Automatic Generation of Benchmarks for Entity Recognition and Linking

Axel-Cyrille Ngonga Ngomo[12], Michael Röder[1], Diego Moussallem[2], Ricardo Usbeck[1], and René Speck[2]

[1]Data Science Department, University of Paderborn, Germany
axel.ngonga|michael.roeder|ricardo.usbeck@upb.de
[2]AKSW Research Group, University of Leipzig, Germany
moussallem|speck@informatik.uni-leipzig.de

Abstract. Benchmarks are central to the improvement of named entity recognition and entity linking solutions. However, recent works have shown that manually created benchmarks often contain mistakes. We hence investigate the automatic generation of benchmarks for named entity recognition and linking from Linked Data as a complement to manually created benchmarks. 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 be free of annotation errors. Moreover, generators for resource-poor languages can foster the development of tools for such languages. We compare the performance of 11 tools on benchmarks generated using our approach with their performance on 16 benchmarks that were created manually. In addition, we perform a large-scale runtime evaluation of entity recognition and linking solutions for the first time in literature. Moreover, we present results achieved on the Portuguese version of our approach on four different tools. Overall, our results suggest that our automatic benchmark generation approach can create varied benchmarks that have characteristics similar to those of existing benchmarks. Our experimental results are available at http://faturl.com/bengalexp.

1 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 [34]. Recently, benchmarking campaigns such as BioASQ [29] have led to an improvement of the F-measure achieved by bio-medical question answering systems by more than 5%. While the manual creation of NER and EL benchmarks has the advantage of yielding benchmarks which reflect human processing, it also exhibits significant disadvantages: (1) *Annotation mistakes*: 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). For example, [11] was able to determine that up to 38,453 of the annotations in commonly used benchmarks (see GERBIL [31] for a list of these benchmarks) were erroneous.

A manual evaluation of 25 documents from the ACE2004 benchmark revealed that 195 annotations were missing and 14 of 306 annotations were incorrect. Similar findings were reported for AIDA/CONLL [27] and OKE2015 [20]. (2) *Volume*: Manually created benchmarks are usually small (commonly $< 2,500$ documents, see Table 2). Hence, they are of little help when aiming to benchmark the scalability of existing solutions (especially when these solutions using caching). (3) *Lack of updates*: 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 [31] link to outdated versions of Wikipedia/DBpedia. (4) *Popularity bias*: Manual benchmarks are often biased towards popular resources [8]. (5) *Lack of availability*: The lack of benchmarks for resource-poor languages inhibits the development of corresponding NER and EL solutions.

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 addresses the drawbacks of manual benchmark generation aforementioned as follows: (1) It alleviates the human annotation error problem by relying on data in RDF which explicitly contain the entities to find. (2) BENGAL is able to generate arbitrarily large benchmarks. Hence, it can enhance both the measurement of both the accuracy and the scalability of approaches.(3) Moreover, BENGAL can be updated easily to reflect the newest terminology and reference KBs. Hence, it can generate corpora that reflect the newest KBs. (4) BENGAL suffers less of a bias towards popular resources as it can choose entities to include in the benchmark generated following a uniform distribution. (5) BENGAL can be ported to any token-based language. This is exemplified by porting BENGAL to Brazilian Portuguese, a language with a limited number of NER and EL benchmarks.

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, we explain our approach and show how verbalized RDF can be used to create NER, EL and even relation extraction (RE) benchmarks. In Section 4, 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 5.

2 Related Work

The work present herein is mostly related to works on NER and EL benchmarks. According to GERBIL [31], the 2003 CoNLL shared task [27] is the most used benchmark dataset for recognition and linking. The ACE2004 and the MSNBC [5] news datasets were used by Ratinov et al. [22] to evaluate their seminal work on linking to Wikipedia. Another often-used corpus is AQUAINT, e.g., used by Milne and Witten [13]. The applied human-driven annotations allow for NER, EL and co-reference

resolution [17] where annotators manually disambiguated pre-recognized entities. Detailed dataset statistics on some of these benchmarks can be found in Table 2.

A recent uptake of publicly available corpora [24,26] based on RDF has led to the creation of many new datasets. The Spotlight corpus and the KORE 50 dataset were proposed to showcase the usability of RDF-based annotations [12]. The multilingual N3 collection [24] was introduced to widen the scope and diversity of NIF-based corpora. It has shown its usability for the evaluation of disambiguation tools [30] and ensemble-learning based NER tools [25]. Another recent observation is the shift towards micropost documents like tweets. For example, the Microposts2014 corpus [4] was created to evaluate NER and ML on smaller pieces of text. The Open Knowledge Extraction challenge [20] released open, manually created datasets containing NIF-based annotations for RDF entities and classes.

Semi-automatic approaches to benchmark creation are commonly crowd-based. They commonly use one or more recognizers to create a first set of annotations and then hand over the tasks of refinement and/or linking to crowd workers to improve the quality. Examples of such approaches include [32] and CALBC [23]. Oramas et al. [21] introduced a voting-based algorithm which analyses the hyperlinks presented in the input texts retrieved from different disambiguation systems such as Babelfy [14]. Each entity mention in the input text is linked based on the degree of agreement across three state-of-the-art EL systems.

Brümmer et al. [2] presents an automatic approach which converts abstracts 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. However, the approach is not guaranteed to return complete not correct annotations. 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. [8], our framework is not biased towards popular resources as it chooses entities following a uniform distribution.

3 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 generate NER and EL benchmarks automatically and at scale.

3.1 Preliminaries and Notation

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[1] is shown below. We will use this fragment in our examples. For the sake of space, our examples are in English. However, note that we ported BENGAL to Brazilian Portuguese so as to exemplify that it not biased towards a particular language.

```
: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.
```

Listing 1.1: Example RDF dataset.

Benchmarks. We define a benchmark as a set C of annotated documents D_i. Each document D_i is a sequence of characters $s_{i1} \dots s_{in}$. Each subsequence $s_{ij} \dots s_{ik}$ (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 $s_{ij} \dots s_{ik}$ stands for the resource r. In case the substring $s_{ij} \dots s_{ik}$ does not stand for a resource, we write $m(i, j, k) = \epsilon$. Let D_0 be the example shown in Listing 1.2. We would write $m(D_0, 0, 14) =$ `:AlbertEinstein`.

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

Listing 1.2: Example sentence.

Verbalization. To the best of our knowledge, there are two main works on verbalizing SPARQL[2], i.e., SPARTIQULATION [7] and SPARQL2NL [18]. 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, SPARTIQULATION [7] assumes the structure of the sentence to be generated is described in a template and fits the verbalization of the components into the template. The notation and formal framework for verbalization in BENGAL is also based on SPARQL2NL [18].

Let W be the set of all words in the dictionary of our target language. We define the realization function $\rho : R \cup P \cup L \to W^*$ 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 ρ 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 ρ by the same label ρ for the sake of simplicity.

[1] `http://dbpedia.org`

[2] SPARQL is the query language for RDF data. The specification can be found at `https://www.w3.org/TR/rdf-sparql-query/`.

We adopt a rule-based approach to devise the extension of ρ, where the rules extending ρ 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 $P_1 \wedge \ldots \wedge P_n \Rightarrow K_1 \wedge \ldots \wedge K_m$. The premises and consequences are explicated by using an extension of the Stanford dependencies.[3] 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 $\texttt{poss}(e_1, e_2)$. For the sake of simplicity, we sometimes reduce the construct `subj(y,x)` $\wedge$ `dobj(y,z)` to the triple `(x,y,z)` $\in W^3$.

Table 1: Dependencies (Dep.) used by BENGAL.

Dependency	Explanation
`cc`	Stands for the relation between a conjunct and a given conjunction (in most cases and or 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 $\wedge$, which we use to state that two dependencies hold in the same sentence. For example `subj(EAT,JOHN)` $\wedge$ `dobj(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.

3.2 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

[3] For a complete description of the vocabulary, see `http://nlp.stanford.edu/software/dependencies_manual.pd`

chosen in the initial phase. This process is repeated as many times as necessary to reach the predefined number of documents. In the following, we present the details of each step underlying our benchmark generation process.

Seed Selection Given that we rely on RDF, we model the seed selection by means of a SPARQL SELECT query with one projection variable. Note that we can use the wealth of SPARQL to devise seed selection strategies of arbitrary complexity. However, given that NER and EL frameworks commonly focus on particular classes of resources, we are commonly confronted with the condition that the seeds must be instances of a set of classes, e.g., `:Person`, `:Organization` or `:Place`. The SPARQL query for our example dataset would be as follows:

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

Listing 1.3: Example seed selection query.

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.As these queries (especially the DBPSB[4] and FEASIBLE[5] 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 generated 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 Graphs. 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_{min} and d_{max}, we select d random triples from $L(s_i)$. For the dataset shown in Listing 1.1 and $d = 2$, we would for example get Listing 1.4.

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

Listing 1.4: Example dataset generated by the star strategy.

Symmetric Star Graphs. As above with $t \in \{(s_i, p, o) \in G \vee (o, p, s_i) \in G\}$.

Path Graphs. 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 1.1 and $d = 2$, we would for example get Listing 1.5.

[4] `http://aksw.org/Projects/DBPSB`
[5] `http://aksw.org/Projects/Feasible`

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

Listing 1.5: Example dataset generated by the path strategy.

Hybrid Graphs. 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:

```
:AlbertEinstein :birthPlace :Ulm .
:AlbertEinstein :deathPlace :Princeton .
:Ulm :country :Germany .
```

Listing 1.6: 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-occurence graph for the said class in which the nodes are the properties selected in the preceding step and the co-occurence of two properties p_1 and p_2 is the instance r of the input class where $\exists o_1, o_2 : (r, p_1, o_1) \in K \wedge (r, p_2, o_2) \in K$. The resulting graph is then clustered (e.g., by using the approach presented in [19]). 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 1.4 for our running example.

Verbalization The verbalization strategy for the first four strategies consists of verbalizing each triple as a single sentence and is derived from SPARQL2NL [18]. To verbalize the subject of the triple $t = (s, p, o)$, we use one of its labels according to Ell et al. [6] (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 $t = (s, p, o)$ depends mostly on the verbalization of the predicate `p`. If `p` can be realized as a noun phrase, then a possessive clause can be used to express the semantics of (s, p, o). For example, if p can be verbalized as a nominal compound like `birth place`, then the triple can be verbalized as shown in equation 1. In case `p`'s realization is a verb, then the triple can be verbalized as in equation 2.

$$\rho(s, p, o) \Rightarrow \texttt{poss}(\rho(p), \rho(s)) \wedge \texttt{subj}(\texttt{BE}, \rho(p)) \wedge \texttt{dobj}(\texttt{BE}, \rho(o)) \quad (1)$$

$$\rho(s, p, o) \Rightarrow \texttt{subj}(\texttt{BE}, \rho(p)) \wedge \texttt{dobj}(\texttt{BE}, \rho(o)) \quad (2)$$

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:

$$\begin{aligned}\rho(s,p_1,o_1) \wedge \rho(s,p_2,o_2) \Rightarrow & \texttt{conj}(\texttt{poss}(\rho(p_1),\rho(s)) \wedge \texttt{subj}(\texttt{BE}_1,\rho(p_1)) \\ & \wedge \texttt{dobj}(\texttt{BE}_1,\rho(o_1)) \wedge \texttt{poss}(\rho(p_2),\rho(\texttt{pronoun}(s))) \\ & \wedge \texttt{subj}(\texttt{BE}_2,\rho(p_2)) \wedge \texttt{dobj}(\texttt{BE}_2,\rho(o_2))\end{aligned} \tag{3}$$

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

Paraphrasing With this step, BENGAL avoids the generation of a large number of sentences that share the same terms and the same structure [33]. 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 [1] 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 [10] combined with POS tagging [28]. 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 1.7). We also refrain from changing sentences that describe type information (e.g., see the first sentence Listing 1.7)

```
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 - Wurttemberg, German Empire and Kingdom of
    Wurttemberg.

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 scientists' birth places are
    Ulm, Baden - Wurttemberg, German Empire and Kingdom of
    Wurttemberg.
```

Listing 1.7: Example Paraphasing.

For step (3), BENGAL looks for synonyms of the noun phrases in the sentence using a dictionary (i.e., WordNet[6] 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

[6] https://wordnet.princeton.edu/

multi-word expressions as synonyms. For example, we would not replace the verb `die` by `kick the bucket`. Therewith, we avoid reducing the readability of the sentence. Verb phrases such as `pass away` are however retrieved and used to replace verbs such as `die` (see third sentence in Listing 1.7). The paraphrasing in BENGAL also addresses the replacement of named entities. Here, the approach makes use of alternative surface forms [3] for resources (see third sentence in Listing 1.7). Furthermore, the paraphrasing module replaces pronouns by surface forms (see last sentence in Listing 1.8, where "It" is replaced by the surface form "Pettus") if these pronouns are used very frequently (in our implementation, more than 3 times).

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

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.
```

Listing 1.8: Example Paraphasing at Summary Generation

4 Experiments and Results

We generated 13 datasets in English (B1-B13) and 4 datasets in Brazilian Portuguese (P1-P4) to evaluate our approach.[7] B1 to B10 were generated by running our five subgraph generation methods with and without paraphrasing. The number of documents was set to 100 while (d_{min}, d_{max}) was set to $(1, 5)$. B11 shows how BENGAL can be used to evaluate the scalability of approaches. Here, we used the hybrid generation strategy to generate 10,000 documents. B12 and B13 comprise 10 longer documents each with d_{min} set to 90. For B12, we focused on generating a high number of entities in the documents while B13 contains less entities but the same number of documents.

We compared B1-B13 with the 16 manually created gold standards for English found in GERBIL. 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. In addition, we assessed the performance of multi-lingual NER and EL systems on the datasets P1-P4 to show that BENGAL can be easily ported to languages other than English.

4.1 English Dataset features

The first aim of our evaluation was to quantify the variability of the datasets B1–B13 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 distributions revealed that the manually

[7] The datasets are available at `http://hobbitdata.informatik.uni-leipzig.de/bengal/bengal_datasets.z`

created datasets (D1–D16) have a high correlation (0.88 on average) with a minimum of 0.61 (D10–D16).[8] 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 2 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 1). 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 2 and Figure 1, BENGAL can be easily configured to generate a wide variety of datasets with similar quality and number of documents to those of real datasets. This is mainly due to our approach being able to generate benchmarks ranging from (1) benchmarks with sentences containing a large number of entities without any filler terms (high entity density) to (2) benchmarks which contain more information pertaining to entity types and literals (low entity density).

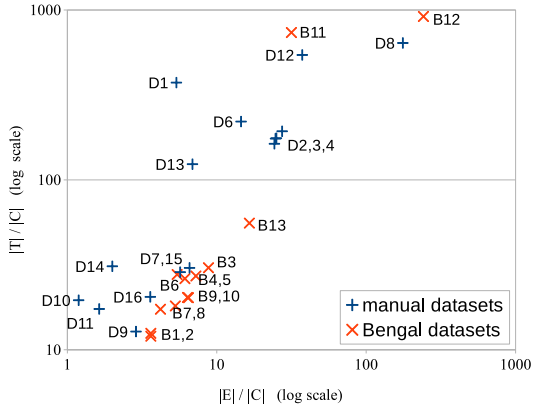

Fig. 1: Average entities and tokens ($|T|$) per document for each dataset.

4.2 Annotator performance

We used GERBIL [31] to evaluate the performance of 11 annotators on the manually created as well as the BENGAL datasets.[9] We evaluated the annotators within an A2KB (annotation to knowledge base) experiment setting: Each document of the corpora was

[8] Our complete results can be found at `https://goo.gl/ZnSgYF`

[9] Complete results: `http://w3id.org/gerbil/experiment?id=201603140002`

Table 2: Excerpt of the features of the datasets used in our evaluation. The datasets B4, B6, B8 and B10 are paraphrased versions of B3, B5, B7 resp. B9 and share similar characteristics.

ID	Name	Doc. $\|C\|$	Tokens $\|T\|$	Entities $\|E\|$	$\|T\|/\|C\|$	$\|E\|/\|C\|$	$\|E\|/\|T\|$
D1	ACE2004	57	21312	306	373.9	5.4	0.01
D2	AIDA/CoNLL-Complete	1393	245008	34929	175.9	25.1	0.14
D8	IITB	104	66531	18308	639.7	176.0	0.28
D11	Microposts2014-Train	2340	40684	3822	17.4	1.6	0.09
D15	OKE 2015 Task 1 evaluation	101	3064	664	30.3	6.6	0.22
B1	BENGAL Path 100	100	1202	362	12.02	3.6	0.30
B2	BENGAL Path Para 100	100	1250	362	12.5	3.6	0.29
B3	BENGAL Star 100	100	3039	880	30.39	8.8	0.29
B5	BENGAL Sym 100	100	2718	725	27.18	7.25	0.26
B9	BENGAL Summary 100	100	2033	637	20.33	6.37	0.31
B11	BENGAL Hybrid 10000	10000	556483	165254	55.6	16.5	0.30
B12	BENGAL Hybrid Long 10	10	9162	2417	241.7	916.2	0.26
B13	BENGAL Star Long 10	10	7369	316	31.6	736.9	0.04

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.

Table 3 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.[10] 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

[10] All values can be found at `https://tinyurl.com/kjre3rh`.

results across BENGAL datasets varies between 0.54 and 1, which further supports that BENGAL can generate a wide range of diverse datasets.

Table 3: 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	AIDA	Babelfy	Spotlight	Dexter	E.eu	FOX	FRED	FREME	WAT	xLisa-NER	xLisa-NGRAM
A2KB	D1	0.26	0.13	0.18	0.21	0.14	0.13	N/A	0.19	0.25	0.36	0.27
	D2	0.68	0.45	0.54	0.47	0.39	0.51	N/A	0.34	0.67	0.43	0.36
	D8	0.14	0.13	0.26	0.21	0.15	0.10	N/A	0.07	0.14	0.07	0.23
	D11	0.38	0.31	0.45	0.39	0.36	0.32	0.07	0.25	0.40	0.36	0.32
	D15	0.57	0.41	0.46	0.47	0.28	0.55	0.33	0.27	0.53	0.53	0.47
	B1	0.65	0.47	0.69	0.70	0.39	0.50	0.45	0.49	0.61	0.45	0.61
	B2	0.67	0.49	0.68	0.70	0.38	0.54	0.41	0.47	0.61	0.44	0.62
	B3	0.62	0.48	0.57	0.65	0.27	0.47	0.35	0.38	0.53	0.36	0.43
	B5	0.42	0.40	0.42	0.44	0.17	0.34	0.29	0.30	0.35	0.24	0.33
	B9	0.51	0.39	0.57	0.52	0.26	0.43	0.39	0.30	0.46	0.44	0.51
	B11	0.68	0.68	0.69	0.74	0.24	0.49	0.41	0.47	0.65	0.44	0.51
	B12	0.83	N/A	0.79	0.84	0.40	0.73	N/A	0.50	0.79	0.23	0.28
	B13	0.33	0.38	0.33	0.40	0.11	0.17	N/A	0.22	0.45	0.44	0.50

4.3 Annotator Performance on Brazilian Portuguese

We implemented BENGAL for Brazilian Portuguese relying on a Portuguese RDF verbalizer [15] and ran four multilingual NER and EL (MAG [16], DBpedia Spotlight, Babelfy, and PBOH [9]) frameworks thereon. In addition, we evaluated the performance of these annotators on subsets of HAREM[11] which is a manually created dataset[12]. While the extension of BENGAL to Portuguese is an important result in itself, our results also provide new insights in the NER and EL performance of existing solutions. Ammongst other, our results suggest that existing solutions are mostly biased towards a high precision but often achieve a lower recall on this language. For example, both Spotlight's and Babelfy's recall remain below 0.6 in most cases while their precision goes up to 0.9. This clearly results from the lack of training data for these resource-poor languages. In future work, we intend to quantify this phenomenon across other resource-poor languages and create datasets to push the development of tools to process these languages.

[11] http://www.linguateca.pt/HAREM/

[12] All Portuguese results can be found at `http://faturl.com/bengalpt`.

4.4 Scalability

An advantage of BENGAL is that it can be used to generate large data corpora. Therewith, BENGAL allows the evaluation of existing systems for 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.[13] We separated the dataset in 5 equal parts that we used for 5 phases of the benchmarking. During the different phases, $1, 2, 4, 8, 2000$ documents/sec 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 2xE5-2630v3 8-Cores (2.4GHz) with 256GB RAM.

Table 4: Runtimes of different NER/EL tools on B11 in seconds.

Phase	FOX	Stanford	Balie	Illinois	Open NLP	Spotlight
1	524.0	1.4	1.9	1.5	1.3	1.3
2	1540.3	1.9	3.3	2.1	1.6	1.6
3	2825.0	5.6	7.8	4.4	4.8	3.7
4	5019.7	149.3	309.0	165.9	101.3	133.0
5	9105.2	700.2	1174.8	747.5	555.1	702.6

Table 4 shows the behavior of six different NER tools in our experiments, which are *the first large-scale runtime evaluation of NER tools*. 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.

5 Discussion and Conclusion

We presented and evaluated BENGAL, an approach for the automatic generation of NER and EL benchmarks. Our results suggest that our approach can generate diverse benchmarks with characteristics similar to those of a large proportion of existing benchmarks in several languages. Importantly, the precautions taken to limit the reverse engineering of BENGAL datasets (which is an obvious weakness of the approach) do not affect

[13] The complete experimental results can be found at `https://goo.gl/9mnbwC`.

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. In future work, we will extend the ability of BENGAL to generate long and complex sentences and increase the amount of adjectives and adverbs in the generated documents.

References

1. Androutsopoulos, I., Malakasiotis, P.: A survey of paraphrasing and textual entailment methods. Journal of Artificial Intelligence Research pp. 135–187 (2010)
2. Brümmer, M., Dojchinovski, M., Hellmann, S.: Dbpedia abstracts: A large-scale, open, multilingual NLP training corpus. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016 (2016)
3. Bryl, V., Bizer, C., Paulheim, H.: Gathering alternative surface forms for dbpedia entities. In: NLP-DBPEDIA@ ISWC. pp. 13–24 (2015)
4. Cano Basave, A.E., Rizzo, G., Varga, A., Rowe, M., Stankovic, M., Dadzie, A.S.: Making sense of microposts (#microposts2014) named entity extraction & linking challenge. In: Proceedings of 4th Workshop on Making Sense of Microposts (2014)
5. Cucerzan, S.: Large-scale named entity disambiguation based on wikipedia data. In: Conference on Empirical Methods in Natural Language Processing-CoNLL (2007)
6. Ell, B., Vrandečić, D., Simperl, E.: Labels in the web of data. ISWC (2011)
7. Ell, B., Vrandečić, D., Simperl, E.: Spartiqulation: Verbalizing sparql queries. In: Extended Semantic Web Conference. pp. 117–131. Springer (2012)
8. van Erp, M., Mendes, P., Paulheim, H., Ilievski, F., Plu, J., Rizzo, G., Waitelonis, J.: Evaluating entity linking: An analysis of current benchmark datasets and a roadmap for doing a better job. In: Proceedings of LREC (2016)
9. Ganea, O.E., Ganea, M., Lucchi, A., Eickhoff, C., Hofmann, T.: Probabilistic bag-of-hyperlinks model for entity linking. In: Proceedings of the 25th International Conference on World Wide Web. pp. 927–938. WWW '16, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland (2016)
10. Gatt, A., Reiter, E.: Simplenlg: A realisation engine for practical applications. In: Proceedings of the 12th European Workshop on Natural Language Generation. pp. 90–93 (2009)
11. Jha, K., Röder, M., Ngomo, A.C.N.: All that glitters is not gold–rule-based curation of reference datasets for named entity recognition and entity linking. In: ISWC (2017)
12. Mendes, P.N., Jakob, M., Garcia-Silva, A., Bizer, C.: DBpedia Spotlight: Shedding Light on the Web of Documents. In: 7th International Conference on Semantic Systems (I-Semantics). pp. 1–8 (2011)
13. Milne, D., Witten, I.H.: Learning to link with wikipedia. In: ACM CIKM (2008)
14. Moro, A., Cecconi, F., Navigli, R.: Multilingual word sense disambiguation and entity linking for everybody. In: Proceedings of the 2014 International Conference on Posters & Demonstrations Track-Volume 1272. pp. 25–28. CEUR-WS. org (2014)
15. Moussallem, D., Ferreira, T.C., Zampieri, M., Cavalcanti, M.C., Xexeo, G., Neves, M., Ngomo, A.C.N.: RDF2PT: Generating Brazilian Portuguese Texts from RDF Data. In: LREC (2018)

16. Moussallem, D., Usbeck, R., Röder, M., Ngonga Ngomo, A.C.: MAG: A Multilingual, Knowledge-base Agnostic and Deterministic Entity Linking Approach. In: K-CAP: Knowledge Capture Conference. p. 8. ACM (2017)
17. Ngomo, A.C.N., Röder, M., Usbeck, R.: Cross-document coreference resolution using latent features. LD4IE'14 (2014)
18. Ngonga Ngomo, A.C., Bühmann, L., Unger, C., Lehmann, J., Gerber., D.: Sorry, i don't speak sparql — translating sparql queries into natural language. In: Proceedings of WWW. pp. 977–988 (2013)
19. Ngonga Ngomo, A.C., Schumacher, F.: Borderflow: A local graph clustering algorithm for natural language processing. In: Computational Linguistics and Intelligent Text Processing, pp. 547–558. Springer (2009)
20. Nuzzolese, A.G., Gentile, A.L., Presutti, V., Gangemi, A., Garigliotti, D., Navigli, R.: Open knowledge extraction challenge. In: Semantic Web Evaluation Challenge (2015)
21. Oramas, S., Anke, L.E., Sordo, M., Saggion, H., Serra, X.: ELMD: an automatically generated entity linking gold standard dataset in the music domain. In: LREC (2016)
22. Ratinov, L., Roth, D., Downey, D., Anderson, M.: Local and global algorithms for disambiguation to wikipedia. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. pp. 1375–1384 (2011)
23. Rebholz-Schuhmann, D., Yepes, A.J.J., Van Mulligen, E.M., Kang, N., Kors, J., Milward, D., Corbett, P., Buyko, E., Beisswanger, E., Hahn, U.: Calbc silver standard corpus. Journal of bioinformatics and computational biology (2010)
24. Röder, M., Usbeck, R., Gerber, D., Hellmann, S., Both, A.: N^3 - A Collection of Datasets for Named Entity Recognition and Disambiguation in the NLP Interchange Format. In: LREC (2014)
25. Speck, R., Ngomo, A.C.N.: Ensemble learning for named entity recognition. In: International Semantic Web Conference. pp. 519–534. Springer (2014)
26. Steinmetz, N., Knuth, M., Sack, H.: Statistical analyses of named entity disambiguation benchmarks. In: 1st Workshop on NLP&DBpedia 2013. pp. 91–102 (2013)
27. Tjong Kim Sang, E.F., De Meulder, F.: 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. pp. 142–147 (2003)
28. Toutanova, K., Manning, C.D.: Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. EMNLP '00 (2000)
29. Tsatsaronis, G., Schroeder, M., Paliouras, G., Almirantis, Y., Androutsopoulos, I., Gaussier, E., Gallinari, P., Artieres, T., Alvers, M., Zschunke, M., Ngonga Ngomo, A.C.: BioASQ: A challenge on large-scale biomedical semantic indexing and question answering. In: AAAI Information Retrieval and Knowledge Discovery in Biomedical Text (2012)
30. Usbeck, R., Ngomo, A.C.N., Röder, M., Gerber, D., Coelho, S.A., Auer, S., Both, A.: AGDISTIS-graph-based disambiguation of named entities using linked data. In: International Semantic Web Conference. Springer (2014)
31. Usbeck, R., Röder, M., Ngonga Ngomo, A.C., Baron, C., Both, A., Brümmer, M., Ceccarelli, D., Cornolti, M., Cherix, D., Eickmann, B., Ferragina, P., Lemke, C., Moro, A., Navigli, R., Piccinno, F., Rizzo, G., Sack, H., Speck, R., Troncy, R., Waitelonis, J., Wesemann, L.: Gerbil: General entity annotator benchmarking framework. In: WWW '15 (2015)
32. Voyer, R., Nygaard, V., Fitzgerald, W., Copperman, H.: A hybrid model for annotating named entity training corpora. In: Proceedings of the 4th Linguistic Annotation Workshop (2010)
33. Young, M., et al.: Technical writer's handbook. University Science Books (2002)
34. Zhang, J., Sivasubramaniam, A., Franke, H., Gautam, N., Zhang, Y., Nagar, S.: Synthesizing representative i/o workloads for tpc-h. In: 10th HPCA. IEEE Computer Society, Washington, DC, USA (2004)