Automatic Knowledge Base Construction using Probabilistic Extraction, Deductive Reasoning, and Human Feedback

Daisy Zhe Wang **Yang Chen** **Sean Goldberg** **Christan Grant** **Kun Li**
Department of Computer and Information Science and Engineering,
University of Florida
{daisyw,yang,sean,cgrant,kli}@cise.ufl.edu

Abstract

We envision an automatic knowledge base construction system consisting of three interrelated components. MADDEN is a knowledge extraction system applying statistical text analysis methods over database systems (DBMS) and massive parallel processing (MPP) frameworks; PROBKB performs probabilistic reasoning over the extracted knowledge to derive additional facts not existing in the original text corpus; CAMEL leverages human intelligence to reduce the uncertainty resulting from both the information extraction and probabilistic reasoning processes.

1 Introduction

In order to build a better search engine that performs semantic search in addition to keyword matching, a knowledge base that contains information about all the entities and relationships on the web and beyond is needed. With recent advances in technology such as cloud computing and statistical machine learning (SML), automatic knowledge base construction is becoming possible and is receiving more and more interest from researchers. We envision an automatic knowledge base (KB) construction system that includes three components: probabilistic extraction, deductive reasoning, and human feedback.

Much research has been conducted on text analysis and extraction at web-scale using SML models and algorithms. We built our parallelized text analysis library MADDEN on top of relational database systems and MPP frameworks to achieve efficiency and scalability.

The automatically extracted information contains errors, uncertainties, and probabilities. We use a probabilistic database to preserve uncertainty in data representations and propagate probabilities through query processing.

Further, not all information can be extracted from the Web (Schoenmackers et al., 2008). A probabilistic deductive reasoning system is needed to infer additional facts from the existing facts and rules extracted by MADDEN.

Finally, we propose to use human feedback to improve the quality of the machine-generated knowledge base since SML methods are not perfect. Crowdsourcing is one of the ways to collect this feedback and though much slower, it is often more accurate than the state-of-the-art SML algorithms.

2 System Overview

Our vision of the automatic knowledge base construction process consists of three main components as shown in Figure 1.

The first component is a knowledge extraction system called MADDEN that sits on top of a probabilistic database system such as BAYESSTORE or PrDB and treats probabilistic data, statistical models, and algorithms as first-class citizens (Wang et al., 2008; Sen et al., 2009). MADDEN specifically implements SML models and algorithms on database systems (e.g., PostgreSQL) and massive parallel processing (MPP) frameworks (e.g., Greenplum) to extract various types of information from the text corpus, including *entities*, *relations*, and *rules*. Different types of information are extracted by different text analysis tasks. For example, the

Proc. of the Joint Workshop on Automatic Knowledge Base Construction & Web-scale Knowledge Extraction (AKBC-WEKEX), pages 106–110,
NAACL-HLT, Montréal, Canada, June 7-8, 2012. ©2012 Association for Computational Linguistics

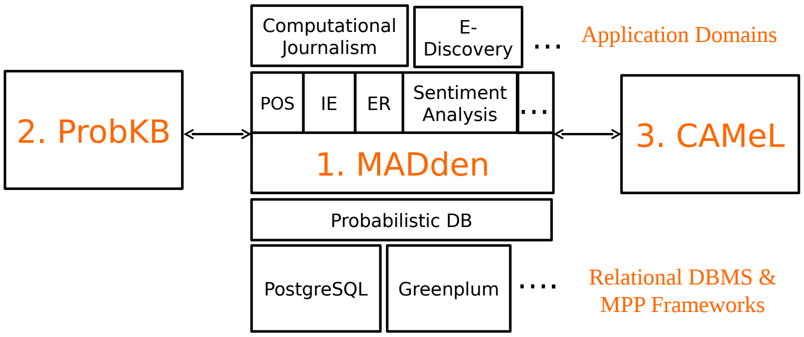

Figure 1: Architecture for Automatic Knowledge Base Construction

named entity recognition (NER) task extracts different types of *entities* including people, companies, and locations from text.

The second component is a probabilistic reasoning system called PROBKB. Given a set of entities, relations, and rules extracted from a text corpus (e.g., WWW), PROBKB enables large-scale inference and reasoning over uncertain entities and relations using probabilistic first-order logic rules. Such inference would generate a large number of new facts that did not exist in the original text corpus. The uncertain knowledge base is modeled by Markov logic networks (MLN) (Domingos et al., 2006). In this model, the probabilistic derivation of new facts from existing ones is equivalent to inference over the MLNs.

The third component is a crowd-based human feedback system called Crowd-Assisted Machine Learning or CAMEL. Given the set of extracted and derived facts, rules and their uncertainties, CAMEL leverages the human computing power from crowdsourcing services to improve the quality of the knowledge base. Based on the probabilities associated with the extracted and derived information in an uncertain knowledge base, CAMEL effectively selects and formulates questions to push to the crowd.

The resulting knowledge base constructed from the extraction, derivation, and feedback steps can be used in various application domains such as computational journalism and e-discovery.

3 MADDEN: Statistical Text Analysis on MPP Frameworks

The focus of the MADDEN project has been to integrate statistical text analytics into DBMS and MPP frameworks to achieve scalability and parallelization. Structured and unstructured text are core assets for data analysis. The increasing use of text analysis in enterprise applications has increased the expectation of customers and the opportunities for processing big data. The state-of-the-art text analysis and extraction tools are increasingly found to be based on statistical models and algorithms (Jurafsky et al., 2000; Feldman and Sanger, 2007).

Basic text analysis tasks include part-of-speech (POS) tagging, named entity extraction (NER), and entity resolution (ER) (Feldman and Sanger, 2007). Different statistical models and algorithms are implemented for each of these tasks with different runtime-accuracy trade-offs. An example entity resolution task could be to find all mentions in a text corpus that refer to a real-world entity X. Such a task can be done efficiently by approximate string matching (Navarro, 2001) techniques to find all mentions that approximately match the name of entity X. Approximate string matching is a high recall and low precision approach when compared to state-of-the-art collective entity resolution algorithms based on statistical models like Conditional Random Fields (CRFs) (Lafferty et al., 2001).

CRFs are a leading probabilistic model for solving many text analysis tasks, including POS tagging, NER, and ER (Lafferty et al., 2001). To support sophisticated text analysis, we implement four key methods: text feature extraction, inference over a

CRF (Viterbi), Markov chain Monte Carlo (MCMC) inference, and approximate string matching.

Text Feature Extraction: To analyze text, features need to be extracted from documents and it can be an expensive operation. To achieve results with high accuracy, CRF methods often compute hundreds of features over each token in the document, which can be high cost. Features are determined by functions over the sets of tokens. Examples of such features include: (1) dictionary features: *does this token exist in a provided dictionary?* (2) regex features: *does this token match a provided regular expression?* (3) edge features: *is the label of a token correlated with the label of a previous token?* (4) word features: *does this the token appear in the training data?* and (5) position features: *is this token the first or last in the token sequence?* The optimal combination of features depends on the application.

Approximate String Matching: A recurring primitive operation in text processing applications is the ability to match strings approximately. The technique we use is based on qgrams (Gravano et al., 2001). We create and index 3-grams over text. Given a string "Tim Tebow" we can create a 3-gram by using a sliding window of 3 characters over this text string. Given two strings we can compare the overlap of two sets of corresponding 3-grams and compute a similarity as the approximate matching score.

Once we have the features, the next step is to perform inference on the model. We also implemented two types of statistical inference within the database: Viterbi (when we only want the most likely answer from a linear-chain CRF model) and MCMC (when we want the probabilities or confidence of an answer from a general CRF model).

Viterbi Inference: The *Viterbi* dynamic programming algorithm (Manning et al., 1999) is a popular algorithm to find the top-k most likely labelings of a document for (linear-chain) CRF models.

To implement the Viterbi dynamic programming algorithm we experimented with two different implementations of macro-coordination over time. First, we chose to implement it using a combination of recursive SQL and window aggregate functions. We discussed this implementation at some length in earlier work (Wang et al., 2010). Second, we chose to implement a Python UDF that uses iterations to drive the recursion in Viterbi. In the Greenplum MPP Framework, Viterbi can be run in parallel over different subsets of the document on a multi-core machine.

MCMC Inference: MCMC methods are classical sampling algorithms that can be used to estimate probability distributions. We implemented two MCMC method: Gibbs sampling and Metropolis-Hastings (MCMC-MH).

The MCMC algorithms involve iterative procedures where the current values depend on previous iterations. We use SQL window aggregates for macro-coordination in this case, to carry "state" across iterations to perform the Markov-chain process. We discussed this implementation at some length in recent work (Wang et al., 2011). We are currently developing MCMC algorithms over Greenplum DBMS.

4 ProbKB: Probabilistic Knowledge Base

The second component of our system is ProbKB, a probabilistic knowledge base designed to derive implicit knowledge from entities, relations, and rules extracted from a text corpus by knowledge extraction systems like MADden. Discovering new knowledge is a crucial step towards knowledge base construction since many valuable facts are not explicitly stated in web text; they need to be inferred from extracted facts and rules.

ProbKB models uncertain facts as Markov logic networks (MLN) (Domingos et al., 2006). Markov logic networks are proposed to unify first-order logic and statistical inference by attaching a weight to each first-order formula (rule). These weights reflect our confidence of the rules being true. To obtain these weighted formulae, we have used natural language processing (NLP) methods to extract entities and relations as described in Section 3 and learned the formulae from the extractions.

One challenge in applying the MLN model is propagating the uncertainty of facts and rules in the inference process. A naive method may be discarding facts with low confidence using ad-hoc thresholds and heuristics, but we decided to maintain all the facts in our knowledge base regardless of their confidences. The rationale behind this is that some facts may have low confidence due to absence or inaccessibility of evidence rather than being incorrect; they may prove to be true when new extractions are available as supporting evidence.

We are experimenting on some state-of-the-art implementations of MLNs like TUFFY (Niu et al., 2011) as a base to develop our large-scale probabilistic inference engine. Taking the MLN and uncertain facts, rules, and their confidence as inputs, the system is able to answer queries like "how likely will Bob develop a cancer?". Though TUFFY is able to handle uncertainties resulted from extraction systems, it is no easy task for the system to scale up to tens of millions of facts and thousands of rules. To address this problem, we are currently researching several possible ways to parallelize the inference computation. One challenge for parallelization is data dependency: the result set (derived facts) of one rule may affect that of another. As a first attempt, we are looking at two different partitioning strategies: partition by rules and partition by facts.

In addition to partitioning techniques, we are also trying to evaluate the possibility of implementing MLNs on different MPP frameworks: Greenplum Database, HadoopDB, and Datapath (Arumugam et al., 2010). These database systems allow effective parallel processing of big data and running of inference algorithms, which is essential for scaling up probabilistic reasoning in the PROBKB project.

5 CAMEL: Crowd-Assisted Machine Learning

The final proposed component for automatic construction of a knowledge base is a crowd-based system, CAMEL, designed for improving uncertainty. CAMEL is built on top of an existing probabilistic knowledge or database like PROBKB and MADDEN.

In addition to using SML techniques for large scale analysis, an increasing trend has been to harness human computation in a distributed manner using crowdsourcing (Quinn et al., 2010; Sorokin and Forsyth, 2008). Benefits can be gained in problems that are too difficult or expensive for computers. Services like Amazon Mechanical Turk (AMT) (Ipeirotis, 2010) have led the way by setting up an infrastructure that allows payment for the combined resources of up to hundreds of thousands of people.

SML is not perfect: for some simple NLP tasks it achieves a relatively high accuracy while for other ones involving context and reasoning the results are much worse. Cases where the model is unable to adequately reason about a difficult piece of data introduces large uncertainties into the output. The need for a new type of data cleaning process has emerged. As discussed in Section 4, one approach is to threshold uncertainty and throw away those facts the machine is unable to reason about, leaving the knowledge base incomplete. Another approach is to convert these high uncertainty examples into questions for the crowd to answer.

The main tenets of CAMEL are its selection model and integration model as described below.

Selection Model: The first important feature of CAMEL is its ability to distinguish and select the most uncertain fields in the knowledge base. For tasks involving CRFs (MADDEN), each hidden node can be marginalized to find a probability distribution over the label space. From the marginal distribution, we can attach a *marginal entropy* to each node in the graph. Our algorithm selects the highest entropy node to be sent to the crowd. Additional research is being done to take advantage of specifics of the graph structure such as the connectivity and dependency relationships of each node.

Integration Model: Questions are posted on AMT and are answered by a number of different Turkers, generally three or five per question. The golden standard for aggregating the crowd response has been to take a majority vote. Since our system is built on top of a probabilistic knowledge base KB, we want to establish a distribution over the possible answers based on the received responses. We use the machinery of Dempster-Shafer's (DS) Theory of Evidence (Dempster, 1967; Shafer, 1976) for combining results in a probabilistic manner. Using an Expectation-Maximization algorithm proposed by Dawid and Skene (Dawid and Skene, 1979) for assessing Turker quality and confidence, answers are aggregated into a single distribution for reinsertion into the database. The more Turkers that are queried, the more fine-tuned the distribution becomes.

6 Conclusion

In this short paper, we described our vision of an automatic knowledge base construction system consisting of three major components—extraction, reasoning, and human feedback. The resulting system is expected to be scalable, efficient, and useful in vaiours application domains.

Acknowledgments

This material is based upon work supported by the national Science Foundation Graduate Research Fellowship under Grant No. DGE-0802270 and a generous gift from Greenplum.

References

Subi Arumugam, Alin Dobra, Christopher M. Jermaine, Niketan Pansare, and Luis Perez. 2010. The datapath system: a data-centric analytic processing engine for large data warehouses. In *Proceedings of the 2010 international conference on Management of data*, SIGMOD '10, pages 519–530, New York, NY, USA. ACM.

A. P. Dawid and A. M. Skene. 1979. Maximum likelihood estimation of observer error-rates using the em algorithm. *Applied Statistics*, 28(1):20–28.

A. P. Dempster. 1967. Upper and lower probabilities induced by a multivalued mapping. *Annals of Mathematical Statistics*, 38:325–339.

Pedro Domingos, Stanley Kok, Hoifung Poon, Matthew Richardson, and Parag Singla. 2006. Unifying logical and statistical ai. In *Proceedings of the 21st national conference on Artificial intelligence - Volume 1*, AAAI'06, pages 2–7. AAAI Press.

R. Feldman and J. Sanger. 2007. *The text mining handbook: advanced approaches in analyzing unstructured data*. Cambridge Univ Pr.

L. Gravano, P.G. Ipeirotis, H.V. Jagadish, N. Koudas, S. Muthukrishnan, L. Pietarinen, and D. Srivastava. 2001. Using q-grams in a dbms for approximate string processing. *IEEE Data Engineering Bulletin*, 24(4):28–34.

P.G. Ipeirotis. 2010. Analyzing the amazon mechanical turk marketplace. *XRDS: Crossroads, The ACM Magazine for Students*, 17(2):16–21.

D. Jurafsky, J.H. Martin, A. Kehler, K. Vander Linden, and N. Ward. 2000. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*, volume 2. Prentice Hall New Jersey.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

C.D. Manning, H. Schütze, and MITCogNet. 1999. *Foundations of statistical natural language processing*, volume 999. MIT Press.

Gonzalo Navarro. 2001. A guided tour to approximate string matching. *ACM Comput. Surv.*, 33(1):31–88, March.

Feng Niu, Christopher Ré, AnHai Doan, and Jude W. Shavlik. 2011. Tuffy: Scaling up statistical inference in markov logic networks using an rdbms. *CoRR*, abs/1104.3216.

A. Quinn, B. Bederson, T. Yeh, and J. Lin. 2010. CrowdFlow: Integrating Machine Learning with Mechanical Turk for Speed-Cost-Quality Flexibility. Technical report, University of Maryland, May.

Stefan Schoenmackers, Oren Etzioni, and Daniel S. Weld. 2008. Scaling textual inference to the web. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 79–88, Stroudsburg, PA, USA. Association for Computational Linguistics.

Prithviraj Sen, Amol Deshpande, and Lise Getoor. 2009. Prdb: managing and exploiting rich correlations in probabilistic databases. *The VLDB Journal*, 18(5):1065–1090, October.

G. Shafer. 1976. *A mathematical theory of evidence*. Princeton university press.

Alexander Sorokin and David Forsyth. 2008. Utility data annotation with Amazon Mechanical Turk. In *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8. IEEE, June.

Daisy Zhe Wang, Eirinaios Michelakis, Minos Garofalakis, and Joseph M. Hellerstein. 2008. Bayesstore: managing large, uncertain data repositories with probabilistic graphical models. *Proc. VLDB Endow.*, 1:340–351, August.

Daisy Zhe Wang, Michael J. Franklin, Minos N. Garofalakis, and Joseph M. Hellerstein. 2010. Querying probabilistic information extraction. *PVLDB*, 3(1):1057–1067.

Daisy Zhe Wang, Michael J. Franklin, Minos Garofalakis, Joseph M. Hellerstein, and Michael L. Wick. 2011. Hybrid in-database inference for declarative information extraction. In *Proceedings of the 2011 international conference on Management of data*, SIGMOD '11, pages 517–528, New York, NY, USA. ACM.