Latent Topics in Graph-Structured Data

Christoph Böhm Gjergji Kasneci Felix Naumann

Hasso Plattner Institute
Potsdam, Germany
firstname.lastname@hpi.uni-potsdam.de

ABSTRACT

Large amounts of graph-structured data are emerging from various avenues, ranging from natural and life sciences to social and semantic web communities. We address the problem of discovering subgraphs of entities that reflect latent topics in graph-structured data. These topics are structured meta-information providing further insights into the data. The presented approach effectively detects such topics by exploiting only the structure of the underlying graph, thus avoiding the dependency on textual labels, which are a scarce asset in prevalent graph datasets. The viability of our approach is demonstrated in experiments on real-world datasets.

Categories and Subject Descriptors

I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods; H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Algorithms

Keywords

Subgraph Mining, Latent Topics, Conceptual Patterns

1. INTRODUCTION

The organization of data in entity-relationship graphs has become the prime knowledge representation formalism for many communities: For instance, in biology, biochemical networks are used to capture regulatory or metabolic processes, protein-protein interactions, etc. In social web platforms, graphs are often used to represent relationships between entities of the platform, e.g., users, photos, homepages, etc. In the semantic web field, the Linked Open Data (LOD) community uses RDF to organize data from online communities, governments, scientific institutions, etc. Given the wealth of graph-structured data, it is a challenging undertaking for data engineers to choose the appropriate data(sub)set for the task at hand. Moreover, many of these sources, e.g., Yago, DBpedia, Freebase, or ProductDB, contain cross-domain knowledge and describe millions of entities but do not provide rich and meaningful meta information. Hence, even when the knowledge source is given, it is a difficult task to extract a subgraph of entities, in which all entities belong to the same latent topic of interest. Such a topical subgraph would avoid the hassle of managing the whole dataset; instead, only a small and "appropriate" fraction could be used. So far, there are no mechanisms to discover, capture, and manage latent topics in entity-relationship graphs. Further, rich textual labels are relatively scarce and thus any mechanism for discovering latent topics in these graphs should mitigate the dependency on textual labels.

Many knowledge discovery tasks are typically topic-related. As an example, consider the task of finding interesting connections between the Nike-sponsored football club FC Barcelona and the company Adidas. By inspecting the DBpedia dataset, it turns out that some of Barcelona's most important players (e.g., Lionel Messi) have an advertising contract with Adidas. This connection is depicted in the sample subgraph of Fig. 1. As a matter of fact, the automated discovery of such interesting connections between entities in large graph datasets is relatively costly. Hence, for the above example, rather than performing the discovery task on the whole graph, it would be advantageous to exploit just the subgraph that interconnects all the *sports-related* entities.

In this paper, we present an approach for discovering latent topics in graph-structured data by exploiting only the graph structure, thus completely avoiding the dependency on textual labels. However, in the presence of rich textual labels in the underlying graph, the proposed approach can be used to complement well-known text-based topic-model techniques, such as Latent Dirichlet Allocation (LDA) or Latent Semantic Analysis (LSA).

Related Work. Besides LDA and LSA, the work most related to ours is by Wu et al. [5] who categorize relational tables according to topics. Further, there is a large body of work on graph community detection [1, 2].

2. MINING LATENT TOPICS

Definition 1. Let E be a set of entities and R be a set of relationship types. An *entity-relationship graph* is a set of triples $G \subseteq E \times R \times E$. We denote by $E_G = \{s \mid (s,p,o) \in G\} \cup \{o \mid (s,p,o) \in G\}$ the set of all entities from G and by $R_G = \{p \mid (s,p,o) \in G\}$ the set of all relationships from G.

CIKM'12, October 29–November 2, 2012, Maui, HI, USA.

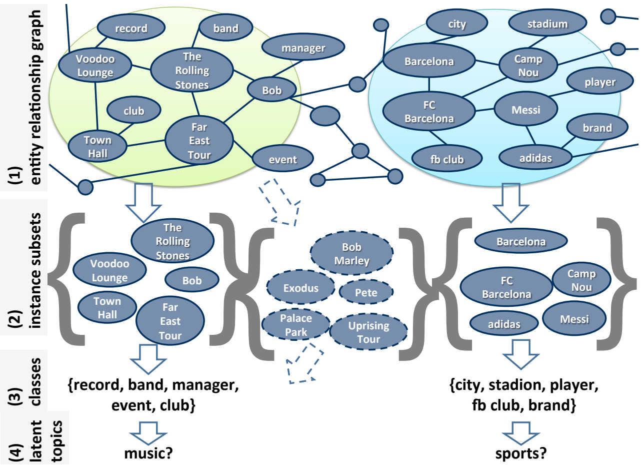

Figure 1: Example of an entity-relationship graph: Ellipses indicate entity nodes.Edges represent relationships among entities. Nodes in the large curly brackets illustrate topically related entities. Their classes, representing that topic, are depicted below. All together they describe the latent topic that corresponds to the subgraph enclosed by the large transparent ellipses on top. Note that there can be multiple topically related entity sets with equal representations, i.e., sets of classes. For instance, *Bob Marley*, etc. stem from another graph fractions (not depicted), but are instances of the same classes.

As with RDF, a triple $(s, p, o) \in G$ corresponds to a subject-predicate-object statement. Note that since our approach relies on the graph structure only, it is not important whether the entities in E_G are represented by unique IDs (e.g., URIs) or by rich textual labels.

Typically, in semantic web graphs, a specific relationship `rdf:type` is used to connect an instance to the classes it belongs to. We refer to it as the `type` relationship. Given an entity-relationship graph G, we denote the set of the immediate classes of all instances from G by $C_G = \{o \mid \exists s : (s, \texttt{type}, o) \in G\}$. Consequently, by $I_G = E_G \setminus C_G$ we denote the set of all instances from G. The function $t : I_G \rightarrow \mathcal{P}(C_G)$ with $t(e) = \{c \mid (e, \texttt{type}, c) \in G\}$ assigns all immediate classes to a given instance from G.

In Fig. 1, for cleaner illustration, the edge labels between the entities have been omitted. Given such an entity relationship graph, we seek subgraphs of topically related entities (background ellipses in the first layer). Such subgraphs comprise instances (second layer) as well as respective classes (third layer). These classes then describe the latent topic (fourth layer) of the underlying entity-relationship subgraph (first layer).

For the discovery of latent topics in entity-relationship graphs, we use a two-phase approach that exploits the inherent structure of the graph. First, we discover coherent connected subgraphs, so-called *conceptual patterns*, that capture "strong" relations between classes. In the second step, we combine these subgraphs to derive larger subgraphs that represent latent topics. We propose two alternative methods for the first phase in Sec. 2.1. Each of these methods discovers connected subgraphs with different structural properties. The combination of the discovered subgraphs is presented in Sec. 2.2.

2.1 Conceptual patterns

We start by introducing the basic subgraph structures that are used by our algorithms. These subgraph structures satisfy three main criteria: (1) the subgraph is connected; (2) for each instance in the subgraph, there is also one of its immediate classes in the subgraph, i.e., the subgraph comprises some conceptual information; (3) the subgraph has a high occurrence frequency in the underlying graph, i.e., it is a salient interconnection pattern.

Definition 2. Let G and G' be two entity-relationship graphs. G' has a *match* in G if there exists a subgraph $G^* \subseteq G$ such that:

1. $|I_{G'}| = |I_{G^*}|$
2. $\forall e \in I_{G^*} : |t(e)| = 1$
3. $\forall (s', p', o') \in G'$ with $s', o' \in I_{G'}, \exists (s, p, o) \in G^*$ with $t(s') = t(s) \wedge t(o') = t(o)$

We call the graph G' a *conceptual pattern* of G and the subgraph G^* a *conceptual match* of G' in G.

Note that a conceptual match is determined by the *class IDs* that occur in the pattern. Instance IDs are irrelevant. In order to find a conceptual match of G' in G, a subgraph G^* of G that is isomorph to G' has to be found. Figure 2 gives an example of two conceptual matches.

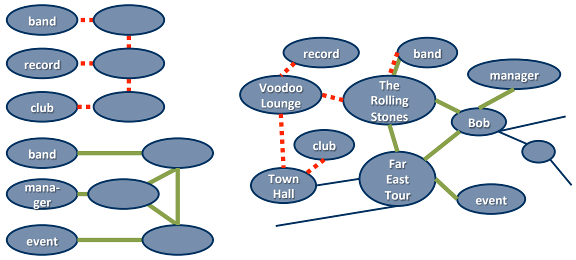

Figure 2: Two conceptual patterns (left) and their conceptual matches in the graph of Fig. 1 (right).

Conceptual Motif Patterns. To discover salient subgraph patterns that interconnect groups of instances with their direct classes (thus capturing conceptual relations between the instances), we adopt the notion of network motifs from [3] for Def. 2. There, a motif is defined as a "significantly often recurring interconnection pattern".

Definition 3. Given an entity-relationship graph G, a *conceptual motif* is a conceptual pattern G' of G that has significantly more conceptual matches in G than in any random graph with equal node properties.

To determine whether a subgraph pattern is a conceptual motif, we generate random graphs that contain all nodes (instances and classes) of the original graph and accept only those that have a similar node degree distribution as the original graph. This can essentially be done by shuffling around edges of the original graph. Then, we use a t-Test ($\alpha = 0.01$ in 30 random graphs) to check the occurrence frequencies of patterns in the original against pattern frequencies in the accepted random graphs. In the following, we refer to conceptual motifs as *CM patterns*.

Mutual Information Patterns. Although CM patterns can be derived directly from the entity-relationship graph, they are computationally relatively expensive. To provide a more efficient pattern discovery mechanism, we work on an abstract version of the entity-relationship graph. This abstract graph contains all *classes* of the original graph as nodes; its edges are derived from the connections between the *instances* in the original graph.

Let G be an entity-relationship graph and let $ni(c) = |\{e \in I_G \mid c \in t(e)\}|$ be the number of instances in G that have c as an immediate class. Similarly, let $nt(c_1, c_2) = |\{(s, p, o) \in G \mid c_1 \in t(s) \wedge c_2 \in t(o)\}|$ be the number of triples in G for which the subject and the object entity have c_1 and c_2 as immediate classes, respectively. Now, we define a weighted graph over the classes of G as follows:

Definition 4. A *weighted abstract graph* over an entity-relationship graph G is a graph $\mathcal{G}_G = (\mathcal{V}, \mathcal{E}, w)$ with

$$\begin{aligned} \mathcal{V} &= C_G \qquad \text{(the set of classes)} \\ \mathcal{E} &= \{(c_1, c_2) \mid c_1, c_2 \in C_G \wedge \\ &\quad \exists (s, p, o) \in G \ : \ c_1 \in t(s) \wedge c_2 \in t(o)\} \end{aligned}$$

$\forall (c_1, c_2) \in \mathcal{E}:$

$$w(c_1, c_2) = \log \frac{nt(c_1, c_2)/\alpha}{ni(c_1)/\beta * ni(c_2)/\beta}$$

where $\alpha = \sum_{c_1, c_2 \in C_G} nt(c_1, c_2)$ is the total number of triples (s, p, o) where s and o have immediate classes. Further, $\beta = \sum_{e \in I_G} |t(e)|$ is the total number of `type` edges in G.

The weighted abstract graph models the strength of relationships between classes with an estimate of the mutual information. In the next step, for a given abstract graph $\mathcal{G}_G$ and an integer m, we address the problem of extracting the top-k maximum-weight subgraphs that interconnect any m nodes in $\mathcal{G}_G$.

Definition 5. Given a weighted abstract graph $\mathcal{G}_G = (\mathcal{V}, \mathcal{E}, w)$ and two integers $k, m \in \mathbb{N}$, the *top-k maximum-weight connected subgraph problem* is to find k connected subgraphs $\mathcal{G}_i = (\mathcal{V}_i, \mathcal{E}_i)$ $(i = 1, ..., k \wedge |V_i| = m)$, such that there is no subgraph $\mathcal{G}_j = (\mathcal{V}_j, \mathcal{E}_j)$ $(j \neq 1, ..., k \wedge |\mathcal{V}_j| = m)$ with $\sum_{e \in \mathcal{E}_i} w(e) < \sum_{e \in \mathcal{E}_j} w(e)$.

Note that the above problem is a variation of the *dense k-subgraph problem*, which is known to be NP-hard. However, given the typical size of abstract graphs (i.e., hundreds of nodes) and the threshold $m \leq 10$ on the subgraph size, as well as a long-tail edge weight distribution, one can determine k maximum subgraphs by means of a bottom-up branch-and-bound strategy. We omit the algorithm for space reasons.

2.2 Percolating Patterns

So far we described two approaches for generating subgraph patterns that capture structural relations between classes. These patterns serve as building blocks for composing larger subgraphs, which in turn represent the latent topics.

Intuitively, patterns percolate through the graph if the graph comprises overlapping matches for similar patterns (inspired by Clique Percolation [4]). Figure 3 depicts the percolation of three different conceptual patterns. Then, the chain of pattern matches forms a latent topic, i.e., a subgraph that is best described by the class IDs occurring in the patterns (here *club*, *record*, *band*, *manager*, and *event*). This collection of classes might stand for the topic "*music*".

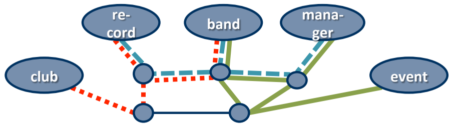

Figure 3: Neighboring pattern matches (dotted, dashed, and solid lines) in the graph of Fig. 1.

We now formalize the above intuition: Given a set P of conceptual patterns and an entity relationship graph G, let M represent all conceptual matches of patterns from P in G. We construct a match graph G_M by combining the matches from M. First, we define the notion of neighboring conceptual patterns. Then, we discuss the construction of the graph G_M that captures such neighborhood information.

Definition 6. Let $d \in \mathbb{N}$, $d > 1$. Two conceptual patterns p_1 and p_2 are *d-neighboring patterns* in an entity-relationship graph G if there exist matches G_1 for p_1 and G_2 for p_2 that share at least d instance nodes in G. We refer to the matches G_1, G_2 as *d-neighboring matches.*

In our implementation we have chosen the lower bound $d = \min(|I_{G_1}|, |I_{G_2}|) - 1$. We use this strict setting to ensure that neighboring matches (and thus the respective patterns) are highly similar, which in turn leads to particularly cohesive subgraph structures. The construction of the graph G_M is based on the above definition of d-neighboring matches.

Definition 7. Given $d \in \mathbb{N}, d > 1$ and the set $M = \{G_1, \ldots, G_n\}$ of all conceptual matches of the patterns $p_1, .., p_m$ in an entity-relationship graph G, the *match graph* G_M is constructed as follows:

1. Each match $G_i \in M$ becomes a node of G_M.
2. A pair $(G_i, G_j) \in M \times M$ becomes an edge of G_M iff G_i and G_j are d-neighboring matches.

Figure 4 depicts a match graph: each match is a node in G_M. Note that G_M is not necessarily connected; in practice, it consists of several connected components. Each connected component in G_M represents a structurally cohesive subgraph of the original entity relationship graph. By the definition of conceptual patterns, these subgraphs also represent cohesive interconnections between classes. By the same rationale these connected subgraphs are also good candidates for capturing latent topics.

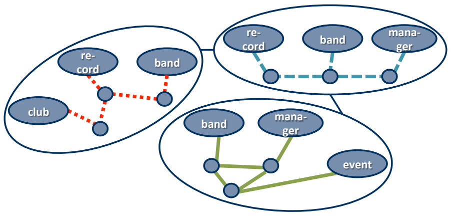

Figure 4: Match graph G_M created from matches shown in Fig. 3.

3. EVALUATION

We now evaluate general properties as well as the semantic coherence of the returned subgraphs.

Input. We used the DBpedia dataset (v3.6) which is highly heterogeneous and covers cross-domain information. We extracted an entity-relationship graph with 1,008,985 nodes and 4,789,940 edges. For this evaluation, we used patterns of size 3 and 4 and show results for a varying number of conceptual patterns used for percolation.

Gold Standard. For the quality evaluation, we compare classes contained in the mined entity-relationship subgraphs to those derived from Wikipedia *portals*. We extracted latent topic representations (sets of classes) from the portals by converting Wikipedia URLs to DBpedia URIs and determining associated classes. In the following the term *topic* refers to its representation, i.e., a *set of classes* from that topic.

Topic Assignment. We now have a set of topics from our approach and a set of 799 reference topics. For assigning our topics to reference topics, we computed Jaccard set similarity values $sim(T_x, T_y)$ for each pair of topics (one from our set and one from the reference set). Then, we assign a topic T_x to a reference topic T_y, iff $sim(T_x, T_y)$ is the maximum for both. That is, there are no topics T'_x or T'_y such that $sim(T_x, T_y) < sim(T_x, T'_y)$ or $sim(T_x, T_y) < sim(T'_x, T_y)$. Though this is a very strict criterion, it models our intuition that not all topics from our approach have a corresponding reference topic and vice versa.

Characteristics of Mined Topics. In general, CM patterns lead to more topics than MI patterns. CM patterns mainly produce topics comprising three to five *distinct* classes. MI patterns typically result in topics comprising three to four classes.

To examine topic overlaps, we determined Jaccard set similarity values for each pair of topics in the result, see Tab. 1. The average similarity for CM pattern topics decreases with the number of patterns under consideration. As for MI pattern topics, we observe only a slight variation over the number of percolating patterns. In general, created topics rarely overlap and MI patterns produce less overlapping topics. This is a desirable property, since otherwise topics are difficult to distinguish.

Table 1: Pairwise Jaccard Set Similarity of topics.

	CM patterns		**MI patterns**	
	avg.	**std.**	**avg.**	**std.**
16k	0.150	0.150	0.062	0.122
32k	0.125	0.139	0.044	0.101
64k	0.109	0.129	0.042	0.096
128k	0.098	0.123	0.042	0.096
256k	0.087	0.117	0.041	0.093
512k	0.081	0.114	0.045	0.092
1024k	0.071	0.110	0.046	0.091

Quality of Mined Topics. At the time of evaluation, Wikipedia featured 1,094 portals. Portals are intended to present a given topic and introduce "the reader to key articles, images, and categories that further describe the subject" (see `http://en.wikipedia.org/wiki/Portal:Contents`). A typical portal is divided into common information, selected articles, selected facts, current news, and the like. However, many of these links to other articles change frequently and cannot be considered a stable reference. Numbers for the extracted topics are also shown in Tab. 1. The number of topics evenly ranges over all sizes, i.e., 30 − 60 topics from size 2 to 26 distinct classes. The similarity among these topics is considerably higher (0.159 ± 0.118 on average) then for our topic sets under consideration. Table 2 depicts average precision, recall and F-measure values for the comparison with mapped topics created from Wikipedia portals. Apparently, the extraction performance is stable over the different numbers of percolating patterns. The precision ranges around 0.83 for topics generated by CM patterns and from 0.65 to 0.71 for those generated by MI patterns. Recall values lie around 0.65 and 0.53, respectively. In general, CM patterns yield better values then MI patterns. Most likely, this is the case, since CM patterns produce more topics. Consequently, more mappings can be found for the reference topics. The number of mapped topics (rightmost column in Tab. 2) supports this hypotheses.

Table 2: Average topic mining performance with respect to Wikipedia portals.

		Prec.	**Rec.**	**F-m.**	**Count**
CM patterns	16k	0.82	0.65	0.70	66
	32k	0.84	0.65	0.72	80
	64k	0.82	0.64	0.70	102
	128k	0.84	0.63	0.69	118
	256k	0.83	0.64	0.69	123
	512k	0.81	0.65	0.68	117
	1024k	0.81	0.65	0.68	134
MI patterns	16k	0.65	0.52	0.55	37
	32k	0.70	0.54	0.58	50
	64k	0.71	0.54	0.57	61
	128k	0.71	0.53	0.57	61
	256k	0.69	0.52	0.56	70
	512k	0.67	0.52	0.55	77
	1024k	0.67	0.52	0.55	83

4. CONCLUSION

We explored the problem of mining latent topics from graph-structured data and presented a novel approach that exploits only the structure of an entity-relationship graph. For the evaluation, we ran our algorithm on the DBpedia dataset and compared the mined result to topics extracted from Wikipedia. The findings show, that we can reasonably reconstruct a portion of this information. This is a remarkable result, given that our approach avoids textual labels.

Acknowledgments. We thank Johannes Gosda and Eyk Kny for their valuable support. This work has been funded by an IBM CAS grant.

5. REFERENCES

[1] S. Fortunato. Community detection in graphs. *Physical Review*, 486:75–174, 2010.

[2] J. Leskovec, K. J. Lang, and M. Mahoney. Empirical comparison of algorithms for network community detection. In *Proc. WWW*, 2010.

[3] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: simple building blocks of complex networks. *Science*, 298:824–827, 2002.

[4] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435:814–818, 2005.

[5] W. Wu, B. Reinwald, Y. Sismanis, and R. Manjrekar. Discovering topical structures of databases. In *Proc. SIGMOD*, 2008.