicolon. An example row in this folder is as follows:

  ```
  parent_node child_1;child_2;child_3
  ```

[5]https://github.com/JanKralj/banded_visualization

Chapter 7

Conclusions

Our work provides novel approaches to data mining by combining network analysis methods with machine learning algorithms. In this section, we provide a summary of the scientific contributions covered in Chapters 3, 4 and 5. We conclude our work with an overview of future work for each area of research.

7.1 Summary of Contributions

The thesis presented three connected areas of research and the advancements made in each.

1. Advances in semi-supervised learning on information networks. We examined a combination of network analysis and data mining on a semi-supervised problem in a network setting. Taking as a basis the network decomposition and propositionalization approach first proposed by Grčar et al. (2013), we designed and tested a framework for node classification in a network setting with two distinct classification options. For the first option, classification through network propositionalization, we proposed and tested several improved heuristics for heterogeneous network decomposition. We adapted seven heuristics for bag-of-words (BOW) vector construction in text mining into heuristics that can be used to weigh intermediary nodes when decomposing a heterogeneous network into individual homogeneous networks. Several of these heuristics give lower weights to words that either appear in too many documents or appear in documents with varying class labels which is a desired property for text analytics. A similar property is desirable in a network setting. Hence, we designed new heuristics to decrease the importance of intermediary nodes that either connect too many base nodes or connect nodes with varying class labels. Experiments on several data sets have shown that the naive heuristic (that can be understood as the adaptation of the term-frequency heuristic for BOW-vector construction) is outperformed by other node weighting heuristics used for network decomposition. The heuristic, adapted from the well known tf-idf (term-frequency-inverse-document-frequency) and the gain ratio heuristic, performed well on the tested data sets.

The second option for classification with heterogeneous network decomposition is to use the label propagation algorithm (D. Zhou et al., 2004). Our experiments have shown that the label propagation algorithm does not perform well on imbalanced data sets where it tends to allow the majority class to disproportionally out-vote the minority classes. We proposed a modification to the label propagation algorithm where initial weights are set to the labeled examples so that the examples from a larger class receive smaller weights and therefore have a lower effect on the label propagation. Our results show that the modified label propagation approach outperforms standard label propagation on data sets where one class is larger than the others, but suffers decreased performance when applied to data sets with very small classes.

2. Network filtering for semantic data mining. We introduced network analysis into the field of Semantic Data Mining (SDM). The work presents an approach to combining the best properties of network analysis methods (scalability and speed) with the benefits of SDM methods (capability of dealing with complex background knowledge and informativeness of results). The new methodology consists of four steps: (1) conversion of a background knowledge network into an information network, (2) assessment of background knowledge terms significance using network analysis methods, (3) pruning of the background knowledge network and (4) application of an SDM algorithm to the pruned network. We proposed and tested two methods – a naive version and one that constructs a hypergraph – of converting a background knowledge network (an input for SDM algorithms) into an information network that can be examined by network analysis methods. To assess the importance of background knowledge terms, we tested two network analysis methods and determined that Personalized PageRank weights are the best heuristic for pruning the background knowledge. Finally, for the step of pruning the background knowledge, we tested a naive and advanced version of node removal, with the naive version simply removing the node and all its connections and the advanced version removing the node but keeping the connections semantically implied by its existence. In the experiments, the advanced node removal method proved to be more consistent and resulted in better performance. Our experiments showed that by using network filtering we can decrease the runtime of the SDM algorithm by a factor of 100, finding in minutes the complex rules that – without network filtering – require several hours to discover.

3. Banded matrix aided visualization for semantic data mining. We proposed a method for data visualization that can be used in parallel with semantic data mining algorithms to provide a better understanding of both the structure of the input data and the nature of the rules resulting from the SDM algorithms. To expose the structure of the data sets, we propose to use banded matrix algorithms that search for permutations of the input data set in order to discover a hidden structure in the data set. Combined with labels for data instances, the discovered structure can further be used to expose groups of similar data instances and their labels if we color each class label with its own color. Alternatively, we can use semantic data mining algorithms to discover rules governing different class labels, and then use the banded matrix structure to expose both the class and the rules governing it. The application to several publicly available data sets showed that the banded matrix approach is useful for discovering hidden structures in data sets in all but one of the cases. The data set in which a hidden structure could not be found was also difficult to cluster using mixture models and difficult to analyze using semantic data mining algorithms which discovered no valuable rules. This gives two independent indicators that the data set simply lacks any structure we could uncover.

4. Publicly available software. The algorithms, used in this work, are publicly available online. The HinMine and NetSDM algorithms were implemented in the ClowdFlows online platform and the banded matrix algorithm is available on GitHub.

7.2 Further Work

The three main research contributions described in Chapters 3-5 show promising results and deserve further investigation.

In semi-supervised learning, we have shown that using advanced network decomposition heuristics can improve the results of network classification algorithms. An issue faced in our research is that while one heuristic can perform well on one data set, it can perform much worse on another data set. This opens an interesting research question. On one hand, further analysis is required to discover which properties of a network correlate with a change in performance of a given network decomposition heuristic. The properties that should be examined can cover any information network property such as the structure of its communities, the network's size, diameter, distribution of node degrees, etc. This analysis could be done on artificial data sets where we have full control of the network properties. On the other hand, the problem of selecting the best decomposition heuristic could also be solved through sampling. Using this approach, we would first select a small random subset of the training set and evaluate the different decomposition heuristics on that. This step may prove to be problematic because sampling of network data is difficult, a difficulty also shown in our results in Chapter 3 (Figure 3.5). After appropriately sampling the data and testing the heuristics, the results would then be used to select the best decomposition heuristic and use it for classification in a real classification problem. In further work, we wish to compare the two methods for decomposition heuristic selection.

In network filtering for semantic data mining, we plan a more comprehensive examination of how the performance of Hedwig compares to enrichment-analysis-based methods like SegMine. We plan a comparison using several biological data sets, including responses of rheumatoid arthritis patients to drug treatment. 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 network. Additionally, a non-deterministic approach to background knowledge pruning could be used instead of the hard cut-off used in the current experiments. In this approach, the scores of background knowledge terms 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 Hedwig algorithm. This can quickly negate the improvements in performance, achieved by the work done so far. Further work in the field of network filtering for semantic data mining will also include applications of the NetSDM methodology to larger data sets. Our experiments so far have been limited to the Gene Ontology background knowledge set. As Hedwig required several hours to discover interesting rules using this ontology, using much larger ontologies was not feasible without dramatically reducing the parameters guiding the search. However, using our network filtering approach, the algorithm can be applied to a much wider array of background knowledge sets. In future, we wish to examine how the network filtering approach can be used to run semantic data mining on the BioMine (Eronen, Hintsanen, & Toivonen, 2012) graph of biological entities and connections between them.

In the field of banded matrix visualization, the main goal of the resulting algorithms is improved comprehensibility of the SDM methods. Currently, the banded matrix algorithms are designed to find a structure of the data without knowing the class labels of data instances. We would like to examine the possibilities of designing algorithms that can also take class labels into account, however, a naive version simply grouping the instances with the same label would present a step back from the existing algorithm – it would not discover any real structure of the data. Additionally, we could expand the idea described in this work and use general methods which discover a structure of binary matrices.

References

Adhikari, P. R., Vavpetič, A., Kralj, J., Lavrač, N., & Hollmén, J. (2016). Explaining Mixture Models through Semantic Pattern Mining and Banded Matrix Visualization. *Machine Learning*, *105*(1), 3–39.

Agrawal, R. & Srikant, R. (1994). Fast Algorithms for Mining Association Rules in Large Databases. In *Proceedings of the 20th International Conference on Very Large Data Bases* (pp. 487–499). San Francisco, California, USA.

Albertson, D. G. (2006). Gene Amplification in Cancer. *Trends in Genetics*, *22*(8), 447–455.

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., ... Eppig, J. T., et al. (2000). Gene Ontology: Tool for the Unification of Biology. *Nature genetics*, *25*(1), 25.

Baebler, Š., Witek, K., Petek, M., Stare, K., Tušek-Žnidarič, M., Pompe-Novak, M., ... Marczewski, W., et al. (2014). Salicylic acid is an indispensable component of the Ny-1 resistance-gene-mediated response against Potato virus Y infection in potato. *Journal of Experimental Botany*, *65*(4), 1095–1109.

Bavelas, A. (1950). Communication Patterns in Task-oriented Groups. *Journal of the Acoustical Society of America*, *22*, 725–730.

Belkin, M., Niyogi, P., & Sindhwani, V. (2006). Manifold Regularization: A Geometric Framework for Learning from Labeled and Unlabeled Examples. *Journal of Machine Learning Research*, *7*(Nov), 2399–2434.

Bellman, R. E. (1961). *Adaptive Control Processes - A Guided Tour*. Princeton University Press.

Bourchtein, A. & Bourchtein, L. (2013). On some analytical properties of a general PageRank algorithm. *Mathematical and Computer Modelling*, *57*(9), 2248–2256.

Burt, R. & Minor, M. (1983). *Applied Network Analysis: A Methodological Introduction*. Sage Publications.

Callahan, A., Cifuentes, J. J., & Dumontier, M. (2015). An Evidence-based Approach to Identify Aging-related Genes in Caenorhabditis Elegans. *BMC Bioinformatics*, *16*(1), 1.

Cantador, I., Brusilovsky, P., & Kuflik, T. (2011). 2nd Workshop on Information Heterogeneity and Fusion in Recommender Systems. In *Proceedings of the 5th ACM Conference on Recommender Systems*. Chicago, IL, USA: ACM.

Chen, C.-H., Hwu, H.-G., Jang, W.-J., Kao, C.-H., Tien, Y.-J., Tzeng, S. L., & Wu, H.-M. (2004). Matrix Visualization and Information Mining. In *Proceedings in Computational statistics: 16th Symposium Held in Prague, Czech Republic* (pp. 85–100). Physica-Verlag HD.

Chen, C. & Paul, R. J. (2001). Visualizing a knowledge domain's intellectual structure. *IEEE Computer*, *34*(3), 65–71.

Consortium. (2008). The Gene Ontology Project in 2008. *Nucleic Acids Research*, *36*(suppl 1), D440–D444.

Cox, L. S. & Faragher, R. (2007). 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.

Crestani, F. (1997). Application of Spreading Activation Techniques in Information Retrieval. *Artificial Intelligence Review*, *11*(6), 453–482.

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

de Sousa, C. A. R., Rezende, S. O., & Batista, G. E. (2013). Influence of Graph Construction on Semi-supervised Learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (pp. 160–175). Springer.

Debole, F. & Sebastiani, F. (2004). Supervised Term Weighting for Automated Text Categorization. In *Text Mining and its Applications* (pp. 81–97). Springer.

Demšar, J. (2006). Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, *7*(Jan), 1–30.

D'Orazio, V., Landis, S. T., Palmer, G., & Schrodt, P. (2014). Separating the Wheat from the Chaff: Applications of Automated Document Classification Using Support Vector Machines. *Polytical Analysis*, *22*(2), 224–242.

Durkin, S. G. & Glover, T. W. (2007). Chromosome Fragile Sites. *Annual Review of Genetics*, *41*(1), 169–192.

Džeroski, S. & Lavrač, N. (Eds.). (2001). *Relational Data Mining*. Berlin: Springer.

Eronen, L., Hintsanen, P., & Toivonen, H. (2012). Biomine: A Network-structured Resource of Biological Entities for Link Prediction. In M. R. Berthold (Ed.), *Bisociative Knowledge Discovery* (Vol. 7250, pp. 364–378). Lecture Notes in Computer Science. Springer.

Freeman, L. C. (1977). A Set of Measures of Centrality Based on Betweenness. *Sociometry*, *40*, 35–41.

Freeman, L. C. (1979). Centrality in Social Networks Conceptual Clarification. *Social Networks*, *1*(3), 215–239.

Fürnkranz, J., Gamberger, D., & Lavrač, N. (2012). *Foundations of Rule Learning*. Springer.

Futreal, P. A., Coin, L., Marshall, M., Down, T., Hubbard, T., Wooster, R., ... Stratton, M. R. (2004). A Census of Human Cancer Genes. *Nature Reviews Cancer*, *4*(3), 177–183.

Garriga, G. C., Junttila, E., & Mannila, H. (2011). Banded Structure in Binary Matrices. *Knowledge and Information Systems*, *28*(1), 197–226.

Grčar, M., Trdin, N., & Lavrač, N. (2013). A Methodology for Mining Document-enriched Heterogeneous Information Networks. *The Computer Journal*, *56*(3), 321–335.

Grover, A. & Leskovec, J. (2016). 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.

Guarino, N., Oberle, D., & Staab, S. (2009). What Is an Ontology? In *Handbook on Ontologies* (pp. 1–17). Springer.

Hämäläinen, W. (2010). *Efficient Search for Statistically Significant Dependency Rules in Binary Data* (Doctoral dissertation, Department of Computer Science, University of Helsinki, Finland).

Han, E.-H. & Karypis, G. (2000). Centroid-Based Document Classification: Analysis and Experimental Results. In *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery* (pp. 424–431). Springer.

Himsolt, M. (1997). *GML: A Portable Graph File Format*. Universität Passau.

Hoehndorf, R., Dumontier, M., & Gkoutos, G. V. (2012). Identifying Aberrant Pathways through Integrated Analysis of Knowledge in Pharmacogenomics. *Bioinformatics*, *28*(16), 2169–2175.

Hollmén, J. & Tikka, J. (2007). Compact and Understandable Descriptions of Mixture of Bernoulli Distributions. In *Proceedings of the 7th International Symposium on Intelligent Data Analysis (IDA 2007)* (Vol. 4723, pp. 1–12). Lecture Notes in Computer Science. Springer-Verlag.

Huang, D. W., Sherman, B. T., & Lempicki, R. A. (2008). Systematic and Integrative Analysis of Large Gene Lists Using DAVID Bioinformatics Resources. *Nature Protocols*, *4*(1), 44–57.

Hwang, T. & Kuang, R. (2010). A Heterogeneous Label Propagation Algorithm for Disease Gene Discovery. In *Proceedings of the 2010 SIAM International Conference on Data Mining* (pp. 583–594).

Jeh, G. & Widom, J. (2002). SimRank: A Measure of Structural-context Similarity. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 538–543). ACM.

Ji, M., Sun, Y., Danilevsky, M., Han, J., & Gao, J. (2010). Graph Regularized Transductive Classification on Heterogeneous Information Networks. In *Proceedings of the 25th European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases* (pp. 570–586).

Jiline, M., Matwin, S., & Turcotte, M. (2011). Annotation Concept Synthesis and Enrichment Analysis: A Logic-based Approach to the Interpretation of High-throughput Experiments. *Bioinformatics*, *27*(17), 2391–2398.

Jones, K. S. (1972). A Statistical Interpretation of Term Specificity and Its Application in Retrieval. *Journal of Documentation*, *28*(1), 11–21.

Katz, L. (1953). A New Status Index Derived from Sociometric Analysis. *Psychometrika*, *18*(1), 39–43.

Kirsch, I. (1993). *The Causes and Consequences of Chromosomal Aberrations*. CRC Press.

Kleinberg, J. M. (1999). Authoritative Sources in a Hyperlinked Environment. *Journal of the ACM*, *46*(5), 604–632.

Klösgen, W. (1996). Explora: A Multipattern and Multistrategy Discovery Assistant. In *Advances in Knowledge Discovery and Data Mining* (pp. 249–271). American Association for Artificial Intelligence.

Kondor, R. I. & Lafferty, J. D. (2002). Diffusion Kernels on Graphs and Other Discrete Input Spaces. In *Proceedings of the 19th International Conference on Machine Learning* (pp. 315–322).

Kralj, J., Valmarska, A., Robnik-Šikonja, M., & Lavrač, N. (2015). Mining Text Enriched Heterogeneous Citation Networks. In *Proceedings of the 19th Pacific-Asia Conference on Knowledge Discovery and Data Mining* (pp. 672–683).

Kranjc, J., Podpečan, V., & Lavrač, N. (2012). Clowdflows: A Cloud Based Scientific Workflow Platform. In *Proceedings of the 12th European Conference on Machine Learning and Knowledge Discovery in Databases* (pp. 816–819). Springer.

Kwok, J. T.-Y. (1998). Automated Text Categorization Using Support Vector Machine. In *Proceedings of the 5th International Conference on Neural Information Processing* (pp. 347–351).

Lan, M., Tan, C. L., Su, J., & Lu, Y. (2009). Supervised and Traditional Term Weighting Methods for Automatic Text Categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*(4), 721–735.

Lavrač, N., Kavšek, B., Flach, P., & Todorovski, L. (2004). Subgroup Discovery with CN2-SD. *Journal of Machine Learning Research*, *5*(Feb), 153–188.

Lavrač, N., Vavpetič, A., Soldatova, L., Trajkovski, I., & Kralj Novak, P. (2011). Using Ontologies in Semantic Data Mining with SEGS and G-SEGS. In *Proceedings of the 14th International Conference on Discovery Science* (pp. 165–178). Springer.

Lawrynowicz, A. & Potoniec, J. (2011). 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* (Vol. 6804, pp. 428–437). Lecture Notes in Computer Science.

LePendu, P., Iyer, S. V., Bauer-Mehren, A., Harpaz, R., Mortensen, J. M., Podchiyska, T., . . . Shah, N. H. (2013). Pharmacovigilance using Clinical Notes. *Clinical Pharmacology & Therapeutics*, *93*(6), 547–555.

Liu, B., Hsu, W., & Ma, Y. (1998). Integrating Classification and Association Rule Mining. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining* (pp. 80–86). New York, New York, USA.

Liu, H., Dou, D., Jin, R., LePendu, P., & Shah, N. (2013). Mining Biomedical Ontologies and Data Using Rdf Hypergraphs. In *12th International Conference on Machine Learning and Applications* (Vol. 1, pp. 141–146). IEEE.

Liu, W. & Chang, S.-F. (2009). Robust Multi-class Transductive Learning with Graphs. In *IEEE Conference on Computer Vision and Pattern Recognition* (pp. 381–388). IEEE.

Lyalina, S., Percha, B., LePendu, P., Iyer, S. V., Altman, R. B., & Shah, N. H. (2013). Identifying Phenotypic Signatures of Neuropsychiatric Disorders from Electronic Medical Records. *Journal of the American Medical Informatics Association*, *20*(e2), e297–e305.

Maglott, D., Ostell, J., Pruitt, K. D., & Tatusova, T. (2005). Entrez Gene: Gene-centered Information at Ncbi. *Nucleic Acids Res*, *33*(Database issue), 54–58.

Manevitz, L. M. & Yousef, M. (2001). One-class SVMs for Document Classification. *Journal of Machine Learning Research*, *2*, 139–154.

Martineau, J. & Finin, T. (2009). Delta TFIDF: An Improved Feature Space for Sentiment Analysis. In *Proceedings of the Third AAAI Internatonal Conference on Weblogs and Social Media*. AAAI Press.

Muggleton, S. (1992). *Inductive Logic Programming*. Academic Press.

Myllykangas, S., Himberg, J., Böhling, T., Nagy, B., Hollmén, J., & Knuutila, S. (2006). DNA Copy Number Amplification Profiling of Human Neoplasms. *Oncogene*, *25*(55), 7324–7332.

Myllykangas, S., Tikka, J., Böhling, T., Knuutila, S., & Hollmén, J. (2008). Classification of Human Cancers Based on DNA Copy Number Amplification Modeling. *BMC Medical Genomics*, *1*(15).

Newman, M. (2010). *Networks: An introduction*. Oxford university press.

Obe, G. & Vijayalaxmi. (2007). *Chromosomal Alterations: Methods, Results, and Importance in Human Health*. Springer.

Ogata, H., Goto, S., Sato, K., Fujibuchi, W., Bono, H., & Kanehisa, M. (1999). KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Research*, *27*(1), 29–34.

Page, D., Costa, V. S., Natarajan, S., Barnard, A., Peissig, P., & Caldwell, M. (2012). Identifying Adverse Drug Events by Relational Learning. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence* (Vol. 2012, p. 790). Toronto, Canada.

Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). *The PageRank Citation Ranking: Bringing Order to the Web*. Stanford InfoLab.

Perianes-Rodriguez, A., Waltman, L., & van Eck, N. J. (2016). Constructing bibliometric networks: A comparison between full and fractional counting. *Journal of Informetrics*, *10*(4), 1178–1195.

Piatetsky-Shapiro, G. (1991). Discovery, Analysis, and Presentation of Strong Rules. In *Knowledge Discovery in Databases* (pp. 229–248).

Podpečan, V., Lavrač, N., Mozetič, I., Kralj Novak, P., Trajkovski, I., Langohr, L., … Motaln, H., et al. (2011). SegMine Workflows for Semantic Microarray Data Analysis in Orange4WS. *BMC Bioinformatics*, *12*(1), 416.

Puzianowska-Kuznicka, M. & Kuznicki, J. (2005). 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.

Ristoski, P. & Paulheim, H. (2014). Feature Selection in Hierarchical Feature Spaces. In S. Džeroski, P. Panov, D. Kocev, & L. Todorovski (Eds.), *International Conference on Discovery Science* (Vol. 8777, pp. 288–300). Lecture Notes in Computer Science. Springer International Publishing.

Robertson, S. E. & Walker, S. (1994). Some Simple Effective Approximations to the 2-Poisson Model for Probabilistic Weighted Retrieval. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 232–241). Springer-Verlag New York, Inc.

Sechidis, K., Tsoumakas, G., & Vlahavas, I. (2011). On the stratification of multi-label data. *Machine Learning and Knowledge Discovery in Databases*, 145–158.

Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., & Eliassi-Rad, T. (2008). Collective Classification in Network Data. *AI Magazine*, *29*(3), 93.

Shaffer, L. G. & Tommerup, N. (2005). *ISCN 2005: An International System for Human Cytogenetic Nomenclature(2005) Recommendations of the International. Standing Committee on Human Cytogenetic Nomenclature*. Karger.

Smyth, G. K. (2005). Limma: Linear Models for Microarray Data. In *Bioinformatics and Computational Biology Solutions Using R and Bioconductor* (pp. 397–420). Springer.

Sotiriou, C., Wirapati, P., Loi, S., Harris, A., Fox, S., Smeds, J., … Haibe-Kains, B., et al. (2006). 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.

Srinivasan, A. (2007). Aleph Manual. Retrieved from http://www.cs.ox.ac.uk/activities/machinelearning/Aleph/.

Storn, R. & Price, K. (1997). Differential Evolution; A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *Journal of Global Optimization*, *11*(4), 341–359.

Sugiyama, K., Tagawa, S., & Toda, M. (1981). Methods for Visual Understanding of Hierarchical System Structures. *IEEE Transactions on Systems, Man, and Cybernetics*, *11*(2), 109–125.

Sun, Y. & Han, J. (2012). *Mining Heterogeneous Information Networks: Principles and Methodologies*. Morgan & Claypool Publishers.

Tan, S. (2006). An Effective Refinement Strategy for KNN Text Classifier. *Expert Systems and Applications*, *30*(2), 290–298.

Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L., & Su, Z. (2008). ArnetMiner: Extraction and Mining of Academic Social Networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 990–998).

Team, R. C. (2012). *R: A Language and Environment for Statistical Computing. Vienna, Austria: R Foundation for Statistical Computing.* The R project for statistical computing.

Tipney, H. & Hunter, L. (2010). An Introduction to Effective Use of Enrichment Analysis Software. *Human Genomics*, *4*(3), 1.

Tong, H., Faloutsos, C., & Pan, J.-Y. (2006). Fast Random Walk with Restart and Its Applications. In *Proceedings of the Sixth International Conference on Data Mining* (pp. 613–622). Washington, DC, USA.

Trajkovski, I., Lavrač, N., & Tolar, J. (2008). SEGS: Search for Enriched Gene Sets in Microarray Data. *Journal of Biomedical Informatics*, *41*(4), 588–601.

Trajkovski, I., Železný, F., Lavrač, N., & Tolar, J. (2008). Learning Relational Descriptions of Differentially Expressed Gene Groups. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, *38*(1), 16–25.

Tufte, E. R. (1986). *The Visual Display of Quantitative Information.* Cheshire, CT, USA: Graphics Press.

Vanunu, O., Magger, O., Ruppin, E., Shlomi, T., & Sharan, R. (2010). Associating Genes and Protein Complexes with Disease Via Network Propagation. *PLoS Computational Biology*, *6*(1).

Vavpetič, A. (2016). *Semantic Subgroup Discovery* (Doctoral dissertation, Jozef Stefan International Postgraduate School).

Vavpetič, A. & Lavrač, N. (2013). Semantic Subgroup Discovery Systems and Workflows in The SDM–toolkit. *The Computer Journal*, *56*(3), 304–320.

Vavpetič, A., Novak, P. K., Grčar, M., Mozetič, I., & Lavrač, N. (2013). Semantic Data Mining of Financial News Articles. In *Proceedings of Sixteenth International Conference on Discovery Science (DS 2013)* (Vol. 8140, pp. 294–307). Lecture Notes in Computer Science. Singapore.

Vavpetič, A., Novak, P. K., & Lavrač, N. (2013). Analysing Financial Vocabulary Using a New Semantic Subgroup Discovery System Hedwig. In *5th Jožef Stefan International Postgraduate School Students Conference* (pp. 219–229).

Vavpetič, A., Podpečan, V., & Lavrač, N. (2014). Semantic Subgroup Explanations. *Journal of Intelligent Information Systems*, *42*(2), 233–254.

Walt, S. v. d., Colbert, S. C., & Varoquaux, G. (2011). The Numpy Array: A Structure for Efficient Numerical Computation. *Computing in Science & Engineering*, *13*(2), 22–30.

Wrobel, S. (1997). 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)* (pp. 78–87). Springer.

Wu, H.-M., Tien, Y.-J., & Chen, C.-H. (2010). GAP: A Graphical Environment for Matrix Visualization and Cluster Analysis. *Computational Statistics and Data Analysis*, *54*(3), 767–778.

Wu, Y., Cao, N., Archambault, D., Shen, Q., Qu, H., & Cui, W. (2017). Evaluation of graph sampling: A visualization perspective. *IEEE transactions on visualization and computer graphics*, *23*(1), 401–410.

Xing, W. & Ghorbani, A. (2004). Weighted PageRank Algorithm. In *2nd Annual Conference on Communication Networks and Services Research* (pp. 305–314). IEEE.

Žáková, M., Železny, F., Garcia-Sedano, J. A., Tissot, C. M., Lavrač, N., Kremen, P., & Molina, J. (2006). Relational Data Mining Applied to Virtual Engineering of Product Designs. In *International Conference on Inductive Logic Programming* (pp. 439–453). Springer.

Zhou, D., Bousquet, O., Lal, T. N., Weston, J., & Schölkopf, B. (2004). Learning with Local and Global Consistency. *Advances in Neural Information Processing Systems*, *16*(16), 321–328.

Zhou, T., Ren, J., Medo, M., & Zhang, Y.-C. (2007). Bipartite network projection and personal recommendation. *Physical Review E*, *76*(4), 046115.

Zhu, X., Ghahramani, Z., Lafferty, J., et al. (2003). Semi-supervised Learning Using Gaussian Fields and Harmonic Functions. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)* (Vol. 3, pp. 912–919).

zur Hausen, H. (2009). The Search for Infectious Causes of Human Cancers: Where and Why. *Virology*, *392*(1), 1–10.

Bibliography

Publications Related to the Thesis

Journal Articles

Adhikari, P. R., Vavpetič, A., Kralj, J., Lavrač, N., & Hollmén, J. (2016). Explaining Mixture Models through Semantic Pattern Mining and Banded Matrix Visualization. *Machine Learning*, *105*(1), 3–39.

Kralj, J., Robnik-Šikonja, M., & Lavrač, N. (2017). HINMine: Heterogeneous Information Network Mining with Information Retrieval Heuristics. *Journal of Intelligent Information Systems*, 1–33.

Kralj, J., Robnik-Šikonja, M., & Lavrač, N. (n.d.). NetSDM: Network Filtering for Semantic Data Mining. *(submitted).*

Conference Papers

Adhikari, P. R., Vavpetič, A., Kralj, J., Lavrač, N., & Hollmén, J. (2014). Explaining Mixture Models through Semantic Pattern Mining and Banded Matrix Visualization. In *International Conference on Discovery Science* (pp. 1–12). Springer.

Kralj, J., Valmarska, A., Robnik-Šikonja, M., & Lavrač, N. (2015). Mining Text Enriched Heterogeneous Citation Networks. In *Proceedings of the 19th Pacific-Asia Conference on Knowledge Discovery and Data Mining* (pp. 672–683).

Kralj, J., Robnik-Šikonja, M., & Lavrač, N. (2015b). Heterogeneous Network Decomposition and Weighting with Text Mining Heuristics. In *International Workshop on New Frontiers in Mining Complex Patterns* (pp. 194–208). Springer.

Kralj, J., Vavpetič, A., Dumontier, M., & Lavrač, N. (2016). Network Ranking Assisted Semantic Data Mining. In *International Conference on Bioinformatics and Biomedical Engineering* (pp. 752–764). Springer.

Book Chapter

Kralj, J., Valmarska, A., Grčar, M., Robnik-Šikonja, M., & Lavrač, N. (2016). Analysis of Text-enriched Heterogeneous Information Networks. In N. Japkowicz & J. Stefanowski (Eds.), *Big Data Analysis: New Algorithms for a New Society* (pp. 115–139). Springer.

Biography

Jan Kralj was born on December 6, 1988 in Jesenice, Slovenia. He finished primary school in Bled and secondary school in Kranj. In 2007, he started his studies at the Faculty of Mathematics and Physics at the University of Ljubljana. In 2010, he defended his BSc thesis entitled "Grigorčikova Grupa in Burnsideov Problem" under the supervision of Asst. Prof. Dr. Aleš Vavpetič, and in 2013, he defended his MSc thesis entitled "The Arnoldi Algorithm for the Generalized Eigenvalue Problem" under the supervision of Prof. Dr. Bor Plestenjak.

In 2013, he was accepted to the position of junior researcher at the Jožef Stefan Institute in Ljubljana, Slovenia, under the supervision of Prof. Dr. Nada Lavrač at the Department of Knowledge Technologies, Jožef Stefan Institute, Ljubljana, Slovenia. In the same year he started his graduate studies at the Jožef Stefan International Postgraduate School. He enrolled in the PhD programme "Information and Communication Technologies" under the supervision of Prof. Dr. Nada Lavrač.

During his work at the Jožef Stefan Institute he collaborated on the EU-funded projects ConCreTe (Concept Creation Technology), WHIM (The What-If-Machine) and HBP (The Human Brain Project). He also served as a member of the International Postgraduate School student council and was a co-organizer of the 9th Jožef Stefan International Postgraduate School Students' Conference.

His research covers the field of data mining with a focus on data mining algorithms for heterogeneous information network, as well as semantic data mining algorithms. Specifically, his work focuses on joining network analysis techniques with existing data mining approaches to develop new efficient algorithms for data analysis. He presented his work in a published journal paper and at international conferences and workshops.