Heterogeneous Subgraph Features for Information Networks

Andreas Spitz
Heidelberg University
spitz@informatik.uni-heidelberg.de

Diego Costa
Heidelberg University
costa@informatik.uni-heidelberg.de

Kai Chen
Heidelberg University
chen@informatik.uni-heidelberg.de

Jan Greulich
Heidelberg University
j.greulich@stud.uni-heidelberg.de

Johanna Geiß
Heidelberg University
geiss@informatik.uni-heidelberg.de

Stefan Wiesberg
Heidelberg University
wiesberg@uni-heidelberg.de

Michael Gertz
Heidelberg University
gertz@informatik.uni-heidelberg.de

ABSTRACT

Networks play an increasingly important role in modelling real-world systems due to their utility in representing complex connections. For predictive analyses, the engineering of node features in such networks is of fundamental importance to machine learning applications, where the lack of external information often introduces the need for features that are based purely on network topology. Existing feature extraction approaches have so far focused primarily on networks with just one type of node and thereby disregarded the information contained in the topology of heterogeneous networks, or used domain specific approaches that incorporate node labels based on external knowledge. Here, we generalize the notion of heterogeneity and present an approach for the efficient extraction and representation of *heterogeneous subgraph features*. We evaluate their performance for rank- and label-prediction tasks and explore the implications of feature importance for prominent subgraphs. Our experiments reveal that heterogeneous subgraph features reach the predictive power of manually engineered features that incorporate domain knowledge. Furthermore, we find that heterogeneous subgraph features outperform state-of-the-art neural node embeddings in both tasks and across all data sets.

KEYWORDS

Heterogeneous networks; information networks; node features; feature engineering; graph encodings

ACM Reference Format:
Andreas Spitz, Diego Costa, Kai Chen, Jan Greulich, Johanna Geiß, Stefan Wiesberg, and Michael Gertz. 2018. Heterogeneous Subgraph Features for Information Networks. In *GRADES-NDA'18 : 1st Joint International Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA), June 10–15, 2018, Houston, TX, USA*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3210259.3210266

GRADES-NDA'18 , June 10–15, 2018, Houston, TX, USA

ACM ISBN 978-1-4503-5695-4/18/06…$15.00
https://doi.org/10.1145/3210259.3210266

1 INTRODUCTION

The observation that everything is connected to everything else, which is frequently (mis-) attributed to Renaissance researcher Leonardo da Vinci, describes the recent advances in Information Retrieval and data modelling increasingly well. Networks of entities offer an intuitive representation of complex connected systems by abstracting them as graphs. Despite this apparent simplicity, even the most basic network structures pose rich analytical challenges. However, completely abstracting real networks as a single type of connected nodes is often an oversimplification of the represented system. As a result, *heterogeneous (information) networks* have shifted into the focus of research, which are composed of different types of nodes or edges. Examples include a variety of data from naturally observed biological networks to constructed entity networks and knowledge bases. Recently, such networks have been applied in tasks as diverse as music and movie recommendation [8, 36], multiplex film citation analysis [26], the identification of a molecular basis for human disease [31], the embedding of language networks [29], or the extraction of events from implicit textual networks of named entities [25]. The terminology in the literature is ambiguous since *heterogeneous* may refer to networks that are node-heterogeneous (also called multi-mode), or edge-heterogeneous (also called multiplex or multi-layer). In the following, we focus on node-heterogeneous networks.

Data Mining in heterogeneous information networks leverages the inherently diverse representation to derive insights into the underlying systems [27]. Frequently, node types are used explicitly, e.g., to recommend movies based on user and actor connections [36]. Similar examples include Wikipedia query intent analyses [22], social network analysis [11], and transductive classification [3]. Most prominently, the availability of large scientific publication networks such as DBLP has motivated investigations into the extraction of information from heterogeneous citation networks, for which the distinction between node types has proven to be successful [30].

To support such predictive analyses, it is paramount to extract representative node features from the network, which can be divided into (1) *intrinsic features* that require domain knowledge in the engineering phase and (2) *graph features* that are based on the topological network structure. Since feature engineering is costly and domain knowledge can be difficult to obtain, some emphasis

has been put on extracting node features or embeddings that characterize node connectivity information [7, 9, 20]. Frequently, these approaches rely on random walks, which can be problematic due to the heavily skewed degree distribution of real-world networks [29]. Despite the growing prevalence of heterogeneous networks, many approaches do not include node labels in the model. While they leverage the topological neighbourhood of nodes and some even include (partial) node labels, so far no approach has abstracted the extraction of labelled features to a purely topological level with no domain knowledge. Furthermore, since node embeddings include a dimension reduction, the resulting features are abstract representations that offer no insights into the structure of the data.

Here, we propose *heterogeneous subgraph features* that include both topological information and node labels. They are designed for settings in which domain-specific features cannot be engineered for heterogeneous networks. Unlike neural embeddings, they offer insights into the network structure in diverse settings. We discuss an implementation of the underlying subgraph census algorithm that avoids the problems caused by the skewed degree distributions common to most networks, since the utilized counts of local subgraphs reflect both abundance and sparsity of connectivity.

Contributions. (i) We introduce subgraph features for heterogeneous networks, which cope well with the skewed topology of real-world networks and can replace features that are engineered with domain knowledge when such knowledge is unavailable. **(ii)** We discuss the interpretability of this new feature in contrast to neural embeddings. **(iii)** We provide an efficient implementation of the feature extraction and encoding framework[1]. **(iv)** We demonstrate the effectiveness of the new features against classic features and neural embeddings for several machine learning techniques in two predictive tasks on three structurally diverse networks.

2 RELATED WORK

Our approach touches on concepts in the fields of Data Mining and Network Analysis as we discuss in the following.

Prediction in Information Networks. Information extraction from information networks encompasses a variety of methods, so we focus on the most closely related ones (for an overview, see [27]).

Guo and Liu consider feature generation for music recommendation in heterogeneous graphs based on random walks [8]. In a similar approach for personalized entity recommendation, Yu et al. represent the connectivity between users and items by extracting path-based latent features [36]. Bangcharoensap et al. propose the transductive prediction of node labels based on edge betweenness [3]. The latent space heterogeneous model by Jacob et al. also addresses transductive classification in social networks by transforming the heterogeneous classification problem into multiple homogeneous problems [11]. Ren et al. use graphs of user queries, web pages and Wikipedia concepts for learning user intent [22]. For scientific publication networks, Dong et al. predict the impact of papers based on a network containing authorship, venue, and citations [4]. Ren et al. derive citation recommendations from clustering heterogeneous networks of scientific publications [21]. In an extension of link prediction on homogeneous networks, Sun et al. predict author collaboration in DBLP data by including a temporal aspect [28]. Huang et al. propose meta-structures as a generalization of meta-paths for relevance computations based on user-specified meta-structures as input seeds [10].

While the above works utilize the heterogeneity of the networks, they also require domain specific knowledge. An approach that does not use domain knowledge is given by Fang et al., who predict the semantic proximity of nodes through the extraction of meta-graphs [5]. However, they rely on bipartite symmetries in the network for proximity prediction, which do not work as universal features. In contrast to all above approaches, we focus on extracting subgraph features from general heterogeneous networks that do not require intrinsic symmetries or domain knowledge.

Node Feature Extraction. To identify the role of graph nodes, Henderson et al. rely on structural equivalence, which postulates that nodes with similar neighbourhoods have similar roles [9]. Perozzi et al. learn latent representations for nodes in a social network by applying an approach that is reminiscent of word embeddings in low-dimensional vector spaces to random walks around nodes [20]. Similarly, Grover and Leskovec [7] learn node embeddings by combining different schemes for localized neighbourhood exploration.

The above approaches do not reflect the heterogeneity of node neighbourhoods. Furthermore, methods that are based on random walks suffer from the sparseness of neighbourhoods around low-degree nodes, which is problematic due to the skewed degree distribution of real-world networks. In contrast, we introduce a subgraph-based method that includes local sparseness in the feature.

Subgraph Mining and Encodings. The research on subgraph mining is too extensive to cover here, so we refer to the survey by Jiang et al. for an in-depth overview [13]. An early contribution to subgraph encodings was given by Yan and Han with DFS-codes as canonical representations [34] that were originally designed for graph indexing [35]. More recently, Mason et al. introduced a scheme for encoding node neighbourhoods in cellular networks by enumerating labellings of local neighbourhoods, which they apply to a topological comparison of cellular networks [15]. Due to its geometrically motivated construction, the scheme is limited to spatially embedded networks. In contrast, we introduce an encoding scheme for the representation of heterogeneous subgraphs that includes node labels of arbitrary graphs in a characteristic sequence.

Network Motifs. Conceptually similar to subgraph mining, *network motifs* were introduced by Milo et al. [18], and represent subgraphs that occur significantly more frequently in an observed network than in a comparable network model. As such, motifs rely on a network model for determining significance. The wrong choice of model may entail biased results, which has led to criticism [1] and makes motifs difficult to use in practice. Wernicke provides an efficient algorithm for extracting all motifs of a given size [32] that is implemented in the tool FANMOD [33]. Ribeiro and Silva introduce an algorithm for the extraction of colored network motifs that employs gTries for subgraph enumeration [23].

In practice, the mining of motifs is inherently different from the extraction of subgraph features. Motif discovery requires the enumeration of all *global* subgraph counts of a network and is prohibitively expensive for all but the smallest subgraphs and networks. In contrast, feature extraction requires a *local* enumeration of rooted subgraphs around the target node. As a result, motif extraction algorithms are ill-suited to subgraph feature extraction.

[1] C++ and Python code are available at https://dbs.ifi.uni-heidelberg.de/resources/hsgf/

3 FEATURE MODEL

Let $G = (V, E)$ denote an undirected graph over the set of nodes V that are connected by edges E. We write $vw \in E$ if nodes v and w are connected by an edge. To represent the distinct labels (i.e., types or classes) of nodes in a heterogeneous network, we use a set of node labels L and a function $\lambda : V \to L$ such that $\lambda(v) \in L \; \forall v \in V$. A heterogeneous network is then a labelled graph $G = (V, E, L)$, but we omit L where it is clear from context. To distinguish between networks with different levels of connectivity between nodes that have different labels, we introduce the *label connectivity graph*, in which all nodes with the same label are aggregated into a single node. In Figure 1A we show an example of a heterogeneous publication network and its corresponding label connectivity graph.

We call $G' = (V', E')$ a *subgraph* of G and write $G' \subseteq G$ if $V' \subseteq V$, $E' \subseteq E$ and $v, w \in V'$ for all $vw \in E'$, i.e., if G' is contained in G. The set of *rooted subgraphs* for a root node $v \in V$ is defined as $S(v) = \{G' \subseteq G \,|\, v \in V'\}$, i.e., the set of all subgraphs of G that contain v. To represent the local neighbourhood around a node v in G, we can thus extract a *census* of distinct subgraphs containing v (i.e., we count all subgraphs), as we discuss in the following.

Graph Isomorphism. During the exploration of the neighbourhood of a root node, the nodes of structurally identical subgraphs may be visited in different order. Therefore, the correctness of the subgraph census depends on a matching of identical subgraphs, independent of the visitation order. Formally, assume that we are given two labelled graphs $G = (V, E, L)$ and $G' = (V', E', L)$. We say that G is isomorphic to G' (and write $G \simeq G'$) if there exists a bijection $\phi : V_G \to V_{G'}$ with the two properties: **(i)** $uv \in E$ iff $\phi(u)\phi(v) \in E' \; \forall u, v \in V$. **(ii)** $\lambda(v) = \lambda(\phi(v)) \; \forall v \in V$. Thus, two isomorphic graphs cannot be distinguished unless node ordering is taken into account. The desired feature of a subgraph encoding scheme is thus the ability to distinguish subgraphs up to isomorphism. Unfortunately, it is unknown whether a polynomial solution to the problem exists [2]. However, subgraph features represent the local neighbourhood of nodes and do not need to be arbitrarily large. Therefore, the subgraphs can be limited to a size where the isomorphism problem is manageable with the correct encoding scheme. Furthermore, we find that a low level of encoding collisions does not decrease the quality of the subsequent predictions, and use these observations to introduce an encoding scheme.

3.1 Subgraph Encoding

The most time-consuming aspect of the subgraph census is the isomorphism test that is performed for every discovered subgraph. While this problem has no known polynomial solution, a suitable encoding can be used to solve it for small subgraphs and approximate it for larger subgraphs. Furthermore, an efficient comparison to previously discovered subgraphs is necessary to avoid a quadratic complexity of the comparisons alone. Thus, we design a pseudo-canonical subgraph encoding such that two small subgraphs are isomorphic iff their encodings are identical. Instead of checking two small subgraphs for isomorphism, it is then sufficient to compare their encodings. Here, a hashable encoding enables the use of hashmaps and reduces the complexity for the extraction of a single subgraph occurrence to $O(1)$ for fixed maximum subgraph size. Our encoding is based on labelled subgraph degree sequences.

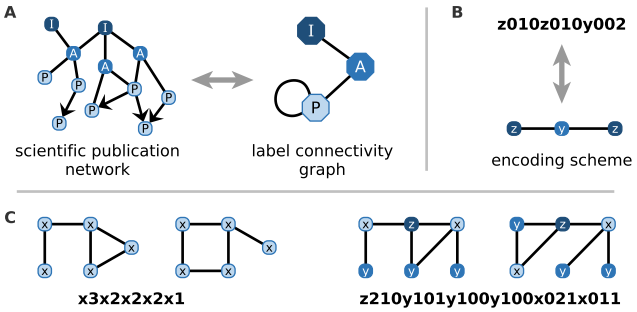

Figure 1: A: Scientific publication network of institutions I, authors A and publications P, with corresponding label connectivity graph. B: Characteristic sequence of a 3-node graph example. C: Two non-isomorphic graphs with a single label sharing the same encoding (left). Two non-isomorphic graphs with three labels and colliding encoding (right).

Characteristic Sequence. Given a graph $H = (V_H, E_H) \subseteq G$, for each vertex $v \in V_H$ we define a sequence $s_v = t_0, t_1, \ldots, t_k$ of $k = |L|$ integers, where $t_0 = \lambda(v)$. For some fixed ordering of labels $l = 1, \ldots, |L|$, each t_l is the number of neighbours of v with label l.

$$t_l := |\{u \in V_H \mid uv \in E_H, \lambda(u) = l\}| \tag{1}$$

Based on the sequences s_v for individual nodes, we define the *characteristic sequence* s_H of H as their concatenation. Thus, let

$$s_H := (s_{v_1}, s_{v_2}, \ldots, s_{v_n}) \tag{2}$$

where n is the number of nodes in the subgraph. The sequences s_{v_i} are sorted in lexicographic order such that $s_{v_1} \geq s_{v_2} \geq \cdots \geq s_{v_n}$.

Example. Consider the set of labels $L = \{x, y, z\}$. The sequence $z010z010y002$ then encodes a graph with three nodes and two edges. The first node has label z and no neighbours with label x, one neighbour with label y, and no neighbours with label z. The second node has identical label and neighbourhood to the first node, while the third node has label y and two neighbours with label z. Overall, the encoding represents a graph that is a path of length three with labels z and y (see Figure 1B). Note that the encoding does not reveal which of the three nodes is the starting node.

Limitations. The above encoding is not unique for graphs of arbitrary size, since some encodings collide for larger subgraphs (see Figure 1C). An analytic derivation of the number of subgraph multiplicities requires the number of graphs with a given degree sequence, which is unknown [16]. Using an enumeration of non-isomorphic labelled graphs, we find that the maximum number of edges that a subgraph may contain to ensure unique encodings is $e_{max} = 5$ for graphs without loops in the label connectivity graph and $e_{max} = 4$ for graphs with loops in the label connectivity graph. In practice, we find that the collisions have no negative impact on the quality of the features since the overall number of collisions is negligible when compared to the number of possible subgraphs. Furthermore, higher values of e_{max} correspond to a higher discriminative power of the features, but also increase the extraction time, which grows roughly exponentially with the size of the subgraph. Thus, e_{max} should be selected as high as possible without impeding the extraction process. In practice, we find that $e_{max} = 5$ is a reasonable value for experiments on real-world data.

3.2 Feature Extraction

Based on the above encoding, we now define heterogeneous subgraph features and discuss their extraction for a given node v. We limit their size by the number of contained edges e_{max}.

Feature Definition. Let $\mathcal{H}$ denote the set of all connected subgraphs of G that contain v and have at most e_{max} edges. Let $R_{\mathcal{H}} \subseteq \mathcal{H}$ denote a representative sub-system of $\mathcal{H}$ with respect to $\simeq$, i.e., every $H \in \mathcal{H}$ is isomorphic to exactly one element in $R_{\mathcal{H}}$. The subgraph census then is a function $sc : V \times R_{\mathcal{H}} \to \mathbb{N}$ with

$$sc(v, H) \mapsto |\{H' \in \mathcal{H} \mid v \in V_{H'} \ \wedge \ H' \simeq H\}| \tag{3}$$

Thus, for each subgraph type, we compute the number of times it can be found in the neighbourhood of the start node v. Using the encoding to replace the isomorphism check, we obtain

$$c(v, H) \mapsto |\{H' \in \mathcal{H} \mid v \in V_{H'} \ \wedge \ s_{H'} = s_H\}| \tag{4}$$

The counts $c(v, H)$ of the encodings of all possible subgraphs H rooted at v then serve as representative features for node v.

Implementation. To design the subgraph extraction algorithm, we first observe that this enumerative task is always computationally expensive [6], which encourages the use of efficient enumeration strategies. Thus, we base our algorithm on four key concepts: **(i)** subgraphs around a given root node are expanded and enumerated incrementally, **(ii)** subgraph encodings allow efficient incremental updates, **(iii)** hashed encodings replace isomorphism tests with constant-time operations, and **(iv)** a node-based enumeration supports by-node parallelization and sampling strategies.

Based on these considerations, we find that a depth-first search approach around the root node performs well if it is adjusted to the enumeration task and network topology. Each time a new node is discovered, it is added to the subgraph and the count of its encoding is updated in a hash map. Hashing the above encoding is trivial since it can be represented as a string. Due to the lexicographic ordering, it is also possible to efficiently update the encoding by adding nodes during the expansion of subgraphs. More involved hashing schemes such as rolling hashes [14] are feasible and allow adding a new node to a previously discovered subgraph without recomputing the hash value of the entire encoding. Once the maximum number of edges per subgraph is reached, backtracking allows the exploration of further subgraphs around the root node. Since the basic approach is straightforward, we omit the algorithm and focus on heuristic optimizations for heterogeneous networks with skewed topology.

Heterogeneous Optimization Heuristic. Due to the heterogeneous structure of the graph, the addition of new nodes to the same adjacent node in an existing subgraph yields identical encodings for each new node with identical label. Thus, we can group neighbouring nodes according to their label and increase the corresponding subgraph counter by the number of adjacent nodes per label, instead of using individual increments. It is therefore sufficient to compute the modified hash value once per label. For graphs with a relatively low number of labels, this decreases the effort from $degree(v)$ to $\min\{degree(v), |L|\}$ hash computations per node. The re-discovery of already known adjacent nodes then has to be handled as a border case, but the entire approach can be implemented efficiently by sorting the adjacency lists of nodes by label.

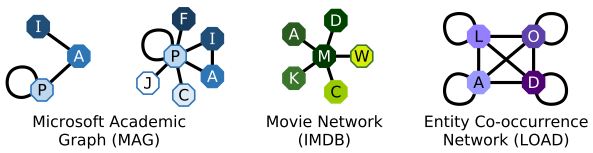

Figure 2: Label connectivity graphs of the evaluation networks. MAG for rank prediction (left) and label prediction (right) with authors A, institutions I, conferences C, journals J, fields F, and papers P. LOAD with locations L, organizations O, actors A, and dates D. IMDB with movies M, actors A, directors D, writers W, composers C, and keywords K.

Topological Optimization Heuristic. A skewed degree distribution is common to most networks, which include some nodes with extraordinarily large degrees called *hubs*. Such hubs play a central role in the computational cost of subgraph enumerations since they **(i)** inflate subgraph counts of adjacent nodes and **(ii)** connect remote regions of the network that share little relation. It is questionable whether subgraphs that are induced by passing through such hubs are actually meaningful for neighbouring nodes with a small degree. Thus, we suggest a maximum degree constraint parameter d_{max} that is used in the exploration phase. If a node w is discovered in the neighbourhood of a node v such that $degree(w) > d_{max}$, then we add w to the subgraphs of v but do not explore beyond w. Note that we still include the label information of the hub itself. This approach considerably reduces the amount of required subgraph explorations. In Section 4.3.4, we analyze the impact of this heuristic on the predictive performance of the features.

4 EVALUATION

We compare the performance of heterogeneous subgraph features to classic features that we engineered with domain knowledge, and to state-of-the-art neural embeddings. As evaluation tasks, we consider the prediction of institution success in scientific publication networks and the prediction of node labels in labelled networks.

4.1 Evaluation Data Sets

To evaluate the performance of the subgraph features, we use three structurally diverse networks. As shown in Figure 2, the network types range from strongly interconnected relationships between labels to the star-like structures of knowledge bases.

Scientific Publication Network. For our experiments, we use the Microsoft Academic Graph (MAG) [24], which contains scientific publication records, citation relationships between those publications, as well as authors, institutions, journals, conferences, and fields of study. For the rank prediction task, we use a subset of institutions, authors, and papers centered on the respective institutions as specified in Section 4.2. For the label prediction task we extract all papers from the conferences KDD and ICML from 2011 to 2015, then add all referenced papers, their conferences, journals, authors, institutions and fields of research. The resulting network has 73,176 nodes, six labels, and 372,737 edges.

Entity Co-occurrence Network. The LOAD network is an entity co-occurrence network that is constructed from disambiguated named entity mentions in the text of Wikipedia, namely locations, organizations, actors, and dates [25], and represents word co-occurrence networks. We use a version of this network that is

constructed from Wikipedia articles about the American Civil War[2]. We extract the four types of entities to obtain a very dense network with four labels, 55,319 nodes, and 1,130,372 edges.

Movie Network. We use movie data from the Internet Movie Database (IMDB). Although it is proprietary, the data is available for research[3] and frequently used as an example of heterogeneous networks or in recommendation tasks. To select a subset of the data, we consider classic movies from the Golden Age of Hollywood (releases between 1930-1940). For each movie, we add the actors, directors, writers, and composers that were involved in the production, as well as keywords to the set of nodes, and connect them to the movie itself. The network has six labels, 48,555 nodes, and 213,562 edges. It is an example of relational data and has a star-like structure that is more sparse than the LOAD network.

4.2 Rank Prediction Evaluation

To compare the performance of classic, subgraph and embedded features on a task with rigorous ground truth data, we predict the relevance of research institutions for conference contributions based on the criteria defined in the 2016 KDD Cup. We consider 741 research institutions with publications at the conferences KDD, ICML, FSE, MM and MobiCom. The 2016 KDD Cup[4] provides labelled data for these conferences for the years 2011-2015, which we extend back to 2007 with crawled data from the ACM Digital Library. We use the years 2007-2014 for training and predict the relevance of institutions for the year 2015. The relevance rel_I of an institution I is defined based on three directives: **(i)** Each accepted full paper at a conference has an equal vote. **(ii)** Each author has an equal contribution to a paper. **(iii)** For authors with multiple affiliations, each affiliation contributes equally. The relevance of an institution is the sum of individual author contributions.

4.2.1 Evaluation Metric. In accordance with the original task, we use the *normalized discounted cumulative gain* for the top-20 rankings to evaluate the predicted relevance ranking. The NDCG at top-n was proposed by Järvelin et al. [12] and is defined as:

$$NDCG_n := \left[\sum_{i=1}^{n} \frac{rel_i}{\log_2(i+1)}\right]\left[\sum_{j=1}^{n} \frac{rel_j}{\log_2(j+1)}\right]^{-1} \quad (5)$$

where i is the predicted ranking position of an institution while j is the real ranking according to the ground truth. NDCG scores lie in the interval $[0, 1]$, with 1 corresponding to a perfect prediction.

4.2.2 Feature Extraction. We distinguish between three feature types. *Classic features* are engineered to encode factors influencing publication success and include linguistic features reflecting article contents. *Subgraph features* are the novel feature type discussed in Section 3. *Embedded features* include the three state-of-the-art neural embedding techniques LINE, node2vec, and DeepWalk.

Classic Features. These include the relevance score of each institution in previous years, both **(i)** as an absolute number and **(ii)** normalized by the number of accepted papers for this conference and year. We also include **(iii)** the amount of full-papers published by each institution, and **(iv)** the amount of all papers, including workshop and demo papers. Using authorship data, we calculate each author's average paper count per year and conference and generate **(v)** the authorship feature by grouping authors by institution and summing their scores. While it is possible for authors to be affiliated with multiple institutions over the years, such cases are exceedingly rare in the data. We furthermore consider the number of authors that each institution had at a conference in the past, split into **(vi)** authors of full papers and **(vii)** authors of short papers. Based on the intuition that the last-author position on a paper often indicates a senior research group member and the name is thus more likely to appear on subsequent papers, we include **(viii)** the number of last author occurrences as a final feature.

Classic Linguistic Features. We augment the set of classic features with linguistically motivated features. For each paper, we extract the number of different institutions, the number of keywords, the length in characters of the title, and the number of stemmed words per title (excluding stopwords). Additionally, we use frequency distributions of words and parts-of-speech in the titles and calculate the fractions of word parts-of-speech in the title. We aggregate these features by institution per conference and year. For each conference, we create a list of the overall top-20 title words from accepted papers and use it to derive the average number of occurrences of these words for each institution. In total, we extract 32 linguistic features for each institution: 4 simple features (average number of institutions, keywords, words in title, and characters in title), 8 features for the word classes (noun, verb, adjective, adverb, numbers, and punctuation), distribution and fraction of words, and 20 features for the usage of the top-20 title words.

Subgraph Features. To predict institution relevance, we focus on the neighbourhood of institutions in the graph and use induced subsets of the MAG that contain institutions, authors, and papers for each target conference and year, as well as all referenced papers with a distance of at most 2 to papers published at the selected conferences. For an overview, see the label connectivity graph in Figure 2 (left). While the MAG contains directed and undirected edges, we found no improvement in performance when using directed edges and report the results only for the undirected case.

We run the subgraph feature extraction algorithm for each institution to extract the frequencies of all subgraphs that contain the institution and have at most $e_{max} = 6$ edges. We use $d_{max} = \infty$, i.e., we do not apply the degree heuristic to this task.

Combined Features. To evaluate how well the classic and subgraph features complement each other and to assess if subgraph features can enhance features derived with domain knowledge, we also consider the combination of classic and subgraph features.

Embedded Features. We use *LINE*, *DeepWalk*, and *node2vec* as state-of-the-art features. All three rely on a neural network embedding of the node neighbourhoods that was originally conceived in natural language processing to learn word representations based on their context [17]. *LINE* optimizes the embedding towards retaining both the local and global network structure by integrating them in a unified objective function, and extracts the first- and second-order proximity of nodes [29]. *DeepWalk* is designed to learn latent representations of nodes in social networks and uses local information obtained from truncated random walks around a node as input sequence [20]. In contrast to this strictly random walk based approach, *node2vec* combines different exploration strategies for

[2] https://dbs.ifi.uni-heidelberg.de/resources/load/
[3] http://imdb.com/interfaces/
[4] https://kddcup2016.azurewebsites.net/

the extraction of local node neighbourhood information [7]. The method uses second-order random walks, which allow a tuning of the exploration towards a more localized or a more in-depth approach and benefits from both breadth-first as well as depth-first search information, at the cost of adding two parameters that require tuning. For all three embedding approaches, we use the recommended default parameters. That is, where applicable, we use an embedding dimension $d = 128$, walks per node $r = 10$, random walk length $l = 80$, context size $k = 10$, return parameter $p = 1$, in-out parameter $q = 1$, and number of negative samplings $K = 5$.

4.2.3 Experimental Setup. We select a set of standard machine learning regressors: linear regression, decision trees, random forests and Bayesian ridge (SVM and stochastic gradient descent performed poorly across all features and are omitted). We use the default configuration for each method provided by the Scikit-learn library [19]. For random forests, we increase the number of trees to 300 to obtain meaningful results for the feature importance analysis. Random forests and Bayesian ridge are robust enough to be trained on the entire set of features. For Bayesian ridge, we find that it yields better results when we select the 60 best features in a univariate analysis. Linear regression and decision trees are less suited for large sets of noisy features and performed poorly in our initial tests, so we select the 5 best features by a univariate test using a quick linear model for these two methods.

4.2.4 Ranking Task Evaluation Results. We provide the results of our experiments in Figure 3. We find that classic and subgraph features perform well overall, while embedded features perform consistently worse. As the only exception, LINE has a reasonable performance as a feature for random forests, where it performs best in one single instance (FSE). Assessing the stability by regression method, Bayesian ridge and random forest are the most stable for all features. For these methods, subgraph features consistently perform better than classic features, while the combination of both yields stable results. Predictions made with Bayesian ridge are always superior when they include subgraph features, and random forests with subgraphs always produce better or comparable results to classic features. Linear regression and decision trees fluctuate strongly by conference. While no feature is clearly preferable for these two methods, classic features perform better in this case.

In Table 1, we show the average NDCG score over all conferences. The highest score is achieved by random forests, for which we observe a tie between subgraphs as stand-alone features and in combination with classic features, while classic features are a close second. With the exception of decision trees, the combination of classic and subgraph features produces the most stable results. While this is not an indication that subgraph features are better than classic features, their performance is comparable. The low performance of neural embedding features for this task is not surprising, given that they only use structural information from the network. On the other hand, it is noteworthy that the subgraph features, which are predominantly structural features as well, perform so much better just by including the label information. As a result, we find that subgraph features can serve as out-of-the-box features on novel data or in instances that do not allow for the extraction of classic features when domain knowledge is not available. Given the immense manual effort that is required to engineer classic features,

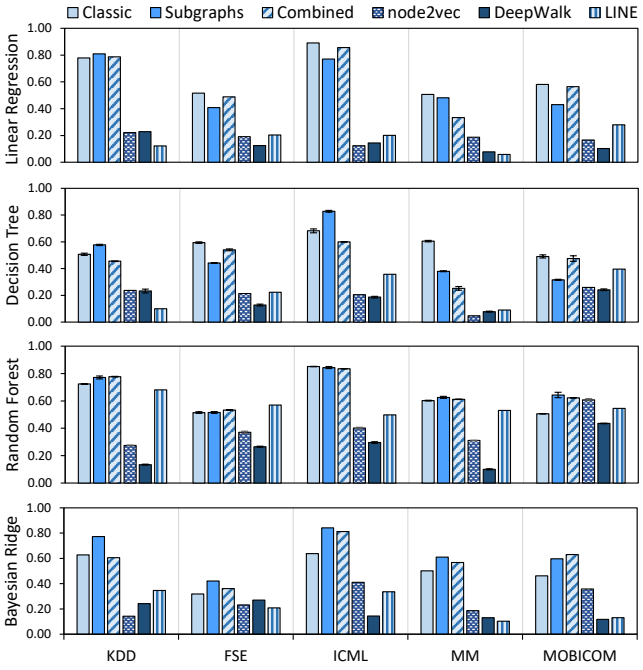

Figure 3: Comparison of NDCG scores of the four predictive methods, using classic, subgraph, combined, and embedded features to predict institution relevance for five conferences. Error bars denote 95% confidence intervals.

Table 1: Average NDCG scores over all conferences per predictive method and type of feature.

	LinRegr	DecTree	RanForest	BayRidge
classic	0.65	0.58	0.64	0.51
subgraph	0.58	0.51	**0.68**	0.65
combined	0.62	0.46	**0.68**	0.60
node2vec	0.18	0.19	0.39	0.27
DeepWalk	0.14	0.17	0.25	0.18
LINE	0.17	0.23	0.56	0.23

this is clearly advantageous, even in settings where classic features can be used. Additionally, subgraph features offer insights into the importance of individual features as we discuss in the following.

4.2.5 Feature Importance. An important aspect of any feature in applied learning is its expressiveness. While embedded features offer no insights into the classification process due to their abstract nature, classic and subgraph features contain further information. We use the random forest regressor to obtain feature importance.

Classic Features. The most expressive classic features are the rank of institutions and the total paper counts in previous years. Other classic and linguistic features play a much less prominent role. Since the prediction of the rank of an institution from the rank in previous years is intuitive and constitutes expected behavior, the knowledge we gain from this feature importance is limited.

Subgraph Features. In contrast, these features allow us to derive more detailed insights. In Figure 4, we show the most discriminative subgraph features for the rank prediction task. The subgraphs can be interpreted to allow conclusions about the structure of the

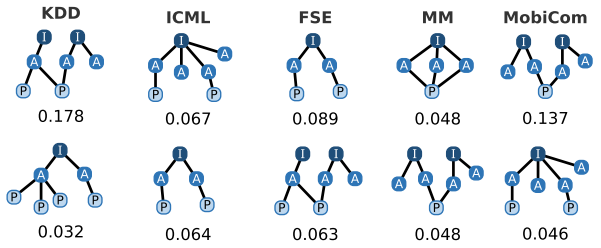

Figure 4: The two most discriminative subgraphs with relevance scores for each conference according random forests.

data, for example by identifying important substructures. For the ranking evaluation, we find that collaboration across institutional boundaries is apparently a good indicator for relevance, as several such structures exist in the most discriminative subgraphs (i.e., two co-authors of different institutions). On the other hand, authors with multiple affiliations do not play a significant role. While such observations are anecdotal, they offer insights into both the data and the task, which opens new possibilities for more specialized and elaborate feature extraction or prediction techniques.

4.3 Label Prediction Evaluation

We also perform label prediction on the three networks introduced in Section 4.1. For two of them, no data is available that would allow us to engineer classic features. Thus, we focus on the comparison of subgraph and embedded features. For each label type, we extract the features of nodes with this label, partition the nodes into training and test data, and train prediction classifiers for each type of feature.

4.3.1 Evaluation Metric. To evaluate the correctness of the predicted node labels, we use the *Macro F_1 score* as the average of the F_1 scores for the individual nodes v in the test set T as

$$\textit{Macro } F_1 := \frac{1}{|T|} \sum_{v \in T} \frac{2 \cdot \text{prec}(v) \cdot \text{rec}(v)}{\text{prec}(v) + \text{rec}(v)} \tag{6}$$

where $\text{prec}(v)$ is the fraction of predicted labels that are correct and $\text{rec}(v)$ is the fraction of correct labels that were predicted. We use the Macro F_1 score for comparability to the results of the embedding features, which were originally evaluated with this metric.

4.3.2 Feature Extraction. For each network, we select 250 nodes of each label and extract all three features for these nodes. For LINE, node2vec, and DeepWalk, we utilize the recommended default parameter values (see Section 4.2.2). We use $e_{max} = 5$ for the heterogeneous subgraph features. Since only the subgraph features encode label information, we adjust the encoding scheme to avoid unfair bias. While the label of the starting node is not obvious from the subgraph encoding itself, the inclusion of the starting node's label may introduce a bias due to the increased frequency of the label in the starting node's features. Therefore, we apply an artificial *starting label* to all start nodes during the extraction process that masks the node's label in the feature and avoids this bias.

4.3.3 Experimental Setup. We use logistic regression as a classifier to be in conformance with the reference evaluations of the embedded features in the original publications [7, 20]. For all features, we tune the regularization strength and use L2 regularization.

Table 2: Macro F_1 scores for subgraph features for varying levels of the maximum degree parameter. The value of d_{max} is set to disable exploration beyond nodes with a degree greater than the maximum degree in the given percentile.

	d_{max} parameter level					
	90%	92%	94%	96%	98%	100%
LOAD	0.76	0.75	0.73	0.76	0.74	–
IMDB	0.44	0.39	0.43	0.55	0.54	0.55
MAG	0.55	0.35	0.36	0.30	0.40	–

Table 3: Execution time per node (in seconds) for feature extraction. Percentiles denote the amount of nodes for which the feature extraction completes in at most the given time.

	subgraph features					n2v	DW	LINE
	mean	75%	90%	95%	max		mean	
LOAD	32.1	19.6	29.7	53.0	1046	0.19	0.11	0.66
IMDB	2.6	1.7	3.0	6.7	47	0.01	0.01	0.64
MAG	25.2	10.4	11.0	19.5	2493	0.02	0.01	0.49

From the extracted node features, we train classifiers in a one vs. all setting such that we obtain one classifier for each label. For prediction, we then select the label with the highest probability score for each node and use it for evaluation.

4.3.4 Maximum Degree Parameter Stability. In Section 3.2, we introduced the parameter d_{max} as a heuristic to avoid the addition of nodes to a subgraph that can be reached only through a hub node. We evaluate this parameter on all three networks by adjusting d_{max} to correspond to the percentage of nodes in the network that have degree d_{max} or less. The results are shown in Table 2. For the two larger networks, we do not show the results for $d_{max} = \infty$ (100%) since the extraction did not finish. The results for LOAD are very stable, while the results for IMDB and MAG are less stable, which correlates with the densities of the networks. Overall, we find that d_{max} is a helpful heuristic for dense networks with large hubs, where it enables efficient subgraph feature extraction, but should not be overused for smaller or less dense networks. For the following evaluations, we use a d_{max} value at the 90% mark.

4.3.5 Runtime Evaluation. In Table 3, we show the time requirements for extracting subgraph features. For comparison, we also include the runtime of the three neural embedding approaches, which are faster to extract than subgraph features. Among the embeddings, LINE is much slower than node2vec and DeepWalk. The significant differences in runtime to the subgraph features can be explained by sampling: while our method enumerates all subgraphs around a node, the embedding techniques sample via a fixed set of random walks or local searches. For the subgraph features, the overall runtime varies and is heavily skewed since it correlates to the skewed degree distribution: extracting features for nodes with a high degree takes longer than for nodes with small degree. Outliers (see column *max*) occur when a hub is the starting node (recall that the degree heuristic does not apply in this case). On the one hand, this problem is easy to avoid by not extracting features for such nodes. On the other hand, such a sampling approach is of course problematic for data in which certain features are unique to nodes with high degree. In practice, we find that the prediction

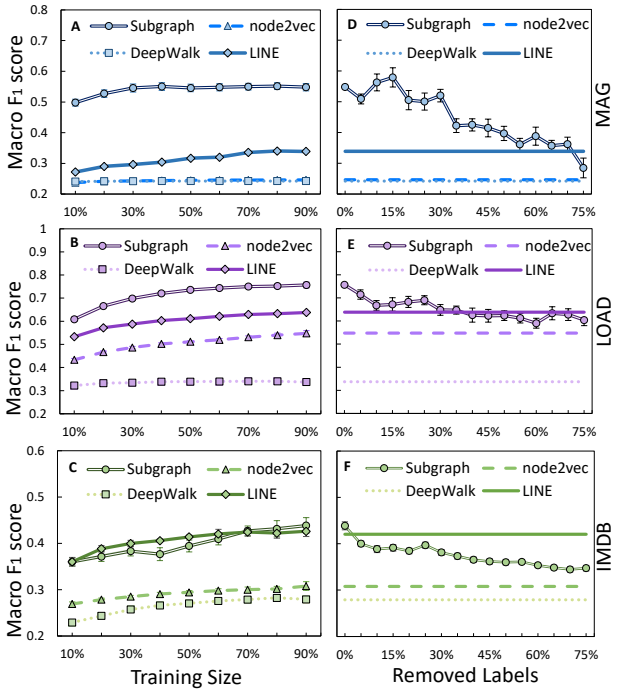

Figure 5: A-C: label prediction performance of the subgraph features, LINE, node2vec, and DeepWalk for the three evaluation data sets. The size of the training data (and thus the test data) is varied in steps of 10%. Error bars represent the 95% confidence interval of the F_1 score for 100 variations of the training/test set. D-F: Performance on data with partially removed node labels for a training size of 90%.

performance does not decrease when we extract features only up to the 95% mark (i.e., if we ignore the 5% of highest degree nodes).

4.3.6 Label Prediction Evaluation Results. Based on the previous considerations, we demonstrate the performance of the subgraph features for the label prediction task. We show the results of our experiments on the three networks in Figure 5A-C for varying percentages of training data (the remainder is used for testing). The performances of all features vary by network due to the varying difficulty of the prediction task in each of the data sets. For the most difficult data set (IMDB), all methods benefit the most from an increased amount of training data, while this effect is less pronounced for the other data sets. The results show that the node2vec features are more performant than the DeepWalk features, which corresponds to previous observations [7]. However, LINE performs better than the other two neural embeddings in all cases, while all neural embedding methods are outperformed by the heterogeneous subgraph features by a large margin. Across all three data sets, only in one instance does LINE provide results that are comparable to the subgraph features. The overall gain in prediction performance by using subgraph features instead of the best embedded features is as high as 68.8% on the MAG data set.

The performance of the subgraph features for label prediction can partially be attributed to the inclusion of label information in the feature. In Figure 5D-F, we thus show the performance for only partially labelled data on the three evaluation networks. To this end, we randomly remove a percentage of labels from nodes in the training data (i.e., we replace their label with an *unlabeled*-label). We evaluate with 90% training and 10% test data. The embedded features are invariant to node label removal and shown as horizontal lines. While the performance of the subgraph features drops as the percentage of unlabeled nodes increases, they still consistently perform better than node2vec and DeepWalk, even when 75% of the nodes have no label information. LINE initially performs worse than the subgraphs, but catches up as larger percentages of node labels are removed. Here, we find that the relative performance of subgraph features compared to LINE strongly depends on the initial performance gap on the data set. The larger this difference, the longer it takes LINE to achieve comparable performance. On the MAG data, LINE only reaches comparable performance once 75% of node label are removed, while this is reduced to 25% for LOAD and 10% for the IMDB data. A pattern that we observe for all three data sets is a pronounced drop in the performance of the subgraph features around 25% of removed node labels. Overall, as long as a substantial fraction of node labels are available, the performance of subgraph features is consistently strong. As a result, while heterogeneous subgraph features are naturally not the best choice for bootstrap label prediction in setting with no label information, they perform well even in settings with limited heterogeneity.

5 CONCLUSION

In this paper, we investigated heterogeneous subgraphs as features for predictive analyses of heterogeneous information networks, based on an efficient encoding scheme for fast (pseudo-) isomorphism tests. When used with machine learning techniques that support the assessment of feature importance, they are directly interpretable and enable the identification of discriminative substructures in the network. We found that the subgraph features perform at least as well as classic features that are extracted with domain knowledge. In settings where such domain knowledge is scarce, subgraph features may thus serve as out-of-the-box replacements, which is remarkable since heterogeneous subgraph features encode only the structural information of the network.

For the task of label prediction on three structurally diverse networks, subgraph features outperformed neural embeddings by a large margin. Existing node embedding techniques provide, without a doubt, powerful features in tasks for which they have been well tuned, but are not quite as universally performant. As versatile features for diverse prediction tasks on unseen data sets without domain knowledge, we find heterogeneous subgraph features to be easier to interpret, more versatile, and more performant, albeit at the cost of an increased extraction time. Based on these results, we provide a parallel implementation of our subgraph feature extraction framework in C++ and Python (see footnote in Section 1).

Future Work. We made no distinction between directed and undirected edges since we found no significant difference in the results for academic citation networks. However, this remains to be investigated for other types of directed networks. Likewise, our results indicate that subgraph encodings can be adapted to edge-heterogeneous networks, but the performance of such encodings is an open question whose answer stands to establish subgraphs as truly universal features for learning in heterogeneous networks.

REFERENCES

[1] Yael Artzy-Randrup, Sarel J Fleishman, Nir Ben-Tal, and Lewi Stone. 2004. Comment on "Network Motifs: Simple Building Blocks of Complex Networks" and "Superfamilies of Evolved and Designed Networks". *Science* 305 (2004), 1107. https://doi.org/10.1126/science.1099334

[2] László Babai and Eugene M. Luks. 1983. Canonical Labeling of Graphs. In *STOC*. https://doi.org/10.1145/800061.808746

[3] Phiradet Bangcharoensap, Tsuyoshi Murata, Hayato Kobayashi, and Nobuyuki Shimizu. 2016. Transductive Classification on Heterogeneous Information Networks with Edge Betweenness-based Normalization. In *WSDM*. https://doi.org/10.1145/2835776.2835799

[4] Yuxiao Dong, Reid A Johnson, and Nitesh V Chawla. 2015. Will This Paper Increase Your h-Index?: Scientific Impact Prediction. In *KDD*. https://doi.org/10.1145/2684822.2685314

[5] Yuan Fang, Wenqing Lin, Vincent Wenchen Zheng, Min Wu, Kevin Chen-Chuan Chang, and Xiaoli Li. 2016. Semantic Proximity Search on Graphs With Metagraph-based Learning. In *ICDE*. https://doi.org/10.1109/ICDE.2016.7498247

[6] Michael R Fellows, Guillaume Fertin, Danny Hermelin, and Stéphane Vialette. 2011. Upper and Lower Bounds for Finding Connected Motifs in Vertex-colored Graphs. *J. Comput. System Sci.* 77, 4 (2011), 799–811. https://doi.org/10.1016/j.jcss.2010.07.003

[7] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *KDD*. https://doi.org/10.1145/2939672.2939754

[8] Chun Guo and Xiaozhong Liu. 2015. Automatic Feature Generation on Heterogeneous Graph for Music Recommendation. In *SIGIR*. https://doi.org/10.1145/2766462.2767808

[9] Keith Henderson, Brian Gallagher, Tina Eliassi-Rad, Hanghang Tong, Sugato Basu, Leman Akoglu, Danai Koutra, Christos Faloutsos, and Lei Li. 2012. RolX: Structural Role Extraction & Mining in Large Graphs. In *KDD*. https://doi.org/10.1145/2339530.2339723

[10] Zhipeng Huang, Yudian Zheng, Reynold Cheng, Yizhou Sun, Nikos Mamoulis, and Xiang Li. 2016. Meta Structure: Computing Relevance in Large Heterogeneous Information Networks. In *KDD*. https://doi.org/10.1145/2939672.2939815

[11] Yann Jacob, Ludovic Denoyer, and Patrick Gallinari. 2014. Learning Latent Representations of Nodes for Classifying in Heterogeneous Social Networks. In *WSDM*. https://doi.org/10.1145/2556195.2556225

[12] Kalervo Järvelin and Jaana Kekäläinen. 2000. IR Evaluation Methods for Retrieving Highly Relevant Documents. In *SIGIR*. https://doi.org/10.1145/345508.345545

[13] Chuntao Jiang, Frans Coenen, and Michele Zito. 2013. A Survey of Frequent Subgraph Mining Algorithms. *The Knowledge Engineering Review* 28, 01 (2013), 75–105. https://doi.org/10.1017/S0269888912000331

[14] Richard M Karp and Michael O Rabin. 1987. Efficient Randomized Pattern-matching Algorithms. *IBM Journal of Research and Development* 31, 2 (1987), 249–260. https://doi.org/10.1147/rd.312.0249

[15] Jeremy K Mason, Emanuel A Lazar, Robert D MacPherson, and David J Srolovitz. 2012. Statistical Topology of Cellular Networks in Two and Three Dimensions. *Physical Review E* 86, 5 (2012), 051128. https://doi.org/10.1103/PhysRevE.86.051128

[16] Brendan D McKay and Nicholas C Wormald. 1991. Asymptotic Enumeration by Degree Sequence of Graphs with Degrees $o(n^{1/2})$. *Combinatorica* 11, 4 (1991), 369–382. https://doi.org/10.1007/BF01275671

[17] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *ICLR*. http://arxiv.org/abs/1301.3781

[18] Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. 2002. Network Motifs: Simple Building Blocks of Complex Networks. *Science* 298, 5594 (2002), 824–827. https://doi.org/10.1126/science.298.5594.824

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830. http://dl.acm.org/citation.cfm?id=2078195

[20] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online Learning of Social Representations. In *KDD*. https://doi.org/10.1145/2623330.2623732

[21] Xiang Ren, Jialu Liu, Xiao Yu, Urvashi Khandelwal, Quanquan Gu, Lidan Wang, and Jiawei Han. 2014. ClusCite: Effective Citation Recommendation by Information Network-based Clustering. In *KDD*. https://doi.org/10.1145/2623330.2623630

[22] Xiang Ren, Yujing Wang, Xiao Yu, Jun Yan, Zheng Chen, and Jiawei Han. 2014. Heterogeneous Graph-based Intent Learning with Queries, Web Pages and Wikipedia Concepts. In *WSDM*. https://doi.org/10.1145/2556195.2556222

[23] Pedro Ribeiro and Fernando Silva. 2014. Discovering Colored Network Motifs. In *Complex Networks V*. Springer, 107–118. https://doi.org/10.1007/978-3-319-05401-8_11

[24] Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June (Paul) Hsu, and Kuansan Wang. 2015. An Overview of Microsoft Academic Service (MAS) and Applications. In *WWW Companion*. https://doi.org/10.1145/2740908.2742839

[25] Andreas Spitz and Michael Gertz. 2016. Terms over LOAD: Leveraging Named Entities for Cross-Document Extraction and Summarization of Events. In *SIGIR*. https://doi.org/10.1145/2911451.2911529

[26] Andreas Spitz and Emőke-Ágnes Horvát. 2014. Measuring Long-Term Impact Based on Network Centrality: Unraveling Cinematic Citations. *PloS one* 9, 10 (2014), e108857. https://doi.org/10.1371/journal.pone.0108857

[27] Yizhou Sun and Jiawei Han. 2012. *Mining Heterogeneous Information Networks: Principles and Methodologies*. Morgan & Claypool Publishers. https://doi.org/10.2200/S00433ED1V01Y201207DMK005

[28] Yizhou Sun, Jiawei Han, Charu C. Aggarwal, and Nitesh V. Chawla. 2012. When Will It Happen?: Relationship Prediction in Heterogeneous Information Networks. In *WSDM*. https://doi.org/10.1145/2124295.2124373

[29] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *WWW*. https://doi.org/10.1145/2736277.2741093

[30] Alex D. Wade, Kuansan Wang, Yizhou Sun, and Antonio Gulli. 2016. WSDM Cup 2016: Entity Ranking Challenge. In *WSDM*. https://doi.org/10.1145/2835776.2855119

[31] Xiujuan Wang, Natali Gulbahce, and Haiyuan Yu. 2011. Network-based Methods for Human Disease Gene Prediction. *Briefings in functional genomics* 10, 5 (2011), 280–293. https://doi.org/10.1093/bfgp/elr024

[32] Sebastian Wernicke. 2006. Efficient Detection of Network Motifs. *IEEE/ACM Trans. Comput. Biology Bioinform.* 3, 4 (2006), 347–359. https://doi.org/10.1109/TCBB.2006.51

[33] Sebastian Wernicke and Florian Rasche. 2006. FANMOD: A Tool for Fast Network Motif Detection. *Bioinformatics* 22, 9 (2006), 1152–1153. https://doi.org/10.1093/bioinformatics/btl038

[34] Xifeng Yan and Jiawei Han. 2002. gSpan: Graph-based Substructure Pattern Mining. In *ICDM*. https://doi.org/10.1109/ICDM.2002.1184038

[35] Xifeng Yan, Philip S Yu, and Jiawei Han. 2004. Graph Indexing: A Frequent Structure-based Approach. In *SIGMOD*. https://doi.org/10.1145/1007568.1007607

[36] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. 2014. Personalized Entity Recommendation: A Heterogeneous Information Network Approach. In *WSDM*. https://doi.org/10.1145/2556195.2556259