Knowledge Graph and Corpus Driven Segmentation and Answer Inference for Telegraphic Entity-seeking Queries

Mandar Joshi *
IBM Research
mandarj90@in.ibm.com

Uma Sawant
IIT Bombay, Yahoo Labs
uma@cse.iitb.ac.in

Soumen Chakrabarti
IIT Bombay
soumen@cse.iitb.ac.in

Abstract

Much recent work focuses on formal interpretation of natural question utterances, with the goal of executing the resulting structured queries on knowledge graphs (KGs) such as Freebase. Here we address two limitations of this approach when applied to open-domain, entity-oriented Web queries. First, Web queries are rarely well-formed questions. They are "telegraphic", with missing verbs, prepositions, clauses, case and phrase clues. Second, the KG is always incomplete, unable to directly answer many queries. We propose a novel technique to segment a telegraphic query and assign a coarse-grained purpose to each segment: a base entity e_1, a relation type r, a target entity type t_2, and contextual words s. The query seeks entity $e_2 \in t_2$ where $r(e_1, e_2)$ holds, further evidenced by schema-agnostic words s. Query segmentation is integrated with the KG and an unstructured corpus where mentions of entities have been linked to the KG. We do not trust the best or any specific query segmentation. Instead, evidence in favor of candidate e_2s are aggregated across several segmentations. Extensive experiments on the ClueWeb corpus and parts of Freebase as our KG, using over a thousand telegraphic queries adapted from TREC, INEX, and WebQuestions, show the efficacy of our approach. For one benchmark, MAP improves from 0.2–0.29 (competitive baselines) to 0.42 (our system). NDCG@10 improves from 0.29–0.36 to 0.54.

*Work done as Masters student at IIT Bombay

1 Introduction

A majority of Web queries mention an entity or type (Lin et al., 2012), as users increasingly explore the Web of objects using Web search. To better support entity-oriented queries, commercial Web search engines are rapidly building up large catalogs of types, entities and relations, popularly called a "knowledge graph" (KG) (Gallagher, 2012). Despite these advances, robust, Web-scale, open-domain, entity-oriented search faces many challenges. Here, we focus on two.

1.1 "Telegraphic" queries

First, the surface utterances of entity-oriented Web queries are dramatically different from TREC- or Watson-style factoid question answering (QA), where questions are grammatically well-formed. Web queries are usually "telegraphic": they are short, rarely use function words, punctuations or clausal structure, and use relatively flexible word orders. E.g., the natural utterance "on the bank of which river is the Hermitage Museum located" may be translated to the telegraphic Web query hermitage museum river bank. Even on well-formed question utterances, 50% of interpretation failures are contributed by parsing or structural matching failures (Kwiatkowski et al., 2013). Telegraphic utterances will generally be even more challenging.

Consequently, whereas TREC-QA/NLP-style research has focused on parsing and precise interpretation of a well-formed query sentence to a strongly structured (typically graph-oriented) query language (Kasneci et al., 2008; Pound et al., 2012; Yahya et al., 2012; Berant et al., 2013; Kwiatkowski et al., 2013), the Web search and information retrieval (IR) community has focused on telegraphic queries (Guo et al., 2009; Sarkas et al., 2010; Li et al., 2011; Pantel et al., 2012; Lin et al., 2012; Sawant and Chakrabarti, 2013). In terms of target schema richness, these efforts may

Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1104–1114,
October 25-29, 2014, Doha, Qatar.

appear more modest. The act of query 'interpretation' is mainly a segmentation of query tokens by *purpose*. In the example above, one may report segments "Hermitage Museum" (a located artifact or named entity), and "river bank" (the target type). This is reminiscent of record segmentation in information extraction (IE). Over well-formed utterances, IE baselines are quite competitive (Yao and Van Durme, 2014). But here, we are interested exclusively in telegraphic queries.

1.2 Incomplete knowledge graph

The second problem is that the KG is always work in progress (Pereira, 2013), and connections found within nodes of the KG, between the KG and the query, or the KG and unstructured text, are often incomplete or erroneous. E.g., Wikipedia is considered tiny, and Freebase rather small, compared to what is needed to answer all but the "head" queries. Google's Freebase annotations (Gabrilovich et al., 2013) on ClueWeb (ClueWeb09, 2009) number fewer than 15 per page to ensure precision. Fewer than 2% are to entities in Freebase but not in Wikipedia.

It may also be difficult to harness the KG for answering certain queries. E.g., answering the query fastest odi century batsman, the intent of which is to find the batsman holding the record for the fastest century in One Day International (ODI) cricket, may be too difficult for most KG-only systems, but may be answered quite effectively by a system that also utilizes evidence from unstructured text.

There is a clear need for a "pay-as-you-go" architecture that involves both the corpus and KG. A query easily served by a curated KG should give accurate results, but it is desirable to have a *graceful interpolation* supported by the corpus: e.g., if the relation $r(e_1, e_2)$ is not directly evidenced in the KG, but strongly hinted in the corpus, we still want to use this for ranking.

1.3 Our contributions

Here, we make progress beyond the above frontier of prior work in the following significant ways. We present a new architecture for structural interpretation of a telegraphic query into these segments (some may be empty):

- Mention/s $\widehat{e_1}$ of an entity e_1,
- Mention $\widehat{r}$ of a relation type r,
- Mention $\widehat{t_2}$ of a target type t_2, and
- Other contextual matching words s (sometimes called *selectors*),

with the simultaneous intent of finding and ranking entities $e_2 \in t_2$, such that $r(e_1, e_2)$ is likely to hold, evidenced near the matching words in unstructured text.

Given the short, telegraphic query utterances, we limit our scope to at most one relation mention, unlike the complex mapping of clauses in well-formed questions to twig and join style queries (e.g., "find an actor whose spouse was an Italian bookwriter"). On the other hand, we need to deal with the unhelpful input, as well as consolidate the KG with the corpus for ranking candidate e_2s. Despite the modest specification, our query template is quite expressive, covering a wide range of entity-oriented queries (Yih et al., 2014).

We present a novel discriminative graphical model to capture the entity ranking inference task, with query segmentation as a by-product. Extensive experiments with over a thousand entity-seeking telegraphic queries using the ClueWeb09 corpus and a subset of Freebase show that we can accurately predict the segmentation and intent of telegraphic relational queries, and simultaneously rank candidate responses with high accuracy. We also present evidence that the KG and corpus have synergistic salutary effects on accuracy.

§2 explores related work in more detail. §3 gives some examples fitting our query template, explains why interpreting some of them is nontrivial, and sets up notation. §4 presents our core technical contributions. §5 presents experiments. Data can be accessed at `http://bit.ly/Spva49` and `http://bit.ly/WSpxvr`.

2 Related work

The NLP/QA community has traditionally assumed that question utterances are grammatically well-formed, from which precise clause structure, ground constants, variables, and connective relations can be inferred via semantic parsing (Kasneci et al., 2008; Pound et al., 2012; Yahya et al., 2012; Berant et al., 2013; Kwiatkowski et al., 2013) and translated to lambda expressions (Liang, 2013) or SPARQL style queries (Kasneci et al., 2008), with elaborate schema knowledge. Such approaches are often correlated with the assumption that all usable knowledge has been curated into a KG. The query is first translated to a structured form and then "executed" on the KG. A

Telegraphic query	$\widehat{e_1}$	$\widehat{r}$	$\widehat{t_2}$	s
first african american nobel prize winner	nobel prize	winner	african american	first
	nobel prize	-	winner	first african american
	-	-	winner	first african american nobel prize
dave navarro first band	dave navarro	band	-	first
	dave navarro	band	band	first
merril lynch headquarters	merril lynch	headquarters	-	-
	merril lynch	-	headquarters	-
spanish poet died in civil war	spanish	died in	poet	civil war
	civil war	died	-	spanish poet
	spanish	in	poet	died civil war
first american in space	-	-	-	first american in space
	-	-	american	first, in space

Figure 1: Example queries and some potential segmentations.

large corpus may be used to build relation expression models (Yao and Van Durme, 2014), but not as supporting evidence for target entities.

In contrast, the Web and IR community generally assumes a free-form query that is often telegraphic (Guo et al., 2009; Sarkas et al., 2010; Li et al., 2011). Queries being far more noisy, the goal of structure discovery is more modest, and often takes the form of a segmentation of the query regarded as a token sequence, assigning a broad *purpose* (Pantel et al., 2012; Lin et al., 2012) to each segment, mapping them probabilistically to a relatively loose schema, and ranking responses in conjunction with segmentations (Sawant and Chakrabarti, 2013). To maintain quality in the face of noisy input, these approaches often additionally exploit clicks (Li et al., 2011) or a corpus that has been annotated with entity mentions (Cheng and Chang, 2010; Li et al., 2010). The corpus provides contextual snippets for queries where the KG fails, preventing the systems from falling off the "structure cliff" (Pereira, 2013).

Our work advances the capabilities of the latter class of approaches, bringing them closer to the depth of the former, while handling telegraphic queries and retaining the advantage of corpus evidence over and above the KG. Very recently, (Yao et al., 2014) have concluded that for current benchmarks, deep parsing and shallow information extraction give comparable interpretation accuracy. The very recent work of (Yih et al., 2014) is similar in spirit to ours, but they do not unify segmentation and answer inference, along with corpus evidence, like we do.

3 Notation and examples

We use e_1, r, t_2, e_2 to represent abstract nodes and edges (MIDs in case of Freebase) from the KG, and $\widehat{e_1}, \widehat{r}, \widehat{t_2}$ to represent their textual mentions or hints, if any, in the query. s is a set of uninterpreted textual tokens in the query that are used to match and collect corpus contexts that lend evidence to candidate entities.

Figure 1 shows some telegraphic queries with possible segmentation into the above parts. Consider another example: dave navarro first band. 'Band' is a hint for type `/music/musical_group`, so it comprises $\widehat{t_2}$. Dave Navarro is an entity, with mention words 'dave navarro' comprising $\widehat{e_1}$. $\widehat{r}$ is made up of 'band', and represents the relation `/music/group_member/membership`. Finally, the word first cannot be mapped to any simple KG artifact, so are relegated to s (which makes the corpus a critical part of answer inference). We use s and $\widehat{s}$ interchangeably.

Generally, there will be enough noise and uncertainty that the search system should try out several of the most promising segmentations as shown in Figure 1. The accuracy of any specific segmentation is expected to be low in such adversarial settings. Therefore, support for an answer entity is aggregated over several segmentations. The expectation is that by considering multiple interpretations, the system will choose the entity with best supporting evidence from corpus and knowledge base.

4 Our Approach

Telegraphic queries are usually short, so we enumerate query token spans (with some restrictions, similar to beam search) to propose segmentations (§4.1). Candidate response entities are lined up for each interpretation, and then scored in a global model along with query segmentations (§4.2). §4.3 describes how model parameters are trained.

1: **input:** query token sequence q
2: initialize segmentations $\mathcal{I} = \varnothing$
3: $\mathcal{E}_1$ = (entity, mention) pairs from linker
4: **for all** $(e_1, \widehat{e_1}) \in \mathcal{E}_1$ **do**
5: assign label E_1 to mention tokens $\widehat{e_1}$
6: **for all** contiguous span $v \subset q \setminus \widehat{e_1}$ **do**
7: label each word $w \in v$ as T_2R
8: label other words $w \in q \setminus \widehat{e_1} \setminus v$ as S
9: add segments (E_1, T_2R, S) to $\mathcal{I}$
10: **end for**
11: **end for**
12: **return** candidate segmentations $\mathcal{I}$

Figure 2: Generating candidate query segmentations.

4.1 Generating candidate query segmentations

Each query token can have four labels, E_1, T_2, R, S, corresponding to the mentions of the base entity, target type, connecting relation, and context words. We found that segments hinting at T_2 and R frequently overlapped (e.g., 'author' in the query zhivago author). In our implementation, we simplified to three labels, E_1, T_2R, S, where tokens labeled T_2R are involved with both t_2 and r, the proposed structured target type and connecting relation. Another reasonable assumption was that the base entity mention and type/relation mentions are contiguous token spans, whereas context words can be scattered in multiple segments.

Figure 2 shows how candidate segmentations are generated. For step 3, we use TagMe (Ferragina and Scaiella, 2010), an entity linker backed by an entity gazette derived from our KG.

4.2 Graphical model

Based on the previous discussion, we assume that an entity-seeking query q is a sequence of tokens $q_1, q_2, \ldots$, and this can be partitioned into different kinds of subsequences, corresponding to e_1, r, t_2 and s, and denoted by a structured (vector) labeling $z = z_1, z_2, \ldots$. Given sequences q and z, we can separate out (possibly empty) token segments $\widehat{e_1}(q,z)$, $\widehat{t_2}(q,z)$, $\widehat{r}(q,z)$, and $\widehat{s}(q,z)$.

A query segmentation z becomes plausible in conjunction with proposals for e_1, r, t_2 and e_2 from the KG. The probability $\Pr(z, e_1, r, t_2, e_2|q)$ is modeled as proportional to the product of several potentials (Koller and Friedman, 2009) in a graphical model. In subsequent subsections, we will present the design of specific potentials.

- $\Psi_R(q,z,r)$ denotes the compatibility between the relation hint segment $\widehat{r}(q,z)$ and a proposed relation type r in the KG (§4.2.1).
- $\Psi_{T_2}(q,z,t_2)$ denotes the compatibility between the type hint segment $\widehat{t_2}(q,z)$ and a proposed target entity type t_2 in the KG (§4.2.2).
- $\Psi_{E_1,R,E_2,S}(q,z,e_1,r,e_2)$ is a novel corpus-based evidence potential that measures how strongly e_1 and e_2 appear in corpus snippets in the proximity of words in $\widehat{s}(q,z)$, and apparently related by relation type r (§4.2.3).
- $\Psi_{E_1}(q,z,e_1)$ denotes the compatibility between the query segment $\widehat{e_1}(q,z)$ and entity e_1 that it purportedly mentions (§4.2.4).
- $\Psi_S(q,z)$ denotes selector compatibility. Selectors are a fallback label, so this is pinned arbitrarily to 1; other potentials are balanced against this base value.
- $\Psi_{E_1,R,E_2}(e_1,r,e_2)$ is A if the relation $r(e_1,e_2)$ exists in the KG, and is $B > 0$ otherwise, for tuned/learnt constants $A > B > 0$. Note that this is a soft constraint ($B > 0$); if the KG is incomplete, the corpus may be able to supplement the required information.
- $\Psi_{E_2,T_2}(e_2,t_2)$ is 1 if e_2 belongs to t_2 and zero otherwise. In other words, candidate e_2s must be proposed to be instances of the proposed t_2 — this is a hard constraint, but can be softened if desired, like Ψ_{E_1,R,E_2}.

Figure 3 shows the relevant variable states as circled nodes, and the potentials as square factor nodes. To rank candidate entities e_2, we pin the node E_2 to each entity in turn. With E_2 pinned, we perform a MAP inference over all other hidden variables and note the score of e_2 as the product of the above potentials maximized over choices of all other variables: score$(e_2) =$

$$\begin{aligned}\max_{z,t_2,r,e_1} & \Psi_{T_2}(q,z,t_2)\Psi_R(q,z,r) \\ & \Psi_{E_1}(q,z,e_1)\Psi_S(q,z) \\ & \Psi_{E_2,T_2}(e_2,t_2)\Psi_{E_1,R,E_2}(e_1,r,e_2) \\ & \Psi_{E_1,R,E_2,S}(q,z,e_1,r,e_2). \qquad (1)\end{aligned}$$

We rank candidate e_2s by decreasing score, which is estimated by max-product message-passing (Koller and Friedman, 2009).

As noted earlier, any of the relation/type, or query entity partitions may be empty. To handle

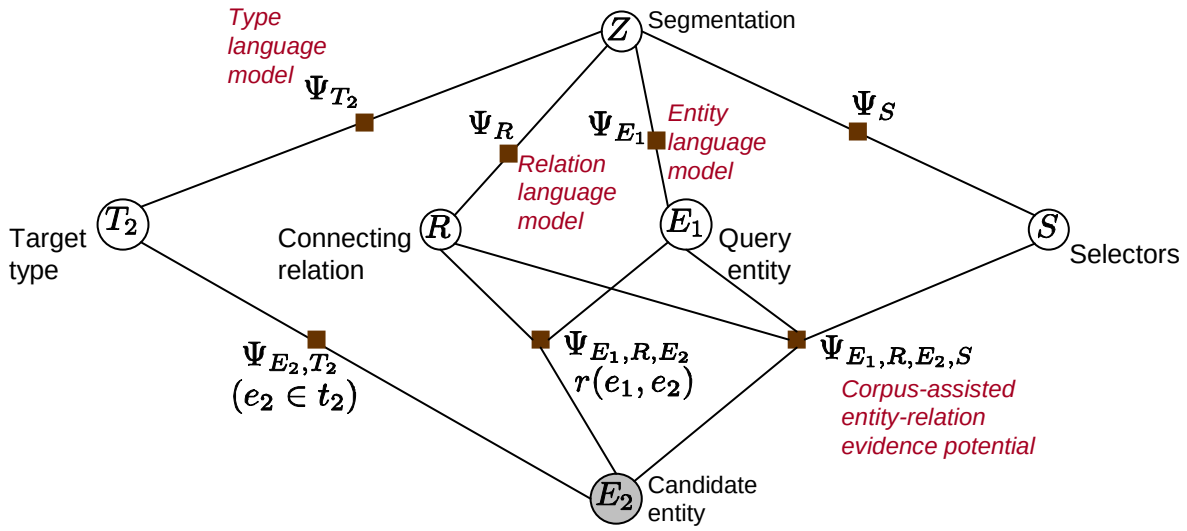

Figure 3: Graphical model for query segmentation and entity scoring. Factors/potentials are shown as squares. A candidate e_2 is observed and scored using equation (1). Query q is also observed but not shown to reduce clutter; most potentials depend on it.

this case, we allow each of the entity, relation or target type nodes in the graphical to take the value $\perp$ or 'null'. To support this, the value of the factor between the query segmentation node Z and $\Psi_{E1}(q, z, e_1)$, $\Psi_{T2}(q, z, t_2)$, and $\Psi_R(q, z, r)$) are set to suitable low values.

Next, we will describe the detailed design of some of the key potentials introduced above.

4.2.1 Relation language model for Ψ_R

Potential $\Psi_R(q, z, r)$ captures the compatibility between $\widehat{r}(q, z)$ and the proposed relation r. E.g., if the query is steve jobs death reason, and $\widehat{r}$ is (correctly chosen as) death reason, then the correct candidate r is `/people/deceased_person/cause_of_death`. An incorrect r is `/people/deceased_person/place_of_death`. An incorrect z may lead to $\widehat{r}(q, z)$ being jobs death.

Using corpus: Considerable variation may exist in how r is represented textually in a query. The relation language model needs to build a bridge between the formal r and the textual $\widehat{r}$, so that (un)likely r's have (small) large potential. Many approaches (Berant et al., 2013; Berant and Liang, 2014; Kwiatkowski et al., 2013; Yih et al., 2014) to this problem have been intensely studied recently. Given our need to process billions of Web pages efficiently, we chose a pattern-based approach (Nakashole et al., 2012): with each r, discover the most strongly associated phrase patterns from a reference corpus, then mark these patterns into much larger payload corpus.

We started with the 2000 (out of approximately 14000) most frequent relation types in Freebase, and the ClueWeb09 corpus annotated with Freebase entities (Gabrilovich et al., 2013). For each triple instance of each relation type, we located all corpus sentences that mentioned both participating entities. We made the crude assumption that if $r(e_1, e_2)$ holds and e_1, e_2 co-occur in a sentence then this sentence is evidence of the relationship. Each such sentence is parsed to obtain a dependency graph using the Malt Parser (Hall et al., 2014). Words in the path connecting the entities are joined together and added to a candidate phrase dictionary, provided the path is at most three hops. (Inspection suggested that longer dependency paths mostly arise out of noisy sentences or botched parses.) 30% of the sentences were thus retained. Finally, we defined

$$\Psi_R(q, z, r) = \frac{n(r, \widehat{r}(q, z))}{\sum_{p'} n(r, p')}, \qquad (2)$$

where p' ranges over all phrases that are known to hint at r, and $n(r, p)$ denotes the number of sentences where the phrase p occurred in the dependency path between the entities participating in relation r.

Assuming entity co-occurrence implies evidence is admittedly simplistic. However, the primary function of the relation model is to retrieve top-k relations that are compatible with the type/s

of e_1 and the given relation hint. Moreover, the remaining noise is further mitigated by the collective scoring in the graphical model. While we may miss relations if they are expressed in the query through obscure hints, allowing the relation to be $\perp$ acts as a safety net.

Using Freebase relation names: As mentioned earlier, queries may express relations differently as compared to the corpus. A relation model based solely on corpus annotations may not be able to bridge that gap effectively, particularly so, because of sparsity of corpus annotations or the rarity of Freebase triples in ClueWeb. E.g., for the Freebase relation `/people/person/profession`, we found very few annotated sentences. One way to address this problem is to utilize relation type names in Freebase to map hints to relation types. Thus, in addition to the corpus-derived relation model, we also built a language model that used Freebase relation type names as lemmas. E.g., the word 'profession' would contribute to the relation type `/people/person/profession`.

Our relation models are admittedly simple. This is mainly because telegraphic queries may express relations very differently from natural language text. As it is difficult to ensure precision of query interpretation stage, our models are geared towards recall. The system generates a large number of interpretations and relies on signals from the corpus and KG to bring forth correct interpretations.

4.2.2 Type language model for Ψ_{T_2}

Similar to the relation language model, we need a type language model to measure compatibility between t_2 and $\widehat{t_2}(q, z)$. Estimating the target entity type, without over-generalizing or over-specifying it, has always been important for QA. E.g., when $\widehat{t_2}$ is 'city', a good type language model should prefer t_2 as `/location/citytown` over `/location/location` while avoiding `/location/es_autonomous_city`.

A catalog like Freebase suggests a straightforward method to collect a type language model. Each type is described by one or more phrases through the link `/common/topic/alias`. We can collect these into a micro-'document' and use a standard Dirichlet-smoothed language model from IR (Zhai, 2008). In Freebase, an entity node (e.g., Einstein, `/m/0jcx`) may be linked to a type node (e.g. `/base/scientist/physicist`) using an edge with label `/type/object/type`.

But relation types provide additional clues to types of the endpoint entities. Freebase relation types have the form `/x/y/z`, where `x` is the domain of the relation, and `y` and `z` are string representations of the type of the entities participating in the relation. E.g., the (directed) relation type `/location/country/capital` connects from from `/location/country` to `/location/citytown`. Therefore, "capital" can be added to the set of descriptive phrases of entity type `/location/citytown`.

It is important to note that while we use Freebase link nomenclature for relation and type language models, our models are not incompatible with other catalogs. Indeed, most catalogs have established ways of deriving language models that describe their various structures. For example, most YAGO types are derived from WordNet synsets with associated phrasal descriptions (lemmas). YAGO relations also have readable names such as actedIn, isMarriedTo, etc. which can be used to estimate language models. DBPedia relations are mostly derived from (meaningfully) named attributes taken from infoboxes, hence they can be used directly. Furthermore, others (Wu and Weld, 2007) have shown how to associate language models with such relations.

4.2.3 Snippet scoring

The factor $\Psi_{E_1,R,E_2,S}(q, z, e_1, r, e_2)$ should be large if many snippets contain a mention of e_1 and e_2, relation r, and many high-signal words from s. Recall that we begin with a corpus annotated with entity mentions. Our corpus is not directly annotated with relation mentions. Therefore, we get from relations to documents via high-confidence phrases. Snippets are retrieved using a combined entity + word index, and scored for a given e_1, r, e_2, and selectors $\widehat{s}(q, z)$.

Given that relation phrases may be noisy and that their occurrence in the snippet may not necessarily mean that the given relation is being expressed, we need a scoring function that is cognizant of the roles of relation phrases and entities occurring in the snippets. In a basic version, $e_1, p, e_2, \widehat{s}$ are used to probe a combined entity+word index to collect high scoring snippets, with the score being adapted from BM25. The second, refined scoring function used a RankSVM-

style (Joachims, 2002) optimization.

$$\min_{\lambda,\xi} \|\lambda\|^2 + C\sum_{e^+,e^-} \xi_{e^+,e^-} \quad \text{s.t.}$$
$$\forall e^+, e^- : \quad \lambda \cdot f(q, D_{e^+}, e^+) + \xi_{e^+,e^-} \geq \lambda \cdot f(q, D_{e^-}, e^-) + 1. \tag{3}$$

where e^+ and e^- are positive and negative entities for the query q and $f(q, D_e, e)$ represents the feature map for the set of snippets D_e belonging to entity e. The assumption here is that all snippets containing e^+ are "positive" snippets for the query. f consolidates various signals like the number of snippets where e occurs near query entity e_1 and a relation phrase, or the number of snippets with high proportion of query IDF, hinting that e is a positive entity for the given query. A partial list of features used for snippet scoring is given in Figure 4.

Number of snippets with distance(e_2, $\widehat{e_1}$) $< k_1$ (k_1 = 5, 10)
Number of snippets with distance(e_2, relation phrase) $< k_2$ (k_2 = 3, 6)
Number of snippets with relation $r = \perp$
Number of snippets with relation phrases as prepositions
Number of snippets covering fraction of query IDF $> k_3$ (k_3 = 0.2, 0.4, 0.6, 0.8)

Figure 4: Sample features used for learning weights λ to score snippets.

4.2.4 Query entity model

Potential $\Psi_{E_1}(q, z, e_1)$ captures the compatibility between $\widehat{e_1}(q, z)$ (i.e., the words that mention e_1) and the claimed entity e_1 mentioned in the query. We used the TagMe entity linker (Ferragina and Scaiella, 2010) for annotating entities in queries. TagMe annotates the query with Wikipedia entities, which we map to Freebase, and use the annotation confidence scores as the potential $\Psi_{E_1}(q, z, e_1)$.

4.3 Discriminative parameter training with latent variables

We first set the potentials in (1) as explained in §4.2 (henceforth called 'Unoptimized'), and got encouraging accuracy. Then we rewrote each potential as

$$\Psi_\bullet(\cdots) = \exp\big(w_\bullet \cdot \phi_\bullet(\cdots)\big) \tag{4}$$
$$\text{or} \quad \log \textstyle\prod_\bullet \Psi_\bullet(\cdots) = \sum_\bullet w_\bullet \cdot \phi_\bullet(\cdots),$$

with $w_\bullet$ being a weight vector for a specific potential $\bullet$, and $\phi_\bullet$ being a corresponding feature vector. During inference, we seek to maximize

$$\max_{q,z,e_1,t_2,r} w \cdot \phi(q, z, e_1, t_2, r, e_2), \tag{5}$$

for a fixed w, to find the score of each candidate entity e_2. Here all $w_\bullet$ and $\phi_\bullet$ have been collected into unified weight and feature vectors w, ϕ. During training of w, we are given pairs of correct and incorrect answer entities e_2^+, e_2^-, and we wish to satisfy constraints of the form

$$\max_{q,z,e_1,t_2,r} w \cdot \phi(q, z, e_1, t_2, r, e_2^+) + \xi \geq 1 + \max_{q,z,e_1,t_2,r} w \cdot \phi(q, z, e_1, t_2, r, e_2^-), \tag{6}$$

because collecting e_2^+, e_2^- pairs is less work than supervising with values of z, e_1, t_2, r, e_2 for each query. Similar distant supervision problems were posed via bundle method by (Bergeron et al., 2008), and (Yu and Joachims, 2009), who used CCCP (Yuille and Rangarajan, 2006). These are equivalent in our setting. We use the CCCP style, and augment the objective with an additional entropy term as in (Sawant and Chakrabarti, 2013). We call this LVDT (latent variable discriminative training) in §5.

5 Experiments

5.1 Testbed

Corpus and knowledge graph: We used the ClueWeb09B (ClueWeb09, 2009) corpus containing 50 million Web documents. This corpus was annotated by Google with Freebase entities (Gabrilovich et al., 2013). The average page contains 15 entity annotations from Freebase. We used the Freebase KG and its links to Wikipedia.

Queries: We report on two sets of entity-seeking queries. A sample of about 800 well-formed queries from WebQuestions (Berant et al., 2013) were converted to telegraphic utterances (such as would be typed into commercial search engines) by volunteers familiar with Web search. We call this WQT (WebQuestions, telegraphic). Queries are accompanied by ground truth entities. The second data set, TREC-INEX, from (Sawant and Chakrabarti, 2013) has about 700 queries sampled from TREC and INEX, available at `http://bit.ly/WSpxvr`. These come with well-formed and telegraphic utterances, as well as ground truth entities.

There are some notable differences between these query sets. For WQT, queries were generated by using Google's query suggestions interface. Volunteers were asked to find answers using single Freebase pages. Therefore, by construction, queries retained can be answered using the Freebase KG alone, with a simple $r(e_1, ?)$ form. In contrast, TREC-INEX queries provide a balanced mix of t_2 and r hints in the queries, and direct answers from triples is relatively less available.

5.2 Implementation details

On an average, the pseudocode in Figure 2 generated 13 segmentations per query, with longer queries generating more segmentations than shorter ones.

We used an MG4J (Boldi and Vigna, 2005) based query processor, written in Java, over entity and word indices on ClueWeb09B. The index supplies snippets with a specified maximum width, containing a mention of some entity and satisfying a WAND (Broder et al., 2003) predicate over words in $\widehat{s}$. In case of phrases in the query, the WAND threshold was computed by adding the IDF of constituent words. The index returned about 330,000 snippets on average for WAND threshold of 0.6.

We retained the top 200 candidate entities from the corpus; increasing this horizon did not give benefits. We also considered as candidates for e_2 those entities that are adjacent to e_1 in the KG via top-scoring r candidates. In order to generate supporting snippets for an interpretation containing entity annotation e, we need to match e with Google's corpus annotations. However, relying solely on corpus annotations fails to retrieve many potential evidence snippets, because entity annotations are sparse. Therefore we probed the token index with the textual mention of e_1 in the query; this improved recall.

We also investigated the feasibility of our proposals for interactive search. There are three major processes involved in answering a query - generating potential interpretations, collecting/scoring snippets, and inference (MAP for Unoptimized and $w\phi(\cdot)$ for LVDT). For the WQT dataset, average time per query for each stage was approximately - 0.2, 16.6 and 1.3 seconds respectively. Our (Java) code did not optimize the bottleneck at all; only 10 hosts and no clever load balancing were used. We believe commercial search engines can cut this down to less than a second.

5.3 Research questions

In the rest of this section we will address these questions:

- For telegraphic queries, is our entity-relation-type-selector segmentation better than the type-selector segmentation of (Sawant and Chakrabarti, 2013)?
- When semantic parsers (Berant et al., 2013; Kwiatkowski et al., 2013) are subjected to telegraphic queries, how do they perform compared to our proposal?
- Are the KG and corpus really complementary as regards their support of accurate ranking of candidate entities?
- Is the prediction of r and t_2 from our approach better than a greedy assignment based on local language models?

We also discuss anecdotes of successes and failures of various systems.

5.4 Benefits of relation in addition to type

Figure 5 shows entity-ranking MAP, MRR, and NDCG@10 (n@10) for two data sets and various systems. "No interpretation" is an IR baseline without any KG. Type+selector is our implementation of (Sawant and Chakrabarti, 2013). Unoptimized and LVDT both beat "no interpretation" and "type+selector" by wide margins. (Boldface implies best performing formulation.) There are two notable differences between S&C and our work. First, S&C do not use the knowledge graph (KG) and rely on a noisy corpus. This means S&C fails to answer queries whose answers are found only in KG. This can be seen from WQT results; they perform only slightly better than the baseline. Second, even for queries that can be answered through the corpus alone, S&C miss out on two important signals that the query may provide - namely the query entity and the relation. Our framework not only provides a way to use a curated and high precision knowledge graph but also attempts to provide more reachability to corpus by the use of relational phrases.

In case of TREC-INEX, LVDT improves upon the unoptimized graphical model, where for WQT, it does not. Preliminary inspection suggests this is because WQT has noisy and incomplete ground truth, and LVDT trains to the noise; a non-convex

Dataset	Formulation	map	mrr	n@10
TREC -INEX	No interpretation	.205	.215	.292
	Type+selector	.292	.306	.356
	Unoptimized	.409	.419	.502
	LVDT	**.419**	**.436**	**.541**
WQT	No interpretation	.080	.095	.131
	Type+selector	.116	.152	.201
	Unoptimized	**.377**	**.401**	**.474**
	LVDT	.295	.323	.406

Figure 5: 'Entity-relation-type-selector' segmentation yields better accuracy than 'type-selector' segmentation.

objective makes matters worse. The bias in our unoptimized model circumvents training noise.

5.5 Comparison with semantic parsers

For TREC-INEX, both unoptimized and LVDT beat SEMPRE (Berant et al., 2013) convincingly, whether it is trained with Free917 or WebQuestions (Figure 6).

SEMPRE's relatively poor performance, in this case, is explained by its complete reliance on the knowledge graph. As discussed previously, the TREC-INEX dataset contains a sizable proportion of queries that may be difficult to answer using a KG alone. When SEMPRE is compared with our systems with a telegraphic sample of WebQuestions (WQT), results are mixed. Our Unoptimized model still compares favorably to SEMPRE, but with slimmer gains. As before, LVDT falls behind.

Dataset	Formulation	map	mrr	n@10
TREC -INEX	SEMPRE(Free917)	.154	.159	.186
	SEMPRE(WQ)	.197	.208	.247
	Unoptimized	.409	.419	.502
	LVDT	**.419**	**.436**	**.541**
WQT	SEMPRE(Free917)	.229	.255	.285
	SEMPRE(WQ)	.374	**.406**	.449
	Unoptimized	**.377**	.401	**.474**
	Jacana	.239	.256	.329
	LVDT	.295	.323	.406

Figure 6: Comparison with semantic parsers.

Our smaller gains over SEMPRE in case of WebQuestions is explained by how WebQuestions was assembled (Berant et al., 2013). Although Google's query suggestions gave an eclectic pool, only those queries survived that could be answered using a single Freebase page, which effectively reduced the role of a corpus. In fact, a large fraction of WQT queries cannot be answered well using the corpus alone, because FACC1 annotations are too sparse and rarely cover common nouns and phrases such as 'democracy' or 'drug overdose' which are needed for some WQT queries.

For WQT, our system also compares favorably with Jacana (Yao and Van Durme, 2014). Given that they subject their input to natural langauge parsing, their relatively poor performance is not unsurprsing.

5.6 Complementary benefits of KG & corpus

Figure 7 shows the synergy between the corpus and the KG. In all cases and for all metrics, using the corpus and KG together gives superior performance to using any of them alone. However, it is instructive that in case of TREC-INEX, corpus-only is better than KG-only, whereas this is reversed for WQT, which also supports the above argument.

Data	Formulation	map	mrr	n@10
TREC-INEX	Unoptimized (KG)	.201	.209	.241
	Unoptimized (Corpus)	.381	.388	.471
	Unoptimized (Both)	**.409**	**.419**	**.502**
	LVDT (KG only)	.255	.264	.293
	LVDT (Corpus)	.267	.272	.315
	LVDT (Both)	**.419**	**.436**	**.541**
WQT	Unoptimized (KG)	.329	.343	.394
	Unoptimized (Corpus)	.188	.228	.291
	Unoptimized (Both)	**.377**	**.401**	**.474**
	LVDT (KG only)	.257	.281	.345
	LVDT (Corpus only)	.170	.210	.280
	LVDT (Both)	**.295**	**.323**	**.406**

Figure 7: Synergy between KB and corpus.

5.7 Collective vs. greedy segmentation

To judge the quality of interpretations, we asked paid volunteers to annotate queries with an appropriate relation and type, and compared them with the interpretations associated with top-ranked entities. Results in Figure 8 indicate that in spite of noisy relation and type language models, our formulations produce high quality interpretations through collective inference.

Figure 9 demonstrates the benefit of collective inference over greedy segmentation followed by

Formulation	Type	Relation	Type/Rel
Unoptimized (top 1)	23	49	60
Unoptimized (top 5)	29	57	68
LVDT (top 1)	25	52	61
LVDT (top 5)	33	61	69

Figure 8: Fraction of queries (%) with correct interpretations of t_2, r, and t_2 or r, on TREC-INEX.

evaluation. Collective inference boosts absolute MAP by as much as 0.2.

Dataset	Formulation	map	mrr	n@10
TREC -INEX	Unoptimized (greedy)	.343	.347	.432
	Unoptimized	.409	.419	.502
	LVDT (greedy)	205	.214	.259
	LVDT	**.419**	.436	.541
WQT	Unoptimized (greedy)	.246	.271	.335
	Unoptimized	.377	.401	.474
	LVDT (greedy)	.212	.246	.317
	LVDT	.295	.323	.406

Figure 9: Collective vs. greedy segmentation

5.8 Discussion

Closer scrutiny revealed that collective inference often overcame errors in earlier stages to produce a correct ranking over answer entities. E.g., for the query automobile company makes spider the entity disambiguation stage fails to identify the car Alfa Romeo Spider (`/m/08ys39`). However, the interpretation stage recovers from the error and segments the query with Automobile (`/m/0k4j`) as the query entity e_1, `/organization/organization` and `/business/industry/companies` as target type t_2 and relation r respectively (from the relation/type hint 'company'), and *spider* as selector to arrive at the correct answer Alfa Romeo (`/m/09c50`). The corpus features also play a crucial role for queries which may not be accurately represented with an appropriate logical formula. For the query meg ryan bookstore movie, the textual patterns for the relation `ActedIn` in conjunction with the selector word 'bookstore' correctly identifies the answer entity You've Got Mail (`/m/014zwb`).

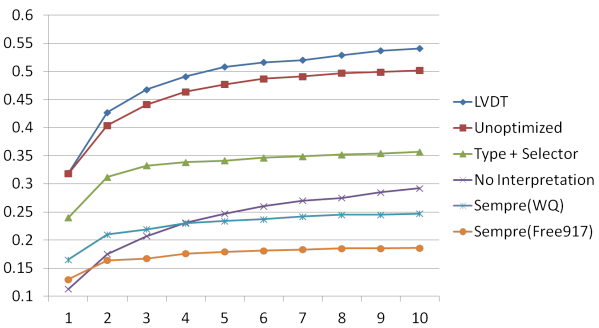

Figure 10: Comparison of various approaches for NDCG at rank 1 to 10, TREC-INEX dataset

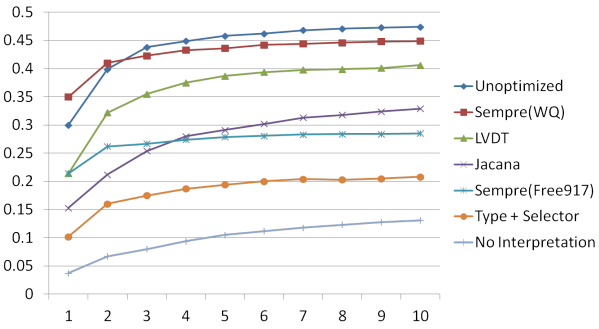

Figure 11: Comparison of various approaches for NDCG at rank 1 to 10, WQT dataset

We also analyzed samples of queries where our system did not perform particularly well. We observed that one of the recurring themes of these queries was that their answer entities had very little corpus support, and the type/relation hint mapped to too many or no candidate type/relations. For example, in the query south africa political system, the relevant type/relation hint 'political system' could not be mapped to `/government/form_of_government` and `/location/country/form_of_government` respectively.

6 Conclusion and future work

We presented a technique to partition telegraphic entity-seeking queries into functional segments and to rank answer entities accordingly. While our results are favorable compared to strong prior art, further improvements may result from relaxing our model to recognize multiple e_1s and rs. It may also help to deploy more sophisticated paraphrasing models (Berant and Liang, 2014) or word embeddings (Yih et al., 2014) for relation hints. It would also be interesting to supplement entity-linked corpora and curated KGs with extracted triples (Fader et al., 2014). Another possibility is to apply the ideas presented here to well-formed questions.

References

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *ACL Conference*.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Charles Bergeron, Jed Zaretzki, Curt Breneman, and Kristin P. Bennett. 2008. Multiple instance ranking. In *ICML*, pages 48–55. ACM.

Paolo Boldi and Sebastiano Vigna. 2005. MG4J at TREC 2005. In Ellen M. Voorhees and Lori P. Buckland, editors, *TREC*, number SP 500-266 in Special Publications. NIST.

Andrei Z. Broder, David Carmel, Michael Herscovici, Aya Soffer, and Jason Zien. 2003. Efficient query evaluation using a two-level retrieval process. In *CIKM*, pages 426–434. ACM.

Tao Cheng and Kevin Chen-Chuan Chang. 2010. Beyond pages: supporting efficient, scalable entity search with dual-inversion index. In *EDBT*. ACM.

ClueWeb09. 2009. `http://www.lemurproject.org/clueweb09.php/`.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2014. Open question answering over curated and extracted knowledge bases. In *SIGKDD Conference*.

Paolo Ferragina and Ugo Scaiella. 2010. TAGME: on-the-fly annotation of short text fragments (by wikipedia entities). *CoRR/arXiv*, abs/1006.3498. `http://arxiv.org/abs/1006.3498`.

Evgeniy Gabrilovich, Michael Ringgaard, and Amarnag Subramanya. 2013. FACC1: Freebase annotation of ClueWeb corpora. `http://lemurproject.org/clueweb12/`, June. Version 1 (Release date 2013-06-26, Format version 1, Correction level 0).

Sean Gallagher. 2012. How Google and Microsoft taught search to 'understand' the Web. ArsTechnica article. `http://goo.gl/NWs0zT`.

Jiafeng Guo, Gu Xu, Xueqi Cheng, and Hang Li. 2009. Named entity recognition in query. In *SIGIR Conference*, pages 267–274. ACM.

Johan Hall, Jens Nilsson, and Joakim Nivre. 2014. Maltparser. `http://www.maltparser.org/`.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *SIGKDD Conference*, pages 133–142. ACM.

Gjergji Kasneci, Fabian M. Suchanek, Georgiana Ifrim, Maya Ramanath, and Gerhard Weikum. 2008. NAGA: Searching and ranking knowledge. In *ICDE*. IEEE.

Daphne Koller and Nir Friedman. 2009. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.

Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke S. Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching. In *EMNLP Conference*, pages 1545–1556.

Xiaonan Li, Chengkai Li, and Cong Yu. 2010. EntityEngine: Answering entity-relationship queries using shallow semantics. In *CIKM*, October. (demo).

Yanen Li, Bo-Jun Paul Hsu, ChengXiang Zhai, and Kuansan Wang. 2011. Unsupervised query segmentation using clickthrough for information retrieval. In *SIGIR Conference*, pages 285–294. ACM.

Percy Liang. 2013. Lambda dependency-based compositional semantics. Technical Report arXiv:1309.4408, Stanford University. `http://arxiv.org/abs/1309.4408`.

Thomas Lin, Patrick Pantel, Michael Gamon, Anitha Kannan, and Ariel Fuxman. 2012. Active objects: Actions for entity-centric search. In *WWW Conference*, pages 589–598. ACM.

Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. 2012. PATTY: A taxonomy of relational patterns with semantic types. In *EMNLP Conference*, EMNLP-CoNLL '12, pages 1135–1145. ACL.

Patrick Pantel, Thomas Lin, and Michael Gamon. 2012. Mining entity types from query logs via user intent modeling. In *ACL Conference*, pages 563–571, Jeju Island, Korea, July.

Fernando Pereira. 2013. Meaning in the wild. Invited talk at EMNLP Conference. `http://hum.csse.unimelb.edu.au/emnlp2013/invited-talks.html`.

Jeffrey Pound, Alexander K. Hudek, Ihab F. Ilyas, and Grant Weddell. 2012. Interpreting keyword queries over Web knowledge bases. In *CIKM*.

Nikos Sarkas, Stelios Paparizos, and Panayiotis Tsaparas. 2010. Structured annotations of Web queries. In *SIGMOD Conference*.

Uma Sawant and Soumen Chakrabarti. 2013. Learning joint query interpretation and response ranking. In *WWW Conference*, Brazil.

Fei Wu and Daniel S Weld. 2007. Automatically semantifying Wikipedia. In *CIKM*, pages 41–50.

Mohamed Yahya, Klaus Berberich, Shady Elbassuoni, Maya Ramanath, Volker Tresp, and Gerhard Weikum. 2012. Natural language questions for the Web of data. In *EMNLP Conference*, pages 379–390, Jeju Island, Korea, July.

Xuchen Yao and Benjamin Van Durme. 2014. Information extraction over structured data: Question answering with Freebase. In *ACL Conference*. ACL.

Xuchen Yao, Jonathan Berant, and Benjamin Van Durme. 2014. Freebase QA: Information extraction or semantic parsing? In *ACL 2014 Workshop on Semantic Parsing (SP14)*.

Wen-tau Yih, Xiaodong He, and Christopher Meek. 2014. Semantic parsing for single-relation question answering. In *ACL Conference*. ACL.

Chun-Nam John Yu and Thorsten Joachims. 2009. Learning structural SVMs with latent variables. In *ICML*, pages 1169–1176. ACM.

A. L. Yuille and Anand Rangarajan. 2006. The concave-convex procedure. *Neural Computation*, 15(4):915–936.

ChengXiang Zhai. 2008. Statistical language models for information retrieval: A critical review. *Foundations and Trends in Information Retrieval*, 2(3):137–213, March.