School of Computing and Information Systems
The University of Melbourne
COMP90042
NATURAL LANGUAGE PROCESSING (Semester 1, 2020)

Sample solutions for discussion exercises: Week 2

Discussion

1. Give some examples of text processing applications that you use on a daily basis.
   - There are lots! For example, Google (or other web search engines), Siri (or other speech-to-text systems), predictive messaging, spelling correction, machine translation, and so on.
2. What is **tokenisation** and why is it important?
   - Tokenisation is the act of transforming a (long) document into a set of meaningful substrings, so that we can compare with other (long) documents.
   - In general, a document is too long — and contains too much information — to manipulate directly. There are some counter-examples, like **language identification**, which we need to perform before we decide how to tokenise anyway.

   (a) What are **stemming** and **lemmatisation**, and how are they different? Give examples from the `01-preprocessing` iPython notebook.
   - Both stemming and lemmatisation are mechanisms for transforming a token into a canonical (base, normalised) form. For example, turning the token *walking* into its base form *walk*.
   - Both operate by applying a series of rewrite operations to remove or replace (parts of) affixes (primarily suffixes). (In English, anyway.)
   - However, lemmatisation works in conjunction with a **lexicon**: a list of valid words in the language. The goal is to turn the input token into an element of this list (a valid word) using the rewrite rules. If the re-write rules can't be used to transform the token into a valid word, then the token is left alone. (For example, the token *lemming* wouldn't be transformed into *lemm* because the latter isn't in the word list.)
   - Stemming simply applies the rewrite rules, even if the output is a garbage token (like *lemm*).
   - One further idea is the difference between **inflectional morphology** and **derivational morphology**:
     - Inflectional morphology is the systematic process (in many but not all languages) by which tokens are altered to conform to certain grammatical constraints: for example, if the English noun *teacher* is plural, then it must be represented as *teachers*. The idea is that these changes don't really alter the meaning of the term. Consequently, both stemming and lemmatisation attempt to remove this kind of morphology.

  - Derivational morphology is the (semi-)systematic process by which we transform terms of one **class** into a different class. For example, if we would like to make the English verb *teach* into a noun (someone who performs the action of *teaching*), then it must be represented as *teacher*. This kind of morphology tends to produce terms that differ (perhaps subtly) in meaning, and the two separate forms are usually **both** listed in the lexicon. Consequently, lemmatisation doesn't usually remove derivational morphology in its normalisation process, but stemming usually does.
- Another example, from the notebook, is the token *this*. Using the lemmatiser, the token remains unchanged, because it is already listed in the lexicon. The stemmer, however, strips the *-s* suffix, so that we end up with *thi*.