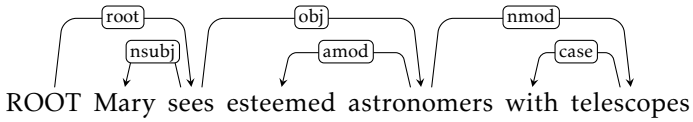
Dependency Parsing exercises: Transition-based arc-standard parsing

May 25, 2021

1. Enumerate the configurations an arc-standard transition-based parser goes through when parsing the sentence:

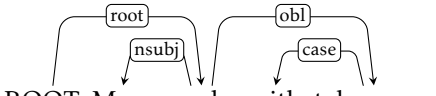


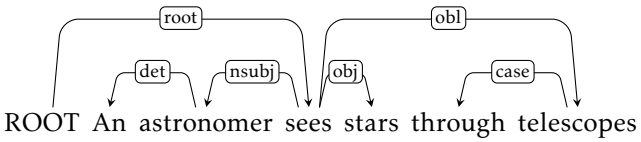
A transition is a left-arc, right-arc, or shift operation (LA, RA, or S). At each step, indicate the operation, the contents of the stack, the input buffer, and which dependency is added, if any:

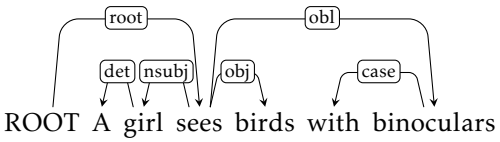
Solution:

TRANS.	STACK	BUFFER	ARCS
	[ROOT]	[Mary sees esteemed . . .]	$\emptyset$
SH	[ROOT Mary]	[sees esteemed astronomers . . .]	
LA_{NSUBJ}	[ROOT]	[sees esteemed astronomers . . .]	+(Mary $\overset{\text{NSUBJ}}{\leftarrow}$ sees)
SH	[ROOT sees]	[esteemed astronomers with . . .]	
SH	[ROOT sees esteemed]	[astronomers with telescopes]	
LA_{AMOD}	[ROOT sees]	[astronomers with telescopes]	+(esteemed $\overset{\text{AMOD}}{\leftarrow}$ astronomers)
SH	[ROOT sees astronomers]	[with telescopes]	
SH	[. . . astronomers with]	[telescopes]	
LA_{CASE}	[ROOT sees astronomers]	[telescopes]	+(with $\overset{\text{CASE}}{\leftarrow}$ telescopes)
RA_{NMOD}	[ROOT sees]	[astronomers]	+(astronomers $\overset{\text{NMOD}}{\rightarrow}$ telescopes)
RA_{OBJ}	[ROOT]	[sees]	+(sees $\overset{\text{OBJ}}{\rightarrow}$ astronomers)
RA_{ROOT}	[]	[ROOT]	+(ROOT $\overset{\text{ROOT}}{\rightarrow}$ sees)
SH	[ROOT]	[]	

2. It turns out that we have a small corpus with some more information on the kinds of attachments "sees" and "telescopes" tend to have:







In this corpus, all prepositions are attached to verbs! Assume that an arc-standard model is trained with the following feature templates:

- the word below the top of the stack
- the word on top of the stack
- the first word in the input buffer
- the second word in the input buffer

An example of a configuration and its features would look as follows:

- Configuration: [ROOT] [Mary works with telescopes] $\emptyset$
- Features: (ε, ROOT, Mary, works)

1. We expect to arrive at a different analysis of the sentence in Ex. 1 if we train on this corpus. Assume we train a simple nearest neighbor classifier, i.e.:

   - For a given parsing configuration, the parser chooses the transition of the single most similar configuration in the training dataset.[1] If there are several candidates, the majority class should be chosen.

   Which configuration, transition, and features in the training dataset will be responsible for changing the attachment of the preposition as compared to the analysis in the previous exercise?

   Solution:

   Let us first determine the transition sequences for the individual trees in the training set:

   - ROOT Mary works with telescopes:
     1. initial state: [ROOT] [Mary works with telescopes]
     2. SH: [ROOT Mary] [works with telescopes]
     3. LA_{NSUBJ}: [ROOT] [works with telescopes]
     4. SH: [ROOT works] [with telescopes]
     5. SH: [ROOT works with] [telescopes]
     6. LA_{CASE}: [ROOT works] [telescopes]
     7. RA_{OBL}: [ROOT] [works]
     8. RA_{ROOT}: [] [ROOT]
     9. SH: [ROOT] []
   - ROOT An astronomer sees stars through telescopes:
     1. initial state: [ROOT] [An astronomer sees stars through telescopes]
     2. SH: [ROOT An] [astronomer sees stars through telescopes]
     3. LA_{DET}: [ROOT] [astronomer sees stars through telescopes]
     4. SH: [ROOT astronomer] [sees stars through telescopes]
     5. LA_{NSUBJ}: [ROOT] [sees stars through telescopes]
     6. SH: [ROOT sees] [stars through telescopes]
     7. RA_{OBJ}: [ROOT] [sees through telescopes]
     8. SH: [ROOT sees] [through telescopes]
     9. SH: [ROOT sees through] [telescopes]
     10. LA_{CASE}: [ROOT sees] [telescopes]
     11. RA_{OBL}: [ROOT] [sees]

[1] To measure the similarity between two configurations, count the number of feature values they share. For instance, $x = (\varepsilon, \text{ROOT}, \text{Mary}, \text{works})$ is more similar to $y = (\varepsilon, \text{ROOT}, \text{John}, \text{works})$ than to $z = (\text{ROOT}, \text{Mary}, \text{works}, \text{with})$ because x and y share the two top elements of the stack (ε and ROOT) and the second word in the buffer (works), while x and z share no common features. Note that while both x and z have the feature value ROOT, in x it corresponds to the top of the stack and in z it corresponds to the word below the top of the stack.

12. RA_{ROOT}: [] [ROOT]
13. SH: [ROOT] []

- ROOT A girl sees birds with binoculars:
  1. initial state: [ROOT] [A girl sees birds with binoculars]
  2. SH: [ROOT A] [girl sees birds with binoculars]
  3. LA_{DET}: [ROOT] [girl sees birds with binoculars]
  4. SH: [ROOT girl] [sees birds with binoculars]
  5. LA_{NSUBJ}: [ROOT] [sees birds with binoculars]
  6. SH: [ROOT sees] [birds with binoculars]
  7. RA_{OBJ}: [ROOT] [sees with binoculars]
  8. SH: [ROOT sees] [with binoculars]
  9. SH: [ROOT sees with] [binoculars]
  10. LA_{CASE}: [ROOT sees] [binoculars]
  11. RA_{OBL}: [ROOT] [sees]
  12. RA_{ROOT}: [] [ROOT]
  13. SH: [ROOT] []

The resulting training set is therefore:[2]

1. (-, ROOT, Mary, works) → SH
2. (ROOT, Mary, works, with) → LA_{NSUBJ}
3. (-, ROOT, works, with) → SH
4. (ROOT, works, with, telescopes) → SH
5. (works, with, telescopes, -) → LA_{CASE}
6. (ROOT, works, telescopes, -) → RA_{OBL}
7. (-, ROOT, works, -) → RA_{ROOT}
8. (-, -, ROOT, -) → SH
9. (-, ROOT, An, astronomer) → SH
10. (ROOT, An, astronomer, sees) → LA_{DET}
11. (-, ROOT, astronomer, sees) → SH
12. (ROOT, astrono., sees, stars) → LA_{NSUBJ}
13. (-, ROOT, sees, stars) → SH
14. (ROOT, sees, stars, through) → RA_{OBJ}
15. (-, ROOT, sees, through) → SH
16. (ROOT, sees, through, telescopes) → SH
17. (sees, through, telescopes, -) → LA_{CASE}
18. (ROOT, sees, telescopes, -) → RA_{OBL}
19. (-, ROOT, sees, -) → RA_{ROOT}
20. (-, -, ROOT, -) → SH
21. (-, ROOT, A, girl) → SH
22. (ROOT, A, girl, sees) → LA_{DET}
23. (-, ROOT, girl, sees) → SH
24. (ROOT, girl, sees, birds) → LA_{NSUBJ}
25. (-, ROOT, sees, birds) → SH
26. (ROOT, sees, birds, with) → RA_{OBJ}
27. (-, ROOT, sees, with) → SH
28. (ROOT, sees, with, binoculars) → SH
29. (sees, with, binoculars, -) → LA_{CASE}
30. (ROOT, sees, binoculars, -) → RA_{OBL}
31. (-, ROOT, sees, -) → RA_{ROOT}
32. (-, -, ROOT, -) → SH

Let's look at the solution of Ex. 1 again and see if the classifier makes the same decision as the oracle. Instead of the arcs column, the table below contains the features corresponding to the individual configurations and the ID of the "nearest neighbor(s)" in the train set (the NNs column):

TR.	STACK	BUFFER	FEATURES	NNs
	[ROOT]	[Mary sees ...]	(-, ROOT, Mary, sees)	1,11,23
SH	[ROOT Mary]	[sees esteemed ...]	(ROOT, Mary, sees, esteemed)	2,12,24
LA_{NSUBJ}	[ROOT]	[sees esteemed ...]	(-, ROOT, sees, esteemed)	13,15,19,...
SH	[ROOT sees]	[esteemed astronomers ...]	(ROOT, sees, esteemed, astronomers)	14,16,18,26,28,30
RA_{OBJ}	[ROOT]	[sees astronomers with ...]	(-, ROOT, sees, astronomers)	13,15,19,...
SH	[ROOT sees]	[astronomers with ...]	**(ROOT, sees, astronomers, with)**	**26**
RA_{OBJ}	[ROOT]	[sees with telescopes]	...	...
...				

The configuration where the parser makes the PP-attachement mistake is ([ROOT sees], [astronomers with ...], ...) with feature representation (ROOT, sees, astronomers, with), to which the most similar element in the training set is 26. Hence, the RA_{OBJ} transition is taken, *astronomers* gets attached to *sees*, and it is no longer possible to attach *with telescopes* (or any

[2] The labels of the arcs are ignored for simplicity.

other word) to *astronomers*. Conclusion: the PP-attachement decision is taken earlier than it might seem!

Note that the first error the parser makes, marked in red, is to attach *esteemed* as a dependent of *sees*. However, there is a tie in the classification procedure and RA_{OBL} or SH transitions could be selected instead.[3] Assuming SH, the remaining transition sequence could start like this:

TR.	STACK	BUFFER	FEATURES	NNs
	[ROOT]	[Mary sees ...]	(-, ROOT, Mary, sees)	1,11,23
SH	[ROOT Mary]	[sees esteemed ...]	(ROOT, Mary, sees, esteemed)	2,12,24
LA_{NSUBJ}	[ROOT]	[sees esteemed ...]	(-, ROOT, sees, esteemed)	13,15,19,...
SH	[ROOT sees]	[esteemed astronomers ...]	(ROOT, sees, esteemed, astronomers)	14,16,18,26,28,30
SH	[...esteemed]	[astronomers with ...]	(sees, esteemed, astronomers, with)	2,3,17,26,27,29
SH	[...astronomers]	[with telescopes]	(esteemed, astronomers, with, telescopes)	4
SH	[...astronomers with]	[telescopes]	(astronomers, with, telescopes, -)	5
LA_{CASE}	[...astronomers]	[telescopes]	(esteemed, astronomers, telescopes, -)	5,6,17,18
RA_{OBL}	[...esteemed]	[astronomers]	...	...
...				

In this case, *with telescopes* gets actually correctly attached to *astronomers* as oblique!

2. Suppose we have a larger corpus about telescopes, and it turns out there is a lot of ambiguity with respect to telescopes and their attachments. What kind of feature would work best to deal with ambiguity: part of speech tags, lemmas, or word forms? What are the trade offs of different feature kinds?

   Solution: In general, different types of features have different trade-offs, for instance:

   - Word forms require little pre-processing but using them directly as features requires large amounts of training data. Besides, two different inflected forms of the same word are (from the system's point of view) two different, unrelated word forms.
   - Part of speech tags (and more generally morphosyntactic tags) are much more coarse-grained then word forms or lemmas and thus make it easier to obtain a robust classifier with less training data. On the other hand, using POS tags on input requires preliminary POS tagging stage which, on the one hand, complicates the overall system and, on the other hand, can lead to error propagation issues.
   - Lemmas (or base forms) group together different forms of the same word and hence smaller amount of training data can be sufficient to obtain a robust system (in comparison with word forms). On the other hand, since lemmas abstract over some morphosyntactic properties of words (e.g. case or number), a system based on lemmas might not be able to capture agreement-related patterns. Besides, using lemmas as features has the same drawback as using POS tags – an additional pre-processing step is necessary, with all the disadvantages of this setup.

   Within the context of PP-attachement ambiguity specifically:

   - Both word forms and lemmas allow to capture bilexical patterns (e.g. that *telescope* attaches more likely to *see* or *astronomer* than to *eat* or *mouse*). The form/lemma of the verb alone can be also informative since some verbs may take oblique dependents more often than others.
   - POS tags seem less informative for the PP-attachment task than word forms or lemmas: knowing that a potential head of a PP is either a noun or a verb doesn't bring the system much closer to making a correct PP-attachment decision (although it can prevent the system from attaching a PP to let's say an adverb, but this is not as much a PP-attachment ambiguity as rejecting an agrammatical structure). However, it may be useful to know if there is a preposition in the buffer at all – if not, the object can be attached directly to the verb.[4] Other morphosyntactic attributes (e.g. person and number) may allow to capture agreement-related patterns and thus exclude some grammatically erroneous interpretations.

[3] Unless we disregard the transition label, in which case RA is clearly the majority class here.

[4] If the system shifts the object of the verb to stack, it will no longer be possible to attach it to the verb without either attaching some right dependents to the object or making the object a left dependent of another word.

It is worth noting that in the era of deep learning word forms are represented by the corresponding embeddings (dense vector representations) which largely solves the sparsity issue without having to resort to additional pre-processing steps such as POS tagging or lemmatization (at the price of making the system and its decisions less transparent).