Approximate subgraph matching for detection of topic variations

Mitja Trampuš
Jozef Stefan Institute
Jamova 39
Ljubljana, Slovenia
mitja.trampus@ijs.si

Dunja Mladenić
Jozef Stefan Institute
Jamova 39
Ljubljana, Slovenia
dunja.mladenic@ijs.si

ABSTRACT

The paper presents an approach to detection of topic variations based on approximate graph matching. Text items are represented as semantic graphs and approximately matched based on a taxonomy of node and edge labels. Best-matching subgraphs are used as a template against which to align and compare the articles. The proposed approach is applied on news stories using WordNet as the predefined taxonomy. Illustrative experiments on real-world data show that the approach is promising.

Categories and Subject Descriptors

I.2.8 [**Artificial intelligence**]: Search—*Graph and tree search strategies*; I.5.4 [**Computing methodologies**]: Applications—*Text processing*

1. INTRODUCTION

One of the classic goals of text mining is to structure natural language text – for obvious reasons: the amount of information we can extract from the data using shallow approaches like bag-of-words is limited. By enhancing text with structure, we can start to observe information that is encoded in more than one word or sentence. Also, structure enables us to bring the additional power of semantic methods and background knowledge into the play.

While reasonably reliable methods have been developed for structuring text by annotating and identifying a specific subset of information, mostly named entities, little work has been done on semantically capturing the macro-level aspects of the text. In this article, we present some early work on constructing *domain templates*, a generic "summary" that fits many pieces of text on a specific theme (e.g. news stories about bombings) at the same time.

The genericness of the template provides for data exploration in two ways:

1. By automatically mapping specific facts and entities in an article to the more general ones in a template, we are providing structure to the articles as entities from potentially many articles get mapped to a single semantic "slot" in a single template.
2. By (possibly statistically) inspecting all the articles that were mapped to a chosen template, we can observe the diversity of articles *in a specific aspect*, exploiting the fact that they are semantically aligned to an extent. For example, if the template contains the statement `happen_at..location`, no further processing is required to find the specific locations which the template-mapped articles describe.

To determine entities and relations that subsume those from individual news articles and thus construct a template, we make use of a general-purpose ontology, in our case WordNet. To represent the templates as well as individual news stories, we use *semantic graphs*, i.e. graphs in which entities represented as nodes and the (binary) relationships connecting them are represented as edges.

A sample of the patterns we obtain can be seen in Figure 2. For example, analyzing a collection of articles describing various bombing attacks, the pattern in the first line emerges: a *person* was killed on a *weekday*; that same *person* was killed in an attack which took place. The concrete instantiations of *person* and *weekday* vary across articles from which the pattern was derived.

Domain specifics.

Note that media companies have a considerable interest in semantically annotating text, particularly news items. For this reason, and because of easy availability of datasets, we focus on the domain of newswire in this paper. Despite this, there is in principle nothing specific in this domain that would limit the applicability of our method to it. In general, the required input data is a collection of text items which are assumed to discuss roughly the same aspects of a single topic. Examples of such collections are "news articles about earthquakes", "Wikipedia articles on football players" or "microwave product reviews".

2. RELATED WORK

Because it aligns articles to a common template, our method has much in common with other information extraction mechanisms. Automatic construction of information extraction templates is already relatively well-researched. Most methods aim for *attribute extraction*, where the goal is to extract a single predefined type of information, e.g. the title of a book. Each separate type of information requires a separate

DiversiWeb 2011. Hyderabad, India.
.

classifier and training data. Examples of such approaches are [1, 2].

More recently, a generalized problem of *relation extraction* has received considerable attention. The goal is to find *pairs* of items related by a predefined relation. As an example, Probst et al. [7] mine product descriptions to simultaneously identify product attributes and their values. Relation extraction is particularly popular in biomedicine where pairs of proteins in a certain relation (e.g. one inhibits the other) are often of interest.

The task in this article is more generalized still; we attempt to decide both what information is interesting to extract as well as perform the extraction. This is known as *domain template extraction*. To our knowledge, little work has been done in the area so far. The most closely related work is by Filatova et al. [4], who find templates by mining frequent parse subtrees. Also closely related is work by Li et al. [6]; similarly to Filatova, they mine frequent parse subtrees but then cluster them into "aspects" with a novel graphical model. Both approaches produce syntax-level patterns. Unlike ours, neither of the two approaches exploits background knowledge. Also belonging to this group is our previous work [10] which mostly shares the goal and data representation ideas with this article, but uses different methods apart from preprocessing.

Graph-based templates are also used in [9] in a context similar to ours, though the semantics are shallower. Also, the authors focus on information extraction and do not attempt to generalize the templates.

Templates somewhat similar to those we aim to construct automatically and with no knowledge of the domain have already been created manually by domain experts. FrameNet [**?**] is a collection of templates for the events like "disclosing a secret", "speaking", "killing", "arresting" etc. They focus mostly on low-level events, of which typically many can be found in a single document, be it a news article or not. The project does not concern itself with the creation of the templates, other than from the methodological point of view. There is little support for automatic annotation of natural language with the FrameNet frames.

3. METHOD OVERVIEW

This section describes the various stages in our data processing pipeline. The assumed input data is, as discussed above, a collection of text items on the same topic. The goal is to identify a pattern which semantically matches a substantial number of the input texts.

The key idea is rather simple: we first represent our input data as semantic graphs, i.e. graphs of ontology-aligned entities and relations. A pattern is then defined as a (smaller) graph such that, by specializing some of its entities, a subgraph of at least θ input graphs (θ being a parameter). We seek to identify all such patterns.

We proceed to describe our approach to the construction of semantic graphs and to the mining of approximate subgraphs.

3.1 Data Preprocessing

Starting with plain text, we first annotate it with some basic semantic and linguistic information. Using the ANNIE tool from the GATE framework, we first detect named entities and tag them as person, location or organization. Following that, we use the Stanford parser [5] to extract subject-verb-object triplets. We then use the web service by Rusu [8] to perform coreference and pronoun resolutions ("Mr. Obama", "President Barack Obama" and "he" might all refer to the same entity within an article).

We acknowledge that the triplets acquired in this way do not necessarily provide a proper semantic description of the article data. The discrepancies go both ways:

- We include some triplets which do not make sense semantically, e.g. "`people..kill..Monday`" coming from "93 people were killed on Monday".
- We fail to create triplets for information not encoded with (lexicogrammatically) transitive verbs. For example, "President's visit to China ..." will not spawn "`president..visit..China`". In our experiments, this shortcoming is alleviated by using redundant information - each story, e.g. president's visit to China, is described by several articles which increases the probability that at least one will convey this information in a form we can detect. However, the problem is not completely overcome this way – some information e.g. the "93" in "93 people were killed on Monday" will never appear as the object of a transitive verb.

As a last step, we align all triplets to WordNet; that is, for each subject, verb and object appearing in any of the triplets, we try to find the corresponding concept (or "synset", as they are called) in WordNet. We first remove inflection from the words using python NLTK (Natural Language Toolkit), then align it to the corresponding synset. If more than one synset matches, we choose the most common sense which is a well-tried and surprisingly good strategy. For words not found in WordNet, we create a new synset on the fly. If the new word (e.g. "Obama") was previously tagged by ANNIE (with e.g. "person"), the new synset's hypernym is set accordingly.

3.2 Semantic Graph Construction

From a collection of triplets, we proceed to construct the semantic graph. Here, we rely rather heavily on the fact that news articles tend to be focused in scope: we do not disambiguate entities other than by name (not necessarily a proper name; e.g. "book" is also a name). As an example, if an article mentions two buildings, one of which burns down and the second of which has a green roof, our method detects a single "building" and assigns both properties to it. In the newswire domain, we have not found this to be a significant issue: entities which do need to be disambiguated are presented with more unique names ("France" instead of "country" etc.). This rationale would have to be revised if one wanted to apply the approach to longer texts.

This assumption greatly simplifies the construction of the semantic graph: we start by treating each triplet as a 2-node component of a single very fragmented graph and then collapse the nodes with the same labels.

Dataset specifics.

In our experiments, each input "document" in the sense described here was in fact a concatenation of actual documents, all of which were reporting on the exact same news event. Section 4 contains the details and rationale.

3.3 Approximate Pattern Detection

Given a collection of labeled graphs, we now wish to identify frequent "approximate subgraphs", i.e. patterns as described at the beginning of Section 3.

Formal task definition: Given a set of labeled graphs $S = \{G_1, \ldots, G_n\}$, a transitive antisymmetric relation on graph labels $genl(\cdot, \cdot)$ (with $genl(A, B)$ interpreted as "label A is a generalization of label B") and a number θ, we wish to construct all maximal graphs H that are *approximate subgraphs* of at least θ graphs from S. A graph H is said to be an approximate subgraph of G iff there is a mapping f of $V(H)$ onto a subset of $V(G)$ such that $genl(v, f(v))$ holds for all $v \in V(H)$.

This is not an easy task. Mining frequent subgraphs is in itself computationally demanding because of isomorphisms; satisfactorily fast algorithms for this seemingly basic problem are relatively recent [11]. By further requiring that the frequent subgraph only match the input graphs in a *soft* way implied by a taxonomy (here WordNet hyperymy), the complexity becomes insurmountable. We compensate by introducing two assumptions.

1. The hierarchy imposed by *genl* has a tree-like form, it is not a general DAG. This is true of WordNet: every synset has at most one hypernym defined.

2. Very generic patterns are not interesting and can (or even should) be skipped. This too is a safe assumption in our scenario: a pattern in which every node is labeled with the most generic label `entity` is hardly informative regardless of its graph structure.

We can now employ a simple but effective three-stage search. The stages are illustrated in 1 with the minimal example of two two-node graphs.

1. Generalize all the labels of input graphs to the maximum extent permissible. Under the first assumption, "generalizing a label" is a well-defined operation. The exact meaning of "maximum extent permissible" is governed by the second assumption; no label should be generalized so much as to fall in the uninteresting category. In our experience with WordNet, the following simple rule worked very well: generalize verbs as much as possible and generalize nouns to two levels below the hierarchy root. See steps 1 to 2 in Fig. 1.

2. Mine θ-frequent maximal subgraphs with support of the generalized input graphs. This step cannot be shown in Fig. 1 as the graphs are too small.

3. Formally, the resulting subgraphs already satisfy our demands. However, to make them as descriptive as possible, we try to specialize the pattern's labels, taking care not to perform a specialization that would reduce the pattern's support below θ. Specialization, unlike generalization, is not a uniquely defined operation (a synset can have many hyponyms), but with some we can afford to recursively explore the whole space of possible specializations. We use the sum of labels' depth in the WordNet hierarchy as a measure of pattern descriptiveness that we optimize. See steps 2 to 3 in Fig. 1.

For frequent subgraph mining, we developed our own algorithm, inspired by the current state-of-art[11, 3]. We included some improvements pertaining to maximality of output graphs and to scalability – all existing open-source software crashed on our full input data.

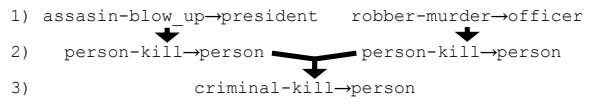

Figure 1: Generalization of input graphs and re-specialization of the pattern.

4. PRELIMINARY EXPERIMENTS AND RESULTS

As a preliminary, let us define some terminology suitable for our experiment domain. An *article* is a single web page which is assumed to report on a single *story*. A story is an event that is covered by one or more articles. Each story may fit some *domain template* (also *event template* or simply *template*) describing a certain type of event.

We obtained a month's worth of articles from Google News by crawling. Each article was cleaned of all HTML markup, advertisements, navigation and similar. Articles were grouped into stories according to Google News.

For each *story*, a semantic graph was constructed. The reason to use an aggregate story graph rather than individual article graphs was twofold. First, by representing each story as a single graph, all stories were represented equivalently (as opposed to the case where each article contributed a graph, resulting in stories being weighted proportionally to the number of their articles). Second, the method for extracting triplets has relatively low precision and recall; it therefore makes sense to employ the redundancy inherent in the collection of articles reporting on the same event. To construct the aggregate story graph, we simply concatenated the plain text of individual articles; aggregation at this early stage has the added benefit of providing cross-article entity resolution. Finally, the collection of semantic graphs from stories on a single topic was input to the pattern mining algorithm.

We defined five topics on which to observe the behavior of the method: bomb attacks, award ceremonies, worker layoffs, political visits and court sentencings. For each, we identified about 10 stories of interest. Note that each story further comprises about 100 articles, clustering courtesy of Google News; in total, about 5000 articles were therefore processed.

As semantic graphs were constructed on the level of stories rather than articles, their structure was relatively rich. They had about 1000 nodes each and an average node degree of roughly 2.5. The 20% most connected nodes, which are also the ones likely to appear in the patterns, had an average degree of about 20.

For each topic, graphs of all its stories were input to the algorithm. The minimal pattern support was set at 30% for all the topics. The algorithm output several patterns for each topic; the sizes of the outputs along with the interesting patterns are presented in Figure 2.

For instance, the last person in the "visits" domain shows that in at least 30% of the stories, there was a male person ("he", e.g. Obama) who traveled to France (a coincidence), and that same person met a "leader" (a president in some of the stories, a minister in other).

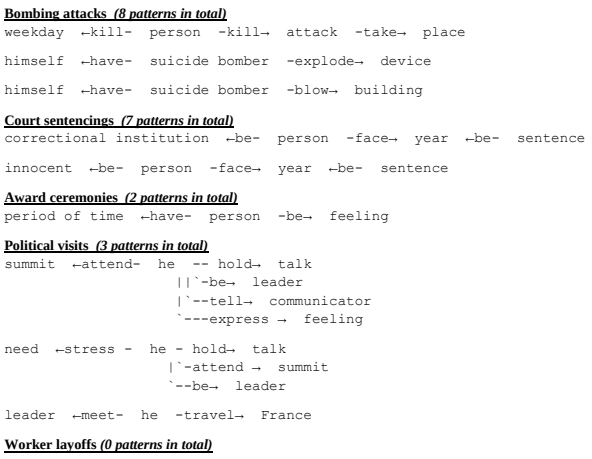

Figure 2: Manually selected best patterns for each domain.

5. DISCUSSION AND FUTURE WORK

The preliminary results seem sound. The mappings of individual stories onto the patterns (not given here) also provide a semantically correct alignment. We can observe how each story fits the template with slightly different entities. Sometimes, the variations are almost imperceptible – "correctional facility" from the "court" domain, for example, appears as either "jail" or "prison", which for some reason are two distinct concepts in WordNet.

In other cases, the distinctions are significant and express the subtopical diversity we were looking for. For example, the groundings for "leader" in the "visits" domain varied even in our small dataset over president, minister, instigator or simply leader. In the same domain, "feeling" was either sorrow, disappointment or satisfaction. The "building" in the "bombings" domain was generalized from mosque, restaurant, hotel and building. It might be interesting to investigate this further and use the amount of variation between pattern groundings as a measure of pattern interestingness.

Unexpectedly, diversity can occasionally be found in the natural clustering that the patterns provide. Observe the two patterns in the "court" domain: in both, the defendant is facing a sentence of (one or more) years, but is found innocent in one cluster and sent to(?) the jail in the other.

While the current experiments are too small to draw any conclusive evidence, we can make some speculations about precision and recall. While the first is low but usable (a data analyst should not mind going through e.g. 5 patterns to identify a useful one), the latter seems a bigger issue. We hope to improve the results significantly by developing a better triplet extractor[1]; the previously discussed deficiencies of current triplets appear to hit performance most.

The tests also indicate that the method is not equally suitable for all domains. The "layoffs" domain, for example, had no single pattern which would occur in 30% of the stories. (A threshold of 25% produces a single rather nonsensical pattern "it—cut⟶job⟵lose—people"). The "awards;; domain does not fare much better. Most probably, these two topics are too broad, causing stories to have only little overlap.

[1]But this is a new project in itself.

Note that in current implementation, all final patterns with less than three nodes (e.g. `worker..lose..job` for the "layoffs" topic) were discarded. Partly this is because we are, in perspective, interested in (dis)proving that structured patterns can provide more information than sentence-level patterns found in related work[2]. Partly, however, it is also because including two-node patterns would introduce additional noise in the output. Even now, the precision is relatively low; it would therefore be interesting to investigate measures of interestingness of patterns other than raw frequency.

6. ACKNOWLEDGMENTS

This work was supported by the Slovenian Research Agency and the IST Programme of the EC under PASCAL2 (IST-NoE-216886), ACTIVE (IST-2008-215040) and RENDER (FP7-257790).

7. REFERENCES

[1] A. Arasu and H. Garcia-Molina. Extracting structured data from web pages. pages 337–348, 2003.

[2] S. Brin. Extracting patterns and relations from the world wide web. *Lecture Notes in Computer Science*, pages 172–183, 1999.

[3] Y. Chi, S. Nijssen, R. Muntz, and J. Kok. Frequent subtree mining-an overview. *Fundamenta Informaticae*, 66(1):161–198, 2005.

[4] E. Filatova, V. Hatzivassiloglou, and K. McKeown. Automatic creation of domain templates. In *Proceedings of COLING/ACL 2006*, pages 207–214, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

[5] D. Klein and C. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 423–430. Association for Computational Linguistics, 2003.

[6] P. Li, J. Jiang, and Y. Wang. Generating templates of entity summaries with an entity-aspect model and pattern mining. *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 640–649, 2010.

[7] K. Probst, R. Ghani, M. Krema, A. Fano, and Y. Liu. Semi-supervised learning of attribute-value pairs from product descriptions. pages 2838–2843, 2007.

[8] D. Rusu, B. Fortuna, M. Grobelnik, and D. Mladenić. Semantic Graphs Derived From Triplets With Application In Document Summarization. *Informatica Journal*, 2009.

[9] H. Tanev and B. Magnini. Weakly supervised approaches for ontology population. 2006.

[10] M. Trampuš and D. Mladenić. Learning event templated from news articles. In *Proceedings of SiKDD09*, 2009.

[11] X. Yan and J. Han. gSpan: Graph-based substructure pattern mining. page 721, 2002.

[2]The "visits" domain is a nice indication that this may be true.