DIDO: a Disease-Determinants Ontology from Web Sources

Victoria Nebot*
Dept. de Lenguajes y Sistemas Informáticos
Universitat Jaume I, Castellón, Spain
romerom@lsi.uji.es

Min Ye, Jae-Hong Eom,
Gerhard Weikum
Max-Planck Inst. CS, Saarbruecken, Germany
{mye, jeom, weikum}@mpi-inf.mpg.de

ABSTRACT

This paper introduces DIDO, a system providing convenient access to knowledge about factors involved in human diseases, automatically extracted from textual Web sources. The knowledge base is bootstrapped by integrating entities from hand-crafted sources like MeSH and OMIM. As these are short on relationships between different types of biomedical entities, DIDO employs flexible and robust pattern learning and constraint-based reasoning methods to automatically extract new relational facts from textual sources. These facts can then be iteratively added to the knowledge base. The result is a semantic graph of typed entities and relations between diseases, their symptoms, and their determining factors, with emphasis on environmental factors but covering also molecular determinants. We demonstrate the value of DIDO for knowledge discovery about causal factors and properties of complex diseases, including factor-disease chains.

Categories and Subject Descriptors

H.0 [**Information Systems**]: General

General Terms

Knowledge Extraction, Knowledge base

General Terms

Algorithms

Keywords

Relation Extraction, Ontology, Biomedical Knowledge Base, Disease Factors

1. INTRODUCTION

Motivation. Recently, very large common-sense knowledge bases (KBs) have been automatically built by extracting entity-relationship-oriented facts from Web sources. Examples are DBpedia [8] and YAGO [13] on the research side, and `freebase.com` and `trueknowledge.com` on the commercial side. All of these have formal knowledge representations, using the RDF data model; therefore, they are an enabling asset for semantic services such as question answering, reasoning and explanation, and knowledge discovery. The methodologies for constructing them include rule-based extraction from high-quality but informal knowledge portals like Wikipedia, and also pattern-based gathering of facts from natural language texts (see [9, 16] for an overview and further references).

It is an intriguing idea to adopt these methods for building similarly structured, semantically rich KBs for specific domains. This paper considers biomedical knowledge, and the goal of capturing it in the form of typed entities (e.g., diseases, environmental factors) and typed relationships between them (e.g., `hasSymptom`, `createsRiskFor`).

In the biomedical domain, in fact, there are already plenty of biomedical knowledge collections available. However, they either focus on highly specialized aspects (e.g., protein interactions, gene expression, metabolic pathways), or merely provide relatively crude taxonomies that organize entities in a sub-concept hierarchy and have limited semantically expressive relations. Some examples of the former are MIPS[4], GO[1], or KEGG[2]; the latter are covered by MeSH[3] and UMLS[7] among others. All of them are essentially handcrafted and extensively curated by human experts.

The difficulty of carrying the methods for common-sense knowledge over to the specific world of biomedical relations stems from two problems. First, there is a huge diversity in the names by which we refer to biomedical entities especially for diseases and their symptoms and contributing factors. Second, the relations that we are interested in can be expressed in many different and sophisticated ways (e.g., extraction of "*chestnutTreePollen* `createsRiskFor` *Asthma*" relation from full text). This is in contrast to simple common-sense relations such as `hasBirthPlace` or `isMarriedTo`, where a few canonical patterns are sufficient for accurate extraction.

In the general biomedical domain, there are some approaches that investigate the relations between diseases and their various causing factors in rather broad contexts [11]. Approaches combining text mining and reasoning have also been tackled [10, 15]. However, there are none or very limited ontological semantics involved in their knowledge representations.

Contribution. This paper presents DIDO (**Di**sease-**D**eterminants **O**ntology), a system that extracts biomedical knowledge from textual Web sources, with emphasis on disease-factors relationships, using tailored methodologies. We describe how to automatically construct a large, ontologi-

*This research has been partially funded by the Spanish National Research Program (TIN2008-01825/TIN). The author was supported by the PhD Fellowship Program of the Education Ministry (AP2007-02569).

WWW 2011, March 28–April 1, 2011, Hyderabad, India.
ACM 978-1-4503-0637-9/11/03.

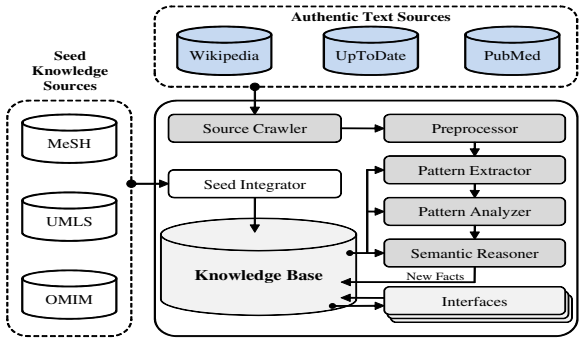

Figure 1: System Architecture of DIDO.

cally clean knowledge base on diseases by integrating information from a variety of hand-crafted, structured sources, and present new methods for pattern-based extraction of relationships, specifically geared for biomedical entities and otherwise barely covered relations. We also demonstrate the practical viability and good accuracy of the relation extraction methods by discovering disease-symptom and disease-factor pairs in a variety of input sources including Wikipedia pages, PubMed research articles, and natural language texts from the medical portal `uptodate.com`.

Figure 1 illustrates the architecture and components of DIDO. The knowledge base (KB) is bootstrapped with entities from the sources on the left-hand side. Subsequently, we use our pattern-learning and consistency-reasoning methods to tap into textual sources shown at the top. DIDO has a form-based query interface, with SPARQL-like functionality, and a visual browser to explore the KB.

Structure of the Paper. Section 2 describes the backbone of the system, i.e. the KB, and how it has been constructed by integrating different sources. Section 3 presents the methods for growing the KB by automatically gathering new relationship facts from textual sources. Section 4 discusses different demonstration scenarios for accessing and exploring the KB, to highlight the usefulness of the system.

2. KNOWLEDGE BASE

For the representation of facts, we follow the model of YAGO [13], which is able to express entities and relations between entities (referred to as facts) in RDFS.

The basic backbone of entities is composed by MeSH descriptors organized by the relations `subClassOf` and `type`. MeSH is a controlled thesaurus containing medical terms organized in a hierarchical structure. Although MeSH is comprehensive in some sub-domains (e.g. chemicals), it lacks specificity in many other domains of interest. Therefore, we have augmented and integrated MeSH with concepts from OMIM[5] and UMLS in order to increase the overall entity coverage. In particular, OMIM diseases have been aligned with MeSH diseases. Moreover, a list of symptoms and related body parts have been automatically extracted from OMIM and linked to MeSH branches 'Signs and Symptoms' and 'Anatomy', respectively. Finally, UMLS descendants of MeSH leaf terms have also been incorporated into the KB.

3. DYNAMIC KNOWLEDGE GATHERING

After the integration process of different types of biomedical entities into a hierarchical and clean taxonomy, the DIDO system is able to populate the KB with instances of a given set of binary relations of interest, provided that there exists a small set of seed facts for the target relations. For example, if we bootstrap the KB with seed facts such as "*Asthma* `hasSymptom` *Coughing*" or "*Smoking* `createsRiskFor` *Asthma*", we can then automatically find good extraction patterns and new facts about symptoms and risk factors of other diseases. The main components of this fact gathering process are explained in the following subsections.

3.1 Entity Detection

This phase is mainly concerned about finding textual or linguistic patterns in the underlying sources as indicators of facts. It can be decomposed into two crucial sub-steps: finding potential candidate entities and gathering the right patterns among the entities.

We have designed different configurations for finding candidate entities in the text, entity mapping and pattern-extraction to extract both the right entities and textual patterns. All of them require only a single traversal of the corpus so that it is scalable to large datasets. For efficiency, we build on a series of shallow linguistic processing techniques such as sentence boundary detection, part-of-speech (POS) tagging and noun phrase (NP) chunking.

Preprocessing. Given a textual corpus, e.g. a set of Wikipedia articles or HTML pages from the Web, we detect and split the text into different structural units such as title, paragraphs, sections, and sentences. We then run a POS tagger[6] and annotate each sentence with POS tags. To detect potential entities in sentences, we use an NP chunking tool to extract NP chunks.

Entity Candidates. To identify potential entities, the standard approach is to take the NP chunks. However, this approach suffers from recall issues, especially when the target entities do not have a canonical representation. Therefore, we have designed and applied a series of heuristic rules to account for possible mistakes of the NP chunker and increase the number of candidate entities:

RULE 1: split NPs joined by coordinating conjunction (i.e. 'and', 'or', etc); e.g., [*chest tightness and coughing*] → [*chest tightness*], [*coughing*]

RULE 2: join NPs separated by preposition or subordinating conjunction (i.e. 'of', 'from', 'for', 'in', etc); e.g., [*shortness*] of [*breath*] → [*shortness of breath*]

RULE 3: consider alternative candidate NPs by taking just the nouns and adjectives of the NP; e.g., [*increased asthma morbidity*] → [*asthma morbidity*] .

RULE 4: consider verbs ending in *-ing* as candidate NP; e.g., [*coughing*], [*sweating*], etc.

Entity Mapping and Selection. Each of the previous candidate entities needs to be mapped to a canonical entity in the KB. For this purpose, we designed a hybrid approach which is flexible enough to account for different variations of the terms and allows approximate matching while at the same time being able to discard lexically similar but not meaningful mappings. The algorithm works as follows:

Step 1: For each candidate string (entity-name mention), retrieve the highest ranked n potential mappings to entities from the KB, using the full text matching function implemented in MySQL.

Step 2: Re-rank the selected target entities according to the following score:

$$sc(c) = 0.6 \times ngram\_score + 0.4 \times onto\_context\_score$$

where *ngram_score* is obtained by computing the n-gram similarity between the candidate string and the potential target entity using the Jaccard's similarity coefficient, while *onto_context_score* is calculated by computing the word overlap of the surrounding entities of the target entity in the KB with the words of the document containing the candidate string.

This score offers a good trade-off between lexical and semantic similarity. Once the target entities are ranked, the mappings still need to be identified as good or discardable. For this purpose, the candidate strings overlapping in the text are grouped together, and for each group we implemented two approaches:

1: select the candidate with the highest score and shortest length.

2: select a subset of candidates such that they do not overlap and the sum of their scores is maximal.

3.2 Pattern Analysis and Fact Candidates

Once entity-name mentions in the input corpus are detected and mapped to the most appropriate entities in the KB, we perform a statistical analysis on the textual phrases that connect entity mentions in order to derive patterns that indicate the presence of an ontological relation.

Pattern Candidates. Every sentence with two or more selected entities serves as a source for potential patterns. We consider as a pattern candidate every sequence of words between two entities in a sentence.

Pattern Analysis. The collected patterns are fed into a frequent n-gram-itemset mining algorithm for identifying strong patterns (see [12]). Seed patterns are patterns whose support and confidence based on their co-occurrence with seed facts is above a specified threshold. Table 1 shows some examples of the learned patterns.

The learned patterns are then applied to gather fact candidates for the relations of interest, by matching the patterns against the sentences of the entire corpus or new textual sources. For each pattern match, we extract the entities connected by the pattern occurrence (using our methods described above). This pair of entities then forms a new fact candidate, whose weight is the highest pattern-matching similarity between its own pattern and any seed pattern using n-gram based Jaccard coefficient.

Table 1: Examples of Learned Patterns

Relation	*Pattern*
`hasSymptom`	{include}
`aggravates`	{can worsen}
`causes`	{can trigger}
`causes`	{may trigger}
`causes`	{may increase, of developing}
`createsRiskFor`	{may develop}
`createsRiskFor`	{is estimated to cause}
`createsRiskFor`	{may increase, of developing}
`reducesRiskFor`	{to prevent}
`reducesRiskFor`	{used to treat}
`reducesRiskFor`	{that is applied to}

3.3 Semantic Reasoning

We use constraint-based reasoning to eliminate a large fraction of false positives among the fact candidates. To this end, we use the SOFIE framework based on approximate MaxSat solving [14]. We have introduced a set of rules specifically geared for our biomedical target relations. Examples of these rules are shown in Table 2. In this table, R_P represents a set of positive relations ({`reducesRiskFor`, `alleviates`, `heals`}), while R_N stands for a set of negative relations ({`createsRiskFor`, `causes`, `aggravates`}). R_O and R_{rel} represent relation sets {`observedIn`, `hasSymptom`} and {`relatedTo`}, respectively. All these constraint rules hold for entity arguments (x and y) having valid domain and range. This form of semantic type checking is very beneficial for eliminating nonsensical fact candidates (e.g., "*Asthma* `reducesRiskFor` *HeartAttack*", which is discarded because `reducesRiskFor` cannot have two diseases or symptoms as arguments). Moreover, this pruning also contributes to the efficiency of the reasoner.

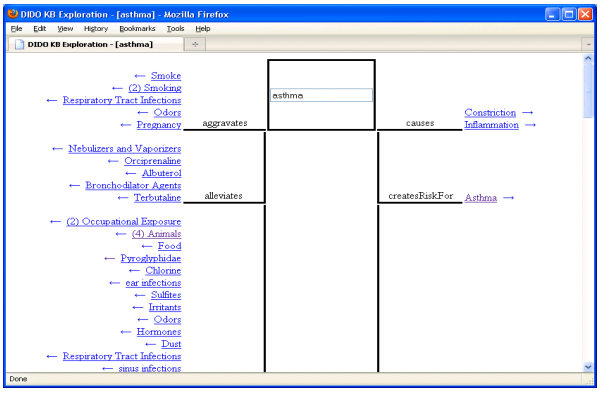

Figure 2: Example of Exploration Interface.

Table 2: Constraint Rules

Rule #	*Rule*
CR1	$R_N(x,y) \Rightarrow \neg R_{rel}(x,y) \wedge \neg R_P(x,y) \wedge \neg R_O(x,y)$
CR2	$R_P(x,y) \Rightarrow \neg R_{rel}(x,y) \wedge \neg R_N(x,y) \wedge \neg R_O(x,y)$
CR3	$\texttt{hasSymptom}(x,y) \Rightarrow \neg R_{rel}(x,y) \wedge \neg R_P(x,y) \wedge \neg R_N(x,y) \wedge \neg\texttt{observedIn}(x,y)$
CR4	$\texttt{observedIn}(x,y) \Rightarrow \neg R_{rel}(x,y) \wedge \neg R_P(x,y) \wedge \neg R_N(x,y) \wedge \neg\texttt{hasSymptom}(x,y)$

The rules in Table 2 have proved to ensure the logical consistency of the KB by pruning candidate facts that are unavoidably collected during the pattern gathering mainly due to the complexity and length of the input sentences.

Table 3: Examples of New Facts

Subject	*Relation*	*Object*
Cholera	`hasSymptom`	Irritability
Cholera	`hasSymptom`	SunkenEyes
Cholera	`hasSymptom`	Lethargy
Dust	`causes`	Rhinitis
Aspirin	`reducesRiskFor`	Swell
Hypersensitivity	`causes`	Asthma
AirPollutionIndoor	`causes`	Asthma
Anti-InflammatoryAgents	`reducesRiskFor`	Pain

Table 3 shows some examples of new facts that have been dynamically extracted from textual sources with the outlined method. Although the reasoner only assigns truth values to fact candidates, the statistical weights assigned to them in the previous phase are propagated when grounding the rules so that the reasoner can make a more informed decision. Therefore, these weights serve as a measure of the goodness of the new extracted fact.

4. DEMONSTRATION SCENARIOS

We provide several access points to the KB that allow both interactive exploration and querying as well as incremental online gathering of new facts.

4.1 KB Exploration

Through the browsing tool shown in Figure 2 the user can type the name of an entity (we allow partial matching) and the direct relations to other surrounding entities are displayed in an intuitive layout. For example, if the user is interested in discovering information regarding "*diabetes mellitus*", (s)he simply types the name of the disease and the system will render all the entities directly related to *diabetes mellitus* along with their corresponding relations. In this case, the output would include facts such as:

BreastFeeding	`alleviates`	*DiabetesMellitus*
Pregnancy	`createsRiskFor`	*DiabetesMellitus*
DiabetesMellitus	`hasSymptom`	*HeartAttack*

Moreover, information about the hierarchical structure of the typed entity (i.e. sub- and superclasses) is also displayed when appropriate. For the previous example the facts displayed include:

DiabetesMellitusT1	`subClassOf`	*DiabetesMellitus*
DiabetesMellitus	`subClassOf`	*GlucoseDisorders*

By clicking on any of previous related entities, the chosen entity becomes the current focus of analysis and all its directly connected entities will be displayed automatically.

Another example of KB exploration centered on behavioral factors instead of diseases could be the entity "*Smoking*". By typing this entity, the user can easily find out which diseases are related to this factor. In this case, the output includes facts such as:

Smoking	`aggravates`	*CoronaryDisease*
Smoking	`causes`	*PregnancyComplications*
Smoking	`createsRiskFor`	*LungDiseases*

4.2 KB Querying

For more detailed and in-depth KB querying and exploration, the system offers a form-based query interface with SPARQL-like functionality. This tool allows executing sophisticated queries where one can narrow down or combine different search conditions for the relations of interest. For example, given two or more known diseases or phenomena (e.g. *Lung Neoplasms*, *Skin Aging* and *Cravings*), the user could be interested in common factors causing these diseases and phenomena. Given the following query:

?x	`causes`	*LungNeoplasms*
?x	`causes`	*SkinAging*
?x	`createsRiskFor`	*Cravings*

the system returns results like: *?x = Smoking*. Another interesting query with the result *?x = Stroke* is shown below:

DiabetesMellitus	`createsRiskFor`	*?x*
CardiovascularDiseases	`causes`	*?x*

Finally, we show an example query to discover even more intertwined interactions between environmental factors, diseases, symptoms, as well as drugs:

DiabetesMellitusType1	`createsRiskFor`	*?x*
?x	`causes`	*?y*
?z	`reducesRiskFor`	*?y*
?z	`reducesRiskFor`	*Swell*

the results for the given query are:

?x	=	*Cardiovascular Diseases*,
?y	=	*Heart Attack*,
?z	=	*Aspirin*

4.3 Incremental Online Fact Harvesting

DIDO also provides incremental ad-hoc gathering of new facts for specific entities. This interactive fact harvesting is currently supported only on limited-size corpora (e.g., abstracts of the latest 1,000 PubMed articles). This way, DIDO can augment the KB coverage and keep it up-to-date.

5. CONCLUSIONS

DIDO is an ongoing project. Our architecture is geared for scalable gathering of new facts from large Web sources. We are in the process of performing such extractions at larger scale. We are also working on further strengthening the resulting precision (absence of false positives) by refining our use of consistency reasoning. Last but not least, additional visualization capabilities will be supported soon for the evaluation of the effectiveness of the method.

6. REFERENCES

[1] GO: The gene ontology. `http://www.geneontology.org/`.

[2] KEGG:. `http://www.genome.jp/kegg/`.

[3] MeSH: Medical Subject Headings. `http://www.nlm.nih.gov/mesh/`.

[4] MIPS: The Mammalian Protein-Protein Interaction Database. `http://www.test.org/doe/`.

[5] OMIM: Online Mendelian Inheritance in Man. `http://www.ncbi.nlm.nih.gov/omim/`.

[6] Stanford Log-linear Part-Of-Speech Tagger. `http://nlp.stanford.edu/software/tagger.shtml`.

[7] UMLS: Unified Medical Language System. `http://www.nlm.nih.gov/research/umls/`.

[8] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. DBpedia: a nucleus for a web of open data. In *Proc. of ISWC '07*, 2007.

[9] A. Doan and et al. (Eds.). Special Issue on Information Extraction. *ACM SIGMOD Record*, 37(4), 2008.

[10] J.-H. Kim, A. Mitchell, T. K. Attwood, and M. Hilario. Learning to extract relations for protein annotation. *Bioinformatics*, 23(13):i256–63, 2007.

[11] Y. I. Liu, P. H. Wise, and A. J. Butte. The "etiome": identification and clustering of human disease etiological factors. *BMC Bioinf.*, 10(S2):S14, 2009.

[12] N. Nakashole, M. Theobald, and G. Weikum. Find your Advisor: Robust Knowledge Gathering from the Web. In *Proc. of WebDB '10*.

[13] F. M. Suchanek, G. Kasneci, and G. Weikum. YAGO: A Core of Semantic Knowledge. In *Proc. of WWW '07*.

[14] F. M. Suchanek, M. Sozio, and G. Weikum. SOFIE: A Self-Organizing Framework for Information Extraction. In *Proc. of WWW '09*.

[15] L. Tari, S. Anwar, S. Liang, J. Cai, and C. Baral. Discovering drug-drug interactions: a text-mining and reasoning approach based on properties of drug metabolism. *Bioinformatics*, 26(18):i547–53, 2010.

[16] G. Weikum and M. Theobald. From Information to Knowledge: Harvesting Entities and Relationships from Web Sources. In *Proc. of PODS '10*, 2010.