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, 12 pages. 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 modelling increasingly well. At their core, networks of entities offer a simple and intuitive representation of complex connected systems by abstracting them to the nodes and edges of a graph. Despite this apparent simplicity, even the most basic forms of such network structures pose rich analytical challenges. However, using a complete abstraction that reduces real networks to just a single type of connected nodes often constitutes a rather arbitrary approach that oversimplifies the represented system. As a result, more diverse networks have shifted into the focus of research, which are frequently referred to as *heterogeneous (information) networks* due to their composition 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 [11, 47], multiplex film citation analysis [35], the identification of a molecular basis for human disease [40], the embedding of language networks [38], or the extraction of events from implicit textual networks of named entities [34], to name but a few. Frameworks have been designed for the extraction, cleaning and analysis of such data [25] and efforts have been undertaken to make their heterogeneous structure more intuitively understandable [43]. The terminology is fairly ambiguous in the literature, where *heterogeneous* may refer to node-heterogeneous networks (also called multi-mode), edge-heterogeneous networks (also called multiplex or multi-layer), or both. Here, we refer to networks with different types of nodes as heterogeneous networks.

Data Mining in heterogeneous information networks strives to leverage the inherently diverse representation of information to derive insights into the underlying systems [36]. Most of the time, the knowledge of node types is used explicitly, for example, to recommend movies based on user and actor connections that contribute in very different ways to the overall result [47]. Similar examples include Wikipedia query intent analyses [29], social network analysis [16] and transductive classification [4]. Most prominently, the availability of large scientific publication networks such as the DBLP data or the Microsoft Academic Graph has recently motivated numerous investigations into the extraction of information from heterogeneous citation networks. In these investigations, the distinction between node types has proven to be highly successful, as the examples of the 2016 KDD Cup and the 2016 WSDM Cup [39] show, in which many of the top-placed contenders leveraged entity type information to great success in their assessment of the scholarly importance of articles and institutions [9, 14].

To support predictive analyses on this data, the extraction and engineering of representative node features from the network is paramount. Such features can be divided into two categories, namely (1) *intrinsic features* that require domain knowledge in the engineering phase and (2) *graph features* that are based on the topological structure of the network. Since feature engineering is costly and domain knowledge can be difficult to obtain, some emphasis has therefore been put on approaches for extracting node features or node embeddings to characterize entities in the graph based on their connectivity information, such as the embedding of the local neighbourhoods of nodes [10, 13, 27]. Frequently, these approaches employ controlled random walks around the node, which can be problematic due to the heavily skewed distribution of nodes with small and large degrees in real-world networks [38]. Despite the growing prevalence of heterogeneous networks, many existing general-purpose approaches to the extraction of node features from such networks do not include the notion of node labels as a distinct dimension. While these approaches 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 of node labels. Furthermore, since (neural) node embeddings include a dimension reduction, they result in inherently abstract representations that offer no insights into the structurally important aspects of the data and do not reflect them in the embedded features.

Here, we propose *heterogeneous subgraph features* that are based on both topological information and node labels. They are designed for settings in which domain specific features cannot be engineered for heterogeneous networks. Unlike node embeddings, they offer insights into the structure of the data itself. 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 counts of local subgraphs that surround a node reflect both abundance and sparsity of connectivity.

Contributions. The contributions of this paper are fourfold. **(i)** We introduce the concept of subgraph features for nodes in heterogeneous networks. This novel type of feature copes well with the skewed topology of real-world networks and can be used as an equivalent replacement of features that are engineered with domain knowledge when no such knowledge is available. **(ii)** We discuss the interpretability of this new feature and its encoding in contrast to neural embeddings, which serve only as abstract features. **(iii)** We provide an efficient implementation of the feature extraction and encoding framework that scales linearly with the number of nodes, and is trivially parallelizable[1]. **(iv)** We demonstrate the effectiveness of the new features against both classic features and neural node embedding techniques. We use a selection of machine learning techniques in two predictive tasks on three structurally diverse networks for an empirical evaluation.

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

Structure of the paper. In Section 2, we discuss related approaches and supporting work. We present the encoding scheme and algorithm for enumerating subgraph features in heterogeneous networks in Section 3. In Section 4, we evaluate the performance of heterogeneous subgraph features for rank prediction tasks, discuss the most discriminative types of subgraphs, and present a performance comparison of subgraph features to state-of-the-art node embeddings for the task of label prediction. We give a summary and outlook in Section 5.

2 RELATED WORK

Our approach touches on several concepts in the fields of Data Mining and network analysis as we discuss in the following.

Prediction in Information Networks. The extraction and prediction of information from (heterogeneous) information networks is comprised of many different methods. Here, we only present those that are most closely related. For an overview, we refer to literature on mining heterogeneous information networks [36].

Guo and Liu consider the task of feature generation for music recommendation in heterogeneous graphs in a collaborative filtering setting based on random walks in the graph [11]. In a similar approach, aimed at the task of personalized entity recommendation, Yu et al. employ heterogeneous relationship information in networks by extracting path-based latent features to represent the connectivity between users and items [47]. Bangcharoensap et al. propose an approach for the transductive prediction of node labels in heterogeneous information network data based on the notion of edge betweenness, and show applications in node labeling tasks for homogeneous networks [4]. The latent space heterogeneous model presented by Jacob et al. also addresses transductive classification in social networks by transforming the heterogeneous classification problem into multiple homogeneous problems [16]. Ren et al. use graphs of user queries, web pages and Wikipedia concepts for learning user intent [29]. A couple of works also employ readily available scientific publication network data sets. For example, Dong et al. predict the impact of scientific papers based on a network containing six different factors, among them authorship, venue, and citations [5]. Ren et al. derive citation recommendations from clustering heterogeneous information networks of scientific publications [28]. In an extension of link prediction on homogeneous networks, Sun et al. use heterogeneous scientific publication networks for collaboration prediction between authors in DBLP data by including a temporal aspect in the prediction [37]. Huang et al. propose meta-structures as a generalized version of meta-paths for relevance computations in heterogeneous information networks [15] by utilizing user-specified meta-structures as input seeds for all subsequent analyses of the network.

All of the above works make more or less explicit use of the heterogeneity of the networks. However, they also include domain specific knowledge. While such information is frequently available in settings such as scientific publication networks, these networks are not representative of the breadth of available data. In contrast, we generalize feature extraction to applications in which only a set of node labels is known in addition to the connectivity information.

An approach that does not use domain knowledge for the extraction of heterogeneous features is given by Fang et al., who learn and predict semantic proximity of nodes through the extraction of meta-graphs as features [7]. However, they rely on the existence of bipartite symmetries in the network to specifically extract meta graphs for proximity prediction, which do not work as universal features. In contrast, we present an approach for extracting subgraph features for general heterogeneous networks that do not require the presence of intrinsic symmetries or are designed with a focus on pairwise proximity computations between graph nodes.

Node Feature Extraction. A number of existing works focus on the extraction of representations or embeddings for nodes of a graph that can then be used for predictive tasks within the networks. To identify the role of nodes in a graph, Henderson et al. present an approach that is based on the premise of structural equivalence, which postulates that nodes with similar neighbourhood structure have similar roles [13]. Perozzi et al. learn latent social representations for nodes in a social network [27]. To this end, they employ an approach that is reminiscent of word embeddings in (relatively) low-dimensional vector spaces and apply it to random walks around nodes in the network. A similar approach by Grover and Leskovec [10] learns node embeddings in graphs based on random walks through local neighbourhoods of nodes. Their method combines different schemes for localized neighbourhood exploration and can be tuned to the current domain.

An important aspect of multi-mode networks that is missing in previous approaches is a reflection of the heterogeneity of node neighbourhoods in the network. Additionally, methods that are based on random walks suffer from the sparsity of the neighbourhood around low-degree nodes, which is problematic in real-world networks with skewed degree distributions. If the immediate neighbourhood of a node is very small and most higher-order neighbours are reached through a high-degree node, then a random walk retrieves non-local information very quickly. It is thus unsurprising that the length of the random walk has been found to have little impact on the performance of the resulting feature [27]. Most networks consist of disproportionately many nodes with a low degree that are well connected through hubs, which compounds this effect. Here, we introduce a subgraph-based method that negates this problem since the local sparsity itself is part of the feature.

Subgraph Mining and Encodings. A large body of research exists on the topic of subgraph mining, which is too extensive to cover here in its entirety. Primarily, a focus is put on efficient methods for the enumeration of a given subgraph (or class of subgraphs) in a large graph data set or data base, i.e., the answering of subgraph queries [12]. Optimizations range from a specialization on the type of graph such as cohesiveness of the subgraphs [31], connectivity of the subgraphs [1], to a parallelization of the implementation [32], diversification of the results [46], reductions in memory usage or machine size [20], or the quantification of graph patterns for association rule mining [6]. For a more in-depth overview of subgraph mining algorithms, we refer to the survey by Jiang et al. [18].

For subgraph encodings, an early contribution was given by Yan and Han with DFS-codes as canonical representations of subgraphs [44] that were originally designed for graph indexing [45]. More recently, Mason et al. introduced a scheme for encoding node neighbourhoods in cellular networks that is based on an enumeration of labellings for the local neighbourhood around a node, which they use for a topological comparison of cellular networks [21]. Due to the geometrically motivated construction, it is strictly limited to spatially embedded networks. Here, we introduce a novel encoding scheme for the representation of heterogeneous subgraphs that includes node labels in a characteristic sequence.

Network Motifs. A concept similar to subgraph mining is known as *network motifs*, which were introduced by Milo et al. [24] and are frequently used, predominantly in the analysis of biological networks. Motifs represent subgraphs that occur significantly more (or less) frequently in an observed network than they do in a comparable network model. As such, network motifs rely heavily on the existence of a proper network model for the data at hand, which may be unavailable in many cases. If a proper model for the network under consideration is unknown, the wrong choice of model for the determination of significance may lead to heavily biased results. This has led to criticism and calls for caution from the community [2], and makes motif-based analysis difficult to use in practice. On an algorithmic level, Wernicke provided an efficient algorithm for the extraction of all subgraphs of a network that are of a given size [41]. The algorithm can be extended to include the comparison to a graph model for significance analysis and is implemented in the tool FANMOD [42]. Ribeiro and Silva [30] introduce an algorithm for the extraction of colored network motifs that employs gTries to enumerate all relevant subgraphs.

Compared to approaches for mining (colored) motifs, the extraction of heterogeneous subgraph features differs in one important aspect, as the enumeration of all *global* subgraph counts of a graph is inherently different from a *local* census of rooted subgraphs around selected nodes. Since motif analysis specifically aims at the enumeration of all subgraphs of a given network (or the generation of a representative sample), it is prohibitively expensive for all but the smallest subgraphs and networks. For feature extraction, we do not require a full enumeration but can rely on the subgraph counts around nodes that are chosen as a representative sample of the entire graph. Thus, using a census of rooted subgraphs around nodes as node features allows us to obtain subgraph counts that are substantially different from those that are used in motif analysis. As a result, motif extraction algorithms are ill-suited to subgraph feature extraction and we thus rely on an efficient encoding and exploration scheme as discussed in the following.

3 FEATURE MODEL

Let $G = (V, E)$ denote an undirected graph without self loops 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 introduce a set of node labels L along with a function $\lambda : V \to L$ such that $\lambda(v) \in L \; \forall v \in V$. A heterogeneous network is then a labelled graph $G = (V, E, L)$. In the following, we omit L where it is clear from context. To distinguish between networks with different levels of connectivity within the classes of nodes that are given by the labels, we introduce the notion of a *label connectivity graph*, in which all nodes with the same label are aggregated into a single node. Thus, the label connectivity graph of a network contains self loops iff the network contains edges between nodes with the same label. 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., a count). In the following, we discuss the extraction and encoding of such heterogeneous subgraph counts from a network.

Graph Isomorphism. A central aspect of subgraph extraction is related to graph isomorphism. During the exploration of the neighbourhood of a starting node, the nodes of structurally identical subgraphs may be visited in a different order, depending on the traversal of the neighbourhood. Therefore, the correctness of the subgraph census depends on a matching of identical subgraphs, independently of the order in which the nodes are visited. To formalize this problem, assume that we are given two labelled graphs $G = (V, E)$ and $G' = (V', E')$ over the same set of labels. We say that G is isomorphic to G' (and write $G \simeq G'$) if there exists a bijection $\phi : V_G \rightarrow V_{G'}$ with the following two properties: **(i)** $uv \in E$ iff $\phi(u)\phi(v) \in E'$ $\forall u, v \in V$. **(ii)** $\lambda(v) = \lambda(\phi(v))$ $\forall v \in V$. Therefore, two isomorphic graphs cannot be distinguished unless some node ordering is taken into account. Thus, the intuitively desired feature of a subgraph encoding scheme is the ability to distinguish subgraphs up to isomorphism. Unfortunately, efficient solutions for isomorphism detection pose a challenge since it is unknown whether a polynomial solution to the problem exists [3]. However, this difficulty is ameliorated by the fact that subgraph features represent the local neighbourhood of nodes and thus do not need to be arbitrarily large. Therefore, the number of nodes in subgraphs is limited to a size where the isomorphism problem becomes manageable with the correct encoding scheme. Furthermore, a low level of imprecision in the encoding does not decrease the quality of predictions that are based on the resulting features, as long as the number of ambiguous encodings is small compared to the overall number subgraphs. We use these observations to introduce an encoding scheme for heterogeneous subgraphs in the following.

3.1 Subgraph Encoding

The most time consuming aspect of the subgraph census is the isomorphism test that has to be performed for every newly 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, the isomorphism test should support an efficient subgraph comparison to avoid an overall quadratic complexity for the comparisons to previously discovered subgraphs. Therefore, we design a pseudo-canonical subgraph encoding in such a way that two small subgraphs are isomorphic iff their encodings are identical. Instead of checking two small subgraphs for isomorphism, it is thus sufficient to compare their encodings. Furthermore, a hashable encoding enables the use of hashmaps to keep track of the subgraph census, thus reducing the complexity for the extraction of a single subgraph occurrence to $O(1)$ for fixed maximum subgraph size. Our encoding is based on the labelled degree sequences of subgraphs, defined as follows.

Characteristic Sequence. Given a subgraph $H = (V_H, E_H) \subseteq G$, for each vertex $v \in V_H$ we define a sequence $s_v = t_0, t_1, \ldots, t_k$

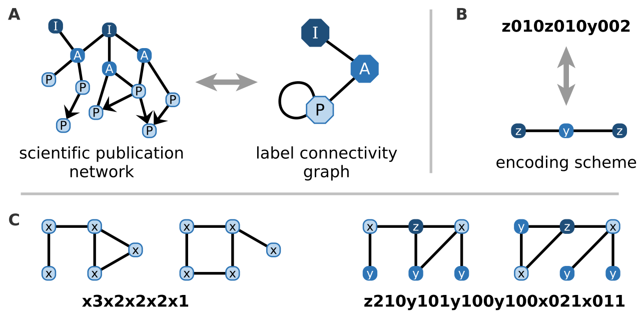

Figure 1: A: Scientific publication network of institutions I, authors A and publications P, with corresponding label connectivity graph. B: Characteristic sequence of a 3-node graph example. C: Two non-isomorphic graphs with a single label sharing the same encoding (left). Two non-isomorphic graphs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

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 these sequences, we then define the *characteristic sequence* s_H of the subgraph H as the concatenation of all sequences of individual nodes. Formally we 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 graphs with three 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, since we found no benefit to such a distinction in our experiments for the heterogeneous subgraph features. If desired, it would be a simple matter to extend the model by introducing a distinct label that is used specifically to mark the starting node.

Limitations. It is important to note that the encoding as introduced above is not unique for graphs of arbitrary size, since graph encodings may collide for larger subgraphs (see Figure 1C). Formal proof of subgraph multiplicities (and ensuing collisions) would require knowledge of the number of graphs with a given degree sequence, which has so far been elusive despite extensive research (see [22] for an introduction). However, an enumeration of all possible non-isomorphic labelled graphs with a pairwise check against the encoding can be used to derive upper bounds. 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, however, extensive testing showed that the collisions have no negative impact on the quality of the resulting features and that larger subgraphs serve as more discriminative features, since the overall number of collisions is negligible when compared to the number of subgraphs. Thus, collisions are no limitation to a practical application of the features.

Furthermore, we find that higher values of e_{max} correspond to a higher discriminative power and thus to a better performance of the features. However, higher values of e_{max} also increase the necessary feature extraction time since the number of possible subgraphs around a node grows roughly exponentially with the size of the subgraph. As a result, e_{max} should be selected as high as possible without impeding the feature extraction process. In practice, we find $e_{max} = 5$ to be a reasonable value for experiments on real-world network 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 all possible subgraph encodings then provide the heterogeneous subgraph feature space. That is, each distinct subgraph encoding serves as one feature, along with its count.

Implementation. In the design of the subgraph extraction algorithm, we first observe that this enumerative task is always computationally expensive [8], which encourages the implementation of efficient enumeration strategies. Thus, we base our algorithm for heterogeneous subgraph counting 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)** the node-based enumeration scheme supports by-node parallelization and sampling strategies.

Based on these considerations, we find that an approach based on depth-first search around the given root node performs well, as long as 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 the resulting encoding is updated in a hash map. Hashing the encoding based on the above scheme is a trivial matter 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. 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 considerations and optimizations that make this approach feasible in practice for heterogeneous networks with skewed topology.

Parallel Space Complexity. With the enumeration being trivially parallelizable by starting node, the memory usage is of interest. Here, we observe that the connectivity information given by the edges is not altered during the execution of the algorithm. The edge list can thus be shared among multiple threads. For each starting node, we only need to store data for all discovered nodes during the current iteration, which is in $O(V)$. For an implementation with t parallel threads, the overall memory usage is thus in $O(tV + E)$. Since most real-world networks tend to require the majority of necessary system memory for storing edge information (i.e., $|V| \ll |E|$), a parallelization is thus unproblematic for the available number of parallel threads on current computing hardware.

Hashing Optimization. Since the length of the characteristic sequences that represent identified subgraphs is bounded from above by a constant (equal to the maximum size of subgraphs times the number of distinct labels), the computational cost for hashing the sequence of a subgraph can be regarded as constant. A representation of the sequences as strings allows for a direct hashing strategy in most programming languages. In practice, however, the conversion to strings and the subsequent hashing of the strings can be costly. Here, we observe that the sequences are represented as vectors of integers and can thus be used directly as input for the computation of the hash value without the conversion to strings. Ideally, a hash function for characteristic sequences should be both fast to compute for integer sequences and also easy to update with new nodes so that we do not have to recompute the hash value of the entire sequence when subgraphs are expanded. One possible hashing scheme that supports such manipulations is provided by rolling hash functions, which were originally introduced for string hashing [19]. Thus, we propose a scheme that is based on the idea of representing the individual integer components of the sequence as factors of the powers of an appropriate base b. Specifically, we use a different base b_l for each label $l \in L$. Recall that the sequence $s_v = t_0, t_1, \ldots, t_k$ encodes the neighbouring labels of one node v in the subgraph, where t_0 is the label of the node itself and the values t_l correspond to the number of neighbours of v that have label l. We select the appropriate base b_v that corresponds to the label of v and compute the contribution of node v to the hash value as

$$h(s_v) := \sum_{i=1}^{|L|} t_i b_v^i \tag{5}$$

Due to the lexicographic ordering of the characteristic sequence, this value is well defined. To obtain the hash value $h(s_H)$ of the entire subgraph, we then simply compute the sum of all individual nodes' contributions and apply an appropriate modulo to keep the size of the hash value manageable. In an implementation of this scheme, we can pre-compute the powers of the base, multiply them by the integers in the characteristic sequence and compute the sum. If a new node is added to the subgraph, it is then sufficient to compute the new node's contribution to the hash value, increase the contributions of adjacent nodes accordingly, and add them to the hash value. Since the computation of this hash value now relies on a series of simple multiplications and additions, it can be computed efficiently and updated in less time than it would take to recompute it for each update of the subgraph.

Heterogeneous Optimization Heuristic. This heuristic exploits the heterogeneous structure of the graph and is based on the observation that the addition of a new node to an existing subgraph yields identical subgraphs for each new node with identical label. Thus, instead of using individual increments for each node, we can group neighbouring nodes according to their label and increase the corresponding subgraph counter by the number of adjacent nodes per label. It is still necessary to explore each of these neighbours separately (since their respective neighbourhoods are likely different). With this approach, it is sufficient to compute the modified hash value only once per label. In particular for graphs with a relatively low number of labels, this decreases the effort dramatically from $degree(v)$ to $\min\{degree(v), |L|\}$ computations (or updates) of hash values per node. The re-discovery of adjacent nodes that are already contained in the subgraph then requires special consideration as a border case, but the entire approach can be implemented efficiently by sorting the adjacency lists of nodes by label.

Topological Optimization Heuristic. A skewed distribution of degrees with a long tail is common to most networks, such that the majority of nodes have a fairly low degree, while only few nodes have extraordinarily large degrees. Such high-degree nodes, so called *hubs*, play a central role in the computational cost of subgraph enumeration 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. Based on this motivation, we suggest the use of 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, since hubs induce many more subgraphs than nodes with lower degree nodes. In Section 4.3.4, we analyze the impact of this heuristic on the predictive performance of the features.

4 EVALUATION

In the following, we compare the predictive performance of heterogeneous subgraph features to features 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.

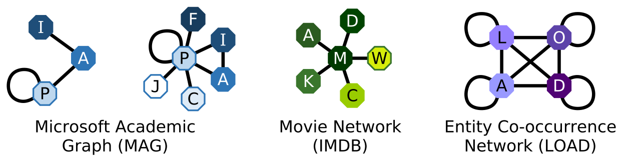
I
A
P
F
I
P
J
C
A
Microsoft Academic Graph (MAG)
A
D
M
W
K
C
Movie Network (IMDB)
L
O
A
D
Entity Co-occurrence Network (LOAD)

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.

4.1 Evaluation Data Sets

To demonstrate the results on diverse data sets and highlight the general applicability of the subgraph features, we select three structurally different networks for evaluation. 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. These networks constitute a well known type of heterogeneous network. For our experiments, we use the Microsoft Academic Graph (MAG) [33], which is a node-heterogeneous graph containing scientific publication records, citation relationships between those publications, as well as authors, institutions, journals, conferences, and fields of study. We use two subsets of this data. For the rank prediction task, we employ 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 [34]. Thus, the network covers the category of word co-occurrence networks that are frequently used for the evaluation of label prediction approaches, but provides more intuitive node labels than the frequently used parts-of-speech. We use a version of this network that is constructed from Wikipedia articles about the American Civil War[2]. We extract the four major types of entities to obtain a very dense network with four labels, 55, 319 nodes, and 1, 130, 372 edges.

Movie Network. As the final network, we use another well known data set that has a very clear network structure, the 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 in the analysis of heterogeneous networks or recommendation tasks. The data set is not provided as a graph but as lists of entities that can be parsed to extract a proper network of movies (we discard data related to TV-series and video games). To select a subset of the data, we consider classic movies from the Golden Age of Hollywood (released between 1930-1940). For each such 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 resulting network has six labels, 48, 555 nodes, and 213, 562 edges. It is an example of data with a traditional relational record-like structure and as one can see from the label connectivity graph, it has a star-like structure that is more sparse in contrast to the LOAD network.

4.2 Rank Prediction Evaluation

To compare the performance of classic, subgraph and embedded features on a task for which rigorous ground truth data is available, we predict the relevance of research institutions for conference contributions in computer science based on the criteria defined in the 2016 KDD Cup. Specifically, we consider 741 research institutions

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

that employ authors who published 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 data that we crawled from the ACM Digital Library. We use the years 2007-2014 for training and predict the relevance of an institution for the year 2015. The relevance rel_I of an institution I is defined based on three directives as follows: **(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 then given by the sum of all individual author contributions. Since the relevance scores are not normalized, we evaluate the performance per conference.

4.2.1 Evaluation Metric. In accordance with the original task definition, we use the *normalized discounted cumulative gain* for the top-20 rankings to evaluate the predicted relevance ranking scores. The NDCG at top-n was proposed by Järvelin et al. [17] and is defined for the purpose of our evaluation as:

$$NDCG_n = \left[\sum_{i=1}^{n} \frac{rel_i}{\log_2(i+1)}\right] \left[\sum_{j=1}^{n} \frac{rel_j}{\log_2(j+1)}\right]^{-1} \tag{6}$$

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 types of features. *Classic features* include features that are engineered to reflect factors influencing publication success, as well as linguistic features that reflect the content of articles. *Subgraph features* are the novel feature type discussed in Section 3. *Embedded features* include the three state-of-the-art neural node embedding techniques LINE, node2vec, and DeepWalk as representatives that perform best among this type of feature. All features are extracted for each institution per conference and year.

Classic Features. These features include the relevance score of each institution for the years 2007-2014, 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 in the past, and **(iv)** the amount of all papers, including workshop and demo papers. We generate features from authorship data to allow predictions of how many papers an author is going to contribute to a conference based on his previous publications. Specifically, 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 within 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 papers in subsequent years, we include **(viii)** the number of last author occurrences for institutions as a final feature.

[4]https://kddcup2016.azurewebsites.net/

Classic Linguistic Features. To augment the set of classic features, we include 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. For feature extraction, we thus use induced subsets of the MAG that contain the 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 (additional tests that we do not include here show that increasing this distance improves the results only slightly). For an overview of the data, see the label connectivity graph in Figure 2 (left). Note that this subset selection step is the only step that could be considered usage of domain knowledge in the subgraph feature extraction. While the MAG contains directed and undirected edges, we found no improvement in the prediction results when using directed edges and thus report the results only for the undirected case.

We run the subgraph extraction algorithm for each institution to extract the frequencies of all heterogeneous 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 maximum degree heuristic to this task.

Combined Features. To evaluate how well the classic and subgraph features complement each other and to assess the effectiveness of subgraph features as an enhancement to features derived with domain knowledge, we also consider the set of combined classic and subgraph features.

Embedded Features. To compare the performance of heterogeneous subgraph features to state-of-the-art neural embedding node features, we include features extracted with *LINE*, *DeepWalk*, and *node2vec*. All three methods rely on a neural network embedding of the neighbourhood of nodes in the network that was originally conceived in natural language processing to learn word representations in high-dimensional vector spaces according to their context [23]. *LINE* is a technique for large information network embedding. It optimizes the embedding towards retaining both the local and global network structure by integrating them in a unified objective function [38]. To this end, the method relies on extracting the first- and second-order proximity of nodes, which are concatenated into a combined vector representation. *DeepWalk* was originally designed to learn latent representations of nodes in social networks and uses local information that is obtained from truncated random walks around a node as input sequence [27]. The required notion of directedness is derived from the sequence of nodes in the random walk. While the method uses the length of the random walk as a parameter, the authors note that they found little impact on the performance when using various walk lengths. In contrast to this strictly random walk based approach, *node2vec* combines different exploration strategies for the extraction of local node neighbourhood information to learn continuous feature representations of nodes as neural network embeddings [10]. The method uses second-order random walks, which allow a tuning of the exploration towards a more localized or a more in-depth approach. As a result, the method benefits from both breadth-first as well as depth-first search information in the node neighbourhood, 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. Since the ranking task is best formulated as a regression problem, we select a set of standard machine learning regressors: linear regression, decision trees, random forests and Bayesian ridge (we also evaluated based on SVM and stochastic gradient descent, but found that these performed poorly across all features and thus omit the results). We do not tune the hyperparameters of the regressors and use the default configuration for each method provided by the Scikit-learn library [26]. For random forests, we increase the number of trees to 300 to obtain meaningful results that we can use in the feature importance analysis. Random forests and Bayesian ridge are robust enough to be trained on the entire set of features. However, for Bayesian ridge, we find that it yields better results (across all features) when we select the 60 best features in a univariate analysis, and thus report these results in the following. Linear regression and decision trees are less suited for large sets of noisy features and performed very poorly in our initial tests. To overcome this, we select the 5 best features by a univariate test using a quick linear model.

4.2.4 Ranking Task Evaluation Results. We provide the results of our experiments in Figure 3. While the performance of the features varies by regression method and target conference, we find that classic and subgraph features perform well overall, but embedded features perform consistently worse. As the only exception, LINE has a reasonably good performance as a feature for random forests, where it outperforms all other features in one single instance (FSE). In all other cases, neural embeddings perform poorly for this task. Analyzing the results by regression method to assess the stability, we find that the results of Bayesian ridge and random forest are the most stable for all features over the different conferences. For these methods, the subgraph features consistently perform better than the classic features, while the combination of both features compensates for performance drops in either of the features and yields stable results. Predictions made with Bayesian ridge are always superior when they include subgraph features, and random forests with subgraphs produce better or comparable results to classic features for all five conferences. The performances of linear regression and decision trees are less stable and fluctuate strongly by conference, which we attribute to the restriction to the top-5 features for these methods. Since subgraph and embedded features are more numerous than classic features, it is sensible that they perform better without this restriction. For linear regression and decision trees, no single feature is clearly preferable, although classic features perform better in this case.

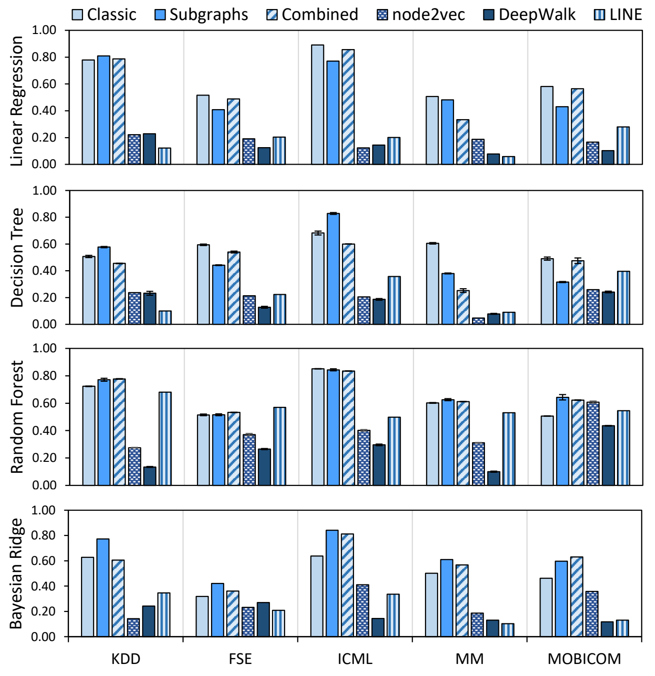

Figure 3: Comparison of NDCG values for the four predictive methods, using classic, subgraph, combined, and embedded features to predict institution relevance for five conferences. Error bars denote 95% confidence intervals.

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

In Table 1, we show the average NDCG score over all conferences. Here, the highest score is achieved for random forests, where 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 results in the overall most stable performance. 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, which is not enough to predict success as a non-structural, abstract feature in the graph. 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 such domain knowledge is not available. Given the immense manual effort that is required to engineer classic features, this is clearly advantageous, even in settings where classic features can be used. Additionally, subgraph features offer further insights into the importance 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 into the data. 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 data, for example by identifying important substructures. In the case of this evaluation, we find that collaboration across institutional boundaries is apparently a reasonably good characteristic of relevance, as several such structures exist in the most discriminative subgraphs (i.e., two authors of different institutions that collaborate on a paper). On the other hand, authors with multiple affiliations do not seem to 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, that neither classic nor embedded features can provide.

4.3 Label Prediction Evaluation

As a second evaluation task, we consider label prediction on the three heterogeneous networks introduced in Section 4.1. For two of the networks, no data is available that would allow us to extract classic features (an observation which holds for the majority of available network data sets). Thus, we focus on the comparison between subgraph and embedded features, since some of the latter were specifically designed and tested for this task. For each label type, we extract the features of nodes with this label, divide the nodes into training and test data, and train prediction classifiers for each type of feature. For nodes in the evaluation set, we then use the trained classifiers to predict the label.

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, defined as

$$Macro\ F_1 = \frac{1}{|T|} \sum_{v \in T} \frac{2 \cdot \text{prec}(v) \cdot \text{rec}(v)}{\text{prec}(v) + \text{rec}(v)} \qquad (7)$$

where prec(v) is the fraction of predicted labels that are correct and rec(v) is the fraction of correct labels that were predicted. Since all nodes in the networks have exactly one label, the F_1 score would be an equally valid metric. We are using this metric specifically for comparability to the results of the embedded features, which are originally evaluated with the Macro F_1 score.

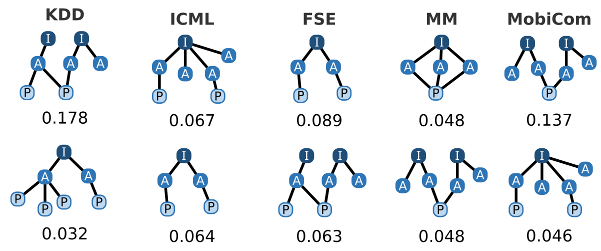

Figure 4: The two most discriminative subgraphs with relevance scores for each conference according random forests.

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	–

4.3.2 Feature Extraction. For each of the networks, we select 250 nodes of each label and extract all three features for these nodes as training and test data. 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 include an adjustment to 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, since the increased frequency of the label in all rooted subgraphs around the node is encoded. To avoid this problem, we apply an artificial *starting label* to all starting nodes during the extraction process that masks the node's label in the feature.

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 respective original publications [10, 27]. For all four types of features, we tune the regularization strength and use L2 regularization. From the extracted node features for each label, we train classifiers in a one vs. all setting such that we obtain one classifier for each label that provides a probability score of nodes in the unseen test data having the respective 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 with extremely high degree. 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

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

results for $d_{max} = \infty$ (100%) since the extraction did not finish due to the large number of subgraphs that are introduced by hubs. The results for LOAD are very stable, while the results for IMDB and MAG are less stable. These observations correlate with the density of the networks, where features in sparser networks become less stable the more hubs are ignored. 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. An important aspect of feature extraction is the computational effort. 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 prediction 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 [10]. However, LINE performs better than the other two neural embeddings in all cases, while all neural embedding methods are outperformed by the heterogeneous

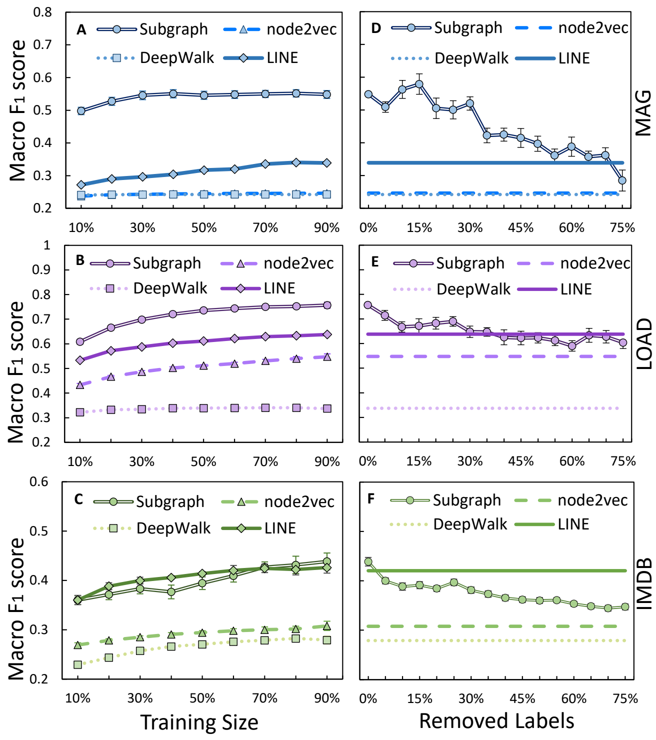

Figure 5: A-C: label prediction performance of the subgraph features, LINE, node2vec, and DeepWalk for the three evaluation data sets. The size of the training data (and thus the test data) is varied in steps of 10%. Error bars represent the 95% confidence interval of the F_1 score, for 100 variations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

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 presented heterogeneous subgraphs as a novel, generalizable approach to engineering features for predictive analyses of heterogeneous information networks. We introduced an efficient encoding scheme for fast (pseudo-) isomorphism testing of small, labelled subgraphs. The resulting features are powerful node representations for predictive learning tasks in heterogeneous networks. When used with machine learning techniques that support the extraction of feature importance, they are directly interpretable and allow the user to identify discriminative features as substructures in the network data. In our evaluation on a ranking task in a scientific publication network, we showed 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 or cannot be used to engineer classic features, subgraph features may thus serve as out-of-the-box replacements without a drop in performance. In particular this observation is remarkable, since heterogeneous subgraph features encode only structural information of the network as well as the existence of node label information, without using the information contained in the labels.

For the task of label prediction, we showed on three structurally diverse network data sets that subgraph features outperform neural node embeddings by a large margin. Existing node embedding techniques provide, without a doubt, powerful features in tasks for which they have been well tuned. They are, however, not quite as universally performant as the recent focus on neural embeddings might suggest. As versatile features for diverse prediction tasks on unseen data sets with little to no domain knowledge, we find heterogeneous subgraph features to be easier to interpret, more versatile and higher performing, albeit at the cost of an increased extraction time. Based on our results, we provide a parallel implementation of our subgraph feature extraction framework in `C++` and Python to the research community (see link in Section 1).

Future Work. For this project, we made no distinction between directed and undirected edges in our analysis, since we found no significant difference in the results for academic citation networks (which are the only network type with meaningful edge directions in our tests). However, this finding remains to be investigated in a more general scope for other types of directed networks. We suspect that for denser directed networks, directed subgraph features may turn out to be more performant than the undirected variety. We also excluded an adaptation of the encoding to edge-heterogeneous graphs in this paper, which remains to be investigated to establish heterogeneous subgraphs as truly universal features for learning in information networks.

REFERENCES

[1] Akhil Arora, Mayank Sachan, and Arnab Bhattacharya. 2014. Mining Statistically Significant Connected Subgraphs in Vertex Labeled Graphs. In *SIGMOD*. https://doi.org/10.1145/2588555.2588574

[2] 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

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

[4] 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

[5] 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

[6] Wenfei Fan, Yinghui Wu, and Jingbo Xu. 2016. Adding Counting Quantifiers to Graph Patterns. In *SIGMOD*. https://doi.org/10.1145/2882903.2882937

[7] 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

[8] 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

[9] Ming-Han Feng, Kuan-Hou Chan, Huan-Yuan Chen, Ming-Feng Tsai, Mi-Yen Yeh, and Shou-De Lin. 2016. An Efficient Solution to Reinforce Paper Ranking using Author / Venue / Citation Information. In *WSDM Cup*.

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

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

[12] Wook-Shin Han, Jinsoo Lee, and Jeong-Hoon Lee. 2013. Turbo$_{ISO}$: Towards Ultrafast and Robust Subgraph Isomorphism Search in Large Graph Databases. In *SIGMOD*. https://doi.org/10.1145/2463676.2465300

[13] 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

[14] Drahomira Herrmannova and Petr Knoth. 2016. Simple Yet Effective Methods for Large-Scale Scholarly Publication Ranking. *CoRR* abs/1611.05222. http://arxiv.org/abs/1611.05222

[15] 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

[16] 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

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

[18] 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

[19] 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

[20] Hyeonji Kim, Juneyoung Lee, Sourav S Bhowmick, Wook-Shin Han, JeongHoon Lee, Seongyun Ko, and Moath HA Jarrah. 2016. DUALSIM: Parallel Subgraph Enumeration in a Massive Graph on a Single Machine. In *SIGMOD*. https://doi.org/10.1145/2882903.2915209

[21] 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

[22] 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

[23] 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

[24] 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

[25] Walaa Eldin Moustafa, Hui Miao, Amol Deshpande, and Lise Getoor. 2013. GRDB: A System for Declarative and Interactive Analysis of Noisy Information Networks.

In *SIGMOD*. https://doi.org/10.1145/2463676.2465257

[26] 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

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

[28] 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

[29] 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

[30] 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

[31] Yingxia Shao, Lei Chen, and Bin Cui. 2014. Efficient Cohesive Subgraphs Detection in Parallel. In *SIGMOD*. https://doi.org/10.1145/2588555.2593665

[32] Yingxia Shao, Bin Cui, Lei Chen, Lin Ma, Junjie Yao, and Ning Xu. 2014. Parallel Subgraph Listing in a Large-scale Graph. In *SIGMOD*. https://doi.org/10.1145/2588555.2588557

[33] 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

[34] 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

[35] 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

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

[37] 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

[38] 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

[39] 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

[40] 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

[41] 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

[42] 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

[43] Ning Yan, Sona Hasani, Abolfazl Asudeh, and Chengkai Li. 2016. Generating Preview Tables for Entity Graphs. In *SIGMOD*. https://doi.org/10.1145/2882903.2915221

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

[45] 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

[46] Zhengwei Yang, Ada Wai-Chee Fu, and Ruifeng Liu. 2016. Diversified Top-k Subgraph Querying in a Large Graph. In *SIGMOD*. https://doi.org/10.1145/2882903.2915216

[47] 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