CS5012 Practical 2 : Grammar Engineering

120010815

March 2017

Introduction

The purpose of this practical is to familiarise ourselves with specifications of syntax. The goal of this practical is to create a grammar that accepts a certain subset of the English language.

A grammar accepts a set of rules defining different structures and terminals in a language. A terminal is, for example, a word, whether a structure would be, for example, a sentence. In a feature grammar, a structure or a terminal can be assigned a set of features such as number, gender, tense etc.

All the basic requirements as well as the extension requirements have been implemented for this practical. The grammar created simulates grammars that have been created by different research groups, and some of the research have been done to see how the task has been approached by these groups (such as The Stanford NLP Group) [2]. The notation used for the grammar mostly corresponds to the one used by Stanford but some changes have been made to make code more readable.

Another important note worth mentioning for this practical is that I am not a native English speaker, so the implementation might contain wrong assumptions about certain linguistic structures in English language, despite every effort taken to validate the structures in different resources.

Design and Implementation

The implementation of the grammar has been done in iterations, and there is some space left for further work. A sentence (S) can have three possible forms: it can be a statement ("I went for a walk"), a question ("When did you come back") and a sort of an inverted sentence that has usually been called "SBAR" in the existing tools but has a more general definition than I have in my grammar [4], [3]. This could also be generalised to contain other structures of sentences.

Statement and SBAR

A statement has a general form of NP(noun phrase) VP(verb phrase).

Noun phrase takes different forms with nominals, nouns, and proper nouns. A nominal is a noun preceded by a possible nominal. Notice the recursive structure in this case: it allows for words like "cotton shirt" to be evaluated as nouns but also creates ambiguity in cases like "I feed the kitten cheese" (since also the word 'feed' in included into two groups of verbs to allow both "I feed him cheese" and "I feed him").

Verb phrases take different forms, too, and use arguments received from the tree children to enforce the structure of the phrase depending on whether a verb is transitive or intransitive. A verb phrase can be preceded by an adverb (RB) or a modal verb (MD, such as 'would', 'can' etc.).

Nouns differentiate in number and person whereas verbs have number, tense and person.

An SBAR (again, this is not exactly a correct name for this structure as it is narrowed down but it seemed reasonable to use the name) contains a question word and two statements following it. It might not be exactly correct from the point of view of linguistics but the two statements are not connected grammatically, so it seemed reasonable to make them siblings in the grammar tree. Alternatively, this kind of sentence could have a form 'when'[SUBCAT=[HEAD=s, TAIL=[HEAD=s, TAIL=nil]]].

Question

A question (What cheese does Wallace eat) consists of a question phrase (what cheese) and the main part (does Wallace eat). The names of the structures, WhP and SQ, were borrowed from different linguistics papers [5], [6]. The structure of the Question might seem a bit redundant at this point (for example, there is no need to have a rule like "WhADVP ->WRB"), but this allows to add more structures later and maintains the structure similar to one that the Stanford Parser has [2]. The SQ contains an auxiliary verb, NP and VP.

Base forms of verbs

Despite the fact that the amount of verbal forms in English is quite limited and could be compressed to one feature (called FORM in the specification), I decided not to do that and to make it more syntactically correct. A verb has a number (sg/pl), person (although for this case, only 3rd person is used specifically), and tense (present simple - 'pres', present particle - 'prespart', past simple - 'past', and past particle - 'pastpart'). Since the FORM is defined by these three parameters, having it in the construct is redundant. Different sources distinguish between the concepts of forms and tenses in different ways, I decided to follow the OWL Writing Lab of the Purdue University [1]. One 'hack' was used for the verb forms: the base form of the verb is assumed to have plural form. To achieve agreement with pronouns 'I' is also assumed to be plural (Having first person pronouns was not in the specification of this practical).

Extension tasks

For the extension tasks, I implemented more structures. Gerund is implemented as a verb in the present particle form followed by an optional argument. Gerund can be an argument of a verb 'like'. Also, pronouns have been added, and the structure of the language has been formalised.

For the questions, the structure Question was considerably improved upon the version that used to be used before (and is not discussed in this report). See the description of the Question structure above. The current Question structure should allow for different types of questions, with or without an object, and with a wide variety of question subjects. In order to achieve that, the grammar differentiates between the "what" used as a question word with no subject (e.g. "what are you doing") — called WP in the practical — and "what" used as a question word with a subject (e.g. “what cheese does Gromit eat”). In a similar way, we distinguish the two usages of the word ‘when’ - as a question word (WRB in “what are you doing”) and as a preposition (IN - ‘When Gromit barks Wallace feeds Gromit’). Also, the determiners distinguish between singular ones (a/an) versus the ones can could be used as plurals as well as singulars (the/my).

Verb forms improved

For the sentence “Wallace should have fed Gromit cheese” I decided to introduce more complicated verb structures. VP[TENSE=presperf] was introduced as a word ‘have’ in a certain form followed by a verb in the past particle form (‘has done’). VP[TENSE=prespart] was described as the word ‘be’ in a certain form followed by a verb in the present particle form (‘is doing’). To achieve that, two new language structured were introduced: HV, for different forms of the verb ‘have’, and BE, for different forms of the verb ‘be’. In addition to that, some other verb forms were extracted separately to prevent the language from accepting syntactically incorrect sentences. We say that a verb can only be used on its own in a VP if it is in the present simple or past simple form (e.g. ‘I went away’ or ‘Gromit eats’). The rest of the verbs have to be included into a verb structure. Tjis means that instead of having a simple structure VP[NUM=?n, PER=?p, TENSE=?t, SUBCAT=?args] ->V[NUM=?n, PER=?p, TENSE=?t, SUBCAT=?args], we have two copies of this rule with two tenses specified already - pres and past. Another rule was added for the present participle forms: VP[NUM=?n, PER=?p, TENSE=prespart, SUBCAT=?args] ->BE[NUM=?n, PER=?p] V[TENSE=prespart, SUBCAT=?args] , and it specifies that the verb has to be preceded by the verb ‘be’ in a proper form. A similar rule exists for the present perfect form. This extraction made the description of a VP less compact (since we have several rules instead of one) but allows for more correctness of the grammar.

Testing

To test the parser, I tried all the sentences given in the practical specification and added my own cases that seemed interesting to me. The first test, on valid sentences, shows that all sentences have at least one parsing tree, and the second test, on invalid sentences, gives to trees at all.

In an attempt to validate the result I used an online syntax parser from the Stanford NLP group and manually compared the structures. In most cases it is analogous, and, in the other cases, quite similar to what my grammar produced.

More than 1 parsing tree

In some cases, the sentences showed some ambiguity and produced more than 1 parsing tree. Usually, if a grammar produces a lot of parsing trees for a sentence, it means that some parts of the grammar are redundant. However, in my case only two sentences give more than one tree.

First, the sentence “Wallace thinks Gromit eats cheese and drinks water” has two trees: one says that Wallace thinks that NP (Gromit) VP (VP(eats cheese) and VP(drinks water)), as in both of the actions are performed by Gromit. The other tree assumes that it is, in fact, Wallace, that performs two actions: 1. thinks that Gromit eats cheese and 2. drinks water. So, it is not clear who drinks water - Wallace or Gromit.

The sentence “Wallace often eats tasty soft cheese in the kitchen after dinner” has even more points of ambiguity. The two PPs “in the kitchen” and “after dinner” could be applied to different objects: cheese (both), eating (both), kitchen (dinner). Overall this produces 9 trees. Again, from the point of view of grammar there is nothing that can be done without trying to understand the meaning of the sentence, and, even then, different variants are possible. In other languages this ambiguity is usually removed via the punctuation, however, this is not the case in this particular situation.

Conclusions

All the basic and the extension requirements have been done. The file contains a script with the grammar and a set of valid and invalid sentences, which are based on the given sentences and supplemented with other interesting cases. A README file gives an instruction how to run the script.

Bibliography

[1] The OWL Writing Lab at Purpue University. Purdue owl: Verb tenses. https://owl.english.purdue.edu/owl/resource/601/01/. (Accessed on 04/17/2017).

[2] Stanford NLP Group. Stanford parser. http://nlp.stanford.edu:8080/parser/index.jsp. (Accessed on 04/17/2017).

[3] Julia Hockenmaier and Mark Steedman. Acquiring compact lexicalized grammars from a cleaner treebank. In *LREC*, 2002.

[4] Martin Kay. Unification in grammar. In *Natural Language Understanding and Natural Language Understanding Workshop*, pages 233–240, 1984.

[5] Raj Nath Patel, Rohit Gupta, Prakash B Pimpale, et al. Reordering rules for english-hindi smt. *arXiv preprint arXiv:1610.07420*, 2016.

[6] Jane J Robinson. Diagram: A grammar for dialogues. *Communications of the ACM*, 25(1):27–47, 1982.