

Evaluating the Quality of a Knowledge Base Populated from Text

James Mayfield

Human Language Technology
Center of Excellence
Johns Hopkins University
james.mayfield@jhuapl.edu

Tim Finin

Computer Science and Electrical Engineering
University of Maryland,
Baltimore County
finin@umbc.edu

Abstract

The steady progress of information extraction systems has been helped by sound methodologies for evaluating their performance in controlled experiments. Annual events like MUC, ACE and TAC have developed evaluation approaches enabling researchers to score and rank their systems relative to reference results. Yet these evaluations have only assessed component technologies needed by a knowledge base population system; none has required the construction of a knowledge base that is then evaluated directly. We describe an approach to the direct evaluation of a knowledge base and an instantiation that will be used in a 2012 TAC Knowledge Base Population track.

1 Introduction

Many activities might fall under the rubric of automatic knowledge base (KB) generation, including information extraction, entity linking, open information extraction and machine reading. The task is broad and challenging: process a large text corpus to extract a KB schema or ontology and populate it with entities, relations and facts. The term *knowledge base population* (KBP) is often used for the narrower task in which we start with a predefined and fixed KB schema or *ontology* and focus on the problem of extracting information from a text corpus to populate the KB with entities, relations and facts using that ontology.

To evaluate progress on such systems, we must answer the question “how do you know that the knowledge base you built is any good?” Before we can say whether an automatically created knowledge base is good, we must first say what a knowledge base is. We define a knowledge base as a combination of four things: a database of facts; a descriptive

schema for those facts; a collection of existing background knowledge; and inference capability.

We are concerned in this paper primarily with knowledge bases that use a known schema. Some of the work in open information extraction addresses the question of how a knowledge schema could be derived from text. While this is important work, it nonetheless falls outside the scope of our current inquiry. We seek to assess whether a KB populated according to a known schema accurately encodes the knowledge sources used to create it. These underlying knowledge sources might be structured (e.g., a database), semi-structured (e.g., Wikipedia Infoboxes), or entirely unstructured (e.g., free text). We also do not wish to directly evaluate the breadth or accuracy of the KB’s background knowledge. Our proposed approach can be used to evaluate the KB’s inferencing ability; however, for the current study, we require that the KB materialize all of the relevant facts it can infer. We also require that the KB justify, where appropriate, the sources (e.g., a document) from which each fact is derived.

Our evaluation approach is characterized by three design decisions. First, we require that KBs be submitted in a simple abstract format that we use to create an equivalent KB in RDF. This gives us a well defined and relatively simple KB that can be tested with mature software tools. Second, instead of assessing the entire KB, the evaluation samples the KB through a set of queries on the RDF KB; each query result is then assessed for correctness. Third, we do not assume an initial set of KB entities with predefined identifiers. We avoid the complexity of aligning entities in the KB and reference model by using the concept of a KB *entry point* specified by an entity mention in an input document.

In the next section we discuss the general problem of KB evaluation and present a concrete proposal for

evaluating a KB constructed from text, which will be implemented at the TAC 2012 evaluation.

2 Knowledge Base Evaluation

Mayfield et al. (2008) introduced the problem of direct evaluation of an automatically populated knowledge base and identified six axes along which they might be evaluated: accuracy, usefulness, augmentation, explanation, adaptation and temporal qualification. In this paper we begin by asking the most elementary of those questions: how accurate is a given static knowledge base? Accuracy has two components, which correspond to the ideas of *recall* and *precision* in information retrieval. First, we would like to know whether all of the facts present in or implied by the underlying sources can be retrieved from the KB. Second, are there facts that are not present in or implied by the underlying sources that can nonetheless be retrieved. If all and only the implied facts can be retrieved, we can conclude that the knowledge base accurately reflects those sources. The central tenet of our evaluation approach is that the KB should be judged based on its responses to direct queries about its content. We call such queries *evaluation queries*.

In practice, it will not be possible to examine all of the possible facts that should be present in a knowledge base unless the underlying sources are extremely small. Even for relatively small KBs, a complete comparison for a moderately expressive representation language like OWL DL is a complex task (Papavassiliou et al., 2009). We believe that an approach using sampling of the space of possible queries is therefore a pragmatic necessity.

A central problem in evaluating a KB is aligning the entities in the KB with known ground truth. For example, if we had a reference ground truth KB, we could try to evaluate the created KB by aligning the nodes of the two KBs, then looking for structural differences. Aligning entities is a complex task that, in the worst case, can have exponential complexity in the number of entities involved. Our approach to avoiding this problem is to use known *entry points* into the KB that are defined by a document and an entity mention string. For example, an entry point could be defined as “the entity that is associated with the mention *Bart Simpson* in document DO14.” We

require that a set of entry points is aligned with the KB by the KB constructor. In practice this is easy if the KB is being constructed from the text that contains the entry point mentions.

Different classes of evaluation queries can assess different capabilities. For example, asking whether two entry points refer to same KB node evaluates coreference resolution (or entity linking if one of the entry points is an existing KB node). Asking facts about the KB node associated with a single entry point evaluates simple slot-filling. More complicated queries that start with one or more entry points can be used to evaluate the overall result of the extraction process involving entity linking, fact extraction, appropriate priors and inference. Note that this approach to KB evaluation is agnostic toward inference. That is, the original KB system may perform sophisticated backward chaining inference or no inference at all; the evaluation mechanism works the same either way.

3 A Specific Proposal

We present a specific proposal for KB evaluation that is both applicable to current research in KB population, and is immediately implementable. The TAC 2012 evaluation will include a *Cold Start Knowledge Base Population* (TAC KBP Web site, 2012). The idea behind this evaluation is to test the ability of systems to extract specific knowledge from text and place it into a KB. The schema for the target KB is specified *a priori*, but the KB is otherwise empty to start. Participant systems will process a document collection, extracting information about entities mentioned in the collection, adding the information to a new KB, and indicating how entry point entities mentioned in the collection correspond to nodes in the KB. In the following subsections, we outline a method for evaluating the TAC task.

3.1 Defining a KB target

We do not want to require that researchers use a particular KB technology to participate in an evaluation experiment. However, until we identify a standard way for a KB to be queried directly, we need to have a common formalism that participants can use to export the KB content to be evaluated, and a common evaluation KB target that can be used during

the evaluation by importing the submitted content. The export format should be simple and the target KB system and tools well defined and accessible to researchers.

We have selected RDF (Lassila and Swick, 1998) as the target representation for our evaluation. RDF is a simple yet flexible representation scheme with a well defined syntax and semantics, an expressive ontology layer OWL (Hitzler et al., 2009), a solid query language SPARQL (Prud’Hommeaux and Seaborne, 2008)), and a large collection of open-source and commercial tools that include KB editors, reasoners, databases and interfaces.

The standard semantics for RDF and OWL is grounded in first order logic. Its representation is based on a simple graph model where KB schema as well as instance data are stored as triples. These may seem like severe limitations – we would like to support the evaluation of KB population tasks in which facts can be tagged with certainty measures and that may have extensive provenance data. However, we can exploit RDF’s reification mechanism to annotate KB axioms, entities, and relations with the additional metadata. Using reification has its drawbacks: it can make a KB much larger than it need be and slow reasoning and querying. While these issues may be important in developing a production system intended to process large volumes of text and generate huge KBs, they are less problematic in an evaluation context where speed and scaling are not a focus. Moreover, reification offers the flexibility to add more annotation properties in the future.

3.2 Target ontology and submission format

We have developed an OWL ontology corresponding to the KB schema used in the 2011 TAC evaluations that includes classes for person, organization and place entities and properties for each with appropriate domain and range restrictions. For testing, we created a sample corpus of articles about the fictional world of the Simpsons television series and a corresponding reference KB of entities and relationships extracted from it.

Figure 1 shows a portion of one of our test documents, some information representing our test RDF KB about one of the entities (:e12), and one of the annotations that indicates that the mention string “Montgomery Burns” in document D011 was linked

```

<DOC source="...">(DOCNO)D011</DOCNO>(TEXT)
The Springfield Nuclear Power Plant is a nuclear power plant
in Springfield owned by Montgomery Burns. The plant has the
monopoly on the city of Springfield’s energy supply, and the
carelessness of Mr. Burns and the plant’s employees (like Homer,
who is employed at Sector 7G) ... </TEXT></DOC>

:e12 a kbp:PER; kbp:canonical_mention "Montgomery Burns";
kbp:mention "Burns"; kbp:title "Mr.".

[ a rdf:Statement; rdf:Subject :e12; rdf:Object "Montgomery
Burns"; rdf:Predicate kbp:canonical_mention; rdf:source doc:D011;
kbp:probability 1.0].

```

Figure 1: A sample document and some KB assertions it generates in RDF using the *turtle* serialization.

with :e12 with certainty 1.0. The export format for a participant’s KB is kept simple; it consists of a file of tab-separated lines where each line specifies a relation tuple with optional evidence (e.g., a source document ID) and certainty factor values. For example, if a KB links the entity with mention “Montgomery Burns” in document D011 to an instance with local ID :e12 and also determines from document D14 that the entity’s age was 104 (with certainty .85), it would export the following two five-tuples.

```

:e12 mention "Montgomery Burns" D011 1.0
:e12 age "104" D014 0.9

```

To simplify the evaluation and avoid potential problems, we restrict the inferencing performed on the submitted KB after its conversion to RDF to a few simple patterns, such as a subset of RDFS entailments (ter Horst, 2005) that follow from the target ontology (e.g., inferring that every *canonical_mention* relation is also a *mention* relation).

3.3 Query-based Knowledge Base Evaluation

We have defined a simplified graph path notation for evaluation queries to make constructing them easier; this notation is then automatically compiled into corresponding SPARQL queries. For example, one pattern starts with an entry point (a mention in a document) and continues with a sequence of properties. The general form of such a path expression is $MDP_1\dots P_n$ where M is a mention string, D is a document identifier, and each P_i is a property from the target ontology. All of the properties in the path except the final one must go from entities to entities. The final one can have a range that is either an entity or a string. For example, to generate a query for “The ages of the siblings of the entity mentioned as

```

SELECT ?CN ?SIBDOC ?A ?ADOC WHERE {
  ?P kbp:mention "Bart Simpson".
  ?P kbp:sibling ?SIB.
  ?SIB kbp:canonical_mention ?CN;
  kbp:age ?A.
  ..x rdf:subject ?P; rdf:predicate kbp:mention; rdf:object "Bart
  Simpson"; kbp:source doc:D12.
  ..x rdf:subject ?P; rdf:predicate kbp:sibling; rdf:object ?SIB;
  kbp:source doc:SIBDOC.
  ..x rdf:subject ?SIB; rdf:predicate kbp:canonical_mention;
  rdf:object ?CN; kbp:source doc:SIBDOC.
  ..x rdf:subject ?SIB; rdf:predicate kbp:age; rdf:object ?A;
  kbp:source doc:ADOC.}

```

Figure 2: This SPARQL query generates data that an assessor can use to evaluate the KB.

“*Bart Simpson*” in document *D012*” we use the path expression `"Bart Simpson" D012 sibling age`.

The SPARQL query generated for this path expression is shown in Figure 2; when run against a submitted KB, it produces data that will allow the assessor to verify that the KB accurately reflects the supported facts:

sibling mention	sib doc	age	age doc
“Lisa Simpson”	D012	“10”	D008
“Maggie Simpson”	D014	“1”	D014

In general, for each entity in the result, a query produces the canonical mention string for that entity in the supporting document (e.g., support for “Lisa Simpson” as Bart’s sister is in *D012*), while for each slot value (e.g., `age:10`), the query produces the value (10) and the document that provided evidence for that value (*D008*). This lets an assessor verify that the correct entities are identified and that there is explicit support for the slot values.

3.4 Metrics

Once SPARQL queries have been designed and run against the knowledge base, the results need to be assessed and scored. Doing so is relatively straightforward; there is a rich history of approaches to assessment and evaluation metrics for similar output that have been widely applied. Two obvious choices are to use binary queries or to use queries that return slot fills. Binary queries such as “is a parent of the ‘Bart’ mentioned in document *D014* the same as a spouse of the ‘Homer’ mentioned in document *D223*?” are easy to assess, and can be scored using a single number for accuracy. Queries that return one or more string values for attributes of an

entity look very much like slot filling queries. TAC is the latest in a long line of evaluations that have scored slot fills. The standard approach is to view the possible fills as a set, and to calculate precision, recall and F-measure on that set. These numbers are widely understood and intuitively satisfying. For TAC 2012, we will cleave as closely as possible to the measures being used to evaluate the TAC slot-filling task. More details on the assessment and scoring process can be found in the Cold Start 2012 task description (TAC KBP Web site, 2012).

3.5 Errors in the Knowledge Base

One issue with our sampling approach to KB evaluation is ensuring that the collection of sample queries has adequate coverage in at least three dimensions: over a set of error types; over the full range of entities types and their properties; and over the extent of the corpus. Different kinds of errors in the KB will be detected by different sorts of queries. For example, for the TAC KBP task, we have identified the following types of errors:

- Two distinct ground truth entities are conflated.
- A ground truth entity is split into several entities.
- A ground truth entity is missing from the KB.
- A spurious entity is present in the KB.
- A ground truth relation is omitted from the KB.
- A spurious relation is present in the KB.
- An entry point is tied to the wrong KB node.

Some queries can be designed to narrowly target specific error types while others may detect that one or more errors are present but not identify which are the actual culprits. Similarly, attention should be paid to providing queries that test a range of entity types and properties as well as data from documents that represent different genres, sizes, languages, etc.

4 Discussion

The evaluation most similar to our proposal is the one used in the DARPA Machine Reading program (Strassel et al., 2010). In this evaluation, a small document collection of order 10^2 documents is exhaustively annotated to produce a gold standard KB. A submitted KB is evaluated by querying it to produce *all* relations of a given type. While this approach gives excellent insight into a system’s operation over the annotated collection, it suffers from requiring a gold standard knowledge base; this both

limits the evaluation’s ability to scale to larger collections, and raises the issue of how a submitted KB is to be aligned to the gold standard once the collection size is successfully increased.

Sil and Yates (2011) propose an extrinsic evaluation framework that measures the quality of an automatically extracted KB by how much it improves a relation extraction system. While our proposal represents an intrinsic evaluation, it can be easily tailored to a given downstream task by selecting evaluation queries that are directly relevant to that task.

The success of a query-based evaluation approach depends on having an appropriate set of KB queries. They must have good coverage along several dimensions: testing all important information extraction aspects (e.g., entity linking, slot filling, provenance, etc.); fairly sampling the full range of slots; testing for both for both missing and extraneous (false) facts; using a representative set of entry point documents; and anticipating and testing for known or expected system failure modes (e.g., over-merging vs. under-merging entities). Since the queries will not be overly complex, parts of the KB that are not “close to” entry points may not be tested. Our simple path-based scheme for representing queries that are automatically compiled into executable SPARQL queries will probably need to be made more complex for future systems.

Our KB model is quite simple; extending it to evaluate more capable knowledge-base technologies will offer challenges. For example, while we admit certainty values for slot values, we have not yet defined that these actually mean, how they they are handled in queries or how to evaluate them. A simple scheme can also produce ambiguity. For example, if the KB has two slot fills for Homer’s children (Bart and Lisa with certainties 0.4 and 0.3) a proper evaluation will also need to also know if the original KB treats these as alternatives or as possible independent values.

Many challenging issues will be raised if we evaluate KBs that represent and exploit indefinite knowledge, which might take the form of Skolem functions, disjunctions or constraints. For example, our ontology may stipulate that every person has exactly one mother and we may read that Patty Bouvier is Bart’s mother’s sister. But if we know that Patty has two sisters, Marge and Selma, we not know which

is Bart’s mother but can still identify Bart as Patty’s nephew. Knowing that every person has exactly one age (a number), a valid answer to “what are the ages of Homer Simpson’s children”, might be “Bart’s age is 10, Lisa’s is 8, and Maggie’s age is unknown.” This response reveals that the KB knows Homer’s has three children even though the age of one has not been populated. A final variation is that a system may not have determined an exact value for a property, but has narrowed its range: reading that Lisa is “too young to vote” in the 2012 U.S. election implies that her age is less than 18.

Future information extraction systems will support many practical features that will need evaluation. Evaluating KBs in which some facts are temporally qualified will add complexity. Our model of provenance is simple and may need to be significantly extended to evaluate systems that represent evidence in a more sophisticated manner, e.g., noting how many documents support a fact and capturing alternative facts that were rejected.

5 Conclusions

While evaluating the quality of an automatically generated knowledge base is an open ended problem, the narrower task of evaluating the results of a knowledge base population task is much easier. This was especially true for the entity linking and slot filling focus of the past TAC KBP tasks, since an initial KB was provided that included not only a schema, but also a fairly complete set of initial entities. This obviated the need for aligning entities between a submitted KB and a reference KB, a major source of evaluation complexity.

Evaluating submissions to the 2012 TAC Cold Start KBP task will be more difficult since the task starts with just a KB schema and no initial entities. We described a general approach to KB evaluation that uses the notion of *KB entry points* specified by mentions in documents to avoid having to align entities between the KB under evaluation and a reference KB. The evaluation can then be done by executing a set of KB queries that sample the results of a submitted KB and generate data to allow a human assessor to evaluate its quality.

References

- Hans Chalupsky. 2012. Story-level inference and gap filling to improve machine reading. In *The 25th International FLAIRS Conference*, May.
- G. Doddington, A. Mitchell, M. Przybocki, L. Ramshaw, S. Strassel, and R. Weischedel. 2004. The automatic content extraction (ACE) program—tasks, data, and evaluation. In *Proceedings of LREC*, volume 4, pages 837–840. Citeseer.
- Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S. Weld. 2008. Open information extraction from the Web. *Communications of the ACM*, 51(12):68–74, December.
- P. Hitzler, M. Krötzsch, B. Parsia, P.F. Patel-Schneider, and S. Rudolph. 2009. OWL 2 web ontology language primer. Technical report.
- H. Ji, R. Grishman, H.T. Dang, K. Griffitt, and J. Ellis. 2010. Overview of the TAC 2010 knowledge base population track. In *Proceedings of the Third Text Analysis Conference (TAC 2010)*.
- O. Lassila and R.R. Swick. 1998. Resource description framework (RDF) model and syntax specification. Technical report.
- James Mayfield, Bonnie J. Dorr, Tim Finin, Douglas W. Oard, and Christine D. Piatko. 2008. Knowledge base evaluation for semantic knowledge discovery. In *Symposium on Semantic Knowledge Discovery, Organization and Use*.
- Vicky Papavassiliou, Giorgos Flouris, Irimi Fundulaki, Dimitris Kotzinos, and Vassilis Christophides. 2009. On detecting high-level changes in RDF/S KBs. In *International Semantic Web Conference*, pages 473–488.
- E Prud’Hommeaux and A. Seaborne. 2008. SPARQL query language for RDF. Technical report, January.
- A. Sil, A. Yates, B. St, and M. Ave. 2011. Machine reading between the lines: A simple evaluation framework for extracted knowledge bases. *Proceedings of the Workshop on Information Extraction and Knowledge Acquisition*, pages 37–40, September.
- S. Strassel, D. Adams, H. Goldberg, J. Herr, R. Keesing, D. Oblinger, H. Simpson, R. Schrag, and J. Wright. 2010. The DARPA machine reading program—encouraging linguistic and reasoning research with a series of reading tasks. In *International Conference on Language Resources and Evaluation*, May.
- TAC KBP Web site. 2012. Cold start knowledge base population at TAC 2012 task description. <http://www.nist.gov/tac/2012/KBP/ColdStart/>. National Institute of Standards and Technology.
- Herman J. ter Horst. 2005. Completeness, decidability and complexity of entailment for RDF Schema and a semantic extension involving the OWL vocabulary. *Journal of Web Semantics*, 3(23):79 – 115.
- Dimitris Zeginis, Yannis Tzitzikas, and Vassilis Christophides. 2011. On computing deltas of RDF/S knowledge bases. *ACM Trans. Web*, 5(3):14:1–14:36, July.