J Intell Inf Syst (2018) 50:29–61
DOI 10.1007/s10844-017-0444-9

HINMINE: heterogeneous information network mining with information retrieval heuristics

Jan Kralj[1,2] · **Marko Robnik-Šikonja**[3] · **Nada Lavrač**[1,2]

Received: 1 July 2016 / Revised: 8 November 2016 / Accepted: 13 January 2017 /
Published online: 28 January 2017

Abstract The paper presents an 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 information retrieval. The methods assign larger weights to nodes which are more informative and characteristic for a specific 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 in propositionalization. The new methodology is tested on three data sets with different properties. For each data set, we perform a series of experiments

✉ Jan Kralj
jan.kralj@ijs.si

Marko Robnik-Šikonja
marko.robnik@fri.uni-lj.si

Nada Lavrač
nada.lavrac@ijs.si

[1] Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia

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

[3] Faculty of Computer and Information Science, University of Ljubljana, Večna pot 113, 1000 Ljubljana, Slovenia

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. Our results show that HINMINE, using different network decomposition methods, can significantly improve the performance of the resulting classifiers, and also that using a modified label propagation algorithm is beneficial when the data set is imbalanced.

Keywords Network analysis · Heterogeneous information networks · Network decomposition · Personalized PageRank · Information retrieval · Text mining heuristics · Centroid classifier · SVM · label propagation · Imbalanced data

1 Introduction

The field of *network analysis* is well established and exists as an independent research discipline since the late seventies (Zachary 1977) and early eighties (Burt and Minor 1983). In recent years, analysis of *heterogeneous information networks* (Sun and Han 2012) has gained popularity. In contrast to standard (homogeneous) information networks, heterogeneous networks describe heterogeneous types of entities and different types of relations between them. One of the well studied problems is mining of *bibliographic information networks*, connecting authors of scientific papers with their papers. Examples of bibliographic information networks include the DBLP network explored by Sun and Han (2012), the network of video lectures, their authors and viewers examined by Grčar et al. (2013), and the ACM citation network examined in this work. Another example of heterogeneous networks are *biological networks*, which contain entity types such as species, genes, Gene Ontology annotations (Consortium 2000), proteins, metabolites, etc. There are diverse types of links between such mixed biological entities; for example, genes can belong to species, encode proteins, be annotated by an ontology annotation, and so on.

This paper addresses the task of mining heterogeneous information networks using an approach where network heterogeneity is handled through network decomposition. Following the work of Grčar et al. (2013), network decomposition results in several homogeneous networks whose links are derived from the original network, where the links are weighed with the number of intermediary nodes connecting two nodes. For example, in a biological network, two proteins are connected if they participate in the same biological process. In the approach by Grčar et al. (2013), after the individual homogeneous networks are constructed, a propositionalization approach was applied to transform individual homogeneous networks into feature vectors (one vector for every node of a homogeneous network) and concatenating these vectors into a single feature vector for nodes of the given node type. The transformation into feature vectors is named *propositionalization* in the rest of this work. In a classification setting, the transformed feature vector representation allows for using standard classifiers such as the centroid and SVM classifier. In our previous work (Kralj et al. 2015), the network decomposition approach using propositionalization was already applied to a large heterogeneous network of scientific papers from the field of psychology. The conclusion in the precious work was that while the approach is effective, the potential drawback of the method is that it becomes computationally very demanding for large network.

Note that after network decomposition, as an alternative to classification using propositionalization, classification of network nodes of individual homogeneous networks could be performed also through label propagation (Zhou et al. 2004). This approach is much less computationally demanding than the propositionalization approach as it only requires one vector calculation for the classification (as opposed to one per node in the propositionalization approach). This work explores also this alternative approach. It introduces of a new methodology, named HINMINE, which advances the network decomposition methodology of Grčar et al. (2013) by applying variants of text mining heuristics to improve the network decomposition step, and by introducing classification through label propagation as an alternative to homogeneous network propositionalization. The main novelty of the work is the proposal of two improvements to the existing methodology: 1) several new heuristics for node weighting used in the decomposition step of homogeneous network construction are proposed, which are inspired by word weighting heuristics used in text mining and information retrieval, and 2) a new variant of label propagation that ensures improved classification performance on imbalanced data sets. We evaluated the new methodology on three data sets: the E-commerce data set, IMDB data set and the ACM paper-author data set.

The paper is structured as follows. Section 2 describes the related work. Section 3 presents the two-stage methodology for classification in heterogeneous networks. We first present a method for constructing homogeneous networks from heterogeneous networks and then two methods for classification of nodes in a network: a network propositionalization technique and a label propagation algorithm. Section 4 presents improvements to the network decomposition and label propagation algorithms. We first introduce a variant of the label propagation algorithm which improves performance on imbalanced data sets. The homogeneous network construction is then adapted using different node weighting heuristics. Section 5 presents the application of the methodology on three data sets: a data set of customers linked to products they purchased, a data set of scientific papers and paper authors, and a data set of movies and actors appearing in the movies. The empirical analysis examines three aspects of HINMINE. First, we examine the effect of using different classifiers in the final step of classification via propositionalization. Second, we test different heuristics for homogeneous network construction. Third, we analyze the improved label propagation method for imbalanced data sets. Section 6 concludes the paper and presents the plans for further work.

2 Related work

The work presented in this paper is connected to and contributes to two main related research areas, network analysis and text mining, briefly reviewed below.

2.1 Network analysis

In network analysis, instances are connected in a network of connections. The instances may all be of the same type (in which case a network is *homogeneous*), or they can belong to one of several different types (the network is then *heterogeneous*). There are several data analysis tasks that are addressed in the field of network analysis. In our work, we are primarily focused on ranking and classification in networks, particularly in heterogeneous networks.

In ranking, methods like Hubs and Authorities (HITS) (Kleinberg 1999), PageRank (Page et al. 1999), SimRank (Jeh and Widom 2002) and diffusion kernels (Kondor and Lafferty 2002), an authority score is propagated via network edges to discover high ranking nodes in the network. Sun and Han (2012) introduced the concept of *authority* ranking for heterogeneous networks with two node types (bipartite networks) to simultaneously rank nodes of both types. Sun et al. (2009) addressed authority ranking of all nodes types in heterogeneous networks with a star network schema, while Grčar et al. (2013) applied the Personalized PageRank algorithm to find feature vectors for nodes of certain type. This final approach is of particular interest to us as it introduces two concepts that are also used in our approach: decomposition of heterogeneous network into several homogeneous networks, and network propositionalization, a process in which feature nodes are calculated for each node in a homogeneous network.

In network classification, a typical task is to find class labels for some of the nodes in the network using known class labels of the remaining network nodes. Several classification methods in a network setting were proposed in the literature. Sen et al. (2008) presented four network classification algorithms (this approach is referred to as *collective classification*) while de Sousa et al. (2013) describe 5 algorithms used for semi supervised learning which is based on constructing a network of nodes and using it to classify the nodes. One of the approaches used is propagation of labels in the network, a concept used by Zhou et al. (2004) and Vanunu et al. (2010). Other possibilities include Gaussian Random Fields (Zhu et al. 2003) and its modification Robust Multi-Class Graph Transduction (Liu and Chang 2009), Laplacian support vector machines and Laplacian Regularized Least Squares (Belkin et al. 2006). The collective classification algorithms and the label propagation are both designed to work on homogeneous information networks (where all nodes are of the same type), while we work on heterogeneous networks. However, in our work, we used the network decomposition approach from Grčar et al. (2013) to construct homogeneous networks out of the heterogeneous network. We were then able to use label propagation algorithms on the homogeneous networks.

Improved methods have been proposed in the literature that allow for classification of nodes in heterogeneous networks. Hwang and Kuang (2010) expand the idea of label propagation used by 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 et al. (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 the 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 highly complex 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 belonging to a given class, most likely belong to the same class. This idea is implemented in the RankClass framework for classification in heterogeneous information networks.

An alternative approach to classification in heterogeneous networks is classification of network nodes through propositionalization, developed by Grčar et al. (2013), where a heterogeneous network is decomposed into several homogeneous networks. The homogeneous networks are then used to create feature vectors corresponding to nodes in the network. The feature vectors are then classified by SVM (Manevitz and Yousef 2002; Kwok 1998; D'Orazio et al. 2014), *k*-nn (Tan 2006) or a centroid classifier (Han and Karypis 2000) to predict class labels of the nodes. The network propositionalization approach was also used in our previous work (Kralj et al. 2015). Our research builds on this approach and extends it in several directions.

2.2 Text mining

Our work is also related to text mining, specifically bag-of-words vector construction. A bag-of-words (BoW) vector is a numeric vector calculated from a text document that belongs to a corpus. Each unique word (or word lemma, or word *n*-gram), called *term*, in a document is assigned to one vector component, resulting in a high-dimensional vector that represents the original document. It is important to correctly set weights of terms. The simplest way is to use term frequency, i.e. the number of times a term appears in a document. However, this simple method is rarely used, as the term-frequency inverse-document-frequency (tf-idf) weighting introduced in (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 into account labels of documents, have been proposed, including the χ^2, information gain and gain ratio (Debole and Sebastiani 2004), Δ-idf (Martineau and Finin 2009), and relevance frequency (Lan et al. 2009). These weighting schemes are only 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 documents of all target classes and therefore poorly discriminate between the classes. These term weighting schemes evaluates the importance of a given term in the entire corpus of documents, while the Okapi BM25 function (Robertson and Walker 1994) evaluates the importance of a term specifically for each document. This function gives smaller weights to terms that appear in long (compared to an average document length) documents.

2.3 Relation to own published work

The work presented in this paper is an extension of our paper entitled Heterogeneous Network Decomposition and Weighting with Text Mining Heuristics, presented at the workshop New Frontiers in Mining Complex Patterns at the ECML 2015 Conference in Porto. We significantly improved the existing work by introducing two new weighting schemes. We evaluated the approaches on two additional data sets. One of the newly introduced data sets is substantially larger than the previously used single data set and the other contains a larger number of target classes. Given the new data, the experimental section is substantially extended, and offers new insights into the results of the experiments.

3 Background and motivation

This work is based on existing methods for heterogeneous network mining, which we extend with text mining inspired heuristics. This section first presents the methodological background and then motivates our improvements, while the actual approach proposed in this paper is described in Section 4.

3.1 Methodological background

This section addresses the problem of mining heterogeneous information networks (Sun and Han 2012), in which a certain type of nodes (called the target type) is labeled. The adopted approach to mining class labeled heterogeneous information networks, originally proposed by Grčar et al. (2013), consists of the following 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 containing only the nodes of the original network of a given target node type. In each homogeneous network two nodes are connected if they share a particular direct or indirect link in the original heterogeneous network. Take an example network originally presented by Grčar et al. (2013). The network contains two types of nodes, *Papers* and *Authors*, and two edge types, *Cites* (linking papers to papers) and *AuthorOf* (linking authors to papers). From it 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 Fig. 1). The choice of links used in the network decomposition step requires an expert who takes the meaning of links into account and chooses only 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 a network in which two papers are connected if they share a common author, each common author induces one connection between the two papers. This is mathematically equivalent to the weight of the link between 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, the homogeneous networks are used to classify the nodes. We compare two approaches to this task: the label propagation algorithm (Zhou et al. 2004) described in Section 3.1.1 and the network propositionalization approach (Grčar et al. 2013) described in Section 3.1.2.

3.1.1 Label propagation

The label propagation algorithm starts with a network adjacency matrix $M \in \mathbb{R}^{n,n}$ and a class matrix $Y \in \mathbb{R}^{n,|C|}$, where $C = \{c_1, \dots, c_m\}$ 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. The algorithm constructs the matrix $S = D^{-\frac{1}{2}} M D^{-\frac{1}{2}}$, 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). The matrix S is sometimes referred to as the symmetric normalized Laplacian matrix of the network. The algorithm iteratively computes $F(t) = \alpha S F(t-1) + (1-\alpha)Y$ until there are no changes in the matrix $F(t)$. The resulting matrix F^* is used to predict the class labels of all unlabeled nodes in the network. Zhou et al. (2004) show that

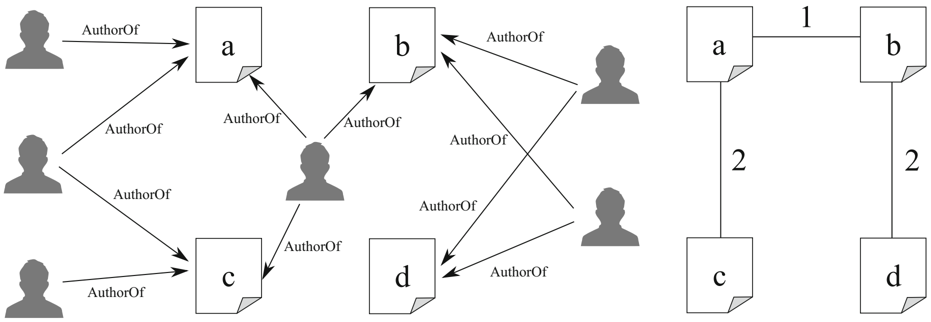

Fig. 1 An example (first used in (Grčar et al. 2013)) of a heterogeneous network (on the left-hand side) and a homogeneous network extracted from it (on the right-hand side). In the homogeneous network, papers are connected if they share a common author. Weights of the edges are equal to the number of authors that contributed to both papers

the iterative process converges to the same matrix F^* regardless of the starting matrix $F(0)$. The limit F^* is shown to be a minimum of a cost function that penalizes distributions F that are too far from the initial matrix Y, and also distributions that contain many connected nodes with very different corresponding values of the matrix F. They also show that the matrix F^* can also be calculated by solving a system of linear equations, as

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

In our work, we 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 $M_1 + M_2 + \cdots + M_k$.

3.1.2 Classification through propositionalization

Network propositionalization is an alternative method for classifying the target nodes in the original heterogeneous network. It calculates feature vectors for each target node in the network using the personalized PageRank (P-PR) algorithm (Page et al. 1999). The personalized PageRank of node v (P-PR$_v$) in a network is defined as the stationary distribution of the position of a random walker who starts the walk in node v and then at each node either selects one of the outgoing connections or jumps back to node v. The probability (denoted p) of continuing the walk is a parameter of the personalized PageRank algorithm and is usually set to 0.85. Once calculated, the resulting PageRank vectors are normalized according to the Euclidean norm. The resulting vector contains information about the proximity of node v to each of the remaining nodes of the network. We consider the P-PR vectors of a node as a propositionalized feature vector of the node. Because two nodes with similar P-PR vectors will be in proximity of similar nodes a classifier should 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 are optimized using the computationally expensive differential evolution (Storn and 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 decided to follow this approach. After the weights are set, many classifiers, for example SVM classifier (Manevitz and Yousef 2002; Kwok 1998; D'Orazio et al. 2014), k-nn classifier (Tan 2006) or a centroid classifier (Han and Karypis 2000) can be used.

3.2 Motivation for advancing the methodology

We made two significant improvements to the original methodology: we improved the network decomposition as well as the label propagation approach. We present the motivation behind the developed advances below.

3.2.1 Improved network decomposition

In the field of text mining, most formulas for calculating the bag-of-words vector (a complete overview of the formulas used is provided in Section 4.2.1) share the same basic structure for weighting a term t in a document d:

$$w(t, d) = f(t, d) \cdot w(t), \tag{2}$$

where $f(t, d)$ is the frequency of the term t in the 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 scheme, the function w is identically equal to 1, meaning all terms in a document receive equal weight. The term-frequency inverse-document frequency scheme is designed to lower the weights for those terms that appear in a large number of documents. Other weighting schemes aim to lower the weights of terms which appear in all target classes with similar frequency. 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 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 the function w used in text mining schemes and translate the function f in (2) to a network setting.

To transform the function f, we rewrite (2). The value $f(t, d)$ is the number of times t appears in d, meaning that we can understand (2) as the sum of $w(t)$ over all appearances of t in the document d, i.e. if we consider a document to be an ordered tuple $(t_1, t_2, \ldots, t_N)$ of terms, (2) equals to

$$w(t, d) = \sum_{\substack{1 \leq i \leq N \\ t_i = t}} w(t). \tag{3}$$

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

In the network decomposition step of the methodology described in Section 3.1, we explained that the weight of the link between two base nodes (i.e. papers) is equal to the number of intermediate nodes (authors) that are linked to (are authors of) both papers. In other words, we can say that the weight of the link between nodes v and u is

$$w(v, u) = \sum_{\substack{m \in M \\ m \text{ is linked to } v \text{ and } u}} 1, \tag{4}$$

where M is the set of all intermediate nodes.

Equation (4) 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 terms appearing in many documents or in documents from different classes. Similarly, in the network setting, we replace the 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 base nodes from different classes. The result is the following formula for computing the weight of the link between u and v

$$w(v, u) = \sum_{\substack{m \in M \\ m \text{ is linked to} v \text{ and} u}} w(m). \tag{5}$$

The novelty of our work concerns the extension of the existing methodology for classification in heterogeneous networks via network decomposition. By using different functions w in (5) 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 a 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 personalized 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 low idf value) and would not affect the PageRank calculation. Similarly, newly introduced functions w that take class labels into account eliminate 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.

3.2.2 Improved label propagation

The label propagation algorithm, as defined in (Zhou et al. 2004), works by propagating class labels from all members belonging to a certain class. By doing so, the algorithm may over-estimate the importance of larger classes (those with more instances) when data is imbalanced. We propose a modification to the label propagation algorithm which fixes that. For example, the left-hand side of Fig. 2 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

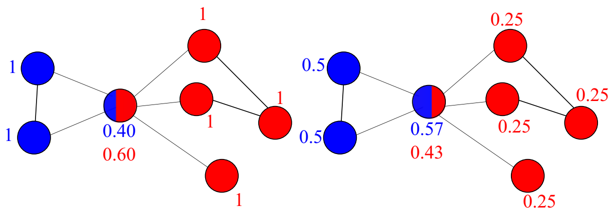

Fig. 2 Results of a label propagation algorithm on an imbalanced data set. If we run the label propagation algorithm as originally defined, each labeled node begin their iteration with a weight of 1 for the class they belong to. In each step of the iteration, every node collects votes from its neighboring 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 a *blue* class consisting of only two nodes

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.

4 The HINMINE methodology

In compliance with the motivation in Section 3.2, this section presents two improvements to the existing methodology, motivated in Section 3.2. It first addresses the means for 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 of label propagation. Figure 3 shows an overview of the proposed approach.

4.1 Imbalanced data sets and label propagation

The reasoning described in Section 3.2.2 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 significantly different. In Section 3.1.1 we described that in each step of the iteration, matrix $F(t)$ is calculated as $F(t) = \alpha S F(t-1) + (1-\alpha)Y$, 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 the 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 (1), 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 larger effect in the iterative calculation of F^*. This prevents votes from more frequent classes to outweigh votes from less frequent classes.

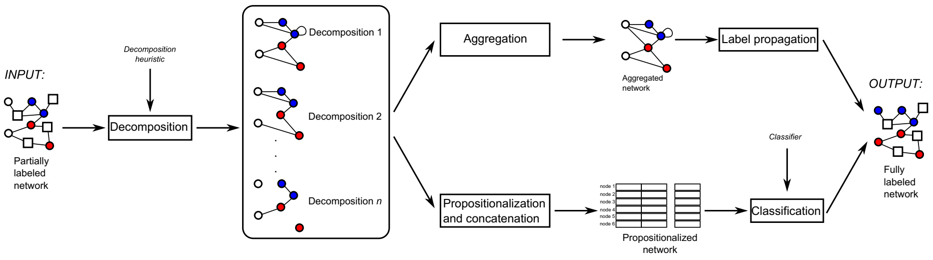

Fig. 3 Overview of the proposed methodology: an input partially labeled heterogeneous network is first decomposed into one or more homogeneous networks. In the second step, these decompositions are merged and used to classify the data using label propagation (*top* branch) or each decomposition is used to calculate feature vectors for each base node in the network (*bottom* branch). In the second case, the feature vectors are then 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

4.2 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".

4.2.1 Term weighting in text mining

In the bag-of-words vector construction one feature vector represents one document in a corpus of documents. In that 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 a term t appears in the document d and D denotes the corpus (a set of 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. We use the notation $t \in d$ to describe that a term t appears in document d. Where used, the term $P(t)$ is the probability that a randomly selected document contains the 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.

Table 1 shows different methods for term weighting. The term frequency (`tf`) weights each term with its frequency in the document. The 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 multiplied by the logarithm of the inverse of the number of documents the term

Table 1 Term weighing schemes and their formulas in text mining

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

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 and 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 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.

The χ^2 (`chi^2`) weighting scheme (Debole and 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 and 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 (`delta-idf`) (Martineau and 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.

4.2.2 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 1 Let us revisit the example from Section 3.1 in which two papers are 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 (Fig. 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

$$\sum_{\substack{m \in T : \\ (m, v) \in E \\ (m, u) \in E}} w(m), \tag{6}$$

where $w(m)$ is the weight assigned to the intermediate node m. The value of $w(m)$ can be calculated in several ways. Table 2 shows the proposed intermediate node weighting heuristics corresponding to term weightings used in document retrieval (Table 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 random base node is connected to the intermediate node node m. We assume that nodes are labeled, each belonging to a class 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 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 authors are 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 `delta-idf` weighting scheme, unlike other term weighting schemes, can assign negative weights to certain terms. Since link weights in graphs are assumed to be positive both by the PageRank and the link propagation algorithm, we must change the weighting scheme before it can be used to construct homogeneous networks. We interpret the original

Table 2 Heuristics for weighting intermediate nodes in the decomposition step of constructing homogeneous networks from the original heterogeneous network

Scheme	Formula
`tf`	1
`if-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`	$\sum_{c \in C}\left(\sum_{c' \in \{c, \neg c\}}\left(\sum_{m' \in \{m, \neg m\}} P(m' \wedge c') \log\left(\frac{P(m' \wedge c')}{P(c')P(m')}\right)\right)\right)$
`gr`	$\sum_{c \in C} \frac{\sum_{c' \in \{c, \neg c\}}\left(\sum_{m' \in \{m, \neg m\}} P(m' \wedge c') \log\left(\frac{P(m' \wedge c')}{P(c')P(m')}\right)\right)}{-\sum c' \in \{c, \neg c\} P(c') \log P(c')}$
`delta-idf`	$\sum_{c \in C}\left\lvert \log \frac{\lvert c\rvert}{\lvert\{b' \in B : b' \in c \wedge (b', m) \in E\}\rvert} - \log \frac{\lvert\neg c\rvert}{\lvert\{b' \in B : b' \notin c \wedge (b', m) \notin E\}\rvert}\right\rvert.$
`rf`	$\sum_{c \in C}\left(\log\left(2 + \frac{\lvert\{b' \in B : b' \in c \wedge (b', m) \in E\}\rvert}{\lvert\{b' \in B : b' \notin c \wedge (b', m) \notin E\}\rvert}\right)\right)$
`bm25`	$\log\left(\frac{\lvert B\rvert}{\lvert\{b \in B : (b,m) \in E\}\rvert}\right) \cdot \frac{k+1}{1+k \cdot \left(1-b+b \cdot \frac{\deg(u)}{\text{avgdeg}}\right)}$

Table 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`	Given a term, measures how dependent the class label of a document is to the appearance of the given word in the document.	Given a intermediate node, measures how dependent the class label of a base node is to the existence of a . 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 word 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.
`delta`	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 other classes.	Measures the difference between the `tf-idf` value of a intermediate node when only observing the base nodes of one class compared to observing the 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 other classes.	Measures the ratio between the `tf-idf` value of a intermediate node when only observing the base nodes of one class compared to observing the 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.

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.

We also change the `bm25` scheme. The other weighting schemes are in the form $f(t, D) \cdot g(t, C)$, where $f(t, D)$ is the frequency of the term t in a document D, and $g(t, C)$ 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 2 depends on the choice of the base node u.

Example 2 Figure 4 shows the construction of a heterogeneous network from Fig. 1, using the χ^2 heuristic. The weight of the central author m is calculated as the sum over both classes as

$$\frac{(P(m \wedge c)P(\neg m \wedge \neg c) - P(m, \neg c)P(\neg m, c))^2}{P(m)P(c)P(\neg m)P(\neg c)}. \tag{7}$$

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 first summand of expression (7) 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 $P(m \wedge c) = P(\neg m \wedge \neg c) = P(m, \neg c) = P(\neg m \wedge c) = \frac{1}{4}$, we see that the second summand of expression (7) is 0 and the total weight of author m is $\frac{1}{3}$.

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.

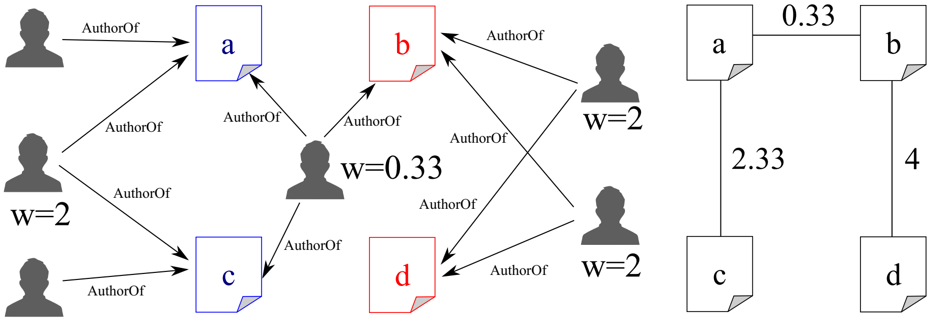

Fig. 4 The construction of a homogeneous network from the toy network in Fig. 1 using the χ^2 heuristic. The *blue* color denotes that the paper belongs to class 1 and the *red* color denotes class 2

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.

5 Experimental setting and results

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

5.1 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 balanced 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 representatives.

5.1.1 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*. 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.

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 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

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

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.

5.1.2 ACM papers data set

The second data set is a collection of papers first used in (Tang et al. 2008). The data set is freely available on the aminer website[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, 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 narrowed our data set to only include conference papers with authors which wrote at least two papers in order to eliminate isolated nodes. 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 of papers in the network. With this large network, we tested the scalability and limitations of our approach.

5.1.3 IMDB data set

The third data set was first published in (Cantador et al. 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 the genre label assigned to each movie.

5.2 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 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 3. In the simplified case described in Fig. 5, we construct one homogeneous network from the original heterogeneous network. However, in the following step, if we are classifying using label propagation, we do not need to aggregate

[2]https://aminer.org/citation

[3]http://grouplens.org/datasets/hetrec-2011/

[4]https://www.imdb.com

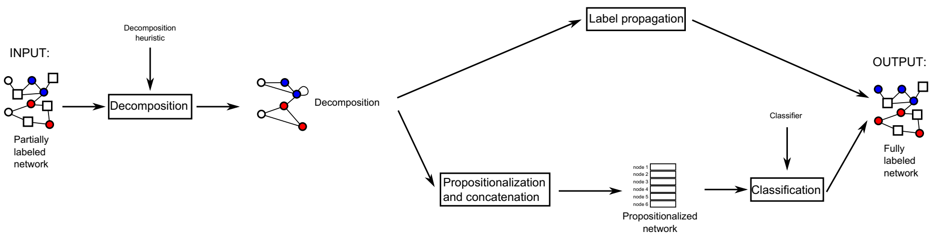

Fig. 5 An overview of the methodology described by Fig. 3 when we only construct one homogeneous network from the heterogeneous network

several networks and can directly perform label propagation to classify the nodes. If we are using 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 the goal of this work is centered around analyzing the effects of using different decomposition heuristics for homogeneous network construction.

5.3 Experiments on E-commerce data set

We performed four sets of experiments on the E-commerce data sets. In this section, we first describe the experiments and then discuss the results.

5.3.1 Experiment description

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 good 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 PPR 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. The tests were performed on feature vectors extracted from all four homogeneous networks. We randomly sampled 3,000 network nodes to train all 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.

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 use the same weights for this set of experiments. The second tested classifier is the label propagation classifier as defined in (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 3.1.1. 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.

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 3.1.2, 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 they are trained on a large data set. This motivated us to train the classifiers on 90 % of the data set and to test their performance on the remaining 10 %.

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 level 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 label propagation 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, and is defined as follows:

$$\frac{\frac{|\{\text{Correctly classified male customers}\}|}{|\{\text{All male customers}\}|} + \frac{|\{\text{Correctly classified female customers}\}|}{|\{\text{All female customers}\}|}}{2}. \tag{8}$$

5.3.2 *Experiment results*

The result of first set of experiments, shown in Table 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 in (Grčar et al. 2013) which opted for an efficient centroid classifier as a default classifier in all settings, our research shows that the optimal classifier for the methodology depends on the characteristics of the data set.

The results of the second set of experiments are shown in Tables 5, 6 and 7. When comparing the results of the two label propagation approaches the results show that label propagation with adjusted starting matrix has 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

Table 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 standards SVM classifier, while SVM_2 marks the classifier using balanced instance weights. Best results are shown in bold

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** %

and even more in the case of B- and C-level networks. This result confirms the intuition explained in Section 3.1.1 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 outperform 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 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 network 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 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

Table 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. Best results are shown in bold

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 %
`delta`	**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 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. Best results are shown in bold

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 %
`delta`	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 %

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 `delta` and `tf-idf` heuristics perform best.

Figure 6 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 sets 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

Table 7 Performance of the balanced 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. Best results are shown in bold

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 %
`delta`	**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 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. Best results are shown in bold

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

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 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.

5.4 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.

5.4.1 Experiment description

As mentioned, calculating the PPR vectors of all base nodes in the network is both time consuming (using our efficient speed-adapted algorithms (Kralj et al. 2015) that compute approximately one PPR vector per second we would need 3–4 days to calculate the vectors for each weighting scheme) and memory intensive (each PPR vector has 320,339

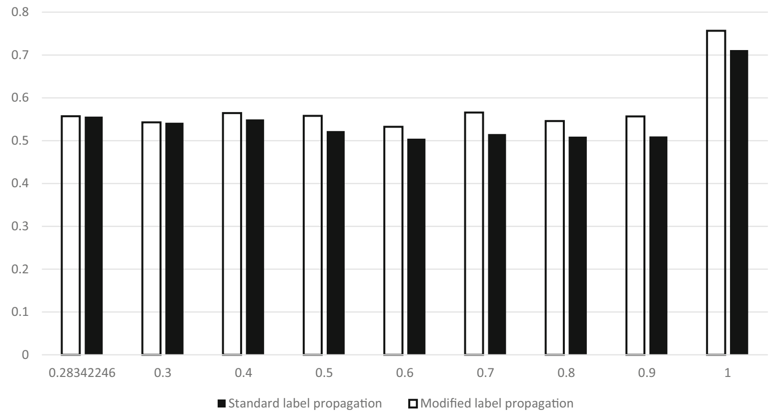

Fig. 6 Performance of the standard and modified label propagation classifiers at different levels of class imbalance. The x axis shows the proportion of female customers chosen (at 28 %, the data set is perfectly balanced) and the y axis shows the classification accuracy. The height of each column is the balanced accuracy score achieved by the modified or standard label propagation classifier

dimensions and takes 30 megabytes of memory, so 10,000 vectors occupy approximately 30 gigabytes of memory).

The second problem was that training times for most machine learning algorithms were 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 slow to run on the full data set. To execute the algorithms in a reasonable time we used the following scenario:

- we sparsified the PPR vectors by cutting off low ranking nodes. The motivation behind this can be seen in Fig. 7 which shows that only a small number of papers in the network has PPR values above 10^{-5}, so discarding the rest of the values should have little impact on calculation. 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 PPR 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.

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.

5.4.2 *Experiment results*

Tables 9, 10 and 11 show the results of the experiments on the ACM data set. We show the accuracy and its standard deviation across 5 random resamples. The results indicate that our decision to cutoff the near-zero values of PageRank vectors was well justified as there is no significant difference in performance due to cutoff values.

Table 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

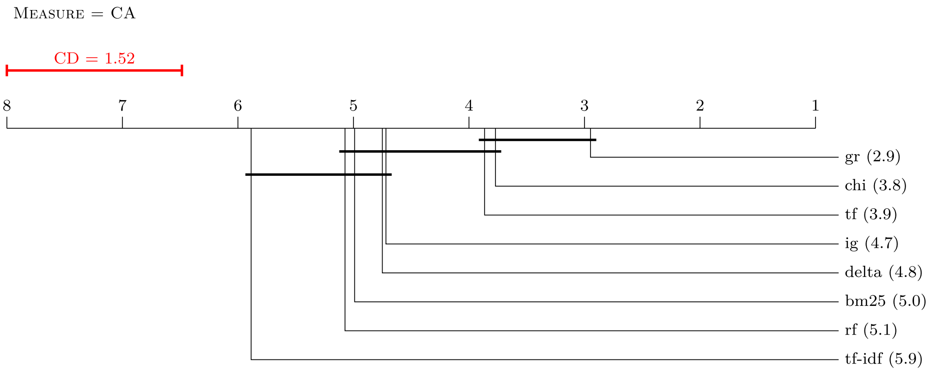

Fig. 7 Critical distance diagram of the weighting schemes and their performance on the ACM data set. The critical distance is calculated at a p value of 0.05

Table 9 Classification accuracy (in %) of the k-nearest neighbor algorithms on the ACM data set. Best results are shown in bold

top n	gr	tf	ig	chi	rf	delta	idf	bm25
$k = 1$								
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	**32.36 ± 0.34**
2	39.30 ± 5.29	**51.43 ± 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	**53.65 ± 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	**62.27 ± 0.51**	46.52 ± 0.50	46.68 ± 0.41	52.61 ± 0.33
$k = 3$								
1	29.70 ± 2.39	**34.57 ± 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	**54.77 ± 0.14**	35.00 ± 7.24	49.26 ± 0.79	**54.93 ± 0.58**	36.64 ± 7.43	32.99 ± 3.30	46.02 ± 4.92
3	**59.64 ± 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	**69.87 ± 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
$k = 5$								
1	29.92 ± 2.70	**34.07 ± 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	**56.23 ± 0.55**	39.87 ± 8.58	50.48 ± 1.09	**55.64 ± 0.91**	45.27 ± 7.50	36.82 ± 4.95	54.71 ± 1.19
3	**61.06 ± 1.47**	**61.02 ± 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	**76.81 ± 4.26**	72.02 ± 0.68	65.14 ± 9.31	76.62 ± 1.49	68.30 ± 0.99
				$k = 10$				
1	31.20 ± 2.81	**36.28 ± 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	**56.78 ± 0.64**	39.44 ± 7.17	51.50 ± 0.97	**56.40 ± 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	**65.16 ± 1.94**	52.92 ± 8.49	58.94 ± 6.81
5	75.47 ± 1.75	76.82 ± 1.74	77.96 ± 5.85	**79.64 ± 4.56**	76.35 ± 2.91	79.36 ± 4.68	75.90 ± 4.04	69.64 ± 2.11

The classifier was trained on PPR 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

Table 10 Classification accuracy (in %) of the k-nearest neighbor algorithms on the ACM data set. Best results are shown in bold

top n	gr	tf	ig	chi	rf	delta	idf	bm25
$k = 1$								
1	**31.52** ± **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	**42.74** ± **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	**46.59** ± **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	**55.82** ± **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
$k = 3$								
1	**30.11** ± **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	**54.37** ± **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	**61.62** ± **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	**70.04** ± **3.76**	64.55 ± 2.81	68.69 ± 0.95	63.84 ± 4.25	56.50 ± 0.55	74.09 ± 1.11
$k = 5$								
1	**34.76** ± **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	**55.02** ± **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	**63.27** ± **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	**71.48** ± **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
$k = 10$								
1	**35.81** ± **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	**56.64** ± **0.79**	39.11 ± 5.74	**56.64** ± **0.65**	53.25 ± 4.95	39.20 ± 8.33	52.03 ± 0.47	52.54 ± 4.08	47.94 ± 4.73
3	**63.20** ± **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	**75.63** ± **0.46**	77.23 ± 3.17	77.51 ± 1.12

The classifier was trained on PPR 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

Table 11 Classification accuracy (in %) of the k-nearest neighbor algorithms on the ACM data set. Best results are shown in bold

top n	gr	tf	ig	chi	rf	delta	idf	bm25
$k = 1$								
1	15.58 ± 2.57	21.83 ± 3.27	19.39 ± 0.72	**22.93 ± 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	**38.88 ± 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	**34.34 ± 4.84**	30.70 ± 1.29	28.54 ± 0.54	28.21 ± 0.84	27.99 ± 0.39
5	42.73 ± 0.54	**45.48 ± 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
$k = 3$								
1	17.65 ± 5.70	**23.43 ± 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	**50.37 ± 0.59**	33.56 ± 2.96	33.96 ± 0.73	36.59 ± 0.97	31.42 ± 0.79
3	40.87 ± 9.50	**54.21 ± 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	**68.14 ± 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
$k = 5$								
1	22.67 ± 8.15	23.43 ± 3.96	24.99 ± 3.92	**33.50 ± 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	**50.80 ± 1.07**	31.96 ± 2.26	41.88 ± 5.88	36.08 ± 0.95	31.78 ± 0.81
3	**59.62 ± 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	**73.64 ± 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
$k = 10$								
1	29.04 ± 1.26	22.69 ± 3.72	30.44 ± 1.91	28.15 ± 6.69	18.52 ± 0.93	**32.95 ± 2.38**	21.35 ± 3.35	22.91 ± 5.61
2	**55.92 ± 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	**66.45 ± 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	**78.68 ± 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

The classifier was trained on PPR 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

Table 12 Classification accuracy (in %) for the label propagation algorithm on the ACM data set. Best results are shown in bold

top n	gr	tf	ig	chi	rf	delta	idf	bm25
Balanced label propagation								
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								
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**

The label propagation was run in homogeneous networks, obtained from the heterogeneous network using eight weighting schemes

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 4.1. This result is not surprising. In Section 4.1, we explain 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 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. We use the Friedmann test to evaluate the null hypothesis that all weighing schemes perform equally. We rejected the hypothesis at a p-value of 10^{-8} and performed a post-hoc Nemenyi test to asses 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 Fig. 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 9, 10, and 11, we recommend using the gain ratio (`gr`) weighting scheme to classify papers in the ACM paper-author-paper network.

5.5 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.

5.5.1 Experiment description

We used the IMDB data set to compare several alternative semi-supervised learning approaches described in Section 2. 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 that 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. 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.

5.5.2 Experiment results

Table 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 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 5076 representatives, the second most frequent, “comedy”, contains 3566, 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), however 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 PPR vector calculation.

In Fig. 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 13 Semi-supervised classifiers accuracies (in %) for the IMDB data set. Best results are shown in bold

top n	gr	tf	ig	chi	rf	delta	idf	bm25
Robust Multi-Class Graph Transduction								
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								
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
Balanced label propagation								
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

The classifiers were trained on PPR vectors of nodes in homogeneous networks, obtained from the original heterogeneous network using eight weighting schemes

Table 14 K-NN classifiers accuracies (in %) for the IMDB data set. Best results are shown in bold

top n	gr	tf	ig	chi	rf	delta	idf	bm25
$k = 1$								
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
$k = 5$								
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	chi	rf	delta	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**

The classifiers were using PPR vectors of nodes in homogeneous networks, obtained from the original heterogeneous network using eight weighting schemes

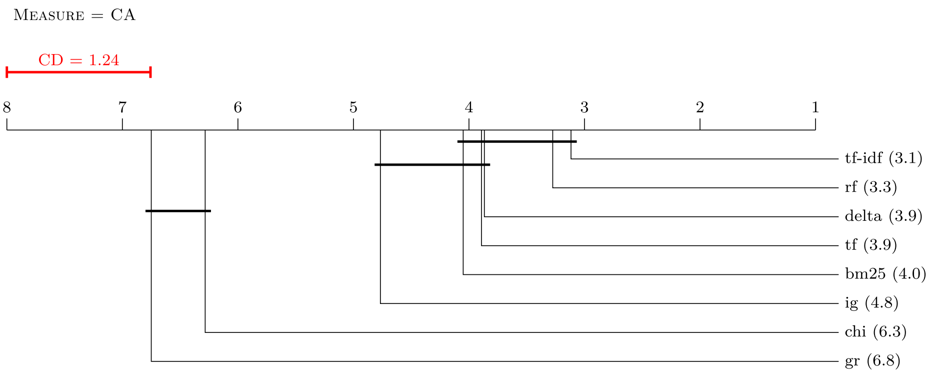

Fig. 8 Critical distance diagram of the weighting schemes and their performance on the IMDB data set. The critical distance is calculated at a p value of 0.05

5.6 Summary of experimental findings

Our experiments show that using different weighting schemes can significantly alter the propositionalization vectors, obtained by following the methodology proposed in (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 out-perform 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 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).

6 Conclusions and further work

We have proposed and implemented several improvements to existing network classification algorithms with a goal of proposing an improved heterogeneous network mining methodology we call HINMINE. The contributions of the paper 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 (`delta` 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 smaller data set with 20 classes, and the third was very large with 11 classes. The results show that 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 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.

In future work, we will work on in-depth analysis of the network construction heuristics and their performance in classifiers applied to homogeneous networks. We plan to design efficient methods for propositionalization of large data sets and decrease the computational load of PageRank calculations by first detecting communities in a network. Such "pre-processing" should reduce the size of networks for PageRank calculations. We plan to further investigate the difference between the standard label propagation algorithm and the modified version, presented in this paper. We will investigate if an optimal balance between the standard initial matrix and the modified matrix (calculated as a convex combination of the two) can outperform both initial matrices.

Acknowledgments This research was supported by the European Commission through the Human Brain Project (Grant number 604102) and three National Research Agency grants: the research programmes Knowledge Technologies (P2-0103), Artificial intelligence and intelligent systems (P2-0209) and project Development and applications of new semantic data mining methods in life sciences (J2-5478). Our thanks goes to Miha Grčar for previous work on this topic, which has inspired the research described in this paper.

References

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*, 2399–2434.

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

Cantador, I., Brusilovsky, P., & Kuflik, T. (2011). 2Nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *Proceedings of the 5th ACM conference on Recommender systems*. New York: ACM. RecSys.

Consortium (2000). Gene ontology: tool for the unification of biology. The gene ontology consortium. *Nature genetics*, *25*(1), 25–29.

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.

Grčar, M., Trdin, N., & Lavrač, N. (2013). A methodology for mining document-enriched heterogeneous information networks. *The Computer Journal*, *56*(3), 321–335.

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.

Hwang, T., & Kuang, R. (2010). A heterogeneous label propagation algorithm for disease gene discovery. In *Proceedings of 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).

Jones, K.S. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, *28*, 11–21.

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

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).

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.

Liu, W., & Chang, S.-F. (2009). Robust multi-class transductive learning with graphs. In *IEEE Conference on Computer Vision and Pattern Recognition, 2009. CVPR 2009* (pp. 381–388): IEEE.

Manevitz, L.M., & Yousef, M. (2002). 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*. San Jose: AAAI Press.

Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). *The PageRank citation ranking: Bringing order to the web*. Technical report: Stanford InfoLab.

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). New York: Springer.

Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., & Eliassi-Rad, T. (2008). Collective classification in network data. *AI magazine*, *29*(3), 93.

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.

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

Sun, Y., Yu, Y., & Han, J. (2009). Ranking-based clustering of heterogeneous information networks with star network schema. In *Proceedings of the 15th ACM SIGKDD I,nternational Conference on Knowledge Discovery and Data Mining* (pp. 797–806).

Tan, S. (2006). An effective refinement strategy for KNN text classifier. *Expert Systems with 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 *KDD'08* (pp. 990–998).

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).

Zachary, W. (1977). An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, *33*, 452–473.

Zhou, D., Bousquet, O., Lal, T.N., Weston, J., & Schölkopf, B. (2004). Learning with local and global consistency. *Advances in N,eural Information Processing Systems*, *16*(16), 321–328.

Zhu, X., Ghahramani, Z., Lafferty, J., et al. (2003). Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, (Vol. 3 pp. 912–919).