On the Modeling of Entities for Ad-Hoc Entity Search in the Web of Data

Robert Neumayer, Krisztian Balog, and Kjetil Nørvåg

Norwegian University of Science and Technology, Trondheim, Norway
{robert.neumayer,krisztian.balog,kjetil.norvag}@idi.ntnu.no

Abstract. The Web of Data describes objects, entities, or "things" in terms of their attributes and their relationships, using RDF statements. There is a need to make this wealth of knowledge easily accessible by means of keyword search. Despite recent research efforts in this direction, there is a lack of understanding of how structured semantic data is best represented for text-based entity retrieval. The task we are addressing in this paper is ad-hoc entity search: the retrieval of RDF resources that represent an entity described in the keyword query. We build upon and formalise existing entity modeling approaches within a generative language modeling framework, and compare them experimentally using a standard test collection, provided by the Semantic Search Challenge evaluation series. We show that these models outperform the current state-of-the-art in terms of retrieval effectiveness, however, this is done at the cost of abandoning a large part of the semantics behind the data. We propose a novel entity model capable of preserving the semantics associated with entities, without sacrificing retrieval effectiveness.

1 Introduction

In recent years the Web of Data (WoD) has emerged as a way of exposing structured data on the Web. The past three years in particular have seen a significant increase both in the number of knowledge bases published as Linked Data (such as DBpedia, Freebase, and others) and in the availability of metadata embedded inside web pages (RDF, RDFa, Microformats, and others) [24]. These knowledge repositories typically contain entities (such as people, locations, organisations, etc.) and the relationships between them (such as birthPlace, academicAdvisor, parentCompany, etc.). Each entity is uniquely identified by its URI (Uniform Resource Identifier). WoD datasets can be queried using formal, structured query languages such as SPARQL. Issuing such queries, however, is beyond the capabilities of ordinary Web users as it requires the knowledge of the underlying schema as well as that of the query language. Also, SPARQL-like languages are often too restrictive, and Boolean-type matches cannot provide a ranked list of results that users would prefer seeing [14].

Keyword search, which has become the main paradigm to perform search tasks on the Web, has recently emerged as a viable alternative. Indeed, there is a growing body of work on adapting IR ranking models for searching in RDF data, e.g., [5, 14, 23]. The introduction of the Semantic Search (SemSearch) evaluation series [4, 15] has further propelled research in this direction, by providing a common evaluation platform to

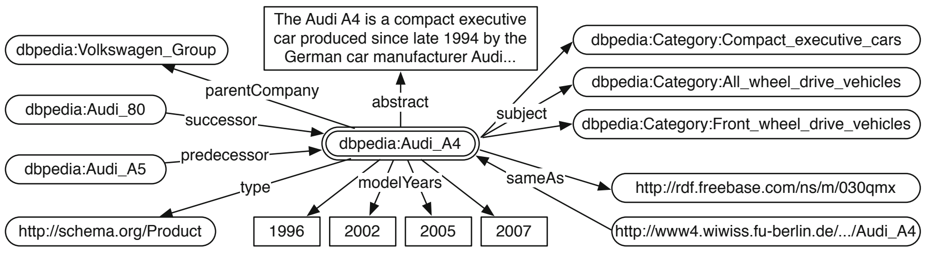
The Audi A4 is a compact executive car produced since late 1994 by the German car manufacturer Audi...
dbpedia:Volkswagen_Group
parentCompany
abstract
dbpedia:Audi_80
successor
dbpedia:Audi_A4
predecessor
dbpedia:Audi_A5
type
modelYears
http://schema.org/Product
1996
2002
2005
2007
subject
dbpedia:Category:Compact_executive_cars
dbpedia:Category:All_wheel_drive_vehicles
dbpedia:Category:Front_wheel_drive_vehicles
sameAs
http://rdf.freebase.com/ns/m/030qmx
http://www4.wiwiss.fu-berlin.de/.../Audi_A4

Fig. 1. Excerpt from an RDF graph. Rounded rectangles represent entities (i.e., URIs), rectangles denote attributes (i.e., literal values). URIs are shortened and predicate prefixes are ignored.

empirically assess methods and algorithms devised for the task that has been termed *ad-hoc entity retrieval*: "answering arbitrary information needs related to particular aspects of objects [entities], expressed in unconstrained natural language and resolved using a collection of structured data" [24]. Since WoD is inherently organised around objects or entities, having entities as the unit of retrieval follows naturally.

Commonly, the ad-hoc entity retrieval task is approached by adapting standard document retrieval methods. A textual representation ("pseudo document") is built for each entity, and these representations can then be ranked using conventional IR models. The main challenge, of course, is how to obtain these textual representations from structured data. WoD conceptually forms a large, directed, labelled graph with nodes corresponding to entities and edges denoting relationships, and is described in the form of subject-predicate-object (SPO) triples of the RDF data model; Figure 1 shows a small excerpt from an RDF graph centred around a given entity.

A natural solution would be to represent each entity using a fielded structure, where fields correspond to predicates (i.e., arrows on Figure 1) and associated nodes (or rather, the text extracted from them) are used as field values. These representations can then be ranked using any fielded document retrieval model, such as BM25F [25] or the mixture of language models (MLM) [20]. However, with this approach the number of document fields soon becomes computationally prohibitive, making the estimation of field weights intractable. A commonly used workaround is to group predicates together into a small set of predefined categories, and as such, create documents comprising of only a handful of fields. This grouping (or "predicate folding") can be based on, for example, the type of predicates (attributes, in/out-relations, etc.) [23] or on their (manually determined) importance [5]. This leads to a data model where the optimisation of field weights is easily tractable, even using exhaustive search over the parameter space. While this approach seems to work well in practice, it seriously limits the semantic expressivity of entity models, as it is no longer possible to access the content of individual predicates or might be too dependent on the data collection.

The overall research question we address in this paper is how to represent entities in the Web of Data for the purpose of text-based retrieval. As our first step on the path to investigating this issue, in Section 2 we formalise the aforementioned two entity modeling strategies within a generative language modeling framework. One approach (Unstructured Entity Model) collapses all text associated with the entity into a single flat-text representation. The other approach (Structured Entity Model) groups predicates together into a small number of categories and considers their weighted combination. We perform an experimental evaluation of the two models using the 2010 and 2011 test sets of the Semantic Search Challenge evaluation campaign in Section 3. We find that these models outperform the current state-of-the-art in terms of retrieval effectiveness on these collections. However, this is done at the cost of abandoning a large part of the semantics behind the data. Subsequently, in Section 4, we propose a novel entity model (Hierarchical Entity Model) capable of preserving the semantics associated with entities. It uses the idea of having a two-level hierarchy for entity representation, one based on the predicate types, another based on the individual predicates. We report on experiments using our hierarchical model in Section 5 and show that modeling individual predicates of a given type is more effective than folding their contents into a flat representation. We put our work in a broader context in Section 6. Finally, we conclude with a discussion of our findings and outline our plans for future research in Section 7.

2 Baseline Entity Models

In this section we formalise and draw upon two existing entity modeling approaches within a generative language modeling framework.

2.1 Retrieval Framework

Language models (LMs) are attractive because of their solid theoretical foundations that couples with good empirical performance [26]. LMs have been successfully applied to a wide range of entity-related search tasks; see, e.g., [1, 8, 14, 18].

We rank candidate entities (e) according to their probability of being relevant given query q: $P(e|q)$. Instead of estimating this probability directly, we apply Bayes' rule and drop the denominator as it does not influence the ranking (for a given query): $P(e|q) = \frac{P(q|e)P(e)}{P(q)} \overset{\text{rank}}{=} P(q|e)P(e)$. Here, $P(e)$ is the prior probability of choosing a particular entity e, that we subsequently attempt to draw the query q from, with probability $P(q|e)$. Here, we assume that $P(e)$ is uniform, thus, does not affect the ranking. Priors could otherwise be used to incorporate query-independent features [6, 10].

Each entity e is represented by a multinomial probability distribution over the vocabulary of terms. The entity model θ_e is used to predict how likely the entity would produce a given term t, that is, $P(t|\theta_e)$. Assuming that query terms are sampled identically and independently, the query likelihood is obtained by taking the product across all the terms in the query, such that: $P(q|\theta_e) = \prod_{t \in q} P(t|\theta_e)^{tf(t,q)}$, where $tf(t,q)$ is the (raw) frequency of term t in the query. Note that $P(t|\theta_e) > 0$ must be ensured for all vocabulary terms, otherwise the product might end up being zero.

The main question we will be concerned with for the remainder of this paper is the estimation of the entity model, i.e., the probability distribution, $P(t|\theta_e)$.

2.2 Unstructured Entity Model

The simplest approach to constructing entity models is to fold all text associated with the entity into one “bag-of-words"; see Figure 2(a) for an illustration. Following the

Text
Audi A4
1996 2002 2005 2007
The Audi A4 is a compact executive car...
Volkswagen Group
Compact executive cars
All wheel drive vehicles
Front wheel drive vehicles
Product
Audi A4
Audi 80
Audi A5
Audi A4

(a) Unstructured Entity Model

Pred. type	Value
Name	Audi A4
Attributes	1996 2002 2005 2007
	The Audi A4 is a compact executive car...
OutRelations	Volkswagen Group
	Compact executive cars
	All wheel drive vehicles
	Front wheel drive vehicles
	Product
	Audi A4
InRelations	Audi 80
	Audi A5
	Audi A4

(b) Structured Entity Model

Fig. 2. Examples of baseline entity models corresponding to Figure 1. URIs are resolved (i.e., replaced with the name of the corresponding entity).

standard language modeling approach to document retrieval, we implement the entity model as a Dirichlet-smoothed multinomial distribution:

$$P(t|\theta_e) = \frac{tf(t,e) + \mu P(t|\theta_c)}{|e| + \mu}, \tag{1}$$

where $tf(t,e)$ is the raw frequency of term t in the representation of e and $|e|$ is the size of this representation, i.e., $\sum_t tf(t,e)$. The smoothing parameter μ is set to the average entity representation size in the collection. The term generation process is shown in Figure 3 (Left).

Several SemSearch participants employed variations of this approach: considering only triples where the entity stands as a subjects (thereby ignoring incoming relations) [3, 8, 19] or extracting text only from the subject itself [13].

2.3 Structured Entity Model Using Predicate-Folding

Instead of folding all predicates together, they might be grouped into multiple categories in order to preserve some of the original structure. Prior work presents examples of such grouping based on the type of the predicates [3, 8, 16, 23] or based on their manually determined importance [5]. We group RDF triples into four main predicate types p_t for a given entity e as follows:

- *Name*: the subject is e, the object is a literal, and the predicate comes from a predefined list (e.g.,*foaf:name* or *rdfs:label*) or ends with "name," "label," or "title".
- *Attributes*: the subject is e, the object is a literal, and the predicate is not of type name.
- *OutRelations*: the subject is e and the object is a URI. We resolve the URI by replacing it with the name of the corresponding entity; see Section 3.1 for details.
- *InRelations*: e stands as the object; the subject entity URI is resolved.

Figure 2(b) presents an example. A separate language model $\theta_e^{p_t}$ is estimated for each predicate type from all predicates of that type (associated with the given entity):

$$P(t|\theta_e^{p_t}) = \frac{tf(t,p_t,e) + \mu_{p_t} P(t|\theta_c^{p_t})}{|p_t,e| + \mu_{p_t}}, \tag{2}$$

where $tf(t, p_t, e)$ is the sum of the term frequencies for t for all predicates of type p_t associated with e, and $|p_t, e| = \sum_t tf(t, p_t, e)$. Essentially, we concatenate all text from the predicates of type p_t and then apply Dirichlet smoothing using a collection-wide background model $P(t|\theta_c^{p_t})$ (that is, a maximum-likelihood estimate from all predicates of that type in the collection). The smoothing parameter μ_{p_t} is set to the average predicate type representation length in the collection.

The entity model is a linear mixture of the predicate type language models ($P(t|\theta_e^{p_t})$), weighted with the importance of that predicate type ($P(p_t)$):

$$P(t|\theta_e) = \sum_{p_t} P(t|\theta_e^{p_t})P(p_t) \tag{3}$$

Figure 3 (Middle) illustrates the term generation process. Viewing entities as documents and predicate types as document fields, this model is equivalent with the Mixture of Language Models approach by Ogilvie and Callan [20].

3 Evaluation of Baseline Entity Models

In this section we first introduce our experimental setup, then perform an evaluation of our baseline entity models.

3.1 Experimental Setup

We use the test suites of the 2010 and 2011 editions of the Semantic Search (SemSearch) Challenge [4, 15]. The task we address is ad-hoc entity search: given a keyword query, targeting a particular entity, provide a ranked list of relevant entities, identified by their URIs. The data set, queries, and relevance assessments are publicly available.[1]

Data set. The data collection used is the Billion Triple Challenge 2009 dataset. It was mainly crawled during February/March 2009 and comprises about 1.14 billion RDF statements and describes entities from domains like *dbpedia.org*, *livejournal.com* or *geonames.org*.[2] In our evaluations we considered the 500 most frequent predicates.

Topics and relevance assessments. There are two topic sets, consisting of 92 and 50 keyword queries for years 2010 and 2011, respectively. The queries were sampled from web search engine logs. Relevance judgments were obtained using Amazon's Mechanical Turk. Human assessors were presented with a simplified HTML summary of entities and had to judge relevance on a 3-point scale (excellent, fair, and irrelevant).

Evaluation metrics. We use standard IR evaluation metrics: Mean Average Precision (MAP), Precision at rank 10 (P@10), and Normalized Discounted Cumulative Gain (NDCG).[3] To check for significant differences between runs, we use a two-tailed paired t-test and write $^{\dagger}/^{\ddagger}$ to denote significance at the 0.05/0.01 levels, respectively.

Resource resolution. If a relation (predicate targeted at a URI rather than a literal value) points to an entity within the collection, we replace the URI with that entity's name. Otherwise, we extract terms from the relative part of the URI.

[1] `http://km.aifb.kit.edu/ws/semsearch{10|11}`
[2] `http://vmlion25.deri.ie/`
[3] Note that the two editions used different gain values for computing NDCG scores: 3 and 2 (2010) vs. 2 and 1 (2011) for excellent and fair, respectively. We use these values unchanged.

Table 1. Retrieval results using the Unstructured Entity Model. Significance is tested against the ALL setting (first line). Best scores for each topic set are typeset boldface

	2010			2011		
Predicate types	**MAP**	**P@10**	**NDCG**	**MAP**	**P@10**	**NDCG**
ALL	0.2069	0.3141	0.3827	**0.2072**	0.1880	**0.2947**
Name-only	$0.2054^{\dagger}$	0.3043	0.3969	0.2004	0.1900	0.3097
Attributes-only	$0.2058^{\ddagger}$	$0.3391^{\dagger}$	0.3915	$0.2200^{\dagger}$	0.1900	$0.3330^{\dagger}$
OutRelations-only	$0.0866^{\ddagger}$	$0.1761^{\ddagger}$	$0.2185^{\ddagger}$	$0.0935^{\ddagger}$	$0.1020^{\ddagger}$	$0.1667^{\ddagger}$
InRelations-only	$0.0955^{\ddagger}$	$0.1967^{\ddagger}$	$0.2257^{\ddagger}$	$0.0689^{\ddagger}$	$0.0980^{\ddagger}$	$0.1540^{\ddagger}$
ALL-but-Name	$0.1568^{\ddagger}$	$0.2620^{\ddagger}$	$0.3210^{\ddagger}$	$0.1563^{\ddagger}$	$0.1440^{\ddagger}$	$0.2467^{\ddagger}$
ALL-but-Attributes	$0.1820^{\ddagger}$	$0.2859^{\ddagger}$	$0.3416^{\ddagger}$	$0.1637^{\ddagger}$	$0.1600^{\ddagger}$	$0.2419^{\ddagger}$
ALL-but-OutRelations	$0.2029^{\ddagger}$	0.3109	$0.3685^{\ddagger}$	$0.2002^{\ddagger}$	0.1800	0.2826
ALL-but-InRelations	**0.2125**	**0.3196**	**0.3848**	$0.2037^{\ddagger}$	**0.1900**	0.2826

3.2 Unstructured Entity Model

Recall that this model creates an unstructured entity representation by collapsing text associated with the entity into one bag-of-words. We perform three sets of experiments, and report the results in Table 1. First, identified by the 'ALL' row, we consider all four predicate types. Next, shown in the 'field type-only' rows, we use only a single predicate type at a time. Finally, in the rows named 'ALL-but-field type,' we present results by omitting one of the types in turn.

Our findings are as follows. The absolute performance of the ALL model (that simply uses all text associated with the entity) is remarkable; on the 2010 topic set it outperforms the best SemSearch 2010 submission (MAP=0.1919) [15], while on the 2011 set it would have ranked third (best was MAP=0.2346) [4]. As for the individual predicate types, *Name* and *Attributes* perform best, with only minor differences between the two; in fact, either of these predicates types alone is on a par with the ALL model. The scores for incoming and outgoing relations are much lower. Looking at the combinations of all but one predicate types, *Name* contributes the most and *Attributes* comes second; without either, the scores drop substantially. Removing either incoming or outgoing relations has hardly any overall impact; in the 'ALL-but-InRelations,' 2010 case the scores marginally improve, but the difference is not statistically significant.

3.3 Structured Entity Model

Our second baseline model employs a weighted combination of four language models, corresponding to predicate types. We consider three different configurations in Table 2: (1) equal weights on all types, (2) all weights allocated to *Name* and *Attributes*, in equal proportion, and (3) most of the weights assigned to *Name* and *Attributes*, again, but this time taking relations too into account. In the lack of training data or any background knowledge about the collection or queries, (1) is a natural choice. (2) and (3) are motivated by the results from the previous subsection; however, knowing from the task definition that each query targets a particular entity, one could intuitively argue for such weight distributions. Note that when a single predicate type is used, the Structured Entity Model is equivalent to the unstructured case (Table 1, middle block).

Table 2. Retrieval results using Structured Entity Models. Significance is tested against the ALL setting of the Unstructured Entity Model (Table 1, first line). Best scores are typeset boldface

Pred. type weights				2010			2011		
Name	**Attr**	**OutRel**	**InRel**	**MAP**	**P@10**	**NDCG**	**MAP**	**P@10**	**NDCG**
.25	.25	.25	.25	**0.2816**‡	0.3989‡	**0.4943**‡	0.2605‡	**0.2420**‡	0.3966‡
.5	.5			0.2490‡	0.3663‡	0.4608‡	0.2506‡	0.2300‡	0.3845‡
.35	.35	.15	.15	0.2804‡	**0.4043**‡	0.4934‡	**0.2614**‡	0.2300‡	**0.3968**‡

All settings improve significantly and substantially over the unstructured baseline. We find, somewhat surprisingly, that the uniform weighting performs best of all; these numbers are the highest that were ever reported for either topic sets (which are: MAP= 0.2705 for 2010 [5] and MAP=0.2346 for 2011 [4]). Even though *Name* and *Attributes* were shown to be the two most 'useful' predicate types, using them without relational information is clearly suboptimal (row 2 vs. rows 1 and 3). The third setting (row 3) suggests that as long as all predicate types are considered, the model is robust with respect to the actual weight distribution used. We experimented with other configurations too, but none of them improved significantly over the uniform weighting.

4 Hierarchical Entity Model

In this section we introduce a novel entity modeling approach. Before presenting our proposal, we briefly review the considerations leading to the choice of this particular model. (1) In a heterogeneous environment the number of distinct predicates is huge. It is not feasible to optimise their weights directly (because of the computational complexity and because of the enormous amounts of training material it would require). (2) Entities are sparse with respect to the different predicates, i.e., most entities have only a handful of distinct predicates associated with them. (3) As we have shown with the Structured Entity Model, folding predicates based on their type is a viable alternative that works well in practice. Nevertheless, we need a different solution if we wish to preserve the semantics in the entity model, i.e., keep individual predicates and their contents accessible (and possibly exploit this information in the retrieval model). Users can only be offered to make use of single predicates if they are preserved individually, such kind of faceted search may not improve effectiveness in the context of the Semantic Search Challenge, but might be a requirement given from the user side.

The main idea behind our model is to organise information belonging to a given entity into a hierarchy of two levels, with predicate types (i.e., name, attributes, incoming and outgoing relations) on the first level and individual predicates of that type on the second level. This preserves the original structure associated with the given entity, and allows for setting the importance individual predicates conditioned on their type and on the entity. Formally, this is expressed as follows:

$$\begin{aligned} P(t|\theta_e) &= \sum_{p_t} P(t|p_t,e)P(p_t|e) && (4)\\ &= \sum_{p_t}\Big(\sum_{p\in p_t} P(t|p,p_t)P(p|p_t,e)\Big)P(p_t|e) && (5)\end{aligned}$$

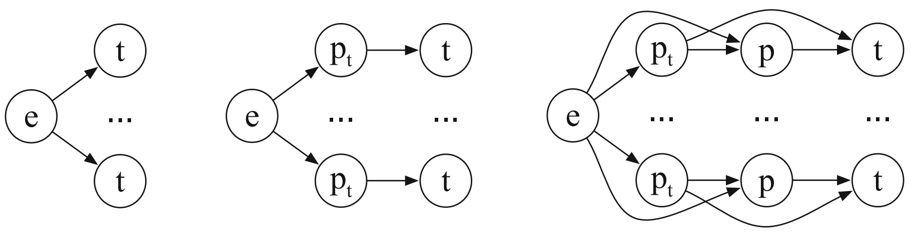

Fig. 3. Graphical representations of entity models: (Left) Unstructured Entity Model, (Middle) Structured Entity Model using Predicate-folding, (Right) Hierarchical Entity Model

The term generation process under this model is shown in Figure 3 (Right). Next, we discuss the three components from Eq. 5: term generation ($P(t|p, p_t)$), predicate generation ($P(p|p_t, e)$), and predicate type generation ($P(p_t|e)$).

Term Generation. The importance of a term is jointly determined by the predicate in which it occurs as well as all other predicates of that type associated with the entity:

$$P(t|p, p_t) = (1 - \lambda)P(t|p) + \lambda P(t|\theta_e^{p_t}), \tag{6}$$

where $P(t|p)$ is a maximum-likelihood estimate (i.e., the relative frequency of term t in predicate p) and $P(t|\theta_e^{p_t})$ is the Dirichlet-smoothed LM for predicate type p_t, estimated using Eq. 2. The parameter $\lambda \in [0..1]$ controls the influence of the predicate type model. For the sake of simplicity, we set λ to 0.5 in our experiments.

Predicate Generation. The importance of a given predicate p is conditioned on the type of the predicate p_t and the entity e: $P(p|p_t, e)$. We consider four natural options:[4]

- *Uniform.* All the predicates of the same type are treated equally important: $P(p|p_t, e) = 1/n(e, p_t)$, where $n(e, p_t)$ is the number of predicates of type p_t assigned to e.[5]
- *Length.* The probability mass is allocated to predicates proportional to their length: $P(p|p_t, e) = |p, e|/|p_t, e|$, where $|p, e|$ and $|p_t, e|$ are the lengths of p and p_t, respectively, measured in the number of terms they contain.
- *Average length.* We use the average length of the predicate p in the collection relative to the average length of all predicates of type p_t.
- *Popularity.* We regard popular predicates more important, where popularity is measured in terms of the number of triples that have the given predicate: $P(p|p_t, e) = n(p)/n(p_t)$, where $n(p)$ and $n(p_t)$ are the total number of triples in the collection with predicate p and predicate type p_t, respectively.

[4] It is by design that predicate importance is independent of the query; this way other, possibly computationally heavy, alternatives for setting this probability could be estimated offline.

[5] Since predicates encoding the entity name are all equally important, we do not vary their importance, hence we always use the uniform distribution for $p_t =$ "$name$".

Table 3. Retrieval results using Hierarchical Entity Models using a single predicate type

Predicate	**Predicate**	**2010**			**2011**		
type	**weighting**	**MAP**	**P@10**	**NDCG**	**MAP**	**P@10**	**NDCG**
Attributes	Uniform	0.2350‡	0.3261	0.4403‡	0.2423‡	0.2140	0.3738
	Length	0.2116‡	0.2870‡	0.4149	0.2292‡	0.2060	0.3477
	AvgLength	0.2129‡	0.2859‡	0.4144	0.2287‡	0.2060	0.3511
	Popularity	0.2318‡	0.3109†	0.4385‡	0.2514‡	0.2140	0.3791
OutRelations	Uniform	0.1185‡	0.1783	0.2607	0.1221‡	0.1220†	0.2046‡
	Length	0.0915‡	0.1663	0.2122	0.0980‡	0.0920	0.1647
	AvgLength	0.0908‡	0.1533†	0.2131	0.0958‡	0.0940	0.1677
	Popularity	0.1067‡	0.1674	0.2439	0.1233‡	0.1260†	0.2070‡
InRelations	Uniform	0.1073‡	0.2054†	0.2387	0.0800‡	0.1160†	0.1749‡
	Length	0.0941‡	0.1793†	0.2188	0.0703	0.0940	0.1593
	AvgLength	0.0921‡	0.1696‡	0.2212	0.0776‡	0.0980	0.1751‡
	Popularity	0.1117‡	0.2022	0.2452†	0.0891‡	0.1240†	0.1903‡

Table 4. Retrieval results using Hierarchical Entity Models and uniform term weighting. Significance is tested against the Structured Entity Model (corresponding row in Table 2). Best scores for each year are typeset boldface.

Pred. type				**2010**			**2011**		
Name	**Att**	**OutRel**	**InRel**	**MAP**	**P@10**	**NDCG**	**MAP**	**P@10**	**NDCG**
.25	.25	.25	.25	0.2349‡	0.3261‡	0.4394‡	0.2423‡	0.2140‡	**0.3738**†
.5	.5			0.2242‡	0.3359‡	0.4125‡	0.2038‡	0.2000‡	0.3195‡
.35	.35	.15	.15	**0.2561**‡	**0.3641**‡	**0.4614**‡	**0.2436**‡	**0.2240**	0.3717†

Predicate Type Generation. The model in Eq. 5 allows us to set predicate type importance on a per-entity basis. To simplify matters, however, we make the conditional independence assumption between predicate types and entities, hence $P(p_t|e) = P(p_t)$. This allows us to use estimation methods (or the actual values) from the Structured Entity Model, as this component is identical in the two models (cf. Eq. 3).

5 Evaluation of the Hierarchical Entity Model

In order to evaluate the hierarchical entity model, we first show the results obtained for each of the predicate types in turn, analogous to the second block in Table 1. This is also the baseline we test against. Table 3 presents the results for each of the predicate types and different weighting models for individual predicates. As indicated, improvements are significant in the majority of cases. We show improvements over the results reported for the unstructured model in every category, illustrating that the additional level of normalisation can cover the structure of the entities. The ‘uniform’ weighting strategy performs best for all types with ‘popularity’ coming second. Therefore we omit the results for the remaining predicate weighting strategies in Table 4, where we show the effect of using multiple field types and combine them. We can not improve results here, which we attribute to a certain inability of the model to exploit the semantic information. This is somewhat surprising since we showed we can cover for individual predicate types and as such can contribute to better predicate modeling (as shown in Table 3).

6 Related Work

It has been shown that that over 40% of Web search queries target entities [24]. Following up on this trend, a range of commercial providers now support entity-oriented search, dealing with a broad range of entity types such as people, companies, services, or locations. This shows that the problem studied in this paper, searching entities in structured semantic data, “has direct relevance to the operation of Web search engines, which increasingly incorporate structured data in their search results pages" [5].

Research on entity retrieval in IR got a major boost with the introduction of the expert finding task at the TREC Enterprise track in 2005 [7]. The task requests a ranked list of experts returned for a given query topic. A separate Entity search track started at TREC in 2009 and defined the related entity finding task: return a ranked list of entities (of a specified type) that engage in a given relationship with a given source entity [2]. Common to both tasks is that entities are not directly represented as retrievable units; one of the challenges is to recognise entities in the document collection, then aggregate the textual information associated with them for the purpose of retrieval [1]. INEX has also featured an Entity Ranking track between 2007 and 2009 [9, 11]. There, entities are represented by their corresponding Wikipedia article. The availability of category information and the rich link structure of Wikipedia clearly distinguishes this task from plain document retrieval [12].

The ad-hoc entity retrieval task over RDF data we studied in this paper was proposed and formalised in [24]. This task bears some resemblance to keyword search in relational databases [22] and to XML retrieval [17]. The former has no direct relevance as methods developed for structured databases are not directly applicable to RDF data. As for the latter, Kim et al. [18] presented a probabilistic retrieval model for semi-structured data (PRM-S) that allows for a weighted mapping of query terms to (entity) attributes. Dalton and Huston [8] tested this model on the SemSearch 2010 data set and found that a key limitation of the PRM-S approach is that “it assumes a collection with a single or very few clearly defined entity types." Unlike the IMDB and Monster collections used in [18], the BTC-2009 corpus is very heterogeneous. Additionally, two specific problems were identified: (1) computing the query term to attribute mapping probabilities suffers from attribute sparsity, and (2) mapping probabilities are estimated for each query term independently. Finally, Ogilvie and Callan [21] considered modeling documents with a hierarchical structure for XML retrieval. Our Hierarchical Entity Model has been developed in a similar spirit, but hierarchy in [21] is defined by document structure (markup), while in our case it is organised around semantic relationships. Both works estimate language models on the component levels and incorporate evidence from multiple levels within the hierarchy, but the task addressed in [21] (XML component retrieval), and hence the actual models, are very different from ours.

7 Discussion and Conclusions

We have addressed the task of ad-hoc entity retrieval in the Web of Data: returning a ranked list of RDF resources that represent an entity described in the keyword query. The main research question we have been concerned with is the modeling of entities, described in the form of subject-predicate-object triples, for text-based retrieval. Prior work suggested two main directions: (1) collapsing all text associated with the entity into a single flat-text representation and then using standard IR models for document retrieval, and (2) grouping predicates together into a small number of categories, representing them as document fields, and applying fielded extensions of document retrieval methods. We formalised both strategies using generative language models and termed them *Unstructured Entity Model* (UEM) and *Structured Entity Model* (SEM). Experimental evaluation was performed using the 2010 and 2011 test sets of the Semantic Search Challenge evaluation campaign. The structured approach was shown to outperform a very strong baseline provided by the unstructured model.

Specifically, we grouped predicates based on their types into four categories: name, attributes, outgoing, and incoming relations. Out of these, name and attributes proved to be the most useful ones for our task. The combination of multiple predicate types failed in the unstructured case; incorporating relations did not bring in any improvements over using names or attributes alone. The reason for this behavior is that there is more textual content for relation type predicates (35.5 and 13.3 terms on average for in- and out-relations, respectively) than for name (3.95) or attributes (26.6), and this way the entity language model may lose the focus from the entity itself. The structured model represents predicate types as language models and considers their weighted linear combination. Taking all types with equal weights delivers very strong results and outperforms the existing state-of-the-art. Importantly, we showed that relations can contribute to overall performance under this approach.

While the above two models perform well empirically, they suffer from a severe limitation: they abandon a large part of the semantics behind the data. We propose a novel approach, referred to as *Hierarchical Entity Model* (HEM), that is capable of preserving the semantics associated with entities. It organises predicates into a two-level structure with predicate types on the top level and individual predicates on the bottom level. The weight of predicate types can be set similarly to the SEM model, while the importance of individual predicates can be estimated in an unsupervised way. Our experiments showed that modeling individual predicates of a given type is more effective than folding their contents into a flat representation. When multiple predicate types are combined for the entity, the HEM model delivers substantially higher results than the UEM baseline, but fails to outperform SEM. One issue for future investigation is to find out why the combination does not benefit from the improved component models. It may be the case that the entity model is now more semantically informed but the query representation and retrieval model are yet unable to exploit this fact. Future work will mainly be concerned with extending the current approach by segmenting the query and mapping its components to individual predicates within the hierarchical entity model.

References

[1] Balog, K., Azzopardi, L., de Rijke, M.: A language modeling framework for expert finding. Inf. Process. Manage. 45, 1–19 (2009)
[2] Balog, K., de Vries, A.P., Serdyukov, P., Thomas, P., Westerveld, T.: Overview of the TREC 2009 entity track. In: Proc. of the 18th Text Retrieval Conference, TREC 2009 (2010)
[3] Balog, K., Ciglan, M., Neumayer, R., Wei, W., Nørvåg, K.: NTNU at SemSearch 2011. In: Proc. of the 4th Intl. Semantic Search Workshop (2011)
[4] Blanco, R., Halpin, H., Herzig, D.M., Mika, P., Pound, J., Thompson, H.S., Duc, T.T.: Entity search evaluation over structured web data. In: Proc. of the 1st International Workshop on Entity-Oriented Search (EOS 2011), pp. 65–71 (2011)
[5] Blanco, R., Mika, P., Vigna, S.: Effective and efficient entity search. In: Proc. of the 9th Intl. Semantic Web Conf., pp. 83–97 (2011)
[6] Campinas, S., Delbru, R., Rakhmawati, N.A., Ceccarelli, D., Tummarello, G.: Sindice BM25F at SemSearch 2011. In: Proc. of the 4th Intl. Semantic Search Workshop (2011)
[7] Craswell, N., De Vries, A.P., Soboroff, I.: The TREC 2005 enterprise track. In: Proc. of the 14th Text Retrieval Conference (TREC 2005) (2005)
[8] Dalton, J., Huston, S.: Semantic entity retrieval using web queries over structured RDF data. In: Proc. of the 3rd Intl. Semantic Search Workshop (2010)
[9] de Vries, A.P., Vercoustre, A.-M., Thom, J.A., Craswell, N., Lalmas, M.: Overview of the INEX 2007 Entity Ranking Track. In: Fuhr, N., Kamps, J., Lalmas, M., Trotman, A. (eds.) INEX 2007. LNCS, vol. 4862, pp. 245–251. Springer, Heidelberg (2008)
[10] Delbru, R., Toupikov, N., Catasta, M., Tummarello, G.: A Node Indexing Scheme for Web Entity Retrieval. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010. LNCS, vol. 6089, pp. 240–256. Springer, Heidelberg (2010)
[11] Demartini, G., de Vries, A.P., Iofciu, T., Zhu, J.: Overview of the INEX 2008 Entity Ranking Track. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2008. LNCS, vol. 5631, pp. 243–252. Springer, Heidelberg (2009)
[12] Demartini, G., Firan, C.S., Iofciu, T., Krestel, R., Nejdl, W.: Why finding entities in Wikipedia is difficult, sometimes. Information Retrieval 13(5), 534–567 (2010)
[13] Demartini, G., Kärger, P., Papadakis, G., Fankhauser, P.: L3S Research Center at the SemSearch 2010 Evaluation for Entity Search Track. In: Proc. of the 3rd Intl. Semantic Search Workshop (2010b)
[14] Elbassuoni, S., Ramanath, M., Schenkel, R., Sydow, M., Weikum, G.: Language-model-based ranking for queries on RDF-graphs. In: Proc. of the 18th Intl. Conf. on Information and Knowledge Management, pp. 977–986 (2009)
[15] Halpin, H., Herzig, D.M., Mika, P., Blanco, R., Pound, J., Thompson, H.S., Tran, D.T.: Evaluating ad-hoc object retrieval. In: Proc. of the Intl. Workshop on Evaluation of Semantic Technologies (2010)
[16] Herzig, D.M., Tran, T.D.: Scoring model for entity search on RDF graphs. In: Proc. of the 3rd Intl. Semantic Search Workshop (2010)
[17] Kamps, J., Geva, S., Trotman, A., Woodley, A., Koolen, M.: Overview of the INEX 2008 Ad Hoc Track. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2008. LNCS, vol. 5631, pp. 1–28. Springer, Heidelberg (2009)
[18] Kim, J., Xue, X., Croft, W.B.: A Probabilistic Retrieval Model for Semistructured Data. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 228–239. Springer, Heidelberg (2009)
[19] Liu, X., Fang, H.: A study of entity search in semantic search workshop. In: Proc. of the 3rd Intl. Semantic Search Workshop (2010)

[20] Ogilvie, P., Callan, J.: Combining document representations for known-item search. In: Proc. of the 26th Annual Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pp. 143–150 (2003)
[21] Ogilvie, P., Callan, J.: Hierarchical Language Models for XML Component Retrieval. In: Fuhr, N., Lalmas, M., Malik, S., Szlávik, Z. (eds.) INEX 2004. LNCS, vol. 3493, pp. 224–237. Springer, Heidelberg (2005)
[22] Park, J., Lee, S.-G.: Keyword search in relational databases. Knowl. Inf. Syst. 26, 175–193 (2011) ISSN: 0219-1377
[23] Pérez-Agüera, J.R., Arroyo, J., Greenberg, J., Iglesias, J.P., Fresno, V.: Using BM25F for semantic search. In: Proc. of the 3rd Intl. Semantic Search Workshop, pp. 1–8 (2010)
[24] Pound, J., Mika, P., Zaragoza, H.: Ad-hoc object retrieval in the web of data. In: Proc. of the 19th Intl. Conf. on World Wide Web, pp. 771–780 (2010)
[25] Robertson, S., Zaragoza, H., Taylor, M.: Simple BM25 extension to multiple weighted fields. In: Proc. of the 13th Intl. Conf. on Information and Knowledge Management, pp. 42–49 (2004)
[26] Zhai, C.: Statistical language models for information retrieval a critical review. Foundations and Trends in Information Retrieval 2, 137–213 (2008)