6-2011

Automatic Domain Model Creation Using Pattern-Based Fact Extraction

Christopher Thomas
Wright State University - Main Campus

Pankaj Mehra

Wenbo Wang
Wright State University - Main Campus, wang.112@wright.edu

Amit P. Sheth
Wright State University - Main Campus, amit.sheth@wright.edu

Gerhard Weikum

See next page for additional authors

Repository Citation

Thomas, C., Mehra, P., Wang, W., Sheth, A. P., Weikum, G., & Chan, V. (2011). Automatic Domain Model Creation Using Pattern-Based Fact Extraction. .
http://corescholar.libraries.wright.edu/knoesis/1001

Authors

Christopher Thomas, Pankaj Mehra, Wenbo Wang, Amit P. Sheth, Gerhard Weikum, and Victor Chan

Automatic Domain Model Creation Using Pattern-Based Fact Extraction

Christopher Thomas[1], Pankaj Mehra[2], Wenbo Wang[1], Amit Sheth[1], Gerhard Weikum[3] and Victor Chan[4]
$\{thomas.258, wang.112, amit.sheth\}$@wright.edu,
pankaj.mehra,@hp.com, weikum@mpi-sb.mpg.de, victor.chan@wpafb.af.mil

[1]Knoesis Center, Wright State University, Dayton, OH, USA, [2]HP Labs, Palo Alto, CA, USA
[3]Max Planck Institute Informatik, Saarbrücken, Germany and [4]AFRL Wright Patterson AFB, Dayton, OH

Abstract

This paper describes a minimally guided approach to automatic domain model creation. The first step is to carve an area of interest out of the Wikipedia hierarchy based on a simple query or other starting point. The second step is to connect the concepts in this domain hierarchy with named relationships. A starting point is provided by Linked Open Data, such as DBPedia. Based on these community-generated facts we train a pattern-based fact-extraction algorithm to augment a domain hierarchy with previously unknown relationship occurrences. Pattern vectors are learned that represent occurrences of relationships between concepts. The process described can be fully automated and the number of relationships that can be learned grows as the community adds more information. Unlike approaches that are aimed at finding single, highly indicative patterns, we use the cumulative score of many pattern occurrences to increase extraction recall. The relationship identification process itself is based on positive-only classification of training facts.

Introduction

Formal representations of domain knowledge can leverage classification, knowledge retrieval and reasoning about domain concepts (Vetere, 2009). Two examples for formal representations are carefully designed, rigorous formal ontologies on the one hand and on the other hand there are less rigorous representations of so-called Linked open Data (LoD). The field of ontology was concerned with the essence and categorization of things, not with the things themselves. Our conceptualization of the world and of domains stays relatively stable whereas the actual things we encounter in the world change rapidly. When looking for information it is mostly these individual things and events that are of interest to us. LoD is updated instantaneously and avoids problems of logical consistency, but despite its vastness, many domains are only sparsely described and no domain boundaries are given.

This means that for the above mentioned tasks neither of these approaches are sufficient. Formal ontologies lack availability whereas LoD has actuality and breadth, but lacks depth. Hence it is crucial to develop ways to automatically build focused domain models that take advantage of the availability of LoD and use information extraction techniques to expand the available data.

In this paper, we describe a pattern-based relationship extraction module that adds facts with named relationships to automatically created domain hierarchies in order to create densely connected models. This module is an addition to the Doozer system that automatically creates focused domain hierarchies from a small set of seed terms using Wikipedia as a corpus (Thomas et al., 2008). Wikipedia's category structure resembles the class hierarchy of a formal ontology to some extent, even though many subcategory and category-membership relationships in Wikipedia are associative rather than being strict *subClassOf* or *type* relationships. For this reason we refrain from calling the resulting domain model an ontology. Whereas formal ontologies that are used for reasoning, database integration, etc. need to be logically consistent, well restricted and highly connected to be of any use, domain models for information retrieval can be more loosely connected and allow or even welcome logical inconsistencies. The goal of this work is to automatically build models that are close in quality to formally more rigorous ontologies and will require little or no further human involvement after the initial community-based creation of the background knowledge on Wikipedia. Much of the previous work in fact extraction has taken advantage of learning algorithms that are able to discriminate using positive and negative training examples. Our belief is that it is more realistic to assume only the availability of positive training examples. Additionally, we assume that we also do not have metadata for relationship types. For some relationship types we may assume that they are functional, which would allow us to take every pattern that expresses a different object than the one given in our fact base to be a negative training example, but this requires user input that currently we do not presuppose. Furthermore, we believe that most relationship types are not functional in character and hence can relate a subject to multiple objects. We thus need to find ways of discriminating between appropriate and inappropriate patterns using only positive training examples. With large corpora of textual and factual knowledge available, such as Wikipedia/DBPedia and MedLine/UMLS, we have training data at a scale that allows the use of large numbers of surface patterns rather than building generalizing and approximating models (Anderson, 2008). In order to make a case for a web-scale application, it is important to keep a tractable performance of the algorithm in mind. The algorithms described in this paper are at most of cubic asymptotic complexity, which is even alleviated by the sparsity of the data.

The running example for domain model creation is a model of Human Performance and Cognition. The first step is to create a domain hierarchy from which a more connected model will be generated in the second step using the pattern-based extraction technique. The paper is structured as follows. The next section discusses related work. Then the algorithms for domain hierarchy creation and connection are described and subsequently evaluated, before concluding the paper.

Related Work

Model Creation: Ponzetto and Strube (2007) take the Wikipedia category hierarchy and uses heuristics and NLP methods to identify those inter-category relationships that are actually is_a relationships. YAGO (Suchanek, Kasneci, and Weikum, 2008) combines knowledge from WikiPedia and WordNet to achieve a 95% accuracy in its facts. Both these efforts are concerned with large, encompassing knowledge bases, whereas our project aims at identifying focused topics of interest.

Pattern-based Fact Extraction: Pattern-based information extraction has been successfully applied to many different tasks. Hearst (Hearst, 1992) identified patterns that indicate hyponym relationships. This line of work has been extended to extracting more general relationships in systems such as KnowItAll (Etzioni et al., 2004). However, these approaches rely on manually identified patterns. Other work, such as LEILA(Suchanek, Ifrim, and Weikum, 2006) or (Ramakrishnan, Kochut, and Sheth, 2006), takes advantage of strong linguistic analysis of the corpus.

The pattern-based approach taken in this work is inspired by P.D.Turney's Turney (2006) work on identifying analogous word pairs. Turney uses simple strict and generalized patterns without first parsing or POS-tagging the text. Also considering the efforts around ConceptNet and AnalogySpace (Speer, Havasi, and Lieberman, 2008), it seems that solely pattern-based methods reach a high level of certainty in predicting very basic kinds of relations that may serve as analogies to more specific types of relationships, a hypothesis that has also been discussed e.g. in (Lakoff and Johnson, 1980) and seems to find some grounding in actual use of natural language.

The authors agree with Wang et al. Wang, Yu, and Zhu (2007) that it is important and more practical to devise an algorithm that can work on positive examples only, but we also see a necessity in using completely unstructured text rather than taking advantage of the structure of Wikipedia pages that start with a short entity description in the first sentence of the article.

Domain Hierarchy Creation

The Doozer system described in Thomas et al. (2008) carves a domain model out of the Wikipedia article- and category graph. The Wikipedia corpus contains a vast category graph on top of its articles. Though these categories do not constitute a formal class hierarchy, they nevertheless closely approximate it. The task here is to carve out a domain hierarchy that clearly focuses on user-interests. This process follows an expand and reduce paradigm that allows us to first explore and exploit the concept space before reducing the concepts that were initially deemed interesting to those that are closest to the actual domain of interest.

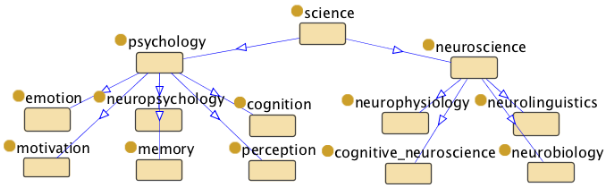

Figure 1: Top Categories in the Human Performance Model

We present an example of a domain hierarchy generated for a cognitive science project. The seed query/focus domain consisted of a selection of pertinent terms to the particular area of cognitive science/neuroscience with special focus on brain chemistry that are important for mental performance. The full hierarchy that was built from these inputs is too large to appropriately show here, but the small excerpt of the top categories in the hierarchy (Figure 1) gives a good idea of the complexity of the generated model.

Concept Identifiers and Synonyms

The Wikipedia article names/URIs are unambiguous identifiers of the concepts. However, the domain model needs to contain different synonyms for the article URIs to be able to identify the concepts in actual text when performing fact extraction. Matching to a lexicon such as WordNet (Fellbaum, 1998) to acquire synonyms is incomplete and adds another level of uncertainty. We determined that the anchor texts of Wiki-internal links are good indicators of synonyms. The probability that a term is a descriptor of an article concept is given by the probability that the term is an anchor text in a link to the article (see Equation 1). These concept identifiers and their probabilities are used to prune the model and as search terms for fact extraction.

$$p_{syn}(term, article) = \frac{|links\_to(term, article)|}{\sum_{a \in AllArticles} |links\_to(term, a)|} \tag{1}$$

Fact Extraction

This section focuses on the fact extraction needed to create connected domain models with named relationships. We define a fact as a statement connecting a subject to an object through a relationship. The relationship types are those available on the LoD cloud, such as DBPedia (Auer et al., 2007) or the unified medical language system (UMLS). These sources provide training facts for the classifier. Refraining from using computationally expensive NLP techniques, we rely completely on analyzing surface patterns that indicate facts.

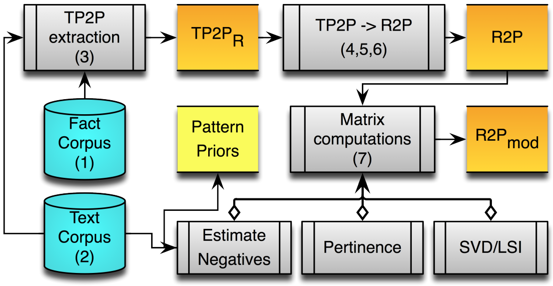

Figure 2: Pattern extraction workflow

These patterns are represented in a (Concept-Pair, Pattern) matrix $CP2P$. Each row in this matrix is a vector representing all the patterns in which a concept pair appears. The concept pairs are taken from the fact corpus, in our case LoD-triples. In the training phase the relationship types these concept pairs appear in are accumulated into a (Relationship, Pattern) matrix $R2P$. This matrix can be seen as a static representation of relationship mentions in text. In the application phase patterns between previously unseen concept pairs are compared to $R2P$ to yield candidate relationship types the concept pair participates in.

The simplifying assumption here is that the concepts participating in the relations are known a priori, as well as their surface representations, i.e. their labels. Patterns are learned and applied in the cases in which they appear between entities. These contextual patterns are thus more accurate and succeed even though the same pattern occurring in contexts other than between two named entities would fail. The pattern-learning algorithm can be broken down into the tasks shown in Figure 2.

Matrix Acquisition

(1) and (2) Fact corpus and text corpus acquisition: Take known facts from the repository on the LoD dataset that closely represents the domain of interest and search for possible occurrences of textual representations of these facts in the text corpus (e.g. Wikipedia, MedLine or WWW pages in general).

(3) Build a matrix $CP2P_R$ that maps concept pairs from a training set (annotated with the relationship they occur in) to their representation in a text corpus in the form of patterns. Patterns are phrases of the form [*Prefix*]$<T_1>$[*Infix*] $<T_2>$[*Postfix*] with T_1 and T_2 indicating any pair of concept labels found using Equation 1 that denote subject and object concepts of the triple respectively. The pattern is added to the dictionary by replacing term pairs with subject and object placeholders. For example, the phrase Albert Einstein was born in Ulm is added to the pattern dictionary as "$<Subject>$was born in $<Object>$". The $CP2P_R$ matrix contains occurrence frequencies of these raw patterns.

(4) Generalization: An example is shown in Table 1. A number of tokens in the pattern can be replaced by dont-care characters. We found that more than 3 generalizations per 5 token pattern yields no information gain. A generalized pattern can have multiple raw patterns as parents, in which case the frequencies of these are added to get an estimated frequency of the generalized pattern.

Table 1: Generalization example

<subject>	,	the	largest	city	in	*	<object>
<subject>	,	the	largest	city	*	the	<object>
<subject>	,	the	largest	*	in	the	<object>
<subject>	,	the	largest	*	in	*	<object>
<subject>	,	the	*	city	in	the	<object>
<subject>	,	the	*	city	in	*	<object>
<subject>	,	*	largest	city	in	the	<object>

(5) Minimization: prune infrequent patterns to reduce noise.

(6) Build a representation for relationship types: The RelationshipPattern Matrix R2P is then built by adding up all concept pair vectors in $CP2P_R$ that indicate one relationship type into one row vector in $R2P$. In this matrix, the rows represent named relationships and the columns represent the surface patterns these relationships occur in. Each field a_{ij} indicates the frequency of a pattern Pj indicating a relationship R_i.

(7) Compute relationship probabilities: Ultimately, the fields in the $R2P$ matrix should contain the conditional probabilities of a pattern unambiguously expressing a relationship (see Equation 2). The following section describes the details of the $R2P$-matrix computations.

$$\begin{aligned} p(R_i|P_j) &= \frac{p(P_j|R_i)}{\sum_{k=1}^{m} p(P_j|R_k)} \\ where & \\ p(P_j|R_i) &= \frac{|P_j \cap R_i|}{|R_i|} \end{aligned} \tag{2}$$

Matrix Computations

So far we implicitly assumed here that a relationship is determined by subject and object. This obviously only holds in few cases. For example, a person can be born and die in the same place. Even if only one of these facts is in the fact corpus, both can be expressed in the text corpus. In this case patterns for both the *birthplace* and *deathplace* relationships can be found although the algorithm only knows of one relationship. Since LoD only contains positive facts, we cannot rely on negative examples to resolve these ambiguities but need to find computational solutions. Despite the occurrences of multiple training facts that share the same subjects and objects but have different relationships, it is likely that over the full training corpus there will be an emphasis on facts with different subject-object combinations. An analysis of the preprocessed DBPedia Infobox facts shows that out of 3,544,160 facts in the corpus there are 846,574 subject-object pairs that occur together more than once.

Probabilistic Approach

A probabilistic model intuitively answers the question: "Which relationship is expressed when I see these entities in the context of this pattern?" It is also easily verifiable, which makes it an ideal candidate for a prototype application. Using only positive examples we are facing the problem of missing data. However, with the large and ever growing datasets of LoD at hand, The number of relationship types that we classify the pattern occurrences in, also grows and thus our ability to discriminate. The remaining task is to discriminate relationships that seem very similar to the classifier because of the above mentioned problems. The main goal of a positive only classifier is thus emphasizing differences and penalizing similarities between different relationships, while not penalizing similarities between similar relationships.

Pertinence

In the end we are interested in seeing in how far a pattern pertains to a relationship type. The pertinence measure for relationships is conceptually related to (Turney, 2006). It boosts the probability of patterns, if they have a high real-world probability of indicating a specific relationship, even though the pattern is shared among different relationship types. In intensionally similar relationships these patterns should not be penalized. It turns out, as can be seen in table 2 that many intensionally similar relationships are also extensionally similar, as defined by the patterns that express them. Problems arise when seeming extensional similarity arises through multiple relationships with equal subject-object pairs. Thus, the differences between similar relationships must as well be emphasized. This is achieved by first applying an entropy transformation to the $R2P$ matrix to give more weight to highly discriminating patterns. Then, to alleviate the effect of entropy on relationship types that are intensional and extensional similar, the conditional probability (see Equation 2) is modified using a weighted sum in the denominator. The fields in the *R2P* matrix are thus computed according to equation 3, where $p(P_j|R_i)$ is the conditional probability of a pattern indicating a relationship, $cos(R^k_{SVD}, R^i_{SVD})$ is the cosine of the pattern vectors that describe the relationships R_k and R_i in the SVD decomposition of $R2P$.

$$\widetilde{R2P}_{ij} = \frac{p(P_j|R_i)}{\sum_{k=1}^{m} p(P_j|R_k) * f\big(1 - sim_{rel}(R_k, R_i)\big)} \quad where \quad sim_{rel}(R_k, R_i) = cos(R^k_{SVD}, R^i_{SVD}) \tag{3}$$

This equation has the effect that similar relationships do not penalize each others shared patterns, whereas dissimilar relationships that share the same patterns get lower scores for these patterns. The function $f : [0..1] \rightarrow [0..1]$ can be any monotonous weighting function. It has proven useful to use e.g. a logistic function that exaggerates the closeness or distance of 2 vectors. In practice, relationship vectors that have a cosine of > 0.8 are very similar and can be assigned even more confidence whereas many relationships share some basic patterns such that a cosine of < 0.2 should be considered as completely dissimilar. The relationship similarity is computed on the SVD decomposition of the matrix because SVD inherently identifies highly descriptive latent dimensions in the data and helps to reduce noise. The SVD matrix, however, does not follow the probabilistic extraction framework so it is only used for similarity computation.

Table 2: Relationship Similarities

Relationship 1	Relationship 2	Similarity
distributingCompany	distributingLabel	0.999999
associatedBand	associatedMusicalArtist	0.999812
draftteam	formerTeam	0.702442
father	predecessor	0.682084
inflow	outflow	0.667261
birthplace	deathplace	0.650833
capital	largestCity	0.547894
followed_by	subsequentWork	0.531366
currentMembers	pastMembers	0.475924

Matrix-Based Fact Extraction

With the fixed *R2P* matrix we can compute a possible relationship between a concept pair (or multiple concept pairs) by extracting all patterns from a corpus that the concept pair (i.e. the concept denotations) occurs in as described in section , albeit without knowing the relationships the concept pair occurs in, count the number of pattern occurrences, generalize these occurrences and align the patterns found between the concept pairs with those in the *R2P* matrix. Each concept pair is then represented by a row of pattern frequencies. The resulting Termpair to Pattern vectors form a matrix (*CP2P*) that is then normalized according to equation 4. The resulting matrix contains the probabilities of a concept pair occurring with a pattern in the dictionary.

$$\widetilde{CP2P}_{ij} = p(P_j|CP_i) = \frac{CP2P_{ij}}{\sum_{k=1}^{n} CP2P_{ik}} \tag{4}$$

The probability that a concept pair $c_1{::}c_2$ is connected by a relationship R is then the product of the probability that a term pair $\{t_1|t_1 labelof c_1\}{::}\{t_2|t_2 labelof c_2\}$ appears in conjunction with the pattern P and the probability that the pattern indicates R. See Equation 5 and the equivalent matrix multiplication (Equation 6).

$$p(R_j|CP_i) = \sum_{k=1}^{m} p(P_k|CP_i) * p(R_j|P_k) \tag{5}$$

$$\widetilde{CP2P} \times \widetilde{R2P}^T = CP2R \tag{6}$$

Evaluation

For this evaluation we extracted all patterns for all distinct *Subject-Object* pairs in either corpus (DBpedia Infobox and UMLS). The resulting matrix was randomly split into 60% training examples and 40% testing examples. The random splitting was repeated 10 times and the results averaged.

Relationship-Similarity

One of the evaluation criteria is to see whether the algorithm is able to spot similar types of relationships. Table 2 shows that indeed relationships we consider as being similar are

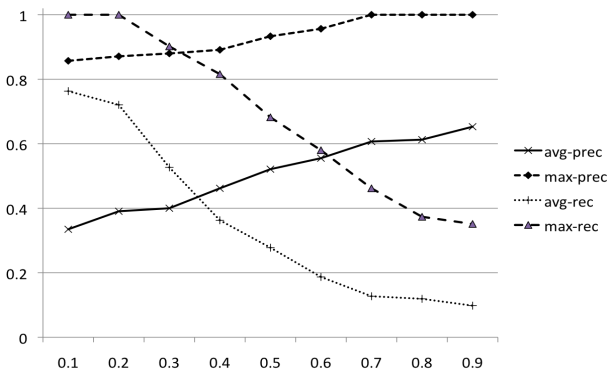

Figure 3: Precision and recall on the DBPedia testing set and the Wikipedia text corpus.

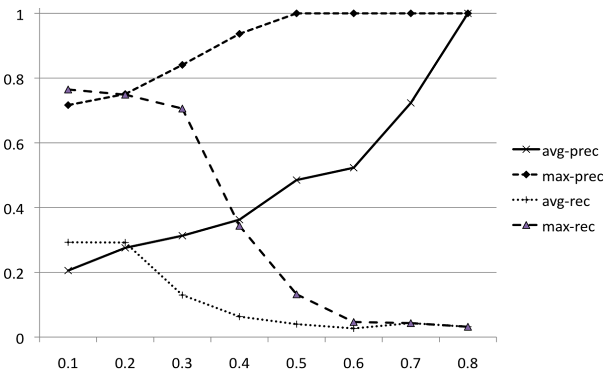

Figure 4: Precision and recall on the UMLS testing set and the MedLine text corpus.

grouped together. Practically, this insight can be used to cluster similar relationships and reduce the number of synonymous relationships. However, we refrain from using this insight to cluster relationships in the extraction evaluation to avoid counting extensionally similar and intensionally dissimilar relationships towards a positive precision.

Fact Extraction Results

Figure 3 shows the automatic evaluation of precision and recall over all cross-evaluation sets of the Wikipedia-DBPedia corpus, Figure 4 does the same for the MedLine-UMLS corpus. 107 relationship types from the DBPedia corpus had enough evidence in Wikipedia to be considered versus 124 UMLS types on the MedLine corpus. Only direct hits in first rank were taken into account. The horizontal axis indicates the confidence cut-off that was used. The average values show the arithmetic mean precision and recall values over all relationship types, the max values show the maximum precision and recall among the relationship types. This tends to lower the values, because the classification performs better on more common relationship types

The evaluation shows some interesting differences between the evaluation sets. Patterns in MedLine that describe UMLS relationship types tend to be more expressive, but also sparser than patterns in WikiPedia that describe DBPedia relationship types. This makes the precision curve steeper and the recall lower. With a random baseline of less than 1% precision in both cases it can be seen that even with a basic probabilistic approach the surface pattern analysis can be used to connect and augment domain models in information retrieval applications and even as suggestions for formal ontologies.

Model Completion

The *R2P* matrix described in the previous section is assumed to be independent of particular text and fact corpora. Using the dictionary associated with its columns, we can extract patterns between any pair of concepts, as long as we find pattern occurrences for it. On the LoD cloud, the DBPedia Infobox dataset contains over 20 Mio. facts, but looking at the full extent of WikiPedia, this still produces a very sparse graph. Using automated fact extraction the graph can be made denser. For the purpose of this paper we apply the fact extraction techniques to small domain models created with Doozer (See section).

In this sense, the fact extraction can be used in two different ways. Either to only find connections between concepts in the existing model or to expand the model by adding new concepts that are connected to the existing ones by relationships important to the domain of interest. Using the same techniques as described in section , a *CP2P* matrix is created containing vectors describing patterns between concept pairs that were found interesting. Figure 5 shows the connected model created for the Human Performance and Cognition domain.

An expert evaluated 415 randomly chosen extracted facts that had a confidence score of 0.7 or higher. Figure 6 shows the scoring. It displays the percentage for each score and cumulative percentages for scores 1-2 (incorrect: 21%) and 3-10 (correct: 79%) respectively. About 30% of the extracted facts was deemed novel and interesting. The scoring rationale is as follows:

7-10: Correct Information not commonly known
5-6: General Information that is correct
3-4: Information that is somewhat correct

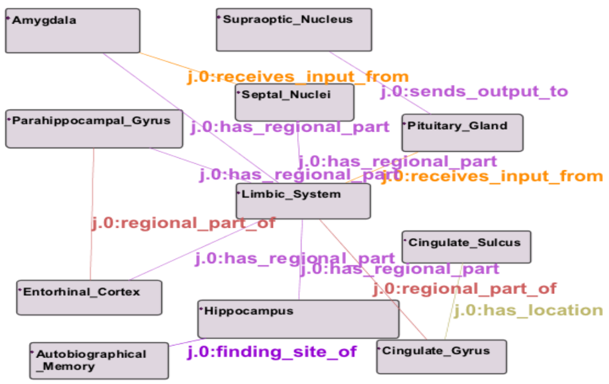

Figure 5: Small excerpt of the connected concept graph. For better visualization, classes have been removed.

1-2: Information that is overall incorrect

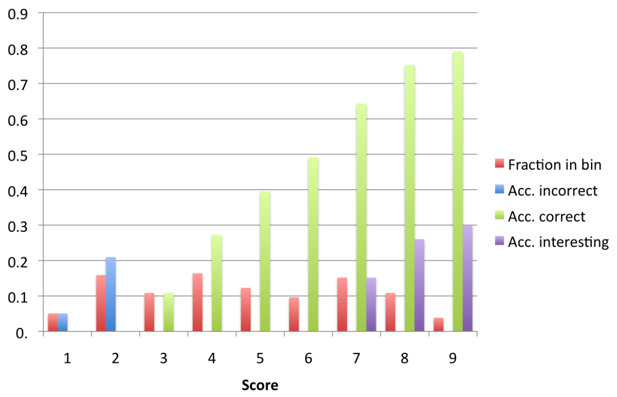

Figure 6: Expert scoring of the previously unknown facts.

Conclusion

Automatic ontology creation is a daunting task. Given the formal requirements and the imperative to include only true knowledge or at least strong belief in an ontology seems to mandate human involvement. In this paper we showed that it is at least possible to create a starting point to leverage the tedious process of ontology engineering and to define domain boundaries in an automated way. Furthermore, the domain models that can be created using the described methods can directly be used for document classification, topic-based information retrieval, focused web search and browsing.

The probabilistic techniques used in this approach allow for human evaluation of the system and for interaction. In future research we want to improve the results by using more advanced classifiers such as SVMs to learn relationship pattern vectors. Furthermore, we can apply domain and range probabilities to each of the relationship types to learn prior probabilities that a concept pair can appear in a relationship. These probabilities can be learned from the subject and object types of DBPedia triples in the Wikipedia category graph.

Acknowledgements

This work has been funded by HP Labs, Palo Alto. Special thanks to Pablo Mendes at the Kno.e.sis Center and Cartic Ramakrishnan at CCHMC for inspired discussions and valuable ideas.

References

Anderson, C. 2008. The end of theory: The data deluge makes the scientific method obsolete. *Wired*.

Auer, S.; Bizer, C.; Lehmann, J.; Kobilarov, G.; Cyganiak, R.; and Ives, Z. 2007. Dbpedia: A nucleus for a web of open data. In *Proceedings of ISWC 2007 (To Appear)*.

Etzioni, O.; Cafarella, M.; Downey, D.; Kok, S.; Popescu, A. M.; Shaked, T.; Soderland, S.; Weld, D. S.; and Yates, A. 2004. Web-scale information extraction in knowitall: (preliminary results). In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, 100–110. New York, NY, USA: ACM.

Fellbaum, C., ed. 1998. *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*. The MIT Press.

Hearst, M. A. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics*, 539–545. Morristown, NJ, USA: Association for Computational Linguistics.

Lakoff, G., and Johnson, M. 1980. *Metaphors We Live By*. Chicago: Chicago University Press.

Ponzetto, s., and Strube, M. 2007. Deriving a large scale taxonomy from wikipedia. In *Proceedings of the 22nd National Conference on Artificial Intelligence (AAAI-07)*, 1440–1447.

Ramakrishnan, C.; Kochut, K.; and Sheth, A. P. 2006. A framework for schema-driven relationship discovery from unstructured text. In *International Semantic Web Conference*, 583–596.

Speer, R.; Havasi, C.; and Lieberman, H. 2008. Analogyspace: reducing the dimensionality of common sense knowledge. In *AAAI'08: Proceedings of the 23rd national conference on Artificial intelligence*, 548–553. AAAI Press.

Suchanek, F. M.; Ifrim, G.; and Weikum, G. 2006. Combining linguistic and statistical analysis to extract relations from web documents. In *12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006)*, 712–717. ACM Press.

Suchanek, F.; Kasneci, G.; and Weikum, G. 2008. Yago: A large ontology from wikipedia and wordnet. *Elsevier Journal of Web Semantics*.

Thomas, C.; Mehra, P.; Brooks, R.; and Sheth, A. 2008. Growing fields of interest - using an expand and reduce strategy for domain model extraction. *Web Intelligence and Intelligent Agent Technology, IEEE/WIC/ACM International Conference on* 1:496–502.

Turney, P. D. 2006. Expressing implicit semantic relations without supervision. In *ACL '06: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, 313–320. Morristown, NJ, USA: Association for Computational Linguistics.

Vetere, G. 2009. From data to knowledge, the role of formal ontology. In *Proceeding of the 2009 conference on Formal Ontologies Meet Industry*, 1–9. Amsterdam, The Netherlands, The Netherlands: IOS Press.

Wang, G.; Yu, Y.; and Zhu, H. 2007. Pore: Positive-only relation extraction from wikipedia text. In *Proceedings of the 6th International Semantic Web Conference and 2nd Asian Semantic Web Conference (ISWC/ASWC2007), Busan, South Korea*, volume 4825 of *LNCS*, 575–588. Berlin, Heidelberg: Springer Verlag.