Towards a Computational Framework for Winograd Schemas Resolution

Nicola Bova
School of Informatics
The University of Edinburgh
Edinburgh, UK EH8-9AB
Email: nbova@inf.ed.ac.uk

Michael Rovatsos
School of Informatics
The University of Edinburgh
Edinburgh, UK EH8-9AB
Email: mrovatso@inf.ed.ac.uk

***Abstract*—This article introduces a preliminary version of a computational framework for the solution of Winograd Schemas, a recently proposed alternative to the Turing test. These are pairs of sentences that differ in only one or two words and that contain an ambiguity that is resolved in opposite ways in the two sentences. Since Winograd schemas are constructed in such a way to be opaque to simple anaphora resolution techniques, Winograd Schema resolution requires the use of a large amount of world knowledge and reasoning for its resolution. The framework developed here translates each schema in First Order Logic relations through the use of Natural Language Processing tools and task-related assumptions. Then, it constructs a context of commonsense concepts and relations by querying the ConceptNet semantic network, stored in a graph database to simplify the querying process. The context is then expressed in First Order Logic, and finally one of the two candidates is selected by performing reasoning through an Automatic Theorem Prover which applies deduction over the expressions constructed earlier. We test our framework on a reduced subset of the Definite Pronoun Resolution Dataset, a dataset akin to the Winograd schemas one, and analyse the obtained results paying particular attention to the components for which there is room for improvement.**

Keywords*—*Winograd Schemas, Coreference Resolution, First Order Logic, Logic, Knowledge Bases, Semantic Networks, Graph Databases, Reasoning, Automatic Theorem Proving.

I. Introduction

A Winograd Schema (WS) [1] consists of a pair of sentences that differ in only one or two words and that contains an ambiguity which is resolved in opposite ways in the two sentences.[1] WSs take their name from a well-known example by Terry Winograd [2]:

> The city councilmen refused the demonstrators a permit because they [feared/advocated] violence. Who [feared/advocated] violence?
> **Answers:** The city councilmen/the demonstrators.

If the word is "feared", then "they" presumably refers to the city council; if it is "advocated" then "they" presumably refers to the demonstrators.

The formal description of a WS consists of three parts:

1) A brief discourse containing the following:
   - Two noun phrases of the same semantic class (male, female, inanimate, or group of objects/people) – the candidates,
   - An ambiguous pronoun that may refer to either of the above noun phrases, and
   - A special word and alternate word, such that if the special word is replaced with the alternate word, the natural resolution of the pronoun changes.
2) A question asking the referent of the ambiguous pronoun, and
3) Two answer choices corresponding to the noun phrases in question.

A machine will be given the problem in a standardised form which includes the answer choices, thus making it a binary decision problem.

On the surface, WS questions simply require anaphora resolution [3]: the machine must identify the antecedent of an ambiguous pronoun in a statement. However, Levesque argues that the task requires the use of world knowledge and reasoning for its resolution [4]. Therefore, WS resolution has been recently proposed as a modern alternative to the Turing test [1]. Levesque suggests [1] that WSs should be:

- Easily disambiguated by the human reader (ideally, so easily that the reader does not even notice that there is an ambiguity);
- Not solvable by simple techniques such as selectional restrictions;
- Google-proof, that is, there is no obvious statistical test over text corpora that will reliably disambiguate these correctly.

Another example of WS is:

> The man couldn't lift his son because he was so [weak/heavy]. Who was [weak/heavy]?
> **Answers:** The man/the son.

To answer this, a computer would have to know that weight has a positive correlation with age, that, in general, heavier people are stronger than lighter ones, that lifting requires sufficient strength to overcome the weight of an object, that weakness is a property of a person that can reduce the default strength, and that light children can be lifted, but probably not heavy ones.

[1] A Collection of WSs - http://www.cs.nyu.edu/davise/papers/WS.html. Accessed 04 June 2015.

At a more abstract level, each WS provides a certain amount of specific knowledge by expressing some statement of facts along with a query about the expressed facts. To answer the query, on the one hand, it is necessary to relate and link the provided knowledge with some relevant context not included in the expressed facts. On the other hand, it is necessary to perform some reasoning on the knowledge expressed as the union of the specified facts and the relevant context.

The aim of this paper is to design a computational framework for WS resolution. This framework translates each schema in First Order Logic (FOL) [5] relations, constructs a suitable context by searching external sources of information, expresses the context in FOL, and selects one of the two candidates by performing reasoning over the logical expressions constructed earlier. To show the feasibility of this approach, we present a preliminary version of this framework that is able to solve a small set of examples from the Definite Pronoun Resolution[2] (DPR) dataset [6].

This research was carried out as part of the ESSENCE Marie Curie Initial Training Network[3], an European project dealing with the Evolution of Shared SEmaNtics in Computational Environments (hence the acronym). For this reason, we aim at a certain degree of flexibility of the framework, to allow us to easily extend it to tackle other semantic-related tasks.

The structure of this article is as follows. Sec. II reviews the existing proposals dealing with WSs. Sec. III introduces the overall structure of the presented framework while Sec. IV describes the translation of schemas from natural language into FOL. Sec. V is devoted to the description of the employed Knowledge Bases (KBs) along with the process of creating a context of relations to connect the information expressed in schemas to commonsense knowledge. Sec. VI describes the translation of the information within the context into FOL and the reasoning process. Sec. VII deals with testing our proposal and the analysis of the obtained results. Finally, Sec. VIII delineates the future work necessary to complete our framework and Sec. IX summarises our conclusions on the work carried out so far.

II. Related work

In linguistics, coreference [7] occurs when two or more linguistic expressions refer to the same entity, that is, they have the same referent, e.g., *Mark said he was hungry*; the proper noun *Mark* and the pronoun *he* refer to the same person, namely to Mark.

To derive the correct interpretation of a discourse, in computational linguistics [8] pronouns and other referring expressions must be connected to the right individuals in the domain of discourse. Algorithms intended to resolve coreferences commonly look first for the nearest preceding individual that is compatible with the referring expression. For example, *he* might make reference to a preceding expression such as *the man* or *Mark*, but not to *Sarah*.

Previous approaches [9]–[17], however, cannot be employed to successfully resolve coreference problems as complex as those found in WSs, as shown in [6]. Other approaches extract world knowledge from online encyclopaedias such as Wikipedia [18], [19], YAGO [20]–[22], and Freebase [17]. However, the resulting extractions are primarily *IS-A* relations (e.g., Barack Obama *IS-A* U.S. president), which would not be useful [6] for resolving definite pronouns.

Differently, a recent statistical approach [6] encodes world knowledge as feature vectors used by a ranker trained with Joachims' SVM^{light} package [23]. The features are calculated on the basis of Narrative Chains [24], Google queries, FrameNet [25], Heuristic and Machine-Learned Polarities, Connective-based relations, Semantic Compatibility, and Lexical Features [6]. This approach largely outperformed (+18%) other state-of-the-art approaches on the DPR dataset with an overall accuracy of 73.05%.

A inference-based approach, radically different from the previous one, was used in [26]. The authors extended Hobbs' weighted abduction [27], an abductive reasoning [28] technique that ranks candidate hypotheses explaining observations according to plausibility, to accommodate unification weights and to show how to learn these weights by applying Machine Learning (ML) [29] techniques. By doing so, they aimed at addressing the *overmerging problem* [30], that is, establishing wrong coreference links among entities. The Knowledge Bases (KBs) [31] used for inference were WordNet [32], FrameNet [33], and Narrative Chains [24]. However, the precision of their approach, enriched with Stanford NLP (SNLP) [17] output, resulted to be lower than that of SNLP alone on the employed datasets. This happened because adding world knowledge resulted in new coreference links, while the overmerging problem was not completely solved [34].

Finally, in [35] the authors presented a method for automatically acquiring instances of sentences that are similar to WSs but easier to disambiguate. By using the Stanford Dependency Parser (SDP) [36] to analyse the structure of a specific WS sentence, they generated a concise Google search query that captures its essential structure and then finds the alignments of the source sentence and its retrieved instances. The obtained results, however, were inferior to those achieved in [6], even if in [35] the method was tested on a reduced version of the same DPR dataset.

III. A Computational framework for Winograd Schema resolution

In our framework, we divide the WS resolution task in three sub-tasks.

1) The sentences in the schema at hand are analysed through standard Natural Language Processing (NLP) [37], [38] techniques to obtain their structures. Then, the components of these structures are translated into FOL expressions.
2) A context of commonsense knowledge relevant to the schema at hand is constructed by querying external KBs. We retrieve a subset of our KBs composed of concepts and relations among them. These concepts are connected to the concepts referred by the symbols of the predicates in the FOL expressions calculated at the previous step. The information contained in the context is translated into FOL. Optionally, the devised context is filtered using ML techniques to ensure that only the most relevant information is added to the set of FOL relations.

[2] http://www.hlt.utdallas.edu/~vince/data/emnlp12/. Accessed 04 June 2015.
[3] https://www.essence-network.com. Accessed 04 June 2015.

3) The combination of the FOL relations derived from the schema and the context constitute the input of a deductive reasoner, such as an Automatic Theorem Prover (ATP) [39]. For each of the two twin sentences, the ATP is run successfully if it is able to prove that one of the two candidate noun phrases is true while, at the same time, the other is false.

While the first sub-task is carried out only once, the two last ones can be repeated multiple times if the output of the third sub-task is not correct. In this case, the output of a specialised software that searches for counterexamples can optionally be added to the input of sub-task two. This software is usually contained within ATP software packages.

The following sections deal with each of the three sub-tasks.

IV. From Natural Language to First Order Logic

The first sub-task we carry out is a two-step process that translates a WS schema into a FOL form. While we are aware that FOL cannot fully express the structural variability contained in natural language, we consider it a good compromise between expressiveness and computability. First, we analyse the sentences using the Stanford CoreNLP Toolkit [40]. This is a comprehensive suite that takes one or more sentences as input and performs a wide range of NLP tasks, including, but not limited to, part-of-speech (POS) tagging, Named Entity Recognition (NER), and Dependency Parsing (DP). Subsequently, we automatically translate the schema into FOL using the SNLP output and WS-related information. These two steps are explained in the next two sub-sections.

A. Natural Language Processing

To translate a schema into FOL, we first need to gather as much structured information as possible from the sentences through performing POS tagging, Stemming, and DP analysis on them. Let us take as an example one of the two twin sentences of the first schema in the DPR dataset, shown in Fig. 1. The POS tagging and Stemming, the dependencies, and the parse tree (obtained using a Context-Free Grammar) of this schema are shown in Tables I, II, and Fig. 2, respectively. An introduction to the POS tags is given in the Penn Treebank [41] while a description of the dependency roles used by SNLP can be found in the Stanford Dependencies manual [42].

Sentence: `The bee landed on the flower because it had pollen.`
Pronoun: `it`
Candidates: `The bee/the flower`
Correct candidate: `the flower`

Fig. 1: One of the two twin sentences of a schema.

Word	**POS**	**Stemmed Word**
The	DT	the
bee	NN	bee
landed	VBD	land
on	IN	on
the	DT	the
flower	NN	flower
because	IN	because
it	PRP	it
had	VBD	have
pollen	NN	pollen
.	.	.

TABLE I: POS tagging and Stemming of the WS in Fig. 1.

Role	**Word**	**Depends On**
root	landed	ROOT
det	The	bee
nsubj	bee	landed
det	the	flower
prep_on	flower	landed
mark	because	had
nsubj	it	had
advcl	had	landed
dobj	pollen	had

TABLE II: DP of the WS in Fig. 1.

B. Representation in First Order Logic

We translate sentences in FOL by analysing SNLP DP output along with the WS structure. We first identify the two candidates. For each of them, we create constants, e.g., C_1 and C_2. Then, for each of them, we create a predicate whose symbol is the stemmed version of the word identifying the candidate (we ignore tenses to simplify the reasoning). In the example of Fig. 1, we would have the expressions `bee(B)` and `flower(F)`. In FOL the symbols standing for predicates do not have intrinsic meaning but they will be linked to relations in the KB (see Sec. V). Since we know from the WS resolution task that the two candidates are distinct entities, we also know that a candidate predicate does not apply to the other's constant. Therefore, we add to our list of assumptions two appropriate negated formulas, in this example, `-bee(F)` and `-flower(B)`. For the same reason, we add an expression representing that either the target pronoun is equal to the first candidate and not to the second, or the other way around. In this example,

```
(IT = B & IT != F) | (IT = F & IT != B).
```

Then, we scan the list of dependencies provided by the DP. Each noun in the list is represented in the same way we did for the two candidates, including the negated formulas. All these predicates are unary. The pronoun is represented by just a constant; we do not instantiate a predicate for it.

Each verb is represented using a predicate whose arguments are the constants defined for the relevant nouns. The order of the arguments in the FOL expression is subject, direct object (if any), and other complements (if any). The symbol of the predicate is the stemmed version of the verb with the exception of copulas, for which we use the subject complement (e.g., `hungry(wolf)` for the proposition "The wolf is hungry"). The arguments of verb predicates are the constants corresponding to each noun in the sentence and/or the pronoun. In this example, the proposition "it had pollen" is represented by the expression `have(IT, P)`. For each expression in which the target pronoun is included, we similarly add a disjunctive

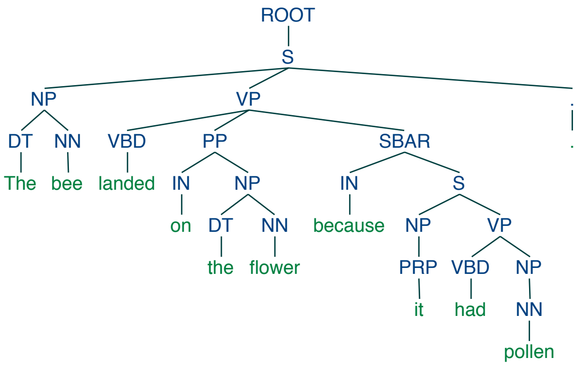

Fig. 2: The parse tree of the WS in Fig. 1 obtained with a Context-Free Grammar.

```
bee(B)
-bee(F)
flower(F)
-flower(B)
( IT = B & IT != F ) | ( IT = F & IT != B )
pollen(P)
-bee(P)
-flower(P)
land(B, F)
have(IT, P)
( have(B, P) & -have(F, P) ) | (have(F, P) & -have(B, P) )
have(IT, P) -> land(B, F)
```

Fig. 3: The list of expressions derived from the sentence and the schema structure.

expression in which the constant of the target pronoun is substituted by one of the candidates. In the example at hand, this would be the expression

```
(have(B, P) & -have(F, P)) | (have(F, P) & -have(B, P)).
```

The next step is to deal with relations among clauses in the sentence.[4] Then, we add an expression causally relating the two propositions. In the example at hand, that would be `have(IT, P) -> land(B, F)`. The final list of expressions derived from the sentence and the schema structure is shown in Fig. 3.

Since many WS sentences include proper names of people, many of which are not included in KBs, for each word recognised by SNLP as a proper name (`NNP` tag), we check it against a list of English names. If we are able to find the corresponding name in the list, we add the predicate `person(C)` to our assumption list, where `C` is the constant associated to that word (e.g., `Robert(R), person(R)`).

In the case of nominal constructions, such as "bus driver", we recognise them as a single entity and, therefore, we construct a single predicate out of them, such as `bus_driver`. However, we keep track of the individual components (e.g., "bus" and "driver"). In particular, we add an expression where the predicate of the root component (e.g., "driver") has the same arguments of the nominal construction predicate. For instance, for the proposition "The bus driver", we would have `bus_driver(B)` and `driver(B)`.

Finally, we define two goal expressions. In one of these, the target pronoun is set equal to one of the two candidates, in the other one, the target pronoun is equal to the other candidate. The ATP will have to prove one of these expressions true, while at the same time, prove the other one false. In the example examined so far, they are `IT = B` and `IT = F`, that in our example from Fig. 1 should evaluate as false and true, respectively.

V. Knowledge Bases

To construct an interpretation of the FOL symbols obtained in the steps described above, we draw upon information from external sources. A wide range of KBs are used in the literature. While we plan to use several of them in the future, in this preliminary version of our framework, we only use ConceptNet 5[5] (CN) [43].

CN is a multilingual KB representing words and phrases used by humans, and commonsense relationships between them. The knowledge in CN is collected from a variety of sources, including crowd-sourced KBs (such as Wiktionary and Open Mind Common Sense), games with a purpose (such as Verbosity and nadya.jp), and expert-created resources (such as WordNet and JMDict).

CN has a graph-based structure, as it is a network of labelled nodes and edges plus additional supporting information about these nodes and edges. The nodes, or concepts, are words, word senses, and short phrases, in a number of different languages.[6] The edges are pieces of common-sense knowledge that connect these concepts to each other with a particular relation. Each edge comes from one or more knowledge sources. The source also assigns a weight to the edge, indicating how important and informative that edge should be, and possibly a surface text showing how this fact of common-sense knowledge was originally expressed in natural language.

CN has several types of relations that were chosen to capture common, informative patterns from the different data sources. All these relations can be prefixed with `Not` to express a negative relation. Table III lists the most common relations in CN. CN has been used in several semantic-related works, such us, for instance [44]–[46].

To ease the generation of a context through CN, we loaded this KB into an instance of Neo4j [47], a flexible, fast, and scalable graph database [48]. The use of a graph database allows us to construct the context by using a wide range of techniques, from the simple approach of calculating the shortest path between two concepts, to more advanced strategies such as Spreading Activation [49], a method for searching associative networks, neural networks, or semantic networks.

[4] In this preliminary version we only consider causal relations, of which we identify antecedent and consequent.

[5] http://conceptnet5.media.mit.edu/. Accessed 04 June 2015.

[6] In our case, however, we only use concepts and relations in English.

Relation	Description and Examples
`RelatedTo`	There is some positive relationship between A and B, but it is undetermined.
`IsA`	A is a subtype or a specific instance of B; every A is a B. `IsA car vehicle`
`PartOf`	A is a part of B. `PartOf gearshift car`
`MemberOf`	A is a member of B; B is a group that includes A.
`HasA`	B belongs to A. HasA is often the reverse of PartOf. `HasA bird wing`
`UsedFor`	A is used for B; the purpose of A is B. `UsedFor bridge cross_water`
`CapableOf`	Something that A can typically do is B. `CapableOf knife cut`
`AtLocation`	A is a typical location for B. `AtLocation butter refrigerator`
`Causes`	A and B are events, and it is typical for A to cause B.
`HasSubevent`	A and B are events, and B happens as a subevent of A.
`HasFirstSubevent`	A is an event that begins with subevent B.
`HasLastSubevent`	A is an event that concludes with subevent B.
`HasPrerequisite`	B is a dependency of A. `HasPrerequisite drive get_in_car`
`HasProperty`	A has B as a property. `HasProperty ice solid`
`MotivatedByGoal`	A is a step toward accomplishing the goal B.
`ObstructedBy`	B is an obstacle in the way of A.
`Desires`	A is a conscious entity that typically wants B. `Desires person love`
`CreatedBy`	B is a process that creates A. `CreatedBy cake bake`
`Synonym`	A and B have very similar meanings.
`Antonym`	A and B are opposites in some relevant way. `Antonym black white`
`DerivedFrom`	A is a word or phrase that appears within B and contributes to B's meaning. `DerivedFrom pocketbook book`
`DefinedAs`	B is a more explanatory version of A.

TABLE III: Some common CN relations.

A. *Context Generation*

To generate an appropriate context for the schema at hand, we start by selecting a set of Base Terms (BTs) derived from the FOL representation of the WS. The set of BTs consists of the concepts in CN corresponding to the symbols of all the predicates generated as described in Sec. IV-B. We also include all the components of nominal constructions (e.g., `bus` and `driver` along with `bus_driver`). In addition, for each predicate derived by verbs, we also add to the BTs the symbol with the added suffix `/v`, which in CN represents a concept that is a verb. Since CN is unfortunately affected by noise,[7] we also keep the standard BT version, as many meaningful relations concerning verbs concept are still included in the base concept. For the example in Fig. 1, the generated BTs correspond to the set `(pollen, bee, flower, have, have/v, land, land/v)`.

In this preliminary version of our framework, we generate the context by calculating all the shortest paths within CN up to a length l_p between all the pairs of concepts in the BTs set. By doing so, the generated context is made up of the BTs along with all the nodes and relations connecting them. Finally, we also retrieve all the relations between all the pairs of concepts within the concepts retrieved at the previous step. The final context is defined as the subgraph composed by all the nodes, each of which stands for a different concept, and relations retrieved so far. Fig. 4 shows the context subgraph calculated for the schema of Fig. 1 with $l_p = 1$.

[7]CN is affected by noise because some of its relations are created by automatic, NLP-based tools, which fail in some cases.

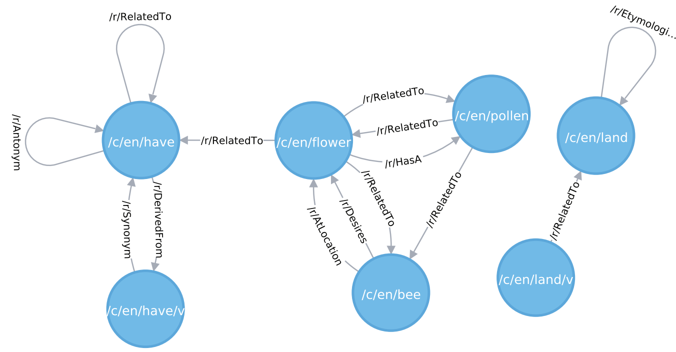

Fig. 4: The context generated for the WS in Fig. 1 with $l_p = 1$. Concept and relation prefixes come from the CN namespace.

VI. Reasoning

In the previous section, we generated a graph-structured context starting from the FOL representation of the WS at hand. Now, we translate this KB information to a FOL representation. To do so, we iterate the list of relations in the context subgraph and translate them into FOL, one by one. The expressions obtained are then added to the list of assumptions of the ATP.

Among the relation types in Table III, we only translate six in this preliminary version of our framework: `IsA`, `HasA`, `Desires`, `MotivatedByGoal`, `CapableOf`, and `HasProperty`. In general, different FOL representations are used for different types of relations. Table IV show the FOL formalisation of the six considered relation types.

A key point is that we have to take into account the arities of the predicates we translate into FOL. With this aim, we keep track of this property and translate predicates accordingly. Arities are kept fixed on a per-schema basis, avoiding unnecessary global constraints. Regardless of the number of arguments of a predicate, the first argument will always be the subject while the second corresponds to the object. For instance, the predicate `eat` for the relation type `MotivatedByGoal` in Table IV has two arguments, the subject and the object. This is so because in this example, coming from a WS in our dataset, the verb "to eat" is instantiated in transitive form.[8]

In four cases (`IsA`, `MotivatedByGoal`, `CapableOf`, and `HasProperty`) the subject arguments of the two predicates are the same. In the remaining two (`HasA`, `Desires`), they are different because the two concepts refer to two different entities. In some cases, an argument of a predicate might be an activity, like in the relation `Desires animal eat`. This would be translated to

```
all x (animal(x) -> want(x, eat(x,y))).
```

However, this is not possible in FOL, which does not allow quantification over functions. In such cases, we ignore the outer predicate and use the inner one as the consequent of the expression. Therefore, the previous example would be translated to

```
all x (animal(x) -> eat(x,y)).
```

[8]The verb "to eat" is actually an ambitransitive verb, i.e., it ca be used in an intransitive and transitive way.

```
all x all y (flower(x) & pollen(y) -> have(x, y))
all x all y (bee(x) & flower(y) -> want(x, y))
```

Fig. 5: The FOL form of the relations extracted from the context of Fig. 4.

Considering the example WS of Fig. 1 and the context shown in Fig. 4, the FOL forms that we extract from the context are shown in Fig. 5.

A. *Reasoner*

Prover9 [50] is the reasoner that we use to solve schemas. Prover9 is an ATP for first order and equational logic, and Mace4 is a specialised software that searches for finite models and counterexamples.

The primary mode of inference used by Prover9 is resolution [51]. It repeatedly makes resolution inferences with the aim of detecting inconsistencies by deriving a contradiction. First it negates the formula given as a goal. It then translates all formulae into clausal form, that is, a form composed of a conjunction of clauses, where a clause is a disjunction of literals. Then it computes inferences. If it detects an inconsistency, it will stop and output a proof consisting of the sequence of resolution rules that generated the inconsistency.

In case we fail to prove one of the two target expressions, we can run Mace4 to get counterexamples. This information could be used to guide the search of new relations or to change the FOL translation of already gathered ones. Considering the schema of Fig. 1, if we do not add the expressions of Fig. 5, Prover9 cannot prove the target formula `IT = F` as true. In this case, the output of Mace4 would be the one shown in Fig. 6.

VII. Results and Analysis

Since we only present a preliminary version of our framework, we do not carry out an extensive experimentation in this paper. However, we show the results obtained over a few examples and we comment on the promising results while indicating where there is room for improvement.

Table V shows the 14 experiments we conducted over six schemas with $l_p \in [1, 2]$, along with the number of relations in the context (n_r), the number of those added to the assumptions list (n_a), and the obtained results. The schemas are part of the training set of the DPR dataset [6]. For these experiments, we considered each twin sentence of a WS as a separate problem. This makes the task harder because while analysing a sentence we do not take advantage of the outcome of the reasoning over the other one. In fact, if we were able to solve one of the two sentences, we would automatically solve the other one as well by simply selecting the other candidate.

```
pollen set([('c',)])
flower set([('b',)])
F b
IT a
bee set([('a',)])
P c
B a
have set([('a', 'c')])
land set([('a', 'b')])
```

Fig. 6: The Mace4 output given the expressions of Fig. 3 for the target expression `IT = F`.

In experiments 1 and 2 we addressed schema 0, the one from Fig. 1. We were able to solve it successfully with $l_p \in [1, 2]$. Meanwhile, experiments 3 and 4 over schema 2 failed. While relations `Desires bee flower` and `HasA flower pollen` are in CN, we would need an expression such as

```
all x all y all z (have(x,y) & want(z,x) -> want(z,y))
```

to infer that since a bee wants a flower, it also wants the pollen contained in it. Unfortunately this ternary relation cannot be expressed in CN and we would need to add other KBs to our framework, such as FrameNet, to address this problem. Experiments 5 and 6 over schema 3 also failed, but for a different reason. In this case, in CN we found the relation `RelatedTo wet splash` with its surface text being `[[wet]] is related to [[splashed]]`. However, currently we do not deal with this kind of general relations in our framework. It is worth noting that we could infer the passive role of the one being splashed by applying NLP on this text so that we would be able to infer that those who were splashed stand as the direct object of the predicate `splash` and not, for instance, the subject.

Experiments 7 and 8 on schema 17 were successful as we correctly identified that it was the bird that sang, through the relation `CapableOf bird sing`. Interestingly, while experiment 9 was successful over schema 210 with $l_p = 1$, experiment 10 with $l_p = 2$ failed. In the former case, we solved the WS through adding the relation `MotivatedByGoal eat hungry`. On the other hand, with $l_p = 2$ we also added relations `Desires animal eat`, `IsA wolf animal`, and `IsA cow animal` which make the ATP prove both target expressions true. This suggests that we should add a filtering mechanism to our framework, as delineated in Sec. VIII.

Experiments 11 and 12 failed on schema 211. In these cases, we found the relations `HasProperty beef delicious`, and `RelatedTo beef cow`. However, currently we do not deal with the latter type of relation (see Sec. VIII).

Experiments 13 and 14 over schema 150 show how setting $l_p > 1$ is of key importance in some cases. While experiment 13 failed with $l_p = 1$, experiment 14 was successful with $l_p = 2$ as the system correctly added the relations `IsA student person` and `HasProperty person lazy`.

Finally, Table VI shows the values of n_r and n_a for $l_p \in [1, 5]$ for schema 211. We noted how the context size and the number of translated expressions increase quickly with l_p. This confirms the need for filtering relations and suggests that we might need more sophisticated strategies for generating the context, such as Spreading Activation.

All in all, the shown results are promising as we showed how it is possible to solve some simple WSs with our framework. In the next section, we cover the modifications we should implement to improve the accuracy of our method.

Relation	FOL Form	Examples
IsA	all x (conceptA(x) -> conceptB(x))	all x (cow(x) -> animal(x))
MotivatedByGoal	all x (conceptA(x) -> conceptB(x))	all x (eat(x, y) -> hungry(x))
CapableOf	all x (conceptA(x) -> conceptB(x))	all x (animal(x) -> drink(x))
HasProperty	all x (conceptA(x) -> conceptB(x))	all x (person(x) -> lazy(x))
HasA	all x all y (conceptA(x) & conceptB(y) -> have(x, y))	all x all y (bus(x) & window(y) -> have(x, y))
Desires	all x all y (conceptA(x) & conceptB(y) -> want(x, y))	all x all y (dog(x) & food(y) -> want(x, y))

TABLE IV: FOL translations and examples of six common CN relations.

E	ID	Sentence	P	Candidates	l_p	n_r	n_a	Result
1	0	The bee landed on the flower because it had pollen.	it	The bee/**the flower**	1	14	2	✓
2	0	The bee landed on the flower because it had pollen.	it	The bee/**the flower**	2	694	36	✓
3	1	The bee landed on the flower because it wanted pollen.	it	**The bee**/the flower	1	11	2	
4	1	The bee landed on the flower because it wanted pollen.	it	**The bee**/the flower	2	364	21	
5	3	When Debbie splashed Tina, she got wet.	she	Debbie/**Tina**	1	6	0	
6	3	When Debbie splashed Tina, she got wet.	she	Debbie/**Tina**	2	200	30	
7	17	The bird perched on the limb and it sang.	it	**The bird**/the limb	1	9	2	✓
8	17	The bird perched on the limb and it sang.	it	**The bird**/the limb	2	134	10	✓
9	210	The wolves ate the cows because they were hungry.	they	**The wolves**/the cows	1	7	1	✓
10	210	The wolves ate the cows because they were hungry.	they	**The wolves**/the cows	2	105	17	
11	211	The wolves ate the cows because they were delicious.	they	The wolves/**the cows**	1	5	0	
12	211	The wolves ate the cows because they were delicious.	they	The wolves/**the cows**	2	117	18	
13	150	Students hate exams because they are lazy.	they	**Students**/exams	1	2	1	
14	150	Students hate exams because they are lazy.	they	**Students**/exams	2	57	19	✓

TABLE V: Results on 14 experiments over a small set of schemas. E is the experiment number, ID refers to the DPR dataset. P represents the target pronoun and the correct candidate is highlighted in boldface. l_p is the maximum length of paths among BTs. n_r is the number of relations in the returned context, n_a is the number of relations added to the assumptions list. A ✓ sign indicates that the experiment was successful.

l_p	n_r	n_a
1	5	0
2	117	18
3	512	95
4	1139	212
5	1389	267

TABLE VI: Values of n_r and n_a for $l_p \in [1, 5]$ for WS 211.

VIII. Future Work

In this section we give an overview of the improvements and extensions that we plan to introduce to each of the components of our framework separately.

A. From Natural Language to First Order Logic

The translation from natural language into FOL is a very critical sub-task which all subsequent ones depend on. Currently, we are not able to correctly express all the schemas in the DPR dataset. In fact, some schemas, especially longer ones, use complex syntactical constructions and subordination dependencies that we cannot yet deal with. Therefore, we plan to complete the implementation of our framework to accommodate, at least in principle, the translation of the remaining cases.

As an alternative to our translation mechanism, we could rely on the English semantic parser Boxer [52], used in [26]. Boxer is a software component for semantic analysis of text, based on Combinatory Categorial Grammar [53] and Discourse Representation Theory (DRT) [54]. Boxer output can be translated into FOL formulas and be processed by standard ATPs for FOL.

Another alternative would be the use of the Enju[9] parser [55], based on Head-driven Phrase Structure Grammar [56]. It performs well in capturing long-distance and unbounded dependencies in language while being able to output both phrase structures and predicate-argument structures.

B. Knowledge Bases

KBs represent external sources of information that we need to ground symbols in our FOL representations. While CN is a comprehensive resource with a significant amount of commonsense knowledge, it is far from perfect. In many cases, the relations we found were irrelevant or simply wrong.[10] In other cases, we were not able to find information on specific facts or entities (Abox information). For instance, one of the schemas in the DPR dataset reads “Americans preferred Obama to McCain because he was younger”. This kind of information is not contained within CN. Therefore, we plan to expand our framework to also take into account KBs other than CN.

Among the numerous KBs, DBpedia [57] and Freebase [58] represent attractive alternatives. DBpedia is a crowd-sourced community effort to extract structured information from Wikipedia and make this information available on the Web. Freebase is a large, structured, and collaborative KB consisting of data mainly gathered by its community members by harvesting from many sources, including individual, user-submitted wiki contributions. In both cases, we will have to import these resources in our graph database and develop procedures to translate information contained therein into FOL. To avoid conflicts with the existing CN concepts, we will use

[9] http://www.nactem.ac.uk/enju/. Accessed 04 June 2015.

[10] An example of wrong relation would be `Causes cook_dinner hungry`, which should be the other way around.

different namespaces for the two KBs.

Context generation is another procedure that we plan to improve. Instead of generating one large context by calculating all the shortest paths up to length l_p among all the pairs of BTs, we could build it iteratively. In fact, by executing sub-tasks 2 and 3 iteratively, we could start with $l_p = 1$, run the reasoner and, in case of failure, expand the context only towards meaningful or promising directions, taking as suggestion the counterexamples provided by Mace4.

On a different level, Spreading Activation [49] could be an alternative to the strategy of calculating the shortest paths, as it was extensively used as a technique in information retrieval [59].

C. Reasoning

Regarding our third sub-task, the first important improvement will be to complete the translation into FOL of the remaining types of relations in CN. Among these, dealing with `RelatedTo` relations is a challenging problem. In fact, these relations, which are the second most common ones in CN, define only a loose coupling between the two concepts, without explicitly expressing the kind of relation connecting them. This is rather unfortunate, since they cannot be translated into FOL in a unique way, as other relations do. Since translating `RelatedTo` relations into FOL involves a substantial number of possible combinations of input, we plan to use a ML-based approach, such as a classifier, to first translate `RelatedTo` relations to more meaningful CN relation types such as `IsA` or `HasA`. Then, we could translate it into FOL using the fixed, manually defined translations to FOL shown in Table IV.

In the planned approach, the output classes will be constituted by all the relation types in CN except `RelatedTo`. The input will be constituted by appropriate representations of the two concepts in the relation, the weight associated to the relation, the source dataset, and the surface text. In fact, since the surface text is not stemmed, it will be possible to analyse this text with SNLP and infer some relevant properties from the syntactic parsing to use them as features. Formally, the input vector v_i associated to instance i will be codified as

$$v_i = [c_{1_1}..c_{1_n}, c_{2_1}..c_{2_n}, h, d, p_1, p_2] \quad (1)$$

where h is the relation weight, d is the source dataset, and p_x is the POS tag for concept x as derived by the SNLP parser on the surface text. d and p_x can be easily represented numerically using a look-up table. Differently, $c_{x_1}..c_{x_n}$ are the components of the vector representation of the English word that points to a concept in CN. Several approaches were researched in the literature to express words using numeric, fixed-length (n) vectorial representations. Among them, GloVe [60] is particularly attractive because the resulting representations showcase interesting linear substructures of the word vector space. In this case, we will simply use the freely available[11] vector representation for word x to obtain $[c_{x_1}..c_{x_n}]$.

To train the classifier we could use the very large set of relations with types other than `RelatedTo` already in CN. After splitting this set into train, test, and validation sets, we will train the classifier using the same input representation and as target label the actual relation type found in CN. Were this approach successful, a remarkable by-product will be the substantial improvement of CN also for different tasks.

Filtering the relations in the context, that is, choosing which ones to translate into FOL and insert into our list of assumptions, is another interesting problem that we plan to tackle using ML. In this case, the output will be binary, that is, the output classes will represent a decision about whether to insert a relation in the assumptions list or not. Different sets of information could constitute the input of the classifier. In the simplest case, the input vector will be the one described in Eq. 1 concatenated with the relation type t. More comprehensive input will imply adding to v_i (see Eq. 2) also some contextual information such as the list of BTs (represented as vectors using GloVe). Formally, the input vector v_i associated to instance i will be codified as

$$v_i = [G(c_1), G(c_2), h, d, p_1, p_2, t, G(b_1)..G(b_k)] \quad (2)$$

where $G(w) = [w_{1_1}..w_{1_n}]$ is the GloVe vector representation of length n of word w and b_i is the i-th of the k BTs. It is worth noting that in this case we will lose the assumption of fixed-length input, as k changes on a per-schema basis. Therefore, we will need to use a classifier that is able to handle arbitrary sequences of input, such as recurrent neural networks [61]. A simpler alternative will be to limit the BTs included as input to a fixed-length window containing the first j BTs (or those more recently added to the list of assumptions) and use, for instance, a FeedForward Neural Network.

The training of the classifier, however, will be challenging since a target label will not be immediately available. Actually, in general, several relevant relations need to be added in FOL-form to the list of assumptions before the ATP output changes and all of them might be needed to solve the schema. A solution to this problem could be to employ Reinforcement Learning (RL) [62] to train the classifier. Recently, a Deep Neural Network trained with RL was able to achieve human-level control [63] on the challenging domain of playing classical Atari 2600 games.

To apply RL to the training process of a classifier, it is necessary to define a reward function that is correlated, at least in the long run, with the actions performed in an environment by the software agent, that is, the classifier. In our case, the environment will be constituted by the input described in Eq. 2 and the actions will correspond to the outcomes of the binary decision of admitting a CN relation to the list of assumptions. We could define the reward function as follows. Typically, without inserting any CN relation, the two expressions that the ATP tries to prove result false. Let us define these two expressions as $P = C_x$, where P and $C_x, x \in (1, 2)$ are the constants representing the target pronoun and the two candidates, respectively. We can now define a four-valued reward function as

$$F_{i\in[1,4]} = [R_1, R_2], R_x \in (\text{True}, \text{False}), x \in (1, 2), \quad (3)$$

where R_x is True if the ATP result is correct for $P = C_x$ and False otherwise. There are only 4 reward values in our reward function. Since a potential problem could be its poor granularity, we could expand the reward function by also taking into account the output of Mace4.

An alternative approach to RL will imply the ability of the classifier to provide several output decisions at the same time,

[11] http://nlp.stanford.edu/projects/glove/. Accessed 04 June 2015.

one for each relation in the context subgraph. This learning model should also be able to take as input the entire context, which is a (sub-)graph. A model which is able to do this is called Graph Neural Network (GNN) [64], [65].

GNNs extend existing NNs methods for processing data represented in a graph domain. The GNN model, which can directly process acyclic, cyclic, directed, and un-directed graphs, implements a transduction function $\tau(G, n) \in \mathbb{R}^m$ that maps a graph G and one of its nodes n into an m-dimensional Euclidean space. GNNs are suitable for both node-focused and graph-focused applications. In node focused applications, the function τ depends on the node n, so that the classification depends on the properties of each node [64]. The intuitive idea underlying the GNN approach is that nodes in a graph represent objects or concepts, and edges represent their relationships. Each concept is naturally defined by its features and the related concepts. Thus, a state $x_n \in \mathbb{R}^s$ is attached to each node n and is based on the information contained in the neighbourhood of n. The variable x_n contains a representation of the concept denoted by n and can be used to produce an output o_n, i.e., a decision about the concept [64]. Operatively, GNNs use and train two Feed Forward NNs, one to calculate x_n and the other to calculate o_n.

GNNs could represent the most appropriate model for the task of filtering the context, and ultimately making a decision about what information is relevant for a schema, as they can deal with most types of information, explicit and implicit, that we have in the context. Individual features will be encoded as explained so far. However, as we are interested in making a decision for each *edge* in the graph and not for each *node*, we will have to take as input the line graph [66] of our context graph. Finally, before introducing GNNs in our framework, we will have to carry out a careful analysis of the computational cost involved, especially considering the significant size of the context we have to deal with, which can reach thousands of relations.

IX. Conclusion

This paper introduced a framework for the solution of WSs based on NLP, FOL theorem proving, the CN semantic network, and graph databases. Since the framework is still at an early stage of development, there is much room for improvement and the results obtained over a reduced set of schemas draw a consistent landscape of future work to adapt the framework to more complex scenarios.

Acknowledgment

This work is supported by the European Commission with the grant agreement No. 607062 (ESSENCE Marie Curie ITN, http://www.essence-network.com/).

References

[1] H. J. Levesque, E. Davis, and L. Morgenstern, "The winograd schema challenge." in *KR*, G. Brewka, T. Eiter, and S. A. McIlraith, Eds. AAAI Press, 2012. [Online]. Available: http://dblp.uni-trier.de/db/conf/kr/kr2012.html#LevesqueDM12

[2] T. Winograd, "Understanding natural language," *Cognitive Psychology*, vol. 3, no. 1, pp. 1 – 191, 1972. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0010028572900023

[3] E. Tognini-Bonelli, *Corpus Linguistics at Work*, ser. Studies in corpus linguistics. J. Benjamins, 2001. [Online]. Available: http://books.google.co.uk/books?id=6YDRH45MpL8C

[4] H. J. Levesque, "On our best behaviour," *Artificial Intelligence*, vol. 212, no. 0, pp. 27 – 35, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0004370214000356

[5] E. Mendelson, *Introduction to Mathematical Logic*, 5th ed. Chapman & Hall/CRC, 2009.

[6] A. Rahman and V. Ng, "Resolving complex cases of definite pronouns: The winograd schema challenge," in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, ser. EMNLP-CoNLL '12. Stroudsburg, PA, USA: Association for Computational Linguistics, 2012, pp. 777–789. [Online]. Available: http://dl.acm.org/citation.cfm?id=2390948.2391032

[7] A. Radford, *Minimalist Syntax: Exploring the Structure of English*, ser. Cambridge Textbooks in Linguistics. Cambridge University Press, 2004. [Online]. Available: http://books.google.co.uk/books?id=y5VJLP-NW1gC

[8] I. A. Bolshakov and A. Gelbukh, *Computational Linguistics: Models, Resources, Applications*. Mexico City: Instituto Politcnico Nacional, 2004.

[9] L. Qiu, M. Kan, and T. Chua, "A public reference implementation of the RAP anaphora resolution algorithm," *CoRR*, vol. cs.CL/0406031, 2004. [Online]. Available: http://arxiv.org/abs/cs.CL/0406031

[10] M. Poesio and M. Kabadjov, "A general-purpose, off-the-shelf anaphora resolution module: Implementation and preliminary evaluation," in *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC)*, Lisbon, Portugal, 2004.

[11] R. Mitkov, R. Evans, and C. Orasan, "A new, fully automatic version of mitkov's knowledge-poor pronoun resolution method." in *CICLing*, ser. Lecture Notes in Computer Science, A. F. Gelbukh, Ed., vol. 2276. Springer, 2002, pp. 168–186. [Online]. Available: http://dblp.uni-trier.de/db/conf/cicling/cicling2002.html#MitkovEO02

[12] E. Charniak and M. Elsner, "Em works for pronoun anaphora resolution," in *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, ser. EACL '09. Stroudsburg, PA, USA: Association for Computational Linguistics, 2009, pp. 148–156. [Online]. Available: http://dl.acm.org/citation.cfm?id=1609067.1609083

[13] Y. Versley, S. P. Ponzetto, M. Poesio, V. Eidelman, A. Jern, J. Smith, X. Yang, and A. Moschitti, "Bart: A modular toolkit for coreference resolution," in *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Demo Session*, ser. HLT-Demonstrations '08. Stroudsburg, PA, USA: Association for Computational Linguistics, 2008, pp. 9–12. [Online]. Available: http://dl.acm.org/citation.cfm?id=1564144.1564147

[14] A. Rahman and V. Ng, "Supervised models for coreference resolution," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, ser. EMNLP '09. Stroudsburg, PA, USA: Association for Computational Linguistics, 2009, pp. 968–977. [Online]. Available: http://dl.acm.org/citation.cfm?id=1699571.1699639

[15] V. Stoyanov, C. Cardie, N. Gilbert, E. Riloff, D. Buttler, and D. Hysom, "Reconcile: A coreference resolution research platform," in *Proceedings of the ACL 2010 Conference Short Papers*, 2010, pp. 156–161.

[16] K. Raghunathan, H. Lee, S. Rangarajan, N. Chambers, M. Surdeanu, D. Jurafsky, and C. Manning, "A multi-pass sieve for coreference resolution," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, ser. EMNLP '10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 492–501. [Online]. Available: http://dl.acm.org/citation.cfm?id=1870658.1870706

[17] H. Lee, Y. Peirsman, A. Chang, N. Chambers, M. Surdeanu, and D. Jurafsky, "Stanford's multi-pass sieve coreference resolution system at the conll-2011 shared task," in *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, ser. CONLL Shared Task '11. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011, pp. 28–34. [Online]. Available: http://dl.acm.org/citation.cfm?id=2132936.2132938

[18] S. P. Ponzetto and M. Strube, "Exploiting semantic role labeling, wordnet and wikipedia for coreference resolution," in *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, ser. HLT-NAACL '06. Stroudsburg, PA, USA: Association for Computational Linguistics, 2006, pp. 192–199. [Online]. Available: http://dx.doi.org/10.3115/1220835.1220860

[19] ——, "Knowledge derived from wikipedia for computing semantic relatedness," *Journal of Artificial Intelligence Research*, vol. 30, pp. 181–212, 2007.

[20] V. Bryl, C. Giuliano, L. Serafini, and K. Tymoshenko, "Using background knowledge to support coreference resolution." in *ECAI*, ser. Frontiers in Artificial Intelligence and Applications, H. Coelho, R. Studer, and M. Wooldridge, Eds., vol. 215. IOS Press, 2010, pp. 759–764. [Online]. Available: http://dblp.uni-trier.de/db/conf/ecai/ecai2010.html#BrylGST10

[21] A. Rahman and V. Ng, "Coreference resolution with world knowledge," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, ser. HLT '11. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011, pp. 814–824. [Online]. Available: http://dl.acm.org/citation.cfm?id=2002472.2002575

[22] O. Uryupina, M. Poesio, C. Giuliano, and K. Tymoshenko, "Disambiguation and filtering methods in using web knowledge for coreference resolution," 2011. [Online]. Available: http://aaai.org/ocs/index.php/FLAIRS/FLAIRS11/paper/view/2614%40misc/3047

[23] T. Joachims, "Optimizing search engines using clickthrough data," in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '02. New York, NY, USA: ACM, 2002, pp. 133–142. [Online]. Available: http://doi.acm.org/10.1145/775047.775067

[24] N. Chambers and D. Jurafsky, "Unsupervised learning of narrative schemas and their participants," 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*, ser. ACL '09. Stroudsburg, PA, USA: Association for Computational Linguistics, 2009, pp. 602–610. [Online]. Available: http://dl.acm.org/citation.cfm?id=1690219.1690231

[25] C. F. Baker, C. J. Fillmore, and J. B. Lowe, "The berkeley framenet project," in *Proceedings of the 17th International Conference on Computational Linguistics - Volume 1*, ser. COLING '98. Stroudsburg, PA, USA: Association for Computational Linguistics, 1998, pp. 86–90. [Online]. Available: http://dx.doi.org/10.3115/980451.980860

[26] N. Inoue, E. Ovchinnikova, K. Inui, and J. Hobbs, "Coreference resolution with ilp-based weighted abduction," in *COLING 2012, 24th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, 8-15 December 2012, Mumbai, India*, 2012, pp. 1291–1308. [Online]. Available: http://aclweb.org/anthology/C/C12/C12-1079.pdf

[27] J. R. Hobbs, M. E. Stickel, D. E. Appelt, and P. Martin, "Interpretation as abduction," *Artif. Intell.*, vol. 63, no. 1-2, pp. 69–142, Oct. 1993. [Online]. Available: http://dx.doi.org/10.1016/0004-3702(93)90015-4

[28] L. Magnani, *Abduction, reason and science - Processes of discovery and explanation*. New York: Kluwer Academic, 2001.

[29] E. Alpaydin, *Introduction to Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2004.

[30] E. Ovchinnikova, *Integration of World Knowledge for Natural Language Understanding*, ser. Atlantis Thinking Machines. Atlantis Press, 2012, vol. 3. [Online]. Available: http://dx.doi.org/10.2991/978-94-91216-53-4

[31] R. Akerkar and P. Sajja, *Knowledge-Based Systems*, 1st ed. USA: Jones and Bartlett Publishers, Inc., 2009.

[32] C. Fellbaum, Ed., *WordNet: an electronic lexical database*. MIT Press, 1998.

[33] J. Ruppenhofer, M. Ellsworth, M. R. Petruck, C. R. Johnson, and J. Scheffczyk, *FrameNet II: Extended Theory and Practice*. Berkeley, California: International Computer Science Institute, 2006, distributed with the FrameNet data.

[34] NaoyaInoue, "Exploiting world knowledge indiscourse processing," Ph.D. dissertation, Graduate School of Information Science - Tohoku University, 1 2013.

[35] C. Kruengkrai, N. Inoue, J. Sugiura, and K. Inui, "An example-based approach to difficult pronoun resolution," in *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computing*, 2014, pp. 358–367. [Online]. Available: http://aclweb.org/anthology/Y14-1042

[36] M. Marneffe, B. Maccartney, and C. Manning, "Generating typed dependency parses from phrase structure parses," in *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC-2006)*. Genoa, Italy: European Language Resources Association (ELRA), May 2006, aCL Anthology Identifier: L06-1260. [Online]. Available: http://www.lrec-conf.org/proceedings/lrec2006/pdf/440_pdf.pdf

[37] D. Jurafsky and J. H. Martin, *Speech and Language Processing (2nd Edition) (Prentice Hall Series in Artificial Intelligence)*, 2nd ed. Prentice Hall, 2008. [Online]. Available: http://www.amazon.com/Language-Processing-Prentice-Artificial-Intelligence/dp/0131873210%3FSubscriptionId%3D13CT5CVB80YFWJEPWS02%26tag%3Dws%26linkCode%3Dxm2%26camp%3D2025%26creative%3D165953%26creativeASIN%3D0131873210

[38] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*, 1st ed. O'Reilly Media, Inc., 2009.

[39] A. Robinson and A. Voronkov, Eds., *Handbook of Automated Reasoning*. Amsterdam, The Netherlands, The Netherlands: Elsevier Science Publishers B. V., 2001.

[40] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2014, pp. 55–60. [Online]. Available: http://www.aclweb.org/anthology/P/P14/P14-5010

[41] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a large annotated corpus of english: The penn treebank," *COMPUTATIONAL LINGUISTICS*, vol. 19, no. 2, pp. 313–330, 1993.

[42] M.-C. de Marneffe and C. D. Manning. (2008, Sep.) Stanford typed dependencies manual. [Online]. Available: http://nlp.stanford.edu/software/dependencies_manual.pdf

[43] H. Liu and P. Singh, "Conceptnet — a practical commonsense reasoning tool-kit," *BT Technology Journal*, vol. 22, no. 4, pp. 211–226, Oct. 2004. [Online]. Available: http://dx.doi.org/10.1023/B:BTTJ.0000047600.45421.6d

[44] C. Havasi, R. Speer, J. Pustejovsky, and H. Lieberman, "Digital intuition: Applying common sense using dimensionality reduction." *IEEE Intelligent Systems*, vol. 24, no. 4, pp. 24–35, 2009. [Online]. Available: http://dblp.uni-trier.de/db/journals/expert/expert24.html#HavasiSPL09

[45] E. Erdem, E. Aker, and V. Patoglu, "Answer set programming for collaborative housekeeping robotics: representation, reasoning, and execution." *Intelligent Service Robotics*, vol. 5, no. 4, pp. 275–291, 2012. [Online]. Available: http://dblp.uni-trier.de/db/journals/isrob/isrob5.html#ErdemAP12

[46] D. I. Diochnos, "Commonsense reasoning and large network analysis: A computational study of conceptnet 4," *CoRR*, vol. abs/1304.5863, 2013. [Online]. Available: http://arxiv.org/abs/1304.5863

[47] J. Webber, "A programmatic introduction to neo4j," in *Proceedings of the 3rd Annual Conference on Systems, Programming, and Applications: Software for Humanity*, ser. SPLASH '12. New York, NY, USA: ACM, 2012, pp. 217–218. [Online]. Available: http://doi.acm.org/10.1145/2384716.2384777

[48] I. Robinson, J. Webber, and E. Eifrem, *Graph Databases*. O'Reilly Media, Inc., 2013.

[49] A. M. Collins and E. F. Loftus, "A spreading-activation theory of semantic processing." *Psychological Review*, vol. 82, no. 6, pp. 407 – 428, 1975. [Online]. Available: http://search.ebscohost.com/login.aspx?direct=true&db=pdh&AN=rev-82-6-407&loginpage=Login.asp&site=ehost-live

[50] W. McCune, "Prover9 and Mace4," 2005–2010. [Online]. Available: http://www.cs.unm.edu/~mccune/prover9/

[51] M. Davis and H. Putnam, "A computing procedure for quantification theory," *J. ACM*, vol. 7, no. 3, pp. 201–215, Jul. 1960. [Online]. Available: http://doi.acm.org/10.1145/321033.321034

[52] J. Bos, "Wide-coverage semantic analysis with boxer," in *Proceedings*

of the 2008 Conference on Semantics in Text Processing, ser. STEP '08. Stroudsburg, PA, USA: Association for Computational Linguistics, 2008, pp. 277–286. [Online]. Available: http://dl.acm.org/citation.cfm?id=1626481.1626503

[53] M. Steedman, *The Syntactic Process*. Cambridge, MA, USA: MIT Press, 2000.

[54] H. Kamp and U. Reyle, *Semantics: An International Handbook of Natural Language Meaning*. de Gruyter, 2011, ch. Discourse Representation Theory, pp. 872–919.

[55] Y. Tsuruoka, Y. Miyao, and J. Tsujii, "Towards efficient probabilistic hpsg parsing: integrating semantic and syntactic preference to guide the parsing," in *Proceedings of IJCNLP-04 Workshop: Beyond shallow analyses - Formalisms and statistical modeling for deep analyses*, Hainan Island, China, 2004.

[56] Y. Miyao and J. Tsujii, "Feature forest models for probabilistic hpsg parsing," *Computational Linguistics*, vol. 34, no. 1, p. 3580, March 2008.

[57] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer, "DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia," *Semantic Web Journal*, vol. 6, no. 2, pp. 167–195, 2015.

[58] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '08. New York, NY, USA: ACM, 2008, pp. 1247–1250. [Online]. Available: http://doi.acm.org/10.1145/1376616.1376746

[59] F. Crestani, "Application of spreading activation techniques in information retrieval," *Artificial Intelligence Review*, pp. 453–482, December 1997. [Online]. Available: http://www.ingentaconnect.com/content/klu/aire/1997/00000011/00000006/00100575

[60] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2014, pp. 1532–1543. [Online]. Available: http://aclweb.org/anthology/D14-1162

[61] M. Hermans and B. Schrauwen, "Training and analysing deep recurrent neural networks," in *Advances in Neural Information Processing Systems 26*, C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 190–198. [Online]. Available: http://papers.nips.cc/paper/5166-training-and-analysing-deep-recurrent-neural-networks.pdf

[62] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*, 1st ed. Cambridge, MA, USA: MIT Press, 1998.

[63] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 02 2015. [Online]. Available: http://dx.doi.org/10.1038/nature14236

[64] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *Neural Networks, IEEE Transactions on*, vol. 20, no. 1, pp. 61–80, Jan 2009.

[65] ——, "Computational capabilities of graph neural networks," *Neural Networks, IEEE Transactions on*, vol. 20, no. 1, pp. 81–102, Jan 2009.

[66] F. Harary and R. Norman, "Some properties of line digraphs," *Rendiconti del Circolo Matematico di Palermo*, vol. 9, no. 2, pp. 161–168, May 1960. [Online]. Available: http://dx.doi.org/10.1007/bf02854581