ITEA Office
High Tech Campus 69 - 3
5656 AG Eindhoven
The Netherlands

T + 31 88 003 6136
E info@itea3.org
W www.itea3.org

ITEA 3 is a EUREKA strategic ICT cluster programme

D2.2.1 Overview and Comparison of Existing Semantic Parsers

ModelWriter

Text & Model-Synchronized Document Engineering Platform

Work Package: WP2
Task: T2.2 – Overview and Comparison of Existing Semantic Parsers

Document History

Version	Author(s)	Date	Remarks
1.0.0	Claire Gardent, Mariem Mahfoudh	02-Feb-2016	Initial Release

D.2.2.1 Overview and comparison of existing deep semantic parsers

Claire Gardent and Mariem Mahfoudh
CNRS/LORIA, Vandoeuvre-les-Nancy, France

Contents

1 Introduction

Given a sentence in natural language (English, French, etc.), semantic parsing aims to produce a formal representation of its meaning [KPT+04].

These meaning representations may be used for reasoning and inference e.g., to query a knowledge-base or to determine whether two texts are in an entailment relation [FZE14, HG01, BCFL13, YHM14, FZE13, UC11, BCW14, BG09, BG10] Or they may be used as an intermediate representations for further processing. For instance, they may be used to construct a text summary [LFT+15, WLZD08, LGMF04].

Depending on the target application, meaning representations will be more or less language independent. At one end of the spectrum, meaning representations are based on words and the signature of the meaning representation language is in effect the set of words occurring in the natural language being parsed. This is the case for instance, for traditional semantic parsers where a grammar is used to build a compositional semantics of the input sentence based on the words occurring in that sentence and on it phrase structure tree [CCB07, BB05, CCMO99, DCM+14, Del90]. At the other end of the spectrum, meaning representations encode information about the real world (grounded learning, [GR04, Poo13]) or about a formal model of that world i.e., an ontology [FZE14] .

Semantic parsing is used in a large range of fields such as: question answering from an existing knowledge base or database [BL14, KCAZ13, BCW14], machine translation [WM06, Liu13], robots control and communication [CFH+03], ontology building and enrichment [CV05, BNT08, CSP+08, SAGC+10, FMPT10, GK14], etc.

In the ModelWriter project, the principal aim is to synchronise text and formal models (UML, EMF, ontologies, etc.). In particular, Work Package 2 (WP2) target the conception and implementation of a reversible process such that text can be automatically mapped to formal models (semantic parsing) and models can be mapped to text (natural language generation [RDF00, Gar14, PGF14]).

Among the applications, the ModelWriter project aims to evaluate the reversible tool on the data of the Airbus company (one of the project partner). We consider its technical documents as textual data (writer part) and its ontologies (which represent components and rules necessary for the functioning of the Airbus planes) as models. We target also the use of the semantic parsing result for the enrichment of the Airbus ontologies.

An overview of the natural language generation approaches has been presented in a previous deliverable [Gar15]. In this report, we are interested in the semantic parsing process. We survey the most important approaches and tools proposed in the literature which address this issue and we focus in particular on the application of semantic parsing on the ontology domain (especially on ontologies building and enrichment).

Section 2 briefly reviews the main application domains of semantic parsing. Section 3 focuses on semantic parsing approaches which stem from the Natural Language Processing community. Section 4 introduces some approaches in the literature which handle the use of semantic parsing for ontology building and enrichment. Section 5 concludes with pointers for tools that could be relevant for the ModelWriter project.

2 Application Domains

As presented in the introduction, the semantic parsing is widely used in many domains. In this section, we introduce two application domains which are question answering and ontology building.

2.1 Question answering

Question answering is a field of computer science and natural language processing which "allow a user to ask a question in everyday language and receive an answer quickly and succinctly, with sufficient context to validate the answer" [HG01]. To understand the meaning of the question and interrogate the database or the knowledge base which store the information, the question answering system need a semantic parsing system. The parser takes the question and should correctly converts its interpretation into the exact database (or knowledge base) query (Figure 2.1). It should, thus, found the correctly formal logic which could have an answer. Several semantic parsing approaches are proposed in the literature for question answering domains which can be classified : 1) task specified grammar [BCFL13] ; 2) strongly typed CCG Grammars [RLS14, CKZ15]; 3) neural network without requiring any grammar [YHM14]. More details are presented in Section 3.

Question : Who did Stephen Hawking marry in 1965?
⇓
Semantic parsing
⇓
Logical Form : Person? ⊓ Marriage.(Spouse.StephenHawking? StartDate.1965)
⇓
Execute logical form
⇓
Response: Jane Wilde

Figure 1: Question answering example.

2.2 Ontology building and evolution

Ontologies are a formal and explicit knowledge representation. They represent a given domain by their concepts and axioms while creating a consensus between a users community [Gru93]. The semantic parsing could be involved in two main activities in the life cycle of an ontology: 1) the step of the ontology building and 2) the ontology evolution activity.

Several strategies have been proposed in the literature to build ontologies: 1) building ontologies from databases [CGY07]; 2) building ontologies by merging a set of ontologies [UK95]; 3) building ontologies from zero [GPFV96] and also 4) building ontologies from texts [MS00, ZN08]. It is in the strategy of building ontologies from texts, that the semantic parsing is often required. The principle could be summarize on the following steps (Figure 2):

1. *Extract relevant terms from texts.* Relevant terms identification is often done using extractor tools such as: TermExtractor [VS07], YaTeA [AH06], Termostat [1] [Dro03], etc.

2. *Identify the synonym terms.* Synonyms terms can be automatically identified based on linguistic ontologies (WordNet, Wolf [2], EuroWordnet[3]) or dictionaries.

3. *Identify the concepts and their hierarchy.* To automatically identify the hierarchical relations, several tools and approaches can be used: Hearst patrons [Hea92], TerminoWeb [BA06], TaxoLearn [DVF12], etc.

4. *Identify the relations between the concepts.* The relations between concepts are often extracted using the machine learning applied on knowledge base (such as Freebase, Wikipedia).

5. *Define axioms.* The last steps of ontology building consists to add axioms and rules.

Besides ontology building, the semantic parsing could be also used in other activities of the life cycle ontology as in the evolution ontology process. It is involved to automatically identify the new requirements of change and thus defining the set of the ontology changes (add new classes or relations add new roles or properties, add new axioms and assertions, remove axiom, rename class, etc. [MFTH15]) from a text corpora (more details in Section 4.2).

[1] olst.ling.umontreal.ca/ drouinp/tra2450
[2] gforge.inria.fr/projects/wolf
[3] www.illc.uva.nl/EuroWordNet

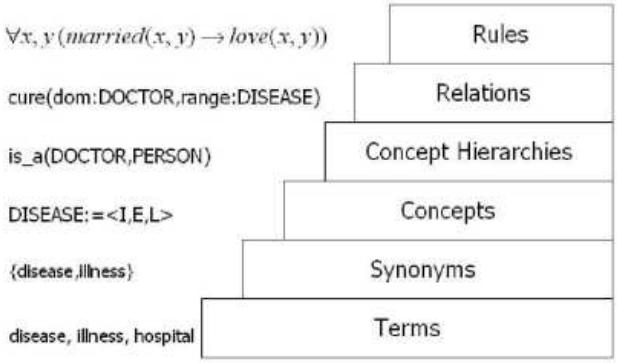

Figure 2: ≪Layer cake≫ strategy to build ontology from texts [BCM05].

3 NLP approaches to Semantic Parsing

NLP approaches to semantic parsing can be divided into three main strands depending on their output semantic representations and on their application.

Logical approaches, derived from earlier work in computational linguistics, seek to map NL sentences to logical formulae based on a strictly compositional interpretation of full semantic parses [BB05, CCB07, MM07]. Often based on higher-order lambda calculus, these approaches include both symbolic [CCB07] and [MM07] statistical approaches to determine the mapping between words and lambda terms.

Another, more recent trend, consists in developing semantic parsers which produce meaning representations that will be tested in some real world applications such as controlling a robot or querying a database [GM09, WM07]. These approaches are trained using data-to-text corpora and evaluated extrinsincally, based on how well they help in realising the target task (e.g., does the meaning representation produced succeed in retrieving the correct answer from the data-base ? Does it provide the correct instruction for the robot ?).

Finally, much recent work has focused on learning domain-independent semantic parsers with application to question answering against both large and small knowledge bases [WBL15, BCW14, BGWB12, KZGS10, KCAZ13]. For these approaches, the key challenge is to be robust to the high variability found in natural language and the many ways of expressing knowledge in a knowledge base.

3.1 Logical Approaches

Many of the main grammatical frameworks used in computational linguistics were extended to support semantic construction (i.e., the computation of a

meaning representation from syntax and word meanings). The HPSG ERG grammar for English was extended to output minimal recursive structures as semantic representations for sentences [CF00]. The LFG (Lexical Functional Grammar) grammars to output lambda terms [Dal99]. Clark and Curran's CCG (Combinatory Categorial Grammar) based statistical parser was linked to a semantic construction module allowing for the derivation of Discourse Representation Structures [BCS+04]. And [GK03] proposes a unification based approach to semantic construction using a Feature-Based Lexicalised Tree Adjoining Grammar (FB-LTAG).

In these approaches, semantic construction can be performed either during or after derivation of a sentence syntactic structure. In the first approach, syntactic structure and semantic representations are built simultaneously. This is the approach sketched by Montague and adopted e.g., in the HPSG ERG and in synchronous TAG [NS06]. In the second approach, semantic construction proceeds from the syntactic structure of a complete sentence, from a lexicon associating each word with a semantic representation and from a set of semantic rules specifying how syntactic combinations relate to semantic composition. This is the approach adopted for instance, in the LFG glue semantic framework, in the CCG approach and in the approaches to TAG-based semantic construction that are based on the TAG derivation tree.

In all of these approaches however, semantic construction – i.e. the derivation of the semantic representation associated with a sentence – is based on Montague's *compositionality principle* according to which

> the meaning of a sentence is a function of the meaning of its pars

In what follows, we illustrate this process using the FB-LTAG approach described in [GK03].

Tree Adjoining Grammars (TAG) TAG is a tree rewriting system [JS97]. A TAG is composed of (i) two tree sets (a set of initial trees and a set of auxiliary trees) and (ii) two rewriting operations (substitution and adjunction). Furthermore, in a Lexicalised TAG, each tree has at least one leaf which is a terminal.

Initial trees are trees where leaf-nodes are labelled either by a terminal symbol or by a non-terminal symbol marked for substitution ($\downarrow$). Auxiliary trees are trees where a leaf-node has the same label as the root node and is marked for adjunction ($\star$). This leaf-node is called a *foot* node.

Further, substitution corresponds to the insertion of an *elementary* tree t_1 into a tree t_2 at a frontier node having the same label as the root node of t_1. Adjunction corresponds to the insertion of an *auxiliary* tree t_1 into a tree t_2 at an inner node having the same label as the root and foot nodes of t_1.

In a Feature-Based TAG, the nodes of the trees are labelled with two feature structures called *top* and *bot*. Derivation leads to unification on these nodes as follows. Given a substitution, the top feature structures of the merged nodes are unified. Given an adjunction, (i) the top feature structure of the inner node receiving the adjunction and of the root node of the inserted tree are unified, and (ii) the bot feature structures of the inner node receiving the adjunction and of the foot node of the inserted tree are unified. At the end of a derivation, the *top* and *bot* feature structures of each node in a derived tree are unified.

Semantics ($\mathbf{L}_U$). The semantic representation language used in [GK03] is a unification-based extension of the PLU language [Bos95]. L_U is defined as follows. Let H be a set of *hole* constants, L_c the set of *label* constants, and L_v the set of *label* variables. Let I_c (resp. I_v) be the set of individual constants (resp. variables), let R be a set of n-ary relations over $I_c \cup I_v \cup H$, and let $\geq$ be a relation over $H \cup L_c$ called the *scope-over* relation. Given $l \in L_c \cup L_v$, $h \in H$, $i_1, \ldots, i_n \in I_v \cup I_c \cup H$, and $R^n \in R$, we have:

1. $l : R^n(i_1, \ldots, i_n)$ is a L_U formula.
2. $h \geq l$ is a L_U formula.
3. ϕ, ψ is L_U formula iff both ϕ and ψ are L_U formulas.
4. Nothing else is a L_U formula.

In short, L_U is a flat (i.e., non recursive) version of first-order predicate logic in which scope may be underspecified and variables can be unification variables[4].

Semantic Construction [GK03] propose a hybrid approach to semantic construction where (i) semantic construction proceeds after derivation and (ii) the semantic lexicon is extracted from a TAG which simultaneously specifies syntax and semantics. In this approach [GK03], the TAG used integrates syntactic and semantic information as follows. Each elementary tree is associated with a formula of L_U representing its meaning. Importantly, the meaning representations of semantic functors include unification variables that are shared with specific feature values occurring in the associated elementary trees. For instance in figure 3, the variables x and y appear both in the semantic representation associated with the tree for *aime* (love) and in the tree itself.

Given such a TAG, the semantics of a tree t derived from combining the elementary trees $t_1, \ldots, t_n$ is the union of the semantics of $t_1, \ldots, t_n$ modulo the unifications that results from deriving that tree. For instance, given the sentence *Jean aime vraiment Marie* (*John really loves Mary*) whose TAG derivation is given in figure 3, the union of the semantics of the elementary

[4]For mode details on L_U, see [GK03].

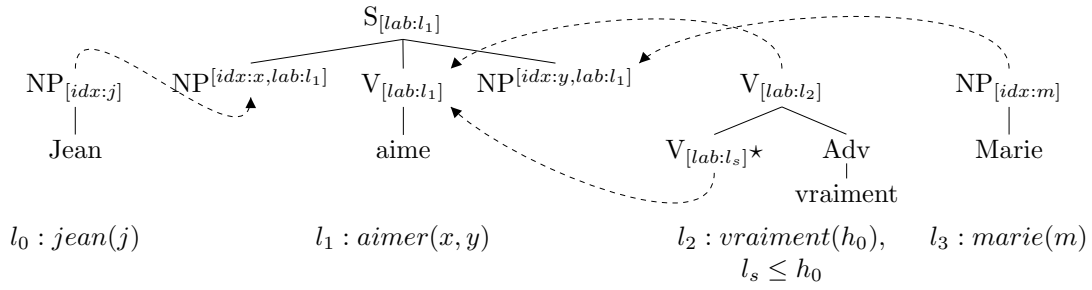

Figure 3: Derivation of "Jean aime vraiment Marie"

trees used to derived the sentence tree is:

$$l_0 : jean(j),\ l_1 : aime(x, y),\ l_2 : vraiment(h_0),$$
$$l_s \leq h_0,\ l_3 : marie(m)$$

The unifications imposed by the derivations are:

$$\{x \rightarrow j, y \rightarrow m, l_s \rightarrow l_1\}$$

Hence the final semantics of the sentence *Jean aime vraiment Marie* is:

$$l_0 : jean(j),\ l_1 : aime(j, m),\ l_2 : vraiment(h_0),$$
$$l_1 \leq h_0,\ l_3 : marie(m)$$

3.2 Domain Specific Approaches

The methods described in the previous section rely on prior knowledge of natural language syntax hence requiring extensive human efforts when porting to a new domain or language. To address this shortcoming, several algorithms have been proposed which learn a semantic parser from a set of natural language sentences and their meaning representation (MR)

[WM06] describes a semantic parser called WASP (Word Alignment Based Semantic Parsing) which uses a statistical word alignment model to acquire a bilingual lexicon mapping NL substrings to their translation in the target MR language. Complete MRs are then formed by combining these NL substrings and their translations using a Synchronous Context Free Grammar (S-CFG). Figure 4 shows an example S-CFG for the CLANG (soccer) domain, Figure 5 an example derivation and Figure 6 an example parse tree.

A maximum entropy model is learned for the semantic parser which defines a conditional probability distribution over derviations given the observed NL string:

$$P_\lambda(d|e) = \frac{1}{Z_\lambda(e)} exp \sum_i \lambda_i f_i(d)$$

$$
\begin{aligned}
\textsc{Rule} &\rightarrow \langle \textit{if}\ \textsc{Condition}_{[1]}\ ,\ \textsc{Directive}_{[2]}\ .\ , \\
&\quad (\textsc{Condition}_{[1]}\ \textsc{Directive}_{[2]})\rangle \\
\textsc{Condition} &\rightarrow \langle \textsc{Team}_{[1]}\ \textit{player}\ \textsc{Unum}_{[2]}\ \textit{has the ball}\ , \\
&\quad (\texttt{bowner}\ \textsc{Team}_{[1]}\ \{\textsc{Unum}_{[2]}\})\rangle \\
\textsc{Team} &\rightarrow \langle \textit{our}\ ,\ \texttt{our}\rangle \\
\textsc{Unum} &\rightarrow \langle \textit{4}\ ,\ \texttt{4}\rangle
\end{aligned}
$$

Figure 4: Example Synchronous Context Free Grammar

$$
\begin{aligned}
&\langle \textsc{Rule}_{[1]}, \textsc{Rule}_{[1]}\rangle \\
\Rightarrow\ &\langle \textit{if}\ \textsc{Condition}_{[1]}, \textsc{Directive}_{[2]}.\ , \\
&(\textsc{Condition}_{[1]}\ \textsc{Directive}_{[2]})\rangle \\
\Rightarrow\ &\langle \textit{if}\ \textsc{Team}_{[1]}\ \textit{player}\ \textsc{Unum}_{[2]}\ \textit{has the ball,}\ \textsc{Dir}_{[3]}.\ , \\
&((\texttt{bowner}\ \textsc{Team}_{[1]}\ \{\textsc{Unum}_{[2]}\})\ \textsc{Dir}_{[3]})\rangle \\
\Rightarrow\ &\langle \textit{if our player}\ \textsc{Unum}_{[1]}\ \textit{has the ball,}\ \textsc{Dir}_{[2]}.\ , \\
&((\texttt{bowner our}\ \{\textsc{Unum}_{[1]}\})\ \textsc{Dir}_{[2]})\rangle \\
\Rightarrow\ &\langle \textit{if our player 4 has the ball,}\ \textsc{Directive}_{[1]}.\ , \\
&((\texttt{bowner our}\ \{\texttt{4}\})\ \textsc{Directive}_{[1]})\rangle \\
\Rightarrow\ &\ldots \\
\Rightarrow\ &\langle \textit{if our player 4 has the ball, then our player 6} \\
&\textit{should stay in the left side of our half.}\ , \\
&((\texttt{bowner our}\ \{\texttt{4}\}) \\
&(\texttt{do our}\ \{\texttt{6}\}\ (\texttt{pos}\ (\texttt{left}\ (\texttt{half our})))))\rangle
\end{aligned}
$$

Figure 5: Example S-CFG Derivation

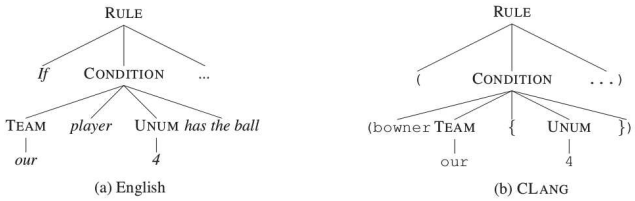

Figure 6: Example Derivation produced by a Synchronous Context Free Grammar

with f_i a feature function, d a derivation, e a string and $Z_\lambda(e)$ a normalising factor.

Similar approaches are described in [LNLZ08, KM06, GM05]. [LNLZ08] represent meaning representations and natural language within a single tree and apply tree walking algorithms to extract them from the data. They then use a custom training procedure for searching over the potential MR transformations. [KM06] use string classifiers to label substrings of the input string with entities from the meaning representation. To focus search,

they impose an ordering constraint based on the structure of the MR tree, which they relax by allowing the re-ordering of sibling nodes. The meaning representation is then extracted from the permuted tree by identifying the most likely input tree given a particular inpust string. Finally, [GM05] takes syntactic parses rather than NL strings and attempts to trans- late them into MR expressions.

3.3 Domain Independent Approaches

The semantic parsers discussed in the previous section have two limitations. They require annotated logical forms for supervision and they operate in limited domain with a small number of logical predicates. Recent research on semantic parsing aims to lift both these limitations and to allow for question answering against large scale, open domain knowledge bases such as Freebase. Three main types of approaches can be distinguished: two-step approaches with traditional semantic parsing and on the fly ontology matching, paraphrase-based approaches and approaches based on vectorial representations of natural language questions and of KB answers.

3.3.1 Two-step approaches

In the two-step approach, a domain independent intermediate logical form is first produced by a traditional semantic parser. Ontology matching is then performed to produce the target KB logical form.

In [KCAZ13], the input question is first parsed using a probabilistic Combinatory Categorial Grammar (CCG,[Ste00]) to produce a logical form meaning representation whose logical constants are not tied to any specific ontology. In particular, the CCG lexicon used is open domain, using no symbols from the ontology and can therefore be reused in every domain.

The ontology matching step then maps this intermediate representation to a logical form that uses constants of the KB.

For example, in Figure 7, l_o denotes the CCG logical form paired with the sentence x, y is the domain dependent logical form produced by the second, ontology matching, step and a is the answer retrieved from the KB.

x :	How many people visit the public library of New York annually
l_0 :	$\lambda x.eq(x, count(\lambda y.people(y) \wedge \exists e.visit(y, \iota z.public(z) \wedge library(z) \wedge of(z, new\ york), e) \wedge annually(e)))$
y :	λx.library.public_library_system.annual_visits(x, new_york_public_library)
a :	13,554,002
x :	What works did Mozart dedicate to Joseph Haydn
l_0 :	$\lambda x.works(x) \wedge \exists e.dedicate(mozart, x, e) \wedge to(haydn, e)))$
y :	λx.dedicated_work(x) $\wedge$ $\exists$e.dedicated_by(mozart, e) $\wedge$ dedication(x, e) $\wedge$ dedicated_to(haydn, e)))
a :	{ String Quartet No. 19, Haydn Quartets, String Quartet No. 16, String Quartet No. 18, String Quartet No. 17 }

Figure 7: Examples of sentences x, domain independent logical forms l_0, ontology specific logical forms y and KB answers a

To build the domain dependent logical form, all constants in the domain

independent LF are replaced with constants of the KN using the transformation rules shown in Figure 8. Figure 9 shows some example transformations.

Operator	Definition and Conditions	Example
a. Collapse Literal to Constant	$P(a_1,\ldots,a_n) \mapsto c$ s.t. $\texttt{type}(P(a_1,\ldots,a_n)) = \texttt{type}(c)$ $\texttt{type}(c) \in \{e,i\}$ $\texttt{freev}(P(a_1,\ldots,a_n)) = \emptyset$	$\iota z.Public(z) \wedge Library(z) \wedge Of(z,NewYork)) \mapsto PublicLibraryOfNewYork$ Input and output have type e. e is allowed in $\mathcal{O}$. Input contains no free variables.
b. Collapse Literal to Literal	$P(a_1,\ldots,a_n) \mapsto Q(b_1,\ldots,b_m)$ s.t. $\texttt{type}(P(a_1,\ldots,a_n)) = \texttt{type}(Q(b_1,\ldots,b_m))$ $\texttt{type}(Q) \in \{\texttt{type}(c) : c \in \mathcal{O}\}$ $\texttt{freev}(P(a_1,\ldots,a_n)) = \texttt{freev}(Q(b_1,\ldots,b_m))$ $\{b_1,\ldots,b_m\} \in \texttt{subexps}(P(a_1,\ldots,a_n))$	$eq(x,count(\lambda y.People(y) \wedge \exists e.Visit(y, PublicLibraryOfNewYork) \wedge Annually(e))) \mapsto CountPeopleVisitAnnually(x, PublicLibraryOfNewYork)$ Input and output have type t. New constant has type $\langle i,\langle e,t\rangle\rangle$, allowed in $\mathcal{O}$. Input and output contain single free variable x. Arguments of output literal are subexpressions of input.
c. Split Literal	$P(a_1,\ldots,a_k,x,a_{k+1},\ldots,a_n) \mapsto Q(b_1,\ldots,x,\ldots b_n) \wedge Q''(c_1,\ldots,x,\ldots c_m)$ s.t. $\texttt{type}(P(\ldots)) = t$ $\{\texttt{type}(Q),\texttt{type}(Q'')\} \in \{\texttt{type}(c) : c \in \mathcal{O}\}$ $\{b_1,\ldots,b_n,c_1,\ldots,c_m\} = \{a_1,\ldots,a_n\}$	$Dedicate(Mozart,Haydn,ev) \mapsto Dedicate(Mozart,ev) \wedge Dedicate''(Haydn,ev)$ Input has type t. This matches output type by definition. New constants have allowed type $\langle e,\langle ev,t\rangle\rangle$. All arguments of input literal are preserved in output.

Figure 8: Transformation rules

l_0 : $\lambda x.eq(x,count(\lambda y.People(y) \wedge \exists e.Visit(y,\iota z.Public(z) \wedge Library(z) \wedge Of(z,NewYork),e) \wedge Annually(e)))$
l_1 : $\lambda x.eq(x,count(\lambda y.People(y) \wedge \exists e.Visit(y,\textbf{PublicLibraryOfNewYork},e) \wedge Annually(e)))$
l_2 : $\lambda x.\textbf{HowManyPeopleVisitAnnually}(x,PublicLibraryOfNewYork)))$

Figure 9: Example transformations

The score of a derivation is a linear function that decomposes over the parse tree Π and the ontological matching steps o (ϕ is a feature vector and θ a weight vector) :

$$\begin{aligned} SCORE(d) &= \phi(d)\theta \\ &= \phi(\Pi)\theta + \textstyle\sum_{o \in M} \phi(o)\theta \end{aligned}$$

The parameters θ are estimated from a set of question-answer pairs using a perceptron to estimate a weight vector θ that separates correct from incorrect answers.

The system is evaluated on the GeoQuery dataset [ZM96] and on the Freebase query (FQ) dataset introduced by [CY13]).

3.3.2 Paraphrase based approaches

Paraphrase based approaches [FZE14, FZE13, WBL15, BCFL13, BL14] derive a set of candidate logical forms from the input question, generate paraphrases based on these logical forms and on the text descriptions of predicates from the KB and finally choose the generated utterance which best paraphrases the input questions and thereby the logical form that generated it.

Question Pattern	Database Query
who r e	r(?, e)
what r e	r(?, e)
who does e r	r(e, ?)
what does e r	r(e, ?)
what is the r *of* e	r(?, e)
who is the r *of* e	r(?, e)
what is r *by* e	r(e, ?)
who is e *'s* r	r(?, e)
what is e *'s* r	r(?, e)
who is r *by* e	r(e, ?)
when did e r	r-in(e, ?)
when did e r	r-on(e, ?)
when was e r	r-in(e, ?)
when was e r	r-on(e, ?)
where was e r	r-in(e, ?)
where did e r	r-in(e, ?)

Figure 10: PARALEX Patterns

PARALEX While in [KCAZ13], the mismatches between words and KB symbols is handled by transformation rules, in the PARALEX system presented in [FZE14, FZE13], this mismatch is handled by learning from a paraphrase corpus of questions called Wikianswers which gathers 18 million paraphrase pairs for 2.4 distinct questions.

[FZE13] automatically construct a lexicon which encodes mappings from natural language to database concepts (entities, relations and queries). This lexicon is then used to derive KB queries from NL questions and derivations are scored using a hidden variable structured perceptron model trained on question-query pairs (parameters are updated to maximise the derivation score of correct queries).

The lexicon is built by first creating a seed lexicon using 16 hand-written question patterns and the identity transformation on entity and relation strings in the database. Figure 10 show the question patterns used. Next MGIZZA is used to extract word and phrase paraphrases from the Wikianswers corpus and these paraphrases used to enrich the seed lexicon. For instance, if the seed lexicon contains the following mappings:

what is the r of e	=	r(?,e)
population	=	population
new york	=	new-york

and the following alignments are extracted from Wikianswers:

what	how
population	big
of	is
new york	NYC

then the following mappings will be added to the seed lexicon:

how r is e	=	r(?,e)
big	=	population
nyc	=	new-york

Paralex is evaluated on REVERB, a set of triples $r(e_1, e_2)$ over a vocabulary of 600K relations and 2M entities. The test set consists of 37 question clusters from Wikianswers and 698 questions. For training, a data set of 48K (x, a, l) question-answer-label tuples is built semi automatically using semantic parsing and manually labelling each question answer pair.

SEMPRE In [BCFL13], a paraphrase lexicon is created from text and KB by aligning phrases and predicates using type checking. Typed phrases (55K binary and 6K unary phrases) are extracted from REVERB output (e.g., "born in" [person, location]). Given a typed phrase R_L from Reverb and a logical predicate R_K from Freebase, R_L will be aligned with R_K if their type signature match and their extensions have non-empty overlap. The output lexicon contains 109K binary predicates pairs and 294K unary. Lexical entries are associated with features which are used by a log linar model in conjuction with other sources of information to score competing derivations.

Given a natural language question, derivations are constructed recursively based on this lexicon and on a small set of composition rules and a discriminative log linear model over derivation is trained on question-answer pairs by maximizzing the log likelihood of the correct answer and used to rank the derivation candidates. More specifically, the lexicon is used to generate single predicate derivations for anay matching span while 5 operations (intersection, join, aggregation, count and bridging) are used to combine non overlapping text spans. Bridging accounts for cases where a KB relation is implicit or weakly verbalised (preposition, copula) and relies on type constraint to best guess binary relations between detected entities.

The approach is evaluated on FREE917 (917 questions with answers in Freebase) and on WEBQUESTIONS (5810 questions with answers in Freebase).

[BL14] extends on [BCFL13] by integrating an explicit generation component into semantic parsing. Given a question x, a set of LFs Z_x is constructed. For each $z \in Z_x$, a set of canonical utterances C_z is generated using templates. The generated canonical utterances $c \in C_z$ are then compared wrt the input question x using a paraphrase model and the LF of the

canonical utterance which is most similar to x is output.

#	Template	Example	Question
1	$p.e$	`Directed.TopGun`	*Who directed Top Gun?*
2	$p_1.p_2.e$	`Employment.EmployerOf.SteveBalmer`	*Where does Steve Balmer work?*
3	$p.(p_1.e_1 \sqcap p_2.e_2)$	`Character.(Actor.BradPitt ⊓ Film.Troy)`	*Who did Brad Pitt play in Troy?*
4	`Type.`$t \sqcap z$	`Type.Composer ⊓ SpeakerOf.French`	*What composers spoke French?*
5	`count(`z`)`	`count(BoatDesigner.NatHerreshoff)`	*How many ships were designed by Nat Herreshoff?*

Table 1: Logical form templates, where p, p_1, p_2 are Freebase properties, e, e_1, e_2 are Freebase entities, t is a Freebase type, and z is a logical form.

	$d(p)$ Categ.	Rule	Example
$p.e$	NP	`WH` $d(t)$ `has` $d(e)$ `as` NP ?	*What **election contest** has **George Bush** as **winner?***
	VP	`WH` $d(t)$ `(AUX)` VP $d(e)$?	*What **radio station serves area New-York**?*
	PP	`WH` $d(t)$ PP $d(e)$?	*What **beer from region Argentina**?*
	NP VP	`WH` $d(t)$ VP `the` NP $d(e)$?	*What **mass transportation system served** the **area Berlin**?*
$\mathbf{R}(p).e$	NP	`WH` $d(t)$ `is the` NP `of` $d(e)$?	*What **location** is the **place of birth** of **Elvis Presley**?*
	VP	`WH` $d(t)$ `AUX` $d(e)$ VP ?	*What **film** is **Brazil featured in**?*
	PP	`WH` $d(t)$ $d(e)$ PP ?	*What **destination Spanish steps near travel destination**?*
	NP VP	`WH` NP `is` VP `by` $d(e)$?	*What **structure** is **designed** by Herod?*

Table 2: Generation rules for templates of the form $p.e$ and $\mathbf{R}[p].e$ based on the syntactic category of the property description. Freebase descriptions for the type, entity, and property are denoted by $d(t)$, $d(e)$ and $d(p)$ respectively. The surface form of the auxiliary `AUX` is determined by the POS tag of the verb inside the VP tree.

Figure 11: Sempre templates

The lexicon described in [BCFL13] is used to map NL phrases to Free-Base predicates. Utterances are generated from LFs using the rules shown in Figure 11. In average, 1423 canonical utterances / input question are produced. Candidate derivations are then scored using a paraphrase model which Decomposes into an association model and a vector space model:

$$\phi_{pr}(x,c)^T\theta_{pr} = \phi_{as}(x,c)^T\theta_{as} + \phi_{vs}(x,c)^T\theta_{vs}$$

The *association model* determines whether x and c contains phrases that are likely to be paraphrases.
Go through all spans of x and c and identify pairs of potential paraphrases looking up (i) a phrase table derived by alignement from Wikianswers (18 M Q/A pairs) and (ii) a lexicon of token pairs which share the same lemma, the same POS tag or are linked thru a derivation link in WN. The weights learned discriminate good from bad associations using the features shown in the table below.

The *vector space model* assigns a vector representation to each utterance and learns a scoring function. For each utterance x, the vector v_x is obtained by averaging the vectors of all content words in x. Word vectors created using word2vec tools on Wikipedia text. The paraphrase score is a weighted combination of the components of the vectors.

Finally, [WBL15] demonstrates how a similar, generation based approach to semantic parsing can be used to quickly develop semantic parsers for arbitrary domains (cf. Figure 12). Synthetic data for training and testing is

Category	Description
Assoc.	lemma($x_{i:j}$) ∧ lemma($c_{i':j'}$) pos($x_{i:j}$) ∧ pos($c_{i':j'}$) lemma($x_{i:j}$) = lemma($c_{i':j'}$)? pos($x_{i:j}$) = pos($c_{i':j'}$)? lemma($x_{i:j}$) and lemma($c_{i':j'}$) are synonyms? lemma($x_{i:j}$) and lemma($c_{i':j'}$) are derivations?
Deletions	Deleted lemma and POS tag

Table 3: Full feature set in the association model. $x_{i:j}$ and $c_{i':j'}$ denote spans from x and c. pos($x_{i:j}$) and lemma($x_{i:j}$) denote the POS tag and lemma sequence of $x_{i:j}$.

created as follows. Databases are created for seven domains by randomly generating facts using entities and properties in the domain. Natural languages questions are then generated using a domain general grammar and paraphrases of these questions are authored by humans using Amazon Mechanical Turk (AMT).

The approach combines a seed lexicon (identical KB/text mapping), a generic grammar generating canonical utterances and their logical form and a paraphrase model trained on the training set $d = \{(x, c, z)\}$ with x the input utterance, c a canonical utterance, z an LF to estimate the log linear distribution $p_\theta(z, c \mid x, w)$. During parsing, beam search is used to generate paraphrases that are most similar to the input query. As in [BL14], the best ranked LF are the LF of the generated paraphrases that are deemed most similar to the input question by the model.

3.3.3 Embedding based approaches

In the PARALEX and SEMPRE approaches, KB symbols and words are assigned discrete representations and the mapping between natural language questions and KB queries involves lexicons, transformations rules and parsing. In contrast, [BCW14, BGWB12] present an approach based on continuous representations, so called word embeddings which are learned to maximise the similarity between KB triples and the corresponding natural language questions.

These embeddings are learned from a training corpus which is automatically constructed as follows.

First, question/answer pairs are created by rewriting KB symbols as words and applying the 16 patterns shown in Figure 13 to generate questions from REVERB triples. Negative examples are additionally created from positive ones. Given this data, embeddings for words and KB triples are

Domain	# pred.	# ex.	Phenomena	Example
CALENDAR	22	837	temporal language	x: *"Show me meetings after the weekly standup day"* c: *"meeting whose date is at least date of weekly standup"* z: type.meeting ⊓ date. > **R**(date).weeklyStandup
BLOCKS	19	1995	spatial language	x: *"Select the brick that is to the furthest left."* c: *"block that the most number of block is right of"* z: argmax(type.block, **R**(λx.count(**R**(right).x)))
HOUSING	24	941	measurement units	x: *"Housing that is 800 square feet or bigger?"* c: *"housing unit whose size is at least 800 square feet"* z: type.housingUnit ⊓ area. > .800
RESTAURANTS	32	1657	long unary relations	x: *"What restaurant can you eat lunch outside at?"* c: *"restaurant that has outdoor seating and that serves lunch"* z: type.restaurant ⊓ hasOutdoorSeating ⊓ serveslunch
PUBLICATIONS	15	801	sublexical compositionality	x: *"Who has co-authored articles with Efron?"* c: *"person that is author of article whose author is efron"* z: type.person ⊓ **R**(author).(type.article ⊓ author.efron)
SOCIAL	45	4419	multi-arity relations	x: *"When did alice start attending brown university?"* c: *"start date of student alice whose university is brown university"* z: **R**(date).(student.Alice ⊓ university.Brown)
BASKETBALL	24	1952	parentheticals	x: *"How many fouls were played by Kobe Bryant in 2004?"* c: *"number of fouls (over a season) of player kobe bryant whose season is 2004"* z: count(**R**(fouls).(player.KobeBryant ⊓ season.2004)

Table 3: We experimented on seven domains, covering a variety of phenomena. For each domain, we show the number of predicates, number of examples, and a (c, z) generated by our framework along with a paraphrased utterance x.

Figure 12: Semantic Parsing on Seven distinct Domains

learned using a ranking loss ($\forall i, \forall t' \neq t_i, S(q_i, t_i) > 0.1 + S(q_i, t')$). That is, the triple that labels a given question should be scored higher than other triples by a margin of 0.1.

The scoring function S is defined as:

$$S(q,t) = \mathbf{f}(q)^T \mathbf{g}(t)$$

where $\mathbf{f}(.)$ maps words from questions into $\mathbb{R}^k$ and is defined as

$$\mathbf{f}(q) = \mathbf{V}^T \phi(q)$$

where $\mathbf{V}$ is the matrix containing the word embeddings to be learned and ϕ the binary representation of q. Similarly, $\mathbf{g}(.)$ maps KB symbols into $\mathbb{R}^k$ and is defined as

$$\mathbf{g}(t) = \mathbf{W}^T \psi(t)$$

where $\mathbf{W}$ is the matrix containg the embeddings for the KB symbols and ψ the binary representation of t.

$\mathbf{W}$ and $\mathbf{V}$ are learned using stochastic gradient descent on the artificial corpus of question/answer pair constructed from REVERB.

To handle paraphrastic variations, the Wikianswers paraphrase corpus for questions is used to learn a scoring function between two questions:

$$S_{prp}(q_1, q_2) = \mathbf{f}(q_1)^T \mathbf{f}(q_2)$$

The matrix $\mathbf{V}$ containing the word embeddings is shared between S and S_{prp} allowing it to encode information from examples from both the REVERB and the Wikianswers corpus.

The approach is evaluated on the test set created by [BCFL13] and outperforms PARALEX by a wide margin (F1: 0.73).

Figure 13: Templates used to generate questions from triples

4 Semantic parsing for ontology building and evolution

As in the frame of the ModelWriter project, we target the creation of links between the Airbus documents and the Airbus ontologies, we review in this section the approaches proposed in the literature that address the semantic parsing in the ontology domain. In particular, we present the approaches and tools that build and/or enrich ontologies from texts.

4.1 Ontology building

Build an ontology is a hard task which depends on several factors and contributors: user requirements, domain experts, engineers ontology, lexical resources, relevant documentations, etc. This task typically requires the interaction with domain experts and needs a lot of time and resources (for instance, the construction of the linguistic ontology WordNet [Mil95] took 10 years).

With the widespread of lexical resources (e.g. VerbNet [KDP+00], FrameNet [JFP+02]), thesaurus (UMLS [LHM93]), dictionaries (BabelNet [NP10]) and knowledge base (Wikipedia[5], Freebase [BEP+08]), several researchers are interested to build ontologies from texts. The idea consists to exploit these resources in order to automatically extract knowledge from texts and thus reduce the cost of the building process.

In the following, we present some approaches and tools which address the issue of the learning ontologies from texts.

[CV05] propose Text2Onto, a framework for ontology learning from textual resources. It is an evolution of TextToOnto, a plugin for kaon tool [BEH+02]. To build an ontology, Text2Onto adopts the cake layer strategy and uses the following steps:

1. *Lexical Entry and Concept Extraction.* The ontology building process starts by extraction relevant terms from the corpus using: 1) TF/IDF (Term Frequency Inverted Document Frequency); 2) Entropy and 3) C-value/NC-value [FAT98].

2. *Subclass-of Relations Identification.* To specify hierarchical relations, Text2Onto is based on: 1) hypernym structure of WordNet and 2) matching Hearst patterns [CPSTS05]. This step is evaluated on a tourism resources corpora. The best result obtained is an F-Measure of 21.81%, a precision of 17.38% and a recall of 29.95%.

3. *Mereological Relations Identification.* Text2Onto gathers the part-of relations between two terms using JAPE tool [CV05].

4. *General Relations Identification.* This step uses a shallow parsing strategy to extract subcategorization frames, such as [CV05]:

transitive, e.g. love(subj, obj)

intransitive + PP-complement, e.g. walk(subj, pp(to))

transitive + PP-complement, e.g. hit(subj, obj, pp(with))

[5] https://www.wikipedia.org

Then, the system maps this subcategorization frames to ontological relations. For example:

hit(subj:person, obj:thing, with:object)

$\Downarrow$

hit(domain:person, range:thing)

hitwith(domain:person, range:object)

Text2Onto is one of the most famous frameworks for building ontology from texts. It combines many mechanisms and tools to automatically identify concepts and relations from texts. However, it has some limits such as it doesn't study the restrictions and axioms (e.g. disjoint axioms identification). Some of its activities are tested (e.g. hierarchical relations identification), but there is no information about the evaluation of the resulting ontology and its quality.

[CSP$^+$08, SAGC$^+$10] propose an approach and platform called DAFOE: a Multi model and Multi method Platform for Building Domain Ontologies. DAFOE uses tools similar to those of Text2Onto, but it offers more freedom to users to choose the combination tools and to validate their results. The platform is composed of three layers:

1. The *Terminological Layer* extracts relevant terms of the domain and identify their relationships using YaTeA term extractor [AH06].
2. The *Termino-Conceptual Layer* represents a semantic structure of unambiguous termino-concepts (TC) and termino-conceptual relations (RTC). It is a manual task done by knowledge engineers.
3. The *Ontology Layer* formalizes TCs and RTCs in a formal language (OWL-DL). It is an automatic task that transforms concepts into classes, relations into objectProperty, etc.

The DAFOE result is typically a Termino-Ontological Resource where the ontology is connected to a lexical component. DAFOE is close to Text2Onto framework, but it requires more users intervention. As Text2Onto, DAFOE doesn't present the evaluation of resulting ontology (is-it consistent or not? is-it expressive or not?, etc.).

[AACT13] propose an approach to build ontology from sources codes program. The idea consists to analyse the identifiers and use the natural language dependency trees generated by the analyzers to extract the concepts and relations for the ontology. The approach consists in the following tasks:

1. *Identifier Parser.* It is composed by several subtasks:
   (a) *Syntactic analysis.* To construct a syntactic analysis for an input identifier, the authors use firstly the tokenization (the process of splitting a text into words or linguistic elements called tokens or terms). After that, they apply the PoS (Part of Speech) tagging.
   (b) *Sentence construction.* To generate a sentence from an identifier term list, the authors have formulated different rules presented in Table 14. The rules are defined for the three main identifier types (Class term list, Method term list and Attributes term list). For example, the class identifier names are usually constructed from a noun, multiple nouns or adjectives followed by nouns.
   (c) *Syntactic analyzer.* Two tools are used: Minipar [Lin03] with PoS tagger and Malt parser [NHN06] with SVMTool PoS tagger.
2. *Ontology extraction.* This step consists on the extraction of ontology concepts, typically the classes and relations. The ontology classes are retrieved from the nouns referred in the parse trees of identifiers, or directly from the class or program names. The ontological relations are obtained from the dependencies found in the parse trees of the identifiers and the verbs used in method names. Four types of relations are extracted: isA, verb, hasProperty and hasState.

Rule	Class term list	Generated sentence	Constraint
CR1	$C=\langle T_1\rangle$	T_1 "*is a thing*"	T_1 is a noun or an adjective
CR2	$C=\langle T_1\rangle$	T_1*er* "*is a thing*"	T_1 is a verb
CR3	$C=\langle T_1, T_2, \ldots\rangle$	$T_1T_2\ldots$ "*is a thing*"	T_1 is a noun or an adjective
CR4	$C=\langle T_1, T_2, \ldots\rangle$	T_1*ing* $T2\ldots$ "*is a thing*"	T_1 is a verb

Rule	Method term list	Generated sentence	Constraint
MR1	$M=\langle T_1\rangle$	"*Subjects*" T_1 "*object*"	T_1 is a verb
MR2	$M=\langle T_1\rangle$	"*Subjects get*" T_1	T_1 is a noun
MR3	$M=\langle T_1, T_2, \ldots\rangle$	"*Subjects*" $T_1T_2\ldots$	T_1 is a verb
MR4	$M=\langle T_1, T_2, \ldots\rangle$	"*Subjects get*" $T_1T_2\ldots$	T_1 is a noun or an adjective
MR5	$M=\langle T_1, T_2, \ldots\rangle$	"*Subjects handle*" $T_2\ldots$	T_1 is the preposition "on"
MR6	$M=\langle T_1, T_2, \ldots\rangle$	"*Subjects convert*" $T_2\ldots$	T_1 is the preposition "to"

Rule	Attribute term list	Generated sentence	Constraint
AR1	$A=\langle T_1\rangle$	T_1 "*is a thing*"	T_1 is a noun or an adjective
AR2	$A=\langle T_1\rangle$	T_1*er* "*is a thing*"	T_1 is a verb which is not a past participle, or T_1 is a past participle verb and A is not of boolean type
AR3	$A=\langle T_1\rangle$	T_1 "*subjects are things*"	T_1 is a past participle verb and A has a boolean type
AR4	$A=\langle T_1, T_2, \ldots\rangle$	$T_1T_2\ldots$ "*is a thing*"	T_1 is a noun or an adjective
AR5	$A=\langle T_1, T_2, \ldots\rangle$	T_1*ing* $T_2\ldots$ "*is a thing*"	T_1 is a verb

Figure 14: Rules to generate sentences from term lists [AACT13].

The approach considers the lightweight ontology which are only composed with concepts and relations.

4.2 Ontology evolution

Ontologies Evolution is defined by Stojanovic et al. as "the timely adaptation of an ontology to the arisen changes and the consistent propagation of these changes to dependent artefacts " [Sto04]. This process consists in the modification of one or many ontology components (class, property, axiom, individual, etc.) and it may be at instances level (Ontology Population) and/or structural level (Ontology Enrichment) [KLL13]. Several ontology changes can be applied on an ontology (e.g. AddClass, AddRelation, RemoveAxiom, etc. [MFTH13, MFTH15]). To define these changes, the ontology engineers could specify by themselves the changes or to base on a set of changes automatically identified by a machine. In the second strategy, the changes are identified from documents and texts which describe the model domain.

The approaches presented in Section 4.1 could be use on the identification of ontology changes and thus in the evolution process. We add here the description of [SCAG13] work.

[SCAG13] propose a tool for ontology building and evolution from texts, called DYNAMO-MAS (DYNAMic Ontology for information retrieval-Multi-Agent System). The approach is composed of four main steps:

1. *Corpora enrichment* adds new documents to the corpora to have large resources.
2. *Knowledge extraction* analyses the corpora documents in order to identify new terms and relations which are not represented in the initial ontology.
3. *Knowledge interpretation and ontology evolution* integrate the new knowledge to the ontology, while proposing to user different strategies of evolution;
4. The *Evolution strategies evaluation* is a manual task which bases on user (ontology engineer) who evaluates the evolution strategies proposed by the system and choose the most appropriate.

Like DAFOE, the DYNAMO-MAS result is also a Termino-Ontological Resource. Although that the approach represents the formalization of several ontology changes, it doesn't take into account the inconsistencies resolution issue. The authors do not present how their system could deny or resolve the inconsistencies during the evolution process.

4.3 Discussion

Build or evolve an ontology from texts is a promising perspective given that it can reduce time and minimize the domain expert intervention. However,

it still suffers from some difficulties [AG12]:

- *Difficulties related to the languages complexity.* We can cite here the case of the polysemy and how to understand the correct mining of a term (e.g. book (read a book), book (book a room)) ;
- *Difficulties related to the NLP tools.* The extraction of the relevant terms depends on the quality of the NLP tools used which should correctly chosen to have good results ;
- *Difficulties related to the modelling.* Understanding text and represent it into concepts differs from one person to another. For that, we can find different conceptualisation for the same sentence.

We would also note that, although the important works proposed for building ontologies from texts, the evaluation of resulting ontologies is often missing. The process of building or evolving ontology from texts should be controlled to obtain a consistent ontology. Resolve ontology inconsistencies can be achieved with two manners : 1) an a priori manner which consists to deny the application of any change which can alter the ontology consistency [MFTH13, MTFH14] or 2) a posteriori manner which consists to check the ontology consistency after the application of changes and it usually uses reasoners such as Pellet, Racer, etc.

5 Conclusion

In this report, we have presented an overview of the semantic parsing. To sum up, existing NLP approaches to semantic parsing can be divided into three main categories.

Logical approaches which use a linguistically grounded grammar to simultaneously construct the syntac and the semantics of a sentence. For ModelWriter such approaches would minimally need to be modified to appropriately account for "real" or "grounded" meaning representations which can effectively be linked to the models being synchronised (e.g., the OWL knowledge base used by AIRBUS to describe plane components and their related System Installation Design Principles Rules). Relatedly, techniques will be needed to improve portability i.e., to allow for the same semantic parser to be used independent of the domain. This requires in particular, the development of high precision methods for automatically producing domain specific grammars and lexicons. On the plus side, a very interesting feature of those logical approaches is that they rely on unification-based grammar and as such, are easily reversible. That is, these approaches have the potential to be used both for parsing and for generation, an important point for ModelWriter since WP2 targets the developement of a reversible semantic processor.

Domain specific approaches are fully automatic and the meaning representations they produce are well grounded in the data being handled (e.g., robot instructions or CLANG knowledge representation language for soccer game descriptions). However they require the existence of a parallel data-to-text corpus and such corpora are costly to construct (humans typically find assigning a sentence some abstract meaning represenation in a given formal language very difficult). This is a major drawback as this means that these approaches are not only costly in terms of human work but also non generic. For each new domain, a new parallel corpus must be build. For the ModelWriter project, we will instead make use of automatica alignment techniques to align text and data at the lexical level but rely on more generic methods for inducing semantic grammars.

Finally, domain independent approaches combine many advantages. They produced meaning representations that are grounded e.g., in knowldege bases such as Freebase and the can handle open domain questions. With respect to ModelWrire's goals however, important limitations of these approaches is that (i) they require extensive training data (millions of sentences and RDF triples) and (ii) they have been tested mainly on very simple questions whose answer is often restricted to single RDF tiples. In contrast, in ModelWriter, training data is limited and the meaning representations to be delivered may be more complex than simple RDF triples. For instance, the AIRBUS SIDP rule (1) should be assigned the Description Logic formula shown in (2).

(1) Spare wire shall be used only on development aircraft.

(2) $\exists useArg2^{-}.[Use \wedge \exists useArg1^{-}.SpareWire] \sqsubseteq DevelopmentAircraft$

Based on these observations, we will target the developement of a semantic parser which combines a small, generic grammar (to mininise manual intervention, to support domain independence and to allow for a reversible approach) with an automatic alignement procedure at the lexical level (to ground the meaning representations generated by the grammar in the models explored by ModelWriter in particular, Description Logic models) and with techniques from open domain question answering (to improve robustness).

The resulting parser will in particular be used in the Usecase UC-FR4 (Synchronisation of regulation documentatio with a design rule repository) to enrich an existing ontology, to check update consistency and to synchronise text and model.

References

[AACT13] Surafel Lemma Abebe, Anita Alicante, Anna Corazza, and Paolo Tonella. Supporting concept location through identifier parsing and ontology extraction. *Journal of Systems and Software*, 86(11):2919–2938, 2013.

[AG12] Nathalie Aussenac-Gilles. Donner du sens à des documents semi-structurés: de la construction d'ontologies à l'annotation sémantique. In *Séminaire IST Inria: le document numérique à l'heure du web de données*, pages 105–140. ADBS, 2012.

[AH06] Sophie Aubin and Thierry Hamon. Improving term extraction with terminological resources. In *Advances in Natural Language Processing*, pages 380–387. Springer, 2006.

[BA06] Caroline Barriere and Akakpo Agbago. Terminoweb: a software environment for term study in rich contexts. In *International Conference on Terminology, Standardization and Technology Transfer (TSTT 2006*, pages 103–113, 2006.

[BB05] Patrick Blackburn and Johan Bos. Representation and inference for natural language. *A first course in computational semantics. CSLI*, 2005.

[BCFL13] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *EMNLP*, pages 1533–1544, 2013.

[BCM05] Paul Buitelaar, Philipp Cimiano, and Bernardo Magnini. *Ontology learning from text: methods, evaluation and applications*, volume 123. IOS press, 2005.

[BCS+04] J. Bos, S. Clark, M. Steedman, J. R. Curran, and J. Hockenmaier. Wide-coverage semantic representations from a ccg parser. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING '04)*, Geneva, Switzerland., 2004.

[BCW14] Antoine Bordes, Sumit Chopra, and Jason Weston. Question answering with subgraph embeddings. *arXiv preprint arXiv:1406.3676*, 2014.

[BEH+02] Erol Bozsak, Marc Ehrig, Siegfried Handschuh, Andreas Hotho, Alexander Maedche, Boris Motik, Daniel Oberle, Christoph Schmitz, Steffen Staab, Ljiljana Stojanovic, et al. Kaon—towards a large scale semantic web. In *E-Commerce and Web Technologies*, pages 304–313. Springer, 2002.

[BEP+08] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM, 2008.

[BG09] Paul Bédaride and Claire Gardent. Semantic normalisation: a framework and an experiment. In *Proceedings of the Eighth International Conference on Computational Semantics*, pages 359–370. Association for Computational Linguistics, 2009.

[BG10] Paul Bedaride and Claire Gardent. Syntactic testsuites and textual entailment recognition. In *The seventh International Conference on Language Resources and Evaluation-LREC 2010*, 2010.

[BGWB12] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. Joint learning of words and meaning representations for open-text semantic parsing. In *International Conference on Artificial Intelligence and Statistics*, pages 127–135, 2012.

[BL14] Jonathan Berant and Percy Liang. Semantic parsing via paraphrasing. In *Proceedings of ACL*, volume 7, page 92, 2014.

[BNT08] Rokia Bendaoud, Amedeo Napoli, and Yannick Toussaint. Formal concept analysis: A unified framework for building and refining ontologies. In *Knowledge Engineering: Practice and Patterns*, pages 156–171. Springer, 2008.

[Bos95] J. Bos. Predicate Logic Unplugged. In *Proceedings of the tenth Amsterdam Colloquium, Amsterdam*, 1995.

[CCB07] James R Curran, Stephen Clark, and Johan Bos. Linguistically motivated large-scale nlp with c&c and boxer. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 33–36. Association for Computational Linguistics, 2007.

[CCMO99] Ann Copestake, John Carroll, Rob Malouf, and Stephan Oepen. The (new) lkb system. *Center for the Study of Language and Information, Stanford University*, 1999.

[CF00] A. Copestake and D. Flickinger. An open-source grammar development environment and broad-coverage english grammar using hpsg. In *Proceedings of LREC*, Athens, Greece, 2000.

[CFH+03] M Chen, E Foroughi, S Heintz, S Kapetanakis, K Kostiadis, J Kummeneje, I Noda, O Obst, P Riley, T Steffens, et al. Robocup soccer server manual for soccer server version 7.07 or latest, 2003.

[CGY07] Nadine Cullot, Raji Ghawi, and Kokou Yétongnon. Db2owl: A tool for automatic database-to-ontology mapping. In *SEBD*, pages 491–494, 2007.

[CKZ15] Eunsol Choi, Tom Kwiatkowski, and Luke Zettlemoyer. Scalable semantic parsing with partial ontologies. In *Proceedings of ACL*, 2015.

[CPSTS05] Philipp Cimiano, Aleksander Pivk, Lars Schmidt-Thieme, and Steffen Staab. Learning taxonomic relations from heterogeneous sources of evidence. *Ontology Learning from Text: Methods, evaluation and applications*, 2005.

[CSP+08] Jean Charlet, Sylvie Szulman, Guy Pierra, Nadia Nadah, Henry Valéry Téguiak, Nathalie Aussenac-Gilles, and Adeline Nazarenko. Dafoe: A multimodel and multimethod platform for building domain ontologies. *2eJournées Francophones sur les Ontologies, Lyon, France: ACM*, 2008.

[CV05] Philipp Cimiano and Johanna Völker. Text2onto : A framework for ontology learning and data-driven change discover. In *Natural language processing and information systems*, pages 227–238. Springer, 2005.

[CY13] Qingqing Cai and Alexander Yates. Large-scale semantic parsing via schema matching and lexicon extension. In *ACL (1)*, pages 423–433. Citeseer, 2013.

[Dal99] Mary Dalrymple, editor. *Semantics and Syntax in Lexical Functional Grammar*. MIT Press, 1999.

[DCM+14] Dipanjan Das, Desai Chen, André FT Martins, Nathan Schneider, and Noah A Smith. Frame-semantic parsing. *Computational Linguistics*, 40(1):9–56, 2014.

[Del90] Rodolfo Delmonte. Semantic parsing with lfg and conceptual representations. *Computers and the Humanities*, 24(5-6):461–488, 1990.

[Dro03] Patrick Drouin. Term extraction using non-technical corpora as a point of leverage. *Terminology*, 9(1):99–115, 2003.

[DVF12] Emmanuelle Dietz, Damir Vandic, and Flavius Frasincar. Taxolearn: A semantic approach to domain taxonomy learning. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2012 IEEE/WIC/ACM International Conferences on*, volume 1, pages 58–65. IEEE, 2012.

[FAT98] Katerina T Frantzi, Sophia Ananiadou, and Junichi Tsujii. The c-value/nc-value method of automatic recognition for multi-word terms. In *Research and advanced technology for digital libraries*, pages 585–604. Springer, 1998.

[FMPT10] Enrico Francesconi, Simonetta Montemagni, Wim Peters, and Daniela Tiscornia. *Integrating a bottomup and topdown methodology for building semantic resources for the multilingual legal domain*. Springer, 2010.

[FZE13] Anthony Fader, Luke S Zettlemoyer, and Oren Etzioni. Paraphrase-driven learning for open question answering. In *ACL (1)*, pages 1608–1618. Citeseer, 2013.

[FZE14] Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. Open question answering over curated and extracted knowledge bases. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1156–1165. ACM, 2014.

[Gar14] Claire Gardent. Syntax and data-to-text generation. In *Statistical Language and Speech Processing*, pages 3–20. Springer, 2014.

[Gar15] Claire Gardent. D2.3.1 overview and comparison of existing generators modelwriter, text and model-synchronized document engineering platform. Technical report, CNRS-LORIA, 2015.

[GK03] C. Gardent and L. Kallmeyer. Semantic construction in FTAG. In *Proceedings of EACL'03, Budapest*, 2003.

[GK14] Anatoly P Getman and Volodymyr V Karasiuk. A crowdsourcing approach to building a legal ontology from text. *Artificial Intelligence and Law*, 22(3):313–335, 2014.

[GM05] Ruifang Ge and Raymond J Mooney. A statistical semantic parser that integrates syntax and semantics. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 9–16. Association for Computational Linguistics, 2005.

[GM09] Ruifang Ge and Raymond J Mooney. Learning a compositional semantic parser using an existing syntactic parser. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 611–619. Association for Computational Linguistics, 2009.

[GPFV96] Ascuncion Gómez-Pérez, Mariano Fernández, and A de Vicente. Towards a method to conceptualize domain ontologies. 1996.

[GR04] Peter Gorniak and Deb Roy. Grounded semantic composition for visual scenes. *Journal of Artificial Intelligence Research*, pages 429–470, 2004.

[Gru93] Thomas R Gruber. A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2):199–220, 1993.

[Hea92] Marti A Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*, pages 539–545. Association for Computational Linguistics, 1992.

[HG01] Lynette Hirschman and Robert Gaizauskas. Natural language question answering: the view from here. *natural language engineering*, 7(04):275–300, 2001.

[JFP+02] Christopher R Johnson, Charles J Fillmore, Miriam RL Petruck, Collin F Baker, Michael Ellsworth, Josef Ruppenhofer, and Esther J Wood. Framenet: Theory and practice, 2002.

[JS97] A. Joshi and Y. Schabes. Tree-adjoining grammars. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, pages 69 – 124. Springer, Berlin, New York, 1997.

[KCAZ13] Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. Scaling semantic parsers with on-the-fly ontology matching. 2013.

[KDP+00] Karin Kipper, Hoa Trang Dang, Martha Palmer, et al. Class-based construction of a verb lexicon. In *AAAI/IAAI*, pages 691–696, 2000.

[KLL13] Asad Masood Khattak, Khalid Latif, and Sungyoung Lee. Change management in evolving web ontologies. *Knowledge-Based Systems*, 37:1–18, 2013.

[KM06] Rohit J Kate and Raymond J Mooney. Using string-kernels for learning semantic parsers. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 913–920. Association for Computational Linguistics, 2006.

[KPT+04] Atanas Kiryakov, Borislav Popov, Ivan Terziev, Dimitar Manov, and Damyan Ognyanoff. Semantic annotation, indexing, and retrieval. *Web Semantics: Science, Services and Agents on the World Wide Web*, 2(1):49–79, 2004.

[KZGS10] Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. Inducing probabilistic ccg grammars from logical form with higher-order unification. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 1223–1233. Association for Computational Linguistics, 2010.

[LFT+15] Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A Smith. Toward abstractive summarization using semantic representations. 2015.

[LGMF04] Jure Leskovec, Marko Grobelnik, and Natasa Milic-Frayling. Learning sub-structures of document semantic graphs for document summarization. In *LinkKDD Workshop*, pages 133–138, 2004.

[LHM93] Donald A Lindberg, Betsy L Humphreys, and Alexa T McCray. The unified medical language system. *Methods of information in medicine*, 32(4):281–291, 1993.

[Lin03] Dekang Lin. Dependency-based evaluation of minipar. In *Treebanks*, pages 317–329. Springer, 2003.

[Liu13] Yang Liu. A shift-reduce parsing algorithm for phrase-based string-to-dependency translation. In *ACL (1)*, pages 1–10, 2013.

[LNLZ08] Wei Lu, Hwee Tou Ng, Wee Sun Lee, and Luke S Zettlemoyer. A generative model for parsing natural language to meaning representations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 783–792. Association for Computational Linguistics, 2008.

[MFTH13] Mariem Mahfoudh, Germain Forestier, Laurent Thiry, and Michel Hassenforder. Consistent ontologies evolution using graph grammars. In *Knowledge Science, Engineering and Management*, pages 64–75. Springer, 2013.

[MFTH15] Mariem Mahfoudh, Germain Forestier, Laurent Thiry, and Michel Hassenforder. Algebraic graph transformations for formalizing ontology changes and evolving ontologies. *Knowledge-Based Systems*, 73:212–226, 2015.

[Mil95] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[MM07] Bill MacCartney and Christopher D Manning. Natural logic for textual inference. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 193–200. Association for Computational Linguistics, 2007.

[MS00] Alexander Maedche and Steffen Staab. Semi-automatic engineering of ontologies from text. In *Proceedings of the 12th international conference on software engineering and knowledge engineering*, pages 231–239. Citeseer, 2000.

[MTFH14] Mariem Mahfoudh, Laurent Thiry, Germain Forestier, and Michel Hassenforder. Algebraic graph transformations for merging ontologies. In *Model and Data Engineering*, pages 154–168. Springer, 2014.

[NHN06] Joakim Nivre, Johan Hall, and Jens Nilsson. Maltparser: A data-driven parser-generator for dependency parsing. In *Proceedings of LREC*, volume 6, pages 2216–2219, 2006.

[NP10] Roberto Navigli and Simone Paolo Ponzetto. Babelnet: Building a very large multilingual semantic network. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 216–225. Association for Computational Linguistics, 2010.

[NS06] Rebecca Nesson and Stuart M. Shieber. Simpler TAG semantics through synchronization. In *Proceedings of the 11th Conference on Formal Grammar*, Malaga, Spain, 29–30 July 2006.

[PGF14] Laura Perez-Beltrachini, Claire Gardent, and Enrico Franconi. Incremental query generation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2014, April 26-30, 2014, Gothenburg, Sweden*, pages 183–191, 2014.

[Poo13] Hoifung Poon. Grounded unsupervised semantic parsing. In *ACL (1)*, pages 933–943, 2013.

[RDF00] Ehud Reiter, Robert Dale, and Zhiwei Feng. *Building natural language generation systems*, volume 33. MIT Press, 2000.

[RLS14] Siva Reddy, Mirella Lapata, and Mark Steedman. Large-scale semantic parsing without question-answer pairs. *Transactions of the Association for Computational Linguistics*, 2:377–392, 2014.

[SAGC+10] Sylvie Szulman, Nathalie Aussenac-Gilles, Jean Charlet, Adeline Nazarenko, Eric Sardet, and HV Teguiak. Dafoe: A platform for building ontologies from texts. In *EKAW 2010*, 2010.

[SCAG13] Zied Sellami, Valérie Camps, and Nathalie Aussenac-Gilles. Dynamo-mas: a multi-agent system for ontology evolution from text. *Journal on Data Semantics*, 2(2-3):145–161, 2013.

[Ste00] Mark Steedman. *The syntactic process*, volume 24. MIT Press, 2000.

[Sto04] Ljiljana Stojanovic. *Methods and Tools for Ontology Evolution*. PhD thesis, University of Karlsruhe, Germany, 2004.

[UC11] Christina Unger and Philipp Cimiano. Pythia: Compositional meaning construction for ontology-based question answering on the semantic web. In *Natural Language Processing and Information Systems*, pages 153–160. Springer, 2011.

[UK95] Michael Uschold and Martin King. *Towards a methodology for building ontologies*. Citeseer, 1995.

[VS07] Paola Velardi and Francesco Sclano. Termextractor: a web application to learn the common terminology of interest groups and research communities. In *7ème Confèrence" Terminologie et intelligence artificielle"*, pages 85–94, 2007.

[WBL15] Yushi Wang, Jonathan Berant, and Percy Liang. Building a semantic parser overnight. In *Association for Computational Linguistics (ACL)*, 2015.

[WLZD08] Dingding Wang, Tao Li, Shenghuo Zhu, and Chris Ding. Multi-document summarization via sentence-level semantic analysis and symmetric matrix factorization. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 307–314. ACM, 2008.

[WM06] Yuk Wah Wong and Raymond J Mooney. Learning for semantic parsing with statistical machine translation. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 439–446. Association for Computational Linguistics, 2006.

[WM07] Yuk Wah Wong and Raymond J Mooney. Learning synchronous grammars for semantic parsing with lambda calculus. In *Annual*

Meeting-Association for computational Linguistics, volume 45, page 960, 2007.

[YHM14] Wen-tau Yih, Xiaodong He, and Christopher Meek. Semantic parsing for single-relation question answering. In *Proceedings of ACL*, 2014.

[ZM96] John M Zelle and Raymond J Mooney. Learning to parse database queries using inductive logic programming. In *Proceedings of the National Conference on Artificial Intelligence*, pages 1050–1055, 1996.

[ZN08] Amal Zouaq and Roger Nkambou. Building domain ontologies from text for educational purposes. *IEEE Transactions on Learning Technologies*, 1(1):49–62, 2008.