# A HMM-based Approach to Question Answering against Linked Data

Cristina Giannone, Valentina Bellomaria, and Roberto Basili

Department of Enterprise Engineering, University of Rome Tor Vergata, Roma, Italy

**Abstract.** In this paper, we present a QA system enabling NL questions against Linked Data, designed and adopted by the Tor Vergata University AI group in the QALD-3 evaluation. The system integrates lexical semantic modeling and statistical inference within a complex architecture that decomposes the NL interpretation task into a cascade of three different stages: (1) The selection of key ontological information from the question (i.e. predicate, arguments and properties), (2) the location of such salient information in the ontology through the joint disambiguation of the different candidates and (3) the compilation of the final SPARQL query. This architecture characterizes a novel approach for the task and exploits a graphical model (i.e. an Hidden Markov Model) to select the proper ontological triples according to the graph nature of RDF. In particular, for each query an HMM model is produced whose Viterbi solution is the comprehensive joint disambiguation across the sentence elements. The combination of these approaches achieved interesting results in the QALD competition. The RTV is in fact within the group of participants performing slightly below the best system, but with smaller requirements and on significantly poorer input information.

## 1 Introduction

Language is the most powerful media for acquiring, communicating and sharing knowledge. It has been optimized through centuries of use, successes and failures. Although the Web of Data claims for machine readable standards, natural language is still the preferred query language for naive users, early adopters and even experts in some knowledge domains. Question answering is thus the crucial bottleneck for a truly and universal adoption of Open Linked Data as a knowledge sharing paradigm and practice.

In general, approaches for question answering range between rule-based (and strongly deductive) systems, whose expressivity is harmonic with the knowledge representation standards in the Web and whose precision is optimal, to shallow, basically lexicalist approaches very close to the bag-of-words practices in document retrieval processes. In the first family of systems, we could mention at least Swoogle [3] or Sindice [13], where entity search exploiting the Linked Data constraints are formulated and high level of precision are in general achieved. Shortcomings of these approaches come from the naive (user's) dictionaries that can be very different from the data dictionary. In [15] a system that produces a SPARQL template that directly mirrors the internal structure of the question and then instantiates the template, using statistical entity identification and predicate detection is proposed. It relies on a linguistic analysis and adopts DRT over parse trees through the extension of the Pythia system [16]. By applying deep linguistic analysis Pythia is demanding w.r.t. the lexical and grammatical knowledge needed to cope with complex questions in heterogeneous domains. A general approach to question answering over Linked Data is described in PowerAqua [8], that makes light assumptions on the ontology vocabulary or schema and emphasizes the combination of large data sets through filtering and ranking heuristics. The weaker linguistic component in Poweraqua makes the treatment of complex questions difficult.

An interesting vocabulary-independent approach is attempted in [5] which combines entity search and lexical similarity metrics to compute semantic relatedness and apply spreading activation onto RDF graphs. This work shares with the approach above presented a *lexicalist* perspective that rely on a strong model of lexical semantic information to solve most of the ambiguity and uncertainty problems arising in the interpretation of the question. A similar combination of statistical inference and logic-based representation is adopted in approaches focused on Semantic Parsing (e.g. [2]) where graphical models are used to converge towards the correct interpretation of a sentence in a predicate logic form, as well as in [12] where a HMM is employed in order to obtain the right RDF subgraph for a natural language query, here the observed data and the background knowledge is used for the HMM parameter estimation.

In line with the latter, in this paper we present a novel contribution to the above research line which combines symbolic reasoning over the semantic constraints on the underlying Open Data repository and statistical inference. This latter is useful to manage the ambiguity introduced by natural language within a complex process for the interpretation of the question, that integrates distributional semantic models of lexical information and probabilistic inference. In this way, we aim at solving at least two problems. The first one is the *localization and retrieval of ontological elements evoked by a question* without relying on strict hypothesis on the resource vocabulary. Second, we jointly solve the different ambiguities arising in the interpretation by integrating question grammatical structures and ontology information. The idea is to map the different inferences into a generative graphical model, i.e. an Hidden Markov Model of the question. It works as a bridge between the linguistic and the RDF structures, i.e the syntactic dependency graph of the question on the one side and the full paths in the RDF graph, on the other. The rest of the paper is organized as follow: Section 2 presents an overview of the systems architecture and introduces the HMM notions. Section 3.1 discusses in more detail the modeling of the input question through a Hidden Markov Model, and the resulting RDF sub-graph obtained. The process of mapping the HMM output into the SPARL Template is discussed in Section 3.4. Finally, in Section 4, a first analysis of the QALD-3 results are discussed with some final considerations.

## 2 Architectural Overview

The overall architecture of the RTV[1] system is shown in Figure 1 where the core components of the system are within the main box and interacts with auxiliary preexisting modules and resources. The treatment of a sentence as carried out by the core system

[1] RTV is the acronym of Roma Tor Vergata

can be summarized as follows. First an analysis of the syntactic (dependency) graph is carried out where proper nouns related to DBpedia resources are detected and classified in the so-called HMM initialization stage. In the HMM all the observations about key entities or classes are selected from the question, as they refer to ontological elements, i.e. mentions to entities, literals and relations (or properties). In the HMM modeling stage, the states, emissions and transitions of the Markov chain are defined. States corresponds to the RDF elements retrieved by a question fragment from DBpedia: these elements may correspond to resources, classes or relations. The HMM best sequence of states is computed in the decoding module, here a disambiguation process is imposed exploiting the combination of statistical and ontological constraints obtaining a state sequence which can be mapped into a RDF sub graph.

The process, triggered by a grammatical analysis of the input question, foresees three main components deal with the initialization, modeling and decoding of the HMM and a final stage to compile the resulting SPARQL query. A detailed description of such modules is given in Section 3.

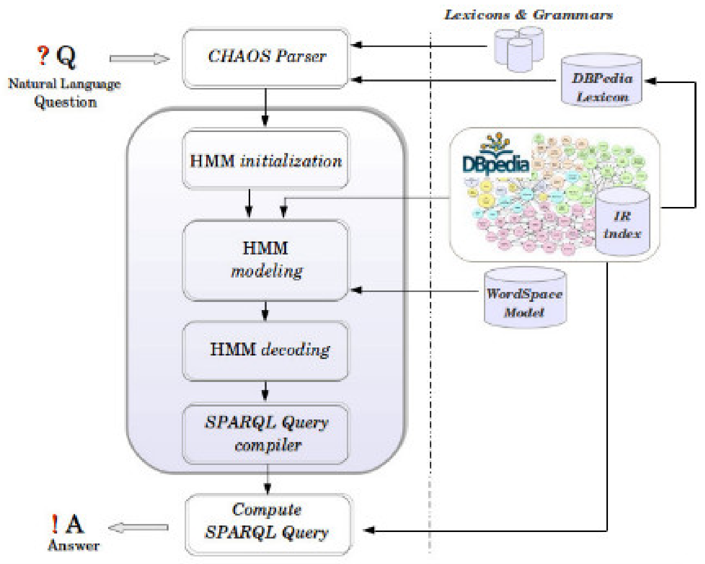

**Fig. 1.** The overall architecture of the RTV QA system

Besides a syntactic parser, i.e. Chaos [1] whose result is a chunk-based dependency graph, a number of external resources are made available:

– A wordspace in line with distributional semantic methods [11, 10, 9] is derived from the Wikipedia corpus. It provides lexical entries in form of real-valued vectors for a large dictionary including Named Entities as well as multiword expressions. Lex-

ical similarity metrics is thus made available between word pairs, modeled in the space as cosine similarity.
– Lucene[2] has been applied as a retrieval tool for DBpedia concepts. DBpedia resources are retrieved (and properly ranked) as pseudo-documents, whereas grammatically meaningful fragments of the question can used as queries. Notice how resources or fragments can be mapped into the above wordspace[3]. Their similarity can be thus applied to rank the retrieved candidates in a semantically meaningful manner;
– DBpedia labels are also exploited, as they provide an extensive catalogue of Proper Nouns made available to the Chaos parser [1]. Given an input question, the parser carried out the recognition of most resource names (as possibly ambiguous Named Entities) as well as of the syntactic dependencies they are involved in.

## 3 An HMM-based QA system over Linked Data

Our objective is the modeling the interpretation of the natural language query $q$ into the ontology domain $D$ through the following set of HMM random variables:

– The observation set $L = \{l_1, ..., l_m\}$, in our case the observation set is computed over the set of symbols of the sentence $q$,
– the set of States $O = \{o_1, ..., o_n\}$, that models the ontological resources $\in D$ candidate to be a suitable interpretation for the observation set $L$,
– the emission matrix $E : L \times O \rightarrow [0, 1]$ models the probability that an observation $l$ evokes the state $o$, i.e., the corresponding ontological resource,
– the transition matrix $T : O \times O \rightarrow [0, 1]$ models the transition probability from the state $o_i$ the the state $o_j$, these probability model the reachability of the state $o_j$ from the state $o_i$. They help explaining the semantics of relationships between hidden states, i.e. inferential steps that correspond to the transversal of RDF graphs along specific paths.

Both emissions and transition probabilities are estimated *on the fly*, i.e. against the incoming sentence, just before applying the traditional decoding algorithms (i.e. Viterbi) to the resulting HMM. The outcome of the decoding stage is a full RDF path, that corresponds to one interpretation of classes (such as `Person`), instances as well as relations. In the next subsections each step of the HMM modeling is discussed.

### 3.1 Initializing the HMM graph

The sentence segmentation for the generation of the observation set $L$ is driven by the syntactical structure of the sentence $q$. The HMM initialization process proceeds by selecting the root of the dependency graph and navigating all the syntactic relations, by thus enumerating the entire set of predicate expressions (e.g. *die* in the sentence

[2] `http://lucene.apache.org/core/`

[3] Short texts are usually expressed by the vector that is the linear combination of vectors corresponding to the involved words.

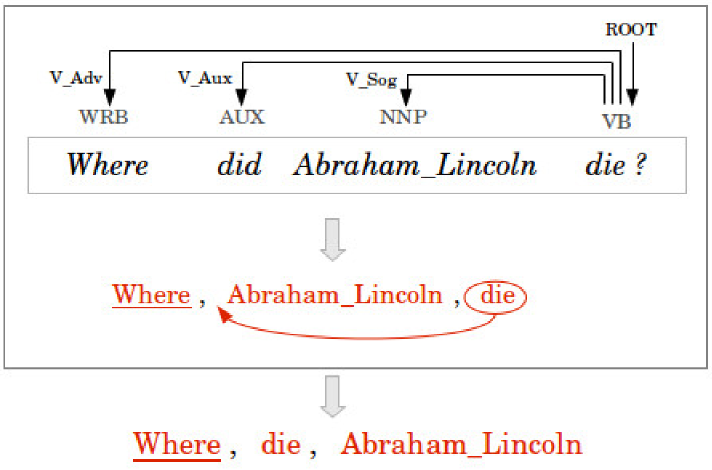
ROOT
V_Adv
V_Aux
V_Sog
WRB
AUX
NNP
VB
Where did Abraham_Lincoln die ?
Where , Abraham_Lincoln , die
Where , die , Abraham_Lincoln

**Fig. 2.** The analysis of the dependency graph for an example sentence: *Where did Abraham Lincoln die*?

in Figure 2) as well as their direct arguments (such as *Where* and *Abraham Lincoln*). The generated set of HMM observations is thus bipartite into two sets: nodes (that corresponds to nodes or literals in the RDF graph) and relations (i.e. links in the RDF graph), alternating in the Markov chain.

The resulting sets of nodes and links gives rise to one chain with the precedence assigned to the focus of the question (i.e. the place introduced by *Where*) used as initial node). The resulting observation chain for the sentence *Where did Abraham Lincoln die*? is shown at the bottom of Fig. 2 in which *Where* corresponds to the first node and the rest of the chain links together the other different nodes and relations. Notice how relations (e.g. *die*) alternate with elements but all of them align into a sequence. Moreover, proper nouns (e.g. *Abraham Lincoln*) are specifically treated as the Chaos parser lexicon has been enriched in the experiments with the catalogue derived from the values of the field `label` in DBpedia elements.

The analysis of the sentence ends when all grammatically relevant elements (basically nouns, verbs and adjective) give rise to an observation in the targeted Markov chain.

### 3.2 Modeling question semantics through states, transitions and emissions: the HMM modeling stage

During the HMM initialization, key elements in the sentence are detected and their associated ontological elements are located in the ontology. Key elements correspond to linguistic expressions defined as observations. In this way, a sequence corresponding to a Markov chain is generated. In the interpretation process targeted here, the HMM is designed as a generative model of the question. The model provides statistical evidence about the way *a question is generated as a request against RDF resources that form a* *graph*. In order to fully define the HMM, we still need to make the set of observations $L$ correspond to states $O$, and determine the corresponding transition and emission probabilities. The following modeling choices must be carried out:

– How to map linguistic expressions, such as proper nouns (e.g. *Where, Abraham Lincoln*) or predicate structures (e.g. *die*), to ontological elements such as DBpedia instances (e.g. possibly ambiguous Wikipedia pages) or relations such as `deathDate`
– How to model the notion of emissions $E$, that characterize the relationship between hidden states and observable linguistic structures
– How to model the notion of transitions $T$ that help explaining the semantics of relationships between hidden states, i.e. inferential steps that correspond to the transversal of RDF graphs along specific paths

In this work, we mapped DBpedia resources, classes as well as relations or properties into HMM states: they justify (as emissions) the observable linguistic structures, i.e. key entities of the question. While observations correspond to an open set of linguistic symbols $W$, by relying on a large scale semantic lexicon developed through corpus analysis, we will be able to map them into ontology element set $D$. Transitions corresponds to links between RDF elements and have a clear ontological status determined by DBpedia. Notice how probabilities equal to 0 can be used to constraint the semantics of a transition and model DBpedia links. On the other hand, more plausible interpretations receive higher probabilities, according to the context of the question.

**Estimating emissions probabilities through distributional semantics** An emission connects a linguistic expression $l$ to an ontological element $o$ and its probability in the HMM is

$$p(l|o)$$

The population of the state space and the estimation of $p(l|o)$ is defined through an information retrieval process. First, the linguistic expression $l$ related to a fragment of the dependency graph is generated. Ontological elements $o$ for different linguistic expressions $l$ are also different, and they must be precisely located in the ontology:

– when **instances** are involved (as proper nouns are detected in the question), first the query is executed against a search engine devoted to index the different DBpedia instances; the index is built over the long abstracts of the resource made available by DBpedia. The query corresponds to the expression $l$ itself (e.g. *Abraham Lincoln*) and a number of resources (in which $l$ is more or less explicitly mentioned) are retrieved. As linguistic expressions are represented in vectors (as in [10]), after retrieval, a further reranking stage is applied. Every returned candidate $o$ also consists in a linguistic expression and a vector $\underline{o}$, whose similarity with the vector $\underline{l}$ corresponding to $l$, is used as the final ranking score: candidates $o$ that are semantically more related to the entire question, i.e. their vector in the space is closer to the vector $\underline{l}$, are given higher priority. More in detail the following probability estimate is used $\hat{p}(l|o_i) = \frac{\alpha_i+\eta_1}{\sum_{j=1}^{k} \alpha_j+(k+1)\eta_1}$ where the factors $\alpha_i$ represent the similarity score between the linguistic expression $l$ and the ontological resource (i.e.

DBpedia entity) $o_i$, as it is measured in the word space: details of the estimation are given in the next subsection.

– when a common noun is involved (as it does not allow to retrieve any specific resource), a corresponding **class** is searched for. Classes $o$ in DBpedia are indexed beforehand: the expressions found in the `label` field of the concept $o$ correspond to a vector representation based on the description field of the corresponding DBpedia resource. If a nominal chunk is extracted from a question, ontological classes $o$ can be retrieved if their vector are more similar to $\underline{l}$ than a given threshold[4] in the space. For every noun $l$, multiple classes are retrieved in general corresponding to the DBpedia classes available closer in the vector space. The estimation of the $p(l|o)$ is again based on the similarity measure between $\underline{l}$ and $\underline{o}$. More in detail the following probability estimate is used $\hat{p}(l|o_i) = \frac{\beta_i+\eta_2}{\sum_{j=1}^{k}\beta_j+(k+1)\eta_2}$ where the $\beta_i$ factors are the cosine similarity estimates of the semantic similarity between the vector representation of the linguistic expression $\underline{l}$ and the vector related to the ontological object $\underline{o_i}$ and $\eta_2$ is a further smoothing factor that enables an empty value (state) as a literal (i.e. non related to any ontological notion) interpretation of $l$ . Notice how given an expression $l$ we retrieve in the wordspace the $k$ closer ontological classes $o_1, ..., o_k$, in order not to generalize too much: the probability estimates increases for more similar $o$'s and decreases according to the decrements of the cosine measure.

– **Relationship labels** are treated similarly to classes. DBpedia relations form a closed dictionary that can be indexed in the word space beforehand. Infact, their lexical labels, such as in the case of *spouse* for the `spouse` relation, are indexed through their word vectors. The relationship dictionary thus formed is used to retrieve the relationships involved in a question. First, all the ontological relations in the dictionary that are linked to someone of the entities evoked in the previous step (i.e. the key entities of the question already derived from DBpedia) are collected: this set is the *target relationship set*, $TRS$. They involve several relations or properties admissible as they linked with the key entities detected in the question. Moreover, every relationship is triggered in the question from a linguistic expression $l$ (such as *married*). Its DBpedia counterpart must be found in the $TRS$ set. First, $l$ is mapped to a vector $\underline{l}$ through its name (such as in the case of *die* that is represented by the vector of *die*) or through a more complex pattern[5]. When $\underline{l}$ and $TRS$ are given, the relationships $rel_i$ in the dictionary that are in $TRS$ and close enough to $\underline{l}$ are retrieved. They are individually mapped to different states corresponding to relations known in DBpedia, and ranked according to their descending semantic relatedness (*die* vs. *deathPlace*) in the word space. Notice that complex class labels (e.g. deathDate) can be also processed through a distributional semantic model: while co-occurrence vectors are obtained from the corpus for the individual words that compose the label, they can be linearly combined to represent complex com-

[4] A threshold of 0.8 on the cosine similarity is imposed in the settings.

[5] Notice that if a DBpedia relation corresponds to a complex linguistic expressions, i.e. `writtenBy`), the distributional analysis can be carried out for the entire pattern (i.e. *written by*) as it occurs in the corpus. In all cases thus gives rise to a unique vector for the entire pattern.

binations. Technically, the estimate for the emission probability for a generic $rel_i$ is: $\hat{p}(l|o_i) = \frac{\sigma_i}{\sum_{o_j \in TRS} \sigma_j}$ where $l$ is the linguistic expression for a relation, $o_j$ belongs to the target relationship set $TRS$ and $\sigma$ are cosine estimates of the lexical similarity between $l$ and $o_j$. No smoothing factor is here applied, so that poorly similar relationship in $TRS$ can be retrieved from $l$ and no external relation (or literal relational expression) is admitted for $l$.

– When no relationship can be found at the previous stage, the system must deal with **complex linguistic relational expressions**, denoted by $cl$. A further attempt is carried out to map these expressions to the relations and properties of individuals extending their linguistic processing. As the target ontological relation in this case is unknown (i.e. the set of DBpedia properties are out of the closed dictionary of relationships delivered with DBpedia [6], parsing is carried out over such an complex $cl$, in order to extract its linguistic head, denoted by $h_{cl}$. The vector $\underline{h_{cl}}$ corresponding to $h_{cl}$ is selected from the word space used as a representative of the multiword expression. Similarly to the previous step, it is used as a query to retrieve the involved ontological relationships from the target relationship set $TRS$. All relations or properties closer than a given threshold to $\underline{h}$ are added to the set of as potential states for the underlying observation (i.e. *cl*). Also in this case, the probability estimate is: $\hat{p}(cl|o_i) = \hat{p}(h_{cl}|o_i) = \frac{\sigma_i}{\sum_{o_j \in TRS} \sigma_j}$ where $cl$ is the complex linguistic expression, $o_j$ still belongs to the target relationship set $TRS$ and $\sigma_j$ are the cosine estimates of the lexical similarity between the expression head $h_{cl}$ and $o_j$. Again, no smoothing is here applied.

At the end of the above phase the entire lattice structure is built, where columns (i.e. states) are fullfilled and emission and transition probabilities are estimated. The vector space treatment of the probabilities is discussed in the next section.

**Probability Estimation in the HMM and distributional lexical similarity** As we defined in the previous section, a word space is derived for representing: (1) the lexicon for individual words as they are used in the DBpedia corpus; (2) the ontological classes and entities through the linguistic expressions corresponding to their labels; (3) the ontological relations and properties as they appear in the DBpedia closed dictionary; (4) unknown ontological relations, occurring in a specific target relation set, i.e. originating by the RDF triples regarding one or more key entities retrieved for a question, but missing from the DBpedia relationship catalogues. In all this case, an ontological element $o_i$ is made corresponding to a word vector, hereafter denoted by $\underline{o_i}$: for example `spouse` correspond to the vector $\underline{spouse}$. In this work we use the distributional behavior of the word as an hint on the relationship semantics, and this distinguishes our HMM based approach either from the work by [5] and [12].

We captured the distributional behavior of words in a word space similarly to [10]. The similarity function applied for the retrieval and ranking of different concepts $o_i$ is based upon a distributional analysis [9]: each concept $o_i$ corresponds to a set of textual contexts in which its corresponding lexical expression $l$ appears (*Distributional*

[6] for the set of DBpedia relations we refer to the file `mappingbased_properties_en.nt`

*Hypothesis*, [7]). In our work, contexts are still words (or features) $w$ that appear in a $n$-window around a target expression $l$. Such a space models a generic notion of semantic relatedness: two words close in the space are likely to be either in paradigmatic or syntagmatic relation, as discussed in [10]. Weights are given to co-occurrences between $l$ and its features $w$ through point-wise mutual information, as discussed in [14]. Finally, the resulting word-by-features context matrix $M$ is decomposed through Singular Value Decomposition (SVD) [6] into the product of three new matrices: $U$, $S$, and $V$ so that $S$ is diagonal and $M = USV^T$. $M$ is approximated by $M_k = U_k S_k V_k^T$ in which only the first $k$ columns of $U$ and $V$ are used, and only the first $k$ greatest singular values are considered. This approximation supplies a way to project a generic expression $l$ (being it or not corresponding to an ontological concept $o_i$) into the $k$-dimensional space using $W = U_k S_k^{1/2}$, where each row corresponds to the representation vectors $\boldsymbol{l}$. Therefore, given an expression $l$ and an ontological element $o_i$, the similarity function $\sigma$ is estimated as the **cosine similarity** between the corresponding projections $\boldsymbol{l}, \boldsymbol{o_i}$, i.e

$$\sigma(l, o_i) = \frac{\boldsymbol{l} \cdot \boldsymbol{o_i}}{\|\boldsymbol{l}\| \|\boldsymbol{o_i}\|}$$

The application of the above semantic relatedness measure to the transitions in an example sentence is shown in Fig. 3: only states that have a not null semantic similarity with an observation are given a numerical score, where scores are mapped in probabilities as in Sect. 3.2.

In QALD-2103, we applied the above method to the DBpedia corpus, made of all the textual descriptions associated with more than 3.77 million of entities in version 3.8. More specifically, to build the matrix $M$, POS tagging is first applied to build rows with pairs ⟨lemma, ::POS⟩, or $lemma :: POS$ in brief. The contexts around these items are the columns of $M$ and co-occurrences are computed within windows of size $[-5, +5]$ around the items $l$. This allows to better capture syntactic properties of individual $l$. The most frequent 20,000 lemmas (i.e. $l$) are selected along with their 20k contexts (i.e. features $w$). The entries of $M$ are the point-wise mutual information between them as computed in the entire corpus. The SVD reduction is then applied to M, with a dimensionality cut of $k = 250$.

## 3.3 HMM decoding for semantic disambiguation

In Fig. 3 the lattice corresponding to the sentence *Where did Abraham Lincoln die*? is shown. Emission probabilities are shown as nodes scores, while the different set of states for the three individual observations are reported. Transition probabilities are not made explicit. Their treatment is very basic in the current system. First we select transitions that are compatible with the range constraints of the DBpedia data: only transitions whose relations (or properties) are semantically compatible with the links in the DBpedia RDF graph receive a non zero probabilities. The graph is strongly simplified in this way although a fully connected HMM is obtained in most of the cases. Then the probability of the transitions are normalized through a maximum entropy perspective: if $n$ nodes $o_1, ..., o_n$ are reachable from a given state $\hat{o}_t$, as they satisfy all semantic constraints, then the transition probability $p(o_i|\hat{o}_t) = 1/n$ for each $i = 1, ..., n$. In this way, every consistent transition is equally likely for the system.

The decoding stage, often called Viterbi [4] decoding, corresponds to the selection of the most likely state sequence (i.e. RDF path) able to justify the key semantic elements of the question. It provides thus the selection of a suitable RDF path (i.e. a subgraph) from which a unique SPARQL query can be easily derived. Notice how likelihood here implies the satisfaction of most semantic constraints as well as the maximization of the overall semantic relatedness, and well captures the need of fuzzy reasoning over the semantic ambiguity introduced linguistically.

In view of increasing robustness, also a form of smoothing is applied as in [12]. One or more special states are artificially added to every observation related to a key entity. In order to account for possible mistakes in the retrieval process, we introduce a *default state* corresponding to a literal interpretation of a linguistic symbol: although it does not give rise to any constraint in the target SPARQL query, the default state is used to account for errors or lack of information in the source RDF data. In Figure 3, we can see two default states related to position one (*Where*) and three (*Abraham Lincoln*): for example, it can be seen as a way to model also the interpretation of *Abraham Lincoln* as a simple string and not an individual. These empty interpretations cumulate a small probability proportional to $\eta$ as introduced in 3.2: these values have been empirically defined and no specific estimation has been applied onto the available training data in QUALD.

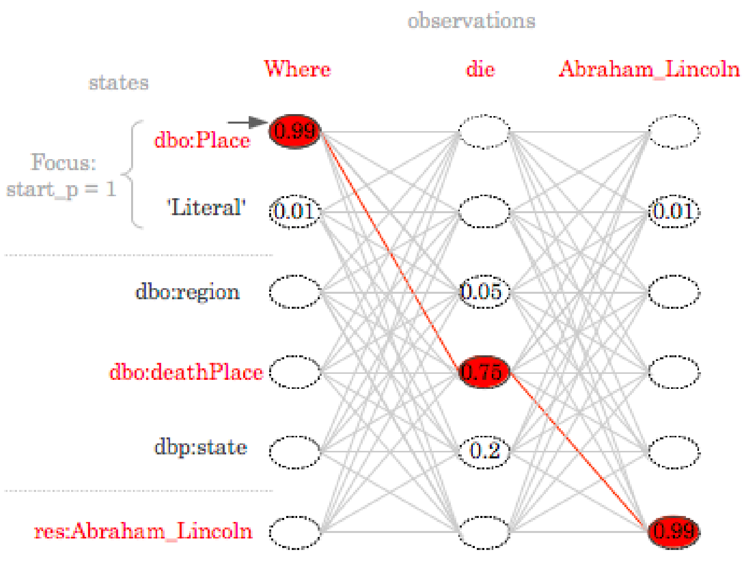

**Fig. 3.** The lattice for the example question: *Where did Abraham Lincoln die*?

### 3.4 SPARQL query compilation

In this module the resulting SPARQL query is generated starting from the RDF path obtained from Viterbi. Here, in order to obtain a consistent SPARQL query, the resulting RDF path is validated wrt the domain and range constraints imposed by the relational object in the RDF graph. For instance,for the HMM trellis in Figure 3 the resulting graph is `?x dbo:deathPlace res:Abraham_Lincoln` . While the single elements are a correct interpretation for the NL input question, the resulting graph has the argument constraint order reversed for `dbo:deathPlace`. Here the correct order for the graph is computed, and the SPARQL query is built selecting the right query form. For the QALD evaluation we exploited the input feature *answertype* to select the right SPARQL form.

## 4 Results and discussion

The RTV system has demonstrated a non-trivial performance, as reported in Table 1. Its third place is mainly due to its poor coverage of the set of sentences with a good balance between precision an recall. Given that, except for the named entity dictionary made available to the parser, the system does not rely on any resource specific dictionary, technique or heuristics this result is very promising. The method is fully portable as language processing and a general parsing technology, i.e. Chaos [1], is adopted. Moreover, fully general representations, methods and assumptions are employed. The core idea to model the disambiguation task as a side effect of large scale indexing of the reference ontology combined with a generative model of semantics is convincing and effective. The probabilistic approach, i.e. HMM, has been applied straightforwardly, and the estimates of its parameters have been fully automated. No tuning of the different model and system parameters has been carried out on the provided training set in QUALD, mostly for lack of time. The obtained results were really the very early outcomes also for the Tor Vergata team.

**Table 1.** The QALD-3 results for the RTV system over the DBpedia test set.

| | Total | Processed | Right | Partially | Recall | Precision | F-mesure |
|---|---|---|---|---|---|---|---|
| squall2sparql | 99 | 96 | 77 | 13 | 0.85 | 0.89 | 0.87 |
| CASIA | 99 | 52 | 29 | 8 | 0.36 | 0.35 | 0.36 |
| Scalewelis | 99 | 70 | 1 | 38 | 0.33 | 0.33 | 0.33 |
| **RTV** | **99** | **55** | **30** | **4** | **0.34** | **0.32** | **0.33** |
| Intui2 | 99 | 99 | 28 | 4 | 0.31 | 0.31 | 0.31 |
| SWIP | 99 | 21 | 14 | 2 | 0.15 | 0.16 | 0.16 |

The adopted model appears very promising especially for its portability across other domains as well as other languages. On the one side emission probabilities model the plausibility of interpretations (i.e. resources, relations or properties) as semantically related to observed syntactic structures (i.e. proper nouns but also verbs or nominal compounds). On the other side transition probabilities model the semantic relatedness across individual structures retrieved through distributional lexical semantic metrics. Finally, the Viterbi decoding guarantees a straight joint disambiguation process that acts on the global level of the sentence, and robustly accounts for individual interpretations interacting against the underlying syntactic structures.

A specific error analysis aiming at assessing the contribution of the individual stages in the processing chain has been carried out. In about 70 partially (or mistakenly) processed sentences, we observed that missing relations in the HMM is the major source of error (about 50%). In these cases the system was unable to lexically translate the underlying relation (e.g. `admittancedate` ): these were made available by the involved resources but it was not possible to properly locate it in the question (basically, unresolved anaphoric references or implicit arguments). An example is the case of a prepositional phrase such as "*(... book) by Kerouac (...)*" that elliptically refers to the predicate "*writing a book*". Given the absence of an explicit lexical reference to the predicate "*writing*", no retrieval of the proper relation `writtenBy` could be obtained, so that the HMM was inconsistently generated.

In a lower percentage of errors (below 20%), the mistake is generated by the Viterbi decoding phase, that fails to properly disambiguate the entities (i.e. select the proper state sequence) even if a well formed and consistent trellis was available: the output path produced by Viterbi was including wrong resources, thus preventing to compile the correct SPARQL query. In a small number of cases, the output of Viterbi is not properly connected to any resource (as the smoothing foreseen for a literal interpretation of a symbol in the sentence is selected for too many observations), and no semantic constraint was available to build the SPARQL query. These queries were simply rejected.

In future work, we will optimize parameters (e.g. the acceptance threshold imposed to the probability attached to the "best" Viterbi path) as these have not been studied in enough detail. Moreover, the current initialization of the HMM, basically dependent on the dependency graph, makes use of no semantics, e.g. no access to the DBpedia resources is done during the HMM initialization stage. We hypothesize here large improvement for methods that directly exploit ontological resources to initialize the HMM: this would allow to improve the treatment of relations as well as to early prune inconsistent interpretations. In the QUALD meeting, while we assume to bring a larger experimental evidence, and a quantitative discussion about the main weaknesses mentioned above, we will report of the system application on RDF data different from the QUALD data, in order to better generalize the system assessment.

## References

1. Basili, R., Zanzotto, F.M.: Parsing engineering and empirical robustness. Nat. Lang. Eng. 8(3), 97–120 (Jun 2002)
2. Chen, D.L., Mooney, R.J.: Learning to interpret natural language navigation instructions from observations pp. 859–865 (August 2011), `http://www.cs.utexas.edu/users/ai-lab/?chen:aaai11`

3. Finin, T., Peng, Y., Scott, R., Joel, C., Joshi, S.A., Reddivari, P., Pan, R., Doshi, V., Ding, L.: Swoogle: A search and metadata engine for the semantic web. In: In Proceedings of the Thirteenth ACM Conference on Information and Knowledge Management. pp. 652–659. ACM Press (2004)
4. Forney, G.D.: The viterbi algorithm. Proc. of the IEEE 61, 268 – 278 (March 1973)
5. Freitas, A., Oliveira, J.a.G., O'Riain, S., Curry, E., Da Silva, J.a.C.P.: Querying linked data using semantic relatedness: a vocabulary independent approach. In: Proceedings of the 16th international conference on Natural language processing and information systems. pp. 40–51. NLDB'11, Springer-Verlag, Berlin, Heidelberg (2011), `http://dl.acm.org/citation.cfm?id=2026011.2026017`
6. Golub, G., Kahan, W.: Calculating the singular values and pseudo-inverse of a matrix. Journal of the Society for Industrial and Applied Mathematics: Series B, Numerical Analysis 2(2), pp. 205–224 (1965), `http://www.jstor.org/stable/2949777`
7. Harris, Z.: Distributional structure. In: Katz, J.J., Fodor, J.A. (eds.) The Philosophy of Linguistics. Oxford University Press (1964)
8. Lopez, V., Nikolov, A., Sabou, M., Uren, V.S., Motta, E., d'Aquin, M.: Scaling up question-answering to linked data. In: EKAW. pp. 193–210 (2010)
9. Pado, S., Lapata, M.: Dependency-based construction of semantic space models. Computational Linguistics 33(2) (2007)
10. Sahlgren, M.: The Word-Space Model. Ph.D. thesis, Stockholm University (2006)
11. Schütze, H.: Word space. In: Hanson, S.J., Cowan, J.D., Giles, C.L. (eds.) NIPS 5, pp. 895–902. Morgan Kaufmann Publishers, San Mateo CA (1993)
12. Shekarpour, S., Ngomo, A.C.N., Auer, S.: Keyword-driven resource disambiguation over rdf knowledge bases. In: Takeda, H., Qu, Y., Mizoguchi, R., Kitamura, Y. (eds.) JIST. Lecture Notes in Computer Science, vol. 7774, pp. 159–174. Springer (2012)
13. Tummarello, G., Delbru, R., Oren, E.: Sindice.com: Weaving the open linked data. In: In Proceedings of the International Semantic Web Conference (ISWC (2007)
14. Turney, P.D., Pantel, P.: From frequency to meaning: Vector space models of semantics. Journal of artificial intelligence research 37, 141 (2010), `doi:10.1613/jair.2934`
15. Unger, C., Bühmann, L., Lehmann, J., Ngomo, A.C.N., Gerber, D., Cimiano, P.: Template-based question answering over rdf data. In: WWW. pp. 639–648 (2012)
16. Unger, C., Cimiano, P.: Pythia: Compositional Meaning Construction for Ontology-Based Question Answering on the Semantic Web. In: MuÃoz, R., Montoyo, A.A., MÃ©tais, E. (eds.) Natural Language Processing and Information Systems, Lecture Notes in Computer Science, vol. 6716, pp. 153–160. Springer Berlin / Heidelberg (2011)