DeepDive: Web-scale Knowledge-base Construction using Statistical Learning and Inference

Feng Niu Ce Zhang Christopher Ré Jude Shavlik

Department of Computer Sciences
University of Wisconsin-Madison, USA

{leonn,czhang,chrisre,shavlik}@cs.wisc.edu

ABSTRACT

We present an end-to-end (live) demonstration system called DeepDive that performs knowledge-base construction (KBC) from hundreds of millions of web pages. DeepDive employs statistical learning and inference to combine diverse data resources and best-of-breed algorithms. A key challenge of this approach is scalability, i.e., how to deal with terabytes of imperfect data efficiently. We describe how we address the scalability challenges to achieve web-scale KBC and the lessons we have learned from building DeepDive.

1. INTRODUCTION

Knowledge-base construction (KBC) is the process of populating a knowledge base (KB) with facts (or assertions) extracted from text. It has recently received tremendous interest from academia, e.g., CMU's NELL [2] and MPI's YAGO [7,9], and from industry, e.g., IBM's DeepQA [5] and Microsoft's EntityCube [16]. To achieve high quality, these systems leverage a wide variety of data resources and KBC techniques. A crucial challenge that these systems face is coping with imperfect or conflicting information from multiple sources [3, 13]. To address this challenge, we present an end-to-end KBC system called DeepDive.[1] DeepDive went live in January 2012 after processing the 500M English web pages in the ClueWeb09 corpus[2], and since then has been adding several million newly-crawled webpages every day. Figure 1 shows several screenshots of DeepDive.

Similar to YAGO [7,9] and EntityCube [16], DeepDive is based on the classic Entity-Relationship (ER) model [1] and employs popular techniques such as *distant supervision* [15] and the Markov logic language [11] to combine a variety of signals. However, DeepDive goes deeper in two ways: (1) Unlike prior large-scale KBC systems, DeepDive performs deep natural language processing (NLP) to extract useful linguistic features such as named-entity mentions and dependency paths[3] from terabytes of text; and (2) DeepDive performs web-scale statistical learning and inference using classic data-management and optimization techniques.

Figure 2 depicts the current architecture of DeepDive. To populate a knowledge base, DeepDive first converts diverse input data (e.g., raw corpora and ontologies) into relational features using standard NLP tools and custom code. These features are then used to train statistical models representing the correlations between linguistic patterns and target relations. Finally, DeepDive combines the trained statistical models with additional knowledge (e.g., domain knowledge) into a Markov logic program that is then used to transform the relational features (e.g., candidate entity mentions and linguistic patterns) into a knowledge base with entities, relationships, and their provenance.

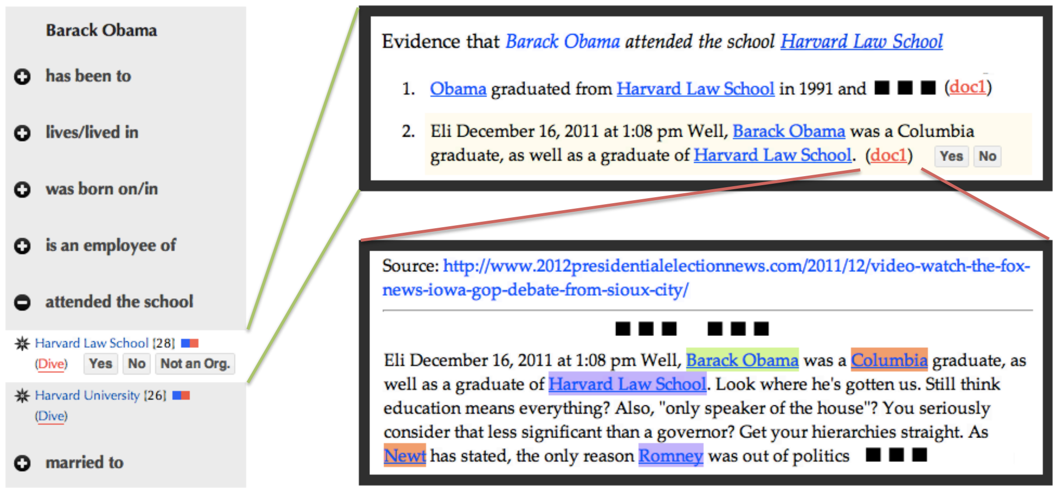
Given the amount of data and depth of processing, a key challenge is scaling feature extraction (see Figure 2). For example, while deep linguistic features such as dependency paths are useful for relationship extraction, it takes about 100K CPU hours for our NLP pipeline to finish processing ClueWeb09. Although we have access to a 100-node Hadoop cluster, we found that the throughput of Hadoop is often limited by pathological data chunks that take very long to process or even crash (due to Hadoop's no-task-left-behind failure model). Fortunately, thanks to the Condor infrastructure[4], we were able to finish feature extraction on ClueWeb09 within a week by opportunistically assigning jobs on hundreds of workstations and shared cluster machines using a best-effort failure model.

A second scalability challenge is statistical learning and inference. There are two aspects of scalability with statistical learning: scale of training examples and scale of performance. To scale up the amount of training examples for KBC, we employ the distant supervision technique [8,14,15] that automatically generates what are called silver-standard examples by heuristically aligning raw text with an existing knowledge base such as Freebase.[5] To scale up the performance of machine learning, we leverage the Bismarck system [4] that executes a wide variety of machine learning techniques inside an RDBMS. For statistical inference, DeepDive employs a popular statistical-inference framework called *Markov logic*. In Markov logic, one can write first-order logic rules with weights (that intuitively model

[1]`http://research.cs.wisc.edu/hazy/deepdive`
[2]`http://lemurproject.org/clueweb09.php/`

[3]`http://nlp.stanford.edu/software/`
[4]`http://research.cs.wisc.edu/condor/`
[5]`http://freebase.com`

VLDS'12 August 31, 2012. Istanbul, Turkey.



Evidence that *Barack Obama attended the school Harvard Law School*

1. Obama graduated from Harvard Law School in 1991 and ■ ■ ■ (doc1)
2. Eli December 16, 2011 at 1:08 pm Well, Barack Obama was a Columbia graduate, as well as a graduate of Harvard Law School. (doc1) Yes No

Source: http://www.2012presidentialelectionnews.com/2011/12/video-watch-the-fox-news-iowa-gop-debate-from-sioux-city/

■ ■ ■ ■ ■ ■

Eli December 16, 2011 at 1:08 pm Well, Barack Obama was a Columbia graduate, as well as a graduate of Harvard Law School. Look where he's gotten us. Still think education means everything? Also, "only speaker of the house"? You seriously consider that less significant than a governor? Get your hierarchies straight. As Newt has stated, the only reason Romney was out of politics ■ ■ ■

Figure 1: Screenshots of DeepDive showing facts about Barack Obama and provenance. (Left) Facts about Obama in DeepDive; (Top Right) Sentences mentioning the fact that Obama went to Harvard Law School; (Bottom Right) Text from a web page annotated with entity mentions.

our confidence in a rule); this allows one to capture rules that are likely, but not certain, to be correct. A Markov logic program (aka *Markov logic network*, or simply MLN) specifies what (evidence) data are available, what predictions to make, and what constraints and correlations there are [10]. However, when trying to apply Markov logic to KBC, we found that existing MLN systems such as ALCHEMY[6] do not scale to our datasets. To cope, we designed and implemented two novel approaches to MLN inference by leveraging data-management and optimization techniques.

In addition to statistical learning and inference, we have found debugging and tuning based on the output of a KBC system to be an effective method to improve KBC quality. To support systematic debugging and tuning, it is important that the underlying statistical models are *well-calibrated*, i.e., predictions with probability around p should have an actual accuracy around p as well. Such calibration is an integral part of the development process of DEEPDIVE.

We describe the DEEPDIVE architecture in Section 2 and some implementation details in Section 3. We summarize the lessons we have learned from DEEPDIVE as follows:

Inference and Learning. Statistical inference and learning were bottlenecks when we first started DEEPDIVE, but we can now scale them with data-management and optimization techniques.

Feature Extraction. Good features are a key bottleneck for KBC; with a scalable inference and learning infrastructure in place, we can now focus on gathering and tuning features for DEEPDIVE.

Debugging and Tuning. Developing KBC systems is an iterative process; systematic debugging and tuning requires well-calibrated statistical models.

2. DEEPDIVE ARCHITECTURE

We first describe a simple KBC model that we use in DEEPDIVE, and then briefly discuss the infrastructure that enables web-scale KBC in DEEPDIVE, namely how we scale up web-scale feature extration, machine learning for KBC, and statistical inference in Markov logic, respectively.

A Conceptual KBC Model. DEEPDIVE adopts the classic Entity-Relationship (ER) model [1]: the schema of the target knowledge base (KB) is specified by an ER graph $G = (\bar{E}, \bar{R})$ where $\bar{E}$ is one or more sets of *entities* (e.g., people and organizations), and $\bar{R}$ is a set of relationships. Define $\mathcal{E}(G) = \cup_{E \in \bar{E}} E$, i.e., the set of known entities. To specify a KBC task to DEEPDIVE, one provides the schema G and a corpus D. Each document $d_i \in D$ consists of a set of (possibly overlapping) *text spans* (e.g., tokens or sentences) $T(d_i)$. Text spans referring to entities or relationships are called *mentions* (see Figure 4). Define $\mathcal{T}(D) = \cup_{d_i \in D} \mathcal{T}(d_i)$. Our goal is to accurately populate the following tables:

- Entity-mention table $M(\mathcal{E}(G), \mathcal{T}(D))$.[7]
- Relationship-mention tables $M_{R_i} \subseteq \mathcal{T}(D)^{k+1}$ for each $R_i \in \bar{R}$; k is R_i's arity, and the first k attributes (resp. last attribute) are entity (resp. relationship) mentions.
- Relationship tables $R_i \in \bar{R}$.

Note that R_i can be derived from M_E and M_{R_i}. By the same token, M_E and M_{R_i} provide provenance that connects the KB back to the documents supporting each fact. The process of populating M_E is called *entity linking*; the process of populating M_{R_i} is called *relation extraction*. Intuitively, the goal is to produce an instance J of these tables that is as large as possible (high recall) and as correct as possible (high precision). As shown in Figure 4, DEEPDIVE populates the target KB based on signals from mention-level features (over text spans, e.g., positions, contained words, and matched regular expressions) and entity-level features (over the target KB, e.g., age, gender, and alias).

[6]`http://alchemy.cs.washington.edu`

[7]For simplicity, we assume that all entities are known, but DEEPDIVE supports generating novel entities for the KB as well (e.g., by clustering "dangling" mentions).

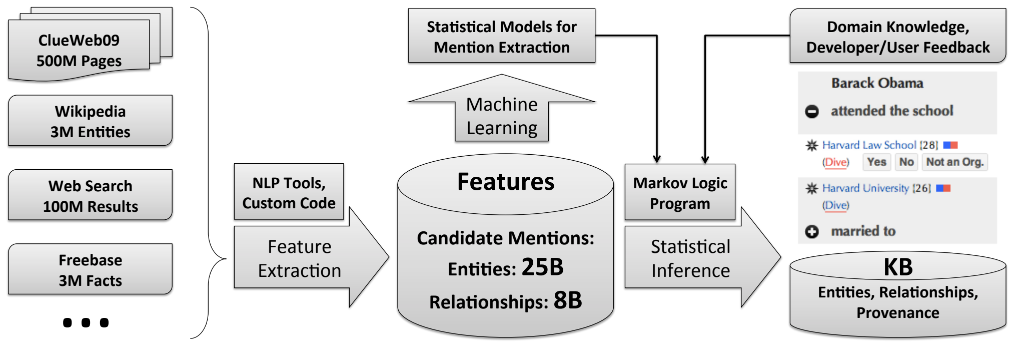


Figure 2: Architecture of DeepDive. DeepDive takes as input diverse data resources, converts them into relational features, and then performs machine learning and statistical inference to construct a KB.

Scaling Feature Extraction at Web Scale. To scale DeepDive to web-scale KBC tasks, we employ high-throughput parallel computing frameworks such as Hadoop[8] and Condor for feature extraction. We use the Hadoop File System for storage, but found a 100-node MapReduce cluster to be insufficient for ClueWeb: (1) Hadoop's all-or-nothing approach to failure handling hinders throughput, and (2) the number of cluster machines for Hadoop is limited. Fortunately, the Condor infrastructure supports a best-effort failure model, i.e., a job may finish successfully even when Condor fails to process a small portion of the input data. Moreover, Condor allows us to simultaneously leverage thousands of machines from across a department, an entire campus, or even the nation-wide Open Science Grid.[9]

Scaling Machine Learning for KBC. Traditional KBC systems rely on manual annotations or domain-specific rules provided by experts, both of which are scarce resources. To remedy these problems, recent years have seen interest in the *distant supervision* approach for relation extraction [8, 14, 15]. The input to distant supervision is a set of *seed facts* for the target relation together with an (unlabeled) text corpus, and the output is a set of (noisy) annotations that can be used by any machine learning technique to train a statistical relation-extraction model. For example, given the target relation `BirthPlace`(person, place) and a known fact `BirthPlace`(John, Springfield), the sentence "*John was born in Springfield in 1946*" would qualify as a positive training example. Despite the noise in such examples, Zhang et al. [15] show that the quality of the TAC-KBP[10] relation-extraction benchmark improves significantly as we increase the training corpus size (Figure 3). DeepDive learns its relation-extraction models (as logistic regression classifiers) on about 1M examples (see Section 3) using the RDBMS-based Bismarck system [4].

Scaling Statistical Inference in Markov Logic. To scale up statistical inference in Markov logic, DeepDive employs the Tuffy [10] and Felix[11] systems. Tuffy is based on the observation that MLN inference consists of a grounding step that essentially performs relational operations, and a search (or sampling) step that often comprises multiple independent subproblems. Thus, Tuffy achieves orders of magnitude speed-up in grounding (compared to Alchemy) by translating grounding into SQL statements that are executed by an RDBMS. Moreover, Tuffy performs graph partitioning at the search step to achieve scalability in inference and (serendipitously) improved quality. Felix is based on the observation that an MLN program (especially those for KBC) often contains routine subtasks such as classification and coreference resolution; these subtasks have specialized algorithms with high efficiency and quality. Thus, instead of solving a whole MLN with generic inference algorithms, Felix splits the program into multiple parts and solves subtasks with corresponding specialized algorithms. To resolve possible conflicts between predictions from different tasks, Felix employs the classic *dual decomposition* technique.

[8]http://hadoop.apache.org/
[9]http://www.opensciencegrid.org
[10]http://nlp.cs.qc.cuny.edu/kbp/2010/
[11]http://research.cs.wisc.edu/hazy/felix

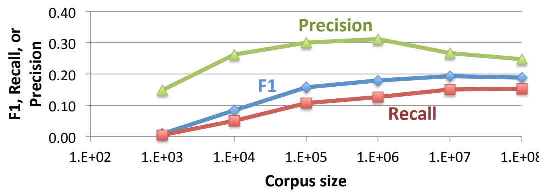


Figure 3: The TAC-KBP quality improves as we increase the number of ClueWeb documents used for distant supervision [15]. Note that F1 and Recall continue to improve as DeepDive processes increasingly large corpora (we do not yet have an explanation for the dip in precision).

3. IMPLEMENTATION DETAILS

A key tenet of DeepDive is that statistical learning and inference enables one to build high-quality KBC systems (see Figure 1) by combining diverse resources. We briefly describe more technical details of DeepDive to conceretely demonstrate the advantage of this approach.

Entity Linking. Recall that entity linking is the task of mapping a textual mention to a real-world entity. We run the state-of-the-art StanfordNER (Named Entity Recogni-

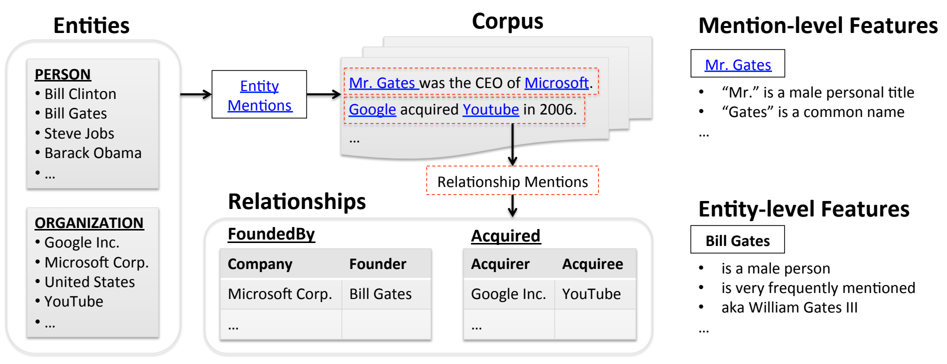


Figure 4: An illustration of the KBC model in DeepDive.

tion)[12] to extract textual mentions and corresponding entity types. DeepDive then tries to map each mention to a Wikipedia entry using the following signals: string matching, Wikipedia redirects and inter-page anchor text, Google and Bing search results that link to Wikipedia pages, entity type compatibility between StanfordNER and Freebase, person-name coreference based on heuristics and proximity, etc. We write a couple dozen MLN rules for these data resources and then train the rule weights using the entity-linking training data from TAC-KBP. Our entity-linking component achieves an F1 score of 0.80 on the TAC-KBP benchmark (human performance is around 0.90). As shown in Figure 1, DeepDive also solicits user feedback on the entity types; we plan to integrate such feedback into the statistical inference process.

Relation Extraction. The relation-extraction models in DeepDive are trained using the methods described in Zhang et al. [15]. Specifically, during feature extraction, we perform dependency parsing using MaltParser[13] and Ensemble.[14] We use Freebase and about 2M high-quality news and blog articles provided by TAC-KBP to perform distant supervision, generating about 1M training examples over 20 target relations (including those shown in Figure 1). We use *sparse logistic regression* (ℓ_1 regularized) classifiers [12] to train statistical relation-extraction models using both lexical (e.g., word sequences) and syntactic (e.g., dependency paths) features. We achieved an F1 score of 0.31 on the TAC-KBP relation extraction benchmark (lower than only the top participant in TAC-KBP 2010 [6]).

4. CONCLUDING REMARKS

We presented DeepDive, an end-to-end demonstration system that performs knowledge-base construction from the web. DeepDive demonstrates that a promising approach to KBC is to integrate diverse data resources and best-of-breed algorithms via statistical learning and inference. We discussed the key lessons we have learned from building DeepDive – including feature extraction, statistical learning and inference, and systematic debugging – and hope that they are of value to other researchers.

[12]`http://nlp.stanford.edu/ner/index.shtml`
[13]`http://www.maltparser.org/`
[14]`http://www.surdeanu.name/mihai/ensemble/`

5. ACKNOWLEDGMENTS

We gratefully acknowledge the support of DARPA grant FA8750-09-C-0181. CR is also generously supported by NSF CAREER award IIS-1054009, ONR award N000141210041, and gifts or research awards from Google, Greenplum, Johnson Controls, Inc., LogicBlox, and Oracle. We thank the generous support from the Center for High Throughput Computing and Miron Livny's Condor research group. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the above companies, DARPA, or the US government.

6. REFERENCES

[1] A. Calì, G. Gottlob, and A. Pieris. Query answering under expressive entity-relationship schemata. *Conceptual Modeling–ER*, 2010.
[2] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. Hruschka Jr, and T. Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, 2010.
[3] L. Fang, A. D. Sarma, C. Yu, and P. Bohannon. Rex: Explaining relationships between entity pairs. *Proc. VLDB Endow.*, 5(3):241–252, Nov. 2011.
[4] X. Feng, A. Kumar, B. Recht, and C. Ré. Towards a unified architecture for in-RDBMS analytics. In *SIGMOD*, 2012.
[5] D. Ferrucci et al. Building Watson: An overview of the DeepQA project. *AI Magazine*, 2010.
[6] H. Ji, R. Grishman, H. Dang, K. Griffitt, and J. Ellis. Overview of the TAC 2010 knowledge base population track. In *Text Analysis Conference*, 2010.
[7] G. Kasneci, M. Ramanath, F. Suchanek, and G. Weikum. The YAGO-NAGA approach to knowledge discovery. *SIGMOD Record*, 2008.
[8] M. Mintz, S. Bills, R. Snow, and D. Jurafsky. Distant supervision for relation extraction without labeled data. In *ACL*, pages 1003–1011, 2009.
[9] N. Nakashole, M. Theobald, and G. Weikum. Scalable knowledge harvesting with high precision and high recall. In *WSDM*, 2011.
[10] F. Niu, C. Ré, A. Doan, and J. Shavlik. Tuffy: Scaling up statistical inference in Markov logic networks using an RDBMS. In *VLDB*, 2011.
[11] M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 2006.
[12] R. Tibshirani. Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society.*, 1996.
[13] G. Weikum and M. Theobald. From information to knowledge: Harvesting entities and relationships from web sources. In *PODS*, 2010.
[14] F. Wu and D. Weld. Autonomously semantifying Wikipedia. In *CIKM*, 2007.
[15] C. Zhang, F. Niu, C. Ré, and J. Shavlik. Big data versus the crowd: Looking for relationships in all the right places. In *ACL*, 2012.
[16] J. Zhu, Z. Nie, X. Liu, B. Zhang, and J. Wen. Statsnowball: A statistical approach to extracting entity relationships. In *WWW*, 2009.