SCIENTIFIC PUBLICATIONS OF THE STATE UNIVERSITY OF NOVI PAZAR
SER. A: APPL. MATH. INFORM. AND MECH. vol. 7, 1 (2015) 7-18

Towards an open software architecture for interleaved knowledge and natural language processing

Marcus Spies

Abstract: Many existing integrations of natural language processing (NLP) and knowledge-based systems build on "shallow" NLP. Prominent examples are term-frequency based document retrieval or document topic extraction systems. Recent progress in NLP, however, has brought "deep" processing features like sentential parsing and semantic dependency analysis to a highly mature level for a substantial set of common spoken languages. Specifically, "deep" NLP outputs beyond conventional syntax trees allow for better interoperability with other information or knowledge management (IM/KM) components, e.g. those using semantic technologies or statistical learning approaches. In this paper, the emerging importance of interleaved knowledge and language processing in the context of cognitive computing is shown. Building blocks for an open architecture for deep NLP applications are introduced and discussed.

Keywords: cognitive computing, knowledge processing, semantic technologies, natural language processing

1 Introduction

Big data infrastructures [42] and big data computing [1] are key challenges for computer science in our decade. In particular, the absolute and relative growth of unstructured data poses massive problems related to content detection and discovery. A subset of unstructured data of particular importance to human society as well as information and knowledge management (IM/KM) is textual data. Textual data is present in several forms, ranging from digital libraries over web page and digital journals content, texts in electronic mail and messaging systems, to text recovered from human speech.

In order to process textual data appropriately in a broad variety of analytic applications, natural language processing needs to be interleaved with computerized knowledge representation and reasoning / inferencing formalisms. One key application of interleaved NLP and knowledge processing is content detection and classification from large textual data sets or streams. Building on early approaches in machine reading [33], recent research has contributed highly performant platforms for this textual analytics task. The IBM Watson[1]

Manuscript received April 28, 2014. ; accepted September 15, 2014.

M. Spies is Professor of Knowledge Management at LMU University of Munich, mspies@lrz.uni-muenchen.de

system has been used to demonstrate that the integration of "deep NLP" with IM/KM will lead to extremely powerful cognitive computing systems emulating human cognition and insight based on natural language to an impressive degree. In a similar vein, an open-source question answering framework has been developed in [6], and a generic semantics based framework for grammar learning and inferencing from text is available from [3].

We briefly mention a second application area related the European Horizon 2020 research framework program. One key objective here is “cracking the multilanguage barrier”. Obviously, having a common logical representation of meaning that can be used to read and represent but also to generate utterances in multiple natural languages is going to be an important contribution in this intended direction.

2 Deep Natural Language Processing

Natural language processing (NLP) has gone through a major paradigm shift in recent years.

Many practical applications of NLP in the information sciences (eg, for digital libraries) and web research have been using a simplified quantitative model of texts based on the *bag-of-words assumption*. By this assumption, any text can be represented as a point in a high-dimensional vector space; a corpus of texts can be summarized in a document-term matrix. The entries in such matrices can be simple term frequencies (*tf*) or combinations of normalized term frequencies with estimates of document-specific occurrence of a term (inverse document frequency, *tf-idf*). In a straightforward representation, the document vector space for a corpus of texts is spanned by the terms in a given vocabulary. More sophisticated representations are obtained from computing conceptual spaces from linear combinations of term dimensions. A prominent approach of this kind is the *latent semantic analysis* which uses a singular-value-decomposition of document-term matrices into a set of conceptual components highlighting covariant terms and a set of concordance components highlighting similar documents [11]. In recent years, the bag-of-words approach has continued to be extremely useful in topic analysis. Topic modelling aims at extracting latent thematic structures from large text corpora using a broad variety of (usually unsupervised) machine learning methods [8, 37].

As this entire tradition of research basically neglects the grammatical and logical structure of texts, it is also often referred as “shallow NLP”, in contrast to “deep NLP”. Regarding deep NLP, a long tradition of computational linguistic research was addressing syntax and semantics formalisms in separate lines of research. While syntactic structures were analyzed by increasingly sophisticated parsers using phrase-structure grammars building on context-free languages, semantic structures were investigated in mostly syntax agnostic lexicon components and research on conceptual structures. Both lines of research had to rely on substantial construction and annotation work by linguistic experts in order to build the resoures needed for software components in specific languages or for specific domains.

A key innovation paving the way to an increased adoption of deep NLP is *information extraction* (IE). IE was introduced as a technique for automatic recognition of named entities (NER, like recognizing actors, politicians, institutions, etc) from text using supervised

machine learning mechanisms. The task is performed by training classifiers with more detailed and sequential data than available in a bag-of-words approach, notably part-of-speech (POS) tags, semantic properties of nearby but also distant words etc. As training data is substantially incomplete in most cases of practical interest (see [39]), an innovative statistical relational learning method is being used (conditional random fields, see 3) in several tools. A key advantage of IE is that texts are now analyzed on a level that comes much closer to information and knowledge engineering than bag-of-word representations. Specifically, entities recognized can suitably be collected in the resource-description framework (RDF) format and organized by domains in RDF-graphs [20]. This opens the door towards reasoning on entities and combining entities by properties in the web ontology language OWL [31] (or a logical programming language like Prolog [12]). For an impressive tool using entity and relationship recognition on a large scale, see [32]. Finally, entities and relationships can be mined with unsupervised learning approaches from large textual datasets or data streams rather than recognized using a preconfigured classifier. This has been done, e.g., in [27].

2.1 Semantic Parsing

While IE is a huge step ahead towards deep NLP, it still fails to recover the full logical structure of meaning in text. Assuming it were equipped with the best IE components, a question-answering system would still be capable only of finding relevant texts relating to a question, but it could not infer or deduce an appropriate *answer*. Therefore, the decisive development enabling a "cognitive" text understanding system had to come from another line of research. More specifically, *semantic parsing* formalisms have been investigated in which syntax analysis and semantic structures are combined in novel ways. Semantic parsing yields a syntactic analysis in combination with logical forms representing predicate-argument structures in a given text. A useful approach to syntax analysis amenable to be combined with logical forms is *dependency parsing* (which exists in several variants, we describe the Stanford parser methodology, see [40], in this paragraph). Dependency parsing reconstructs the syntactical structure of a sentence using a set of grammatical categorical relationships, see[13] for the meta-model containing the dependency types, [14] for its justification and some comparisons to related approaches.

For an example see Figure 1. The input sentence is represented by a graph that allows terms to participate in several dependency relationships. These relationships can be used to infer semantic relationships and even predicate argument structures as will be explained below.

In the IBM Watson system, semantic parsing is based on the English slot grammar formalism, see [30], which produces dependency trees in a similar way to [14]. However, ESG uses a particular dependency taxonomy augmented by external type information (in [30], WordNet is used). The output of the dependency analysis is then further processed to yield candidate relation predicate argument structures (PAS) covering also long-distance textual relationships. [41] uses a hierarchical relation topic model on top of these predicate argument structures to detect relation instances in text.

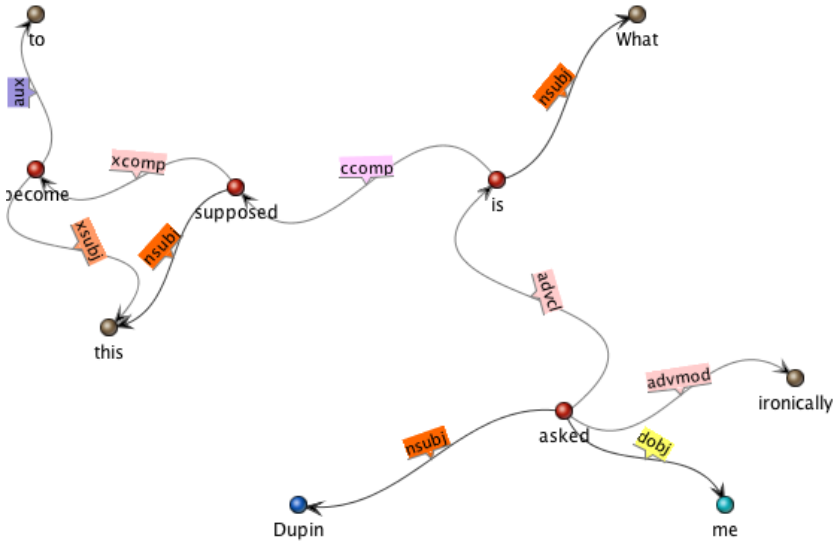

Fig. 1. Dependency parse result for the sentence `What is this supposed to become, Dupin asked me ironically.` using [40] version 3.3.1 and [9]. Dependencies are as defined in [13].

Semantic parsing is also one of the key characteristics of Combinatory Categorial Grammar (CCG, [38, 2]). CCG defines contextual grammatical categories for lexical items, contextual meaning here that these categories include descriptors for slot-filling further item categories needed to achieve a meaningful linguistic context. As a trivial example, an adjective would not only be categorized by its POS tag but also as an entity that can be combined with a noun to build a simple noun phrase. Furthermore, CCG also defines semantics as logical forms for lexical items (usually again with λ-calculus, but other related formalisms are in use, see [10]). A lexicon entry for a term or a phrase in CCG therefore provides a grammatical category which includes syntactic and semantic composition features and a logical form. In constructing a semantic parse for a sequence of phrases both the grammatical and the logical components of the corresponding lexical items are composed using some general and additional optional grammar-specific composition rules.

The key distinguishing feature compared to other semantic parsing formalisms is that composition rules on grammatical categories can be mapped to composition rules on the logical forms. This leads to integrated derivations of sentence structure and meaning. As a consequence, CCG uses (or builds during a learning process) rich lexica with many distinctive features for the category and logical representation aspects. A useful resource for working with CCG parses is [4]. An example parsing derivation with CCG is given in fig. 2.

2.2 Deriving Logical Forms – Compositional Semantics

Logical forms for natural language expressions are conveniently expressed using the lambda-calculus [25, 35]. The need to have a more powerful formalism than first-order

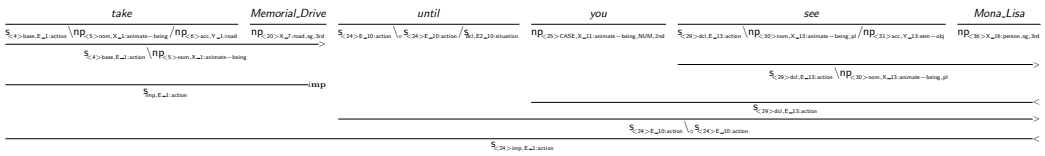

Fig. 2. Semantic parse derivation of an imperative sentence with combinatory categorial grammar (CCG). Generated with [4]. For the precise meaning of the CCG notation, see[10].

logic comes from the basic natural language property referred to as semantic *compositionality*. Compositionality implies that the meaning of a (single or multiple) word expression can usually be modified by pre- or suffixing the expression with other words. In theories of concepts [28], it has been shown that compositionality is usually different from a conjunction of concepts or of properties. On the other hand, compositionality is a built-in capability of the lambda-calculus. Eq. 1 gives an illustrative lexical item with lambda-form example for the interpretation of *take* in fig. 2. It shows the grammatical category composition and the λ-form with the second-order predicate action and its associated subject/object semantic types. Note that the generation of a query against a suitable RDF-graph from this kind of representation is straightforward (assuming the higher-order predicate being represented as usual `rdf : type` triple associated with the next-lower order predicate).

$$take \vdash (S\backslash NP)|NP : \lambda x.\lambda y.\texttt{action}(take) \wedge \texttt{animateBeing}(x) \wedge \texttt{road}(y) \wedge take(x,y) \quad (1)$$

With the appropriate relationship instances for a sentence, logical forms representing the meaning of words or word spans in a lexicon can be combined to yield a logical form representing the meaning of the entire sentence.

In the IBM Watson[1] system, see [16], dependency trees resulting from semantic parsing are further decomposed into frames isolating branches of maximal depth 2. These frames correspond to candidate parts of axioms or candidate axioms in a logical form representation of the textual input. [16] uses a technique of frame projection to a few statement schemes (like Subject-Verb-Object or Verb-Object type) together with an adaptable type system to extract candidate ground axioms and terminological (non-ground) axioms from large document corpora. A key technique for combining simple frames into more complex meaningful ones is frame projection, which results in unifying, e.g., the object of a verb in one frame subset with the object type of the same verb in other frames and thus allows to build more complex logical forms from simpler ones.

[5] uses λ-dependency based compositional semantics ([26, 25]) for generating candidate logical forms representing the meaning of sentences in text. This approach applies some generic fundamental compositional rules on entities and relationships discovered from corpus alignment of training texts with [18] to construct candidate existential restrictions, concept intersections and basic aggregation operators. For a more complex case of compositionality, the authors of [5] use a particular bridging operation which is similar to frame projection in [16], however, increases the scope of possible unifications by generating additional candidate existence restrictions.

These constructs are used to build more elaborate semantic representations into a lexicon to be used in building derivation trees on test phrases. As a consequence, lexica in a semantic parsing NLP approach contain mappings from natural language phrases to logical forms (most often expressed using typed λ-calculus). More formally, a lexeme $\mathbf{l}$ can be modelled as a binary relation associating a phrase $\overrightarrow{w}$ with a sequence of logical constants. Using list notation like in Prolog and ordinary vector notation $(\cdot,\cdot)$, we have $\mathbf{l} = (\overrightarrow{w_\mathbf{l}}, [c_1, \ldots, c_{n(\mathbf{l})}])$ assuming $n(\mathbf{l})$ logical constants being associated with $\overrightarrow{w_\mathbf{l}}$. Note that one phrase can be usually represented by several lexemes.

Recent implementations of CCG use *factored lexica* (see [22, 2]). In a factored lexicon, a candidate semantic representation for $\overrightarrow{w_\mathbf{l}}$ is constructed as composition of a lexeme relation with a lexical template. A lexical template maps a lexeme to a full lexicon entry or lexical item as defined in section 2.1. Again, for one lexeme there are usually several lexicon entries. Conversely, one lexical item can be generated from multiple combinations of lexemes and lexical templates.

Representing these ambiguities as template matching and composition allows to capture patterns of lexical template structures for phrase contexts or for lexeme contexts. These patterns are then used as features in the discriminative statistical models built with deep learning algorithms as outlined in 3.

In a DCS-based approach, the lexicon maps a phrase $\overrightarrow{w}$ to a predicate based on an alignment procedure with a training corpus containing type information. In the case of $\overrightarrow{w}$ representing a named entity the predicate is a nominal, i.e. maps to exactly one individual in knowledge base $\mathscr{K}$.

2.3 Interleaving Semantic Parsing and Knowledge Processing

The need to interleave natural language and knowledge processing is particularly evident in innovative domains like question answering. The key task of question analysis in IBM Watson[1] is to derive the lexical answer type for the reply. The approach to this reported in [23] uses the standard unstructured information management architecture (UIMA), which defines common annotation structures (CAS) to be produced and consumed along text processing pipelines. The benefit of this architecture is that different knowledge processing components (like evidential processing of candidate answer-relevant texts) can be flexibly integrated. In the case of the question answering approach in [23], a logic programming system (Prolog engine) is used to infer lexical answer types from question and question domain annotations.

In a related sense, the approach in [6] is based on question-answer training data, which in turn uses [18] as knowledge base $\mathscr{K}$. As a result, semantic parses of new sentences can establish new *factual statements* to be inserted into $\mathscr{K}$ or new *terminological statements* extending the schema of $\mathscr{K}$. In semantic web terms, these two options are called ontology population vs. ontology learning. The general consistency condition on logical forms in semantic parsing is that the denotation $[\![\cdot]\!]$ of the logical form z derived from a question given in utterance x must equal the correct answer(s) t over knowledge base $\mathscr{K}$, $[\![z]\!] = \{t | t \in \mathscr{K}, (t,x) \in T\}$, where T is a set of training data with paired question-answer data.

It should be noted that there are further uses of interleaved NLP and knowledge processing which address knowledge in agent systems. Here, imperative sentences representing instructions (like in our example in fig. 2) are an important class of statements. Processing them can open new pathways for speech-enabled applications in everyday scenarios like driver assistance systems (see examples in [2]).

We now turn from discussing semantic parsing approaches to effective mechanisms for constructing the artifacts used in these approaches from training data.

3 Deep Learning

Deep learning refers to several statistical inference methods using latent features or variables. Regarding NLP, the need for latent features becomes obvious already in NER, as training data are systematically incomplete (many entities do not appear at all, many feature combinations are too rare to be observed even in large training corpora).

One branch of deep learning contains novel approaches to training multi-layer neural networks improving on the established backpropagation method by a blend of unsupervised and supervised learning phases [24]. Another branch of deep learning contains new methods for training Boltzmann machines that improve on the earlier EM-algorithm based training [19]. In addition, deep Boltzmann machines also feature a constrained layered connectivity. Specifically, in restricted Boltzmann machines (RBM), connectivities can be limited to cross-layer connections [34].

Conditional random fields (CRF, [39]) are the key machine learning framework for discriminative graphical statistical models combining the functionality of single or multi-label classification with the power of graphical models like in Bayesian networks using directed graphs or (for the undirected case) in conventional Boltzmann machines. As an example, RBM disallow connections involving network elements representing input variables. This restriction actually makes the inferred RBM model a discriminative model as opposed to the generative model computed in earlier Boltzmann machine approaches. More generally, discriminative models parameterize conditional distributions of latent and output variables given input variables, while generative models construct all full parameter set for the entire multivariate distribution of input, hidden and output variables.

Deep learning is essential for setting up machine learning approaches to semantic parsing. All approaches discussed in section 2 use deep learning in several phases of system setup.

The basic formal setup for learning semantic parsing structures given training data is this– we want to evaluate the posterior probability of a logical form z (like the one given in eq. 1) associated with a candidate parse y (like the one given in fig. 1) of a training sentence or utterance x. Structurally, y and z are represented each as a set of nodes describing the formal components (e.g. syntactic categories, semantic types, etc) in a graph, where edges represent deterministic associations (e.g. for a grammatical composition of syntactic catetories) for each fixed y parse or fixed logical form z. The specific need for deep learning in this problem setting lies in the fact that the lexical items to be learned are not explicitly

labeled in the training data, see [22], p 1513.

Now, given training data these structures become *instantiated* with specific vocabulary items, lexical types involved in a parse, components of a λ-expressions etc. for each training sentence x. As a result, the state of the deterministic associations will vary depending on the input sentence and the candidate parse. So, for a set of parses / logical forms the structure inferred from a training sentence will contain the empirical or observed activation frequencies of the nodes and edges involved in the structure representation based on the training data. Furthermore we assume these frequencies to be sufficient statistics for the parameters of an exponential-family probability distributions. This leads to a representation of each parse or logical form structure element by a feature function. Additional features may be defined using quantitative or qualitative evaluations of the parse in question (this is used in both recent papers [6] and [22]). The overall type of statistical model applicable is then either a Markov random field or a conditional random field [39]. As we are interested in a conditional model of parses and logical forms based on training sentences, the model type of choice is a CRF. An instantiated structure graph based on a training set of data x is referred to a *ground graph* in Markov logic networks [15] (MLN) and other relational probabilistic model approaches [36]. In fact, MLNs based on evidence data (x in our application) can be viewed as CRF with declaratively defined feature structures. This makes MLN tools like [21] applicable to the inference problems discussed here.

More formally, the CRF expressing the posterior conditional probability of a particular candidate parse y and logical form z can be stated as follows. We use feature functions $f_y \in \mathscr{F}_y$ for y and $f_z \in \mathscr{F}_z$ for z whose values depend on the currently processed input sentence x. Each candidate combination of a parse y and a logical form z corresponds to a state of the CRF. The conditional probability model for this state involves parameters from a set Θ for each feature activated in the state. Denoting these parameters by θ_{f_y} and θ_{f_z}, we obtain a generic CRF model as given in eq. 2.

$$\Pr(y,z|x;\Theta) = \frac{1}{Z(x)} \exp\left(\sum_{f_y \in \mathscr{F}_y, f_z \in \mathscr{F}_z} \theta_{f_y} f_y(x) + \theta_{f_z} f_z(x) \right) \tag{2}$$

Please note that $Z(x)$ in eq. 2 is a normalization constant given a fixed input item x (unlike the overall constant Z used in the unconditional or generative Markov random field approach). Of course this model, once trained is viable also for the inference task which requires finding most likely parses and logical forms for test data.

Details of the learning procedure and results for the λ-DCS approach are given in [26], for probabilistic CCG in [22]. A challenging task for further research is the automatic induction of more compositional logical predicates [6].

4 Conclusion–Components of an integrated architecture

"Shallow NLP" architectures as used in text mining are often represented as processing pipelines that comprise at least these steps for each document in a corpus or a text stream –

- text extraction (e.g. by searching an XML document tree for news texts typically available in XML)
- tokenization
- stop-word elimination and other cleaning procedures
- stemming or reduction to vocabulary terms (in some cases, improved by lemmatization or POS tagging)
- computation of document term raw frequency matrices
- *tf-idf* correction (see section 2) of raw frequencies

For an implementation in the R statistical software package *tm*, see [17]. After these steps are completed, the vector space model for the texts is set up and further analyses can be performed. A processing pipeline of this kind can readily be reconfigured to serve as *data preprocessor package* also for deep NLP.

Setting up an architecture for deep NLP will require a *semantic parser*. This parser is a package in usual UML architecture terms. The parser package will enclose several components that need to use existing artifacts like dependency taxonomies and specific lexica (see section 2.2) for some usual words or phrases.

The artifacts accessed and created by parser components are lexica, more specifically, the grammar, logical form expressions, entity and relation type systems etc making up a semantic parser lexicon. The format of the lexica will differ according to the overall semantic parsing approach (in the present paper, we discussed CCG, ESG, and λ-DCS). Technically, CCG comes with appropriate XML schemata for all parsing components that would allow a message-based interaction with CCG components in a service-oriented architecture.

In addition, a *machine learning package* will be needed to improve and augment the imported lexica based on training data. This applies to any domain specific application, and even more so to an open-domain approach like in IBM Watson[1]. In an open architecture, this learning module should not be coupled to the specific approach to semantic parsing adopted in the parser package and its components. For instance [29] is a rather encompassing open source inference and machine learning module useful for CRFs (and extended for some out-of-the-box NLP tasks, as well). Markov logic networks (MLN) engines can also be used.

Interfacing a machine learning engine with semantic parser artifacts is an important step towards an open deep NLP architecture. This requires a declarative language that is suited to stating a probabilistic model derived from eq.2 for semantic parsing learning. MLN engines usually come with a simplified first-order logic dialect input language which can readily be used by non-programmers. A domain specific modelling language covering a wide variety of statistical inference problems is [7]. Other machine learning engines supporting CRFs mostly require users to code their model directly in programming code (e.g., a Java class). However, stating a complex CRF model in a programming language has some drawbacks once the number of the model elements and parameters becomes very

high. Consistency checks would need an additional module. Therefore a next highly useful step towards an open architecture for deep NLP is a *domain specific modelling language* for semantic parsing learning models together with a generator of platform specific code for some given machine learning engine. It should be noted that a MLN engine, by the built-in capability to generate the ground graph as discussed in section 3, will make the model formulation task easier from the very beginning.

To complete the architecture, a knowledge base is an additional required package holding denotations of lexical forms in a real use case. Recent research uses logical programming and RDF-based graph databases.

To sum up, an open architecture for semantics-enabled applications with deep NLP is a realistic perspective for upcoming research.

Notes

[1]IBM Watson is a trademark, service mark or registered trade mark of International Business Machines Corporation in the United States, other countries, or both.

References

[1] Rajendra Akerkar, ed. *Big Data Computing*. Chapman and Hall/CRC, 2013. ISBN: 978-1-4665-7837. DOI: doi:10.1201/b16014. URL: http://dx.doi.org/10.1201/b16014-1.

[2] Yoav Artzi, Nicholas FitzGerald, and Luke Zettlemoyer. *Semantic Parsing with Combinatory Categorial Grammars*. Tutorial. University of Washington, 2013. URL: http://yoavartzi.com/pub/afz-tutorial.acl.2013.pdf.

[3] Yoav Artzi and Luke Zettlemoyer. *UW SPF: The University of Washington Semantic Parsing Framework*. 2013. eprint: arXiv:1311.3011.

[4] Jason Baldridge and Michael White. *OpenCCG*. Tech. rep. University of Edinburgh, 2006. URL: http://sourceforge.net/projects/openccg.

[5] Jonathan Berant et al. "Semantic Parsing on Freebase from Question-Answer Pairs". In: *Proceedings of EMNLP*. 2013. URL: http://cs.stanford.edu/~pliang/papers/freebase-emnlp2013.pdf.

[6] J. Berant et al. "Semantic Parsing on Freebase from Question-Answer Pairs". In: *Empirical Methods in Natural Language Processing (EMNLP)*. 2013.

[7] Stephen Bishop and et al. *infer.NET User Guide*. Tech. rep. Microsoft Research, 2013. URL: http://research.microsoft.com/en-us/um/cambridge/projects/infernet/.

[8] David M. Blei. "Probabilistic Topic Models". In: *Communications of the ACM* 55.4 (2012). A high-level overview of probabilistic topic models., pp. 77–84.

[9] Bernard Bou. *GrammarScope*. 2014. URL: http://grammarscope.sourceforge.net/.

[10] Cem Bozşahin, Geert-Jan M. Kruijff, and Michael White. *Specifying Grammars for OpenCCG: A Rough Guide*. Tech. rep. School of Informatics; University of Edinburgh, 2013.

[11] Roger B. Bradford. *Implementation techniques for large-scale latent semantic indexing applications*. 2011.

[12] Mats Carlsson. *SICStus Prolog Users Manual*. Tech. rep. Swedish Institute of Computer Science, 2011.

[13] Marie-Catherine de Marneffe and Christopher D. Manning. *Stanford typed dependencies manual*. 2008.

[14] Marie-Catherine de Marneffe, Bill McCartney, and Christopher D. Manning. "Generating Typed Dependency Parses from Phrase Structure Parses". In: *Proceedings 5th Int. Conf. on Language Resources and Evaluation (LREC)*. European Language Resources Association. 2006. URL: http://www.lrec-conf.org/proceedings/lrec2006/.

[15] Pedro Domingos and Matthew Richardson. "Markov Logic: A Unifying Framework for Statistical Relational Learning". In: *Introduction to Statistical Relational Learning*. Ed. by Lise Getoor and Ben Taskar. Cambridge, MA: MIT Press, 2007. Chap. 12, pp. 339–372.

[16] J. Fan et al. "Automatic knowledge extraction from documents". In: *IBM Journal of Research and Development* 56.3.4 (May 2012), 5:1–5:10. ISSN: 0018-8646. DOI: 10.1147/JRD.2012.2186519.

[17] Ingo Feinerer. *An introduction to the tm package – Text mining in R*. 2012.

[18] *Freebase*. 2012. URL: https://www.freebase.com/.

[19] Geoffrey Hinton. *A Practical Guide to Training Restricted Boltzmann Machines*. URL: http://www.cs.toronto.edu/~hinton/absps/guideTR.pdf.

[20] G. Klyne and J.J. Caroll. *Resource Description Framework (RDF): Concepts and Abstract Syntax*. 2009.

[21] Stanley Kok et al. *The Alchemy System for Statistical Relational AI*. 2005. URL: http://www.cs.washington.edu/ai/alchemy.

[22] Tom Kwiatkowski et al. "Lexical Generalization in CCG Grammar Induction for Semantic Parsing". In: *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Ed. by Association for Computational Linguistics. Association for Computational Linguistics. Edinburgh, Scotland, UK, July 2011, pp. 1512–1523. URL: http://aclweb.org/anthology//D/D11/D11-1140.pdf.

[23] A. Lally et al. "Question analysis: How Watson reads a clue". In: *IBM Journal of Research and Development* 56.3.4 (May 2012), 2:1–2:14. ISSN: 0018-8646. DOI: 10.1147/JRD.2012.2184637.

[24] Honglak Lee. *Tutorial on Deep Learning and Applications*. 2010. URL: http://deeplearningworkshopnips2010.files.wordpress.com/2010/09/nips10-workshop-tutorial-final.pdf.

[25] Percy Liang. "Lambda Dependency-Based Compositional Semantics". In: *Computing Research Repository (CoRR)* abs/1309.4408v2 (2013).

[26] Percy Liang, Michael I. Jordan, and Dan Klein. "Learning Dependency-Based Compositional Semantics". In: *arxiv.org/CoRR* abs/1109.6841 (2011).

[27] Thomas Lin, Mausam Etzioni, and Oren Etzioni. "Entity Linking at Web Scale". In: *AKBC-WEKEX 2012 | The Knowledge Extraction Workshop at NAACL-HLT 2012*. Ed. by Turing Center KnowItAll Project. Montréal, Canada, June 2012. URL: https://akbcwekex2012.files.wordpress.com/2012/05/25%5C_paper.pdf.

[28] Eric Margolis. *Concepts: Core Readings*. Cambridge: Cambridge University Press, 1999.

[29] Andrew McCallum, Karl Schultz, and Sameer Singh. "FACTORIE: Probabilistic Programming via Imperatively Defined Factor Graphs". In: *Neural Information Processing Systems (NIPS)*. 2009.

[30] M. C. McCord, J. W. Murdock, and B.K. Boguraev. "Deep parsing in Watson". In: *IBM Journal of Research and Development* 56.3.4 (May 2012), 3:1–3:15. ISSN: 0018-8646. DOI: 10.1147/JRD.2012.2185409.

[31] B. Motik, P. Patel-Schneider, and B. Parsia. *OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax*. 2009.

[32] *Opencalais*. 2012. URL: http://viewer.opencalais.com/.

[33] Hoifung Poon and Pedro Domingos. "Machine Reading: A "Killer App" for Statistical Relational AI". In: *Proc. AAAI Conference*. 2010.

[34] Ruslan Salakhutdinov and Geoffrey Hinton. "Deep Boltzmann Machines". In: *Proceedings of the 12 th International Conference on Artificial Intelligence and Statistics (AISTATS)*. Ed. by K. Murphy and B. et al. Schölkopf. Vol. 5. 12. 2009, pp. 448–455. URL: http://machinelearning.wustl.edu/mlpapers/paper%5C_files/AISTATS09%5C_SchmidtBFM.pdf.

[35] Peter Selinger. *Lecture Notes on the Lambda Calculus*. Ed. by Dpt. of Mathematics and Statistics. Halifax, Canada: Dalhousie University, 2013. URL: http://www.mathstat.dal.ca/~selinger/papers/lambdanotes.pdf.

[36] Marcus Spies. "Knowledge Discovery from Constrained Relational Data: A Tutorial on Markov Logic Networks". In: *Business Intelligence - Second European Summer School, eBISS 2012, Brussels, Belgium, July 15-21, 2012, Tutorial Lectures*. Ed. by Marie-Aude Aufaure and Esteban Zimányi. Vol. 138. Lecture Notes in Business Information Processing. Springer, 2013, pp. 78–102. ISBN: 978-3-642-36317-7.

[37] Marcus Spies and Monika Jungemann-Dorner. “Big Textual Data Analytics and Knowledge Management”. In: *Big Data Computing*. Ed. by Rajendra Akerkar. Chapman and Hall/CRC, 2013. Chap. 23, pp. 501–537. ISBN: 978-1-4665-7837. DOI: doi:10.1201/b16014-23. URL: http://dx.doi.org/10.1201/b16014-1.

[38] Mark Steedman and Jason Baldridge. “Combinatory Categorial Grammar”. In: ed. by R. Borsley and . K. Borjars (eds.) 181-224. Blackwell, 2011.

[39] C. Sutton and A. McCallum. “An Introduction to Conditional Random Fields”. In: *ArXiv e-prints* (Nov. 2010). arXiv: 1011.4088 [stat.ML].

[40] *The Stanford Parser: A statistical parser*. Apr. 2014. URL: http://nlp.stanford.edu/software/lex-parser.shtml.

[41] C. Wang et al. “Relation extraction and scoring in DeepQA”. In: *IBM Journal of Research and Development* 56.3.4 (May 2012), 9:1–9:12. ISSN: 0018-8646. DOI: 10.1147/JRD.2012.2187239.

[42] Paul C. Zikopoulos et al. *Harness the Power of Big Data: The IBM Big Data Platform*. New York: McGraw-Hill, 2013.