AIDA: An Online Tool for Accurate Disambiguation of Named Entities in Text and Tables

Mohamed Amir Yosef, Johannes Hoffart, Ilaria Bordino, Marc Spaniol, Gerhard Weikum
Max Planck Institute for Informatics, Saarbrücken, Germany
{mamir,jhoffart,mspaniol,weikum}@mpi-inf.mpg.de, bordino@yahoo-inc.com

ABSTRACT

We present AIDA, a framework and online tool for entity detection and disambiguation. Given a natural-language text or a Web table, we map mentions of ambiguous names onto canonical entities like people or places, registered in a knowledge base like DBpedia, Freebase, or YAGO. AIDA is a robust framework centred around collective disambiguation exploiting the prominence of entities, similarity between the context of the mention and its candidates, and the coherence among candidate entities for all mentions. We have developed a Web-based online interface for AIDA where different formats of inputs can be processed on the fly, returning proper entities and showing intermediate steps of the disambiguation process.

1. INTRODUCTION

Motivation. News articles, postings in blogs and online communities, and other Web pages contain mentions of named entities such as people, places, or organizations. This entity information is a great asset for making sense of the raw and often noisy contents, and key to enabling business-intelligence and semantic-search applications. However, names are often ambiguous: the same name can have many different meanings. For example, given a text snippet like "Harry is the opponent of you know who", how can we tell that "Harry" denotes Harry Potter rather than Dirty Harry or Prince Harry of England? Establishing this mapping between the mention and the actual entity is known as the problem of *named-entity disambiguation (NED)* (aka. entity-name resolution).

Even structured Web data such as tables and lists in Web pages face this problem, as their cells merely contain names, often in abbreviated or otherwise non-canonical form (e.g., "Mac" instead of "Apple Macintosh") [4, 11]. How can we tell that in a table like

Liverpool	Manchester	2:2
Tottenham	Newcastle	4:1
Celtic	Rangers	0:0

the city names denote football clubs such as FC Liverpool and "Celtic" and "Rangers" are the major teams in Glasgow? The Linked-Data cloud of semantically interconnected data and knowledge bases [2], has the goal of providing extensive cross-referencing at the entity level, but the current forms of manually or heuristically created mappings have very limited coverage. For example, the archive of the New York Times (`data.nytimes.com`) has only about 5000 people and 2000 locations (manually) linked to other sources.

State of the Art. The NED problem has been addressed in different communities. For structured data with schematic attributes, *record-linkage* algorithms aim to find equivalence classes of records that denote the same entities [5]. This is based on attribute-wise similarities such as string edit distance or n-gram overlap. The emphasis has been on scalable batch processing, as needed for de-duplication [13]. Annotating cells in Web tables with proper entities has recently received attention [4, 11]. NLP research has harnessed Wikipedia as a means for linking words and phrases onto canonical entities [3, 6]. This theme has been further pursued in Web mining, most notably, the work of [12] and [10].

Recent methods leverage knowledge bases such as DBpedia [1], `freebase.com`, or YAGO [15]. These contain millions of entities, with fine-grained assignment to semantic types (e.g., heavy metal rock guitarists, fictional characters in mystery novels, etc.), and billions of relational facts between entities. Also, they provide dictionaries of short names and paraphrases for entities. This way, one can quickly identify candidate entities for mentions, but these sets tend to be very large.

Knowledge bases can guide NED in several ways. First, a *popularity prior* for each mention-entity pair can estimate how often the mention phrase is used together with the entity. A good source for this estimation is the anchor texts of Wikipedia links, as these have both short mentions and unique designation of entities. Second, a variety of mention-entity *similarity* measures can be computed, based on the *context of the mention* in its input text and the *context of the entity* in the knowledge base. On the mention side, we can construct a bag-of-words representation from the words in the mention's proximity, possibly filtered or weighted by other mentions, common nouns, etc. On the entity side, a bag-of-words can be constructed from names of related entities, the entities' types in the taxonomy, or semantic properties. Third, we can reason on the *coherence* between the entities that different mentions in a text or table would potentially denote. For example, once we map "you know who" to Lord Voldemort or "Manchester" to Manchester United, it is likely that the meanings of "Harry" and "Liverpool" are Harry Potter and FC Liverpool.

Recently, collective learning methods have been used for jointly mapping all mentions to entities in one step, based on probabilistic graphical models [14, 10, 16]. The leading method of [10] approximates the best mapping by solving a relaxed linear program with subsequent rounding [10]. The best methods achieve 70-90 % accuracy, depending on how clean the input text is. For very short texts or texts that switch between different topics, the result is far from near-human quality. The best prior methods are computationally expensive, as they build on sampling over factor

Proceedings of the VLDB Endowment, Vol. 4, No. 12

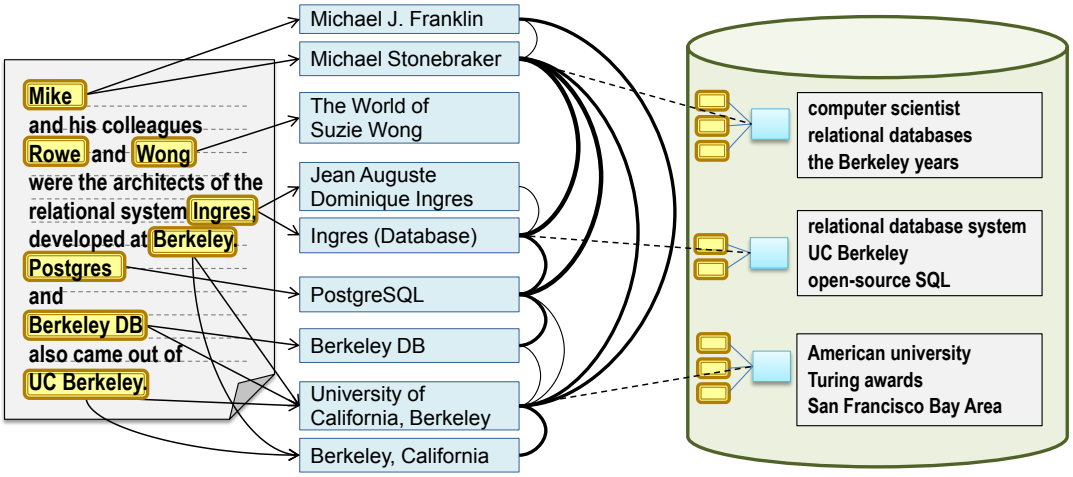

Figure 1: Example for Mention-Entity Graph.

graphs or linear programs for optimization. With the exception of [12], which offers a Web service at `wdm.cs.waikato.ac.nz:8080/service?task=wikify`, none of the above methods is suitable for online computation, with interactive response time. The service of [12] is highly geared to (an old version of) Wikipedia articles, and did not perform well when we tried it with news articles.

Our Approach. This paper presents the AIDA system, which includes an efficient and accurate NED method, suited for online usage. Our approach leverages the YAGO2 knowledge base [8], as an entity catalog and a rich source of relationships among entities. We cast the joint mapping into a graph problem: mentions from the input text and candidate entities define the node set, and we consider weighted edges between mentions and entities, capturing context similarities, and weighted edges among entities, capturing coherence. The AIDA system is accessible online at the URL `www.mpi-inf.mpg.de/yago-naga/aida/`. It accepts plain text as well as HTML, and also supports semistructured inputs like tables, lists, or short XML files.

2. FRAMEWORK AND ALGORITHMS

The input to AIDA is an arbitrary text, optionally with HTML or XML markup or in the RDF N3 form, with mentions of named entities (people, music bands, songs, universities, etc.). The goal is to find the correct mapping for the mentions onto canonical entities in a knowledge base (currently YAGO).

Mentions are automatically detected using the Stanford NER Tagger (`nlp.stanford.edu/software/CRF-NER.shtml`). For collective mapping, we use a graph-based approach. The graph is constructed with mentions and their candidate entities as nodes. We have 2 types of edges:

- **mention-entity edges:** between mentions and their candidate entities with weights that capture the similarity between the context of a mention and a candidate;
- **entity-entity edges:** between different entities with weights that capture the coherence (semantic relatedness) between two entities.

Our goal is to reduce this graph to a dense sub-graph where each mention node is connected to one and only one candidate entity node, which provides our output mapping. Density here refers to the total weight of the sub-graph's edges, or alternatively, to the minimum weighted degree in the sub-graph. Once the graph is constructed, we use a greedy algorithm to compute the sub-graph. In each iteration, we perform two steps:

- identify the entity node that has the lowest weighted degree (sum of the weights of the node's incident edges), and
- remove this node and its incident edges from the graph unless it is the last remaining candidate entity for one of the mentions.

Figure 1 illustrates the mention-entity graph for an input text with highlighted mentions (left) and candidate entities (middle) based on a knowledge base (right). The thickness of edges between entities depicts different edge weights. Next, we describe the features and measures for computing the edge weights.

The *similarity* between a mention and a candidate entity is computed as a linear combination of two ingredients. The first one is the prominence of an entity, e.g., Prince Harry of England vs. Harry Kelly, a lesser known American basketball player. This acts as a prior probability for each potential mapping. We compute this prior by collecting statistics on href anchor texts and their link targets in Wikipedia. The second ingredient for the mention-entity edge weights is based on the overlap between a mention's context and a candidate entity's context. For a mention we consider the full input text as its context. For entities, we consider *entity keyphrases*, pre-computed from the Wikipedia articles that YAGO's entities connect to. We define the notion of keyphrases to be all phrases in link anchors, including category names, citation titles, and external references in the entity article. We extended this further by considering also the titles of incoming links for an entity's article, as an additional source of describing the entity.

While keyphrase overlap between the contexts of a mention and entity is an expressive similarity measure, we will rarely find perfect matches for multi-word keyphrases in the input text. Consider entity `Manchester United`, which has keyphrases such as "2008 UEFA Champions League Winner" But the input text may say "winner of the champions league in 2008", not nearly a full match of the keyphrase. Therefore, we devised a partial-match model to improve coverage. To avoid degrading accuracy, we consider the size of the window that covers all words of the keyphrase that appear in the input text. For example, the above wording in the text matches 4 of the 6 words of the keyphrase within a window of size 7. Moreover, we penalize the keyphrases that occur in the text by their distance from the mention under consideration. Obviously, once we allow partial matches, different words in a phrase have different degrees of importance. To accommodate this aspect, we collect statistics from a

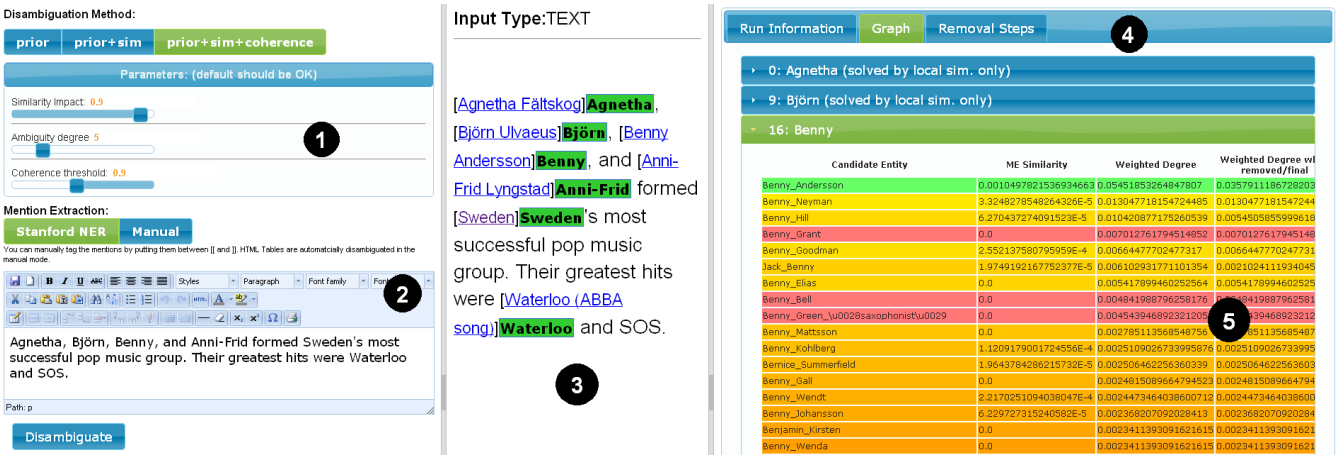

Figure 2: User Interface of AIDA.

large corpus (e.g., Wikipedia) about the co-occurrence frequency of a word an the entity of interest. We use the Mutual Information (MI) measure (aka. relative entropy) to quantify the specificy of a word for an entity. These values serve as per-word weights for scoring the partial matches in the input text. The scores for all matches are aggregated by summation with distance decay.

For the *coherence* weights of entity-entity edges, we harness the Wikipedia link structure. We define the coherence between two entities to be proportional to the number of incoming links that are shared between their Wikipedia articles [12]. For the dense sub-graph that yields the final disambiguation, we expect the final candidate entities of different mentions to be mutually connected by high edge weights. More details on the features and algorithms of this approach are included in [9].

3. SYSTEM IMPLEMENTATION

AIDA is a framework that encompasses a suite of methods for NED. This includes methods based on popularity prior, different notions of similarity, and the graph-based notion of coherence. The latter includes various techniques for setting edge weights. AIDA draws on the YAGO2 knowledge base [8] and its rich statistics derived from Wikipedia and other sources.

Figure 2 shows a screenshot of AIDA's UI in a browser window.

1) The online UI offers three major methods: prior only, prior and similarity together, and the graph-based method that integrates prior, similarity, and coherence. For the graph method, various parameters can be adjusted by sliders.
2) The user can input any text, e.g. by copy-and-paste from news articles, or HTML table. By default, the Stanford NER Tagger would identify noun phrases that can be interpreted as entity mentions. As this is error-prone, the user can alternatively flag mentions by putting them in double brackets, e.g.: "Harry is the opponent of [[you know who]]."
3) The output shows for each mention (in green), the entity that the chosen method has assigned to the mention, in the form of a clickable link. The links point to the corresponding Wikipedia articles. Alternatively, they could point to the YAGO2 entries, or any comparable knowledge source in the Linked-Data world.
4) For each input text, once it is disambiguated, the user can see a variety of run-time information and statistics about the input's ambiguity, the constructed graph, etc.
5) For each mention, one can inspect the candidate entities and associated weights for similarity and coherence. The entities are sorted in the order in which they were dropped from consideration (removed from the graph, if the graph-based method is used). The coloring reflects this order: the entity finally selected is shown in green and the alternatives are in the yellow-to-red spectrum. The darker the color the earlier the entity is dropped. The user can also drill down, upon further clicks, into the related entities that led to assessing a candidate entity to be more or less appropriate. Another view (not shown in the screenshot) lists the individual steps of the algorithm for finding the best sub-graph.

In addition to YAGO2, AIDA makes use of several software tools, notably, PostgreSQL as a back-end server and the Stanford NER Tagger. The system is available online at `www.mpi-inf.mpg.de/yago-naga/aida/`.

4. DEMO SCENARIOS

The actual demo of AIDA will highlight a variety of use cases. Users will be able to freely interact with the system.

Text as input. The first use case is that users type in their own texts with ambiguous names. This may be a short and difficult text, like the ones we have as examples earlier; or it can be a news article, blog posting, or content from a discussion forum that is copy-and-pasted into AIDA. Users can choose different methods, vary their configurations, explore the effects in terms of candidate entities, the weighted graph of mentions and entities, and the output quality. Figure 3 shows such a text-centric example using the graph-based method. Note that the difficulty of this intentionally sophisticated example: only first names of people, and Berkeley and Stanford both could denote cities as well as universities or historic persons. The prior-only and the prior-and-similarity methods would make many mistakes; for example, "Ingres" would be mapped to the painter (after whom the database system was named), and "Michael" to Michael Jackson or Michael (archangel). Our graph-based method gets all disambiguations perfectly right for this difficult example.

Table as input. An interesting option that demonstrates the versatility of AIDA is to copy-and-paste Web tables in HTML (or XML) format. Figure 4 shows an example with a table that lists some European football clubs, one of the seasons in their national leagues, and the top-scoring player of that season. Note the high ambiguity here, as the clubs are merely given by city name or short-hand notation and the players by last name only. The figure shows the outcomes of the three main methods with prior only (left), with prior and similar-

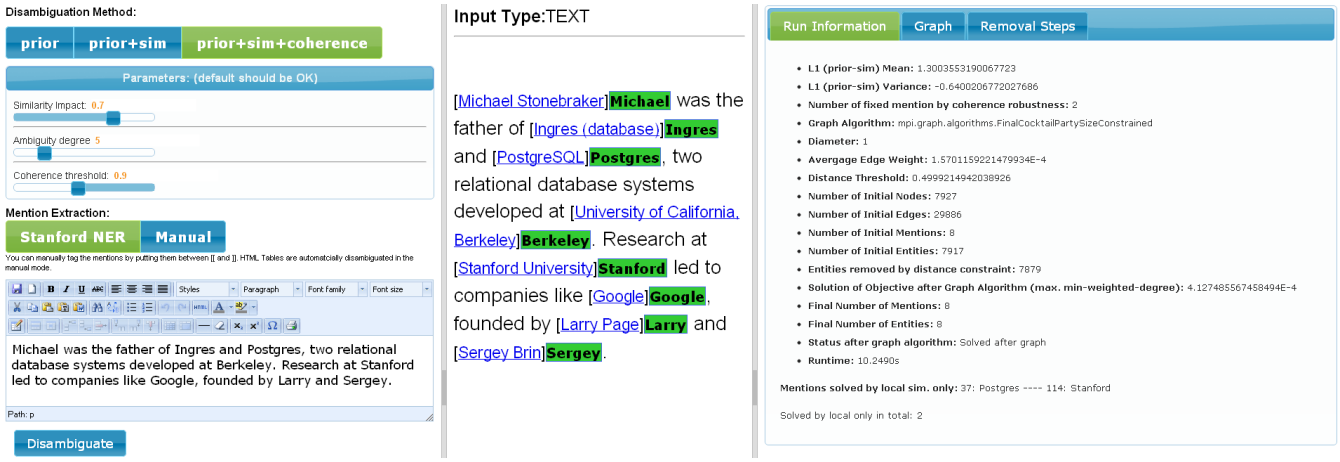

Figure 3: AIDA with Text Input.

Figure 4: AIDA with Table as Input.

ity (middle), and with the full-fledged graph method (right). There are major differences in accuracy, underlining the high quality of AIDA's graph-based approach. For example, the prior-only method maps Barcelona to the city, and Real to Real County, Texas. The graph method maps these to the corresponding football clubs, and is even able to correctly assign the mention "2008/09" to the entity 2008-09 La Liga, which is the 2008-09 season of the Spanish first league in football.

We are currently working on supporting also mixed forms of input data and more richly structured input. These include semantically annotated HTML with microformats such as hCard or hProduct, RDF triples from linked-data sources enriched with textual context, as well as JSON-encoded semantic objects.

5. REFERENCES

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z.G. Ives: DBpedia: A Nucleus for a Web of Open Data. ISWC/ASWC 2007:722-735

[2] C. Bizer, T. Heath, T. Berners-Lee: Linked Data - The Story So Far. Int. J. Semantic Web Inf. Syst. 5(3):1-22, 2009

[3] R.C. Bunescu, M. Pasca: Using Encyclopedic Knowledge for Named entity Disambiguation. EACL 2006

[4] M.J. Cafarella, A.Y. Halevy, N. Khoussainova: Data Integration for the Relational Web. PVLDB 2(1):1090-1101, 2009

[5] W.W. Cohen: Data integration using similarity joins and a word-based information representation language. ACM TOIS 18(3):288-321, 2000

[6] S. Cucerzan: Large-Scale Named Entity Disambiguation Based on Wikipedia Data. EMNLP-CoNLL 2007:708-716

[7] J.R. Finkel, T. Grenager, C. Manning: Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. ACL 2005. `nlp.stanford.edu/software/CRF-NER.shtml`

[8] J. Hoffart, F. Suchanek, K. Berberich, E. Lewis-Kelham, G. de Melo, G. Weikum: YAGO2: Exploring and Querying World Knowledge in Time, Space, Context, and Many Languages. Demo Paper, WWW 2011:229-232, data at `www.mpi-inf.mpg.de/yago-naga/yago/`

[9] J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, G. Weikum: Robust Disambiguation of Named Entities in Text. EMNLP 2011

[10] S. Kulkarni, A. Singh, G. Ramakrishnan, S. Chakrabarti: Collective annotation of Wikipedia entities in web text. KDD 2009:457-466

[11] G. Limaye, S. Sarawagi, S. Chakrabarti: Annotating and Searching Web Tables Using Entities, Types and Relationships. PVLDB 3(1):1338-1347, 2010

[12] D.N. Milne, I.H. Witten: Learning to link with wikipedia. CIKM 2008:509-518

[13] F. Naumann, M. Herschel: An Introduction to Duplicate Detection. Morgan & Claypool, 2010

[14] P. Singla, P. Domingos. Entity resolution with Markov Logic. ICDM 2006:572-582

[15] F.M. Suchanek, G. Kasneci, G. Weikum: YAGO: a Core of Semantic Knowledge. WWW 2007:697-706

[16] M.L. Wick, A. Culotta, K. Rohanimanesh, A. McCallum: An Entity Based Model for Coreference Resolution. SDM 2009:365-376