A Dataset for Web-scale Knowledge Base Population

Michael Glass and Alfio Gliozzo

Knowledge Induction and Reasoning Group, IBM Research AI

Abstract. For many domains, structured knowledge is in short supply, while unstructured text is plentiful. Knowledge Base Population (KBP) is the task of building or extending a knowledge base from text, and systems for KBP have grown in capability and scope. However, existing datasets for KBP are all limited by multiple issues: small in size, not open or accessible, only capable of benchmarking a fraction of the KBP process, or only suitable for extracting knowledge from title-oriented documents (documents that describe a particular entity, such as Wikipedia pages). We introduce and release CC-DBP, a web-scale dataset for training and benchmarking KBP systems. The dataset is based on Common Crawl as the corpus and DBpedia as the target knowledge base. Critically, by releasing the tools to build the dataset, we enable the dataset to remain current as new crawls and DBpedia dumps are released. Also, the modularity of the released tool set resolves a crucial tension between the ease that a dataset can be used for a particular subtask in KBP and the number of different subtasks it can be used to train or benchmark.

1 Introduction

Populating knowledge bases from text is an important element in resolving the knowledge acquisition bottleneck. However, a shortage of open, large scale and broadly scoped datasets has limited development and comparison of approaches for this task. This paper explains the creation and use of a new resource for training and benchmarking knowledge base population systems. The dataset can be used as training data for a system to expand the DBpedia ontology from text or used as a benchmark for developing new approaches for extending knowledge bases in general.

In section 2 we define the task of Knowledge Base Population (KBP). In section 3 we consider existing datasets for this task, and examine the differences between these datasets and the new dataset created in this work. Section 4 explains how the dataset was created, and what variations of the dataset can be created using the modular tool set we distribute. Finally, in section 5 we provide some statistics on this dataset to provide a picture of its properties.

2 Knowledge Base Population

We define Knowledge Base Population (KBP) as extending a knowledge base with information extracted from text. In this view, there is already a substantial amount of knowledge. The goal may be to update it, keeping it current with new information. Also, the knowledge base is likely incomplete, even with respect to just the current

knowledge. Fortunately, a large volume of the missing knowledge is in unstructured text. We do not assume the existence of any annotated text, nor of any process for writing rules. Although the task of KBP does not prohibit those approaches.

To be clear, it is not our true goal to label mentions of relations. KBP is not typically focused on using populating a knowledge base as a task to help train a relation extractor. KBP is fundamentally not an attempt to label any particular sentence with anything at all, neither in training nor when applying the model.

What we call KBP is also referred to by other terms in the literature. Automatic Knowledge Base Construction (AKBC) is often used to refer to this task, although often it is the case that there is no existing knowledge base in AKBC. Slot filling is a version of KBP where specific missing information for certain query entities is desired. Knowledge base validation or refinement is a related task where evidence for or against triples in the knowledge base is gathered from text and used to estimate a confidence for existing triples.

It is important to clarify some core concepts for this type of task. The knowledge in the knowledge base is in the form of *triples*. Triples connect two entities - or more generally *nodes* - by a *relation*. A node may be an entity (such as a person, place, creative work, etc) or a literal (such as a date, number, or string), or a type (such as Person, Sporting Event, etc). The relation for each triple is drawn from a fixed set of relations. Example relations include: is-a, born-on, member-of, and instance-of.

In KBP, the new knowledge added is in the form of triples with confidence. The nodes in the added triples could be existing nodes in the knowledge base or totally new entities or literals. The relation in the added triple is drawn from the fixed set of relations. Note that attaching a probability to a triple is not only motivated by the relation prediction's imperfections in its ability to interpret the textual evidence. The textual evidence itself is often inconclusive. In this view of KBP, the relation prediction should provide a probability well above zero for ⟨ Samuel Smith born-in New York ⟩ from the sentence "Samuel Smith went to school in New York".

2.1 Knowledge Graph View

We can also view the knowledge in a knowledge base as a knowledge graph. In this view, the result of a KBP system is to add both nodes and edges to our knowledge graph. The edges are labeled with the name of the relation, as well as the confidence from the relation prediction, a probability. Figure 1 shows a diagram of a knowledge graph with nodes and edges. New edges and nodes created from textual evidence are displayed with a dotted line.

Formally, there is a set of relation types R. There is a multigraph G with nodes N and edges E. The multigraph is directed and edge labeled. Each directed edge is labeled with a relation type from R and confidence in the range $[0, 1]$. We extend the knowledge graph by adding to N and E, while R remains fixed.

2.2 Subtasks of Knowledge Base Population

The process of KBP can be divided into a number of subtasks. We do not suggest that these are necessarily trained independently; there could be a single end-to-end deep

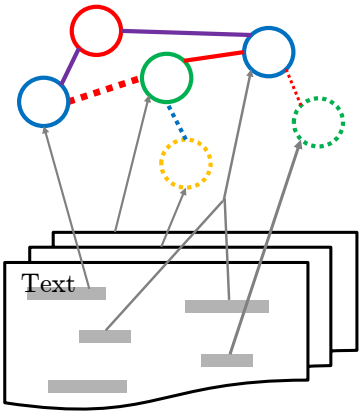

Fig. 1. Knowledge Graph from Text

architecture to perform all of these subtasks. Nor do we suggest that the decisions from earlier subtasks cannot be adjusted by information present in later subtasks. Rather, these are logically distinct steps. And, crucially for the introduction of a new dataset, these are subtasks that are carved along the boundaries of existing research programs.

These subtasks include Entity Detection and Linking (EDL), Context Set Construction (CSC), relation prediction and reasoning. Figure 2 illustrates the interaction of these components.

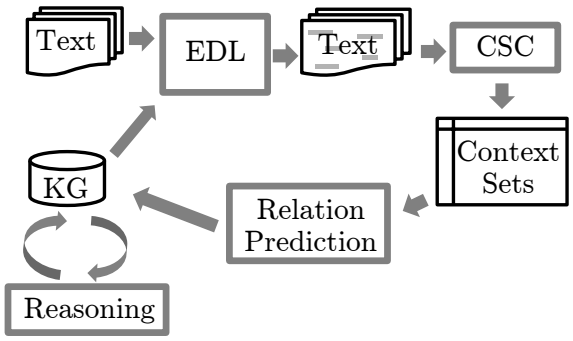

Fig. 2. Subtasks of Knowledge Base Population

The subtask of Entity Detection and Linking (EDL) itself can be viewed as up to three subtasks: entity detection, co-reference and entity linking. A system for EDL finds mentions of the nodes that are, or should be, in the knowledge base. A mention is a span in text that refers to a specific node. Although we refer to the task as *Entity* Detection and Linking, we are concerned with nodes in general. For example, the term 'quarterly' is not an entity. But there are knowledge bases that contain information about magazines

and journals including the frequency that they are published. So an appropriate EDL system would locate mentions of terms like 'quarterly' in text, as well as mentions of the magazines and journals. These mentions are also linked. In the case of nodes that are in the knowledge base, their mentions (which may be pro-nominal mentions) are linked to the node. For mentions of nodes that should be, but are not, in the knowledge graph, a new node is first created and then the mentions of it are all linked to that new node. This could also be seen as clustering mentions for nodes not present in the knowledge base.

In figure 3 we see three node mentions: 'The Pub Corp', 'magazine' and 'every month'. 'The Pub Corp' is a mention of a new entity, while 'magazine' is a mention of a type node that matches the label exactly, and 'every month' is a mention of a literal node that does not exactly match the label of the node.

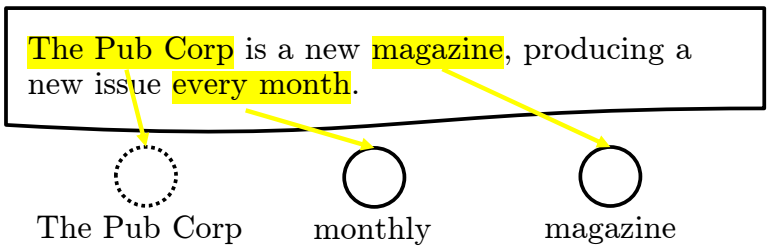

Fig. 3. Example of EDL

Context Set Construction (CSC) is the next step. A CSC system is responsible for gathering textual evidence for the relation prediction phase by identifying contexts in documents where two node mentions occur together. For a simple case, consider contexts defined as two node mentions in the same sentence. We refer to a context identified by the CSC system as a node-pair context. The CSC system is also responsible for grouping the contexts by node-pair to gather a context set for each node-pair. A node-pair context set is the direct textual evidence for the relationship (if any) between two nodes.

From this textual evidence KBP proceeds to the subtask of relation prediction. Training on sets of contexts for node-pairs is often called distantly supervised relation extraction [9]. Although, that term is also sometimes used to describe noisy supervision created by labeling each context for a node-pair by the set of relations that hold between them. Additional textual evidence may also be used to predict relations, such as the contexts of each node independently. We exclude from this logical subtask the use of other triples from the knowledge base as evidence.

The final subtask, which we call reasoning, encompasses any use of structured knowledge to predict, or adjust the confidence of, new triples. So Knowledge Base Completion (KBC) approaches based on tensor factorization such as TransE [3] qualify, as well as approaches based on induction and application of probabilistic logic [14].

2.3 Variant Definitions

A typical variation of KBP expects explicit provenance for each predicted triple. In this case the goal is still to populate a knowledge base, but with the constraint that each new triple must point to a specific sentence or passage that justifies it. Unlike our formulation of KBP, this version sharply limits the role of reasoning and extraction of implicit evidence. There is more knowledge that can be gathered from text and an existing knowledge base than just what is explicitly stated in text. So the task of knowledge base population as we have defined it, is a broader view. The KBP system can use explicitly stated relation mentions, but is also permitted to look at what is implicit in text, and perform reasoning or do knowledge base completion to probabilistically conclude new triples from existing triples. And all of these sources can be combined, using new information from text as well as existing knowledge to draw conclusions.

2.4 Evaluation

A key challenge of knowledge base population research is the method of evaluation. Three types of evaluation are commonly applied: Held-Out, Exhaustive Annotation, and Pooled Response. The first two types, Held-Out and Exhaustive Annotation are automatic evaluations, while Pooled Response requires manual judgements for any new triples produced by a system.

In a held-out evaluation, some triples of the knowledge base are removed from the training for the KBP system and must be predicted. Triples predicted by the system that match this held-out set are correct, while triples that do not match are wrong. One important advantage of this type of evaluation is that it can be applied whenever a substantial knowledge base exists, without an additional large manual effort. In exhaustive annotation, all triples that are stated in the corpus are annotated, though not necessarily with a specific location in text. Pooled response is a type of retrospective evaluation, where the ground truth is built in response to the outputs of a set of systems. The triples predicted by all systems are pooled and each distinct triple is manually judged. Table 4 summarizes the way each approach to evaluation tabulates the true positives, false positives and false negatives.

Method	True Positives	False Positives	False Negatives
Held-Out	System triples in KB	System triples *not* in KB	KB triples not in system triples
Exhaustive Annotation	System triples in annotation	System triples *not* in annotation	Annotated triples not in system triples
Pooled Response	System triples verified to be true	System triples verified to be false	Triples verified to be true, produced by a different system

Fig. 4. Evaluation Style Comparison

We can see that in all evaluation approaches there is a difficulty with the tabulation of false negatives. Even in 'exhaustive' annotation, the recall basis is formed by what

is explicitly stated in text - limiting both the relation prediction and especially the reasoning. However, for a benchmark used to compare different approaches, the absolute recall, or area under the precision/recall curve is not needed. Judging the relative value is enough. Figure 5 shows that the determining the number of false negatives not produced by either system is not needed to compare the P/R curves of two approaches. Whether we choose a recall basis of a, b, or c the relative area of the two curves is unchanged.

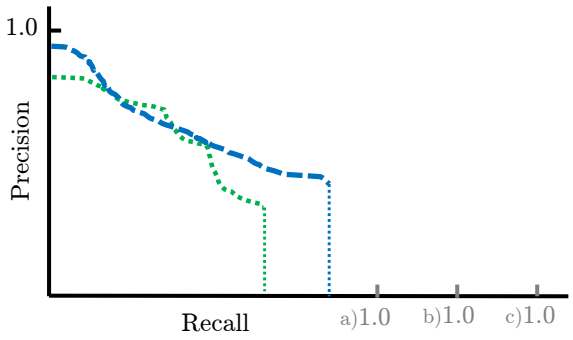

Fig. 5. Irrelevance of Shared False Negatives for Relative Precision / Recall Curve

3 Related Datasets

With the background information established, and the task clearly defined, we can catalog the existing datasets that are suitable for some or all of the subtasks in KBP. The most relevant existing datasets are: NYT-FB, TAC-KBP, FB15K-237, WikiReading and T-REx.

NYT-FB [10] is a dataset based on a subset of Freebase [2] relations: 56 relations in total, but some with a negligible amount of training and test data. NYT-FB was built using an NER system and a simple entity linker to locate the node-pair-contexts in the New York Times news corpus [11]. The unit of context used is a sentence. The data is prepared to the point of providing the node-pair context sets and also providing extracted features, such as dependency path information, on each of the contexts (sentences). So NYT-FB is immediately suitable for benchmarking the relation prediction subtask. However, the ease of use for relation prediction comes at the price of flexibility in benchmarking variations on EDL or CSC. The unit of context is fixed at a single sentence, and the EDL system is fixed as well, with no co-reference. The dataset can be combined with the rest of Freebase to benchmark approaches to combined relation prediction and reasoning.

There are a number of datasets from different TAC-KBP[1] competitions. TAC-KBP is organized by NIST and covers many parts of the knowledge base population task:

[1] https://tac.nist.gov/

EDL, CSC and relation prediction. All of these subtasks can be addressed with varied approaches to determine how they impact the final score. Unfortunately, a few serious practical issues limit its effectiveness. TAC-KBP is not an automatic evaluation. Many researchers have, regardless, used the pooled evaluation judgements from past TAC-KBP as a ground truth for an automatic evaluation.This is known to introduce a serious bias, favoring systems that were used to construct the pooled evaluation [4]. Furthermore, the TAC-KBP datasets are subject to considerable constraints. The data must be specifically requested and is not available under any form of open source license.

KBP Online[2] is working to adapt TAC-KBP data to an open evaluation and to accelerate the scoring based on pooled evaluation with crowd sourcing. But even with crowd sourcing, it is not a benchmark that can support dozens of rapid experiments. The task itself is also slightly different. In TAC-KBP slot filling, the fillers are not new or existing nodes, instead they are regarded simply as strings. Provenance - a justifying sentence - must be provided. This constraint naturally limits the types of approaches that can be applied. Information from multiple sentences can not be combined through reasoning.

FB15K-237 [12] is an extension of the FB15K dataset [3] with additional textual evidence gathered from ClueWeb12[3]. The provided textual evidence is the dependency parse path from one Freebase entity to another. The dataset has been used to combine textual relations with the structured relations when doing knowledge base completion.

Of the datasets we list, FB15K-237 has the most heavily processed text data. The textual mention files have lines with two entities connected by a lexicalized dependency path, along with a count of the number of times such a path connected the two entities. So it is very limited in what approaches can be explored even for relation prediction. Figure 6 shows an example of a lexicalized dependency path.

$$X :\leftarrow nn : fact :\leftarrow pobj : in :\leftarrow prep : game :\leftarrow nsubj :' s : ccomp \rightarrow: pivot : nsubj \rightarrow: Y$$

Fig. 6. Lexicalized Dependency Path from FB15K-237

WikiReading [7] is also suitable for some kinds of knowledge base population research. In this dataset, the knowledge base of WikiData [13] is aligned with English Wikipedia[4]. The DBpedia URI for WikiData resources is mapped to its Wikipedia page, for those WikiData resources that have them. The task is to predict the slot fillers for the WikiData node using the text of the Wikipedia page. Unlike most other datasets, this dataset relies on what are sometimes called title-oriented-documents (TODs). A TOD is a document that describes a single node in a knowledge graph. TODs are not a very typical format for documents, although they are also not rare, Wikipedia is one example, but there are others as well such as Mayo Clinic[5] articles in the medical domain. The focus on TODs limits the EDL to finding nodes that are slot fillers. Like FB15K-237, the evaluation is at the knowledge base level so approaches to relation predictions

[2] https://kbpo.stanford.edu/
[3] http://lemurproject.org/clueweb12/FACC1/
[4] https://en.wikipedia.org
[5] https://www.mayoclinic.org

based on implicit textual evidence, as well as or reasoning, can be evaluated. The data is distributed in a custom JSON format or TensorFlow data format.

A new dataset called T-REx[6] has also been created. This dataset aligns DBpedia triples with Wikipedia text. The EDL system employs DBpedia Spotlight [5] and co-reference. The CSC system is somewhat sophisticated, using paragraph and sentence boundaries. However, only cases where two entities occur together, when those two entities have a relation are annotated. So it is suitable for relation classification, but not relation prediction, since there are no unrelated node-pair contexts. Like our approach, T-REx supplies the code used to create the dataset, so variations are possible, but it is not designed as a modular benchmark where different components can be cleanly substituted.

Across several of these datasets we see a tradeoff between data that is immediately useable for at least one subtask and data that is flexible enough to allow for different approaches to many subtasks. By providing baseline - but replaceable - implementations for subtasks in the early pipeline, we can meet the goal of a dataset that is immediately useable while also flexible enough to explore the full scope of the KBP task. The tools allow for the creation of an immediately usable relation prediction dataset, but also allow for the creation of very different versions.

The size of the dataset, and the diversity of relations present are also important considerations. Table 7 compares the size and availability of the different datasets to CC-DBP.

Dataset	Corpus	Relation Types	Node-Pair Contexts	Available
NYT-FB	1.8M sent.	56	157K	partially
TAC-KBP	90K sent.	41	122K	closed
FB15K-237	2.7M dep.-paths	237	2.7M	public
WikiReading	4.7M articles	884	N/A	public
T-REx	6.2M sent.	642	11M	public
CC-DBP	173M sent.	298	3M	public

Fig. 7. Dataset Size Comparison

4 Dataset Creation

The CC-DBP[7] dataset creation process begins by filtering both Common Crawl and DBpedia to a useful subset. In the case of DBpedia, the useful subset is the set of triples that can be supported with textual evidence. The useful subset of Common Crawl is the text in the same language as the version of DBpedia used, in this case English. Baseline systems for EDL and CSC are also provided to enable the dataset to be immediately useable for relation prediction or combined relation prediction and reasoning. Figure 8 shows the pipeline to produce the base dataset as well as more processed forms.

[6] https://hadyelsahar.github.io/t-rex/
[7] https://github.com/IBM/cc-dbp

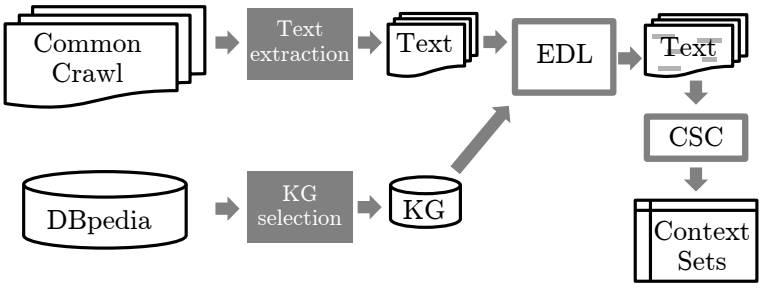

Fig. 8. CC-DBP Construction

4.1 Common Crawl

Common Crawl[8] is a large, publicly available crawl of web pages. It is updated on a monthly basis to remain current and includes new seed URLs to expand the scope and size.

The dataset is available in Web ARChive (WARC) format, which stores the raw responses from the websites crawled. Although it is also available in WET format, providing the HTML stripped text, we use the WARC format so that we can apply Boilerpipe to discard the boilerplate text from web pages. In it's raw form it contains over three billion web pages and hundreds of terabytes of content (uncompressed).

In order to find the English content text that will be used as textual evidence for training and benchmarking KBP systems we execute the following pipeline:

- Obtain the list of WARC paths from the Common Crawl website
- Download the WARC files hosted on Amazon S3
- Filter the WARC entries to text or HTML response pages
- Remove the boilerplate, such as navigation menus and headers, extracting only the main textual content
- Remove web pages not in the desired language, selected to match the version of DBpedia used

The initial steps are straightforward. Filtering the boilerplate and removing the HTML formatting to retain only the text content is more challenging. We use the open source library Boilerpipe [8], configured with the KeepEverythingWithMinKWordsExtractor filter. This removes any text-blocks that have fewer than k (configured as 5) words. By inspection, this was effective in selecting the main textual content. Filtering the web pages, now text documents, by language was accomplished through another open source library: the Optimaize language-detector library[9]. This library supports 71 languages and detects the language by comparing the profile of N-grams to a reference profile.

The result of this processing on the June 2017 crawl was 4.6 million documents with 173 million sentences and 1.5 billion words.

[8] http://commoncrawl.org
[9] https://github.com/optimaize/language-detector

4.2 DBpedia

DBpedia [1] is a mature knowledge base built from the infoboxes of Wikipedia. It is publicly available under a Creative Commons Attribution-ShareAlike 3.0 License as well as the GNU Free Documentation License.

We first downloaded the main portions of the most recent DBpedia English knowledge base (http://wiki.dbpedia.org/downloads-2016-10 as of this writing). These files are the mappingbased_objects_en.ttl.bz2, mappingbased_literals_en.ttl.bz2, labels_en.ttl.bz2, and dbpedia_2016-04.owl. The "mappingbased" files are the higher quality (but lower quantity) versions of triples involving two DBpedia resources, or a DBpedia resource and a literal. We collapse the distinction between resources and literals, considering both simply as nodes. The labels file provides, for each resource, a term or terms that can be used to refer to it in text. The dbpedia_2016-04.owl file provides the type and relation taxonomy.

To avoid having relations that have very few positive examples in either train or test, we limit our focus to the relations with a large number of triples. The relations were further filtered with the goal of finding relations where both arguments could be matched in text. We examined the most frequent 400 relations. From these we exclude relations based on the following criteria:

- We exclude all relations that connect a resource to a textual label for that resource. These are important for EDL, but are out of scope for relation prediction. Examples of these relations are: foaf:name, dbo:synonym or dbo:birthName.
- We excluded a relation (dbo:filename) where the filler did not have a textual label, instead referencing the URL for a file.
- We removed relations where the filler was often a long phrase or free-form text. Examples include dbo:description, dbo:identificationSymbol and dbo:otherServingLines.
- Relations were filtered when they have numeric fillers that are unlikely to be found in text exactly, especially floating point numbers or values where multiple units of measure may apply. This was a substantial category, with relations such as dbo:militaryUnitSize, dbo:topSpeed, dbo:mouthElevation and dbo:numberSold.
- Relations with date slot fillers were converted to years or discarded if there was already another relation for the year value. This was done to avoid the complexity of date detection, normalization and matching.
- We also removed all mediator relations. These are relations that connect to a blank node, often used to represent n-ary relations. These included dbo:careerStation, where the "career station" is a blank node used to represent a time period in a person's career, as well as several others.

The result of applying these filters is 298 relations, not including super-relations from the DBpedia relation taxonomy.

To address some of the mostly spurious ambiguity in EDL, we collapse nodes with only a case difference (or no difference) in preferred labels into one, where the preferred label for a literal is the string representation of the literal.

4.3 Baseline EDL

For a baseline Entity Detection and Linking (EDL) system we build a gazetteer from the DBpedia resource labels, and for literals the string representations. In cases where the same label could link to multiple resources we connected to only the more popular resource. To measure popularity our baseline EDL simply counts the number of triples in which the node appears.

To avoid annotating the corpus with mentions of extremely generic nodes, we filter out labels that occur more than a threshold frequency (300000) unless they are multi-words or numbers. This removes mappings for low value nodes such as "dbl:had" and "dbl:he".

The resulting gazetteer is used in a simple string matching annotator, serving as the baseline EDL.

4.4 Baseline CSC

For the baseline Context Set Construction (CSC) we use a sentence as the unit of context. Sentence detection was performed by the open source OpenNLP library [10]. In addition, we excluded node-pairs from CSC based on a type filter. To enable simple type based filtering, we constructed a shallow, coarse grained type system of 35 types. The type filter ignores node-pairs where there is no triple in the training set that has the same (unordered) pair of types.

5 Dataset Statistics

In this section we present some statistics on the relations and nodes present in the dataset.

Figure 9 shows the distribution of entity frequency in the corpus. The most common number of times for an entity to be mentioned is 1, with over 200,000 entities having this count. While there are about 1000 entities with over a million mentions. Both axes are log scale and each bar groups together entities whose frequency is closest to the indicated value. This can be seen as a probability mass function plot in log-log space, with the near-linear relationship indicating a typical Zipfian distribution.

Since a node-pair can be related by multiple relations, this is a multi-label task. Figure 10 shows how much the task is multi-label. Considering only leaf relation types in the relation hierarchy, we see that the most frequent number of relations between a node-pair is one, with over five hundred thousand entity-pairs having a single relation type label. Fewer than one hundred node-pairs have six different labels. Highly related pairs are typically a person and a creative work, such as Peter Jackson and the movie *Bad Taste* related by: cinematography, director, editing, producer, starring, and writer. Figure 10 shows only the positive node-pairs. But by far the most common number of relation labels for a node-pair is zero. The negatives are typically downsampled when training relation prediction.

[10] https://opennlp.apache.org/

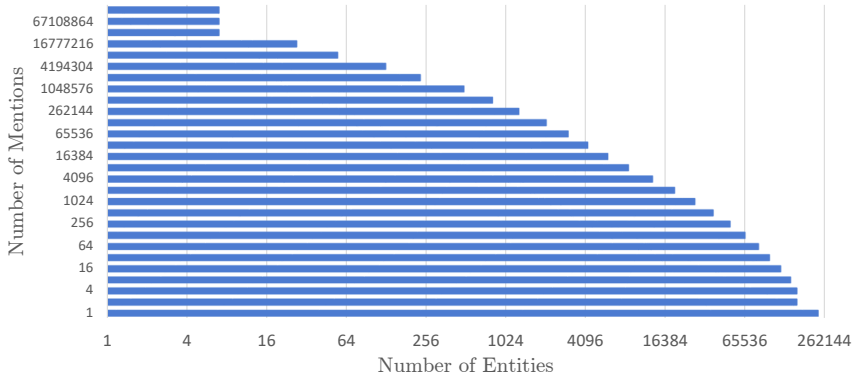

Fig. 9. Entity Mention Count Distribution

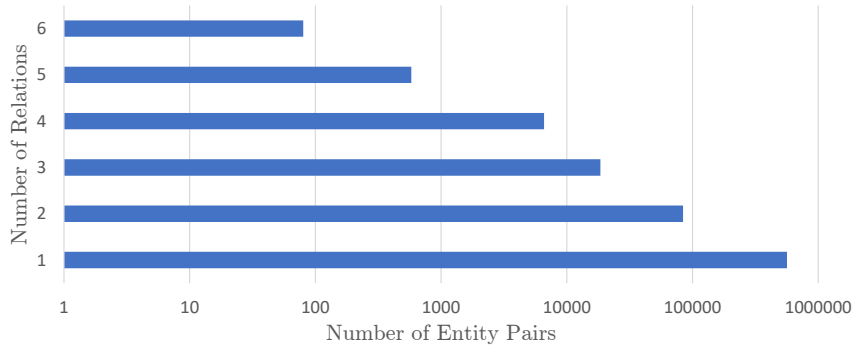

Fig. 10. Number of Relations for an Entity Pair

For the distinct entities mentioned in the corpus, figure 11 shows the distribution of different node types. Most of the nodes are entities of type person, place or organization with significant numbers of creative works, as well as string and number literals, and plants and animals. For comparison, the NYT-FB dataset uses a Named Entity Recognition (NER) system to locate mentions of entities so almost all entities are people, places or organizations.

Considering a sentence as the unit of context, figure 12 shows the distribution of the number of contexts for each node-pair. Again we see a Zipfian distribution, with the most frequent number of sentences for a node-pair to share being only one. For CC-DBP, 63% of node-pairs have only a single shared context, while in NYT-FB, 81% of entity-pairs have only a single context.

Figure 13 shows what relations hold between the entity-pairs that co-occur (when a relation does hold). Perhaps most striking is the diversity of relations that have a substantial fraction of instances. The most frequent 47 relation types make up 80% of the positives. In contrast just 5 relation types make up 83% of the positives for NYT-FB.

Fig. 11. Entity Type Distribution

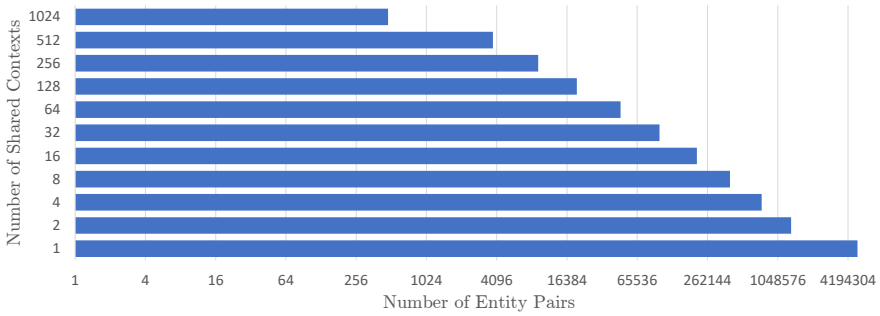

Fig. 12. Number of Contexts for an Entity Pair

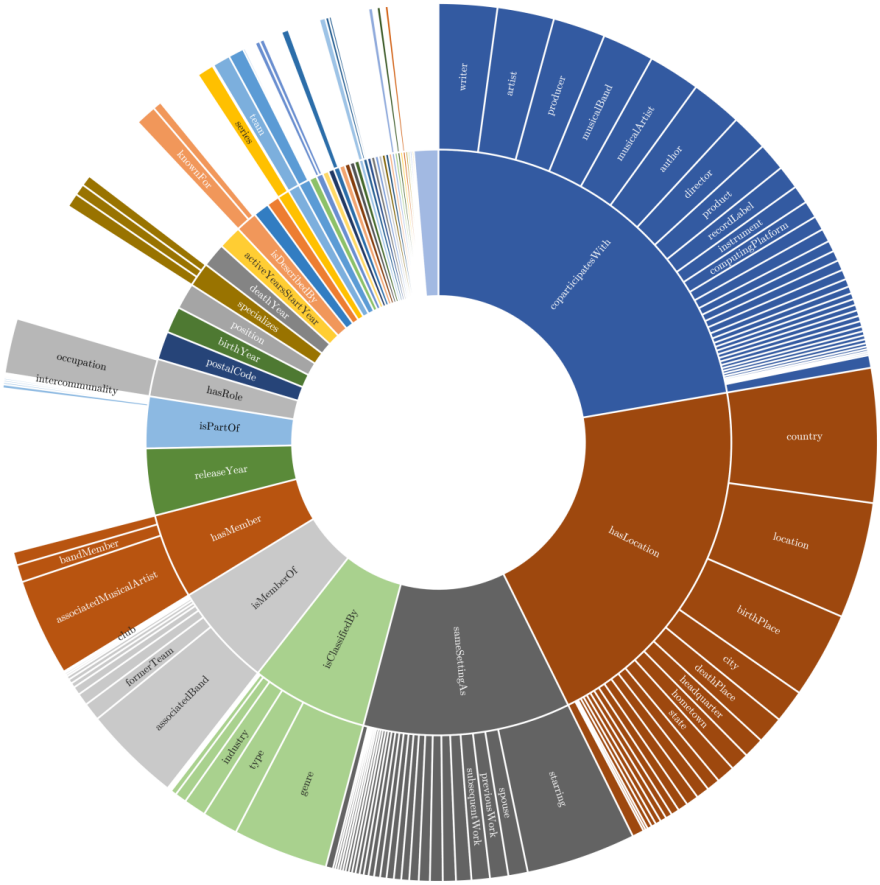

Fig. 13. Relation Type Distribution

6 Conclusion and Future Work

CC-DBP[11] is a dataset for training and benchmarking KBP systems. Unlike previous efforts, its modular approach allows immediate application to the relation prediction subtask, while also permitting the exploration and evaluation of new approaches in the EDL and CSC subtasks.

In future work we hope to address n-ary relations by supporting a more flexible definition of a context set, this will eliminate the need to filter mediator relations from DBpedia. We also will include a more advanced EDL module based on DBpedia Spotlight [5] or Tagme [6]. This will require blinding the models for these systems to the subset of DBpedia used for the held-out validation and test data.

[11] Available at: https://github.com/IBM/cc-dbp (Apache 2.0 License)

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Ives, Z.: Dbpedia: A nucleus for a web of open data. In: In 6th Intl Semantic Web Conference, Busan, Korea. pp. 11–15. Springer (2007)
2. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD international conference on Management of data. pp. 1247–1250. AcM (2008)
3. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Advances in neural information processing systems. pp. 2787–2795 (2013)
4. Chaganty, A., Paranjape, A., Liang, P., Manning, C.D.: Importance sampling for unbiased on-demand evaluation of knowledge base population. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 1049–1059 (2017)
5. Daiber, J., Jakob, M., Hokamp, C., Mendes, P.N.: Improving efficiency and accuracy in multilingual entity extraction. In: Proceedings of the 9th International Conference on Semantic Systems (I-Semantics) (2013)
6. Ferragina, P., Scaiella, U.: Fast and accurate annotation of short texts with wikipedia pages. IEEE software 29(1), 70–75 (2012)
7. Hewlett, D., Lacoste, A., Jones, L., Polosukhin, I., Fandrianto, A., Han, J., Kelcey, M., Berthelot, D.: Wikireading: A novel large-scale language understanding task over wikipedia. In: Proceedings of the Conference on Association for Computational Linguistics (2016)
8. Kohlschütter, C., Fankhauser, P., Nejdl, W.: Boilerplate detection using shallow text features. In: Proceedings of the Third ACM International Conference on Web Search and Data Mining. pp. 441–450. WSDM '10, ACM, New York, NY, USA (2010), http://doi.acm.org/10.1145/1718487.1718542
9. Mintz, M., Bills, S., Snow, R., Jurafsky, D.: Distant supervision for relation extraction without labeled data. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2. pp. 1003–1011. ACL '09, Association for Computational Linguistics, Stroudsburg, PA, USA (2009), http://dl.acm.org/citation.cfm?id=1690219.1690287
10. Riedel, S., Yao, L., McCallum, A.: Modeling relations and their mentions without labeled text. Machine learning and knowledge discovery in databases pp. 148–163 (2010)
11. Sandhaus, E.: The new york times annotated corpus. Linguistic Data Consortium, Philadelphia 6(12), e26752 (2008)
12. Toutanova, K., Chen, D., Pantel, P., Poon, H., Choudhury, P., Gamon, M.: Representing text for joint embedding of text and knowledge bases. In: EMNLP. vol. 15, pp. 1499–1509 (2015)
13. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Communications of the ACM 57(10), 78–85 (2014)
14. Wang, W.Y., Cohen, W.W.: Learning first-order logic embeddings via matrix factorization. In: IJCAI. pp. 2132–2138 (2016)