Big Data Methods for Computational Linguistics

Gerhard Weikum, Johannes Hoffart, Ndapandula Nakashole,
Marc Spaniol, Fabian Suchanek, Mohamed Amir Yosef

Max Planck Institute for Informatics
Saarbruecken, Germany
E-mail: weikum@mpi-inf.mpg.de

Abstract

Many tasks in computational linguistics traditionally rely on hand-crafted or curated resources like thesauri or word-sense-annotated corpora. The availability of big data, from the Web and other sources, has changed this situation. Harnessing these assets requires scalable methods for data and text analytics. This paper gives an overview on our recent work that utilizes big data methods for enhancing semantics-centric tasks dealing with natural language texts. We demonstrate a virtuous cycle in harvesting knowledge from large data and text collections and leveraging this knowledge in order to improve the annotation and interpretation of language in Web pages and social media. Specifically, we show how to build large dictionaries of names and paraphrases for entities and relations, and how these help to disambiguate entity mentions in texts.

1 Introduction

Methods for data analytics are relevant for all kinds of information, including text. Although we live in the era of Big Data, Linked Data, Data-Driven Science, and the Data Deluge, for humans the most informative contents on the Internet still is in *natural-language text*: books, scholarly publications, news, social media, online communities, etc. Making sense of natural language falls into the field of computation linguistics (CL). Semantics-focused tasks in CL include question answering (e.g., of the kind addressed by IBM Watson), word sense disambiguation (i.e., mapping ambiguous words such as "plant" or "star" to their meanings) co-reference resolution (e.g., mapping pronouns to a preceding noun phrase), semantic role labeling, paraphrasing and textual entailment, automatic summarization, and more [2, 21, 28, 36, 39, 40].

Data scarceness: The models and methods that computational linguistics has developed for these tasks over several decades benefit from data collections and text corpora, but usually require human-quality markup and ground-truth annotations, compiled and curated by experts (incl. quality measures such as inter-annotator agreement). CL refers to these assets as resources, as opposed to the raw sources. As human effort is the bottleneck, the available resources tend to be fairly small. A corpus with thousands of short documents (e.g., news articles), ten thousands of sentences, and millions of words is considered large. For tasks that need only unlabeled data, such as statistical machine translation, Web corpora can be harnessed and have indeed strongly influenced the state of the art. However, the semantic tasks mentioned above rely on deeper resources with

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

fine-grained labeling. Note that crowdsourcing is not a viable alternative for expert-quality annotations. Entity recognition and disambiguation, for instance, often requires thoughtfulness that goes beyond what the typical mturk worker delivers. For example, the sentence "Berlin talks to Paris about the Euro problem with Greece and Spain" should yield four entities: German government (not the city of Berlin, a case of metonymy), French government, financial crisis in Greece, financial crisis in Spain.

Time for big data: This situation has changed in the last few years. Starting in 2006, projects like DBpedia [3], freebase.com, WikiTaxonomy [42], and YAGO [47] have constructed huge knowledge bases (KBs) of entities (people, places, etc.), their semantic classes (e.g., musicians, waterfalls, love songs, etc.), and relationships between entities (e.g., worksFor, wonPrize, marriedTo, diedIn). To this end, YAGO tapped into knowledge-sharing communities like Wikipedia, and integrated the data derived there with existing CL resources like the WordNet thesaurus [12] as a semantic backbone. The resulting KBs are big-data assets that combine the rigor and quality of traditional resources with the wealth and scale of automatically harvested Web sources. This trend is still ongoing: KBs keep growing, specialized KBs are created, KBs accelerate the CL-style annotation of large corpora, and many of these resources are semantically interconnected at the entity level in the Web of Linked Data [18]. The following two examples illustrate how such big data assets can contribute to advanced tasks in computational linguistics.

Example 1: Consider a question answering scenario where a user asks: "Which composer from the eternal city wrote the score for the Ecstasy scene?", perhaps with the additional cue "..., which was later covered by a classical string player and a metal band". There is plenty of data available to retrieve the answer (in imdb.com, freely available KBs, or Web pages), but the crux is to match the natural-language input against the representations in the data sources or Web contents. The computer needs to understand ambiguous words and phrases such as "eternal city", "Ecstasy", "score", "player", and "metal band", and needs to match them against answer candidates such as "Ennio Morricone is a composer born in Rome, who became famous for the film music of several westerns. His Ecstasy of Gold was later performed also by cellist Yo-Yo Ma and by Metallica." For the matchmaking, the computer needs to solve difficult problems of *named entity disambiguation (NED)* (e.g., "eternal city" means Rome) and *word sense disambiguation (WSD)* (e.g., "cellist" matches "string player").

Example 2: Consider an entity search engine or a text analytics tool that makes recommendations for *related entities*. For example, when the user has discovered Ennio Morricone, the system may point out: "You may also be interested in Sergio Leone." A good system should augment this recommendation with an *explanation* of how Leone and Morricone are related. For example, the system could state that both are born in Rome, drawing from the facts in a KB or structured Web data. A more elaborated hint could be that Leone directed the dollar trilogy and Morricone composed the music for these movies, or more simply that both contributed to the movie "The Good, the Bad, and the Ugly". Here, both "born in" and "contributed to" are binary relations, where the former may be explicit in a KB and the latter is a priori merely a verbal phrase in a large text corpus. For being able to generate such informative explanations, the computer needs to have large dictionaries of relational paraphrases and understand how their synonyms and other lexical properties.

In this paper, we give an overview of our recent and ongoing work along these lines. We demonstrate a virtuous cycle in harvesting knowledge from large data and text collections and leveraging this knowledge in order to improve the annotation and interpretation of language in Web pages and social media. Specifically, we show how to build large dictionaries of names and paraphrases for entities and relations, and how these help to disambiguate entity mentions in texts. So we move from natural language to explicit knowledge structures, and then apply this knowledge for better machine-reading of language.

Section 2 reviews the transition from traditional CL resources to Web-scale KBs. Section 3 shows that KBs are key to building comprehensive high-quality dictionaries that connect names and phrases with entities and relationships. This in turn is a major asset for disambiguating surface names of entities in natural-language texts like news or social media, as explained in Section 4. Finally, Section 5 discusses the use case of how all this is beneficial for improving question answering over KBs and Linked Data sources.

2 Background on Knowledge Bases

The most widely used, traditional-style CL resource is the WordNet thesaurus [12]: a lexical collection of *words* and their word *senses*. Each word, such as "player", is mapped to one or more (usually more) concepts, and each concept is represented by its synonymous words that express the concept: a so-called synset accompanied by a short gloss. Two examples for synsets (with glosses in parentheses) are $\{player, participant\}$ (*a person who participates in or is skilled at some game*) and $\{musician, instrumentalist, player\}$ (*someone who plays a musical instrument as a profession*). These concepts are organized in a DAG-like hierarchy, with generalizations (hypernyms) such as $contestant$ (which, for example, has another child $athlete$) for the first sense of "player" and $performing\ artist$ for the second sense, as well as specializations (hyponyms) such as $football\ player$ or $organist$, $singer$, $soloist$, etc. WordNet contains more than 100,000 concepts and more than 200,000 word-sense pairs, all hand-crafted. It does not rigorously distinguish between classes that have entities as instances, e.g., $football\ player$, and general concepts, e.g., $harmony$, $sunset$, etc. Nevertheless, the class hierarchy of WordNet is the world's most comprehensive taxonomy of entity types.

The main deficiency of WordNet is that its classes have few instances; for example, WordNet does not know any football player or organist, and merely a few dozen singers. KB projects like YAGO closed this gap by 1) harvesting individual entities from Wikipedia and similar sources (e.g., geonames.org or musicbrainz.org), and 2) automatially mapping these entities into their proper WordNet classes. To this end, YAGO uses a noun phrase parser (a CL technique) to analyze the names of Wikipedia categories and identify their head words (e.g., "composers" in "Italian composers of film scores"), which determine candidates for superclasses of a given category. This often leaves ambiguity (e.g., "score" in the sense of grading someone's performance or in the sense of a musical composition) and also nonsensical candidates (e.g., "member" – of what?); YAGO uses simple but powerful heuristics for disambiguation. In total, this procedure yields ca. 10 million entities ontologically organized into ca. 350,000 classes: all WordNet classes plus those from Wikipedia that can be successfully mapped. Manual assessment over samples (with statistical confidence) shows that the mappings are correct in 95% of all cases. Together with relational facts derived from Wikipedia infoboxes and all provenance and context data (extraction rules or patterns, extraction source, etc.), YAGO contains nearly half a billion RDF triples. Interestingly, the entire construction of YAGO takes only a few hours on a reasonably configured server – there is no need for Map-Reduce parallelism or other kinds of scale-out techniques.

A number of projects have taken this line of automatic KB construction further in various ways:

1) increasing the number of facts about entities, by more aggressively tapping into infoboxes and other structured sources – notable examples are DBpedia [3] and freebase.com;
2) extending the KB with multilingual names of entities and classes – examples are UWN [8, 9], BabelNet [37], and WikiNet [35], and creating cross-lingual mappings between entity infoboxes [38];
3) collecting more entities from the Web, covering the long tail of out-of-Wikipedia entities – see, for example, the work on WebSets [7] and on instances-and-attributes discovery [1, 41];
4) collecting more fine-grained classes from the Web and placing them in the class taxonomy – Probase [54] and the work on doubly anchored patterns [22] are examples;
5) discovering and compiling new relations and their instances in an "Open Information Extraction" manner – from natural-language texts (e.g., TextRunner/ReVerb [4, 10]), from Web tables and lists (e.g., [25, 27, 49, 51]), by ab-initio machine learning (e.g., NELL [5]) or by crowdsourcing (e.g., ConceptNet [17, 45]).

3 From Language to Knowledge: Paraphrases at Web Scale

Names and keyphrases of entities: KBs are not an end by themselves; they allow us to connect Web contents with entities and their semantic properties, and thus enable more intelligent interpretation of contents. Therefore,

it is crucial that a KB also captures *surface names* for entities, as a counterpart to the WordNet synsets for general concepts. For example, for people, we should compile official full names (e.g., "Diana Frances Spencer") as well as short names (e.g., "Diana Spencer", "Lady Diana"), nicknames (e.g., "Lady Di"), role names (e.g., "Princess of Wales", "first wife of Prince Charles"), and other paraphrases (e.g., "queen of hearts"). For locations, organizations, and products, an even wider variety of equivalent names often exists. In addition to such names and paraphrases, it is often useful to know further *keyphrases* that are strongly connected with entities. For example, "British Royal Family" or "Paris car crash" with Lady Diana. Such keyphrases can be mined from large corpora by identifying noun phrases (via part-of-speech tagging and chunk parsing, both CL techniques) and weighing them by mutual information (MI, aka. relative entropy) with the name of the given entity [20, 48]. We have systematically compiled both names and keyphrases from Wikipedia redirects, href anchors, headings, and references, thus populating the two YAGO relations $means(name, entity)$ and $hasKeyphrase(entity, phrase)$ with more than 10 million and 80 million triples, respectively. At such input scales (the English Wikipedia contains more than 100 million href links) and output sizes, Map-Reduce techniques come in handy for gathering candidates and computing MI weights of phrases. Recently, Google has compiled a similar and even larger dictionary from their Web index, by considering href anchors pointing to Wikipedia articles [46].

Relational patterns: In high-quality KBs, the facts between entities are limited to a small set of prespecified relations. YAGO knows ca. 100 relations; DBpedia and Freebase provide a few thousand, but these are dominated by attribute-style properties with literals as arguments, e.g., hasRevenue, hasPopulation, hasGDP, hasCallingCode, etc. Interesting methods for compiling paraphrases for verb relations have been developed in [16, 23, 31, 26, 52]. Until recently, only ReVerb [10] offered a larger number of *relational patterns*. However, these are verbal phrases between noun phrases rather than properly typed relations, and consequently exhibit a large amount of noise. Examples of the resulting "factoid" triples are:

⟨ *"Carlos", "composed for", "voice and keyboards"* ⟩,
⟨ *"Maestro Morricone", "composed for", "the film"* ⟩,
⟨ *"Ennio Morricone", "was expected to win", "the Oscar"* ⟩,
⟨ *"Coulais", "has not won", "any of them"* ⟩.

Note that the first two patterns look identical but have different semantics, as the first refers to instruments as right-hand arguments and the second to movies. The other two patterns likely express the same relation (not winning an award), but this is not captured here as patterns are merely surface forms. Finally note that the arguments of the relational patterns are mere phrases, with ambiguous meanings (Carlos, the terrorist, or Juan Carlos, the Spanish king, or the composer Wendy Carlos or . . . ? Are Ennio Morricone and Maestro Morricone the same person?) and not necessarily denoting an entity at all ("any of them").

SOL patterns: This lack of semantic rigor motivated our approach, where we leverage the existing KB of canonicalized entities and their rich type system. We have compiled, by mining Web corpora, *relational paraphases* and organize them into *pattern synsets* with each synset having a *type signature*. To this end, we define a notion of syntactic-ontological-lexical patterns, *SOL patterns* for short, which are frequently occurring sequences of a) words (in lemmatized form, e.g., infinitive for verbs), b) part-of-speech tags (e.g., ADJ for adjectives or PRP$ for possessive pronouns), c) wildcards "∗" that stand for arbitrary words or word sequences, and d) semantic types as placeholders for arbitrary entities of a specific type (e.g., instruments, songs, movies, etc.). For example,

$\langle musician \rangle \ * \ compose \ for \ * \ \langle instrument \rangle$

is the SOL pattern that adds typing to the first example above, while a different SOL pattern

$\langle musician \rangle \ * \ compose \ for \ * \ \langle movie \rangle$

could be inferred from occurrences of individual musicians with individual movies and the "compose for" phrase. Further SOL patterns could be

$\langle person \rangle \ * \ honor \ by \ * \ \langle award \rangle$,
$\langle person \rangle \ * \ not \ win \ * \ \langle award \rangle$, or
$\langle person \rangle \ * \ disappointed \ * \ PRP\$ \ nomination \ * \ \langle award \rangle$,

the latter being derived from sentences such as "... was disappointed that her nomination did not result in ...". The second and the third pattern could be combined to form a *pattern synset* of equivalent patterns.

Big data methods: Our method for mining SOL patterns and organizing them into a pattern taxonomy proceeds in four steps: 1) extracting type-agnostic patterns, 2) enhancing these patterns with type signatures, 3) identifying synsets of equivalent SOL patterns, 4) inferring subsumptions among pattern synsets. For step 1 we employ techniques for *frequent sequence mining* [15, 14], after detecting entity names in sentences and mapping them to entities registered in a KB. Here we utilize the large dictionary names and their potential meanings, discussed in Section 2; the actual disambiguation method is discussed in Section 4. As we process many millions or even billions of sentences, the frequent sequence mining makes use of Map-Reduce parallelization (see [33] for details). For step 2 we replace entities by their semantic types and then compute frequent sequences with *type generalizations*. For example, the pattern

$\langle organist \rangle * composed\ for * \langle western\ movie \rangle$

may not be frequent enough, but then we lift the pattern into $\langle musician \rangle \ldots \langle movie \rangle$ or $\langle artist \rangle \ldots \langle movie \rangle$, etc. In steps 3 and 4 we consider the subsumption and equivalence of the resulting SOL patterns. To this end, we compare the *support sets* of patterns, i.e., the sets of entity pairs that co-occur with patterns. We say that a pattern p, say

$\langle singer \rangle * PRP\$\ ADJ\ voice\ in * \langle song \rangle$,

is subsumed by pattern q, e.g.,

$\langle musician \rangle * performed * \langle song \rangle$,

if the type signature of q is more general than (or the same as that of) p and the support set of p is, to a large extent, contained in the support set of q. The degree of set containment and the confidence in the pattern subsumption are quantified by statistical measures. Mutual subsumption between two patterns then yields synsets of (approximately) equivalent patterns. Finally, step 4 post-processes the output to ensure that the pattern taxonomy forms a proper DAG without cycles. Steps 3 and 4 operate on a prefix tree that encodes patterns and their support sets. All steps are parallelized using the Map-Reduce platform Hadoop.

Further details of this method are described in [33]. The resulting pattern taxonomy is called PATTY, and is available at the Web site `http://www.mpi-inf.mpg.de/yago-naga/patty/`. We currently offer PATTY collections built from the ClueWeb'09 corpus (a large crawl with 500 million English Web pages) and from the full text of the English Wikipedia (ca. 4 million articles). The former is larger; the latter is cleaner and contains more than 300,000 relational patterns with a sampling-based accuracy of 85% determined by human judges. Our Web site includes demos for several tasks [34]:

1) PATTY as a thesaurus of relational synsets including paraphrases for the canonicalized relations offered by DBpedia and YAGO,
2) schema-free search over entities and relational patterns, treating PATTY as a database of RDF triples, and
3) PATTY as a tool for explaining the relatedness of given entities (cf. [11, 29]).

4 From Knowledge to Language: Named Entity Disambiguation

NED problem: To demonstrate the enormous benefits of the compiled KB and dictionaries of names, keyphrases, and patterns, we now turn to the problem of named entity disambiguation, *NED* for short. Consider the example text "Carlos played the Moog in the Ludwig van scenes and arranged an electronic chorus for the fourth movement of the Ninth". There are standard techniques from computational linguistics for detecting spans of words that potentially denote named entities, so-called *mentions* of entities. Here, "Carlos", "Moog", "Ludwig van", and "the Ninth" are mentions that can be detected by shallow parsing and noun-phrase analysis, using conditional random fields [13]. This step also faces ambiguity, e.g., considering "Ludwig van" vs. "Ludwig van scenes" as candidate mentions, but the methodology is fairly mature. The task that NED subsequently needs to solve is to map each of the four mentions to exactly one (or, alternatively, at most one) entity registered in a KB.

The candidate space for the mapping is often huge. For example, Wikipedia and thus YAGO know four people with last name "Carlos" and more than 50 with first name "Carlos", including a king, a terrorist, many athletes, many musicians, etc. In addition, there are also cities, movies, fictional characters, and a guitar brand named "Carlos". There are more than 20 ninth symphonies, and "Ninth" has all kinds of other meanings as well (e.g., the Ninth Avenue in Manhattan).

The NED problem is tackled by combining different ingredients from the following considerations [6, 30, 24]. Our method combines all of them in a judicious manner [19].

- **Popularity of entities:** An ambiguous name is more likely to denote a prominent entity than some lesser known person or location in the long tail. For example, "Moog" is more likely to mean the synthesizer or its inventor Robert Moog, rather than the painter and poet Peter Moog or the Moog software package. The techniques for mining name-entity pairs (see Section 3) can easily collect frequency information from href anchors or could consider additional information like the lengths and link degrees of Wikipedia articles, in order to estimate the conditional probability $P[e|n]$ for entity e when observing name n. This notion of popularity is good as a *prior* for NED, but always needs to be combined with other components.
- **Similarity of contexts:** Mentions are surrounded by text, and this context can be compared to textual descriptions of candidate entities. One way of doing this is to construct bags-of-words or statistical language models (using IR techniques) over words, bigrams (word pairs), or entire phrases in the mention's proximity of the input text, on one hand, and over the same style of tokens in the Wikipedia article of an entity. Once these models are built, similarity measures like cosine (for bags-of-words vectors) or Kullback-Leibler divergence (for language models) can be used to assess how well a candidate entity e fits with a mention m, given the surrounding contexts $cxt(\ldots)$: $P[e|n, cxt(e), cxt(n)] \sim sim(cxt(e), cxt(n))$.
- **Coherence among entities:** Entities occur together in a text not at random, but only if they are semantically related to each other. For example, why would Carlos, the terrorist, be mentioned with the Moog synthesizer in a news article or blog posting? Instead, it is more likely that a musician like Wendy Carlos co-occurs with instruments like the Moog, regardless of which surface forms are used to refer to these entities. This suggests a notion of coherence for a set of entities. When entities are registered in a KB and have corresponding Wikipedia articles, one way of quantifying the coherence between two entities is based on overlap measures between the articles' incoming links [30]. For tractability, the coherence of a set E with more than two entities is factorized into pair-wise coherence: $coh(E) = P[\{e|e \in E\}] = \prod_{a,b \in E} P[a, b] \sim overlap(in(a), in(b))$.

Keyphrases for similarity: In our work on AIDA, we developed a particularly powerful kind of context-similarity model, based on the keyphrases of entities gathered by the methods of Section 3. Instead of using all words or bigrams in comparing the context of a mention and the context of an entity, we use only entity-specific keyphrases. For example, Wendy Carlos has associated keyphrases "Moog synthesizer", "Well Tempered Synthesizer", "electronic musician", etc. These are partly matched in the input text surrounding the mention "Carlos" (our example above). In [48], we developed a sophisticated model for *partial keyphrase matching* which we adopted and extended to the task of NED. This model considers the proximity of words in a partial match, uses mutual-information weights for both words and phrases, and aggregates scores over all keyphrases of a given entity. For example, when comparing Beethoven's keyphrase "Ninth Symphony's Ode to Joy" could be against the sentence "...Ode in the fourth movement of the Ninth ...", the score contribution is proportional to the total weight of the two matched words and inversely proportional to the length of the text window that contains them. This is a very powerful model, but one needs a lot of care for an efficient implementation, using big-data techniques like min-hash sketches and locality-sensitive hashing [43].

Graph algorithms for coherence: AIDA expresses the joint inference on mention-entity context similarity and entity-entity coherence as a problem of computing dense subgraphs. It builds a graph with mentions and candidate entities as nodes. A combination of keyphrase-based similarity and popularity scores is used for weights of mention-entity edges; entity-entity edges are weighed by link-overlap-based coherence. Figure 4

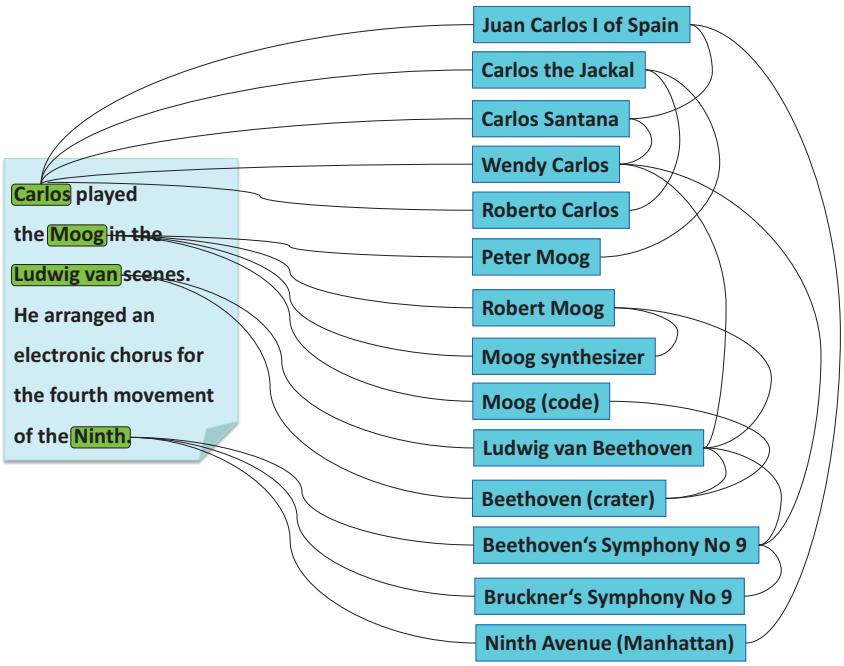

Figure 1: Example graph for named entity disambiguation

shows an excerpt for our running example. Realistic graphs can easily have many thousands of nodes, say for a news article that uses common last names or highly ambiguous phrases.

Given such a graph, the task of NED becomes a problem of computing a dense subgraph with high edge weights. Inspired by work on social network analysis [44], that is, "big data", we actually pursue the goal of finding a subgraph whose minimum weighted degree (the total weight of a node's incident edges) is as large as possible, with appropriate constraints for the NED mapping. The intuition here is that NED should be done such that the weakest link is as strong as possible. Not surprisingly, this problem is NP-hard; we devised practically viable approximation algorithms [19]. The NED quality of this method achieves top results on benchmark tasks of the CL community. AIDA is available online at `http://www.mpi-inf.mpg.de/yago-naga/aida/`.

Big data methods: In this CL-centered work, we leveraged large-scale KB's and dictionaries as key assets as well as a variety of methods typical for big data analytics: dictionaries and statistics from Web data, approximate matching, dense-subgraph computation. One of the next steps that we aim for is to apply NED to an entire corpus in a near-real-time and high-throughput manner. As a use case, consider that an analyst wants to study the media coverage of a country's politicians in one day's news (in thousands of newspapers) and social media, and also investigate the relationships with other people and organizations as expressed in this day's media. Another scenario is that a journalist wants to track a politician or celebrity over an extended time period in a large media archive, to discover patterns and trends in the entities and contexts. In both of these use cases, we need to scale up NED to process huge amounts of input documents. While this can be scaled out by partitioning the inputs, there may be better choices (e.g., in terms of memory consumption and parallel throughput) like partitioning the dictionary or dedicating compute nodes to specific subsets of target entities.

5 Use Case: Translating Questions into Queries

To illustrate how the knowledge-language interplay helps for advanced tasks, reconsider our initial question answering scenario where a user asks: "Which composer from the eternal city wrote the score for the Ecstasy

scene?" We have developed methods and a prototype system, called DEANNA [50], for automatically translating the natural language question into a structured SPARQL query that can be evaluated over subject-predicate-object (SPO) data in the RDF model. The choice of RDF and SPARQL is motivated by their schema-relaxed or schema-free nature and the fact that the Web of Linked Data provides huge amounts of informative and diverse SPO triples. The example question is translated into the query:

Select ?p Where {
?p type composer . ?p bornIn Rome . ?p created Ecstasy_of_Gold . }

where *?p* is a variable and the appearance of the same variable in different triple patterns denotes joins.

A major difficulty in this question-to-query translation lies in mapping phrases like "composer", "eternal city", or "Ecstasy" into classes and entities and phrases like "from", and "score for" into relations of the underlying KBs and other RDF sources. This resembles the NED problem discussed in Section 4, but we face a more general disambiguation problem here. A priori, a word like "score" could be either a class (e.g., soundtracks), or an entity (e.g., the movie "The Score", or a music album, or a company), or even a relation (e.g., hasSoundtrack between movies and music pieces, or scoresFor between football players and their clubs). Our DEANNA system generates all these potential meanings, using the dictionaries and pattern taxonomies discussed in Section 3. DEANNA then imposes type constraints on the feasible combinations. If we consider mapping a phrase like "score for" into the relation wroteMusic, we constrain the left and right arguments of the phrase, "Which composer" and "Ecstasy", to be of types composer and piece of music, respectively. These and other constraints are encoded into an integer linear program (ILP) for a combined selection-and-assignment problem: select appropriate phrases from the input question and assign them to classes, entities, or relations. The objective function is to maximize the coherence among the chosen candidates. The resulting ILP is not exactly small, even for short inputs, but modern solvers (e.g., `http://www.gurobi.com`) can handle this in reasonable time, usually a few seconds.

Further details on this method are given in [50]. For the example question, DEANNA generates the SPARQL query shown above, and evaluating this query over YAGO returns the correct answer: Ennio Morricone.

6 Conclusion

This paper has given an overview of our work on tackling various problems in computational linguistics, by methods that are typical for big data analytics: frequent sequence mining at Web scale, co-occurrence statistics, approximate matching, graph algorithms, scalable optimization such as ILP. Details on our work can be found in the original papers cited in each section.

We believe that these methods and the semantic assets from large knowledge bases and the Web of Linked Data are game-changers for some notoriously hard problems in computational linguistics. Text understanding in terms of entities and relationships, and question answering in natural language can greatly benefit from big data and the accompanying methodology.

References

[1] Enrique Alfonseca, Marius Pasca, Enrique Robledo-Arnuncio: Acquisition of Instance Attributes via Labeled and Related Instances. SIGIR 2010: 58-65

[2] Ion Androutsopoulos, Prodromos Malakasiotis: A Survey of Paraphrasing and Textual Entailment Methods, Journal of Artificial Intelligence Research 38: 135–187, 2010

[3] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, Zachary G. Ives: DBpedia: A Nucleus for a Web of Open Data. ISWC/ASWC 2007: 722-735

[4] Michele Banko, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, Oren Etzioni: Open Information Extraction from the Web. IJCAI 2007: 2670-2676

[5] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., Tom M. Mitchell: Toward an Architecture for Never-Ending Language Learning. AAAI 2010: 1306-1313

[6] Silviu Cucerzan: Large-Scale Named Entity Disambiguation Based on Wikipedia Data. EMNLP-CoNLL 2007: 708-716

[7] Bhavana Bharat Dalvi, William W. Cohen, Jamie Callan: WebSets: Extracting Sets of Entities from the Web using Unsupervised Information Extraction. WSDM 2012: 243-252

[8] Gerard de Melo, Gerhard Weikum: MENTA: Inducing Multilingual Taxonomies from Wikipedia, CIKM 2010: 1099-1108

[9] Gerard de Melo, Gerhard Weikum: UWN: A Large Multilingual Lexical Knowledge Base, ACL (System Demonstrations) 2012: 151-156

[10] Anthony Fader, Stephen Soderland, Oren Etzioni: Identifying Relations for Open Information Extraction. EMNLP 2011: 1535-1545

[11] Lujun Fang, Anish Das Sarma, Cong Yu, Philip Bohannon: REX: Explaining Relationships between Entity Pairs. PVLDB 5(3): 241-252, 2011

[12] Christiane Fellbaum, George Miller (Editors): WordNet: An Electronic Lexical Database, MIT Press, 1998

[13] Jenny Rose Finkel, Trond Grenager, Christopher Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. ACL 2005: 363-370

[14] Gösta Grahne, Jianfei Zhu: Fast Algorithms for Frequent Itemset Mining Using FP-Trees. IEEE Transactions on Knowledge and Data Engineering 17(10): 1347-1362, 2005

[15] Jiawei Han, Jian Pei, Yiwen Yin, Runying Mao: Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. Data Mining and Knowledge Discovery 8(1): 53-87, 2004

[16] Chikara Hashimoto, Kentaro Torisawa, Kow Kuroda, Stijn De Saeger, Masaki Murata, Jun'ichi Kazama: Large-Scale Verb Entailment Acquisition from the Web. EMNLP 2009: 1172-1181

[17] Catherine Havasi, Robert Speer, and Jason Alonso. ConceptNet: A Lexical Resource for Common Sense Knowledge. In: Recent Advances in Natural Language Processing, Volume 5. John Benjamins Publ., 2009

[18] Tom Heath, Christian Bizer: Linked Data: Evolving the Web into a Global Data Space Morgan & Claypool Publishers 2011

[19] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, Gerhard Weikum: Robust Disambiguation of Named Entities in Text, EMNLP 2011: 782-792

[20] Johannes Hoffart, Fabian M. Suchanek, Klaus Berberich, Gerhard Weikum: YAGO2: A Spatially and Temporally Enhanced Knowledge Base from Wikipedia, Artificial Intelligence 2012

[21] IBM Journal of Research and Development 56(3-4), Special Issue on "This is Watson", 2012

[22] Zornitsa Kozareva, Ellen Riloff, Eduard H. Hovy: Semantic Class Learning from the Web with Hyponym Pattern Linkage Graphs. ACL 2008: 1048-1056

[23] Zornitsa Kozareva, Eduard H. Hovy: Learning Arguments and Supertypes of Semantic Relations Using Recursive Patterns. ACL 2010: 1482-1491

[24] Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, Soumen Chakrabarti: Collective Annotation of Wikipedia Entities in Web Text. KDD 2009: 457-466

[25] Girija Limaye, Sunita Sarawagi, Soumen Chakrabarti: Annotating and Searching Web Tables Using Entities, Types and Relationships. PVLDB 3(1): 1338-1347, 2010

[26] Dekang Lin, Patrick Pantel: DIRT – Discovery of Inference Rules from Text. KDD 2001: 323-328

[27] Ashwin Machanavajjhala, Arun Shankar Iyer, Philip Bohannon, Srujana Merugu: Collective Extraction from Heterogeneous Web Lists. WSDM 2011: 445-454

[28] Inderjeet Mani: Automatic Summarization, John Benjamin Publ., 2001

[29] Steffen Metzger, Shady Elbassuoni, Katja Hose, Ralf Schenkel: S3K: Seeking Statement-Supporting Top-K Witnesses. CIKM 2011: 37-46

[30] David N. Milne, Ian H. Witten: Learning to Link with Wikipedia. CIKM 2008: 509-518

[31] Thahir Mohamed, Estevam R. Hruschka Jr., Tom M. Mitchell: Discovering Relations between Noun Categories. EMNLP 2011: 1447-1455

[32] Ndapandula Nakashole, Martin Theobald, Gerhard Weikum: Scalable Knowledge Harvesting with High Precision and High Recall, WSDM 2011: 227-236

[33] Ndapandula Nakashole, Gerhard Weikum, Fabian Suchanek: PATTY: A Taxonomy of Relational Patterns with Semantic Types, EMNLP-CoNLL 2012: 1135-1145

[34] Ndapandula Nakashole, Gerhard Weikum, Fabian Suchanek: Discovering and Exploring Relations on the Web, PVLDB 5(12): 1982-1985, 2012

[35] Vivi Nastase, Michael Strube, Benjamin Boerschinger, Ccilia Zirn, Anas Elghafari: WikiNet: A Very Large Scale Multi-Lingual Concept Network. LREC 2010: 1015-1022

[36] Roberto Navigli: Word Sense Disambiguation: A survey. ACM Computing Surveys 41(2): 10:1-10:69, 2009

[37] Roberto Navigli, Simone Paolo Ponzetto: BabelNet: Building a Very Large Multilingual Semantic Network. ACL 2010: 216-225

[38] Thanh Hoang Nguyen, Viviane Moreira, Huong Nguyen, Hoa Nguyen, Juliana Freire: Multilingual Schema Matching for Wikipedia Infoboxes. PVLDB 5(2): 133-144, 2011

[39] Martha Palmer, Daniel Gildea, Nianwen Xue: Semantic Role Labeling, Morgan & Claypool Publishers, 2010

[40] Bo Pang, Lillian Lee: Opinion Mining and Sentiment Analysis, Foundations and Trends in Information Retrieval 2(1-2), Now Publishers, 2008

[41] Joseph Reisinger, Marius Pasca: Fine-Grained Class Label Markup of Search Queries. ACL 2011: 1200-1209

[42] Simone Paolo Ponzetto, Michael Strube: Deriving a Large-Scale Taxonomy from Wikipedia. AAAI 2007: 1440-1445

[43] Anand Rajaraman, Jeffrey David Ullman: Mining of Massive Datasets, Cambridge University Press, 2011

[44] Mauro Sozio, Aristides Gionis: The Community-Search Problem and How to Plan a Successful Cocktail Party. KDD 2010: 939-948

[45] Robert Speer, Catherine Havasi, and Harshit Surana Using Verbosity: Common Sense Data from Games with a Purpose. FLAIRS 2010: 104-109

[46] Valentin I. Spitkovsky, Angel X. Chang: A Cross-Lingual Dictionary for English Wikipedia Concepts, LREC 2012: 3168-3175

[47] Fabian M. Suchanek, Gjergji Kasneci, Gerhard Weikum: Yago: a Core of Semantic Knowledge. WWW 2007: 697-706

[48] Bilyana Taneva, Mouna Kacimi, Gerhard Weikum: Finding Images of Difficult Entities in the Long Tail. CIKM 2011: 189-194

[49] Petros Venetis, Alon Y. Halevy, Jayant Madhavan, Marius Pasca, Warren Shen, Fei Wu, Gengxin Miao, Chung Wu: Recovering Semantics of Tables on the Web. PVLDB 4(9): 528-538, 2011

[50] Mohamed Yahya, Klaus Berberich, Shady Elbassuoni, Maya Ramanath, Volker Tresp, Gerhard Weikum: Natural Language Questions for the Web of Data, EMNLP-CoNLL 2012: 379-390

[51] Mohamed Yakout, Kris Ganjam, Kaushik Chakrabarti, Surajit Chaudhuri: InfoGather: Entity Augmentation and Attribute Discovery by Holistic Matching with Web Tables. SIGMOD Conference 2012: 97-108

[52] Limin Yao, Aria Haghighi, Sebastian Riedel, Andrew McCallum: Structured Relation Discovery using Generative Models. EMNLP 2011: 1456-1466

[53] Mohamed Amir Yosef, Johannes Hoffart, Ilaria Bordino, Marc Spaniol, Gerhard Weikum: AIDA: An Online Tool for Accurate Disambiguation of Named Entities in Text and Tables. PVLDB 4(12): 1450-1453, 2011

[54] Wentao Wu, Hongsong Li, Haixun Wang, Kenny Qili Zhu: Probase: a Probabilistic Taxonomy for Text Understanding. SIGMOD Conference 2012: 481-492