NLP Assignment 3

Andreea Bacanu, Christ's, aeb83

Friday 6th December, 2019

Code location: `https://github.com/andreea794/nlp-part2`

Task 1

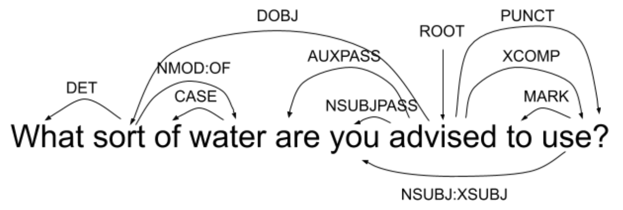


Figure 1: Output of the Stanford parser on Question 5, Text 1

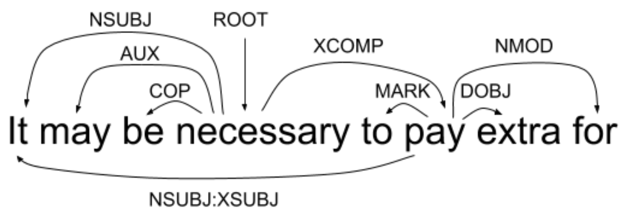


Figure 2: Output of the Stanford parser on Question 12, Text 2

Dependencies directly relate words in a sentence through labelled or unlabelled edges representing relationships (Copestake 2019, Lecture 5). Figure 1 and Figure 2 show the output of the Stanford CoreNLP dependency parser[1] for the 2 questions considered in this task.

For the first question, the determiner *what* restricts the search space of expected answers to noun phrases. We use RAKE[2] to extract keywords *use* and *water* as the most significant verb and noun. The nominal modifier relation nmod(*sort*,*water*) implies the presence of an adjectival modifier or a compound relation between a syntactic structure x and the word *water*. The dependency path between *what* and *use* also carries meaning if we consider the hypothesis of Comas et al. (2010): the syntactic relations between keywords in answer contexts and candidate answers should be similar to the ones between question keywords and the question tag. *use* is a keyword and, for factoid questions[3], the tag is always the *wh*–structure, in our case the word *what*. The path between *what* and *use*, $[DET - DOBJ - ROOT - XCOMP]$, can be further simplified by removing the determiner (for lack of relevance).

This helps reduce the search space of answers to the two sentences presented in Figure 3 and Figure 4. In Figure 3 we can identify a compound dependency between *tap* and *water*, as well as the employment of *using*[4] as an open clausal complement and of *water* as a direct object. Figure 4 shows *distilled* as an adjectival modifier of *water* and *water* as a direct object of *use*, as queried by the parser. To further reduce the answer space to one answer, heuristics can be applied.

The second question gives us the candidate answers already, noun phrases. We employ each of the 4 and generate the parser outputs in Figures 5, 6, 7, 8. As before, we use RAKE to extract keyphrase *pay extra*. Identifying the answer is no longer straightforward as the structure we would hope to find is *supplementary charges*. It would involve employing synonymy to recognize the similarity between the two. Syntactically, we would also need to restructure the keyphrase into *extra payment*, coverting *extra* into an adjective modifier to match the syntax.

[1]`https://corenlp.run/`

[2]Rapid Automatic Keyword Extraction – widely-known python library that can be used for keyword extraction: `https://github.com/aneesha/RAKE`

[3]questions whose answers are semantic or syntactic entities e.g. person names, locations, dates

[4]we assume preprocessing of the input data includes stemming

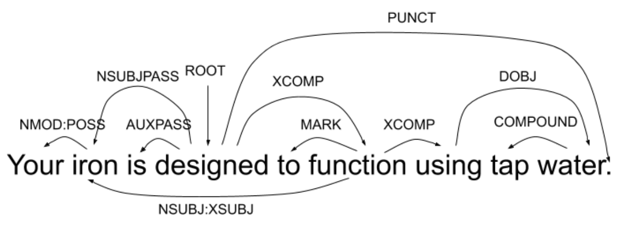


Figure 3: Output of the Stanford parser on candidate 1.1

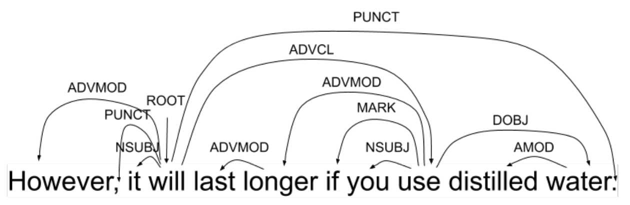


Figure 4: Output of the Stanford parser on candidate 1.2

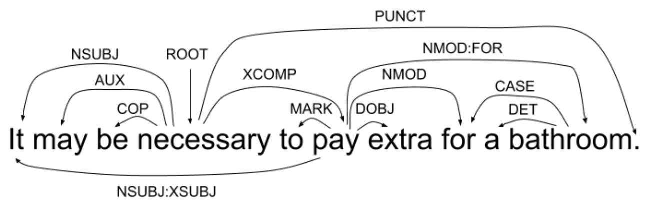


Figure 5: Output of the Stanford parser on candidate 2.1

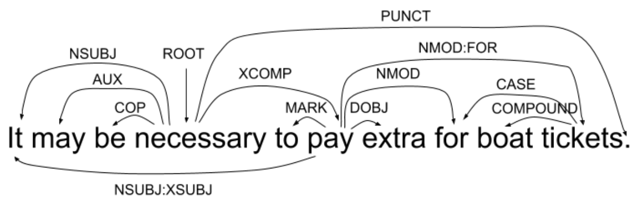
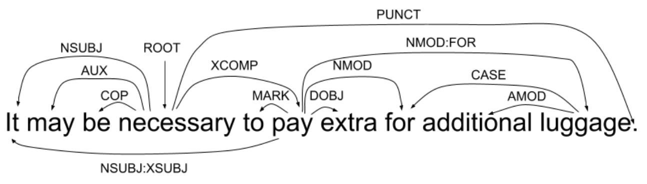


Figure 6: Output of the Stanford parser on candidate 2.2

Figure 7: Output of the Stanford parser on candidate 2.3



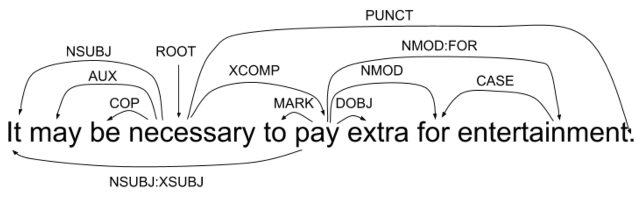


Figure 8: Output of the Stanford parser on candidate 2.4

Word count: 343[5]

Task 2

No lexical match for *drop* exists in the text. This is a case where instead of an explicit meaning postulate, a similarity metric can can be used to relate features (Copestake 2019, Lecture 6). WordNet is the main lexical database of English, used primarily for its size and availability. Parts of speech are grouped into synsets representing different concepts (Copestake 2019, Lecture 7). Using WordNet to find semantically similar words to *drip* yields *trickle*, *dribble*, *dripping*, *drop*, none of which can be found in the text. However, recursively querying WordNet reveals that *drop* is a hypernym of *droplet*.

GoogleNews `word2vec` embeddings can't generate any helpful similar word in this case. The model finds features stemming from *drip*, such as *dripping* and *drips*. A model trained on stemmed data would be more useful in this case given that we stem the query words as well.

But even with this improvement, `word2vec` has known issues. For instance, if the same semantic similarity approach were applied for question 25 (*The club is open to non-members on Tuesday evenings.*), the most similar word to *open* would, according to the model, be *closed* (excluding

[5]word count performed by the TexCount web interface: `https://app.uio.no/ifi/texcount/online.php`

opened, which yields a higher similarity, but we are assuming stemming[6]), with a similarity of 0.58. Antonyms have high semantic similarity, which can introduce wrong candidate answers and therefore lead to the possibility of incorrectly answering the question. If a different similiarity metric were used, such as ontology-based similiarity, *closed* would be recognised as an antonym.

Another example of the potential introduction of wrong answer candidates can be reproduced when trying to answer question 9 (*If you want to sit at the front of the coach*). Querying `word2vec` on the most similar feature for *coach* (in context, the vehicle) returns *Head_coach*. Although not explicitly, this raises the issue of homonymy, the most frequent case of polysemy (Copestake 2019, Lecture 7). As a solution, a WSD[7] technique would need to be employed to distinguish between different meanings of the same word.

Word count: 324

Task 3

The reasoning chain, as produced by a human, would somewhat follow these steps:

- a functioning iron has a hot metal surface
- the hot metal surface comes in contact with clothes being ironed, making them hot
- when wearing clothes, they are in direct contact with the skin
- therefore, wearing the clothes while ironing them involves a hot surface coming in contact with the skin
- skin can burn if touched by a hot surface, therefore one's skin can burn if they iron clothes they are wearing
- burns hurt, therefore ironing clothes while wearing them can cause someone harm

Automating this task involves building a system that can perform inference based on new information it is given and prior knowledge from a knowledge base. This requires finding a suitable knowledge representation, which can be first-order logic (Copestake 2019, Leture 6). We model the task as a logical inference question: find some_use(iron) such that hurt(person). Letting x be the misuse of the iron we are searching for, we have:

$$x(iron) \implies hurt(person)$$

A helpful sentence that we would hope the system would use alongside the two presented in the task is *Do not attempt to remove creases from an item of clothing that is being worn.* A potential logical chain could be:

$$\begin{aligned} \exists(p, c) &\in People \times Clothes \\ &: iron(p, c) \land wear(p, c) \\ &\implies hot(c) \land wear(p, c) \\ &\implies burnt(skin(p)) \\ &\implies hurt(p) \end{aligned}$$

[6]this would imply training on a stemmed dataset and therefore different inferred similarities, but the hypothesis regarding antonymy still stands

[7]Word Sense Disambiguation

Unfortunately this would require access to an extensive knowledge base that includes "common sense" inferences such as *Clothes worn by a person are in direct touch with the skin.* Some of the rules in the inference chain can result from training the model on the text, but most require training on a large enough corpus or manual addition to the knowledge base. Although this new hypothetical system could now achieve a decent performance on a domain that includes the functionality of irons or more generally house appliances[8], scaling that to all other fields would involve an infinitely large knowledge base with manual updates for any new information category.

Word count: 333

References

Comas, P. R., Turmo, J. & Marquez, L. (2010), Using dependency parsing and machine learning for factoid question answering on spoken documents, *in* 'Proceedings of Interspeech 2010 : spoken language processing for all', Annual Conference of the International Speech Communication Association, pp. 1–4.

Copestake, A. (2019), Overview of natural language processing lectures. unpublished.

[8]generalisation here is relative – there are still specific functional details of each appliance that the system wouldn't have learnt by being trained on this text alone