ClusType: Effective Entity Recognition and Typing by Relation Phrase-Based Clustering

Xiang Ren† Ahmed El-Kishky† Chi Wang‡ Fangbo Tao† Clare R. Voss ⋆ Heng Ji ♯ Jiawei Han†
† University of Illinois at Urbana-Champaign, Urbana, IL, USA
‡ Microsoft Research, Redmond, WA, USA
♯ Computer Science Department, Rensselaer Polytechnic Institute, USA
⋆ Information Sciences Directorate, Army Research Laboratory, Adelphi, MD, USA
†{xren7, elkishk2, ftao2, hanj}@illinois.edu ‡chiw@microsoft.com ⋆clare.r.voss.civ@mail.mil ♯jih@rpi.edu

ABSTRACT

Entity recognition is an important but challenging research problem. In reality, many text collections are from specific, dynamic, or emerging domains, which poses significant new challenges for entity recognition with increase in name ambiguity and context sparsity, requiring entity detection without domain restriction. In this paper, we investigate entity recognition (ER) with distant-supervision and propose a novel relation phrase-based ER framework, called **ClusType**, that runs *data-driven* phrase mining to generate entity mention candidates and relation phrases, and enforces the principle that relation phrases should be *softly* clustered when propagating type information between their argument entities. Then we predict the type of *each* entity mention based on the type signatures of its co-occurring relation phrases and the type indicators of its surface name, as computed over the corpus. Specifically, we formulate a joint optimization problem for two tasks, *type propagation with relation phrases* and *multi-view relation phrase clustering*. Our experiments on multiple genres—news, Yelp reviews and tweets—demonstrate the effectiveness and robustness of ClusType, with an average of 37% improvement in F1 score over the best compared method.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database applications—*Data mining*

Keywords

Entity Recognition and Typing; Relation Phrase Clustering;

1. INTRODUCTION

Entity recognition is an important task in text analysis. Identifying token spans as entity mentions in documents and labeling their types (*e.g.*, people, product or food) enables effective structured analysis of unstructured text corpus. The extracted entity information can be used in a variety of ways (*e.g.*, to serve as primitives for information extraction [20] and knowledge base (KB) population [2]. Traditional named entity recognition systems [18, 15] are usually designed for several major types (*e.g.*, person, organization, location) and general domains (*e.g.*, news), and so require additional steps for adaptation to a new domain and new types.

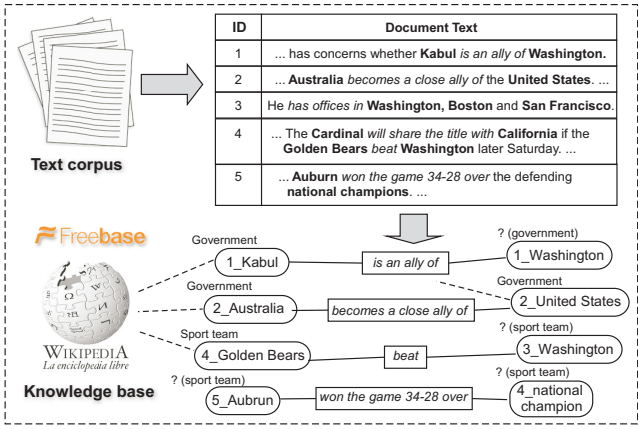


Figure 1: An example of distant supervision.

Entity linking techniques [21] map from given entity mentions detected in text to entities in KBs like Freebase [1], where type information can be collected. But most of such information is manually curated, and thus the set of entities so obtained is of limited coverage and freshness (*e.g.*, over 50% entities mentioned in Web documents are unlinkable [11]). The rapid emergence of large, domain-specific text corpora (*e.g.*, product reviews) poses significant challenges to traditional entity recognition and entity linking techniques and calls for methods of recognizing entity mentions of target types with minimal or no human supervision, and with no requirement that entities can be found in a KB.

There are broadly two kinds of efforts towards that goal: weak supervision and distant supervision. Weak supervision relies on manually-specified seed entity names in applying pattern-based bootstrapping methods [7, 9] or label propagation methods [24] to identify more entities of each type. Both methods assume the seed entities are unambiguous and sufficiently frequent in the corpus, which requires careful seed entity selection by human [10]. Distant supervision is a more recent trend, aiming to reduce expensive human labor by utilizing entity information in KBs [16, 11] (see Fig. 1). The typical workflow is: i) detect entity mentions from a corpus, ii) map candidate mentions to KB entities of target types, and iii) use those confidently mapped {mention, type} pairs as labeled data to infer the types of remaining candidate mentions.

KDD'15, August 10-13, 2015, Sydney, NSW, Australia.

DOI: http://dx.doi.org/10.1145/2783258.2783362.

In this paper, we study the problem of *distantly-supervised entity recognition in a domain-specific corpus*: Given a domain-specific corpus and a set of target entity types from a KB, we aim to effectively and efficiently detect entity mentions from that corpus, and categorize each by target types or Not-Of-Interest (NOI), with distant supervision. Existing distant supervision methods encounter the following limitations when handling a large, domain-specific corpus.

• **Domain Restriction:** They assume entity mentions are already extracted by existing entity detection tools such as noun phrase chunkers. These tools are usually trained on general-domain corpora like news articles (clean, grammatical) and make use of various linguistic features, but do not work well on specific, dynamic or emerging domains (*e.g.*, tweets or restaurant reviews).

• **Name Ambiguity:** Entity names are often ambiguous—multiple entities may share the same surface name. In Fig. 1, for example, the surface name "*Washington*" can refer to either the U.S. government, a sport team, or the U.S. capital city. However, most existing studies [22, 9] simply output a type distribution for each surface name, instead of an exact type for *each* mention of the entity.

• **Context Sparsity:** Previous methods have difficulties in handling entity mentions with sparse context. They leverage a variety of contextual clues to find sources of shared semantics across different entities, including keywords [24], Wikipedia concepts [22], linguistic patterns [16] and textual relations [11]. However, there are often many ways to describe even the same relation between two entities (*e.g.*, "*beat*" and "*won the game 34-28 over*" in Fig. 1). This poses challenges on typing entity mentions when they are isolated from other entities or only share infrequent (sparse) context.

We address these challenges with several intuitive ideas. First, to address the domain restriction, we consider a domain-agnostic phrase mining algorithm to extract entity mention candidates with minimal dependence of linguistic assumption (*e.g.*, part-of-speech (POS) tagging requires fewer assumptions of the linguistic characteristics of a domain than semantic parsing). Second, to address the name ambiguity, we do not simply merge the entity mention candidates with identical surface names but model each of them based on its surface name and contexts. Third, to address the context sparsity, we mine *relation phrases* co-occurring with the mention candidates, and infer synonymous relation phrases which share similar type signatures (*i.e.*, express similar types of entities as arguments). This helps to form connecting bridges among entities that do not share identical context, but share synonymous relation phrases.

To systematically integrate these ideas, we develop a novel solution called **ClusType**. First, it mines both entity mention candidates and relation phrases by POS-constrained phrase segmentation; this demonstrates great cross-domain performance (Sec. 3.1). Second, it constructs a heterogeneous graph to faithfully represent candidate entity mentions, entity surface names, and relation phrases and their relationship types in a unified form (see Fig. 2). The entity mentions are kept as individual objects to be disambiguated, and linked to surface names and relation phrases (Sec. 3.2-3.4). With the heterogeneous graph, we formulate a graph-based semi-supervised learning of two tasks jointly: (1) type propagation on graph, and (2) relation phrase clustering. By clustering synonymous relation phrases, we can propagate types among entities bridged via these synonymous relation phrases. Conversely, derived entity argument types serve as good features for clustering relation phrases. These two tasks mutually enhance each other and lead to

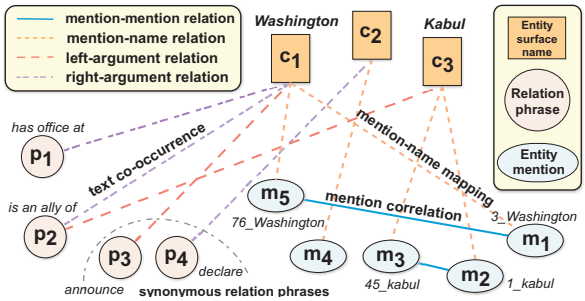


Figure 2: The constructed heterogeneous graph.

quality recognition of unlinkable entity mentions. In this paper, we present an alternating minimization algorithm to efficiently solve the joint optimization problem, which iterates between type propagation and relation phrase clustering (Sec. 4). To our knowledge, this is the first work to *integrate entity recognition with textual relation clustering*.

The major novel contributions of this paper are as follows: (1) we develop an efficient, domain-independent phrase mining algorithm for entity mention candidate and relation phrase extraction; (2) we propose a relation phrase-based entity recognition approach which models the type of each entity mention in a scalable way and softly clusters relation phrases, to resolve name ambiguity and context sparsity issues; (3) we formulate a joint optimization problem for clustering-integrated type propagation; and (4) our experiments on three datasets of different genres—news, Yelp reviews and tweets— demonstrate that the proposed method achieves significant improvement over the state-of-the-art (*e.g.*, 58.3% enhancement in F1 on the Yelp dataset over the best competitor from existing work).

2. PROBLEM DEFINITION

The input to our proposed ER framework is a document collection $\mathcal{D}$, a knowledge base Ψ with type schema $\mathcal{T}_\Psi$, and a *target type set* $\mathcal{T} \subset \mathcal{T}_\Psi$. In this work, we use the type schema of Freebase [1] and assume $\mathcal{T}$ is covered by Freebase.

An *entity mention*, m, is a token span in the text document which refers to a real-world entity e. Let c_m denote the *surface name* of m. In practice, people may use multiple surface names to refer to the same entity (*e.g.*, "*black mamba*" and "*KB*" for *Kobe Bryant*). On the other hand, a surface name c could refer to different entities (*e.g.*, "*Washington*" in Fig. 1). Moreover, even though an entity e can have multiple types (*e.g.*, *J.F.K. airport* is both a location and an organization), the type of its specific mention m is usually unambiguous. We use a type indicator vector $\mathbf{y}_m \in \{0,1\}^T$ to denote the entity type for each mention m, where $T = |\mathcal{T}| + 1$, *i.e.*, m has type $t \in \mathcal{T}$ or is Not-of-Interest (NOI). By estimating $\mathbf{y}_m$, one can predict type of m as type $(m) = \text{argmax}_{1 \le i \le T}\ y_{m,i}$.

Extracting textual relations from documents has been previously studied [4] and applied to entity typing [16, 11]. A *relation phrase* is a phrase that denotes a unary or binary relation in a sentence [4] (see Fig. 3 for example). We leverage the rich semantics embedded in relation phrases to provide type cues for their entity arguments. Specifically, we define the *type signature* of a relation phrase p as two indicator vectors $\mathbf{p}_L, \mathbf{p}_R \in \mathbb{R}^T$. They measure how likely the left/right entity arguments of p belong to different types ($\mathcal{T}$ or NOI). A large positive value on $p_{L,t}$ ($p_{R,t}$) indicates that the left/right argument of p is likely of type t.

Let $\mathcal{M} = \{m_1, ..., m_M\}$ denote the set of M candidate entity mentions extracted from $\mathcal{D}$. Suppose a subset of entity

Over:RP the weekend the system:EP dropped:RP nearly inches of snow in:RP western Oklahoma:EP and at:RP [Dallas Fort Worth International Airport]:EP sleet and ice caused:RP hundreds of [flight cancellations]:EP and delays. ..... It is forecast:RP to reach:RP [northern Georgia]:EP by:RP [Tuesday afternoon]:EP, Washington:EP and [New York]:EP by:RP [Wednesday afternoon]:EP, meteorologists:EP said:RP.

`EP: entity mention candidate; RP: relation phrase.`

Figure 3: Example output of candidate generation.

mentions $\mathcal{M}_L \subset \mathcal{M}$ can be confidently mapped to entities in Ψ. The type of a linked candidate $m \in \mathcal{M}_L$ can be obtained based on its mapping entity $\kappa_e(m)$ (see Sec. 4.1). This work focuses on predicting the types of *unlinkable candidate mentions* $\mathcal{M}_U = \mathcal{M} \backslash \mathcal{M}_L$, where $\mathcal{M}_U$ may consist of (1) mentions of the emerging entities which are not in Ψ; (2) new names of the existing entities in Ψ; and (3) invalid entity mentions. Formally, we define the problem of **distantly-supervised entity recognition** as follows

DEFINITION 1 (PROBLEM DEFINITION). *Given a document collection $\mathcal{D}$, a target type set $\mathcal{T}$ and a knowledge base Ψ, our task aims to: (1) extract candidate entity mentions $\mathcal{M}$ from $\mathcal{D}$; (2) generate seed mentions $\mathcal{M}_L$ with Ψ; and (3) for each unlinkable candidate mention $m \in \mathcal{M}_U$, estimate its type indicator vector $\mathbf{y}_m$ to predict its type.*

In our study, we assume each mention within a sentence is only associated with a single type $t \in \mathcal{T}$. We also assume the target type set $\mathcal{T}$ is given (It is outside the scope of this study to generate $\mathcal{T}$). Finally, while our work is independent of entity linking techniques [21], our ER framework output may be useful to entity linking.

Framework Overview. Our overall framework is as follows:

1. Perform phrase mining on a POS-tagged corpus to extract candidate entity mentions and relation phrases, and construct a heterogeneous graph G to represent available information in a unified form, which encodes our insights on modeling the type for each entity mention (Sec. 3).
2. Collect seed entity mentions $\mathcal{M}_L$ as labels by linking extracted candidate mentions $\mathcal{M}$ to the KB Ψ (Sec. 4.1).
3. Estimate type indicator $\mathbf{y}$ for unlinkable candidate mention $m \in \mathcal{M}_U$ with the proposed type propagation integrated with relation phrase clustering on G (Sec. 4).

3. CONSTRUCTION OF GRAPHS

We first introduce candidate generation in Sec. 3.1, which leads to three kinds of objects, namely candidate entity mentions $\mathcal{M}$, their surface names $\mathcal{C}$ and surrounding relation phrases $\mathcal{P}$. We then build a heterogeneous graph G, which consists of multiple types of objects and multiple types of links, to model their relationship. The basic idea for constructing the graph is that: the more two objects are likely to share the same label (*i.e.*, $t \in \mathcal{T}$ or NOI), the larger the weight will be associated with their connecting edge.

Specifically, the constructed graph G unifies three types of links: *mention-name link* which represents the mapping between entity mentions and their surface names, *entity name-relation phrase link* which captures corpus-level co-occurrences between entity surface names and relation phrase, and *mention-mention link* which models distributional similarity between entity mentions. This leads to three subgraphs $G_{\mathcal{M},\mathcal{C}}$, $G_{\mathcal{C},\mathcal{P}}$ and $G_{\mathcal{M}}$, respectively. We introduce the construction of them in Secs. 3.2–3.4.

3.1 Candidate Generation

To ensure the extraction of informative and coherent entity mentions and relation phrases, we introduce a scalable, data-driven phrase mining method by incorporating both corpus-level statistics and syntactic constraints. Our method adopts a *global significance score* to guide the filtering of low-quality phrases and relies on a set of generic POS patterns to remove phrases with improper syntactic structure [4]. By extending the methodology used in [3], we can partition sentences in the corpus into non-overlapping segments which meet a significance threshold and satisfy our syntactic constraints. In doing so, entity candidates and relation phrases can be jointly extracted in an effective way.

Table 1: Performance on entity detection.

Method	NYT		Yelp		Tweet	
	Prec	Recall	Prec	Recall	Prec	Recall
Our method	**0.469**	**0.956**	**0.306**	**0.849**	0.226	**0.751**
NP chunker	0.220	0.609	0.296	0.247	**0.287**	0.181

First, we mine *frequent contiguous patterns* (*i.e.*, sequences of tokens with no gap) up to a fixed length and aggregate their counts. A greedy agglomerative merging is then performed to form longer phrases while enforcing our syntactic constraints. Suppose the size of corpus $\mathcal{D}$ is N and the frequency of a phrase S is denoted by $v(S)$. The phrase-merging step selects the most significant merging, by comparing the frequency of a potential merging of two consecutive phrases, $v(S_1 \oplus S_2)$, to the expected frequency assuming independence, $N\frac{v(S_1)}{N}\frac{v(S_2)}{N}$. Additionally, we conduct syntactic constraint check on every potential merging by applying an entity check function $I_e(\cdot)$ and a relation check function $I_p(\cdot)$. $I_e(S)$ returns one if S is *consecutive nouns* and zero otherwise; and $I_p(S)$ return one if S (partially) matches one of the patterns in Table 2. Similar to Student's t-test, we define a score function $\rho_X(\cdot)$ to measure the significance and syntactic correctness of a merging [3], where X can be e (entity mention) or p (relation phrase).

$$\rho_X(S_1, S_2) = \frac{v(S_1 \oplus S_2) - N\frac{v(S_1)}{N}\frac{v(S_2)}{N}}{\sqrt{v(S_1 \oplus S_2)}} \cdot I_X(S_1 \oplus S_2) \quad (1)$$

At each iteration, the greedy agglomerative algorithm performs the merging which has highest scores (ρ_e or ρ_p), and terminates when the next highest-score merging does not meet a pre-defined significance threshold. Relation phrases without matched POS patterns are discarded and their valid sub-phrases are recovered. Because the significance score can be considered analogous to hypothesis testing, one can use standard rule-of thumb values for the threshold (*e.g.*, Z-score$\geq$2) [3]. Overall the threshold setting is not sensitive in our empirical studies. As all merged phrases are frequent, we have fast access to their aggregate counts and thus it is efficient to compute the score of a potential merging.

Fig. 3 provides an example output of the candidate generation on New York Times (NYT) corpus. We further compare our method with a popular noun phrase chunker[1] in terms of entity detection performance, using the extracted entity mentions. Table 1 summarizes the comparison results on three datasets from different domains (see Sec. 5 for details). Recall is most critical for this step, since we can recognize false positives in later stages of our framework, but no chance to later detect the misses, *i.e.*, false negatives.

Table 2: POS tag patterns for relation phrases.

Pattern	Example
V	disperse; hit; struck; knock;
P	in; at; of; from; to;
V P	locate in; come from; talk to;
VW*(P)	caused major damage on; come lately

`V-verb; P-prep; W-{adv | adj | noun | det | pron}`
`W* denotes multiple W; (P) denotes optional.`

3.2 Mention-Name Subgraph

In practice, directly modeling the type indicator for each candidate mention may be infeasible due to the large num-

[1]TextBlob: `http://textblob.readthedocs.org/en/dev/`

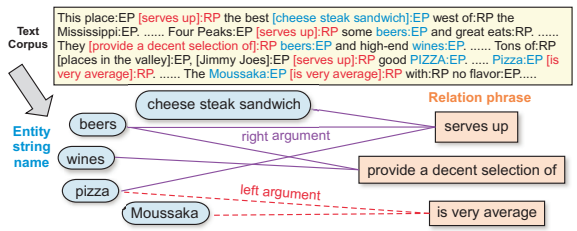


Figure 4: Example entity name-relation phrase links from Yelp reviews.

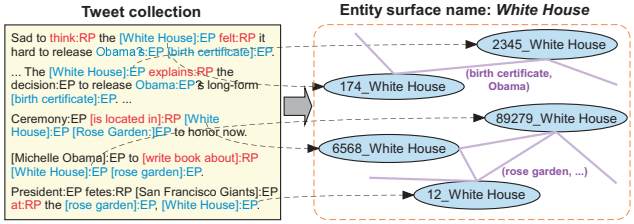


Figure 5: Example mention-mention links for entity surface name "*White House*" from Tweets.

ber of candidate mentions (*e.g.*, $|\mathcal{M}| > 1$ million in our experiments). This results in an intractable size of parameter space, *i.e.*, $\mathcal{O}(|M|T)$. Intuitively, both the entity name and the surrounding relation phrases provide strong cues on the type of a candidate entity mention. In Fig. 1, for example, the relation phrase "*beat*" suggests "*Golden Bears*" can mention a person or a sport team, while the surface name "*Golden Bears*" may refer to a sport team or a company. We propose to model the type indicator of a candidate mention based on the type indicator of its surface name and the type signatures of its associated relation phrases (see Sec. 4 for details). By doing so, we can reduce the size of the parameter space to $\mathcal{O}((|\mathcal{C}|+|\mathcal{P}|)T)$ where $|\mathcal{C}|+|\mathcal{P}| \ll |\mathcal{M}|$ (see Table 3 and Sec 5.1). This enables our method to scale up.

Suppose there are n unique surface names $\mathcal{C} = \{c_1, ..., c_n\}$ in all the extracted candidate mentions $\mathcal{M}$. This leads to a biadjacency matrix $\Pi_{\mathcal{C}} \in \{0,1\}^{M\times n}$ to represent the subgraph $G_{\mathcal{M},\mathcal{C}}$, where $\Pi_{\mathcal{C},ij} = 1$ if the surface name of m_j is c_j, and 0 otherwise. Each column of $\Pi_{\mathcal{C}}$ is normalized by its ℓ_2-norm to reduce the impact of popular entity names. We use a T-dimensional type indicator vector to measure how likely an entity name is subject to different types ($\mathcal{T}$ or NOI) and denote the type indicators for $\mathcal{C}$ by matrix $\mathbf{C} \in \mathbb{R}^{n\times T}$. Similarly, we denote the type indicators for $\mathcal{M}$ by $\mathbf{Y} \in \mathbb{R}^{M\times T}$.

3.3 Name-Relation Phrase Subgraph

By exploiting the aggregated co-occurrences between entity surface names and their surrounding relation phrases across multiple documents *collectively*, we weight the importance of different relation phrases for an entity name, and use their connected edge as bridges to propagate type information between different surface names by way of relation phrases. For each mention candidate, we assign it as the *left (right, resp.) argument* to the closest relation phrase appearing on its right (left, resp.) in a sentence. The *type signature* of a relation phrase refers to the two type indicators for its left and right arguments, respectively. The following hypothesis guides the type propagation between surface names and relation phrases.

HYPOTHESIS 1 (ENTITY-RELATION CO-OCCURRENCES). *If surface name c often appears as the left (right) argument of relation phrase p, then c's type indicator tends to be similar to the corresponding type indicator in p's type signature.*

In Fig. 4, for example, if we know "*pizza*" refers to food and find it frequently co-occurs with the relation phrase "*serves up*" in its right argument position, then another surface name that appears in the right argument position of "*serves up*" is likely food. This reinforces the type propagation that "*cheese steak sandwich*" is also food.

Formally, suppose there are l different relation phrases $\mathcal{P} = \{p_1, ..., p_l\}$ extracted from the corpus. We use two biadjacency matrices $\Pi_L, \Pi_R \in \{0,1\}^{M\times l}$ to represent the co-occurrences between relation phrases and their left and right entity arguments, respectively. We define $\Pi_{L,ij} = 1$ ($\Pi_{R,ij} = 1$) if m_i occurs as the *closest* entity mention on the left (right) of p_j *in a sentence*; and 0 otherwise. Each column of Π_L and Π_R is normalized by its ℓ_2-norm to reduce the impact of popular relation phrases. Two bipartite subgraphs $G_{\mathcal{C},\mathcal{P}}$ can be further constructed to capture the aggregated co-occurrences between relation phrases $\mathcal{P}$ and entity names $\mathcal{C}$ across the corpus. We use two biadjacency matrices $\mathbf{W}_L, \mathbf{W}_R \in \mathbb{R}^{n\times l}$ to represent the edge weights for the two types of links, and normalize them.

$$\mathbf{W}_L = \Pi_{\mathcal{C}}^T \Pi_L \quad \text{and} \quad \mathbf{W}_R = \Pi_{\mathcal{C}}^T \Pi_R;$$

$$\mathbf{S}_L = \mathbf{D}_L^{(\mathcal{C})^{-\frac{1}{2}}} \mathbf{W}_L \mathbf{D}_L^{(P)^{-\frac{1}{2}}} \text{and} \quad \mathbf{S}_R = \mathbf{D}_R^{(\mathcal{C})-\frac{1}{2}} \mathbf{W}_R \mathbf{D}_R^{(P)-\frac{1}{2}},$$

where $\mathbf{S}_L$ and $\mathbf{S}_R$ are normalized biadjacency matrices. For left-argument relationships, we define the diagonal surface name degree matrix $\mathbf{D}_L^{(\mathcal{C})} \in \mathbb{R}^{n\times n}$ as $D_{L,ii}^{(\mathcal{C})} = \sum_{j=1}^{l} W_{L,ij}$ and the relation phrase degree matrix $\mathbf{D}_L^{(\mathcal{P})} \in \mathbb{R}^{l\times l}$ as $D_{L,jj}^{(\mathcal{P})} = \sum_{i=1}^{n} W_{L,ij}$. Likewise, we define $\mathbf{D}_R^{(\mathcal{C})} \in \mathbb{R}^{n\times n}$ and $\mathbf{D}_R^{(\mathcal{P})} \in \mathbb{R}^{l\times l}$ based on $\mathbf{W}_R$ for the right-argument relationships.

3.4 Mention Correlation Subgraph

An entity mention candidate may have an ambiguous name as well as associate with ambiguous relation phrases. For example, "*White House*" mentioned in the first sentence in Fig. 5 can refer to either an organization or a facility, while its relation phrase "*felt*" can have either a person or an organization entity as the left argument. It is observed that other co-occurring entity mentions (*e.g.*, "*birth certificate*" and "*rose garden*" in Fig. 5) may provide good hints to the type of an entity mention candidate. We propose to propagate the type information between candidate mentions of *each* entity name based on the following hypothesis.

HYPOTHESIS 2 (MENTION CORRELATION). *If there exists a strong correlation (i.e., within sentence, common neighbor mentions) between two candidate mentions that share the same name, then their type indicators tend to be similar.*

Specifically, for each candidate entity mention $m_i \in \mathcal{M}$, we extract the set of entity surface names which co-occur with m_i *in the same sentence*. An n-dimensional TF-IDF vector $\mathbf{f}^{(i)} \in \mathbb{R}^n$ is used to represent the importance of these co-occurring names for m_i where $f_j^{(i)} = v_s(c_j) \cdot \log\left(|\mathcal{D}|/v_{\mathcal{D}}(c_j)\right)$ with term frequency in the sentence $v_s(c_j)$ and document frequency $v_{\mathcal{D}}(c_j)$ in $\mathcal{D}$. We use an affinity subgraph to represent the mention-mention link based on k-nearest neighbor (KNN) graph construction [8], denoted by an adjacency matrix $\mathbf{W}_{\mathcal{M}} \in \mathbb{R}^{M\times M}$. Each mention candidate is linked to its k most similar mention candidates which share the same name in terms of the vectors $\mathbf{f}$.

$$W_{\mathcal{M},ij} = \begin{cases} \text{sim}(\mathbf{f}^{(i)}, \mathbf{f}^{(j)}), & \text{if } \mathbf{f}^{(i)} \in N_k(\mathbf{f}^{(j)}) \text{ or } \mathbf{f}^{(j)} \in N_k(\mathbf{f}^{(i)}) \\ & \text{and } c(m_i) = c(m_j); \\ 0, & \text{otherwise.} \end{cases}$$

where we use the heat kernel function to measure similarity, *i.e.*, $\text{sim}(\mathbf{f}^{(i)}, \mathbf{f}^{(j)}) = \exp\left(-\|\mathbf{f}^{(i)} - \mathbf{f}^{(j)}\|^2/t\right)$ with $t = 5$ [8]. We use $N_k(\mathbf{f})$ to denote k nearest neighbors of $\mathbf{f}$ and $c(m)$ to denote the surface name of mention m. Similarly, we normalize $\mathbf{W}_{\mathcal{M}}$ into $\mathbf{S}_{\mathcal{M}} = \mathbf{D}_{\mathcal{M}}^{-\frac{1}{2}} \mathbf{W}_{\mathcal{M}} \mathbf{D}_{\mathcal{M}}^{-\frac{1}{2}}$ where the degree matrix $\mathbf{D}_{\mathcal{M}} \in \mathbb{R}^{M\times M}$ is defined by $D_{\mathcal{M},ii} = \sum_{j=1}^{M} W_{\mathcal{M},ij}$.

4. CLUSTERING-INTEGRATED TYPE PROPAGATION ON GRAPHS

This section introduces our unified framework for joint type propagation and relation phrase clustering on graphs.

A straightforward solution is to first perform hard clustering on the extracted relation phrases and then conduct type propagation between entity names and relation phrase clusters. Such a solution encounters several problems. One relation phrase may belong to multiple clusters, and the clusters so derived do not incorporate the type information of entity arguments. As such, the type prediction performance may not be best optimized by the mined clusters.

In our solution, we formulate a joint optimization problem to minimize both a graph-based semi-supervised learning error and a multi-view relation phrase clustering objective.

4.1 Seed Mention Generation

We first collect type information for the extracted mention candidates $\mathcal{M}$ by linking them to the KB. This yields a set of type-labeled mentions $\mathcal{M}_L$. Our goal is then to type the remaining unlinkable mention candidates $\mathcal{M}_U = \mathcal{M}/\mathcal{M}_L$.

We utilize a state-of-the-art entity name disambiguation tool[2] to map each candidate mention to Freebase entities. Only the mention candidates which are mapped with high confidence scores (*i.e.*, $\eta \geq 0.8$) are considered as valid output. We denote the mapping entity of a linked mention m as $\kappa_e(m)$, and the set of types of $\kappa_e(m)$ in Freebase as $\mathcal{T}(m)$. The linked mentions which associate with multiple target types (*i.e.*, $|\mathcal{T}(m) \cap \mathcal{T}| > 1$) are discarded to avoid type ambiguity. This finally leads to a set of labeled (seed) mentions $\mathcal{M}_L$. In our experiments, we found that only a very limited amount of extracted candidate entity mentions can be confidently mapped to Freebase entities (*i.e.*, $|\mathcal{M}_L|/|\mathcal{M}| < 7\%$). We define the type indicator $\mathbf{y}_m$ for a linked mention $m \in \mathcal{M}_L$ as $\mathbf{y}_{m,t} = 1$ if $\mathcal{T}(m) \cap \mathcal{T} = \{t\}$ and 0 otherwise, for $t \in \mathcal{T}$. Meanwhile, $\mathbf{y}_{m,\text{NOI}}$ is assigned with 1 if $\mathcal{T}(m) \cap \mathcal{T} = \emptyset$ and 0 otherwise.

4.2 Relation Phrase Clustering

In practice, we observe that many extracted relation phrases have very few occurrences in the corpus. This makes it hard to model their type signature based on the aggregated co-occurrences with entity names (*i.e.*, Hypothesis 1). In our experimental datasets, about 37% of the relation phrases have less than 3 unique entity surface names (in right or left arguments) in $G_{\mathcal{C},\mathcal{P}}$. Intuitively, by softly clustering synonymous relation phrases, the type signatures of frequent relation phrases can help infer the type signatures of infrequent (sparse) ones that have similar cluster memberships, based on the following hypothesis.

HYPOTHESIS 3 (TYPE SIGNATURE CONSISTENCY). *If two relation phrases have similar cluster memberships, the type indicators of their left and right arguments (type signature) tend to be similar, respectively.*

There has been some studies [6, 14] on clustering synonymous relation phrases based on different kinds of signals and clustering methods (see Sec. 6). We propose a general relation phrase clustering method to incorporate different features for clustering, which can be integrated with the graph-based type propagation in a mutually enhancing framework, based on the following hypothesis.

HYPOTHESIS 4 (RELATION PHRASE SIMILARITY). *Two relation phrases tend to have similar cluster memberships, if they have similar (1) strings; (2) context words; and (3) left and right argument type indicators.*

In particular, type signatures of relation phrases have proven very useful in clustering of relation phrases which have infrequent or ambiguous strings and contexts [6]. In contrast to previous approaches, our method leverages the type information derived by the type propagation and thus does not rely strictly on external sources to determine the type information for all the entity arguments.

Formally, suppose there are n_s (n_c) unique words $\{w_1, ..., w_{n_s}\}$ ($\{w'_1, ..., w'_{n_c}\}$) in all the relation phrase strings (contexts). We represent the strings and contexts of the extracted relation phrases $\mathcal{P}$ by two feature matrices $\mathbf{F}_s \in \mathbb{R}^{l \times n_s}$ and $\mathbf{F}_c \in \mathbb{R}^{l \times n_c}$, respectively. We set $F_{s,ij} = 1$ if p_i contains the word w_j and 0 otherwise. We use a text window of 10 words to extract the context for a relation phrase from each sentence it appears in, and construct context features $\mathbf{F}_c$ based on TF-IDF weighting. Let $\mathbf{P}_L, \mathbf{P}_R \in \mathbb{R}^{l \times T}$ denote the type signatures of $\mathcal{P}$. Our solution uses the derived features (*i.e.*, $\{\mathbf{F}_s, \mathbf{F}_c, \mathbf{P}_L, \mathbf{P}_R\}$) for multi-view clustering of relation phrases based on joint non-negative matrix factorization, which will be elaborated in the next section.

4.3 The Joint Optimization Problem

Our goal is to infer the label (type $t \in \mathcal{T}$ or NOI) for *each* unlinkable entity mention candidate $m \in \mathcal{M}_U$, *i.e.*, estimating $\mathbf{Y}$. We propose an optimization problem to unify two different tasks to achieve this goal: (i) type propagation over both the type indicators of entity names $\mathbf{C}$ and the type signatures of relation phrases $\{\mathbf{P}_L, \mathbf{P}_R\}$ on the heterogeneous graph G by way of graph-based semi-supervised learning, and (ii) multi-view relation phrase clustering. The seed mentions $\mathcal{M}_L$ are used as initial labels for the type propagation. We formulate the objective function as follows.

$$\mathcal{O}_{\alpha,\gamma,\mu} = \mathcal{F}(\mathbf{C}, \mathbf{P}_L, \mathbf{P}_R) + \mathcal{L}_\alpha\big(\mathbf{P}_L, \mathbf{P}_R, \{\mathbf{U}^{(v)}, \mathbf{V}^{(v)}\}, \mathbf{U}^*\big) + \Omega_{\gamma,\mu}(\mathbf{Y}, \mathbf{C}, \mathbf{P}_L, \mathbf{P}_R). \quad (2)$$

The first term $\mathcal{F}$ follows from Hypothesis 1 to model *type propagation* between entity names and relation phrases. By extending local and global consistency idea [8], it ensures that the type indicator of an entity name is similar to the type indicator of the left (or right) argument of a relation phrase, if their corresponding association is strong.

$$\mathcal{F}(\mathbf{C}, \mathbf{P}_L, \mathbf{P}_R) = \sum_{Z \in \{L,R\}} \sum_{i=1}^{n} \sum_{j=1}^{l} W_{Z,ij} \left\| \frac{\mathbf{C}_i}{\sqrt{D^{(\mathcal{C})}_{Z,ii}}} - \frac{\mathbf{P}_{Z,j}}{\sqrt{D^{(\mathcal{P})}_{Z,jj}}} \right\|_2^2, \quad (3)$$

The second term $\mathcal{L}_\alpha$ in Eq. (2) follows Hypotheses 3 and 4 to model the *multi-view relation phrase clustering* by joint non-negative matrix factorization. In this study, we consider each derived feature as one *view* in the clustering, *i.e.*, $\{\mathbf{F}^{(0)}, \mathbf{F}^{(1)}, \mathbf{F}^{(2)}, \mathbf{F}^{(3)}\} = \{\mathbf{P}_L, \mathbf{P}_R, \mathbf{F}_s, \mathbf{F}_c\}$ and derive a four-view clustering objective as follows.

$$\mathcal{L}_\alpha\big(\mathbf{P}_L, \mathbf{P}_R, \{\mathbf{U}^{(v)}, \mathbf{V}^{(v)}\}, \mathbf{U}^*\big) \quad (4)$$
$$= \sum_{v=0}^{d} \beta^{(v)} \big(\|\mathbf{F}^{(v)} - \mathbf{U}^{(v)} \mathbf{V}^{(v)T}\|_F^2 + \alpha \|\mathbf{U}^{(v)} \mathbf{Q}^{(v)} - \mathbf{U}^*\|_F^2 \big).$$

The first part of Eq. (4) performs matrix factorization on each feature matrix. Suppose there exists K relation phrase clusters. For each view v, we factorize the feature matrix $\mathbf{F}^{(v)}$ into a cluster membership matrix $\mathbf{U}^{(v)} \in \mathbb{R}^{l \times K}_{\geq 0}$ for all relation phrases $\mathcal{P}$ and a type indicator matrix $\mathbf{V}^{(v)} \in \mathbb{R}^{T \times K}_{\geq 0}$ for the K derived clusters. The second part of Eq. (4) enforces the consistency between the four derived cluster membership matrices through a *consensus matrix* $\mathbf{U}^* \in \mathbb{R}^{l \times K}_{\geq 0}$,

[2] `http://spotlight.dbpedia.org/`

which applies Hypothesis 4 to incorporate multiple similarity measures to cluster relation phrases. As in [13], we normalize $\{\mathbf{U}^{(v)}\}$ to the same scale (*i.e.*, $\|\mathbf{U}^{(v)}\mathbf{Q}^{(v)}\|_F \approx 1$) with the diagonal matrices $\{\mathbf{Q}^{(v)}\}$, where $Q^{(v)}_{kk} = \sum_{i=1}^{T} V^{(v)}_{ik}/\|\mathbf{F}^{(v)}\|_F$, so that they are comparable under the same consensus matrix. A tuning parameter $\alpha \in [0,1]$ is used to control the degree of consistency between the cluster membership of each view and the consensus matrix. $\{\beta^{(v)}\}$ are used to weight the information among different views, which will be automatically estimated. As the first part of Eq. (4) enforces $\{\mathbf{U}^{(0)}, \mathbf{U}^{(1)}\} \approx \mathbf{U}^*$ and the second part of Eq. (4) imposes $\mathbf{P}_L \approx \mathbf{U}^{(0)}\mathbf{V}^{(0)T}$ and $\mathbf{P}_R \approx \mathbf{U}^{(1)}\mathbf{V}^{(1)T}$, it can be checked that $U^*_i \approx U^*_j$ implies both $P_{L,i} \approx P_{L,j}$ and $P_{R,i} \approx P_{R,j}$ for any two relation phrases, which captures Hypothesis 3.

The last term $\Omega_{\gamma,\mu}$ in Eq. (2) models the type indicator for each entity mention candidate, the mention-mention link and the supervision from seed mentions.

$$\Omega_{\gamma,\mu}(\mathbf{Y},\mathbf{C},\mathbf{P}_L,\mathbf{P}_R) = \big\|\mathbf{Y} - f(\Pi_{\mathcal{C}}\mathbf{C}, \Pi_L\mathbf{P}_L, \Pi_R\mathbf{P}_R)\big\|_F^2 + \frac{\gamma}{2}\sum_{i,j=1}^{M} W_{\mathcal{M},ij}\left\|\frac{\mathbf{Y}_i}{\sqrt{D^{(\mathcal{M})}_{ii}}} - \frac{\mathbf{Y}_j}{\sqrt{D^{(\mathcal{M})}_{jj}}}\right\|_2^2 + \mu\|\mathbf{Y}-\mathbf{Y}_0\|_F^2. \quad (5)$$

In the first part of Eq. (5), the type of each entity mention candidate is modeled by a function $f(\cdot)$ based on the the type indicator of its surface name as well as the type signatures of its associated relation phrases. Different functions can be used to combine the information from surface names and relation phrases. In this study, we use an equal-weight linear combination, *i.e.*, $f(\mathbf{X}_1,\mathbf{X}_2,\mathbf{X}_3) = \mathbf{X}_1+\mathbf{X}_2+\mathbf{X}_3$. The second part follows Hypothesis 2 to model the mention-mention correlation by graph regularization, which ensures the consistency between the type indicators of two candidate mentions if they are highly correlated. The third part enforces the estimated $\mathbf{Y}$ to be similar to the initial labels from seed mentions, denoted by a matrix $\mathbf{Y}_0 \in \mathbb{R}^{\mathcal{M}\times T}$ (see Sec. 4.1). Two tuning parameters $\gamma, \mu \in [0,1]$ are used to control the degree of guidance from mention correlation in $G_{\mathcal{M}}$ and the degree of supervision from $\mathbf{Y}_0$, respectively.

To derive the exact type of each candidate entity mention, we impose the 0-1 integer constraint $\mathbf{Y} \in \{0,1\}^{M\times T}$ and $\mathbf{Y1} = \mathbf{1}$. To model clustering, we further require the cluster membership matrices $\{\mathbf{U}^{(v)}\}$, the type indicator matrices of the derived clusters $\{\mathbf{V}^{(v)}\}$ and the consensus matrix $\mathbf{U}^*$ to be non-negative. With the definition of $\mathcal{O}$, we define the joint optimization problem as follows.

$$\min_{\substack{\mathbf{Y},\mathbf{C},\mathbf{P}_L,\mathbf{P}_R,\mathbf{U}^*\\ \{\mathbf{U}^{(v)},\mathbf{V}^{(v)},\beta^{(v)}\}}} \mathcal{O}_{\alpha,\gamma,\mu} \quad (6)$$
$$\text{s.t.} \quad \mathbf{Y} \in \{0,1\}^{M\times T}, \mathbf{Y1} = \mathbf{1},\ \mathbf{U}^* \geq 0,\ \{\mathbf{U}^{(v)},\ \mathbf{V}^{(v)}\} \geq 0,\ \textstyle\sum_{v=0}^{d} e^{-\beta^{(v)}} = 1,$$

where $\sum_{v=0}^{d} e^{-\beta^{(v)}} = 1$ is used for avoiding trivial solution, *i.e.*, solution which completely favors a certain view.

4.4 The ClusType Algorithm

The optimization problem in Eq. (6) is mix-integer programming and thus is NP-hard. We propose a two-step approximate solution: first solve the real-valued relaxation of Eq. (6) which is a non-convex problem with $\mathbf{Y} \in \mathbb{R}^{M\times T}$; then impose back the constraints to predict the exact type of each candidate mention $m_i \in \mathcal{M}_U$ by type $(m_i) = \text{argmax}\ Y_i$.

Directly solving the real-valued relaxation of Eq. (6) is not easy because it is non-convex. We develop an alternating minimization algorithm to optimize the problem with respect to each variable alternatively, which accomplishes two tasks iteratively: type propagation on the heterogeneous graph, and multi-view clustering of relation phrases.

First, to learn the type indicators of candidate entity mentions, we take derivative on $\mathcal{O}$ with respect to $\mathbf{Y}$ while fixing other variables. As links only exist between entity mentions sharing the same surface name in $\mathbf{W}_{\mathcal{M}}$, we can efficiently estimate $\mathbf{Y}$ with respect to each entity name $c \in \mathcal{C}$. Let $\mathbf{Y}^{(c)}$ and $\mathbf{S}^{(c)}_{\mathcal{M}}$ denote the sub-matrices of $\mathbf{Y}$ and $\mathbf{S}_{\mathcal{M}}$, which correspond to the candidate entity mentions with the name c, respectively. We have the update rule for $\mathbf{Y}^{(c)}$ as follows:

$$\mathbf{Y}^{(c)} = \big[(1+\gamma+\mu)\mathbf{I}_c - \gamma\mathbf{S}^{(c)}_{\mathcal{M}}\big]^{-1}\big(\mathbf{\Theta}^{(c)} + \mu\mathbf{Y}^{(c)}_0\big), \quad \forall c \in \mathcal{C}, \quad (7)$$

where $\mathbf{\Theta} = \Pi_{\mathcal{C}}\mathbf{C} + \Pi_L\mathbf{P}_L + \Pi_R\mathbf{P}_R$. Similarly, we denote $\Theta^{(c)}$ and $\mathbf{Y}^{(c)}_0$ as sub-matrices of Θ and $\mathbf{Y}_0$ which correspond to the candidate mentions with name c, respectively. It can be shown that $[(1+\gamma+\mu)\mathbf{I}_c - \gamma\mathbf{S}^{(c)}_{\mathcal{M}}]$ is positive definite given $\mu > 0$ and thus is invertible. Eq. (7) can be efficiently computed since the average number of mentions of an entity name is small (*e.g.*, < 10 in our experiments). One can further parallelize this step to reduce the computational time.

Second, to learn the type indicators of entity names and the type signatures of relation phrases, we take derivative on $\mathcal{O}$ with respect to $\mathbf{C}$, $\mathbf{P}_L$ and $\mathbf{P}_R$ while fixing other variables, leading to the following closed-form update rules.

$$\mathbf{C} = \frac{1}{2}\big[\mathbf{S}_L\mathbf{P}_L + \mathbf{S}_R\mathbf{P}_R + \Pi_{\mathcal{C}}^T(\mathbf{Y} - \Pi_L\mathbf{P}_L - \Pi_R\mathbf{P}_R)\big]; \quad (8)$$
$$\mathbf{P}_L = \mathbf{X}_0^{-1}\big[\mathbf{S}_L^T\mathbf{C} + \Pi_L^T(\mathbf{Y} - \Pi_{\mathcal{C}}\mathbf{C} - \Pi_R\mathbf{P}_R) + \beta^{(0)}\mathbf{U}^{(0)}\mathbf{V}^{(0)T}\big];$$
$$\mathbf{P}_R = \mathbf{X}_1^{-1}\big[\mathbf{S}_R^T\mathbf{C} + \Pi_R^T(\mathbf{Y} - \Pi_{\mathcal{C}}\mathbf{C} - \Pi_L\mathbf{P}_L) + \beta^{(1)}\mathbf{U}^{(1)}\mathbf{V}^{(1)T}\big];$$

where we define $\mathbf{X}_0 = [(1+\beta^{(0)})\mathbf{I}_l + \Pi_L^T\Pi_L]$ and $\mathbf{X}_1 = [(1+\beta^{(1)})\mathbf{I}_l + \Pi_R^T\Pi_R]$ respectively. Note that the matrix inversions in Eq. (8) can be efficiently calculated with linear complexity since both $\Pi_L^T\Pi_L$ and $\Pi_R^T\Pi_R$ are diagonal matrices.

Finally, to perform multi-view clustering, we first optimize Eq. (2) with respect to $\{\mathbf{U}^{(v)}, \mathbf{V}^{(v)}\}$ while fixing other variables, and then update $\mathbf{U}^*$ and $\{\beta^{(v)}\}$ by fixing $\{\mathbf{U}^{(v)}, \mathbf{V}^{(v)}\}$ and other variables, which follows the procedure in [13].

We first take the derivative of $\mathcal{O}$ with respect to $\mathbf{V}^{(v)}$ and apply Karush-Kuhn-Tucker complementary condition to impose the non-negativity constraint on it, leading to the multiplicative update rules as follows:

$$V^{(v)}_{jk} = V^{(v)}_{jk}\frac{[\mathbf{F}^{(v)T}\mathbf{U}^{(v)}]_{jk} + \alpha\sum_{i=1}^{l} U^*_{ik}U^{(v)}_{ik}}{\mathbf{\Delta}^{(v)}_{jk} + \alpha\big(\sum_{i=1}^{l} U^{(v)2}_{ik}\big)\big(\sum_{i=1}^{T} V^{(v)}_{ik}\big)}, \quad (9)$$

where we define the matrix $\mathbf{\Delta}^{(v)} = \mathbf{V}^{(v)}\mathbf{U}^{(v)T}\mathbf{U}^{(v)} + \mathbf{F}^{(v)^-}\mathbf{U}^{(v)}$. It is easy to check that $\{\mathbf{V}^{(v)}\}$ remains non-negative after each update based on Eq. (9).

We then normalize the column vectors of $\mathbf{V}^{(v)}$ and $\mathbf{U}^{(v)}$ by $\mathbf{V}^{(v)} = \mathbf{V}^{(v)}\mathbf{Q}^{(v)-1}$ and $\mathbf{U}^{(v)} = \mathbf{U}^{(v)}\mathbf{Q}^{(v)}$. Following similar procedure for updating $\mathbf{V}^{(v)}$, the update rule for $\mathbf{U}^{(v)}$ can be derived as follows:

$$U^{(v)}_{ik} = U^{(v)}_{ik}\frac{[\mathbf{F}^{(v)^+}\mathbf{V}^{(v)} + \alpha\mathbf{U}^*]_{ik}}{[\mathbf{U}^{(v)}\mathbf{V}^{(v)T}\mathbf{V}^{(v)} + \mathbf{F}^{(v)^-}\mathbf{V}^{(v)} + \alpha\mathbf{U}^{(v)}]_{ik}}. \quad (10)$$

In particular, we make the decomposition $\mathbf{F}^{(v)} = \mathbf{F}^{(v)^+} - \mathbf{F}^{(v)^-}$, where $A^+_{ij} = (|A_{ij}| + A_{ij})/2$ and $A^-_{ij} = (|A_{ij}| - A_{ij})/2$, in order to preserve the non-negativity of $\{\mathbf{U}^{(v)}\}$.

The proposed algorithm optimizes $\{\mathbf{U}^{(v)}, \mathbf{V}^{(v)}\}$ for each view v, by iterating between Eqs. (9) and (10) until the following reconstruction error converges.

$$\delta^{(v)} = \|\mathbf{F}^{(v)} - \mathbf{U}^{(v)}\mathbf{V}^{(v)T}\|_F^2 + \alpha\|\mathbf{U}^{(v)}\mathbf{Q}^{(v)} - \mathbf{U}^*\|_F^2 \quad (11)$$

Algorithm 1 The **ClusType** algorithm

Input: biadjacency matrices $\{\Pi_{\mathcal{C}}, \Pi_L, \Pi_R, \mathbf{W}_L, \mathbf{W}_R, \mathbf{W}_{\mathcal{M}}\}$, clustering features $\{\mathbf{F}_s, \mathbf{F}_c\}$, seed labels $\mathbf{Y}_0$, number of clusters K, parameters $\{\alpha, \gamma, \mu\}$

Initialize $\{\mathbf{Y}, \mathbf{C}, \mathbf{P}_L, \mathbf{P}_R\}$ with $\{\mathbf{Y}_0, \Pi_{\mathcal{C}}^T\mathbf{Y}_0, \Pi_L^T\mathbf{Y}_0, \Pi_R^T\mathbf{Y}_0\}$, $\{\mathbf{U}^{(v)}, \mathbf{V}^{(v)}, \beta^{(v)}\}$ and $\mathbf{U}^*$ with positive values.
repeat
  Update candidate mention type indicator $\mathbf{Y}$ by Eq. (7)
  Update entity name type indicator $\mathbf{C}$ and relation phrase type signature $\{\mathbf{P}_L, \mathbf{P}_R\}$ by Eq. (8)
  for $v = 0$ to 3 **do**
    repeat
      Update $\mathbf{V}^{(v)}$ with Eq. (9)
      Normalize $\mathbf{U}^{(v)} = \mathbf{U}^{(v)}\mathbf{Q}^{(v)}$, $\mathbf{V}^{(v)} = \mathbf{V}^{(v)}\mathbf{Q}^{(v)-1}$
      Update $\mathbf{U}^{(v)}$ by Eq. (10)
    until Eq. (11) converges
  end for
  Update consensus matrix $\mathbf{U}^*$ and relative feature weights $\{\beta^{(v)}\}$ using Eq. (12)
until the objective $\mathcal{O}$ in Eq. (6) converges
Predict the type of $m_i \in \mathcal{M}_U$ by $\text{type}(m_i) = \text{argmax}\, Y_i$.

With optimized $\{\mathbf{U}^{(v)}, \mathbf{V}^{(v)}\}$, we update $\mathbf{U}^*$ and $\{\beta^{(v)}\}$ by taking the derivative on $\mathcal{O}$ with respect to each of them while fixing all other variables. This leads to the closed-form update rules as follows:

$$\mathbf{U}^* = \frac{\sum_{v=0}^{d} \beta^{(v)}\mathbf{U}^{(v)}\mathbf{Q}^{(v)}}{\sum_{v=0}^{d} \beta^{(v)}}; \quad \beta^{(v)} = -\log\Big(\frac{\delta^{(v)}}{\sum_{i=0}^{d} \delta^{(i)}}\Big). \quad (12)$$

Algorithm 1 summarizes our algorithm. For convergence analysis, ClusType applies block coordinate descent on the real-valued relaxation of Eq. (6). The proof procedure in [26] (not included for lack of space) can be adopted to prove convergence for ClusType (to the local minimum).

4.5 Computational Complexity Analysis

Given a corpus $\mathcal{D}$ with $N_{\mathcal{D}}$ words, the time complexity for our candidate generation and generation of $\{\Pi_{\mathcal{C}}, \Pi_L, \Pi_R, \mathbf{F}_s, \mathbf{F}_c\}$ is $\mathcal{O}(N_{\mathcal{D}})$. For construction of the heterogeneous graph G, the costs for computing $G_{\mathcal{C},\mathcal{P}}$ and $G_{\mathcal{M}}$ are $\mathcal{O}(nl)$ and $\mathcal{O}(MM_{\mathcal{C}}d_{\mathcal{C}})$, respectively, where $M_{\mathcal{C}}$ denotes average number of mentions each name has and $d_{\mathcal{C}}$ denotes average size of feature dimensions ($M_{\mathcal{C}} < 10, d_{\mathcal{C}} < 5000$ in our experiments). It takes $\mathcal{O}(MT)$ and $\mathcal{O}(MM_{\mathcal{C}}^2 + l^2)$ time to initialize all the variables and pre-compute the constants in update rules, respectively.

We then study the computational complexity of ClusType in Algorithm 1 with the pre-computed matrices. In each iteration of the outer loop, ClusType costs $\mathcal{O}(MM_{\mathcal{C}}T)$ to update $\mathbf{Y}$, $\mathcal{O}(nlT)$ to update $\mathbf{C}$ and $\mathcal{O}\big(nT(K+l)\big)$ to update $\{\mathbf{P}_L, \mathbf{P}_R\}$. The cost for inner loop is $\mathcal{O}\big(t_{in}lK(T + n_s + n_c)\big)$ supposing it stops after t_{in} iterations ($t_{in} < 100$ in our experiments). Update of $\mathbf{U}^*$ and $\{\beta^{(v)}\}$ takes $\mathcal{O}(lK)$ time. Overall, the computational complexity of ClusType is $\mathcal{O}\big(t_{out}nlT + t_{out}t_{in}lK(T + n_s + n_c)\big)$, supposing that the outer loop stops in t_{out} iterations ($t_{out} < 10$ in our experiments).

5. EXPERIMENTS

5.1 Data Preparation

Our experiments use three real-world datasets[3]: (1) **NYT:** constructed by crawling 2013 news articles from New York Times. The dataset contains 118,664 articles (57M tokens and 480k unique words) covering various topics such as Politics, Business and Sports; (2) **Yelp:** We collected 230,610 reviews (25M tokens and 418k unique words) from the 2014 *Yelp dataset challenge*; and (3) **Tweet:** We randomly selected 10,000 users in Twitter and crawled at most 100 tweets for each user in May 2011. This yields a collection of 302,875 tweets (4.2M tokens and 157k unique words).

[3]Code and datasets used in this paper can be downloaded at: `http://web.engr.illinois.edu/~xren7/clustype.zip`.

Table 3: Statistics of the heterogeneous graphs.

Data sets	NYT	Yelp	Tweet
#Entity mention candidates (M)	4.88M	1.32M	703k
#Entity surface names (n)	832k	195k	67k
#Relation phrases (l)	743k	271k	57k
#Links	29.32M	8.64M	3.59M
Avg#mentions per string name	5.86	6.78	10.56

Table 4: Target type sets $\mathcal{T}$ for the datasets.

NYT	*person*, *organization*, *location*, *time_event*
Yelp	*food*, *time_event*, *job_title*, *location*, *organization*
Tweet	*time_event*, *business_consumer_product*, *person*, *location*, *organization*, *business_job_title*, *time_year_of_day*

1. Heterogeneous Graphs. We first performed lemmatization on the tokens using NLTK WordNet Lemmatizer[4] to reduce variant forms of words (*e.g.*, eat, ate, eating) into their lemma form (*e.g.*, eat), and then applied Stanford POS tagger [25] on the corpus. In candidate generation (see Sec. 3.1), we set maximal pattern length as 5, minimum support as 30 and significance threshold as 2, to extract candidate entity mentions and relation phrases from the corpus. We then followed the introduction in Sec. 3 to construct the heterogeneous graph for each dataset. We used 5-nearest neighbor graphs when constructing the mention correlation subgraph. Table 3 summarizes the statistics of the constructed heterogeneous graphs for all three datasets.

2. Clustering Feature Generation. Following the procedure introduced in Sec. 4.2, we used a text window of 10 words to extract the context features for each relation phrase (5 words on the left and the right of a relation phrase), where stop-words are removed. We obtained 56k string terms (n_s) and 129k context terms (n_c) for the NYT dataset, 58k string terms and 37k context terms for the Yelp dataset and 18k string terms and 38k context terms for the Tweet dataset, respectively all unique term counts. Each row of the feature matrices were then normalized by its ℓ-2 norm.

3. Seed and Evaluation Sets. For evaluation purposes, we selected entity types which are popular in the dataset from Freebase, to construct the target type set $\mathcal{T}$. Table 4 shows the target types used in the three datasets. To generate the set of seed mentions $\mathcal{M}_L$, we followed the process introduced in Sec. 4.1 by setting the confidence score threshold as $\eta = 0.8$. To generate the evaluation sets, we randomly selected a subset of documents from each dataset and annotated them using the target type set $\mathcal{T}$ (each entity mention is tagged by one type). 1k documents are annotated for the NYT dataset (25,451 annotated mentions). 2.5k reviews are annotated for the Yelp dataset (21,252 annotated mentions). 3k tweets are annotated for the Tweet dataset (5,192 annotated mentions). We removed the mentions from the seed mention sets if they were in the evaluation sets.

5.2 Experimental Settings

In our testing of ClusType and its variants, we set the number of clusters $K = \{4000, 1500, 300\}$ for NYT, Yelp and Tweet datasets, respectively, based on the analyses in Sec. 5.3. We set $\{\alpha, \gamma, \mu\} = \{0.4, 0.7, 0.5\}$ by five-fold cross validation

[4] `http://www.nltk.org/`

Table 5: Performance comparisons on three datasets in terms of Precision, Recall and F1 score.

Data sets	NYT			Yelp			Tweet		
Method	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Pattern [7]	0.4576	0.2247	0.3014	0.3790	0.1354	0.1996	0.2107	0.2368	0.2230
FIGER [12]	0.8668	0.8964	0.8814	0.5010	0.1237	0.1983	**0.7354**	0.1951	0.3084
SemTagger [9]	0.8667	0.2658	0.4069	0.3769	0.2440	0.2963	0.4225	0.1632	0.2355
APOLLO [22]	0.9257	0.6972	0.7954	0.3534	0.2366	0.2834	0.1471	0.2635	0.1883
NNPLB [11]	0.7487	0.5538	0.6367	0.4248	0.6397	0.5106	0.3327	0.1951	0.2459
ClusType-NoClus	0.9130	0.8685	0.8902	0.7629	0.7581	0.7605	0.3466	0.4920	0.4067
ClusType-NoWm	0.9244	0.9015	0.9128	0.7812	0.7634	0.7722	0.3539	**0.5434**	0.4286
ClusType-TwoStep	0.9257	0.9033	0.9143	0.8025	0.7629	0.7821	0.3748	0.5230	0.4367
ClusType	**0.9550**	**0.9243**	**0.9394**	**0.8333**	**0.7849**	**0.8084**	0.3956	0.5230	**0.4505**

(of classification accuracy) on the seed mention sets. For convergence criterion, we stop the outer (inner) loop in Algorithm 1 if the relative change of $\mathcal{O}$ in Eq. (6) (reconstruction error in Eq. (11)) is smaller than 10^{-4}, respectively.

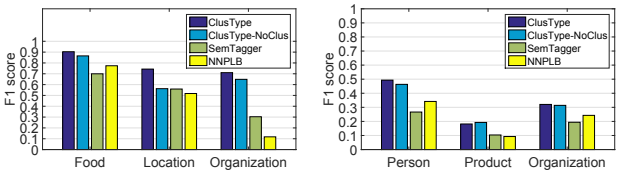
Compared Methods: We compared the proposed method (ClusType) with its variants which only model part of the proposed hypotheses. Several state-of-the-art entity recognition approaches were also implemented (or tested using their published codes): (1) **Stanford NER** [5]: a CRF classifier trained on classic corpora for several major entity types; (2) **Pattern** [7]: a state-of-the-art pattern-based bootstrapping method which uses the seed mention sets $\mathcal{M}_L$; (3) **SemTagger** [9]: a bootstrapping method which trains contextual classifiers using the seed mention set $\mathcal{M}_L$ in a self-training manner; (4) **FIGER** [12]: FIGER trains sequence labeling models using automatically annotated Wikipeida corpora; (5) **NNPLB** [11]: It uses ReVerb assertions [4] to construct graphs and performs entity name-level label propagation; and (6) **APOLLO** [22]: APOLLO constructs heterogeneous graphs on entity mentions, Wikipedia concepts and KB entities, and then performs label propagation.

All compared methods were first tuned on our seed mention sets using five-fold cross validation. For ClusType, besides the proposed full-fledged model, **ClusType**, we compare (1) **ClusType-NoWm**: This variant does not consider mention correlation subgraph, *i.e.*, set $\gamma = 0$ in ClusType; (2) **ClusType-NoClus**: It performs only type propagation on the heterogeneous graph, *i.e.*, Eq. (4) is removed from $\mathcal{O}$; and (3) **ClusType-TwoStep**: It first conducts multi-view clustering to assign each relation phrase to a single cluster, and then performs ClusType-NoClus between entity names, candidate entity mentions and relation phrase clusters.

Evaluation Metrics: We use F1 score computed from Precision and Recall to evaluate the entity recognition performance. We denote the #system-recognized entity mentions as J and the # ground truth annotated mentions in the evaluation set as A. Precision is calculated by $\text{Prec} = \sum_{m \in J \cap A} \omega(t'_m = t_m)/|J|$ and Recall is calculated by $\text{Rec} = \sum_{m \in J \cap A} \omega(t'_m = t_m)/|A|$. Here, t_m and t'_m denote the true type and the predicted type for m, respectively. Function $\omega(\cdot)$ returns 1 if the predicted type is correct and 0 otherwise. Only mentions which have correct boundaries and predicted types are considered correct. For cross validation on the seed mention sets, we use classification accuracy to evaluate the performance.

5.3 Experiments and Performance Study

1. Comparing ClusType with the other methods on entity recognition. Table 5 summarizes the comparison results on the three datasets. Overall, ClusType and its three variants outperform others on all metrics on NYT and Yelp and achieve superior Recall and F1 scores on Tweet. In particular, ClusType obtains a 46.08% improvement in F1 score and 168% improvement in Recall compared to the best baseline FIGER on the Tweet dataset and improves F1 by 48.94% compared to the best baseline, NNPLB, on the Yelp dataset.



(a) Yelp (b) Tweet

Figure 6: Performance breakdown by types.

FIGER utilizes a rich set of linguistic features to train sequence labeling models but suffers from low recall moving from a general domain (*e.g.*, NYT) to a noisy new domain (*e.g.*, Tweet) where feature generation is not guaranteed to work well (*e.g.*, 65% drop in F1 score). Superior performance of ClusType demonstrates the effectiveness of our candidate generation and of the proposed hypotheses on type propagation over domain-specific corpora. NNPLB also utilizes textual relation for type propagation, but it does not consider entity surface name ambiguity. APOLLO propagates type information between entity mentions but encounters severe context sparsity issue when using Wikipedia concepts. ClusType obtains superior performance because it not only uses semantic-rich relation phrases as type cues for each entity mention, but also clusters the synonymous relation phrases to tackle the context sparsity issues.

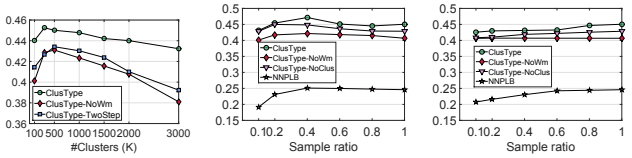
2. Comparing ClusType with its variants. Comparing with ClusType-NoClus and ClusType-TwoStep, ClusType gains performance from integrating relation phrase clustering with type propagation in a mutually enhancing way. It always outperforms ClusType-NoWm on Precision and F1 on all three datasets. The enhancement mainly comes from modeling the mention correlation links, which helps disambiguate entity mentions sharing the same surface names.

3. Comparing on different entity types. Fig. 6 shows the performance on different types on Yelp and Tweet. ClusType outperforms all the others on each type. It obtains larger gain on *organization* and *person*, which have more entities with ambiguous surface names. This indicates that modeling types on entity mention level is critical for name disambiguation. Superior performance on *product* and *food* mainly comes from the domain independence of our method because both NNPLB and SemTagger require sophisticated linguistic feature generation which is hard to adapt to new types.

4. Comparing with trained NER. Table 6 compares ours with a traditional NER method, *Stanford NER*, trained using classic corpora like ACE corpus, on three major types—person, location and organization. ClusType and its variants outperform Stanford NER on the corpora which are dynamic (*e.g.*, NYT) or domain-specific (*e.g.*, Yelp). On the Tweet dataset, ClusType has lower Precision but achieves a 63.59% improvement in Recall and 7.62% improvement in F1 score. The superior Recall of ClusType mainly comes from the domain-independent candidate generation.

Table 6: F1 score comparison with trained NER.

Method	NYT	Yelp	Tweet
Stanford NER [5]	0.6819	0.2403	0.4383
ClusType-NoClus	0.9031	0.4522	0.4167
ClusType	**0.9419**	**0.5943**	**0.4717**



(a) #Clusters (b) Seed set size (c) Corpus size

Figure 7: Performance changes in F1 score with #clusters, #seeds and corpus size on Tweets.

5. Testing on sensitivity over the number of relation phrase clusters, K. Fig. 7(a), ClusType was less sensitive to K compared with its variants. We found on the Tweet dataset, ClusType achieved the best performance when K=300 while its variants peaked at K=500, which indicates that better performance can be achieved with fewer clusters if type propagation is integrated with clustering in a mutually enhancing way. On the NYT and the Yelp datasets (not shown here), ClusType peaked at K=4000 and K=1500, respectively.

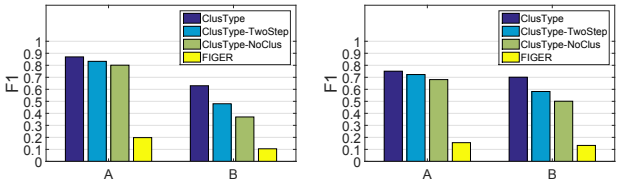
6. Testing on the size of seed mention set. Seed mentions are used as labels (distant supervision) for typing other mentions. By randomly selecting a subset of seed mentions as labeled data (sampling ratio from 0.1 to 1.0), Fig. 7(b) shows ClusType and its variants are not very sensitive to the size of seed mention set. Interestingly, using all the seed mentions does not lead to the best performance, likely caused by the type ambiguity among the mentions.

7. Testing on the effect of corpus size. Experimenting on the same parameters for candidate generation and graph construction, Fig. 7(c) shows the performance trend when varying the sampling ratio (subset of documents randomly sampled to form the input corpus). ClusType and its variants are not very sensitive to the changes of corpus size, but NNPLB had over 17% drop in F1 score when sampling ratio changed from 1.0 to 0.1 (while ClusType had only 5.5%). In particular, they always outperform FIGER, which uses a trained classifier and thus does not depend on corpus size.

5.4 Case Studies

1. Example output on two Yelp reviews. Table 7 shows the output of ClusType, SemTagger and NNPLB on two Yelp reviews: ClusType extracts more entity mention candidates (*e.g.*, "*BBQ*", "*ihop*") and predicts their types with better accuracy (*e.g.*, "*baked beans*", "*pulled pork sandwich*").

2. Testing on context sparsity. The type indicator of each entity mention candidate is modeled in ClusType based on the type indicator of its surface name and the type signatures of its co-occurring relation phrases. To test the handling of different relation phrase sparsity, two groups of 500 mentions are selected from Yelp: mentions in *Group A* co-occur with frequent relation phrases (∼4.6k occurrences in the corpus) and those in *Group B* co-occur with sparse relation phrases (∼3.4 occurrences in the corpus). Fig. 8(a) compares their F1 scores on the Tweet dataset. In general, all methods obtained better performance when mentions co-occurring with frequent relation phrases than with sparse relation phrases. In particular, we found that ClusType and its variants had comparable performance in Group A but ClusType obtained superior performance in Group B. Also,



(a) Context sparsity (b) Surface name popularity

Figure 8: Case studies on context sparsity and surface name popularity on the Tweet dataset.

ClusType-TwoStep obtained larger performance gain over ClusType-NoClus in Group B. This indicates that clustering relation phrases is critical for performance enhancement when dealing with sparse relation phrases, as expected.

3. Testing on surface name popularity. We generated the mentions in Group A with high frequency surface names (∼2.7k occurrences) and those in *Group B* with infrequent surface names (∼1.5). Fig. 8(b) shows the degraded performance of all methods in both cases—likely due to ambiguity in popular mentions and sparsity in infrequent mentions. ClusType outperforms its variants in Group B, showing it handles well mentions with insufficient corpus statistics.

4. Example relation phrase clusters. Table 8 shows relation phrases along with their corpus frequency from three example relation phrase clusters for the NYT dataset ($K = 4000$). We found that not only synonymous relation phrases, but also both sparse and frequent relation phrases can be clustered together effectively (*e.g.*, "*want hire by*" and "*recruited by*"). This shows that ClusType can boost sparse relation phrases with type information from the frequent relation phrases with similar group memberships.

Table 8: Example relation phrase clusters and their corpus frequency from the NYT dataset.

ID	Relation phrase
1	recruited by (5.1k); employed by (3.4k); want hire by (264)
2	go against (2.4k); struggling so much against (54); run for re-election against (112); campaigned against (1.3k)
3	looking at ways around (105); pitched around (1.9k); echo around (844); present at (5.5k);

6. RELATED WORK

1. Entity Recognition: Existing work leverages various levels of human supervision to recognize entities, from fully annotated documents (supervised), seed entities (weakly supervised), to knowledge bases (distantly supervised).

Traditional supervised methods [18, 15] use fully annotated documents and different linguistic features to train sequence labeling model. To obtain an effective model, the amount of labeled data is significant [18], despite of semi-supervised sequence labeling [19].

Weakly-supervised methods utilize a small set of typed entities as seeds and extract more entities of target types, which can largely reduce the amount of required labeled data. Pattern-based bootstrapping [7, 23] derives patterns from contexts of seed entities and uses them to incrementally extract new entities and new patterns unrestricted by specific domains, but often suffers low recall and semantic drift [12]. Iterative bootstrapping, such as probabilistic method [17] and label propagation [24] softly assign multiple types to an entity name and iteratively update its type distribution, yet cannot decide the exact type for each entity mention based on its local context.

Distantly supervised methods [16, 11, 12] avoid expensive human labels by leveraging type information of entity

Table 7: Example output of ClusType and the compared methods on the Yelp dataset.

ClusType	SemTagger	NNPLB
The best **BBQ:Food** I've tasted in **Phoenix:LOC** ! I had the **[pulled pork sandwich]:Food** with **coleslaw:Food** and **[baked beans]:Food** for lunch. ...	The best BBQ I've tasted in **Phoenix:LOC** ! I had the pulled **[pork sandwich]:LOC** with **coleslaw:Food** and **[baked beans]:LOC** for lunch. ...	The best **BBQ:Loc** I've tasted in **Phoenix:LOC** ! I had the pulled pork **sandwich:Food** with coleslaw and baked **beans:Food** for **lunch:Food**. ...
I only go to **ihop:LOC** for **pancakes:Food** because I don't really like anything else on the menu. Ordered **[chocolate chip pancakes]:Food** and a **[hot chocolate]:Food**.	I only go to ihop for pancakes because I don't really like anything else on the menu. Ordered **[chocolate chip pancakes]:LOC** and a **[hot chocolate]:LOC**.	I only go to ihop for pancakes because I don't really like anything else on the menu. Ordered chocolate chip pancakes and a hot chocolate.

mentions which are confidently mapped to entries in KBs. Linked mentions are used to type those unlinkable ones in different ways, including training a contextual classifier [16], learning a sequence labeling model [12] and serving as labels in graph-based semi-supervised learning [11].

Our work is also related to knowledge base population methods [22] which study entity linking and fine-grained categorization of unlinkable mentions in a unified framework, which shares the similar idea of modeling each entity mention individually to resolve name ambiguity.

2. Open Relation Mining: Extracting textual relation between subjective and objective from text has been extensively studied [4] and applied to entity typing [11]. Fader *et al.* [4] utilize POS patterns to extract verb phrases between detected noun phrases to form relation assertion. Schmitz *et al.* [20] further extend the textual relation by leveraging dependency tree patterns. These methods rely on linguistic parsers that may not generalize across domains. They also do not consider significance of the detected entity mentions in the corpus (see comparison with NNPLB [11]).

There have been some studies on clustering and and canonicalizing synonymous relations generated by open information extraction [6]. These methods either ignore entity type information when resolving relations, or assume types of relation arguments are already given.

7. CONCLUSIONS

We have studied distantly-supervised entity recognition and proposed a novel relation phrase-based entity recognition framework. A domain-agnostic phrase mining algorithm is developed for generating candidate entity mentions and relation phrases. By integrating relation phrase clustering with type propagation, the proposed method is effective in minimizing name ambiguity and context problems, and thus predicts each mention's type based on type distribution of its string name and type signatures of its surrounding relation phrases. We formulate a joint optimization problem to learn object type indicators/signatures and cluster memberships simultaneously. Our performance comparison and case studies show a significant improvement over state-of-the-art methods and demonstrate its effectiveness.

8. ACKNOWLEDGEMENTS

Research was sponsored in part by the U.S. Army Research Lab. under Cooperative Agreement No. W911NF-09-2-0053 (NSCTA), National Science Foundation IIS-1017362, IIS-1320617, and IIS-1354329, HDTRA1-10-1-0120, and grant 1U54GM114838 awarded by NIGMS through funds provided by the trans-NIH Big Data to Knowledge (BD2K) initiative (www.bd2k.nih.gov), and MIAS, a DHS-IDS Center for Multimodal Information Access and Synthesis at UIUC.

References

[1] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, 2008.

[2] X. L. Dong, T. Strohmann, S. Sun, and W. Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *SIGKDD*, 2014.

[3] A. El-Kishky, Y. Song, C. Wang, C. R. Voss, and J. Han. Scalable topical phrase mining from text corpora. *VLDB*, 2015.

[4] A. Fader, S. Soderland, and O. Etzioni. Identifying relations for open information extraction. In *EMNLP*, 2011.

[5] J. R. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *ACL*, 2005.

[6] L. Galárraga, G. Heitz, K. Murphy, and F. M. Suchanek. Canonicalizing open knowledge bases. In *CIKM*, 2014.

[7] S. Gupta and C. D. Manning. Improved pattern learning for bootstrapped entity extraction. In *CONLL*, 2014.

[8] X. He and P. Niyogi. Locality preserving projections. In *NIPS*, 2004.

[9] R. Huang and E. Riloff. Inducing domain-specific semantic class taggers from (almost) nothing. In *ACL*, 2010.

[10] Z. Kozareva and E. Hovy. Not all seeds are equal: Measuring the quality of text mining seeds. In *NAACL*, 2010.

[11] T. Lin, O. Etzioni, et al. No noun phrase left behind: detecting and typing unlinkable entities. In *EMNLP*, 2012.

[12] X. Ling and D. S. Weld. Fine-grained entity recognition. In *AAAI*, 2012.

[13] J. Liu, C. Wang, J. Gao, and J. Han. Multi-view clustering via joint nonnegative matrix factorization. In *SDM*, 2013.

[14] B. Min, S. Shi, R. Grishman, and C.-Y. Lin. Ensemble semantics for large-scale unsupervised relation extraction. In *EMNLP*, 2012.

[15] D. Nadeau and S. Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

[16] N. Nakashole, T. Tylenda, and G. Weikum. Fine-grained semantic typing of emerging entities. In *ACL*, 2013.

[17] K. Nigam and R. Ghani. Analyzing the effectiveness and applicability of co-training. In *CIKM*, 2000.

[18] L. Ratinov and D. Roth. Design challenges and misconceptions in named entity recognition. In *ACL*, 2009.

[19] S. Sarawagi and W. W. Cohen. Semi-markov conditional random fields for information extraction. In *NIPS*, 2004.

[20] M. Schmitz, R. Bart, S. Soderland, O. Etzioni, et al. Open language learning for information extraction. In *EMNLP*, 2012.

[21] W. Shen, J. Wang, and J. Han. Entity linking with a knowledge base: Issues, techniques, and solutions. *TKDE*, (99):1–20, 2014.

[22] W. Shen, J. Wang, P. Luo, and M. Wang. A graph-based approach for ontology population with named entities. In *CIKM*, 2012.

[23] S. Shi, H. Zhang, X. Yuan, and J.-R. Wen. Corpus-based semantic class mining: distributional vs. pattern-based approaches. In *COLING*, 2010.

[24] P. P. Talukdar and F. Pereira. Experiments in graph-based semi-supervised learning methods for class-instance acquisition. In *ACL*, 2010.

[25] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *HLT-NAACL*, 2003.

[26] P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *JOTA*, 109(3):475–494, 2001.