Big Data
Volume 5 Number 1, 2017
© Mary Ann Liebert, Inc.
DOI: 10.1089/big.2017.0012

ORIGINAL ARTICLE

Liberal Entity Extraction: Rapid Construction of Fine-Grained Entity Typing Systems

Lifu Huang,[1,*] Jonathan May,[2] Xiaoman Pan,[1] Heng Ji,[1] Xiang Ren,[3] Jiawei Han,[3] Lin Zhao,[4] and James A. Hendler[1]

Abstract
The ability of automatically recognizing and typing entities in natural language without prior knowledge (e.g., predefined entity types) is a major challenge in processing such data. Most existing entity typing systems are limited to certain domains, genres, and languages. In this article, we propose a novel unsupervised entity-typing framework by combining symbolic and distributional semantics. We start from learning three types of representations for each entity mention: general semantic representation, specific context representation, and knowledge representation based on knowledge bases. Then we develop a novel joint hierarchical clustering and linking algorithm to type all mentions using these representations. This framework does not rely on any annotated data, predefined typing schema, or handcrafted features; therefore, it can be quickly adapted to a new domain, genre, and/or language. Experiments on genres (news and discussion forum) show comparable performance with state-of-the-art supervised typing systems trained from a large amount of labeled data. Results on various languages (English, Chinese, Japanese, Hausa, and Yoruba) and domains (general and biomedical) demonstrate the portability of our framework.

Keywords: fine-grained entity typing; unsupervised learning; Liberal Information Extraction; multi-level entity mention and representation

Introduction

One important area of human language technology is the area of information extraction (IE). Systems, which perform IE, start from natural language inputs, usually in the form of a document set, and attempt to identify the various entities and events being described in each document. These extracted entities are then used to create knowledge bases or other tools that allow later systems to access the unstructured text through collections of the extracted entities, or to enhance systems for search, post-processing, etc. IE systems that extract only a small number of entity types are referred to as coarse grained, those that can extract many more entity types, as we describe hereunder, are known as fine-grained IE systems. When the inputs can come from only one known domain and the system can be specialized to that area, the IE system is known as a "closed domain" system; when the documents can come from any domain, not necessarily known in advance, the system is referred to as an "open domain" system.

Open-domain IE remains a challenging and costly task. Previous IE programs mainly focused on a small set of predefined coarse-grained types and closed domains. For example, a commonly used test set, known as MUC-7,[1] was based on the three most common types: person, organization, and location. The representation used in that corpus, Automatic Content Extraction (ACE), separated geopolitical entities from (natural) locations and introduced weapons, vehicles, and facilities. These entity types are very useful for many downstream natural language processing (NLP)[2–10] and information retrieval[11–14] tasks. However, such manually defined type schemas often fail to generalize to new domains, such as the biomedical domain. In addition, traditional IE methods are highly dependent on

[1]*Rensselaer Polytechnic Institute, Troy, New York.*
[2]*Information Sciences Institute, Marina del Rey, California.*
[3]*University of Illinois at Urbana-Champaign, Urbana, Illinois.*
[4]*Research and Technology Center, Robert Bosch LLC, Palo Alto, California.*

*Address correspondence to: *Lifu Huang, Department of Computer Science, Rensselaer Polytechnic Institute, 100 Mcchesney Avenue, Troy, NY 12180,* E-mail: huangl7@rpi.edu

human annotations, so they suffer from poor scalability and portability when moving to a new language, domain, or genre.

Considering these challenges, we have developed a new "Liberal" IE paradigm, which can simultaneously discover a domain-rich schema and extract information units with fine-grained types efficiently. It allows a "cold-start" (or minimal supervision from existing knowledge bases) and can be adapted to any domains, genres, or languages without any human annotated data. The only input to a Liberal IE system is an arbitrary corpus from any domain or topic. The output includes a schema, which contains a flexible hierarchy of types with multilevel granularities and is customized for the specific input corpus.

In this research, we demonstrate a new Liberal IE paradigm by showing automatic discovery of fine-grained entity types. Recent work[15,16] suggests that using a larger set of fine-grained types can lead to substantial improvement for these downstream NLP applications. To demonstrate the motivations of our unsupervised fine-grained entity-typing framework, let us begin by considering the following examples, which motivate several heuristics that have guided our approach:

E1. **Mitt Romney** was born on March 12, 1947, at **Harper University Hospital** in **Detroit**, **Michigan**, the youngest child of automobile executive **George Romney**.

E2. **Yuri dolgoruky**, *equipped* by Bulava nuclear-armed missile, is the first in a series of new *nuclear submarines* to be *commissioned* this year.

E3. **OWS** *activists* were part of the *protest*.

E4. The effects of the **MEK** inhibitor on total **HER2**, **HER3**, and on phosphorylated **pHER3** were dose dependent.

In E1, mentions such as *Mitt Romney*, *George Romney*, *Detroit*, and *Michigan* are commonly used and have no type ambiguity. That is, their types can be easily determined by their general semantics. Our first intuition is the following.

Heuristic 1: ***The types of common entities can be effectively captured by their general semantics.***

However, many entities are polysemantic and can be used to refer to different types in specific contexts. For example, *Yuri Dolgoruky* in E2, which generally refers to the Russian prince, is the name of a submarine in this specific context. Likewise, *OWS* in E3, which refers to *Occupy Wall Street*, is a very novel emerging entity. It may not exist in the word vocabulary, and its general semantics may not be learned adequately because of its low frequency in the data. Such types are difficult to capture with general semantics alone, but can be inferred by their specific contexts, such as *nuclear submarines*, *equip*, *commission*, *activists*, and *protest*. Thus, our second intuition is the following.

Heuristic 2: ***The types of uncommon, novel, emerging, and polysemantic entities can be inferred by their specific contexts.***

In E4, *MEK*, *HER2*, *HER3*, and *pHER3* are biomedical domain-specific entities. Their types can be inferred from domain-specific knowledge bases (KBs). For example, the properties for *pHER3* in biomedical ontologies include *Medical*, *Oncogene*, and *Gene*. Therefore, we derive the third intuition.

Heuristic 3: ***The types of domain-specific entities largely depend on domain-specific knowledge.***

Based on these heuristics, we have developed an unsupervised fine-grained entity-typing framework that combines general entity semantics, specific contexts, and domain-specific knowledge. Because it does not need a predefined typing schema, manual annotations, or handcrafted linguistic features, this framework can be easily applied to new domains, genres, or languages. The types of all entity mentions are automatically discovered based on a set of clusters, which can capture fine-grained types customized for any input corpus.

We compare the performance of our approach with state-of-the-art name tagging and fine-grained entity-typing methods, and show the performance on various domains, genres, and languages. The results are comparable to state-of-the-art systems that are much more complex and handcrafted.

Related Work

Several recent studies have focused on fine-grained entity typing. Fleischman and Hovy[17] classified person entities into eight fine-grained subtypes based on local contexts. Sekine[18] defined more than 200 types of entities. The abstract meaning representation (AMR[19]) defined more than 100 types of entities. Fine-Grained Entity Recognizer (FIGER)[16] derived 112 entity types from Freebase[20] and trained a linear-chain conditional random field (CRF) model[21] for joint entity identification and typing. Gillick et al.[22] and Yogatama et al.[23] proposed the task of context-dependent fine-grained entity typing, whereby the acceptable type labels are limited to only those deducible from local contexts (e.g., a sentence or a document). Similar to FIGER, this work also derived the label set from Freebase and generated the training data automatically from entities resolved in Wikipedia. Lin et al.[24] proposed propagating the types from linkable entities to unlinkable noun phrases based on a set of features. Hierarchical Type Classification for Entity Names (HYENA)[25] derived a very fine-grained type taxonomy from Yet Another Great Ontology (YAGO)[26] based on a mapping between Wikipedia categories and WordNet synsets. This type structure incorporated a large hierarchy of 505 types organized under five top level classes (person, location, organization, event, and artifact), with 100 descendant types under each of them. Although these methods can handle multiclass multilabel assignment, the automatically acquired training data are often too noisy to achieve good performance. In addition, the features they exploited are language dependent, and their type sets are rather static.

Our work is also related to embedding techniques. Turian et al.[27] explored several unsupervised word representations including distributional representations and clustering-based word representations. Mikolov et al.[28] examined vector space word representations with a continuous space language model. Besides word embedding, several phrase embedding techniques have also been proposed. Yin and Schütze[29] computed embeddings for generalized phrases, including both conventional linguistic phrases and skip bigrams. Mitchell and Lapata[30] proposed an additive model and a multiplicative model. Linguistic structures have been proven useful to capture the semantics of basic language units.[31–34]

Socher et al.[33] designed a Dependency Tree Recursive Neural Network (DT-RNN) model to map sentences into compositional vector representations based on dependency trees. Hermann and Blunsom[32] explored a novel class of combinatory categorial autoencoders to utilize the role of syntax in combinatory categorial grammar to model compositional semantics. Socher et al.[34] designed a recursive neural tensor network to compute sentiment compositionality based on the Sentiment Treebank. Huang et al.[31] proposed to induce event schemas based on compositional event structure representations. Compared with these efforts, in this work, we attempt to compose the context information to infer the fine-grained types. Considering not all contexts are meaningful, we carefully selected specific types of relations to capture concept-specific local contexts instead of sentence-level or corpus-level contexts.

Approach Overview

Figure 1 shows the major components of our system, which can automatically discover fine-grained entity types based on entity linking techniques and distributed semantic representations. It takes the boundaries of all entity mentions as input and produces a type label for each mention as output. The framework starts from learning three kinds of representations:

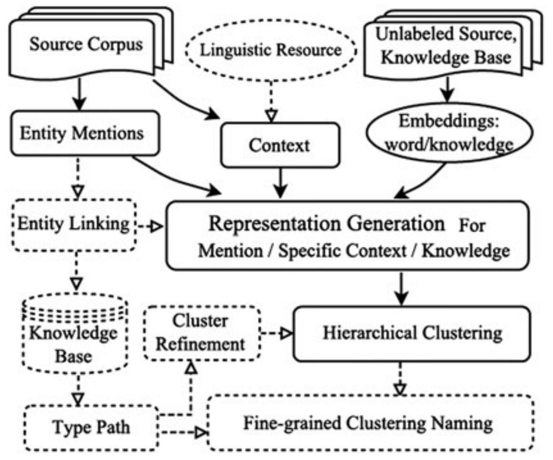

Source Corpus
Linguistic Resource
Unlabeled Source, Knowledge Base
Entity Mentions
Context
Embeddings: word/knowledge
Entity Linking
Representation Generation For Mention / Specific Context / Knowledge
Knowledge Base
Cluster Refinement
Hierarchical Clustering
Type Path
Fine-grained Clustering Naming

FIG. 1. Approach overview (solid boxes are required whereas the dotted boxes are optional).

(1) a general entity distributed representation based on global contexts,
(2) a specific context representation based on local context words, and
(3) a knowledge representation, to model domain-specific knowledge for each mention.

For example, Figure 2 shows how these three types of information can be used to infer the type of "pHER3." It shows how the type of pHER3 (Gene) can be inferred from similar words (e.g., erbB3, HER3) based on general semantics, specific context words such as "phosphorylated," as well as the properties from KB, e.g., oncogenes.

After learning general and context-specific semantics, we apply unsupervised entity linking to link entity mentions to a domain-specific knowledge base. Based on the linking results, we can determine the knowledge representation and extract a type path for each entity mention, which can be linked to KB. Finally, we use these three representations as input to a hierarchical X-means clustering algorithm[35] and incorporate an optimal partition search algorithm to discover the optimal clustering and typing results.

Representation Generation

General entity representation

Based on Heuristic 1, we can infer the types of most entity mentions. For example, "Mitt Romney" and "John

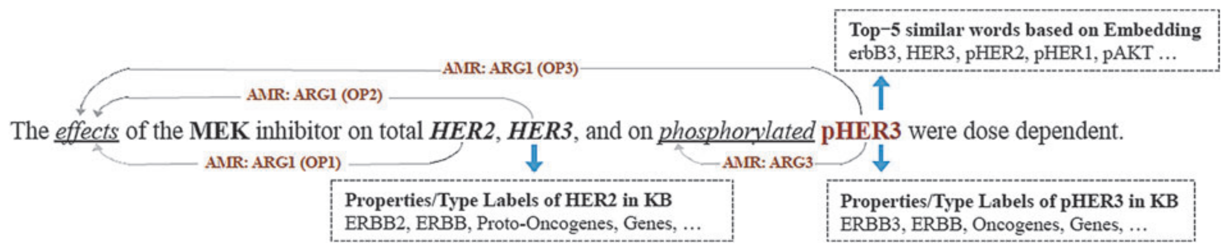

Top−5 similar words based on Embedding
erbB3, HER3, pHER2, pHER1, pAKT …
AMR: ARG1 (OP3)
AMR: ARG1 (OP2)
The effects of the MEK inhibitor on total HER2, HER3, and on phosphorylated pHER3 were dose dependent.
AMR: ARG1 (OP1)
AMR: ARG3
Properties/Type Labels of HER2 in KB
ERBB2, ERBB, Proto-Oncogenes, Genes, …
Properties/Type Labels of pHER3 in KB
ERBB3, ERBB, Oncogenes, Genes, …

FIG. 2. Information that can be used to infer the type of pHER3.

McCain" are both politicians from the same country, "HER2" and "HER3" refer to similar "ERBB (Receptor Tyrosine-Protein Kinase)," and thus they have the same entity type "Enzyme."

We start by capturing the semantic information of entity mentions based on general lexical embedding, which is an effective technique to capture general semantics of words or phrases based on their global contexts. Several models[30,36–38] have been proposed to generate word embeddings. Here, we utilize the Continuous Skip-gram model[36] based on a large amount of unlabeled in-domain data set.

Specific context representation

General embeddings can effectively capture the semantic types of most entity mentions, but many entities are polysemantic and can refer to different types in various contexts. For example, "ADH" in the biomedical domain can be used to refer to an enzyme "alcohol dehydrogenase" or a disease "atypical ductal hyperplasia"; "Yuri Dolgoruky" may refer to a Russian prince or a submarine. In addition, novel or uncommon entities may not exist in the word vocabulary or their semantic embeddings may not be adequately trainable. To solve these problems, based on Heuristic 2, we propose to incorporate specific contexts to infer the entity type.

Considering E2 again, the type of "Yuri Dolgoruky" can be inferred from its context-specific relational concepts such as "nuclear submarines" and "equip." In this work, we use AMR[19] to carefully select the meaningful context words. AMR captures a whole sentence's meaning in a rooted, directed, labeled, and (predominantly) acyclic graph structure. The AMR language contains rich relations, including frame arguments (e.g., *:ARG0* and *:ARG1*), general semantic relations (e.g., *:mod*, *:topic*, and *:domain*), relations for quantities, date entities, or lists (e.g., *:quant*, *:date*, and *:op1*), and the inverse of all these relations (e.g., *:ARG0-of* and *:quant-of*). We carefully select eight entity-related relation types (*:ARG0*, *:ARG1*, *:ARG2*, *:ARG3*, *:conj*, *:domain*, *:topic*, and *:location*) from AMR for entity typing.

Figure 3 depicts the context-specific representation generation for "pHER3" in the example E4. Given an entity mention, for example, "pHER3," we first select its related concepts. For each AMR relation, for example, :ARG1, we generate a representation based on the general embeddings of these related concepts. If a related concept does not exist in the vocabulary, we randomly generate a vector for this concept. If there are several argument concepts involved in a specific relation, we average their representations. For example, we average the representations of "HER3" and "HER2" to get the representation for "Conj" relation. We concatenate the vector representations of all selected relations into one single vector. Although we have carefully aggregated and selected the popular relation types, the representation of each entity mention is still sparse. To reduce the dimensions and generate a high-quality embedding for the specific context, we utilize the sparse autoencoder framework[39] to learn more low-dimensional representations.

Knowledge representation

Existing broad-coverage knowledge bases such as DBpedia, Freebase, or YAGO, as well as domain-specific ontologies such as BioPortal and NCBO can provide useful knowledge for inferring specific fine-grained types. For example, in DBPedia, both properties (e.g., *birthPlace*, *party* for **Mitt Romney**) and type labels (e.g., *Person*, *Governor* for **Mitt Romney**) can be used for entity typing. For the biomedical domain, we can consult BioPortal for domain-specific properties and type labels (e.g., *Oncogenes*, *Genes* for **HER2**). In this work, we construct a knowledge graph based on these properties and type labels

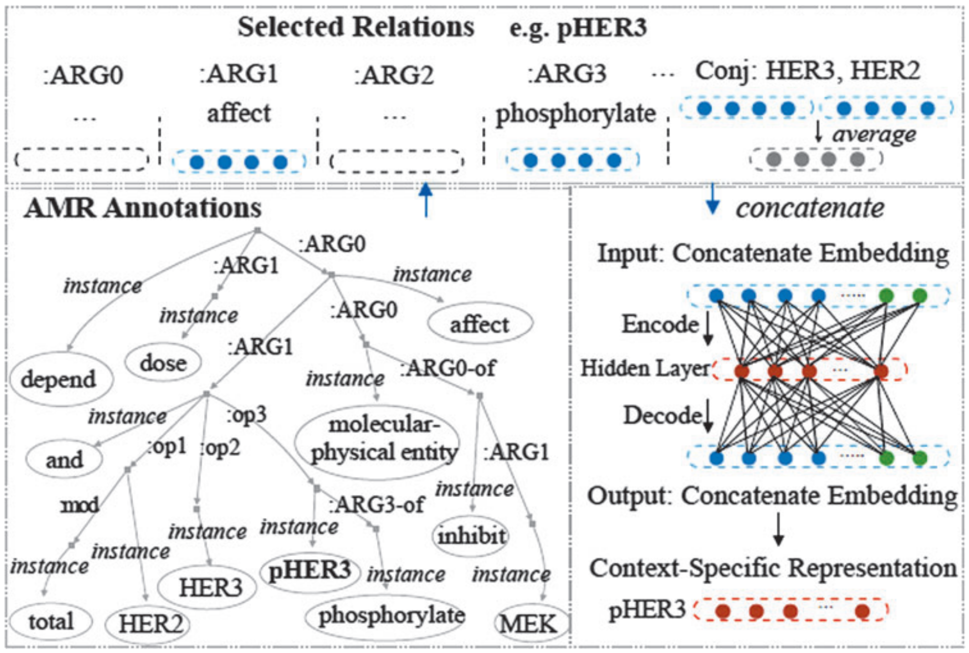


FIG. 3. Context-specific representation generation for pHER3 based on AMR annotation. AMR, abstract meaning representation.

and generate knowledge representations for all entities based on a graph embedding framework.[40] (The details of the graph construction are presented in Appendix 1.)

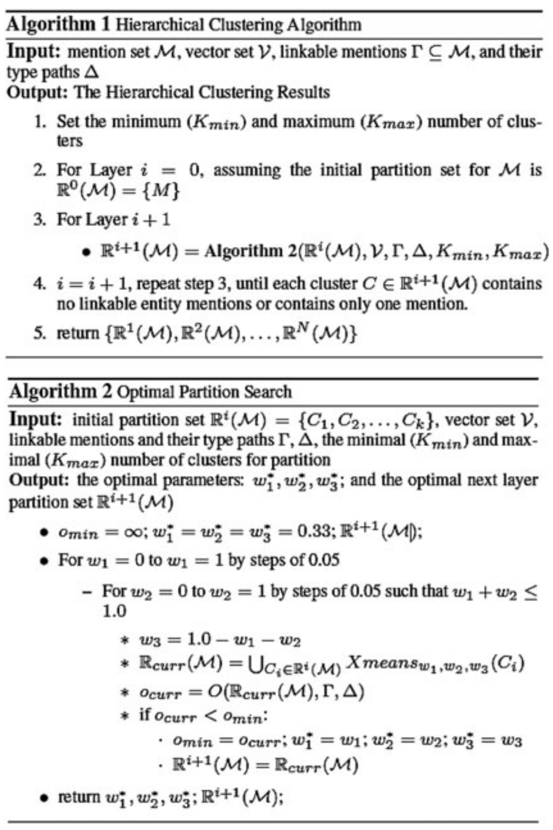
Next we utilize a domain- and language-independent entity linking system[41] to link each mention to existing KBs to determine its knowledge representation. This system is based on an unsupervised collective inference approach and selects the most confident candidate (confidence score >0.95) as the appropriate entity for linking; mentions selected according to this criterion are referred to as *highly linkable* in subsequent sections. If a mention cannot be linked to a KB (i.e., it is not highly linkable), we will assign a random vector as its knowledge representation, and later, this vector will be used for all the similar mentions. In our experiments, about 77.7% entity mentions in the general news domain and about 71.4% in the biomedical domain can be linked to KBs with high confidence.

Joint Linking, Hierarchical Typing, and Naming

Hierarchical typing

For an entity mention $m \in M$, the vector representation v of m is the concatenation of the three parts already mentioned: the distributed general semantic representation v_E, the local context-specific representation v_C, and the knowledge representation v_K based on entity linking. We designed a hierarchical X-means clustering algorithm to detect the hierarchical types of entities. X-means[35] is an extension of the well-known K-means algorithm for which the number of clusters is estimated instead of being fixed by the user. It has two major enhancements compared with K-means: (1) it is fast and scales well with respect to the time it takes to complete each iteration and (2) it can automatically estimate the number of clusters and also obtain local and global optimals for specific data sets. (Details of computing the X-means algorithm are presented in Appendix 2.)

Given the set of all mentions M, we select highly linkable mentions (confidence score >0.95) $\Gamma \subseteq M$ and their corresponding type paths Δ based on the entity linking system described in the Knowledge Representation Section for typing and naming. Here, the type path denotes the longest path from the KB title to the root of the type hierarchy in the KB. For example, we can link the entity mention **Mitt Romney** in the Introduction section to YAGO and extract a type path from the entity title to the root: "Mitt Romney $\rightarrow$ Governor $\rightarrow$ Politician $\rightarrow$ Leader $\rightarrow$ Person $\rightarrow$ Entity." As outlined in Algorithm 1



```
Algorithm 1 Hierarchical Clustering Algorithm
Input: mention set $\mathcal{M}$, vector set $\mathcal{V}$, linkable mentions $\Gamma \subseteq \mathcal{M}$, and their type paths $\Delta$
Output: The Hierarchical Clustering Results
    1. Set the minimum ($K_{min}$) and maximum ($K_{max}$) number of clusters
    2. For Layer $i = 0$, assuming the initial partition set for $\mathcal{M}$ is $\mathbb{R}^0(\mathcal{M}) = \{M\}$
    3. For Layer $i + 1$
        • $\mathbb{R}^{i+1}(\mathcal{M}) = \textbf{Algorithm 2}(\mathbb{R}^i(\mathcal{M}), \mathcal{V}, \Gamma, \Delta, K_{min}, K_{max})$
    4. $i = i + 1$, repeat step 3, until each cluster $C \in \mathbb{R}^{i+1}(\mathcal{M})$ contains no linkable entity mentions or contains only one mention.
    5. return $\{\mathbb{R}^1(\mathcal{M}), \mathbb{R}^2(\mathcal{M}), \ldots, \mathbb{R}^N(\mathcal{M})\}$
```

```
Algorithm 2 Optimal Partition Search
Input: initial partition set $\mathbb{R}^i(\mathcal{M}) = \{C_1, C_2, \ldots, C_k\}$, vector set $\mathcal{V}$, linkable mentions and their type paths $\Gamma$, $\Delta$, the minimal ($K_{min}$) and maximal ($K_{max}$) number of clusters for partition
Output: the optimal parameters: $w_1^*, w_2^*, w_3^*$; and the optimal next layer partition set $\mathbb{R}^{i+1}(\mathcal{M})$
    • $o_{min} = \infty; w_1^* = w_2^* = w_3^* = 0.33; \mathbb{R}^{i+1}(\mathcal{M}\|);$
    • For $w_1 = 0$ to $w_1 = 1$ by steps of 0.05
        – For $w_2 = 0$ to $w_2 = 1$ by steps of 0.05 such that $w_1 + w_2 \leq 1.0$
            * $w_3 = 1.0 - w_1 - w_2$
            * $\mathbb{R}_{curr}(\mathcal{M}) = \bigcup_{C_i \in \mathbb{R}^i(\mathcal{M})} Xmeans_{w_1,w_2,w_3}(C_i)$
            * $o_{curr} = O(\mathbb{R}_{curr}(\mathcal{M}), \Gamma, \Delta)$
            * if $o_{curr} < o_{min}$:
                · $o_{min} = o_{curr}; w_1^* = w_1; w_2^* = w_2; w_3^* = w_3$
                · $\mathbb{R}^{i+1}(\mathcal{M}) = \mathbb{R}_{curr}(\mathcal{M})$
    • return $w_1^*, w_2^*, w_3^*; \mathbb{R}^{i+1}(\mathcal{M})$;
```

FIG. 4. Algorithms 1 and 2.

(Fig. 4), we start from the initial set of all entity mentions M and vector representations V to generate hierarchical partitions $\{\mathbb{R}^{\mathbb{1}}(M), \mathbb{R}^{\mathbb{2}}(M), \ldots, \mathbb{R}^{\mathbb{N}}(M)\}$, where $\mathbb{R}^{\mathbb{i}}(M)$ represents the partition of M based on vector representation set V at layer i.

For each layer i, to get further partition set $\mathbb{R}^{\mathbb{i}+\mathbb{1}}(M)$ based on $\mathbb{R}^{\mathbb{i}}(M) = \{C_1, C_2, \ldots, C_k\}$, we define $Xmeans_{w_1, w_2, w_3}(C)$ as the partition of mention set C based on running X-means with D parameterized by the parameter set w_1, w_2, w_3. It remains to search for the optimal w_1, w_2, w_3. To judge an optimal partition for each layer, we utilize information from the KB: Γ and Δ, as truth and invoke the following heuristic.

Heuristic 4: ***The clustering results of all mentions are optimal when the clustering results of all linkable mentions are optimal.***

We then define an objective function O that evaluates a certain layer of partition set $\mathbb{R}(\mathbb{M}) = \{C_1, C_2, \ldots, C_k\}$:

$$O(\mathbb{R}, \ \Gamma, P) = \mathbb{D}_{\mathbb{inter}} + \mathbb{D}_{\mathbb{intra}},$$

$$\mathbb{D}_{\mathbb{inter}} = \sum_{i \neq j = 1}^{n} \sum_{m'_u \in C_i, \ m'_v \in C_j} w(m'_u, m'_v),$$

$$\mathbb{D}_{\mathbb{intra}} = \sum_{i = 1}^{n} \sum_{m'_u, \ m'_v \in C_i} (1 - w(m'_u, m'_v)),$$

where $w(\cdot)$ is defined in the Knowledge Representation section based on type paths. Algorithm 2 (Fig. 4)encapsulates the search for w_1, w_2, w_3, the parameter set that optimizes O.

Hierarchical typing naming

The entity linking system described in the Knowledge Representation section can extract highly linkable entity mentions and their corresponding type name paths. Considering the examples in the Introduction section again, we can link the entity mention **Mitt Romney** to YAGO and extract a type path from the entity title to the root: "Mitt Romney → Governor → Politician → Leader → Person → Entity." Similarly, we can link **HER2** to "ERBB2" in BioPortal and extract the type name path from the entity to the root of an ontology as: "ERBB2 → Proto-Oncogenes → Oncogenes → Genes → Genome Components → Genome → Phenomena and Processes → Topical Descriptor → MeSH Descriptors." We first normalize the type name paths and remove those two general type name candidates (e.g., "Entity," "Topical Descriptor"). In our experiments, a type name is removed if more than 90% of type paths contain it. Then, we generate the most confident type label n_C for each cluster C based on high-confidence linking results as follows.

For a specific cluster C, the mentions within this cluster are denoted as M_C and the highly linkable mentions are $\Gamma_C \subseteq M_C$. We collect all the type names N_C from the type paths of all $m \in \Gamma_C$, then we determine which type name is the most fine grained and also match with cluster C based on two metrics: Majority and Specificity. Majority is measured based on the frequency of the specific type name n in the type name set N_C. This metric is designed based on our intuition that the type name should be able to represent the types of as many entity mentions as possible. Specificity is designed to measure the granularity degree of the type name in the whole name path. These two metrics are computed as follows:

$$majority_n^C = Count(n, C)/|M_C|,$$

$$specificity_n^{p(n)} = Position(n, p(n))/|p(n)|,$$

where $Count(n, C)$ represents the frequency of a type name n in the set N_C, $|M_C|$ represents the number of members in cluster C, $p(n)$ represents the longest type name path, including n, and $Position(n, p(n))$ represents the position of n in $p(n)$ (from the root to n).

We combine these two metrics and choose n_C as follows: for each cluster C we define $N_C^m = \{n : n \in N_C \wedge majority_n^C \geq \lambda\}$, where λ is a threshold parameter (we set λ to 0.75 in our experiments). We then select $n_C = argmax\ specificity_n^{p(n)}$. For example, if the majority of *Proto-Oncogenes* and *Genes* are both larger than λ, we should choose *Proto-Oncogenes* because it is much more fine grained than *Genes* in the whole type name path. After naming for each hierarchical cluster, we will generate a type hierarchy, which is also customized for the specific corpus.

Experiments and Evaluation

In this section, we present an evaluation of the proposed framework on various genres, domains, and languages, as well as a comparison with state-of-the-art systems.

Data

We first introduce the data sets for our experiments. To compare the performance of our framework against state-of-the-art name taggers and evaluate its effectiveness on various domains and genres, we first conduct experiments on AMR data sets, which include perfect mention boundaries with fine-grained entity types. For the experiment on multiple languages, we use data sets from the DARPA LORELEI program and foreign news agencies. The detailed data statistics are summarized in Table 1.

As our approach is based on word embeddings, which need to be trained from a large corpus of unlabeled in-domain articles, we collect all the English and Japanese articles from the August 11, 2014, Wikipedia dump to learn English and Japanese word/phrase embeddings and collect all the articles of the 4th edition of the Chinese Gigaword Corpus to learn Chinese word/phrase embeddings. For the biomedical domain, we utilize the word2vec model, which is trained based on all article abstracts from PubMed and full-text documents from the PubMed Central Open Access subset. We also collect all entities and their properties and type labels from DBpedia and more than 300 biomedical domain-specific ontologies crawled from BioPortal[42] to learn knowledge embeddings.

Table 1. Statistics of test data sets

	No. of docs	*No. of mentions*	*No. of types*
English news	367	15,002	183
Biomedical articles	14	2055	51
Discussion forum	329	4157	149
Chinese news	20	1683	44
Japanese news	25	489	47
Hausa news	90	1508	3
Yoruba news	239	7456	2

Evaluation metrics

Our framework can automatically discover many fine-grained types. Some of the types can be mapped to the human annotated types, while some cannot. Therefore, in addition to mention-level precision, recall, and F-measure, we also exploit standard clustering measures of purity, F-measure, and entropy to evaluate the performance of new entity types (which are defined in Appendix 3).

Comparison with state-of-the-art systems

We compare with two high-performing name taggers, Stanford NER[43] and FIGER,[16] on both coarse-grained types (person, location, and organization) and fine-grained types. We utilize the AMR parser developed by Flanigan et al.[44] and manually map AMR types and system-generated types to three coarse-grained types. To compare identification results, we design a simple mention boundary detection approach based on capitalization features and part-of-speech features. We compare the performance of our system with both perfect AMR and system AMR annotations with the performance of NER and FIGER. We conduct the experiments on English news data set and link entity mentions to DBPedia.[45] The mention-level F-scores are shown in Table 2.

Besides these three coarse-grained types, there are also many new types (e.g., vehicle and medium) discovered by fine-grained entity typing approaches. We compare our framework with FIGER based on its

Table 2. Coarse-grained mention-level F-score comparison

Layer (no. of clusters)	*System*[a]	*System*[b]	*Stanford NER*	*FIGER*[a]
L1 (5)	0.649	0.628	0.712	0.663
L2 (21)	0.668	0.647		
L3 (92)	0.689	0.681		
L4 (146)	0.713	0.709		
L5 (201)	0.728	0.721		

[a]Based on perfect AMR.
[b]Based on system AMR.
AMR, abstract meaning representation.

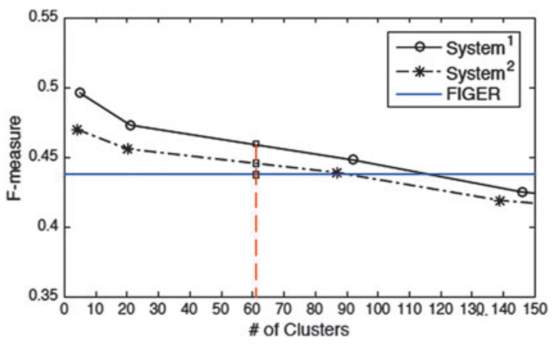


FIG. 5. Fine-grained cluster level F-score comparison (the red dashed line shows the same number of clusters for comparison).

112-classes classification model. The cluster-level F-scores are shown in Figure 5.

From Table 2 we can see that, on coarse-grained level, compared with Stanford NER, which contains many features and is trained on about 945 annotated documents ($\sim$203,000 tokens), our approach with both system AMR and perfect AMR achieved comparable performance. Compared with FIGER on coarse-grained level, our approach with system AMR and perfect AMR also achieved better results. Figure 5 shows the fine-grained level performance. The number of clusters, to some extent, can reflect the granularity of fine-grained typing. Although we cannot directly map the granularity of FIGER to our system, considering that the classification results of FIGER are highly biased toward a certain subset of types (about 60 types), our approach with both system AMR and perfect AMR should slightly outperform FIGER, which is trained based on 2 million labeled sentences.

Both Stanford NER and FIGER heavily rely on linguistic features, such as tokens, context n-grams, and part-of-speech tags, to predict entity types. Compared with lexical information, semantic information is more indicative to infer its type. For example, in ***Bernama*** *said Malaysia had procured very short-range air defense systems from Russia,* ***Bernama*** is assigned the type *Person* by the FIGER system. However, based on general semantic information, the most similar concepts to ***Bernama*** include ***Malaysiakini***, ***Utusan***, and ***Kompas***, which can effectively help infer the correct type as *News Agency*. In addition, in many cases, the fine-grained types of entity mentions are heavily dependent on their knowledge information. For example, in ***Antonis Samaras*** *is cheered by supporters after his statement in Athens June 17, 2012,* it is difficult to infer the fine-grained type of ***Antonis Samaras*** based on context words. However, we can utilize more knowledge from KBs and find that the most similar concepts to ***Antonis Samaras*** include ***Kostas Karamanlis***, ***Georgios Papastamkos***, and ***Giannis Valinakis*** based on knowledge representation, which can help infer the fine-grained type of ***Antonis Samaras*** as *Politician*.

Comparison on genres

For comparison between news and discussion forum genres, we utilize perfect entity boundaries and perfect AMR annotation results to model local contexts and link entity mentions to DBpedia.[45] Figure 6 shows the performance.

We can see that our system performs much better on news articles than discussion forum posts, because of two reasons: (1) many entities occur as abbreviations in discussion forum posts, which brings challenges to both entity typing and linking. For example, in the following post: *The joke will be on* ***House Dems*** *who are being promised a bill to "fix" the problems with the Senate bill.*, it is difficult to generate accurate general semantic and knowledge representations for the mentions such as ***House*** (which refers to *United States House of Representatives*) and ***Dems*** (which refers to *Democratic Party of United States*). (2) More novel and uncommon entities appear in discussion forums. Take the following sentence as an example: *Mend some fences and get this country moving. He could call it* ***APOLOGIES ON BEER****. Hell, sell tickets and hire the Chinese to cater the event.* ***APOLOGIES ON BEER*** is a novel emerging entity, thus it will be difficult to predict its fine-grained type *tour*, even with semantic and knowledge representations.

In addition, our system can outperform the FIGER system, of which the results are focused on about 50 types on the discussion forum data set, on both Purity and F-measure. As discussed in the Comparison with State-of-the-Art Systems section, FIGER is trained based on a rich set of linguistic features. When it is applied to a new informal genre, feature generation cannot be guaranteed to work well. Our system is mainly based on semantic representations, which will not be affected by the noise.

Comparison on domains

To demonstrate the domain portability of our framework, we take the biomedical domain as a case study.

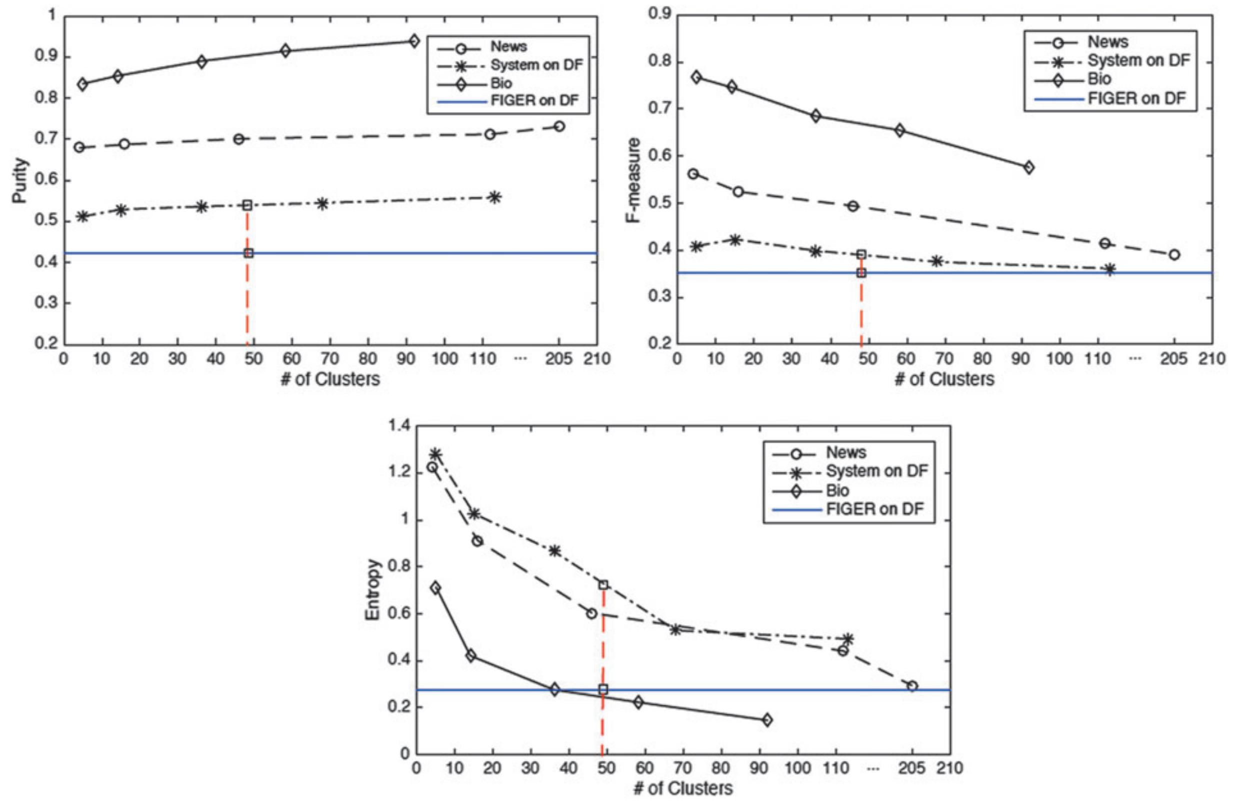


FIG. 6. Typing results for different genres and domains with perfect AMR (the red dashed line shows the same number of clusters for comparison).

For fair comparison, we used perfect AMR semantic graphs and perfect mention boundaries. Figure 6 compares the performance for news and biomedical articles.

As shown in Figure 6, our system performs much better on biomedical data than on general news data. In an in-depth analysis of the experiment results, we found that most of the entity mentions in the biomedical domain are unique and unambiguous, and the mentions with the same type often share the same name strings. For example, ***HER2***, ***HER3***, and ***HER4*** refer to similar *Proto-Oncogenes*; ***A-RAF***, ***B-RAF***, and ***C-RAF*** share the same type *RAF Kinases*. However, it is always the opposite in the general news domain. For example, although ***Shenzhen***, ***Shenzhen Maoye***, and ***Shenzhen Gymnasium*** share the same name string *Shenzhen*, they have different entity types: ***Shenzhen*** refers to a city, ***Shenzhen Maoye*** is a company, and ***Shenzhen Gymnasium*** is a facility. What is more, ambiguity commonly exists in general news domain, especially for persons and locations. For example, both of ***Sokolov*** and ***Chamberlain*** can refer to a *city* or a *person*. We utilize the ambiguity measure defined in Ji et al.[46] as the criteria to demonstrate the ambiguity degree of news and biomedical domains.

$$ambiguity = \frac{\#name\ strings\ belong\ to\ >\ 1\ cluster}{\#name\ strings}.$$

Figure 7 shows the ambiguity comparison results between the general news and biomedical domains. Owing to the low ambiguity of the biomedical domain, the general semantic representation and knowledge representation can better capture the domain-specific types of these entity mentions. This analysis can also be verified by the final optimal weights for three kinds of representations $w_1 = 0.45$, $w_2 = 0.05$ $w_3 = 0.5$ for biomedical domain, whereas $w_1 = 0.45$, $w_2 = 0.2$ $w_3 = 0.35$ for news domain, which shows the different contributions of three-layer representations for entity typing.

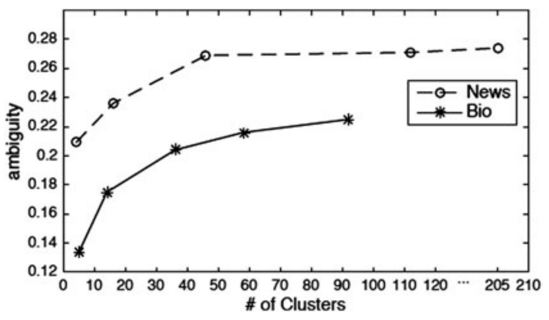


FIG. 7. Ambiguity comparison for different domains.

Comparison on languages

Our framework is also highly portable to new languages. Different languages may have different linguistic resources available. For example, English has rich linguistic resources (e.g., AMR) that can be utilized to model local contexts, whereas some languages (e.g., Chinese and Japanese) do not. To evaluate the impact of the local contexts on entity typing, we compare the performance based on AMR and the embeddings of context words that occur within a limited-size window. In our experiment, the window size is 6. Figure 8 shows the performance on English, Chinese, and Japanese news data sets.

Figure 8 shows that our framework on Chinese and Japanese also achieved comparable performance as English. The main reason is that entities in Chinese and Japanese have less ambiguity than English. Almost all of the same name strings refer to the same type of entity. Based on the ambiguity measure in the Comparison on Domains section, the ambiguity is lower than 0.05 both for Chinese and Japanese.

In addition, for low resource languages, there are not enough unlabeled documents to train word embeddings, and KBs may not be available for these languages. In this case, we can utilize other feature representations such as bag-of-words *tf-idf* instead of embedding-based representations. To prove this, we

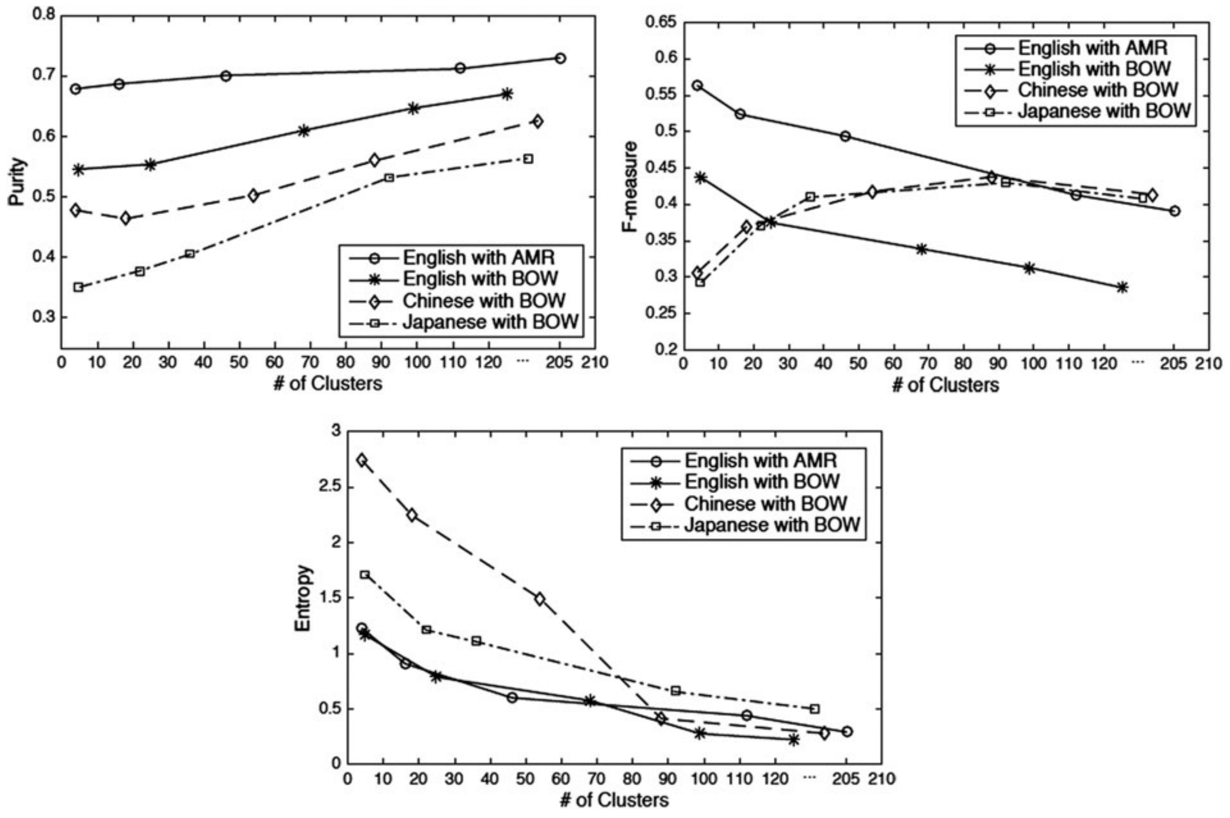


FIG. 8. Typing results for various languages.

apply our framework to two low-resource languages: Hausa and Yoruba. The mention-level typing accuracy with perfect boundary is very promising: 85.42% for Hausa and 72.26% for Yoruba.

Conclusions and Future Work

In this work, we demonstrated a new Liberal IE paradigm. Using fine-grained entity typing task as a study case, for the first time, we show an unsupervised framework, which incorporates entity general semantics, specific contexts, and domain-specific knowledge to discover the fine-grained types. This framework takes the human out of the loop and requires no annotated data or predefined types. Without the needs of language-specific features and resources, this framework can be easily adapted to other domains, genres, and languages. We also incorporate a domain- and language-independent unsupervised entity linking system to improve the clustering performance and discover corpus-customized domain-specific fine-grained typing schema.

Our framework achieves performance comparable to state-of-the-art entity typing systems trained from a large amount of labeled data. The results are encouraging considering the simplicity of our system. In ongoing research, we are extending the Liberal Information Extraction framework to other tasks, such as Event Extraction and Relation Extraction, to automatically induce schemas without the need for predefined types and human annotation.

Acknowledgments

We would like to thank Kevin Knight and Jonathan May (ISI) for sharing biomedical AMR annotations. This work was supported by the U.S. ARL NS-CTA No. W911NF-09-2-0053 and DARPA DEFT No. FA8750-13-2-0041, and in part by NSF IIS-1523198, IIS-1017362, IIS-1320617 and IIS-1354329, and NIH BD2K grant 1U54GM114838.

Author Disclosure Statement

No competing financial interests exist. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

1. Grishman R, Sundheim B. Message understanding conference-6: A brief history. In: COLING, 1996, pp. 466–471.
2. Bunescu R, Mooney RJ. A shortest path dependency kernel for relation extraction. In: HLT-EMNLP, 2005, pp. 724–731.
3. Bunescu R, Pasca M. Using encyclopedic knowledge for named entity disambiguation. In: EACL, Volume 6, 2006, pp. 9–16.
4. Corro LD, Abujabal A, Gemulla R, Weikum G. Finet: Context-aware fine-grained named entity typing. In: EMNLP, 2015, pp. 868–878.
5. Culotta A, Sorensen J. Dependency tree kernels for relation extraction. In: ACL, 2004, pp. 423–429.
6. Durrett G, Klein D. A joint model for entity analysis: Coreference, typing, and linking. In: TACL, 2014, pp. 477–490.
7. Han X, Zhao J. Named entity disambiguation by leveraging wikipedia semantic knowledge. In: CIKM, 2009, pp. 215–224.
8. Hasegawa T, Sekine S, Grishman R. Discovering relations among named entities from large corpora. In: ACL, 2004, pp. 415–422.
9. Lin T, Etzioni O, et al. Entity linking at web scale. In Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction, 2012, pp. 84–88.
10. Rodríguez MA, Egenhofer MJ. Determining semantic similarity among entity classes from different ontologies. In: TKDE, 2003, pp. 442–456.
11. Balog K, Neumayer R. Hierarchical target type identification for entity-oriented queries. In: CIKM, 2012, pp. 2391–2394.
12. Bazzanella B, Stoermer H, Bouquet P. Searching for individual entities: A query analysis. In: IRI, 2010, pp. 115–120.
13. Mollá D, Van Zaanen M, Smith D, et al. Named entity recognition for question answering. Proceedings of the 2006 Australasian Language Technology Workshop, 2006, pp. 51–58.
14. Sun H, Ma H, Yih W.-t, et al. C hang. Open domain question answering via semantic enrichment. In: WWW, 2015, pp. 1045–1055.
15. Lee C, Hwang Y.-G, Oh H.-J, et al. Fine-grained named entity recognition using conditional random fields for question answering. In: Information Retrieval Technology, 2006, pp. 581–587.
16. Xiao L, Weld DS. Fine-grained entity recognition. In AAAI, 2012, pp. 94–100.
17. Fleischman M, Hovy E. Fine grained classification of named entities. In: COLING, 2002, pp. 1–7.
18. Sekine S. Extended named entity ontology with attribute information. In: LREC, 2008, pp. 52–57.
19. Banarescu L, Bonial C, Cai S, et al. Abstract meaning representation for sembanking. In: ACL Workshop on Linguistic Annotation and Interoperability with Discourse, 2013, pp. 178–186.
20. Bollacker K, Evans C, Paritosh P, et al. Freebase: A collaboratively created graph database for structuring human knowledge. In: SIGMOD, 2008, pp. 1247–1249.
21. Lafferty J, McCallum A, Pereira FC. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: ICML, 2001, pp. 282–289.
22. Gillick D, Lazic N, Ganchev K, et al. Context-dependent fine-grained entity type tagging. In: CoRR, 2014, pp. 1–9.
23. Yogatama D, Gillick D, Lazic N. Embedding methods for fine grained entity type classification. In: ACL-IJCNLP, 2016, pp. 291–296.
24. Lin T, Etzioni O, et al. No noun phrase left behind: Detecting and typing unlinkable entities. In: ACL, 2012, pp. 893–903.
25. Yosef A, Bauer S, Hoffart J, et al. Hyena: Hierarchical type classification for entity names. In: COLING, 2012, pp. 1361–1370.
26. Hoffart J, Suchanek FM, Berberich K, Weikum G. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. In: IJCAI, 2013, pp. 28–61.
27. Turian J, Ratinov L, Bengio Y. Word representations: A simple and general method for semi-supervised learning. In: ACL, 2010, pp. 384–394.
28. Mikolov T, Yih W.-T, Zweig G. Linguistic regularities in continuous space word representations. In: HLT-NAACL, 2013, pp. 746–751.
29. Yin W, Schütze H. An exploration of embeddings for generalized phrases. In: ACL Workshop on Student Research, 2014, pp. 41–47.
30. Mitchell J, Lapata M. Composition in distributional models of semantics. Cogn Sci 2010;34:1388–1429.
31. Huang L, Cassidy T, Feng X, et al. Liberal event extraction and event schema induction. In: ACL, 2016, pp. 258–268.
32. Hermann K, Blunsom P. The role of syntax in vector space models of compositional semantics. In: ACL, 2013, pp. 894–904.

33. Socher R, Karpathy A, Le Q, et al. Grounded compositional semantics for finding and describing images with sentences. In TACL, 2013, pp. 207–218.
34. Socher R, Perelygin A, Wu J, et al. Recursive deep models for semantic compositionality over a sentiment treebank. In: EMNLP, 2013, pp. 1642–1653.
35. Pelleg D, Moore A. X-means: Extending k-means with efficient estimation of the number of clusters. In: ZCML, 2000, pp. 727–734.
36. Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space. In: CoRR, 2013, pp. 1–12.
37. Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality. In: NIPS, 2013, pp. 3111–3119.
38. Zhao Y, Huang S, Dai X, et al. Learning word embeddings from dependency relations. In: IALP, 2014, pp. 123–127.
39. Ng A. Sparse autoencoder. CS294A Lecture Notes, 72, 2011, pp. 1–19.
40. Tang J, Qu M, Wang M, et al. Line: Large-scale information network embedding. In: WWW, 2015, pp. 1067–1077.
41. Wang H, Zheng JG, Ma X, et al. Language and domain independent entity linking with quantified collective validation. In: EMNLP, 2015, pp. 695–704.
42. Zheng JG, Howsmon D, Zhang B, et al. Entity linking for biomedical literature. In: BMC Medical Informatics and Decision Making, 2014, pp. 1–9.
43. Finkel JR, Grenager T, Manning C. Incorporating non-local information into information extraction systems by gibbs sampling. In: ACL, 2005, pp. 363–370.
44. Flanigan J, Thomson S, Carbonell J, et al. A discriminative graph-based parser for the abstract meaning representation. In: ACL, 2014, pp. 1426–1436.
45. Pan X, Cassidy T, Hermjakob U, et al. Unsupervised entity linking with abstract meaning representation. In: HLT-NAACL, 2015, pp. 1130–1139.
46. Ji H, Grishman R, Dang HT, et al. Overview of the tac 2010 knowledge base population track. In: TAC, 2010. pp. 3–35.

Cite this article as: Huang L, May J, Pan X, Ji H, Ren X, Han J, Zhao L, Hendler JA (2017) Liberal entity extraction: rapid construction of fine-grained entity typing systems. *Big Data* 5:1, 19–31, DOI: 10.1089/big.2017.0012.

Abbreviations Used

AMR	= abstract meaning representation
IE	= information extraction
NLP	= natural language processing

Appendix 1

Graph Construction Details

We construct a weighted undirected graph $G=(\mathrm{E},\ \mathrm{D})$ from all domain-specific KBs, where E is the set of entities and $d_{ij} \in D$ implies that two entities, e_i and e_j, share some common properties or type labels. The weight of $d_{ij},\ w_{ij}$ is calculated as

$$w_{ij}=\frac{|p_i \cap p_j|}{max(|p_i|,\ |p_j|)},$$

where p_i is the set of property and type labels of e_i.

After constructing the knowledge graph, we apply the graph embedding framework proposed by Tang et al.[40] to generate a knowledge representation for each entity. The basic idea is to optimize the preservation of both local structures, which are represented by the observed edges in the graph, and global graph structures, which are determined through the shared neighborhood structures of the entities. Two objectives that preserve the local and global structures are as follows:

$$O_{local}=-\sum_{d_{ij}\in \mathrm{D}} w_{ij}log(p_1(e_i,e_j)),$$

$$p_1(e_i,e_j)=\frac{1}{1+exp(-u_i^T u_i)},$$

$$O_{global}=-\sum_{d_{ij}\in \mathrm{D}} w_{ij}log(p_2(e_j|e_i)),$$

$$p_2(e_j|e_i)=\frac{exp(u_j'Tu_i)}{\sum_{k=1}^{|E|} exp(u_k'Tu_i)},$$

where p_1 and p_2 are joint and conditional probabilities, respectively, u_1 is the low-dimensional vector representation of e_i, and u_i' is the representation of e_i when it is treated as a specific "context." We randomly initialize all u_i and u_i'. By tuning u_i and minimizing O_{local}, we can generate a representation for each entity. To train O_{global} efficiently, we adopt a negative sampling approach. Let K be a set of "noise" entities drawn with replacement from $E \propto g_k^{3/4}$, where g_k is the degree of $e_k \in E$. We thus define the objective for d_{ij} as

$$log\sigma(u_j'T\cdot u_i)+\sum_{k=1}^{|K|} log\sigma(-u_j'T\cdot u_i),$$

where $\sigma(x)=1/(1+exp(-x))$ is the sigmoid function. The first term models the observed edges, whereas the second term models the negative edges in the noise sample K_i. To optimize the negative sampling objective function, we adopt the asynchronous stochastic gradient algorithm. After training O_{local} and O_{global}

separately, we concatenate the optimized embeddings for each entity as their knowledge representations.

Appendix 2

X-Means Metric

X-means requires a distance metric D to calculate its clusters. We define D given two entity mentions m_1 and m_2 with vector representations v_1 and v_2, respectively (we regard the vector of the cluster centroid as the same as the vector of the mention), as

$$D(m_1,\ m_2) = w_1 \cdot D(v_E^{m_1}, v_E^{m_2}) + w_2 \cdot D\left(v_C^{m_1}, v_C^{m_2}\right) + w_3 \cdot D(v_K^{m_1}, v_K^{m_2}),$$

where $D(\cdot)$ is the Euclidean distance between two vectors, w_1,w_2, and w_3 are the balance parameters among three types of representations, and

$$w_1 + w_2 + w_3 = 1.$$

Appendix 3

Metrics

(1) Purity: To compute purity, each system cluster is assigned to the reference class with the greatest mention overlap. The sum of all matching mentions given this assignment is then divided by N.

$$purity = \frac{\sum_{j=1}^{K} max_{1\le i\le M}\left|C_j \cap R_i\right|}{N},$$

where N is the total number of mentions, K is the number of system-generated clusters, M is the number of clusters in the answer key, C_j is the set of mentions in the j^{th} cluster in our system, and R_i is the set of mentions for the i^{th} type in the answer key.

(2) Precision, recall, F-measure (F)

Here, we utilize F-measure to evaluate the entity mentions that match with the ground truth data set.

$$prec\left(R_i, C_j\right) = \left|R_i \cap C_j\right| / C_j,$$

$$rec\left(R_i, C_j\right) = \left|R_i \cap C_j\right| / R_i,$$

$$F - measure\left(R_i, C_j\right) = \frac{2 \times prec\left(R_i, C_j\right) \times rec\left(R_i, C_j\right)}{rec\left(R_i, C_j\right) + rec\left(R_i, C_j\right)}.$$

Intuitively, $F - measure\left(R_i, C_j\right)$ measures how good a class R_i can be described by a cluster C_j and the success of capturing a class R_i is measured by using the "best" cluster C_j for R_i, that is, C_j that maximizes $F - measure\left(R_i, C_j\right)$.

$$F - measure = \frac{\sum_{i=1}^{M} |R_i| max_{1\le j\le K} F - measure\left(R_i, C_j\right)}{N}.$$

(3) Entropy

The entropy measures the diversity of a cluster C_j and it is defined as

$$entropy\left(C_j\right) = -\sum_{i=1}^{M} P(i,j) \times logP(i,j),$$

$$P(i,j) = \frac{R_i \cap C_j}{N},$$

where $P(i,j)$ represents the probability of a mention in the key cluster R_i being assigned to the system cluster C_j. The lower the value of entropy, the better the clustering is. The overall cluster entropy is then computed by averaging the entropy of all clusters:

$$entropy = \frac{\sum_{i=1}^{K} \left|C_j\right| \times entropy\left(C_j\right)}{N}.$$