

A Framework for Benchmarking Entity-Annotation Systems

Marco Cornolti
Dipartimento di Informatica
University of Pisa, Italy
cornolti@di.unipi.it

Paolo Ferragina
Dipartimento di Informatica
University of Pisa, Italy
ferragina@di.unipi.it

Massimiliano Ciaramita
Google Research
Zürich, Switzerland
massi@google.com

ABSTRACT

In this paper we design and implement a benchmarking framework for fair and exhaustive comparison of entity-annotation systems. The framework is based upon the definition of a set of problems related to the entity-annotation task, a set of measures to evaluate systems performance, and a systematic comparative evaluation involving all publicly available datasets, containing texts of various types such as news, tweets and Web pages. Our framework is easily-extensible with novel entity annotators, datasets and evaluation measures for comparing systems, and it has been released to the public as open source¹. We use this framework to perform the first extensive comparison among all available entity annotators over all available datasets, and draw many interesting conclusions upon their efficiency and effectiveness. We also draw conclusions between academic versus commercial annotators.

Categories and Subject Descriptors

D.2.8 [Software Engineering]: Metrics—*performance measures*; I.2.7 [Artificial Intelligence]: Natural Language Processing—*text analysis*

Keywords

Benchmark Framework; Entity annotation; Wikipedia

1. INTRODUCTION

Classic approaches to document indexing, clustering, classification and retrieval are based on the bag-of-words paradigm. The limitations of this paradigm are well-known to the IR community and in recent years a good deal of work has attempted to move beyond by “grounding” the processed texts with respect to an adequate semantic representation, by designing so-called *entity annotators*. The key idea is to identify, in the input text, short-and-meaningful sequences of terms (also called *mentions*) and annotate them with unambiguous identifiers (also called *entities*) drawn from a catalog. Most recent work adopts anchor texts occurring in Wikipedia as entity mentions and the respective Wikipedia pages as the mentioned entity, because Wikipedia offers today the best trade-off between catalogs with a rigorous struc-

ture but low coverage (such as WordNet, CYC, TAP), and a large text collection with wide coverage but unstructured and noisy content (like the whole Web). The process of entity annotation involves three main steps: (1) *parsing* of the input text, which is the task to detect candidate entity mentions and link each of them to *all possible* entities they could mention; (2) *disambiguation* of mentions, which is the task of selecting the most pertinent Wikipedia page (i.e., entity) that best describes each mention; (3) *pruning* of a mention, which discards a detected mention and its annotated entity if they are considered not interesting or pertinent to the *semantic interpretation* of the input text.

The focus around entity annotators has increased significantly in the last few years, with several interesting and effective algorithmic approaches to solve the mention-entity match problem, possibly using other knowledge bases such as DBpedia, Freebase or Yago (see e.g. [2, 3, 7, 9, 13, 16, 18, 19, 23, 5, 15]). Unfortunately, the research has investigated some specific tasks using non-uniform terminology, non-comparable evaluation metrics, and limited datasets and systems. As a consequence, we only have a partial picture of the efficiency and effectiveness of known annotators which makes it difficult to compare them in a fair and complete way. This is a particularly important issue because those systems are being used as black-boxes by more and more IR tools, built on top of them, such as [1, 8, 21, 22]. Motivated by these considerations, [20] recently attempted to compare entity-annotation systems mainly coming from the commercial realm². However, as the authors state in the concluding section of their paper, their evaluation is limited to strict metrics which account for “exact” matches over mentions and entities (and, rather observe that “a NE type might be not wrong but not precise enough.”), it does not consider datasets fully annotated by humans (i.e. mentions are only the ones derived by few parsers), and misses to consider the best performing tools which have been published recently in the scientific literature. This last issue is a crucial limitation because, as [12] showed recently, the DBpedia Spotlight system (the best according to [20]) achieves much worse performance than some of the systems tested in this paper.

Given this scenario, we aim with this paper at defining and implementing a framework for comparing in a complete, fair and meaningful way the most efficient, effective and *publicly available* entity-annotation systems: namely, AIDA [7], Illinois Wikifier [19], TagMe [3], Wikipedia-miner [16], and

¹See <http://acube.di.unipi.it/>

²AlchemyAPI, DBpedia Spotlight, Evri, Extractiv, Lupe-
dia, OpenCalais, saplo, Wikimeta, Yahoo! Content Analysis
and Zemanta.

Problem	Input	Output	Description
Disambiguate to Wikipedia (D2W)	Text, Set of mentions	Set of relevant annotations	Assign to each input mention its pertinent entity (possibly <i>null</i>). This problem has been introduced in [2].
Annotate to Wikipedia (A2W)	Text	Set of relevant annotations	Identify the relevant mentions in the input text and assign to each of them the pertinent entities. This problem has been introduced in [16].
Scored-annotate to Wikipedia (Sa2W)	Text	Set of relevant and scored annotations	As A2W, but here each annotation is assigned a score representing the likelihood that the annotation is correct. This problem has been introduced in [16].
Concepts to Wikipedia (C2W)	Text	Set of relevant tags	Tags are taken as the set of relevant entities that are mentioned in the input text. This problem has been defined in [13].
Scored concepts to Wikipedia (Sc2W)	Text	Set of relevant and scored tags.	As C2W, but here each tag is assigned a score representing the likelihood that the annotation is correct.
Ranked-concepts to Wikipedia (Rc2W)	Text	Ranked list of relevant tags	Identify the entities mentioned in a text and rank them in terms of their relevance for the topics dealt with in the input text. This problem has been defined in [13].

Table 1: A set of entity-annotation problems.

DBpedia Spotlight; which currently define the state-of-the-art for the entity-annotation task. In order to achieve this goal, we will introduce (1) a hierarchy of entity-annotation problems, that cover the wide spectrum of annotation goals such systems could address; (2) a set of novel measures to finely evaluate the effectiveness of these systems; (3) all public datasets available for the entity-annotation task, allowing us to explore the efficiency and effectiveness performance of the annotation process according to the introduced measures and hierarchy of problems.

We have made this framework publicly available (<http://acube.di.unipi.it/>), with all datasets and software used in this paper, to make our experiments reproducible and provide a common ground for developing new and better solutions for this challenging problem.

2. BACKGROUND AND TERMINOLOGY

We use the following terminology:

- An *entity* (or *concept*, *topic*) is a Wikipedia article which is uniquely identified by its page-ID.
- A *mention* is the occurrence of a sequence of terms located in a text. It is encoded by the integer pair $\langle p, l \rangle$, where p is the position of the occurrence and l is the length of the mention.
- An *annotation* is the linking of a mention to an entity. It is encoded by the pair $\langle m, e \rangle$ where $m = \langle p, l \rangle$ is the mention and e is the page-ID of the mentioned entity.
- A *tag* is the annotation of a text with an entity which captures a topic (explicitly mentioned) in the input text. (Thus, a tag comes with no mention.)
- A *score* is a real value $s \in [0, 1]$ that is assigned to an annotation or a tag to indicate its correctness.

As an example, consider the following text fragment: “Obama issues Iran ultimatum”. “Obama”, “Iran” and “ultimatum” are mentions which occur as anchors in Wikipedia and can thus be linked with the entities represented by the Wikipedia pages of Barack Obama, the nation of Iran and the threat to declare war, respectively.

Figure 1 and Table 1 exemplify and formalize, respectively, a set of problems which cover the wide spectrum of goals the entity-annotation systems could be asked to solve.

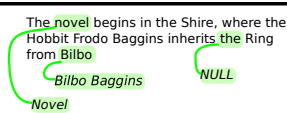
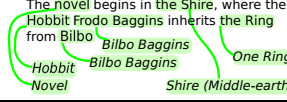
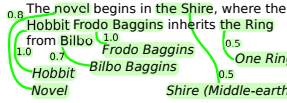
Problem	Input	Output
D2W	The novel begins in the Shire, where the Hobbit Frodo Baggins inherits the Ring from Bilbo	The novel begins in the Shire, where the Hobbit Frodo Baggins inherits the Ring from Bilbo 
A2W	The novel begins in the Shire, where the Hobbit Frodo Baggins inherits the Ring from Bilbo	The novel begins in the Shire, where the Hobbit Frodo Baggins inherits the Ring from Bilbo 
Sa2W	The novel begins in the Shire, where the Hobbit Frodo Baggins inherits the Ring from Bilbo	The novel begins in the Shire, where the Hobbit Frodo Baggins inherits the Ring from Bilbo 
C2W	The novel begins in the Shire, where the Hobbit Frodo Baggins inherits the Ring from Bilbo	{Novel, Shire (Middle-earth), Hobbit, Frodo Baggins, One Ring, Bilbo Baggins}
Sc2W	The novel begins in the Shire, where the Hobbit Frodo Baggins inherits the Ring from Bilbo	{(Novel, 0.8), (Shire (Middle-earth), 0.5), (Hobbit, 1.0), (Frodo Baggins, 1.0), (One Ring, 0.5), (Bilbo Baggins, 0.8)}
Rc2W	The novel begins in the Shire, where the Hobbit Frodo Baggins inherits the Ring from Bilbo	1. Hobbit 2. Frodo Baggins 3. Novel 4. Bilbo Baggins 5. One Ring 6. Shire (Middle-earth)

Figure 1: Some instances of annotation tasks. Numbers in problems Sa2W and Sc2W denote the correctness likelihood of each annotation/tag.

Some of these problems have been introduced in the literature, others are introduced in this paper as reasonable variations which complete the picture. This will allow us to contextualize and compare the known systems, and assess their relations and limitations. These problems can be casted in two main classes: the first consists of three problems which address the identification of (possibly scored) annotations, and thus the identification of mention-entity pairs; the second consists of three further problems that involve finding tags (possibly scored or ranked), thus accounting only for the entities.

Given that these annotation problems are strictly related to each other, it is possible to introduce a few simple *reductions* that induce a *hierarchy* among them. Such a hierarchy, in the form of a DAG, will ease the definition of our framework and the set-up of the experiments for comparing the annotators. We adopt the notation $A \prec B$ to indicate that

problem A reduces to problem B , meaning that an instance I_A of A can be transformed efficiently into an instance I_B of B in such a way that, a solution S_B for I_B can be used to determine a solution S_A of A . In this context “efficient” means linear time (in the instance size), so that the cost of the reduction does not impact onto the cost of the indirect solution of the problem A via B . This way B is (computationally) harder than A , since an algorithm that efficiently solves B also solves A efficiently.

As an example, take the two problems $C2W \prec A2W$, the former asks only for the entities, the latter asks for the pairs entity-mentions (see Table 1). It is clear that a solution for $A2W$ can be adapted to a solution for $C2W$ by discarding the mentions and leaving just the retrieved entities. The input to both problems is the same, and thus the input to $C2W$ does not even need any adaptation to fit the problem $A2W$. In a similar vein it is easy to derive the reductions presented in Figure 2, where arrows go from harder to easier problems. At this stage, the reader may ignore the names of the systems (in purple) and the name of the datasets (in yellow) indicated in the boxes. The most general and difficult problem is $Sa2W$, because all other problems reduce to it; problems $D2W$ and $C2W$ are the most specific ones. $C2W$ is introduced to provide a common ground over which it is possible to compare systems designed to solve $Sc2W$ and $A2W$.

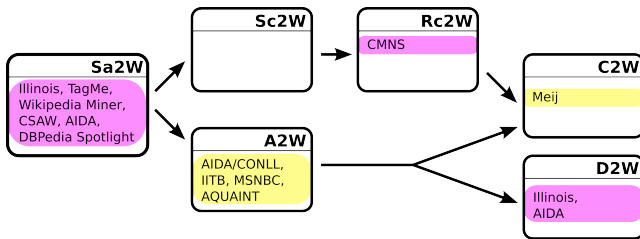


Figure 2: DAG of reductions among entity-annotation problems. Arrows go from harder to easier problems. For each problem (box), we show the systems that solve it (in purple) and known datasets used as ground truth (in yellow).

The DAG in Figure 2 allows us to identify some problems on which entity-annotation systems can be tested in a complete and significant way, without jeopardizing their algorithmic nature. In fact, the DAG can be used to derive gold-standards for the problems as follows: if $\hat{P} \prec P$, then a dataset providing a gold-standard for P can be adapted to be a gold standard for the easier problem \hat{P} . As a result, given two systems S' and S'' which respectively solve problems P' and P'' , we can compare them over the gold standard for problem P , as long as P is reachable in the DAG from both P' , P'' . Since these reductions are simple and efficient, runtime measures among systems will be accurate, and can be inferred for all problems in the hierarchy.

3. ENTITY-ANNOTATION SYSTEMS

Table 2 lists the most effective annotators published in the literature. Unfortunately, only five of them are publicly available either through a Web access or downloading the software; the authors of CMNS and CSAW communicated to us that these annotators are not yet available to the pub-

lic. For each of those systems we also report the problems they have addressed natively, and briefly summarize their underlying algorithmic techniques.

Name	Authors Affiliation	Addressed problem	References
AIDA	Max Planck Institute for Informatics	Sa2W, D2W	[23, 7]
CMNS	University of Amsterdam	C2W	[13] (software not yet available)
CSAW	Indian Institute of Technology Bombay	Sa2W	[9] (software not yet available)
Illinois Wikifier	University of Illinois at Urbana-Champaign	Sa2W, D2W	[19]
DBpedia Spotlight	Freie Universitaet Berlin	Sa2W	[14]
TagMe	University of Pisa	Sa2W	[3]
Wikipedia Miner	University of Waikato	Sa2W	[16]

Table 2: Best known entity-annotation systems.

AIDA searches for mentions using the Stanford NER Tagger and adopts the YAGO2 knowledge base [6] as catalog of entities, including their semantic distance. Disambiguation comes in three variants: PriorOnly (each mention is bound to its most-commonly linked entity in the knowledge base), LocalDisambiguation (each mention is disambiguated independently from others, according to a set of features which describe the mention and the entities), CocktailParty (YAGO2 is used to perform a collective disambiguation which aims at maximizing the coherence among the selected annotations, via an iterative graph-based approach). AIDA has been designed to deal with English documents of arbitrary length, it offers a publicly available API (as of July 2012).³

CMNS generates a ranked list of candidate entities for all N-grams in the input text. The list is created through lexical matching and language modeling. The disambiguation is done with a method based on supervised machine learning that takes as input a set of (short) texts and, for each of them, a set of human-annotations. CMNS has been designed to deal with very-short texts only (mainly tweets). Currently, this system can not be accessed or downloaded.⁴

CSAW searches the input text for mentions extracted from Wikipedia anchors and titles. This system introduced two novelties with respect to the previous [2, 16]: it used two scores for each annotation, one local and one global. The local score involves 12 features built upon the terms around the mention and the candidate entities. The global score involves all the other annotations detected for the input text and averages the relatedness among them. This was the first system to formulate the disambiguation problem as a quadratic-programming optimization problem aiming for a global coherence among all mentions. CSAW has been designed to deal with English documents of arbitrary

³<http://www.mpi-inf.mpg.de/yago-naga/aida/>.

⁴Personal communication by E. Meij.

length, but is quite slow because of the quadratic-programming approach. Currently, this system can not be accessed or downloaded.⁵

Illinois Wikifier searches the input text for mentions extracted from Wikipedia anchors and titles, using the Illinois NER system [18]. Disambiguation is formulated as an optimization problem which aims at global coherence among all mentions. It uses a novel relatedness measure between Wikipedia pages based on NGD (Normalized Google similarity distance) and pointwise mutual information. Wikifier has been designed to deal with English documents of arbitrary length, its software can be downloaded (as of August 2012).⁶

DBpedia Spotlight searches the input text for mentions extracted from Wikipedia anchors, titles and redirects; the parser is the LingPipe Exact Dictionary-Based Chunker. It then associates a set of candidate entities to each mention using the DBpedia Lexicalization dataset. Given a spotted mention and a set of candidate entities, both the context of the mention and all contexts of each entity are cast to a Vector-Space Model (using a BOW approach) and the candidate whose context has the highest cosine similarity is chosen. Note that no semantic coherence is estimated among the chosen entities. Spotlight has been designed to deal with English documents of arbitrary length, it offers a publicly available API (as of November 2012).⁷

TagMe 2 searches the input text for mentions defined by the set of Wikipedia page titles, anchors and redirects. Each mention is associated with a set of candidate entities. Disambiguation exploits the structure of the Wikipedia graph, according to the *relatedness measure* introduced in [17] which takes into account the amount of common incoming links between two pages. TagMe’s disambiguation is enriched with a *voting scheme* in which all possible bindings between mentions and entities are scored and then they express a vote for each other binding. A proper mix of heuristics is eventually adopted to select the best annotation for each mention. TagMe has been designed to deal with short texts, it offers a publicly available API.⁸

Wikipedia Miner is an implementation of the Wikification algorithm presented in [16, 11], one of the first approaches proposed to solve the entity-annotation problem. This system is based on a machine-learning approach that is trained with links and contents taken from Wikipedia pages. Three features are then used to train a classifier that selects valid annotations discarding irrelevant ones: (i) the *prior probability* that a mention refers to a specific entity, (ii) the *relatedness* with the context from which the entity is extracted, given by the non-ambiguous spotted mentions, and (iii) the *context quality* which takes into account the number of terms involved, the extent they relate to each other, and how often they are used as Wikipedia

links. Wikipedia-Miner has been designed to deal with English documents of arbitrary length, it offers a publicly available API (as of September 2012).⁹

Dataset	Gold Standard	Num Docs	Avg length	Num Anns	Avg Anns/Doc
AIDA/CONLL	A2W	231	1039	4485	19.4
AQUAINT	A2W	50	1415	727	14.5
MSNBC	A2W	20	3316	658	32.9
IITB	A2W	103	3879	11245	109.1
Meij	C2W	502	80	812	1.6

Table 3: *Gold standard* indicates the type of the dataset. *Num Docs* is the number of documents in the dataset, *Avg length* is their average length, expressed in characters. *Num Anns* is the total number of annotations or tags. *Avg Anns/Doc* is the average number of annotations or tags per document.

4. DATASETS

For our experiments we collected all publicly available datasets that have been evaluated in the literature and we used them to test all reviewed entity-annotation systems. Details about these datasets are given below and in Table 3. We observe that they range from news to tweets, up to Web pages, providing a wide spectrum of text sources.

AIDA/CoNLL builds on the CoNLL 2003 entity-recognition task. Documents are news taken from Reuters Corpus V1. A large subset of mentions referring to named entities are annotated, but the common names are not. Entities are annotated at each occurrence of a mention. The dataset was introduced in [7]. This dataset is divided into three chunks: Training, Test A and Test B. Since the AIDA system has been tuned over the first two chunks, we will use only Test B, made up of 231 documents, for our experiments.

AQUAINT contains a subset of the original AQUAINT corpus, consisting of English newswire texts. Not all occurrences of the mentions are annotated: only the first mention of each entity, and only the most important entities are retained. This reflects the Wikipedia-style linking. The dataset was introduced in [16].

IITB contains over a hundred of manually annotated texts drawn from popular Web pages about sport, entertainment, science and technology, and health [9]. This is the most detailed dataset since almost all mentions, including those whose related concepts are not highly relevant, are annotated.

Meij contains tweets annotated with all occurring entities. Tweets are poorly composed and very short, less than 140 chars. The dataset was introduced in [13].

MSNBC contains newswire text from MSNBC news network. It annotates only important entities and their referring mentions. The dataset was introduced in [2].

Three datasets are from news but they have different characteristics that will be useful to provide a full comparison among the tested systems. Some of those datasets include

⁹<http://wikipedia-miner.cms.waikato.ac.nz/>.

⁵Although the software for CMNS and CSAW are not available, we used their datasets to compare the other annotators.

⁶<http://www.mpi-inf.mpg.de/yago-naga/aida/>.

⁷<http://spotlight.dbpedia.org/rest/annotate>.

⁸<http://acube.di.unipi.it/tagme>. Its release 2.0 has been provided in August 2012.

annotations to Wikipedia pages that no longer exist in the current version of Wikipedia. This may happen whenever a page has been deleted or renamed. In our experiments we discarded annotations to no-longer existing entities.

5. EVALUATION MEASURES

The definition of a *correct match* between two annotations is challenging because it involves two possible *dimensions*: the “syntactic” one of the mentions, and the “semantic” one of the entities. A match in each dimension can be more or less “correct” depending on the nature of the match itself. As an example, let us consider again the text “President Barack Obama issues Iran ultimatum”. A system might detect the mention “President Barack Obama” whereas the ground-truth identifies as the true mention “Barack Obama”. Also, a system could predict for the corresponding entity the personal page of Barack Obama whereas the annotators could have preferred the page of the President of the United States – which also contains information about Barack Obama – or vice versa. It seems intuitive that simply counting these as errors is a sub-optimal choice as it transfers the human-annotators bias into the evaluation. Furthermore, factoring out such biases from the evaluation metrics could also alleviate the dangers of overfitting on small data samples that only offer a glimpse of the complexity of the total entity space. Such issues arise more frequently than one would expect given that, often, the human-labeled instances in the available datasets offer a wide spectrum of entity/mention possibilities, all of them pertinent to some extent. To overcome these problems we introduce novel *fuzzy measures* to evaluate entity/mention matching which account for slight, but non-significant, misalignments in the detected mentions, as well as for different, but yet pertinent, entities.

In practice, we propose to generalize standard evaluation measures such as true/false positives, true/false negatives, precision and recall by defining them on top of a binary relation M which specifies the notion of “correct match” between either two tags or two annotations. Given an input text t , let g be the correct items (being them mentions or entities or full annotations) specified in the ground truth for t and let s be the solution found by the tested system. We can introduce the following definitions:

$$\begin{aligned} tp(s, g, M) &= \{x \in s \mid \exists x' \in g : M(x', x)\} \\ fp(s, g, M) &= \{x \in s \mid \nexists x' \in g : M(x', x)\} \\ tn(s, g, M) &= \{x \notin s \mid \exists x' \in g : M(x', x)\} \\ fn(s, g, M) &= \{x \in g \mid \nexists x' \in s : M(x', x)\} \end{aligned} \quad (1)$$

Based on these we can generalize the definitions of precision, recall and F1 [10] and their (*micro*- and *macro*-) dataset-wise versions. The *macro*-measures are the average of the corresponding measure over each document in the dataset D , while the *micro*-measures take into account all annotations together thus giving more importance to documents with more annotations. Here follow the *micro*- and *macro*-measures, where we use s_t and g_t to denote, respectively, the solution found by an annotator and the gold standard for a document $t \in D$.

$$\begin{aligned} P(s, g, M) &= \frac{|tp(s, g, M)|}{|tp(s, g, M)| + |fp(s, g, M)|} \\ R(s, g, M) &= \frac{|tp(s, g, M)|}{|tp(s, g, M)| + |fn(s, g, M)|} \\ F1(s, g, M) &= \frac{2 \cdot P(s, g, M) \cdot R(s, g, M)}{P(s, g, M) + R(s, g, M)} \end{aligned} \quad (2)$$

$$\begin{aligned} P_{mic}(S, G, M) &= \frac{\sum_{t \in D} |tp(s_t, g_t, M)|}{\sum_{t \in D} (|tp(s_t, g_t, M)| + |fp(s_t, g_t, M)|)} \\ R_{mic}(S, G, M) &= \frac{\sum_{t \in D} |tp(s_t, g_t, M)|}{\sum_{t \in D} (|tp(s_t, g_t, M)| + |fn(s_t, g_t, M)|)} \\ F1_{mic}(S, G, M) &= \frac{2 \cdot P_{mic}(S, G, M) \cdot R_{mic}(S, G, M)}{P_{mic}(S, G, M) + R_{mic}(S, G, M)} \\ P_{mac}(S, G, M) &= \frac{\sum_{t \in D} P(s_t, g_t, M)}{|D|} \\ R_{mac}(S, G, M) &= \frac{\sum_{t \in D} R(s_t, g_t, M)}{|D|} \\ F1_{mac}(S, G, M) &= \frac{2 \cdot P_{mac}(S, G, M) \cdot R_{mac}(S, G, M)}{P_{mac}(S, G, M) + R_{mac}(S, G, M)} \end{aligned} \quad (3)$$

5.1 Two-side measures

Here we consider appropriate matching relations M which deal with both mentions and annotations, thus addressing the case in which the entity-annotation problem accounts for both of them.

M_e for the C2W problem. It is quite straightforward to devise a proper M for C2W, given that it considers only the match between entities. We start from the observation that pages of Wikipedia can be divided into two distinct subsets: R be the set of redirect pages, nR be the set of non-redirect pages (thus $R \cap nR = \emptyset$). For example “Obama” and “Barrack Hussein Obama” are redirects to “Barack Obama”. Redirects are meant to ease the finding of pages by the Wikipedia users. Redirects can be seen as many-to-one bindings from all synonyms (pages in R) to the most common form of the same entity (page in nR). Since some annotators’ output, as well as some datasets, may contain annotations to redirect pages, we have to consider the de-referenced concept. Let $d : R \cup nR \mapsto nR$ be a *dereference* function that implements the many-to-one binding into nR . We say that there exists a *strong entity match* M_e between two entities e_1 and e_2 iff $d(e_1) = d(e_2)$. M_e is reflexive, symmetric, and transitive.

M_a for the D2W problem. The output for D2W consists of a set of annotations, some of them involving a *null* entity. Thus the match relation must deal with mention-entity pairs. Say we have to compute $M_a(a_1, a_2)$ where $a_1 = \langle \langle p_1, l_1 \rangle, e_1 \rangle$ and $a_2 = \langle \langle p_2, l_2 \rangle, e_2 \rangle$. We define the *strong annotation match* M_a as a binary relation that is verified iff $p_1 = p_2$ and $l_1 = l_2$ and $d(e_1) = d(e_2)$, where d is the de-reference function specified before. Note that, though in D2W the mentions to be annotated are given in the input, there can be false-negative annotations, namely those that are assigned to a *null* concept.

M_w for the A2W problem. Like D2W, the output of the A2W problem is a set of annotations; unlike D2W, A2W requires the systems to discover the corresponding mentions. So the *strong annotation match* proposed for D2W would be too strict, imposing a perfect text-alignment between the mentions specified in the ground-truth and the mentions outputted by the tested system. Here it is crucial to be *fuzzy*, yet meaningful, in the counted matches so we introduce the notion of *weak annotation match* $M_w(a_1, a_2)$ which is true iff the mentions (textually) overlap¹⁰ over the input text and $d(e_1) = d(e_2)$. M_w is reflexive and symmetric, but it is not transitive nor anti-symmetric.

¹⁰Let $e_1 = p_1 + l_1 - 1$ and $e_2 = p_2 + l_2 - 1$ be the last character index of the two mentions. Two mentions overlap iff $p_1 \leq p_2 \leq e_1 \vee p_1 \leq e_2 \leq e_1 \vee p_2 \leq p_1 \leq e_2 \vee p_2 \leq e_1 \leq e_2$.

M_e for the Rc2W problem. As pointed out in [13], since the output of Rc2W is a ranking of entities, common metrics like P1, R-prec, Recall, MRR and MAP [10] should be used only after M_e is computed.

M for the Sc2W and Sa2W problems. These problems ask annotation systems to produce a likelihood score for each annotation. In practice, it does not make much sense to compare the likelihood scores (reals) between the ground truth and the system’s annotation, and indeed the literature does not offer such datasets. Nonetheless we have systems that solve Sa2W, so in order to evaluate them, we adapt their output to some easier problems in the DAG-hierarchy by introducing a threshold on the annotation scores. After that, it is enough to set $M = M_e$ for Sc2W and $M \in \{M_w, M_a\}$ for Sa2W, and then compute the generalized metrics discussed at the beginning of this section and detailed in Section 6.

5.2 One-side measures

The measures presented above for Sa2W and A2W quantify the ability of a system to find the correct annotation, which includes finding both the correct mention *and* the correct entity. Here we consider the two measures M_e and the new M_m to dissect annotators by testing in depth two main features: their ability to recognize the mentions or their ability to disambiguate them. In particular, we will address two questions.

1. How much of the error is determined by the lack of a correct mention recognition? To answer this question, we introduce a *mention annotation match* relation M_m that only checks the overlap of mentions, ignoring its associated entity. This will be useful to evaluate the precision of the parsers used by the systems to detect the mentions.
2. How much of the error is determined by a wrong association to an entity? That’s exactly what is evaluated by the metric M_e introduced for problem C2W. This evaluation takes into account only the entities, and not their mentions, thus it is crucial in all applications interested in retrieving the concepts (tag/entity) a text is about.

Finally, although the entity-matching relation considers the weak dereference function d , we will complete our experimental evaluations by considering a finer measure which will account for the “closeness” of entities in Wikipedia based on the Milne-Witten relatedness function [16] (see Figure 5).

5.3 Similarity between systems

The similarity between systems can be measured document-wise, by comparing annotations over single documents, or dataset-wise, by comparing annotations over all documents in a dataset. This comparison can help asses in which phase of the annotation process systems behave differently, providing insights that could help the design of new annotation approaches. In the former case, given two sets of annotations/tags A_t and B_t for an input text t and a matching relation M , we define the similarity between the systems that produced them as:

$$Sim(A_t, B_t, M) = \frac{|\{x \in A_t \mid \exists y \in B_t : M(x, y)\}| + |\{x \in B_t \mid \exists y \in A_t : M(x, y)\}|}{|A_t| + |B_t|} \quad (4)$$

This measure is inspired by the Jaccard coefficient [10], but takes into account the fuzziness of the matching relation M . Unlike the Jaccard measure, triangle inequality does not hold. Note that all matching relations introduced above are reflexive and symmetric, thus Sim is symmetric too and $Sim(x, x, M) = 1$. Sim ranges in $[0, 1]$.

In order to measure the similarity of two annotators over a dataset (call it D), we apply iteratively Sim over all documents and then “average” the values. The average can either give the same importance to all annotations/documents regardless of their size (S_{macro}) or can be computed as the overall “intersection” divided by the overall size, giving more importance to bigger sets (S_{micro}). Formally:

$$S_{macro}(D, M) = \frac{1}{|D|} \cdot \sum_{t \in D} Sim(A_t, B_t, M)$$

$$S_{micro}(D, M) = \frac{\sum_{t \in D} (|\{x \in A_t \mid \exists y \in B_t : M(x, y)\}| + |\{x \in B_t \mid \exists y \in A_t : M(x, y)\}|)}{\sum_{t \in D} (|A_t| + |B_t|)} \quad (5)$$

The actual interpretation of this measures depends only on the chosen relation M . For the tag-based problems (C2W, Rc2W, Sc2W), the only available matching function is M_e , thus Sim measures the fraction of commonly annotated entities. For all other problems (Sa2W, A2W, D2W) the use of M_a gives the fraction of common annotations (having same concept and same mention); using M_w gives the fraction of common overlapping annotations (having same concept and overlapping mention); using M_m gives the fraction of common overlapping mentions.

6. EXPERIMENTAL RESULTS

We ran a *robustness* test for the five entity-annotation systems listed in Table 2 over the five (publicly available) datasets described in Section 4. The annotators were tested as they are, without any fine-tuning or training over the datasets, using the most up-to-date APIs or software made available by the authors of the tested systems, as explained in Section 3. The following sections present our findings in terms of effectiveness (precision, recall and F1) and efficiency (speed) of the annotation process by means of the evaluation measures introduced above. We observe that the datasets AIDA-CoNNL and AQUAINT annotate only a subset of the mentions (See Section 4), therefore entity annotators are penalized, as they found many reasonable annotations that are not contained in the (limited) gold standard, and this explains the not much large F1 figures achieved in the experiments.



Figure 3: DAG of problems involved in our experiments.

Figure 3 shows the hierarchy of problems restricted to those involved in our experiments: we notice that all tested systems solve Sa2W natively, the three news datasets and the Web-page dataset provide a gold standard for A2W, whereas the Twitter dataset provides a gold standard for C2W. Thus, when evaluating over news and Web-pages, we

	Type	Datasets	Problem	Match-relations
<i>Experiment 1</i>	News	AIDA/CONLL AQUAINT MSNBC	A2W	M_a M_w M_m M_e
<i>Experiment 2</i>	Tweets	Meij	C2W	M_t
<i>Experiment 3</i>	Web pages	IITB	A2W	M_a M_w M_m M_e

Table 4: Experimental setup.

will adapt the systems output to solve A2W, by discarding annotations using a threshold; whereas when evaluating for tweets, we will adapt systems to solve C2W, by applying a threshold on the annotations score and by discarding all mentions. Table 4 summarizes the setup.

6.1 Experiment 1: News

In this experiment the evaluation focuses on the A2W problem, the hardest for which datasets are available, according to the reductions of Figure 2. To gain a deeper understanding of systems performance, we used all three news datasets, because of their differences (see Table 3).

Since the reduction from Sa2W (the problem all annotators can solve) to A2W needs a threshold on the likelihood score, our experiments will take into account all possible values for that threshold, ranging from 0 to 1. We run three types of experiments to test the effectiveness of the systems in detecting correct (full) annotations, or just the mentions (discarding the entities), or just the entities (discarding the mentions).

6.1.1 Finding the annotations

We computed all evaluation measures presented in Section 5. For lack of space, we show in Figure 4 the plot of micro-F1 using the weak-annotation match M_w over the AIDA/CONLL dataset (the largest with 231 news and 4485 annotations). The interesting data is given by the maximum value reached by $F1_{micro}$, varying $t \in [0, 1]$. Each annotator gives a different meaning to the annotation-scores, thus it does not make sense to compare the performance of two annotators for the same value of t . Hence that maximum can be seen as the best performance achievable by each system (according to micro-F1). Overall TagMe reaches maximum F1, followed by Illinois Wikifier, Wikipedia miner, AIDA in its three variants and the poorest performance is achieved by DBpedia Spotlight. Detailed results for all news datasets are reported in Table 5.¹¹

AQUAINT and MSNBC have longer documents. Surprisingly TagMe gives the best results, even if it was designed to annotate short texts [3, 4]. For AQUAINT, Wikipedia Miner is the second best followed by Illinois Wikifier, DBpedia Spotlight and then AIDA. For MSNBC, the dataset with the longest documents, the second best annotator is AIDA with the CocktailParty and Local algorithms, followed by Wikipedia Miner, Illinois Wikifier and DBpedia Spotlight.

We also compute the threshold that maximizes $F1_{micro}$ for each system and for each dataset using the *strong annotation match* M_a .

Table 6 reports results only for AIDA/CONLL for lack of space, the other two news datasets generate consistent results. Measurements based on M_a give slightly worse results

¹¹We will provide full tables of results for all experiments in a web appendix with the final version of paper.

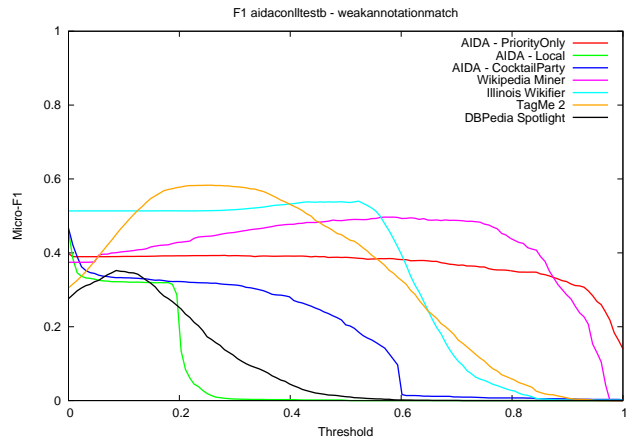


Figure 4: Micro-F1 measure for Sa2W systems on the dataset AIDA/CONLL, varying $t \in [0, 1]$.

than those based on M_w , especially for annotators such as TagMe, Illinois Wikifier and Wikipedia Miner. This shows that these systems detect mentions which do not coincide with those in the gold standard, even though they overlap with the human-labeled ones.

Overall, TagMe is the best annotator in terms of micro-F1, followed by Illinois Wikifier and Wikipedia Miner. This suggests which algorithm employed by these annotators is the most performing: TagMe’s voting scheme, based on the relatedness measure introduced in [16]. Searching for the best global coherence as done by Illinois Wikifier also seems a good approach. Next comes Wikipedia Miner’s algorithm, using machine-learning on Wikipedia’s links. Among the three versions of AIDA, the best is CocktailParty, followed by Local and then PriorityOnly. AIDA seems to consistently trade Recall for Precision (see also Section 6.1.2). Moreover, its performance is affected by the poor mention recognition, as inspected further.

Of course it is hard to judge an algorithm by the performance of the system implementing it, since many variables contribute: implementation (bugs, optimizations), configuration, the Wikipedia version it is based upon, etc. Consequently, our results show which system (according to its configuration available to the public) performs best, and so they give only a clue of the pro/cons of its underlying algorithm.

Let’s answer another question. If we consider the annotated entities that differ from the ground truth, how close are they to the correct ones? In order to make our analysis as complete as possible, we estimated, for each system, the distribution of the Milne-Witten’s relatedness [16] between the false-positive entities found by the systems and the corresponding (namely, those whose mentions overlap) entities in the ground-truth. The closer to 1 is the relatedness, the better is the correspondence between the entity annotated by the system and the one annotated by humans. Notice that graph-based measures could be used directly to define weak entity annotation relation (M_e). We leave this for future work.

Figure 5 gives, for $r \in [0, 1]$, the cumulative percentage of the number of false-output vs ground-truth entities whose

Dataset	Annotator	Best t	$F1_{mic}$	P_{mic}	R_{mic}
AIDA/CO-NLL (avg-len 1039 chars)	AIDA-Local	0.000	45.8	72.8	33.4
	AIDA-CocktailParty	0.000	46.7	74.1	34.0
	AIDA-PriorityOnly	0.000	39.8	63.3	29.1
	DBPedia Spotlight	0.086	35.2	31.2	40.4
	Illinois Wikifier	0.523	54.0	56.0	52.1
	TagMe 2	0.258	58.3	61.4	55.5
	Wikipedia Miner	0.570	49.7	46.9	52.8
AQUAINT (avg-len 1415 chars)	AIDA-Local	0.000	22.2	37.0	15.8
	AIDA-CocktailParty	0.000	21.2	35.4	15.1
	AIDA-PriorityOnly	0.000	22.0	36.8	15.7
	DBPedia Spotlight	0.078	27.6	20.1	44.0
	Illinois Wikifier	0.523	34.2	29.0	41.7
	TagMe 2	0.188	50.7	45.9	56.7
MSNBC (avg-len 3316 chars)	Wikipedia Miner	0.570	47.2	37.8	62.9
	AIDA-Local	0.000	47.2	74.3	34.6
	AIDA-CocktailParty	0.000	47.4	74.6	34.8
	AIDA-PriorityOnly	0.000	35.2	55.1	25.8
	DBPedia Spotlight	0.070	33.5	31.8	35.2
	Illinois Wikifier	0.477	40.8	34.1	50.9
	TagMe 2	0.188	51.6	48.5	55.0
Wikipedia Miner	0.758	46.0	54.9	39.5	

Table 5: Results based on the weak-annotation match among all annotators over news datasets. Column $F1_{mic}$ indicates the maximum micro-F1 computed for the best score-threshold t^* . Columns P_{mic} and R_{mic} indicate the micro-precision and micro-recall for that t^* . Figures are given as a percentage.

Dataset	Annotator	Best t	$F1_{mic}$	P_{mic}	R_{mic}
AIDA/CO-NLL (avg-len 1039 chars)	AIDA-Local	0.000	45.6	72.4	33.2
	AIDA-CocktailParty	0.000	46.4	73.7	33.9
	AIDA-PriorityOnly	0.000	39.6	62.9	28.9
	DBPedia Spotlight	0.086	33.9	30.0	38.9
	Illinois Wikifier	0.523	50.7	48.7	52.9
	TagMe 2	0.289	56.7	62.1	52.1
	Wikipedia Miner	0.586	46.0	40.1	53.9

Table 6: Results based on the strong-annotation match over the AIDA/CONLL dataset.

relatedness is above r . The highest a line is, the most entities close to the correct ones are found. For $r = 0.5$, TagMe finds entities more closely related to the correct ones, while Illinois Wikifier is the worst, Wikipedia Miner and DBPedia Spotlight are about the same, as are AIDA-PriorityOnly and AIDA-CocktailParty. For $r = 0.8$, AIDA-CocktailParty and Aida-Local achieve the best position showing their effectiveness in finding entities closer to the correct matches.

A comment is in order at this point. In [7], AIDA has been evaluated considering its ability to solve the D2W problem, namely passing the mentions contained in the dataset to the system and restricting its evaluation to only those mentions. This way, mentions found by the tested system but not contained in the dataset were discarded, and thus, not counted as errors. This is clearly a more restricted annotation challenge than the one investigated in this paper. Moreover, in that paper, AIDA’s parameters were tuned over the CONLL dataset, thus achieving the interesting figures of 0.816 macro-precision@1 and 0.825 micro-precision@1. These results are clearly not comparable with those showed in Table 5. We also tried to tune TagMe over this specific dataset and achieved comparable perfor-

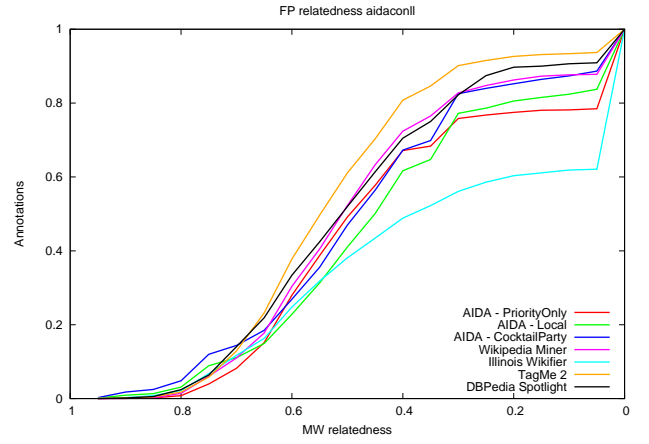


Figure 5: Distribution of Milne-Witten’s relatedness between gold/false-output entities whose mentions overlap.

Dataset	Annotator	Best t	$F1_{mic}$	P_{mic}	R_{mic}
AIDA/CO-NLL (avg-len 1039 chars)	AIDA-Local	0.000	58.7	93.3	42.8
	AIDA-CocktailParty	0.000	58.7	93.3	42.8
	AIDA-PriorityOnly	0.000	58.7	93.3	42.8
	DBPedia Spotlight	0.086	50.4	44.5	57.9
	Illinois Wikifier	0.367	68.5	60.5	78.9
	TagMe 2	0.203	74.6	72.1	77.3
	Wikipedia Miner	0.477	70.0	63.1	78.7

Table 7: Results based on the weak-mention match over the AIDA/CONLL dataset.

mance.¹² This shows that there is ample room for improvement for those systems whenever the annotation problem is restricted, dataset is fixed and tuning is possible. Precise results on TagMe’s performance in this specific scenario will be posted on its website.

6.1.2 Finding just the mentions

For the sake of space we report in Table 7 the results only for $F1_{micro}$ and over the AIDA/CONLL dataset, by using the (weak) mention-annotation match M_m . This basically shows the performance of the NER algorithms used by the various annotation systems. TagMe offers the best parsing thanks to its very high precision. The second best is Wikipedia Miner, Illinois Wikifier is very close, then AIDA’s variants which all use the Stanford NER tagger, and finally DBPedia Spotlight. AIDA’s NER algorithm has a poor F1 because of a very low recall. This ranking holds also for the other news datasets, i.e. AQUAINT and MSNBC. The good result obtained by TagMe and Wikipedia Miner is probably due to the choice of the mention corpus: their use of Wikipedia anchors and titles from the one hand does not let those systems recognize all the mentions contained in a text, but on the other hand it selects better candidates, given the well-defined mention corpus (drawn from Wikipedia), and this leads to a better overall performance.

6.1.3 Finding just the entities

The performance for this task can be computed via the re-

¹²The same training was not possible over the other systems, since they do not provide an interface for training.

Dataset	Annotator	Best t	$F1_{mic}$	P_{mic}	R_{mic}
AIDA/CO-NLL (avg-len 1039 chars)	AIDA-Local	0.000	55.3	77.6	43.0
	AIDA-CocktailParty	0.000	55.7	79.8	42.7
	AIDA-PriorityOnly	0.000	47.7	65.8	37.4
	DBPedia Spotlight	0.117	35.9	37.6	34.4
	Illinois Wikifier	0.523	56.6	53.5	60.2
	TagMe 2	0.336	65.6	68.8	62.7
	Wikipedia Miner	0.586	50.9	47.1	55.3

Table 8: Results based on entity-match for the AIDA/CONLL dataset.

lation M_e . Table 8 reports the results for the AIDA/CONLL dataset. TagMe outperforms the other annotators in terms of F1 and recall, while the highest precision is achieved by AIDA-CocktailParty, at the cost of very low recall. The CocktailParty variant seems to be the best choice for AIDA’s variants, resulting slightly better than Local and significantly better than PriorityOnly. By comparing entities match against weak-mentions match, we notice that the smallest drop in performance is for TagMe (9% in micro-F1), the largest is for Wikipedia Miner (about 19%), Illinois Wikifier (about 12%) and then DBpedia Spotlight (about 15%). AIDA’s drop depends on the variant (from about 3% to 11%): This suggests that the most robust disambiguator is TagMe’s one, together with AIDA-CocktailParty’s; however, the better NER parser makes TagMe the absolute best in the whole annotation process. This ranking among the systems holds also for the other news datasets, namely AQUAINT and MSNBC.

6.1.4 Similarity between systems

Table 9 reports the values of the similarity measure Sim_{macro} using as match relation both the (weak) mention-annotation match M_m and the entity-annotation match M_e . Both similarities are computed on the output produced by the systems when the score-threshold t is set to its value maximizing the annotator performance on either the mention-detection or the entity-match (see Tables 7 and 8).

As expected, the highest similarities are between the three variants of AIDA. A similarity around 65% is shown among Illinois Wikifier, Wikipedia Miner and TagMe, which are rather dissimilar to the AIDA system.

We have seen the similarity on the whole output of two systems, but how good are the dissimilar annotations? To answer this question, we have inspected these annotations by counting the amount of true and false positives. Most correct dissimilarities fall in the range of 30-40%, meaning that the annotations found by one system and not by the other are mostly wrong. But there are some exceptions, the most notable being that the dissimilarity of both AIDA-Local and AIDA-CocktailParty against all other systems has a true-positive rate of 53-62%. In particular, the dissimilarity against Wikipedia Miner is 61% correct, against TagMe it is 62% correct and against Illinois Wikifier it is 53% correct. This demonstrates that AIDA-Local and AIDA-CocktailParty find a set of annotations that are not found by the other annotators but still are mostly correct. This suggests that an improved annotator could be designed with a NER algorithm and a disambiguation algorithm combining those of, e.g., TagMe and AIDA-CocktailParty.

6.2 Experiment 2: Tweets

This experiment aims at testing systems over very-short

	AIDA-CocktailParty	AIDA-Prior-Only	DBPed. Spotl.	Illinois Wik.	TagMe 2	Wikip. Miner
AIDA-Local	93/96	87/91	41/41	42/49	47/48	50/56
AIDA-CocktailParty	100/100	85/89	42/41	42/49	46/49	49/55
AIDA-PriorityOnly		100/100	43/41	41/47	47/48	51/56
DBPedia Spotlight			100/100	45/46	53/51	55/48
Illinois Wikifier				100/100	63/68	65/69
TagMe 2					100/100	70/68
Wikipedia Miner						100/100

Table 9: Sim_{macro} similarity among systems over the AIDA/CONLL dataset, computed both for recognition of mentions/entities.

Dataset	Annotator	Best t .	$F1_{mic}$	P_{mic}	R_{mic}
Meij (avg-len 80 chars)	AIDA-Local	0.000	23.6	53.0	15.1
	AIDA-CocktailParty	0.000	23.0	51.7	14.8
	AIDA-PriorityOnly	0.469	26.8	73.5	16.4
	DBPedia Spotlight	0.102	33.1	29.7	37.4
	Illinois Wikifier	0.406	41.7	51.0	35.3
	TagMe 2	0.133	45.3	43.8	46.9
	Wikipedia Miner	0.289	47.9	48.4	47.3

Table 10: Results based on strong tag match for the Meij dataset.

and poorly composed texts, as tweets. This makes it hard for NER algorithms to recognize the mentions and find pertinent entities. As expected results on tweets are lower than those on news stories. A similar study on Twitter’s posts was conducted by E. Meij and was published on his private blog¹³, but focused on different measures and was conducted on a few/different set of annotators. Since Meij’s dataset is for C2W, the output of known systems must be adapted by taking only the annotations whose score is higher than a threshold. Since only TagMe is specifically designed to address the annotation of short texts (as CMNS), this experiment is aimed also at testing the flexibility of all other systems.

Wikipedia Miner proves the best in F1 (because of its very good recall), followed by TagMe. Illinois Wikifier achieves worse results, especially in terms of recall, while DBpedia Spotlight loses significantly in precision. All variants of AIDA have a poor performance, although with highest precision; this is likely due to the poor performance of the Stanford parser on this type of text.

6.3 Experiment 3: Web pages

This experiment aims at testing systems over long texts drawn from Web pages. This dataset is challenging for TagMe which was designed for very-short texts only. The key difficulty is that attempting to optimize global coherence of the annotations, as many systems do, may be dangerous because of concept drifts in the long input text. Moreover, the text provided by the IITB dataset includes, without providing any structure, accessory textual elements taken from the web page (like menus, footers, etc.) that are generally not semantically linked to the main body.

Table 11 shows that Wikipedia Miner is the best annota-

¹³<http://edgar.meij.pro/comparison-semantic-labeling-algorithms-twitter-data/>

Dataset	Annotator	Best t	$F1_{mic}$	P_{mic}	R_{mic}
IITB (avg-len 3879 chars)	AIDA-Local	0.000	7.7	66.3	4.1
	AIDA-CocktailParty	0.000	7.6	65.7	4.1
	AIDA-PriorityOnly	0.000	7.5	63.6	4.0
	DBpedia Spotlight	0.023	48.0	46.2	50.0
	Illinois Wikifier	0.438	44.3	58.0	35.9
	TagMe 2	0.102	43.6	45.2	42.0
	Wikipedia Miner	0.219	52.2	56.8	48.2

Table 11: Results based on the weak-annotation match over the IITB dataset.

System	Meij	AIDA/- CO- NLL	AQUA- INT	MSNBC	IITB
AIDA-Local	2228	13003	12030	22957	21535
AIDA-CocktailParty	865	16876	13913	28335	30402
AIDA-PriorityOnly	78	623	633	1187	1363
DBpedia Spotlight	1057	9045	168	1289	1236
Illinois Wikifier	102	1277	1303	3891	2801
TagMe 2	1	33	42	103	127
Wikipedia Miner	17	171	126	1282	1245

Table 12: Avg. annotation time per document (ms).

tor, closely followed by DBpedia Spotlight, the third being Illinois Wikifier and then TagMe. However, unlike the news datasets with long texts (i.e. AQUAINT, MBSC), AIDA is worse because of the low recall of the Stanford NER Tagger.

6.4 Runtime efficiency

Time efficiency is a key feature of entity-annotation systems because it impacts onto their scalability. Of course, network latency and the HW/SW features of the remote/local servers may affect time measures. Nevertheless, we adopted some care in order to make these evaluations unbiased (see Table 12 for a summary). For Illinois Wikifier, which run locally, the runtime has been recorded as the difference between the system time after and before the library call. For AIDA, the time has been recorded as the one returned by the method `getOverallRuntime()` of the RMI Service, and thus it does not include the network latency. Similarly, as TagMe’s runtime, it has been considered that included in the response returned by the RESTful API. For Wikipedia Miner and DBpedia Spotlight, accessed through a Web service, the time has been recorded as the difference between the system time after and before the query, and the timing has been calibrated subtracting the time needed to process an empty query, which should account for the network latency.

TagMe is definitely the fastest annotator, being about 10 times faster than Wikipedia Miner, which is the second one. DBpedia Spotlight is also fast, except for a peak of 9s over AIDA/CONLL and 1s on Meij. Illinois Wikifier is about three times slower. AIDA-PriorityOnly is the fastest one among all AIDA variants, due to the basic disambiguation process, AIDA-Local and CocktailParty are slower, being about 300 times slower than TagMe. For all systems, the runtime looks linear in the size of the text – which is good. A special case is the dataset of Meij, for which texts are very short; TagMe is extremely fast, due to the engineering tuned on short texts, then comes Wikipedia Miner, AIDA-PriorityOnly and Illinois Wikifier, the others being way slower.

We conclude by reporting in Figure 6 a 2d plot of the runtime and the best F1 of each tested annotator over the

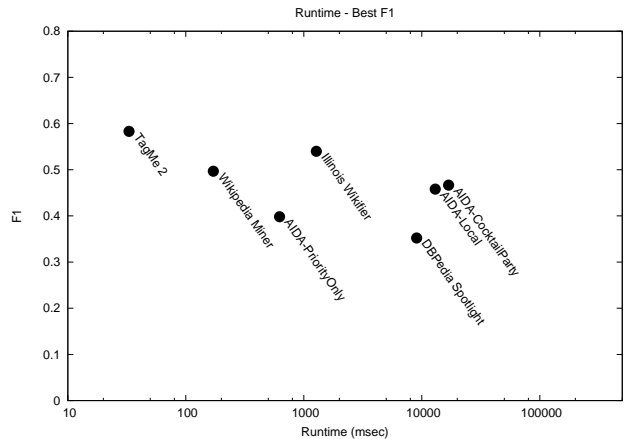


Figure 6: Average runtime (in log-scale) for dataset AIDA/CONLL and best achieved F1 measures (metrics based on *weak* annotation match).

AIDA/CONLL dataset. For all other datasets the plotting is similar. The net result is that TagMe dominates all other systems in terms of runtime and F1, AIDA-priority is faster but at the prize of lower effectiveness. Despite the difficulties indicated before concerning the evaluation of systems’ efficiency, our figures differ so much that they provide a reasonable picture of systems’ speed.

7. CONCLUSIONS

We designed, implemented and tested a benchmarking framework to fairly and fully compare entity-annotation systems. Our goal was to help improving the understanding of this challenging new problem. Our framework is easily extensible with new annotation systems, new datasets, new matching relations and new evaluation measures. It is written in Java and it has been released to the public as open source code. Using the framework we provided the first complete comparison among all publicly available entity annotators over all available datasets for the entity-annotation task, showing that some of these annotators are very effective and ready to scale over large datasets and real applications. We believe that our framework will spur new research on this interesting topic, providing a common ground upon which researchers will test the efficiency and effectiveness of their proposals.

8. ACKNOWLEDGMENTS

We thank all our colleagues who provided prompt support for their software, datasets and web services: Johannes Hoffart, Edwin Lewis-Kelham, Edgar Meij, Lev Ratinov, Mark Sammons, Ugo Scaiella, Daniele Vitale, Gerhard Weikum, Ian Witten. We also thank the Wikimedia Foundation and the Wikipedia contributors, whose knowledge-sharing philosophy made this work possible. This research was supported by the PRIN ARS-technomedia grant and by a Google research award 2013.

9. REFERENCES

- [1] I. Bordino, G. De Francisci Morales, I. Weber, and F. Bonchi. From Machu_Picchu to “rafting the urubamba river”: Anticipating information needs via the Entity-Query Graph. In *Proc. WSDM*, 2013 (to appear).
- [2] S. Cucerzan. Large-scale named entity disambiguation based on wikipedia data. In *Proc. EMNLP and CNLL*, 708–716, 2007.
- [3] P. Ferragina and U. Scaiella. Tagme: on-the-fly annotation of short text fragments (by Wikipedia entities). In *Proc. CIKM*, 1625–1628, 2010.
- [4] P. Ferragina and U. Scaiella. Fast and accurate annotation of short texts with Wikipedia pages. *IEEE Software*, 29(1): 70–75, 2012.
- [5] E. Gabrilovich and S. Markovitch. Wikipedia-based semantic interpretation for natural language processing. *J. Artif. Int. Res.*, 34(1):443–498, 2009.
- [6] J. Hoffart, F. M. Suchanek, K. Berberich, E. Lewis-Kelham, G. de Melo, and G. Weikum. Yago2: exploring and querying world knowledge in time, space, context, and many languages. In *Proc. WWW*, 229–232, 2011.
- [7] J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust disambiguation of named entities in text. In *Proc. EMNLP*, 782–792, 2011.
- [8] P. Jiang, H. Hou, L. Chen, S. Chen, C. Yao, C. Li, and M. Wang. Wiki3C: Exploiting Wikipedia for Context-aware Concept Categorization. In *Proc. WSDM*, 2013 (to appear).
- [9] S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. Collective annotation of Wikipedia entities in web text. In *ACM KDD*, 457–466, 2009.
- [10] C.D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*. Cambridge University Press, 2008.
- [11] O. Medelyan, D. Milne, C. Legg, and I. H. Witten. Mining meaning from Wikipedia. *Int. J. Hum.-Comput. Stud.*, 67(9):716–754, 2009.
- [12] E. Meij. A Comparison of five semantic linking algorithms on tweets. Personal Blog: <http://altur1.com/aujuc>, 2012.
- [13] E. Meij, W. Weerkamp, and M. de Rijke. Adding semantics to microblog posts. In *Proc. WSDM*, 563–572, 2012.
- [14] P.N. Mendes, M. Jakob, A. García-Silva, and C. Bizer. DBpedia spotlight: shedding light on the web of documents. In *Proc. I-SEMANTICS*, 1–8, 2011.
- [15] R. Mihalcea and A. Csomai. Wikify!: linking documents to encyclopedic knowledge. In *Proc. ACM CIKM*, 233–242, 2007.
- [16] D. Milne and I. H. Witten. Learning to link with wikipedia. In *Proc. CIKM*, 509–518, 2008.
- [17] D. Milne and I. H. Witten. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. *AAAI Workshop on Wikipedia and Artificial Intelligence*, 2008.
- [18] L. Ratinov and D. Roth. Design challenges and misconceptions in named entity recognition. In *Proc. CoNLL*, 147–155, 2009.
- [19] L. Ratinov, D. Roth, D. Downey, and M. Anderson. Local and global algorithms for disambiguation to wikipedia. In *Proc. ACL-HLT*, 1375–1384, 2011.
- [20] G. Rizzo, R. Troncy, S. Hellmann, and M. Brummer. Nerd meets nif: Lifting nlp extraction results to the linked data cloud. In *Proc. LDOW*, 2012.
- [21] U. Scaiella, P. Ferragina, A. Marino, M. Ciaramita. Topical Clustering of Search Results. In *Proc. WSDM*, 223–232, 2012.
- [22] D. Vitale, P. Ferragina, U. Scaiella. Classification of short texts by deploying topical annotations. In *Proc. ECIR*, 376–387, 2012.
- [23] M. A. Yosef, J. Hoffart, I. Bordino, M. Spaniol, and G. Weikum. AIDA: an online tool for accurate disambiguation of named entities in text and tables. *PVLDB*, 4(12):1450–1453, 2011.