School of Computing and Information Systems
The University of Melbourne
COMP90042
NATURAL LANGUAGE PROCESSING (Semester 1, 2022)

Sample solutions: Week 7

Discussion

1. What are **contextual representations**?
   - The contextual representation of a word is a representation of the word based on a particular usage. It captures the different senses or nuances of the word depending on the context.
   - Contextualised representations are different to word embeddings (e.g. Word2Vec) which gives a single representation for every word type.
   - Contextual representations that are pre-trained on large data can be seen as a model that has obtained fairly comprehensive knowledge about the language.
2. How does a **transformer** captures dependencies between words? What advantages does it have compared to RNN?
   - Transformer uses **attention** to capture dependencies between words.
   - For each target word in a sentence, transformer attends to every other words in the sentence to create its contextual embeddings.
   - As the computation of the contextual embeddings of a target word is independent to other target words, we can parallelise the computation. This is an important distinction to RNN, which rely on sequential processing: the contextual embedding of the current target word cannot be computed until we have processed the previous word.
   - This allows transformer-based models to scale to very large data that is difficult for RNN-based models.
3. What is **discourse segmentation**? What do the segments consist of, and what are some methods we can use to find them?
   - In Discourse Segmentation, we try to divide up a text into discrete, cohesive units based on sentences.
   - By interpretting the task as a boundary–finding problem, we can use rule–based or unsupervised methods to find sentences with little lexical overlap (suggesting a discourse boundary). We can also use supervised methods, by training a classifier around paragraph boundaries.
4. What is an **anaphor**?
   - From the lectures: an anaphor is a linguistic expression that refers back to one or more elements in the text (generally preceding the anaphor)
   - These tend to be pronouns (*he*, *she*) but can also be determiners (*which*, *the*, etc.).

(a) What is **anaphora resolution** and why is it difficult?

- This is the problem of working out which element (generally a noun or noun phrase, but sometimes a whole clause) a given anaphor is actually referring to.
- For example:

  Mary gave John a cat for **his** birthday. (i) **She** is generous. (ii) **He** was surprised. (iii) **He** is fluffy.

  his [birthday] obviously refers to John; (i) (presumably) refers to *Mary*; (ii) (presumably) refers to *John*; and (iii) (presumably) refers to *[the] cat*.

(b) What are some useful heuristics (or features) to help resolve anaphora?

- The most obvious (but inherently unreliable) heuristic is the **recency heuristic**: given multiple possible referents (that are consistent in meaning with the anaphor), the mostly intended one is the one most recently used in the text.
- A better heuristic is that the most likely referent (consistent in meaning with the anaphor) is the focus of the discourse (the "center").
- We can also build a supervised machine learning model, usually based around the semantic properties of the anaphor/nearby words and the sentence/discourse structure.

Programming

1. In the iPython notebook `10-bert`, we provide an example on how we can use a pre-trained BERT model and fine-tune it for a sentiment analysis task. As we'll need a GPU to train BERT, we'll be running the notebook on colab, which provides one free GPU. So the first step is to go to: `https://colab.research.google.com/` and sign up or login to a Google account. Next go to "File > Upload Notebook" and upload the notebook (`10-bert.ipynb`) to colab.

   - Fine-tune the model with more epochs (e.g. 4), and take the best model (based on development performance) and measure its performance on the test set.
   - Modify the code so that you can freeze the BERT parameters from updating during fine-tuning. What performance do you now get?
   - If you're interested in using TPUs (which stands for tensor processing units, hardware designed specifically for neural network models) on colab, we've also provided `10-bert-tpu.ipynb` which modifies the code to use TPU.

Get ahead

- Extend the notebook `10-bert` for other tasks:
  - Sentence similarity (STS 2017): `http://alt.qcri.org/semeval2017/task1/index.php?id=data-and-tools`
  - Question answering (SQuAD v1.1): `https://rajpurkar.github.io/SQuAD-explorer/`