Stemming and lemmatization

For grammatical reasons, documents are going to use different forms of a word, such as *organize*, *organizes*, and *organizing*. Additionally, there are families of derivationally related words with similar meanings, such as *democracy*, *democratic*, and *democratization*. In many situations, it seems as if it would be useful for a search for one of these words to return documents that contain another word in the set.

The goal of both stemming and lemmatization is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form. For instance:

am, are, is ⇒ be
car, cars, car's, cars' ⇒ car

The result of this mapping of text will be something like:

the boy's cars are different colors ⇒
the boy car be differ color

However, the two words differ in their flavor. *Stemming* usually refers to a crude heuristic process that chops off the ends of words in the hope of achieving this goal correctly most of the time, and often includes the removal of derivational affixes. *Lemmatization* usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the *lemma* . If confronted with the token *saw*, stemming might return just *s*, whereas lemmatization would attempt to return either *see* or *saw* depending on whether the use of the token was as a verb or a noun. The two may also differ in that stemming most commonly collapses derivationally related words, whereas lemmatization commonly only collapses the different inflectional forms of a lemma. Linguistic processing for stemming or lemmatization is often done by an additional plug-in component to the indexing process, and a number of such components exist, both commercial and open-source.

The most common algorithm for stemming English, and one that has repeatedly been shown to be empirically very effective, is *Porter's algorithm* (Porter, 1980). The entire algorithm is too long and intricate to present here, but we will indicate its general nature. Porter's algorithm consists of 5 phases of word reductions, applied sequentially. Within each phase there are various conventions to select rules, such as selecting the rule from each rule group that applies to the longest suffix. In the first phase, this convention is used with the following rule group:

(F)

Rule			Example		
SSES	→	SS	caresses	→	caress
IES	→	I	ponies	→	poni
SS	→	SS	caress	→	caress
S	→		cats	→	cat

Many of the later rules use a concept of the *measure* of a word, which loosely checks the number of syllables to see whether a word is long enough that it is reasonable to regard the matching portion of a rule as a suffix rather than as part of the stem of a word. For example, the rule:

$$(m > 1) \quad \text{EMENT} \quad \rightarrow$$

would map *replacement* to *replac*, but not *cement* to *c*. The official site for the Porter Stemmer is:

http://www.tartarus.org/~martin/PorterStemmer/

Sample text: Such an analysis can reveal features that are not easily visible from the variations in the individual genes and can lead to a picture of expression that is more biologically transparent and accessible to interpretation

Lovins stemmer: such an analys can reve featur that ar not eas vis from th vari in th individu gen and can lead to a pictur of expres that is mor biolog transpar and acces to interpres

Porter stemmer: such an analysi can reveal featur that ar not easili visibl from the variat in the individu gene and can lead to a pictur of express that is more biolog transpar and access to interpret

Paice stemmer: such an analys can rev feat that are not easy vis from the vary in the individ gen and can lead to a pict of express that is mor biolog transp and access to interpret

Figure 2.8: A comparison of three stemming algorithms on a sample text.

Other stemmers exist, including the older, one-pass Lovins stemmer (Lovins, 1968), and newer entrants like the Paice/Husk stemmer (Paice, 1990); see:

http://www.cs.waikato.ac.nz/~eibe/stemmers/
http://www.comp.lancs.ac.uk/computing/research/stemming/

Figure 2.8 presents an informal comparison of the different behaviors of these stemmers. Stemmers use language-specific rules, but they require less knowledge than a lemmatizer, which needs a complete vocabulary and morphological analysis to correctly lemmatize words. Particular domains may also require special stemming rules. However, the exact stemmed form does not matter, only the equivalence classes it forms.

Rather than using a stemmer, you can use a *lemmatizer* , a tool from Natural Language Processing which does full morphological analysis to accurately identify the lemma for each word. Doing full morphological analysis produces at most very modest benefits for retrieval. It is hard to say more, because either form of normalization tends not to improve English information retrieval performance in aggregate - at least not by very much. While it helps a lot for some queries, it equally hurts performance a lot for others. Stemming increases recall while harming precision. As an example of what can go wrong, note that the Porter stemmer stems all of the following words:

operate operating operates operation operative operatives operational

to oper. However, since *operate* in its various forms is a common verb, we would expect to lose considerable precision on queries such as the following with Porter stemming:

operational and research
operating and system
operative and dentistry

For a case like this, moving to using a lemmatizer would not completely fix the problem because particular inflectional forms are used in particular collocations: a sentence with the words *operate* and *system* is not a good match for the query operating and system. Getting better value from term normalization depends more on pragmatic issues of word use than on formal issues of linguistic morphology.

The situation is different for languages with much more morphology (such as Spanish, German, and Finnish). Results in the European CLEF evaluations have repeatedly shown quite large gains from the use of stemmers (and compound splitting for languages like German); see the references in Section 2.5 .

Exercises.

- Are the following statements true or false?
  1. In a Boolean retrieval system, stemming never lowers precision.
  2. In a Boolean retrieval system, stemming never lowers recall.
  3. Stemming increases the size of the vocabulary.
  4. Stemming should be invoked at indexing time but not while processing a query.
- Suggest what normalized form should be used for these words (including the word itself as a possibility):
  1. 'Cos
  2. Shi'ite
  3. cont'd
  4. Hawai'i
  5. O'Rourke
- The following pairs of words are stemmed to the same form by the Porter stemmer. Which pairs would you argue shouldn't be conflated. Give your reasoning.
  1. abandon/abandonment
  2. absorbency/absorbent
  3. marketing/markets
  4. university/universe
  5. volume/volumes
- For the Porter stemmer rule group shown in porter-rule-group:
  1. What is the purpose of including an identity rule such as SS → SS?
  2. Applying just this rule group, what will the following words be stemmed to?

     circus canaries boss

  3. What rule should be added to correctly stem *pony*?
  4. The stemming for *ponies* and *pony* might seem strange. Does it have a deleterious effect on retrieval? Why or why not?