Automatic Generation of Benchmarks for Entity Recognition and Linking

Axel-Cyrille Ngonga Ngomo∗†, Michael Röder∗, Diego Moussallem†, Ricardo Usbeck∗ and René Speck†

∗Data Science Department
University of Paderborn, Paderborn, Germany
Email: {axel.ngonga, michael.roeder, ricardo.usbeck}@upb.de
†AKSW Research Group
University of Leipzig, Leipzig, Germany
Email: {moussallem, speck}@informatik.uni-leipzig.de

Abstract—The velocity dimension of Big Data plays an increasingly important role in processing unstructured data. Heretofore, no large-scale benchmarks were available to evaluate the performance of named entity recognition and entity linking solutions. This unavailability was due to the creation of gold standards for named entity recognition and entity linking being a time-intensive, costly and error-prone task. We hence investigate the automatic generation of benchmark texts with entity annotations for named entity recognition and linking from Linked Data. The main advantage of automatically constructed benchmarks is that they can be readily generated at any time, and are cost-effective while being guaranteed to achieve gold-standard quality. We compare the performance of 11 tools on the benchmarks we generate with their performance on 16 benchmarks that were created manually. Our results suggest that our automatic benchmark generation approach can create varied benchmarks that have characteristics similar to those of existing benchmarks. In addition, we perform a large-scale runtime evaluation of entity recognition and linking solutions for the first time in literature. Our experimental results are available at http://faturl.com/bengalexps?open.

Keywords—Benchmarks; Entity Linking; Entity Recognition; Disambiguation

I. INTRODUCTION

Benchmarking is of central importance for the objective assessment and development of approaches all around computer science. For example, developments in the database area suggest that benchmarks such as TPC-H were instrumental for increasing the performance of relational databases by orders of magnitude [1]. Recently, benchmarking campaigns such as BioASQ [2] have led to an improvement of the F-measure achieved by bio-medical question answering systems by more than 5%. While the precision, recall and F-measure of named entity recognition (NER) and entity linking (EL) systems have been evaluated against a plethora of benchmarks, the runtime performance of these solutions has been paid little attention to. This is due to the current creation of benchmarks being commonly carried out manually. The manual creation of NER and EL benchmarks has the advantage of yielding benchmarks which reflect human processing but also exhibits significant disadvantages:

1) Human annotators have to read through every sentence in the corpus and often (a) miss annotations or (b) assign wrong resources to entities for reasons as various as fatigue or lack of background knowledge (and this even when supported with annotation tools).
2) Manually created benchmarks are usually small (commonly < 2,500 documents, see Table I). Hence, they are of little help when aiming to benchmark the scalability of existing solutions. This holds especially in scenarios where techniques such as caching have to be circumvented.
3) Manual benchmark generation approaches lead to static corpora which tend not to reflect the newest reference knowledge graphs, also called knowledge bases (KBs). For example, several of the benchmarks presented in GERBIL [3] link to outdated versions of Wikipedia or DBpedia and hence contain resources that do not exist anymore.
4) Finally, manual benchmarks are commonly biased towards popular resources (e.g., resources with a high PageRank) [4].

The recent study [5] has shown that in addition to being resource-intensive, the manual benchmark creation process for NER and EL leads to benchmarks which often do not achieve gold standard quality (i.e., they are often not free of annotation mistakes). [5] was able to determine that up to 38,453 of the annotations in commonly used benchmarks (see GERBIL [3] for a list of these benchmarks) were erroneous. A manual evaluation of a sample of 25 documents from the ACE2004 benchmark revealed that 195 annotations were missing and 14 of 306 annotations were incorrect. A manual evaluation of AIDA/CONLL [6] and OKE2015 [7] revealed similar numerous mistakes in terms of number in existing benchmarks.

We argue that automatic methods are a viable and supplementary approach for the generation of benchmarks for NER and EL, especially as they address some of the weaknesses of the manual benchmark creation process. The main contribution of our paper is thus a novel approach for the automatic generation of benchmarks for NER and EL dubbed BENGAL. Our approach relies on the abundance of structured data in RDF on the Web and is based on
verbalizing such data to generate automatically annotated natural-language statements. Our automatic benchmark creation method addressed the drawbacks of manual benchmark generation aforementioned as follows:

1) It alleviates the human annotation error problem by relying on structured data (here, Linked Data in RDF\(^1\)) which explicitly contain the entities to find.
2) It solves the size problem by being able to generate arbitrarily large benchmarks. Hence, in addition to enhancing the measurement of the accuracy of approaches, it also ensures that the scalability of current solutions can be benchmarked.
3) Moreover, it can be updated easily to reflect the newest terminology and reference KBs. Hence, it can generate corpora that reflect the newest KBs.
4) It is not biased towards popular resources as it can choose entities to include in the benchmark generated following a uniform distribution.

The rest of this paper is structured as follows: We begin with an overview of the state of the art in benchmarking NER and EL. Then, in Section II, we explain our approach towards verbalizing RDF graphs and show how verbalized RDF can be used to create NER, EL and even relation extraction (RE) benchmarks. In Section IV we compare the features of our generated benchmarks as well as the results achieved by 11 state-of-the-art NER and EL frameworks with the features and results of manually crafted benchmarks. We discuss the insights provided by our evaluation and possible extensions of BENGAL in Section V.

II. RELATED WORK

A. Current benchmark datasets for NER and EL

According to GERBIL\(^2\), the 2003 CoNLL shared task\[^6\] is the most used benchmark dataset for recognition and linking. The corpus contains 1,393 manually annotated documents and is usually split into two testing and one training corpus. The ACE2004 and the MSNBC\[^8\] news datasets were used by Ratinov et al.\[^9\] to evaluate their seminal work on linking to Wikipedia. Another often used corpus is AQUAINT, e.g., used by Milne and Witten\[^10\]. The applied human-driven annotations allow for NER, EL and co-reference resolution\[^11\] where annotators manually disambiguated pre-recognized entities. Detailed dataset statistics on some of these benchmarks can be found in Table IV\[^4\].

Recently, there has been an uptake of publicly available corpora\[^12\],\[^13\] based on RDF. The Spotlight corpus and the KORE 50 dataset were proposed to show case the usability of RDF-based annotations\[^14\]. The multilingual N3 collection\[^12\] was introduced to widen the scope and diversity of NIF-based corpora. It has shown its usability for the evaluation of disambiguation tools\[^15\] and ensemble-learning based NER tools\[^16\]. Another recent observation is the shift towards micropost documents like tweets. The Microposts2014 corpus\[^17\] has been used to evaluate NERDML\[^18\]. The Open Knowledge Extraction challenge\[^19\] released open, manually created datasets containing NIF-based annotations for RDF entities and classes. Recently, van Erp et al.\[^4\] presented an analysis of existing benchmarks for NER and EL showing that these benchmarks tend to rely on very popular entities while entities with a low pagerank in the knowledge graph are rarely considered.

B. NER and EL benchmarking frameworks

Several benchmarking platforms for NER and EL evaluation have been introduced recently. The BAT framework\[^19\] is designed to facilitate the benchmarking of NER, EL and concept tagging approaches. Rizzo et al.\[^18\] present a study of NER and EL systems for annotating newswire and micropost documents. The GERBIL framework\[^3\] builds partly upon the formal foundation of the BAT framework but addresses the lack of treatment of NIF\[^4\].

C. Automatic generation of gold standards for NER and EL

Crowd-based approaches for the generation of gold standards/benchmarks for NER and EL commonly annotate existing text using one or more recognizers and then hand over the tasks of refinement and/or linking to crowd workers to improve the quality. For example, Voyer et al.\[^20\] used a pre-trained annotator to pre-annotate entities. Afterwards, naive crowd workers take a binary judgement to indicate the type of an entity. Semi-automatic approaches for the automatic generation of benchmarks use existing annotation approaches to generate a coherent annotation of existing text. For example, the CALBC silver standard\[^21\] provides biomedical text corpora based on the annotations of several existing tools which are then aligned using a voting scheme.

Recently, two approaches have been proposed for generating gold standard data sets automatically. First, Brümmner et al.\[^22\] presented an approach which converts abstractions from DBpedia (dbo:abstract) to benchmarkable datasets. For any given abstract, they gather the first paragraph of the corresponding Wikipedia page and use the text to extract entities through their own Wikipedia links. Second, Oramas et al.\[^23\] introduced a voting-based algorithm which analyses the hyperlinks presented in the input texts retrieved from different disambiguation systems such as Babelfy\[^24\]. Each entity mention in the input text is linked based on the degree of agreement across three state-of-the-art EL systems.

To the best of our knowledge, BENGAL is the first automatic approach that makes use of structured data and can be replicated on any KB for EL benchmarks. In contrast to the approaches reviewed by van Erp et al.\[^4\], our framework

\(^1\)Resource Description Format, see https://www.w3.org/RDF/

\(^2\)That is, entities that exist in the real world but not in the reference KB.
is not biased towards popular resources as it chooses entities following a uniform distribution.

III. THE BENGAL APPROACH

BENGAL is based on the observation that more than 30 billion facts pertaining to more than 3 billion entities are available in machine-readable form on the Web (i.e., as RDF triples). The basic intuition behind our approach is hence as follows: Given that NER and EL are often used in pipelines for the extraction of machine-readable facts from text, we can invert the pipeline and go from facts to text, thereby using the information in the facts to produce a gold standard that is guaranteed to contain no errors. In the following, we begin by giving a more formal overview of RDF. Thereafter, we present how we use RDF to automatically generate NER and EL benchmarks at scale.

A. Preliminaries and Notation

1) RDF: The notation presented herein is based on the RDF 1.1 specification. An RDF graph $G$ is a set of facts. Each fact is a triple $t = (s, p, o) \in (R \cup B) \times P \times (R \cup B \cup L)$ where $R$ is the set of all resources (i.e., things of the real world), $P$ is the set of all predicates (binary relations), $B$ is the set of all blank nodes (which basically express existential quantification) and $L$ is the set of all literals (i.e., of datatype values). We call the set $R \cup P \cup L \cup B$ our universe and call its elements entities. A fragment of DBpedia is shown below. We will use this fragment in our examples.

Listing 1: Example RDF dataset.

```
:AlbertEinstein dbo:birthPlace :Ulm .
:AlbertEinstein dbo:deathPlace :Princeton .
:AlbertEinstein rdf:type dbo:Scientist .
:AlbertEinstein dbo:field :Physics .
:Ulm dbo:country :Germany .
:AlbertEinstein rdfs:label "Albert Einstein" @en .
```

2) Benchmarks: We define a benchmark as a set $C$ of annotated documents $D_i$. Each document $D_i$ is a sequence of characters $s_{i1} \ldots s_{in}$. Each subsequence $s_{ij} \ldots s_{ik}$ (with $j < k$) of the document $D_i$ which stands for a resource $r \in R$ is assumed to be marked as such. We model the marking of resources by the function $m : C \times \mathbb{N} \times \mathbb{N} \to R$ and write $m(D_i, j, k) = r$ to signify that the substring $s_{ij} \ldots s_{ik}$ stands for the resource $r$. In case the substring $s_{ij} \ldots s_{ik}$ does not stand for a resource, we write $m(i, j, k) = \epsilon$. Let $D_0$ be the example shown in Listing 2. We would write $m(D_0, 0, 14) = :AlbertEinstein$.

Listing 2: Example sentence.

```
Albert Einstein was born in Ulm.
```

3) Verbalization: To the best of our knowledge, there are two main works on verbalizing SPARQL, i.e., SPARTIQUATION and SPARQL2NL. Our approach to verbalizing RDF is based on SPARQL2NL because it is extensible by virtue of being bottom-up, i.e., of specifying reusable rules to verbalize atomic constructs (e.g., RDF triples) and to combine their verbalization into sentences. In contrast, SPARTIQUATION assumes the structure of the sentence to be generated is known and fits the verbalization of the components into the template. The notation and formal framework for verbalization in BENGAL is also based on SPARQL2NL and presented below.

Let $W$ be the set of all words in the dictionary of our target language. We define the realization function $\rho : R \cup P \cup L \rightarrow W^{*}$ as the function which maps each entity to a word or sequence of words from the dictionary. Formally, the goal of the verbalization is to devise the extension of $\rho$ to conjunctions of RDF triples. This extension maps all triples $t$ to their realization $\rho(t)$ and defines how these atomic realizations are to be combined. We denote the extension of $\rho$ by the same label $\rho$ for the sake of simplicity. We adopt a rule-based approach to devise the extension of $\rho$, where the rules extending $\rho$ to RDF triples are expressed in a conjunctive manner. This means that for premises $P_1, \ldots, P_n$ and consequences $K_1, \ldots, K_m$ we write $P_1 \land \ldots \land P_n \Rightarrow K_1 \land \ldots \land K_m$. The premises and consequences are explicated by using an extension of the Stanford dependencies. We rely especially on the constructs explained in Table 1. For example, a possessive dependency between two phrase elements $e_1$ and $e_2$ is represented as $\text{poss}(e_1, e_2)$. For the sake of simplicity, we sometimes reduce the construct $\text{subj}(y, x) \land \text{dobj}(y, z)$ to the triple $(x, y, z) \in W^3$.

B. Approach

BENGAL assumes that it is given (1) a RDF graph $G \subseteq (R \cup B) \times P \times (R \cup B \cup L)$, (2) a number of documents to generate, (3) a minimal resp. maximal document size (i.e., number of triples to use during the generation process) $d_{min}$ resp. $d_{max}$, (4) a set of restrictions pertaining to the resources to generate and (5) a strategy for generating single documents. Given the graph $G$, BENGAL begins by selecting a set of seed resources from $G$ based on the restrictions set using parameter (4). Thereafter, it uses the strategy defined via parameter (5) to select a subgraph of $G$. This subgraph contains a randomly selected number $d$ of triples with $d_{min} \leq d \leq d_{max}$. The subgraph is then verbalized. The verbalization is annotated automatically and finally returned as a single document. Each single document then may be paraphrased if this option is chosen in the initial

---

1. Here we use the expression machine-readable in the sense of the Semantic Web, see http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225.
2. http://dbpedia.org/
3. SPARQL is the query language for RDF data. The specification can be found at https://www.w3.org/TR/rdf-sparql-query.
4. For a complete description of the vocabulary, see http://nlp.stanford.edu/software/dependencies_manual.pdf.
Table I: Dependencies (Dep.) used by BENGAL.

| Dependency | Explanation |
|------------|-------------|
| cc         | Stands for the relation between a conjunct and a given conjunction (in most cases and/or). For example in the sentence John eats an apple and a pear, cc(PEAR, AND) holds. We mainly use this construct to specify reduction and replacement rules. |
| conj*      | Used to build the conjunction of two phrase elements, e.g. conj(subj(EAT, JOHN), subj(DRINK, MARY)) stands for John eats and Mary drinks. conj is not to be confused with the logical conjunction \( \land \), which we use to state that two dependencies hold in the same sentence. For example subj(EAT, JOHN) \( \land \) subj(EAT, FISH) is to be read as John eats fish. |
| dobj       |Dependency between a verb and its direct object, for example dobj(EAT, APPLE) expresses to eat an/the apple. |
| nn         | The noun compound modifier is used to modify a head noun by the means of another noun. For instance nn(FARMER, JOHN) stands for farmer John. |
| poss       |Expresses a possessive dependency between two lexical items, for example poss(JOHN, DOG) expresses John’s dog. |
| subj       | Relation between subject and verb, for example subj(BE, JOHN) expresses John is. |

### Dependency Explanation

- **cc**: Stands for the relation between a conjunct and a given conjunction (in most cases and/or). For example in the sentence John eats an apple and a pear, cc(PEAR, AND) holds. We mainly use this construct to specify reduction and replacement rules.
- **conj***: Used to build the conjunction of two phrase elements, e.g., `conj(subj(EAT, JOHN), subj(DRINK, MARY))` stands for John eats and Mary drinks. `conj` is not to be confused with the logical conjunction `\( \land \)`, which we use to state that two dependencies hold in the same sentence. For example `subj(EAT, JOHN) \( \land \) subj(EAT, FISH)` is to be read as John eats fish.
- **dobj**: Dependency between a verb and its direct object, for example `dobj(EAT, APPLE)` expresses to eat an/the apple.
- **nn**: The noun compound modifier is used to modify a head noun by the means of another noun. For instance `nn(FARMER, JOHN)` stands for farmer John.
- **poss**: Expresses a possessive dependency between two lexical items, for example `poss(JOHN, DOG)` expresses John’s dog.
- **subj**: Relation between subject and verb, for example `subj(BE, JOHN)` expresses John is.

---

**2) Subgraph Generation:** Our approach to generating subgraphs is reminiscent of SPARQL query topologies as available in SPARQL query benchmarks such as DBPSB, BSBM, FEASIBLE and FedBench [27]. As these queries (especially the DBPSB and FEASIBLE queries) describe real information needs, their topology must stand for the type of information that is necessitated by applications and humans. We thus distinguish between three main types of subgraphs to be gathered from RDF data: (1) **star graphs** provide information about a particular entity, most commonly a resource (e.g., the short biography of a person); (2) **path graphs** describe the relation between two entities (e.g., the relation between a gene and a side-effect); (3) **hybrid graphs** are a mix of both and commonly describe a specialized subject matter involving several actors (e.g., a description of the cast of a movie).

**Star Graph Generation:** For each \( s_i \in S \), we simply gather all triples of the form \( t = (s_i, p, o) \in R \times P \times (R \cup L) \). Note that we do not consider blank nodes as they cannot be verbalized due to the existential quantification they stand for. The triples are then added to a list \( L(s_i) \) sorted in descending order according to a hash function \( h \). After randomly selecting a document size \( d \) between \( d_{\text{min}} \) and \( d_{\text{max}} \), we select \( d \) random triples from \( L(s_i) \). For the dataset shown in Listing 1 and \( d = 2 \), we would for example get Listing 4:

```
SELECT ?x WHERE { (?x a :Person.) UNION (?x a :Organization.) UNION (?x a :Place.) }
```

Listing 3: Example seed selection query.

**Path Graph Generation:** For each \( s_i \in S \), we begin by computing list \( L(s_i) \) as in the symmetric star graph generation. Then, we pick a random triple \((s_i, p, o)\) or \((o, p, s_i)\) from \( L(s_i) \) that is such that \( o \) is a resource. We then use \( o \) as seed and repeat the operation until we have generated \( d \) triples, where \( d \) is randomly generated as above. For the example dataset shown in Listing 1 and \( d = 2 \), we would for example get Listing 5:

```
{AlbertEinstein :birthPlace :Ulm . :AlbertEinstein :deathPlace :Princeton . }
```

Listing 4: Example dataset generated by the star strategy.

**Symmetric Star Graph Generation:** As above with \( t \in \{(s_i, p, o) \in G \lor (o, p, s_i) \in G\} \).

**Path Graph Generation:** For each \( s_i \in S \), we begin by computing list \( L(s_i) \) as in the symmetric star graph generation. Then, we pick a random triple \((s_i, p, o)\) or \((o, p, s_i)\) from \( L(s_i) \) that is such that \( o \) is a resource. We then use \( o \) as seed and repeat the operation until we have generated \( d \) triples, where \( d \) is randomly generated as above. For the example dataset shown in Listing 1 and \( d = 2 \), we would for example get Listing 5:

```
{AlbertEinstein :birthPlace :Ulm . :Ulm :country :Germany . }
```

Listing 5: Example dataset generated by the path strategy.

**Hybrid Graph Generation:** This is a 50/50 mix of the star and path graph generation approaches. In each...
iteration, we choose and apply one of the two strategies above randomly. For example, the hybrid graph generation can generate:

\[ t = (s, p, o) \]

\[ s:AlbertEinstein \ r:birthPlace \ o:Ulm . \]
\[ s:AlbertEinstein \ r:deathPlace \ o:Princeton . \]
\[ o:Ulm \ r:country \ o:Germany . \]

Listing 6: Example dataset generated by the hybrid strategy.

Summary Graph Generation: This last strategy is a specialization of the star graph generation where the set of triples to a resource is not chosen randomly. Instead, for each class (e.g., :Person) of the input KB, we begin by filtering the set of properties and only consider properties that (1) have the said class as domain and (2) achieve a coverage above a user-set threshold (60% in our experiments) (e.g., :birthPlace, :deathPlace, :spouse). We then build a property co-occurrence graph for the said class in which the nodes are the properties selected in the preceding step and the co-occurrence of two properties \( p_1 \) and \( p_2 \) is the instance \( r \) of the input class where \( \exists o_1, o_2 : (r, p_1, o_1) \in K \land (r, p_2, o_2) \in K \). The resulting graph is then clustered (e.g., by using the approach presented in [28]). We finally select the clusters which contain the properties with the highest frequencies in \( K \) that allow the selection of at least \( d \) triples from \( K \). For example, if :birthPlace (frequency = 10), :deathPlace (frequency = 10) were in the same cluster while :spouse (frequency = 8) were in its own cluster, we would choose the pair (:birthPlace, :deathPlace) and return the corresponding triples for our input resource. Hence, we would return Listing 4 for our running example.

3) Verbalization: The verbalization strategy for the first four strategies consists of verbalizing each triple as a single sentence and is derived from SPARQL2NL [26]. To verbalize the subject of the triple \( t = (s, p, o) \), we use one of its labels according to Ell et al. [29] (e.g., the rdfs:label). If the object \( o \) is a resource, we follow the same approach as for the subject. Importantly, the verbalization of a triple

\[ \rho(s \ p \ o) \Rightarrow \text{poss}(\rho(p), \rho(s)) \land \text{dobj}(\rho(E), \rho(o)) \]

In our running example, verbalizing (:AlbertEinstein, dbo:birthDate, :Ulm) would thus lead to Albert Einstein’s birth place is Ulm., as birth place is a noun. In the case of summary graphs, we go beyond the verbalization of single sentences and merge sentences that were derived from the same cluster. For example, if \( p_1 \) and \( p_2 \) can be verbalized as nouns, then we apply the following rule: resume

\[ \rho(s \ p_1 \ o_1) \land \rho(s \ p_2 \ o_2) \Rightarrow \text{conj}(\text{poss}(\rho(p_1), \rho(s))) \land \text{dobj}(\rho(E_1), \rho(o_1)) \land \text{poss}(\rho(p_2), \rho(\text{pronoun}(s))) \land \text{dobj}(\rho(E_2), \rho(o_2)) \]

Note that \( \text{pronoun}(s) \) returns the correct pronoun for a resource based on its type and gender. Therewith, we can generate Albert Einstein’s birth place is Ulm and his death place is Princeton.

4) Paraphrasing: With this step, BENGAL avoids the generation of a large number of sentences that share the same terms and the same structure [30]. Additionally, this step makes the use of reverse engineering strategies for the generation more difficult as it increases the diversity of the text in the benchmarks. Our paraphrasing is largely based on [31] and runs as follows: (1) change the structure of the sentence, (2) change the voice from active to passive and (3) look for synonyms based on the context. For each document,
we run the paraphrasing sequentially on all sentences. For steps (1) and (2), BENGAL relies on syntactic structure analysis \cite{32} combined with part-of-speech tagging \cite{33}. We first determine the location of the verb in the sentence. In most cases, the subject and object of the verb are then swapped and the verb rendered in the passive voice. We however refrain from using the passive if the verb is a form of *to be* as the sentences would not sound natural. Instead, we make use of the symmetry of *to be* and swap subject and object (see second sentence in Listing 7). We also refrain from changing sentences that describe type information (e.g., see the first sentence Listing 7).

### Listing 7: Example Paraphrasing.

| Original: Albert Einstein is a scientist. His birth date is March 12, 1879. His field is Physics. Albert Einstein died in April 16, 1955. This scientists' birth places are Ulm, Baden - Württemberg, German Empire and Kingdom of Württemberg. |
| Paraphrase: Albert Einstein is a scientist. March 12, 1879 is his birth date. Physics is his area. This physicist passed away in April 16, 1955. This scientist's birth places are Ulm, Baden - Württemberg, German Empire and Kingdom of Württemberg. |

For step (3), BENGAL looks for synonyms of the noun phrases in the sentence using a dictionary (i.e., WordNet\footnote{https://wordnet.princeton.edu/} in our current implementation). Synonyms are selected based on their synsets. Each word is queried along with its POS-tag to avoid ambiguity. If one word returns more than a given number of synonyms (5 in our experiments) we assume it to be ambiguous and maintain the original. For example, we do not alter the verb *get* due to the plurality of its meanings. In the same vein, we do not retrieve multi-word expressions as synonyms. For example, we would not replace the verb *kick the bucket* by *die* or *pass away* however retrieved and used to replace verbs such as *die* (see third sentence in Listing 7). The paraphrasing in BENGAL also addresses the replacement of named entities. Here, the approach makes use of alternative surface forms \cite{34} for resources (see third sentence in Listing 7). Furthermore, the paraphrasing module replaces pronouns by surface forms (see last sentence in Listing 8 where “It” is replaced by the surface form “Pettus”) if these pronouns are used very frequently (in our implementation, more than 3 times).

### Listing 8: Example Paraphrasing at Summary Generation

| Paraphrased: Edmund Pettus Bridge is a bridge. It crosses Alabama River. Through arch bridge is its type. Pettus was declared a National Historic Landmark on March 11, 2013. |

### IV. Experiments and Results

We generated 13 datasets to evaluate our approach\footnote{The datasets are available at \url{http://hobbitdata.informatik.uni-leipzig.de/bengal/bengal_dtypes.zip}}. The first 10 were generated by running our five sub-graph generation methods with and without paraphrasing. The number of documents was set to 100 while \((d_{\text{min}}, d_{\text{max}})\) was set to \((1, 5)\). The 11th dataset shows how BENGAL can be used to evaluate the scalability of approaches. Here, we used the hybrid generation strategy to generate 10,000 documents. The 12th and 13th generated datasets comprise 10 longer documents each with \(d_{\text{min}}\) set to 90. For the 12th dataset, we focused on generating a high number of entities in the documents while the 13th dataset contains less entities but the same number of documents.

We compared the 13 generated datasets with the 16 manually created gold standards presented below. The comparison was carried out in two ways. First, we assessed the features of the datasets. Then, we compared the micro F-measure of 11 NER and EL frameworks on the manually and automatically generated datasets. We chose to use these 11 frameworks because they are included in GERBIL. This inclusion ensures that their interfaces are compatible and their results comparable.

#### A. Dataset features

The first aim of our evaluation was to evaluate the variability of the datasets generated by BENGAL. To this end, we compared the distribution of the part of speech (POS) tags of the BENGAL datasets with those of that of the 11 benchmark datasets. An analysis of the Pearson correlation of these distribution revealed that the manually created datasets (D1 to D16) have a high correlation (0.88 on average) with a minimum of 0.61 (D10–D16)\footnote{Our complete results can be found at \url{https://goo.gl/WVzzjk}}. The correlation of the POS tag distributions between BENGAL datasets and a manually created dataset vary between 0.34 (D7–B11) and 0.89 (D14–B9) with an average of 0.67. This shows that BENGAL datasets can be generated to be similar to manually created datasets (D14–B9) as well as to be very different to them (D7–B11). Hence, BENGAL can be used for testing sentence structures that are not common in the current manually generated benchmarks.

We also studied the distribution of entities and tokens across the datasets in our evaluation. Table \ref{tab:dataset_features} gives an overview of these distributions, where \(E\) is the set of entities in the corpus \(C\). The distribution of values for the different
features is very diverse across the different manually created datasets (see Figure 2). This is mainly due to (1) different ways to annotate entities and (2) the domains of the datasets (news, description of entities, microposts). As shown in Table I and Figure 2, BENGAL can be easily configured to generate a wide variety of datasets with similar quality and number of documents to those of real datasets. Because our approach can generate benchmarks ranging from (1) benchmarks with sentences containing a large number of entities without any unnecessary filler terms in between (high entity density) to (2) benchmarks which contain more information pertaining to entity types and literals (low entity density).

![Figure 2: Average entities per document and average tokens (|T|) per document for each dataset.](image)

**B. Annotator performance**

We used GERBIL [3] to evaluate the performance of 11 annotators on the manually created as well as the BENGAL datasets. We evaluated the annotators within an A2KB (annotation to knowledge base) experiment setting: Each document of the corpora was sent to each annotator. The annotator had to find and link all entities to a reference KB (here DBpedia). We measured both the performance of the NER and the EL steps. For the evaluation of the annotators’ performance, we used the weak annotation match. Table III shows the micro F1-score of the different annotators on chosen datasets. The manually created datasets showed diverse results. We analyzed the results further by using the F1-scores of the annotators as features of the datasets. Based on these feature vectors, we calculated the Pearson correlations between the datasets to identify datasets with similar characteristics.

The Pearson correlations of the F-measures achieved by the different annotators on the AIDA/CoNLL datasets (D2–D5) are very high (0.95–1.00) while the correlation between the results on the Spotlight corpus (D7) and N3-Reuters-128 (D13) is around 0.62. The results on D1 and D12–D15 have a correlation to the AIDA/CoNLL results (D2–D5) that is higher than 0.5. In contrast, the correlations of D7 and D8 to the AIDA/CoNLL datasets range from -0.54 to -0.36. These correlations highlight the diversity of the manually created datasets and suggest that creating an approach which emulates all datasets is non-trivial.

Like the correlations between the manually created datasets, the correlations between the results achieved on BENGAL datasets and hand-crafted datasets vary. The results on BENGAL correlate most with the results on the OKE 2015 data. The highest correlations were achieved with the OKE 2015 Task 1 dataset and range between 0.89 and 0.92. This suggests that our benchmark can emulate entity-centric benchmarks. The correlation of BENGAL with OKE is however reduced to 0.82 in D13, suggesting that BENGAL can be parametrized so as to diverge from such benchmarks. A similar observation can be made for the correlation D12 and ACE2004, where the correlation increased with the size of the documents in the benchmark. The correlation between the results across BENGAL datasets varies between 0.54 and 1, which further supports that BENGAL can generate a wide range of diverse datasets.

**C. Scalability**

An advantage of BENGAL is that it allows the generation of large data corpora. Therewith, BENGAL allows the testing of existing systems for their scalability while circumventing technologies such as caching, which an approach based on running through the same small benchmark several times would be confronted with. To showcase this feature of BENGAL, we created the dataset B11 with 10,000 documents using the hybrid graph generation. Every document has between 3 and 20 sentences. The dataset features for B11

---

12The complete result set can be found online at [http://w3id.org/gerbil/experiment?id=201603140002](http://w3id.org/gerbil/experiment?id=201603140002)

13In the weak annotation setting, partial matches of entities are considered correct. For example, the annotation “Barack Obama” is considered correct even if the gold standard annotates “President Barack Obama”. Using a strong matching leads to counting this result as wrong while humans might rate it as correct. We hence chose the weak match in our evaluation.

14All values can be found at [https://tinyurl.com/kjre3rb](https://tinyurl.com/kjre3rb)
Table II: Features of datasets. The datasets B4, B6, B8 and B10 are paraphrased versions of B3, B5, B7 resp. B9 and share similar characteristics.

| ID  | Name                          | Doc. | Tokens | Entities | $|T|/|C| | $|E|/|C| | $|E|/|T| |
|-----|-------------------------------|------|--------|----------|-----------|-----------|-----------|
| D1  | ACE2004                       | 57   | 21312  | 306      | 373.9     | 5.4       | 0.01      |
| D2  | AIDA/CoNLL-Complete           | 1393 | 245008 | 34929    | 175.9     | 25.1      | 0.14      |
| D3  | AIDA/CoNLL-Test A            | 216  | 41757  | 5917     | 193.3     | 27.4      | 0.14      |
| D4  | AIDA/CoNLL-Test B            | 231  | 37687  | 5616     | 163.1     | 24.3      | 0.15      |
| D5  | AIDA/CoNLL-Training          | 946  | 165564 | 23396    | 175.0     | 24.7      | 0.14      |
| D6  | AQUAINT                       | 50   | 11024  | 727      | 220.5     | 14.5      | 0.07      |
| D7  | DBpediaSpotlight              | 58   | 1661   | 330      | 28.6      | 5.7       | 0.20      |
| D8  | IITB                          | 104  | 66531  | 18308    | 639.7     | 176.0     | 0.28      |
| D9  | KORE50                        | 50   | 640    | 144      | 12.8      | 2.9       | 0.23      |
| D10 | Microposts2014-Test           | 1055 | 20648  | 1256     | 19.6      | 1.2       | 0.06      |
| D11 | Microposts2014-Train          | 2340 | 40684  | 3822     | 17.4      | 1.6       | 0.09      |
| D12 | MSNBC                        | 20   | 10877  | 747      | 543.9     | 37.4      | 0.07      |
| D13 | N3-Reuters-128                | 128  | 15842  | 880      | 123.8     | 6.9       | 0.06      |
| D14 | N3-RSS-500                    | 500  | 15504  | 1000     | 31.0      | 2.0       | 0.06      |
| D15 | OKE 2015 Task 1 evaluation   | 101  | 3064   | 664      | 30.3      | 6.6       | 0.22      |
| D16 | OKE 2015 Task 1 train         | 95   | 1946   | 341      | 20.5      | 3.6       | 0.18      |

| ID  | Name                          | Doc. | Tokens | Entities | $|T|/|C| | $|E|/|C| |
|-----|-------------------------------|------|--------|----------|-----------|-----------|
| B1  | BENGAL Path 100               | 100  | 1202   | 362      | 12.02     | 3.6       |
| B2  | BENGAL Path Para 100         | 100  | 1250   | 362      | 12.5      | 3.6       |
| B3  | BENGAL Star 100               | 100  | 3039   | 880      | 30.39     | 8.8       |
| B4  | BENGAL Star Para 100          | 100  | 2772   | 543      | 27.72     | 5.43      |
| B5  | BENGAL Sym 100                | 100  | 2718   | 725      | 27.18     | 7.25      |
| B6  | BENGAL Sym Para 100           | 100  | 2621   | 612      | 26.21     | 6.12      |
| B7  | BENGAL Hybrid 100             | 100  | 1811   | 528      | 18.11     | 5.28      |
| B8  | BENGAL Hybrid Para 100        | 100  | 1732   | 420      | 17.32     | 4.2       |
| B9  | BENGAL Summary 100            | 100  | 2033   | 637      | 20.33     | 6.37      |
| B10 | BENGAL Summary Para 100       | 100  | 2025   | 646      | 20.25     | 6.46      |
| B11 | BENGAL Hybrid 10000           | 10000| 556483 | 165254   | 55.6      | 16.5      |
| B12 | BENGAL Hybrid Long 10         | 10   | 9162   | 2417     | 241.7     | 916.2     |
| B13 | BENGAL Star Long 10           | 10   | 7369   | 316      | 736.9     |

Figure 3: Runtimes of different NER/EL tools on B11.

can be found in Table II. We separated the dataset in 5 parts that we used for 5 phases of the benchmarking. During the different phases, 1, 2, 3, 8, 2000 documents were sent to the annotation systems. All experiments were carried out on a Docker Swarm cluster of 3 servers, each running Ubuntu 12.4 on 2x E5-2630v3 8-Cores (2.4GHz) with 256GB RAM. Figure 3 shows the behavior of six different NER tools in our experiments. This is the first large-scale runtime evaluation of NER tools to the best of our knowledge. As expected, the processing time per document increases with the number of documents sent per time unit, with the best performing tools needing approximately 0.8s per document on average when under a small load (Phase I) and up to 10,000s per document on average when faced with a batch of 2000 documents. This long time was caused by documents having to wait in a queue if they can not be processed directly due to missing free resources. This clearly suggest that load balancing strategies for NER tools should be taken into consideration in future works. Interestingly, all tools based on single algorithms (FOX is an ensemble learning framework) perform in a comparable fashion. While the scaling of other tools will clearly be different from our experimental results, this experiment confirms that BENGAL paves the way for scalability benchmarking experiments for NER and EL within a variety of settings.

V. DISCUSSION AND CONCLUSION

We presented and evaluated an approach for generating NER and EL benchmarks automatically. Our results suggests

---

The complete experimental results can be found at http://master.project-hobbit.eu/#/experiments/details?id=1502263219770%2C1502192757931%2C1501852574348%2C1501852527351%2C1501852487461%2C1501852242060.
Table III: Excerpt of micro F1-scores of the annotators for the A2KB experiments on chosen datasets. N/A means that the annotator stopped with an error.

| Experiment | Dataset ID | A2KB | D1 | D2 | D3 | D4 | D5 | D6 | D7 | D8 | D9 | D10 | D11 | D12 | D13 | D14 | D15 | D16 | B1 | B2 | B3 | B4 | B5 | B6 | B7 | B8 | B9 | B10 | B11 | B12 | B13 |
|------------|------------|------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| A2KB       |            | 0.26 | 0.68 | 0.26 | 0.36 | 0.65 | 0.66 | 0.66 | 0.72 | 0.14 | 0.14 | 0.21 | 0.15 | 0.14 | 0.22 | 0.17 | 0.21 | 0.13 |
|            |            | 0.31 | 0.66 | 0.3 | 0.66 | 0.66 | 0.66 | 0.66 | 0.56 | 0.66 | 0.66 | 0.66 | 0.56 | 0.56 | 0.56 | 0.66 | 0.66 | 0.56 |
|            |            | 0.15 | 0.66 | 0.3 | 0.66 | 0.66 | 0.66 | 0.66 | 0.56 | 0.66 | 0.66 | 0.66 | 0.56 | 0.56 | 0.56 | 0.66 | 0.66 | 0.56 |
|            |            | 0.25 | 0.66 | 0.3 | 0.66 | 0.66 | 0.66 | 0.66 | 0.56 | 0.66 | 0.66 | 0.66 | 0.56 | 0.56 | 0.56 | 0.66 | 0.66 | 0.56 |
|            |            | 0.19 | 0.66 | 0.3 | 0.66 | 0.66 | 0.66 | 0.66 | 0.56 | 0.66 | 0.66 | 0.66 | 0.56 | 0.56 | 0.56 | 0.66 | 0.66 | 0.56 |
|            |            | 0.21 | 0.66 | 0.3 | 0.66 | 0.66 | 0.66 | 0.66 | 0.56 | 0.66 | 0.66 | 0.66 | 0.56 | 0.56 | 0.56 | 0.66 | 0.66 | 0.56 |

that our approach can generate diverse benchmarks with characteristics similar to those of a large proportion of existing benchmarks. Importantly, the precautions taken to limit the reverse engineering of BENGAL datasets (which is an obvious weakness of the approach) do not affect the performance of the tools as revealed by the correlation of tool results on original documents and their paraphrases being strongly correlated (between 0.95 and 1). In addition, BENGAL allows the study of aspects of frameworks (such as scalability) which are hard to analyze with current benchmarks. Overall, our results suggest that BENGAL benchmarks can ease the development of NER and EL tools by providing developers with insights into their performance at virtually no cost. Hence, BENGAL can improve the push towards better NER and EL frameworks.

REFERENCES

[1] J. Zhang, A. Sivasubramaniam, H. Franke, N. Gautam, Y. Zhang, and S. Nagar, “Synthesizing representative i/o workloads for tpc-h,” in 10th HPCA. Washington, DC, USA: IEEE Computer Society, 2004.

[2] G. Tsatsaronis, M. Schroeder, G. Paliouras, Y. Almirantis, I. Andreoustopoulos, E. Gaussier, P. Gallinari, T. Artieres, M. Alvers, M. Zschunke, and A.-C. Ngonga Ngomo, “BioASQ: A challenge on large-scale biomedical semantic indexing and question answering,” in AAAI Information Retrieval and Knowledge Discovery in Biomedical Text, 2012.

[3] R. Usbeck, M. Röder, A.-C. Ngonga Ngomo, C. Baron, A. Both, M. Brümmer, D. Caccarelli, M. Cornolti, D. Cherix, B. Eickmann, P. Ferragina, C. Lemke, A. Moro, R. Navigli, F. Piccinno, G. Rizzo, H. Sack, R. Speck, R. Troncy, J. Waitelonis, and L. Wesemann, “Gerbil: General entity annotator benchmarking framework,” in WWW ’15. International World Wide Web Conferences Steering Committee, 2015.

[4] M. van Erp, P. Mendes, H. Paulheim, F. Ilievski, J. Plu, G. Rizzo, and J. Waitelonis, “Evaluating entity linking: An analysis of current benchmark datasets and a roadmap for doing a better job,” in Proceedings of LREC, 2016.

[5] K. Jha, M. Röder, and A.-C. N. Ngomo, “All that glitters is not gold–rule-based curation of reference datasets for named entity recognition and entity linking,” in The Semantic Web–ISWC 2017, 2017.

[6] E. F. Tjong Kim Sang and F. De Meulder, “Introduction to the conll-2003 shared task: language-independent named entity
recognition,” in Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4, 2003, pp. 142–147.

[7] A. G. Nuzzolese, A. L. Gentile, V. Presutti, A. Gangemi, D. Gariglioni, and R. Navigli, “Open knowledge extraction challenge,” in Semantic Web Evaluation Challenge. Springer, 2015, pp. 3–15.

[8] S. Cucerzan, “Large-scale named entity disambiguation based on wikipedia data,” in Conference on Empirical Methods in Natural Language Processing-CoNLL, 2007.

[9] L. Ratinov, D. Roth, D. Downey, and M. Anderson, “Local and global algorithms for disambiguation to wikipedia,” in Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, 2011, pp. 1375–1384.

[10] D. Milne and I. H. Witten, “Learning to link with wikipedia,” in 17th ACM CIKM, 2008, pp. 509–518.

[11] A.-C. N. Ngomo, M. Röder, and R. Usbeck, “Cross-document coreference resolution using latent features,” ser. LD4IE’14, 2014.

[12] M. Röder, R. Usbeck, D. Gerber, S. Hellmann, and A. Both, “N! - A Collection of Datasets for Named Entity Recognition and Disambiguation in the NLP Interchange Format,” in LREC, 2014.

[13] N. Steinmetz, M. Knuth, and H. Sack, “Statistical analyses of named entity disambiguation benchmarks,” in 1st Workshop on NLP&DBpedia 2013, 2013, pp. 91–102.

[14] P. N. Mendes, M. Jakob, A. Garcia-Silva, and C. Bizer, “DBpedia Spotlight: Shedding Light on the Web of Documents,” in 7th International Conference on Semantic Systems (1-Semantics), 2011, pp. 1–8.

[15] R. Usbeck, A.-C. N. Ngomo, M. Röder, D. Gerber, S. A. Coelho, S. Auer, and A. Both, “AGDISTIS-graph-based disambiguation of named entities using linked data,” in International Semantic Web Conference. Springer, 2014.

[16] R. Speck and A.-C. N. Ngomo, “Ensemble learning for named entity recognition,” in International Semantic Web Conference. Springer, 2014, pp. 519–534.

[17] A. E. Cano Basave, G. Rizzo, A. Varga, M. Rowe, M. Stankovic, and A.-S. Dadzie, “Making sense of microposts (#microposts2014) named entity extraction & linking challenge,” in Proceedings of 4th Workshop on Making Sense of Microposts (#Microposts), vol. 1141, 2014, pp. 54–60.

[18] G. Rizzo, M. van Erp, and R. Troncy, “Benchmarking the extraction and disambiguation of named entities on the semantic web,” in 9th LREC, 2014, pp. 4593–4600.

[19] M. Cornolti, P. Ferragina, and M. Ciaramita, “A framework for benchmarking entity-annotation systems,” in 22nd WWW Conference, 2013, pp. 249–260.

[20] R. Voyer, V. Nygaard, W. Fitzgerald, and H. Copperman, “A hybrid model for annotating named entity training corpora,” in Proceedings of the Fourth Linguistic Annotation Workshop, 2010, pp. 243–246.

[21] D. Rebholz-Schuhmann, A. J. J. Yepes, E. M. Van Mulligen, N. Kang, J. Kors, D. Milward, P. Corbett, E. Buyko, E. Beisswanger, and U. Hahn, “Calbc silver standard corpus,” Journal of bioinformatics and computational biology, 2010.

[22] M. Brümmel, M. Dojchinovski, and S. Hellmann, “Dbpedia abstracts: A large-scale, open, multilingual NLP training corpus,” in Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016., 2016.

[23] S. Oramas, L. E. Anke, M. Sordo, H. Saggon, and X. Serra, “ELMD: an automatically generated entity linking gold standard dataset in the music domain,” in 10th LREC 2016, Portorož, 2016.

[24] A. Moro, F. Cecconi, and R. Navigli, “Multilingual word sense disambiguation and entity linking for everybody,” in Proceedings of the 2014 International Conference on Posters & Demonstrations Track-Volume 1272. CEUR-WS. org, 2014, pp. 25–28.

[25] B. Ell, D. Vrandečić, and E. Simperl, “Spartiquation: Verbalizing sparql queries,” in Extended Semantic Web Conference. Springer, 2012, pp. 117–131.

[26] A.-C. Ngonga Ngomo, L. Bühmann, C. Unger, J. Lehmann, and D. Gerber., “Sorry, i don’t speak sparql — translating sparql queries into natural language,” in Proceedings of WWW, 2013, pp. 977–988.

[27] M. Saleem, Y. Khan, A. Hasnain, I. Ermilov, and A.-C. Ngonga Ngomo, “A fine-grained evaluation of sparql endpoint federation systems,” Semantic Web, pp. 1–26, 2015.

[28] A.-C. Ngonga Ngomo and F. Schumacher, “Borderflow: A local graph clustering algorithm for natural language processing,” in Computational Linguistics and Intelligent Text Processing. Springer, 2009, pp. 547–558.

[29] B. Ell, D. Vrandečić, and E. Simperl, “Labels in the web of data,” The Semantic Web–ISWC 2011, pp. 162–176, 2011.

[30] M. Young et al., Technical writer’s handbook. University Science Books, 2002.

[31] I. Androutsopoulos and P. Malakasiotis, “A survey of paraphrasing and textual entailment methods,” Journal of Artificial Intelligence Research, pp. 135–187, 2010.

[32] A. Gatt and E. Reiter, “SimpleNlg: A realisation engine for practical applications,” in Proceedings of the 12th European Workshop on Natural Language Generation, 2009, pp. 90–93.

[33] K. Toutanova and C. D. Manning, “Enriching the knowledge sources used in a maximum entropy part-of-speech tagger,” ser. EMNLP ’00, 2000.

[34] V. Bryl, C. Bizer, and H. Paulheim, “Gathering alternative surface forms for dbpedia entities.” in NLP-DBPEDIA@ ISWC, 2015, pp. 13–24.