Generic Statistical Relational Entity Resolution in Knowledge Graphs[*]

Jay Pujara
University of California, Santa Cruz
Santa Cruz, CA 95064
jay@cs.umd.edu

Lise Getoor
University of California, Santa Cruz
Santa Cruz, CA 95064
getoor@soe.ucsc.edu

Abstract

Entity resolution, the problem of identifying the underlying entity of references found in data, has been researched for many decades in many communities. A common theme in this research has been the importance of incorporating relational features into the resolution process. Relational entity resolution is particularly important in knowledge graphs (KGs), which have a regular structure capturing entities and their interrelationships. We identify three major problems in KG entity resolution: (1) intra-KG reference ambiguity; (2) inter-KG reference ambiguity; and (3) ambiguity when extending KGs with new facts. We implement a framework that generalizes across these three settings and exploits this regular structure of KGs. Our framework has many advantages over custom solutions widely deployed in industry, including collective inference, scalability, and interpretability. We apply our framework to two real-world KG entity resolution problems, ambiguity in NELL and merging data from Freebase and MusicBrainz, demonstrating the importance of relational features.

Introduction

Entity resolution has been a longstanding challenge (Elmagarmid, Ipeirotis, and Verykios 2007) that has lead to significant research in many communities, including databases (Hernández and Stolfo 1995), statistics (Winkler 2006; Fellegi and Sunter 1969b), information retrieval (Dong, Halevy, and Madhavan 2005), and natural language processing (Culotta, Wick, and McCallum 2007). While entity resolution occurs in many settings, one setting that is particularly relevant to the current research landscape is entity resolution for knowledge graphs. In the past decade, a myriad of research projects in academia and industry have sought to automatically extract information from freely available text, images, video, and audio and assemble these extractions into entity-centric knowledge bases known as knowledge graphs (Weikum and Theobald 2010; Nickel et al. 2015).

In comparison to the general problem of entity resolution, knowledge graphs presents additional opportunities, complexities and challenges. We analyze two key facets of entity resolution problems arising from the structure of knowledge graphs: using knowledge graph features and supporting collective dependencies in co-reference judgments. We begin by identifying the problems confronting entity resolution in knowledge graphs and then develop a general model adaptable to many entity resolution tasks and scenarios.

The general problem of entity resolution is to take a set of references, such as proper names found in text or spoken language or bounding boxes found in images or video, and produce a mapping from these references to entities, which represent a single concept. This problem has two popular formulations: clustering or pairwise prediction. When entity resolution is formulated as a clustering problem, the set of references are clustered, and each cluster of references represents an entity. In contrast, when entity resolution is formulated as a pairwise matching problem, each pair of references are assessed for equality and a connected component of equal references represents an entity. In both formulations, a key problem is measuring the similarity of references, either to determine cluster coherence or to produce pairwise co-reference predictions.

The earliest entity resolution research focused on developing specialized similarity measures for strings and attributes (Winkler 2006). More recent work in entity resolution has focused on using relationships between references to generate *relational* features. These relational features introduce dependencies between co-reference decisions for different references, resulting in a collective model that can outperform conventional approaches (Rastogi, Dalvi, and Garofalakis 2011; Kalashnikov and Mehrotra 2005; Singla and Domingos 2006). For example, (Bhattacharya and Getoor 2007) introduce relational features and similarities, and using a collective relational clustering approach, demonstrate superior results to non-collective approaches. However, in many cases, these relational entity resolution models require cumbersome feature engineering and careful implementation that preserves scalability. One key requirement for knowledge graph entity resolution is the ability to translate knowledge graph features, such as attributes, types, and the many different relationships between entities, into features that can be used to determine the similarity of references.

A second key requirement for entity resolution in knowledge graphs is correctly handling collective dependencies in

[*] This manuscript was previously published as part of the doctoral dissertation, "Probabilistic Models for Scalable Knowledge Graph Construction"

entity resolution decisions. Entity resolution problems are inherently collective due to transitivity or functionality constraints of equality. More formally, when resolving a set of references, a transitivity constraint requires that if A and B are co-referent, and B and C are co-referent, then A and C must also be co-referent. A functionality constraint can exist in a setting where a bijective mapping between references in two sets, **S** and **T**, is desired, if A $\in$ **S** and B $\in$ **T** are co-referent, then, for all C $\in$ **T**, A and C cannot be co-referent. While transitivity and functionality are standard examples of collective entity resolution challenges, the knowledge graph setting often includes more sophisticated examples of collective reasoning. For example, if we have two knowledge graphs that include references with relations pertaining to genealogical information, we might have references such as: $\textsc{Rel}(E_1, O_1, \texttt{parent})$, $\textsc{Rel}(E_2, O_2, \texttt{parent})$, then determining that E_1 and E_2 are co-referent can provide useful information that O_1 and O_2 are potentially co-referent as well.

Problem Definition

Our discussion hints at the diversity of entity resolution problems in knowledge graphs. Different phases of knowledge graph construction may face unique entity resolution challenges. We enumerate three general cases where entity resolution is necessary in knowledge graphs. Entity resolution may be required to:

1. resolve ambiguity in a set of candidate extractions
2. incorporate new extractions into an existing knowledge graph
3. combine information from two or more knowledge graphs

We discuss each of these scenarios in detail in the following paragraphs.

Ambiguity In Candidate Extractions

Knowledge graphs are commonly constructed by incorporating the outputs of information extraction methods. These information extraction techniques are subject to many sources of ambiguity. Each technique may process the same information differently, yielding many references from the same source material. Furthermore, the extraction source material may be inherently ambiguous, using different references for the same entity within a document, such as partial names or titles. Another common problem is anaphora, such as when a pronoun is used to with an ambiguous referent. Finally, the extractions are drawn from a corpus of documents, and each document may have variations in the representation of references, such as alternate spellings, prefixes, suffixes, and abbreviations. In addition to the noise in entity references, noise also exists in attributes and relations ascribed to each reference. In this scenario, the goal is to cluster a set of noisy references with noisy attributes and relations into a coherent set of entities.

Adding New Extractions to a Knowledge Graph

A somewhat simpler problem is extending an existing knowledge graph using new extractions. In this setting, the goal is to map each reference to an existing entity in the knowledge graph, or introduce a new entity into the knowledge graph. One strategy for dealing with new entities that do not exist in the knowledge graph is skolemization, where each potential new entity is given a unique identifier. References can now be matched with existing entities or the new, skolemized entities in the knowledge graph, casting the problem into the well-studied task of surjective bipartite matching from references to entities.

Through this formulation, the added constraint that each reference must match a single entity can often simplify the entity resolution process. While the attributes and relationships of the extracted reference may be noisy, as motivated in the previous scenario, the attributes and relationships of entities in the knowledge graph are expected to be highly reliable. As a result, relational features and attribute similarity play a more significant role in determining whether a reference can be resolved to an existing entity in the knowledge graph, or due to conflicting information, the reference should be added as a new entity with different attributes and relations.

Combining Multiple Knowledge Graphs

The final knowledge graph entity resolution scenario adheres most closely to the traditional approaches to entity resolution, where the goal is to combine information from two or more databases. In this setting, the goal is to find a mapping between entities in knowledge graphs, and then combine the attributes and relations of these entities. This problem can be formulated as mapping each knowledge graph to a "canonical" knowledge graph or instead be cast as a pairwise matching task between each pair of knowledge graphs. The latter formulation can introduce additional complexity in the form of transitivity constraints for equality across all knowledge graphs. These constraints can add new features for entity resolution, but may also make the desired mapping more computationally demanding. A further complication in this setting is that the knowledge graphs may use different schemas and ontologies. This problem is not covered in this work, but the development of standard ontologies and the problems of ontology matching or schema mapping have been extensively researched.

While these three entity resolution settings each present unique challenges, our goal is to provide a unified model for entity resolution. The goal of this model is to adapt to the diverse circumstances present in knowledge graph construction tasks. In the next section, we outline the structural elements of this model, and then introduce a probabilistic model for entity resolution that incorporates these elements into an entity resolution system.

Approach

The crucial aspect that distinguishes knowledge graphs from standard entity resolution problems is the rich and regular structure of the knowledge graph, which provides relational features. Our goal is to leverage this structure to build an entity resolution model that is easy to understand and customize, while still capturing the rich information present in the knowledge graph. We consider two dimensions to the entity resolution model: feature granularity and collective inference. First, we organize the features in knowledge graphs based on the granularity of knowledge required. While the most basic features rely on string similarity or generic rules of functionality and transitivity, more complicated features involve new entities, attribute similarity, equivalence classes of relations, and domain-specific patterns. Each of these features can be classified as local (involving a single co-reference decision) or collective (imposing a dependency between two or more co-reference decisions). Table 1 summarizes the knowledge graph features used by the entity resolution methods, and the following subsections delve more deeply into each of these feature sets. For each type of feature, we provide examples of corresponding logical rules. These rules can be combined in a probabilistic modeling framework, such as probabilistic soft logic, to produce a collective probabilistic graphical model for entity resolution.

Local and Collective Knowledge Graph Features

As motivated earlier, there are two broad classes of features in knowledge graphs: local and collective. Local features are those that can be computed for a pair of entities (or references) independently of the co-reference decisions of other entities in the knowledge graph. Examples of local features include string similarity of names, image similarity of photographs, type agreement, and attribute agreement. One key characteristic for a local feature is that its value does not depend on the entity resolution decisions for other pairs of entities. This characteristic allows local features to be computed once for a pair of features and reused. Consequentially, relying on local features for entity resolution can decrease computational overhead and improve entity resolution performance.

In contrast to local features, collective features involve dependencies between co-reference decisions, making collective features are more difficult to compute, particularly as co-reference decisions are updated or refined. The transitivity and functionality constraints in the introduction are examples of common collective features that have been used in entity resolution. However, the structure of knowledge graphs allow many more collective features to be generated using relationships between entities. Knowledge graph features can be abstract, such as the overlap of object-arguments for a reference's relations, or very concrete, such as the link between parents and children in the earlier example.

Knowledge Graph Models at Different Granularity

In this section, we develop components for a knowledge graph entity resolution model. The components have been classified into four categories:

1. **basic** features that are common to all entity resolution scenarios
2. **new entity** features that helpful when adding new entities into a knowledge graph
3. **abstract KG** features that are universal across many knowledge graph structures
4. **domain-specific** features that are designed to resolve a particular class of entities

In the subsequent sections, we will introduce logical rules for each type of feature, distinguishing between local and collective rules. The goal of these rules is to determine a pairwise resolution between two entities, denoted by $\textsc{Same}(\mathrm{E}_1, \mathrm{E}_2)$ for entities E_1 and E_2. Note that the $\textsc{Same}$ predicate is distinct from the $\textsc{SameAs}$ predicate, which is used to capture ontological information, such as `owl:sameAs`.

Since knowledge graphs routinely contain millions of entities, assessing pairwise equality between all entities is infeasible. A common technique to avoid the polynomial explosion of entity matching is **blocking**, which uses a simple heuristic to produce potential entity matches. Using this smaller set of possible resolutions can substantially improve scalability. In the following rules, we will represent a blocked pair of entities with the predicate $\textsc{CandSame}$. Blocking can also be used to restrict matches based on the entity resolution scenario. For example, when incorporating new extractions into a knowledge graph, where the goal is to map references in a set of extractions to an existing knowledge graph, blocking can be used to scope entity resolution to only allow matches between extractions and the knowledge graph, disallowing matches within the extractions or within the knowledge graph.

Modeling Knowledge Graph Entity Resolution

Basic Features

Rules for Local Features

Basic features are those common to all entity resolution scenarios, such as similarity functions and prior probabilities. we introduce three rules corresponding to basic local features. Rule 1 and Rule 2 are priors for $\textsc{Same}$. Often, a negative prior (1) is useful to implement a default policy that entities are not co-referent unless supported by evidence. A positive prior can also be useful in some models to establish a baseline match confidence for two entities that have been blocked.

The final basic local rule uses a similarity function, $\textsc{Sim}$, to assess whether two entities are co-referent. In general, the similarity function can depend on the representation of the entities (e.g. images, sound files, or textual representations). A great deal of research in entity resolution has been devoted to designing effective similarity functions for entity resolution. Examples of popular similarity functions are Levenstein (Navarro 2001; Wagner and Fischer 1974), Jaro (Jaro 1995), Jaro-Winkler (Winkler 1999), Monge-Elkan (Elkan and Monge 1996), Fellegi-Sunter (Fellegi and Sunter 1969a), Needleman-Wunsch, and Smith-Waterman (Durbin et al. 1998). In Rule 3 the similarity function is not explicitly specified, but a popular similarity function or combination of functions (Bilenko and Mooney 2003) can be used to determine similarity.

$$\neg\textsc{Same}(E_1, E_2) \qquad (1)$$

Table 1: Knowledge graph features categorized based on collective dependencies and level of granularity

	local	collective
basic	similarity scores	transitive, functional, sparsity
new entity	new entity prior	new entity penalty (sparsity)
abstract KG	type matching, type penalty	relation matching/equivalence
domain-specific	restricted type matching	restricted relation matching

$$\begin{aligned} &\textsc{CandSame}(E_1, E_2) \\ &\qquad \Rightarrow \textsc{Same}(E_1, E_2) \end{aligned} \tag{2}$$

$$\begin{aligned} &\textsc{CandSame}(E_1, E_2) \wedge \textsc{Sim}(E_1, E_2) \\ &\qquad \Rightarrow \textsc{Same}(E_1, E_2) \end{aligned} \tag{3}$$

Rules for Collective Features

The collective basic features incorporate the fundamental properties of equality: symmetry (Rule 4) and transitivity (Rule 5). Symmetry enforces the constraint that the order of the arguments to Same do not matter. Transitivity, discussed in the introduction, ensures that the co-reference process generates tight clusters of entities by encouraging co-reference cliques. Finally, Rule 6 has an opposite effect, encouraging sparsity by promoting functionality for the Same predicate. Not all entity resolution scenarios require functionality for co-references, but those discussed in the sections on extending a knowledge and combining knowledge graphs can benefit from such constraints.

$$\begin{aligned} &\textsc{Same}(E_1, E_2) \\ &\qquad \Rightarrow \textsc{Same}(E_2, E_1) \end{aligned} \tag{4}$$

$$\begin{aligned} \textsc{CandSame}(A, B) \;&\wedge\; \textsc{CandSame}(B, C) \\ \wedge\; \textsc{CandSame}(A, C) \;&\wedge\; \textsc{Same}(A, B) \\ &\wedge\; \textsc{Same}(B, C) \\ &\Rightarrow\; \textsc{Same}(A, C) \end{aligned} \tag{5}$$

$$\begin{aligned} \textsc{CandSame}(A, B) \;&\wedge\; \textsc{CandSame}(A, C) \\ &\wedge\; \textsc{Same}(A, B) \\ &\Rightarrow\; \neg\textsc{Same}(A, C) \end{aligned} \tag{6}$$

New Entity Features

Rules for Local Features

In problem settings where entity resolution is matching with respect to an existing knowledge graph, such as extending a knowledge graph and merging multiple knowledge graphs, the appropriate entity may not exist in the target knowledge graph. In these settings, we generate a new entity placeholder for each source reference. This placeholder will have no inherent relations, types, or attributes and will have a default similarity value. we designate these entities using the NewEntity predicate. Rule 7 establishes a prior that any reference matches a new entity. In subsequent rules, the NewEntity will be used to scope the rule to existing entities, which avoids penalizing new entity matches based on relations, types and attributes which are missing.

$$\begin{aligned} &\textsc{CandSame}(E_1, E_2) \wedge \textsc{NewEntity}(E_1) \\ &\qquad \Rightarrow \textsc{Same}(E_1, E_2) \end{aligned} \tag{7}$$

Rules for Collective Features

While a prior can be helpful, the desired behavior in entity resolution systems is to add a new entity only when no other entity in the knowledge graph appears to match. Rule 8 prevents a new entity from matching when a previously existing entity is a strong match for a reference.

$$\begin{aligned} \textsc{Same}(E_1, E_2) \;&\wedge\; \textsc{CandSame}(E_1, E_3) \\ &\wedge\; \textsc{NewEntity}(E_3) \\ &\Rightarrow\; \neg\textsc{Same}(E_1, E_3) \end{aligned} \tag{8}$$

Abstract Knowledge Graph Features

Abstract knowledge graph features use the relational structure and attributes shared by all knowledge graphs. The key strength is that these features are broadly applicable to any knowledge graph entity resolution problem. In scenarios such as disambiguating references within a knowledge graph, abstract knowledge graph rules can be used to collectively infer relations and labels in the knowledge graph while simultaneously determining entity co-references. However, one drawback of abstract knowledge graph rules is that their broad applicability may limit their usefulness. Rules that are agnostic to the particular labels and relations in a knowledge graph may have difficulty prioritizing which labels and relations are useful for entity resolution. One potential solution to this issue when ample training data is available is to introduce rules and then learn rule weights for each label and relation separately.

Rules for Local Features

Knowledge graph entities have associated properties such as attributes, labels, and type information that provide the basis for local features. Rule 9 specifies that these properties agree for two entities. Since many potential candidate matches may share the same properties, the rule is mediated by the candidate similarity, supporting similar matches more strongly than dissimilar matches. While entities with agreeing properties are a signal of co-reference, properties that are missing or explicitly disagree can be strong signals against co-reference. Rule 10 requires that co-referent entities share properties, but provides an exception for new entities which lack properties. Note that a symmetric rule for the second entity is not shown. These rules are most useful in entity resolution settings where knowledge graph information is relatively complete, whereas noisy or incomplete extractions may hamper entity resolution. Rule 11 provides a stronger signal by incorporating the knowledge graph ontology, disallowing entities with mutually-exclusive properties from matching. Even when extractions are noisy and properties incomplete, this signal can provide strong evidence against a potential co-reference match.

$$\begin{aligned}\textsc{CandSame}(E_1,E_2) &\wedge \textsc{Sim}(E_1,E_2)\\ \wedge\ \textsc{Lbl}(E_1,L) &\wedge \textsc{Lbl}(E_2,L)\\ &\Rightarrow \textsc{Same}(E_1,E_2)\end{aligned}\tag{9}$$

$$\begin{aligned}\textsc{CandSame}(E_1,E_2) &\wedge \textsc{Lbl}(E_1,L)\\ \wedge\ \neg\textsc{Lbl}(E_2,L) &\wedge \neg\textsc{NewEntity}(E_2)\\ &\Rightarrow \neg\textsc{Same}(E_1,E_2)\end{aligned}\tag{10}$$

$$\begin{aligned}\textsc{CandSame}(E_1,E_2) &\wedge \textsc{Lbl}(E_1,L_1)\\ \wedge\ \textsc{Lbl}(E_2,L_2) &\wedge \textsc{Mut}(L_1,L_2)\\ &\Rightarrow \neg\textsc{Same}(E_1,E_2)\end{aligned}\tag{11}$$

Rules for Collective Features

The collective abstract knowledge graph entity resolution parallel the local rules, but operate over relations and involve pairs of co-referent entities. Rule 12 requires that two co-referent entities have the same relation with co-referent objects. The collective nature of the rule introduces a dependence between entities that participate in the same relation across knowledge graphs, supporting co-references between the subjects and objects of the relation. Rule 13 has the opposite effect, penalizing co-references for matches between existing entities that do not share the same relations. Echoing the previous remarks on knowledge graph rules, these rules have limited usefulness in noisy or incomplete knowledge graphs where many relations may be missing. However, Rule 14 uses the ontology to find a stronger signal in mutually-exclusive relations.

$$\begin{aligned}\textsc{CandSame}(E_1,E_2) &\wedge \textsc{CandSame}(O_1,O_2)\\ \wedge\ \textsc{Sim}(E_1,E_2) &\wedge \textsc{Same}(O_1,O_2)\\ \wedge\ \textsc{Rel}(E_1,O_1,R) &\wedge \textsc{Rel}(E_2,O_2,R)\\ &\Rightarrow \textsc{Same}(E_1,E_2)\end{aligned}\tag{12}$$

$$\begin{aligned}\textsc{CandSame}(E_1,E_2) &\wedge \textsc{CandSame}(O_1,O_2)\\ \wedge\ \textsc{Same}(O_1,O_2) &\wedge \neg\textsc{Rel}(E_1,O_1,R)\\ \wedge\ \neg\textsc{NewEntity}(E_1) &\wedge \neg\textsc{NewEntity}(O_1)\\ &\wedge \textsc{Rel}(E_2,O_2,R)\\ &\Rightarrow \neg\textsc{Same}(E_1,E_2)\end{aligned}\tag{13}$$

$$\begin{aligned}\textsc{CandSame}(E_1,E_2) &\wedge \textsc{CandSame}(O_1,O_2)\\ \wedge\ \textsc{Same}(O_1,O_2) &\wedge \textsc{Rel}(E_1,O_1,R)\\ \wedge\ \textsc{Rel}(E_2,O_2,S) &\wedge \textsc{RMut}(R,S)\\ &\Rightarrow \neg\textsc{Same}(E_1,E_2)\end{aligned}\tag{14}$$

Domain-Specific Knowledge Graph Features

While abstract knowledge graph features provide a generally-applicable tool for knowledge graph entity resolution, in many cases domain experts can rely on experience to assess the most important features for matching knowledge graphs. Since our model uses interpretable rules that are easy to generate, domain experts can easily add and remove rules to the model to capture the most relevant relationships. In this section, we provide some example rules for the task of matching knowledge graphs in the music domain. These rules are derived from rules used in an industry knowledge graph matching system, supporting the assertion that rules are a natural and common form of supplying domain expertise for knowledge graphs.

Rules for Local Features

One example of a domain rule that strongly supports co-reference are relations with categorical domains. The `release_type` relation in musical domains differentiates between singles, EPs, and albums. Since the domain of the relation is a small, enumerated set, matching release types across co-references is important. Rule 15 incorporates this domain knowledge in a rule for release type matching. Just as some relations are more important than others, so are types, attributes and labels. While general purpose ontologies have a `person` type, a more specific type can be far more useful in matching. Rule 16 provides a special case for `artist`, a subtype of `person`. One way of interpreting this rule is a type-specific prior for entity matches. By choosing appropriate weights, these rules can also moderate the importance of a similarity metric in a particular domain. For example, a high similarity value may not be meaningful for a broad domain (e.g. `person`) but can provide a stronger disambiguating signal for a more selective domain (e.g. `artist`).

$$\begin{aligned}\textsc{CandSame}(E_1,E_2) &\wedge \textsc{Sim}(E_1,E_2)\\ &\wedge \textsc{Rel}(E_1,L,\texttt{release\_type})\\ &\wedge \textsc{Rel}(E_2,L,\texttt{release\_type})\\ &\Rightarrow \textsc{Same}(E_1,E_2)\end{aligned}\tag{15}$$

$$\begin{aligned} \textsc{CandSame}(E_1, E_2) \;\wedge\;& \textsc{Sim}(E_1, E_2) \\ \wedge\;& \textsc{Lbl}(E_1, \texttt{artist}) \\ \wedge\;& \textsc{Lbl}(E_2, \texttt{artist}) \\ \Rightarrow\;& \textsc{Same}(E_1, E_2) \qquad (16) \end{aligned}$$

Rules for Collective Features

Similarly, domain experts can select the most important relations for resolution in a domain. Rule 17 which focuses on co-referent releases of the same album can be more useful than a rule which focuses on `release_label` since a label typically has many releases. Domain rules can also incorporate more complex criteria. Rule 18 has a similar form to normal collective relational rules, but includes an additional constraint that the albums and artists must all come from the same genre.

$$\begin{aligned} \textsc{CandSame}(E_1, E_2) \;\wedge\;& \textsc{Sim}(E_1, E_2) \\ \wedge\; \textsc{CandSame}(O_1, O_2) \;\wedge\;& \textsc{Same}(E_2, E_1) \\ \wedge\;& \textsc{Rel}(E_1, O_1, \texttt{release\_album}) \\ \wedge\;& \textsc{Rel}(E_2, O_2, \texttt{release\_album}) \\ \Rightarrow\;& \textsc{Same}(O_1, O_2) \qquad (17) \end{aligned}$$

$$\begin{aligned} \textsc{CandSame}(E_1, E_2) \;\wedge\;& \textsc{CandSame}(O_1, O_2) \\ \wedge\; \textsc{Sim}(E_1, E_2) \;\wedge\;& \textsc{Sim}(O_1, O_2) \\ \wedge\;& \textsc{Rel}(E_1, O_1, \texttt{album\_artist}) \\ \wedge\;& \textsc{Rel}(E_2, O_2, \texttt{album\_artist}) \\ \wedge\;& \textsc{Rel}(E_1, G, \texttt{album\_genre}) \\ \wedge\;& \textsc{Rel}(E_2, G, \texttt{album\_genre}) \\ \wedge\;& \textsc{Rel}(O_1, G, \texttt{artist\_genre}) \\ \wedge\;& \textsc{Rel}(O_2, G, \texttt{artist\_genre}) \\ \wedge\;& \textsc{Same}(O_1, O_2) \\ \Rightarrow\;& \textsc{Same}(E_1, E_2) \qquad (18) \end{aligned}$$

Synthesis

The previous section introduced a number of rules for entity resolution, categorized by whether the rule used local or collective information and the granularity of the knowledge graph features used. In the discussion of each rule, we referenced the three knowledge graph entity resolution scenarios and the conditions under which the rule was applicable to the scenario. The rules and this discussion is summarized in Table 2. Note that some of the entries have question marks, which reinforce the guidance that the corresponding rules may be appropriate based on dataset characteristics such as noise and sparsity.

The knowledge graph entity resolution model presented in this section is a general and versatile approach to entity resolution in richly structured domains. Since the requirements of different entity resolution scenarios vary, care must be taken to select the appropriate rules and design meaningful domain-specific rules. However the proliferation of domain-specific entity resolution methods (Durbin et al. 1998; Winkler 2006) and anecdotal evidence from many projects in industry suggest that many bespoke entity resolution systems are already in use. Despite the widespread use of such systems and substantial research in entity resolution, no general-purpose, collective framework has been adopted across domains.

This work provides a general guide to designing entity resolution systems for knowledge graphs. The rules presented can be used as templates for many approaches that jointly model entity resolution decisions, such as linear programs and probabilistic models. The modeling approach lends itself to many techniques for efficient inference, such as lifted inference. To demonstrate the value of the proposed entity resolution, we implement this model in a probabilistic programming framework. We use the rules as the basis for a probabilistic soft logic (PSL) program for performing entity resolution. PSL is a natural choice for entity resolution models, since entity resolution models have many collective dependencies, use real-valued similarity measures, and often include a vast number of entities.

Evaluation

We evaluate our knowledge graph entity resolution approach on two very different datasets from different entity resolution scenarios. The first dataset, corresponding to the scenario where references in ambiguous extractions are resolved, involves clustering unresolved references with associated relations and attributes from different web sources. The second dataset, corresponding to the scenario in merging multiple knowledge graphs, involves resolving entities between the MusicBrainz music knowledge graph and the Freebase broad-coverage knowledge graph.

NELL NELL extracts a series of facts from text, and uses a set of heuristics to map textual references to entities. This entity mapping process includes two phases: first, textual references are clustered to identify senses and then these textual references are mapped to the appropriate senses. The entity mapping process does not use the context of the knowledge graph, which can improve the performance on entity mapping. Furthermore, the entity mapping process does not attempt to perform entity resolution between different textual references that refer to the same underlying entity.

In order to investigate the effectiveness of entity resolution applied to ambiguous candidate extractions, we worked with the NELL team to collect data from a new NELL instance that performed only the first phase of entity mapping, clustering textual references to generate senses. The second phase of entity mapping was not performed, so this NELL instance produced raw candidate extractions in terms of the original textual references. The goal in this setting is to collectively determine the facts in the knowledge graph along with the entity co-references.

NELL's Entity Resolver produces match scores for pairs of textual references. We extend these match scores by computing a number of string similarity metrics for each pair of textual references, using the SecondString library (Cohen, Ravikumar, and Fienberg 2003). The set of string similarities includes the Jaccard overlap (of characters), Jaro, Jaro-

Table 2: Summarizing entity resolution rules and matching them to application

	Local/ collective	New extractions	Extend KG	Multiple KGs
Negative prior (1)	L	Y	Y	Y
Positive prior (2)	L	Y	Y	Y
Similarities (3)	L	Y	Y	Y
Symmetry (4)	C	Y	Y	Y
Transitivity (5)	C	Y	N	Y
Sparsity (6)	C	N	N	Y
New Entity prior (7)	L	N	Y	Y
New Entity penalty (8)	C	N	Y	Y
Label agreement (9)	L	N?	Y	Y
Label disagreement (10)	L	N?	Y?	Y?
Label exclusion (11)	L	Y	Y	Y
Relational agreement (12)	C	N?	Y	Y
Relational disagreement (13)	C	N?	Y?	Y?
Relational exclusion (14)	C	Y	Y	Y
Domain-specific categorical relations (15)	L	Y	Y	Y
Domain-specific prior (16)	L	Y	Y	Y
Domain-specific relations (17)	C	Y?	Y	Y
Domain-specific relational criteria (18)	C	Y?	Y	Y

Winkler, Levenshtein, Monge-Elkan, and Smith-Wasserman similarity functions. These string similarities constituted local features for entity resolution.

Using data from the first iteration of NELL yielded 145K candidate relations, 200K candidate labels, 170K unique textual references that mapped to 190K potential entities. The NELL EntityResolver candidate generation produced 4K potential entity co-references. Since the dataset was collected from a new NELL instance, no existing entity match information was used or available. Furthermore, since there were no pre-existing entities, each textual reference was considered unknown and no special handling of new entities was required.

NELL does not have a reliable source of entity resolution data, so we manually labeled entity co-references. For each method, we chose the top 50 as well as 50 randomly selected entity co-references from each method. This selection process yielded 421 co-references after duplicates were removed. We then removed the truth values and randomized the order of the co-references for judging.

Entities were judged to be co-referent when there was an unambiguous interpretation of the textual references that corresponded to one entity. This, for example, excludes “Giants” matching “San Jose Giants” since many other sports teams share the same name. Similarly, when a textual reference was the amalgamation of two entities, matches with either entity were disallowed. For example, this invalidates “Quito” from matching with “Quito and Cuenca”. However, merged entities were judged co-referent, allowing “Konica” and “Konica Minolta” to be co-referent since the company Konica merged with Minolta to become the merged company.

Results for the NELL entity resolution are reported in Table 3. Given the small amount of labeled data, weights were manually specified: the negative prior was given a weight of 0.6, similarity rules were given a weight of 1, attractive rules (transitivity, relationa agreement and label agreement) were given a weight of 25, and repulsive rules (relational disagreement and label disagreement) were given a weight of 40. The baseline, Basic-Local entity resolution uses only priors and the various similarity metrics. Using these extremely general features has high recall and fast inference, but low precision. Adding non-collective features from the knowledge graph (Basic & KG-Local) improves the precision substantially with a small loss of recall and slower inference (three minutes instead of four seconds). However, a bigger gain in precision comes from extending the Basic-Local rules by adding collective inference rules, such as transitivity. The Basic-All model has higher AUC, F1 and precision than either of the local models, with fast inference although this comes at the price of lowered recall. Finally, a model that incorporates all local and collective features, including basic similarity metrics and transitivity, and the knowledge graph features that include type and relationship matching and restriction, has the highest precision, F1 and AUC of the models evaluated, but also has the lowest recall. This model is also the slowest, since enumerating collective relational paths between entity pairs during grounding is expensive. Although the final precision of the most sophisticalyed model at 0.38 is relatively low, the improvement in precision is substantial when compared to the 0.21 precision of the naive model.

MusicBrainz and Freebase The second dataset for entity resolution involved mapping entities between two knowledge graphs. The first knowledge graph was from the MusicBrainz music knowledge base, available courtesy of the

Method	AUPRC	F1	Prec.	Recall	Inf. Time (s)
Basic, Local	0.267	0.333	0.214	0.759	5
Basic & KG, Local	0.247	0.426	0.298	0.747	220
Basic, All	0.307	0.446	0.333	0.675	8
Basic & KG, All	**0.351**	**0.453**	0.383	0.554	4000

Table 3: Comparing the performance of knowledge graph entity resolution rules in for the NELL dataset. Performance improves by adding knowledge graph features and collective features, with the best performance with both.

Method	AUPRC	F1	F1 (Exist)	F1 (New)
Basic & NewEntity, Local	0.416	0.734	0.169	0.744
Basic & Domain, All; NewEntity, Local	0.569	0.805	**0.305**	0.831
Basic & Domain & NewEntity, All	**0.724**	**0.840**	0.070	**0.895**

Table 4: Comparing the performance of knowledge graph entity resolution rules when merging MusicBrainz entities into the Knowledge Graph. Due to a skew toward new entities, the collective new entity rules dramatically improve overall performance, but with a substantial drop in the F1 measure for existing entities

LinkedBrainz project.[1] The second knowledge graph was the publicly available Freebase knowledge base.

An existing, proprietary pipeline to map entities between these two knowledge graphs exists. The pipeline uses Boolean rules restricted to discrete features. The system is designed to consider entity resolutions sequentially, and as a result cannot use all collective information between resolution decisions. When a match decision for an entity cannot be made by the pipeline, the entity is manually resolved by a human annotator. Evaluation of the existing pipeline showed a high error rate, while manually annotated entities contained no errors. Our experiments focus on the entities that were not successfully matched using the existing pipeline, which constitute the most difficult entity resolution decisions.

We begin with a dataset of 11K entities added to the MusicBrainz knowledge graph between 5/5/2014 and 6/29/14 that were manually annotated and have reliable ground truth. We identify 332K Freebase entities that are potential candidate matches for the MusicBrainz entities using a string similarity measure that is normalized for entity frequency. Since these newly added entities are often not found in Freebase, we generate new entity candidates for each MusicBrainz entity. The entity resolution dataset also includes 1M known entity mappings between the two knowledge graphs and 15.7M relations between entities. Weights for model rules were learned using known Freebase-Musicbrainz mappings.

Table 4 summarizes the results of these experiments. The baseline method uses only local rules, and achieves an area under the precision-recall curve (AUPRC) of 0.416 and an F1 measure of 0.734. Adding collective rules and domain-specific features that use the knowledge graph improves performance, with an AUPRC of 0.569 and an F1 of 0.805. Incorporating rules to handle new entities further improves performance with an AUPRC of 0.724 and an F1 measure of 0.840. Since these models were run on a best-effort, shared tenancy distributed computing platform, running times are difficult to reliably estimate and are not reported.

To better understand the performance, we separate the F1 measure for existing entities and new entities. In the dataset we collected, the entity mappings are skewed toward new entities, so that approximately 75% of entities in the MusicBrainz knowledge graph are not found in the Freebase entities. Thus the New Entity rules can have a dramatic influence on the performance by improving the performance on new entities while having a marked decrease in performance in existing entities.

Discussion

The growing importance of knowledge graphs has resulted in an increased emphasis on entity resolution for such structured domains. The collective relationships in a knowledge graph provide the key to improving the performance of entity resolution. In this paper, we provided an inventory of important features necessary for entity resolution in the context of knowledge graphs and identified the corresponding knowledge graph settings where these features are important. Building entity resolution models, particularly collective models, has required a great deal of time and effort. Our proposed framework provides a general solution that is applicable to many different problem settings. We argue that these generally structured models, in concert with the many statistical relational inference toolkits, provide a superior solution to the custom-engineered solutions that proliferate in practical applications. As an example, the PSL implementation of our entity resolution model provides an accessible platform allowing rapid prototyping and experimentation for a variety of entity resolution problems. This model allows easy integration of domain-specific rules from experts. Our entity resolution experiments mapping MusicBrainz entities to Freebase show that our model can handle the most difficult instances that existing systems fail to resolve.

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

Acknowledgements

This manuscript is an adaptation of a chapter of the dissertation “Probabilistic Models for Scalable Knowledge Graph Construction” (Pujara 2016). Data for disambiguating NELL extractions was collected with the assistance of Bryan Kisiel, Jayant Krishnamurthy, and William Cohen. Work on entity resolution for Freebase data was undertaken during an internship with the Google Knowledge Vault team, and with the help of Luna Dong, Curtis Janssen, and Kevin Murphy.

Portions of this work were supported by the National Science Foundation (NSF), under by grant IIS1218488. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of NSF or the U.S. Government.

References

[Bhattacharya and Getoor 2007] Bhattacharya, I., and Getoor, L. 2007. Collective Entity Resolution in Relational Data. *ACM Transactions on Knowledge Discovery and Datamining* 1(1).

[Bilenko and Mooney 2003] Bilenko, M., and Mooney, R. J. 2003. Adaptive duplicate detection using learnable string similarity measures. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 39–48. ACM.

[Cohen, Ravikumar, and Fienberg 2003] Cohen, W.; Ravikumar, P.; and Fienberg, S. 2003. A Comparison of String Matching Tasks for Names and Addresses. In *IJCAI Workshop on Information Integration on the Web*.

[Culotta, Wick, and McCallum 2007] Culotta, A.; Wick, M.; and McCallum, A. 2007. First-Order Probabilistic Models for Coreference Resolution. In *HLT-NAACL*.

[Dong, Halevy, and Madhavan 2005] Dong, X.; Halevy, A.; and Madhavan, J. 2005. Reference Reconciliation in Complex Information Spaces. In *SIGMOD*. ACM.

[Durbin et al. 1998] Durbin, R.; Eddy, S. R.; Krogh, A.; and Mitchison, G. 1998. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press.

[Elkan and Monge 1996] Elkan, C., and Monge, A. 1996. The field matching problem: Algorithms and applications. In *Proc. of the Second International Conference on Knowledge Discovery and Data Mining, AAAI Press*.

[Elmagarmid, Ipeirotis, and Verykios 2007] Elmagarmid, A. K.; Ipeirotis, P. G.; and Verykios, V. S. 2007. Duplicate Record Detection: A Survey. *Transactions on Knowledge and Data Engineering* 19(1).

[Fellegi and Sunter 1969a] Fellegi, I. P., and Sunter, A. B. 1969a. A theory for record linkage. *Journal of the American Statistical Association* 64(328):1183–1210.

[Fellegi and Sunter 1969b] Fellegi, I. P., and Sunter, A. B. 1969b. A theory for record linkage. *Journal of the American Statistical Association* 64(328).

[Hernández and Stolfo 1995] Hernández, M. A., and Stolfo, S. J. 1995. The Merge/Purge Problem for Large Databases. In *SIGMOD*. ACM.

[Jaro 1995] Jaro, M. A. 1995. Probabilistic linkage of large public health data files. *Statistics in medicine* 14(5-7):491–498.

[Kalashnikov and Mehrotra 2005] Kalashnikov, D. V., and Mehrotra, S. 2005. A Probabilistic Model for Entity Disambiguation Using Relationships. In *SDM*.

[Navarro 2001] Navarro, G. 2001. A Guided Tour to Approximate String Matching. *ACM Comput. Surv.* 33(1):31–88.

[Nickel et al. 2015] Nickel, M.; Murphy, K.; Tresp, V.; and Gabrilovich, E. 2015. A Review of Relational Machine Learning for Knowledge Graphs: From Multi-Relational Link Prediction to Automated Knowledge Graph Construction. *arXiv preprint arXiv:1503.00759*.

[Pujara 2016] Pujara, J. 2016. *Probabilistic Models for Scalable Knowledge Graph Construction*. Ph.D. Dissertation, University of Maryland, College Park.

[Rastogi, Dalvi, and Garofalakis 2011] Rastogi, V.; Dalvi, N.; and Garofalakis, M. 2011. Large-scale Collective Entity Matching. *VLDB* 4(4).

[Singla and Domingos 2006] Singla, P., and Domingos, P. 2006. Entity Resolution with Markov Logic. In *ICDM*.

[Wagner and Fischer 1974] Wagner, R. A., and Fischer, M. J. 1974. The String-to-String Correction Problem. *J. ACM* 21(1):168–173.

[Weikum and Theobald 2010] Weikum, G., and Theobald, M. 2010. From Information to Knowledge: Harvesting Entities and Relationships Grom Web Sources. In *Proceedings of the 29th ACM Symposium On Principles Of Database Systems*, 65–76. ACM.

[Winkler 1999] Winkler, W. E. 1999. The state of record linkage and current research problems. In *Statistical Research Division, US Census Bureau*. Citeseer.

[Winkler 2006] Winkler, W. E. 2006. Overview of record linkage and current research directions. Technical Report RRS Statistics 2006-2, Bureau of the Census.