Computing lab

Natural language morphology

We are going to discover some practical aspects of natural language processing via a commercial text tool by **Xerox**, and via a research-oriented multi-lingual corpus processor **Unitex**.

Exercice 1. **(Xerox morphological tools)** Connect to the URL containing demo versions of Xerox morphological tools: http://www.xrce.xerox.com/competencies/content-analysis/toolhome.en.html. Choose English. Enter the text: *He gave her a forget-me-not.*

a. **(Tokenization)** Choose *Tokenization* and examine the number of tokens found. How are separators treated by this tokenizer?

b. **(Morphological analysis)** Go back to the previous page and examine the list of *Morphological categories*.

i. Identify inflectional classes and inflectional values.

ii. Which Xerox "categories" are ambiguous?

iii. Run the morphological analysis.

1. Which word forms are ambiguous?
2. How many real interpretations are given to the verb *gave*?
3. Which word form does not appear in the morphological dictionary?

c. **(Tagging)** Go back to the demo page and examine the *Part-of-speech tagset*. Note that the annotations are not the same as in the morphological analysis.

i. The tag +VPRES is ambiguous because it does not specify the person and number of the verb. Find another example of a tag that remains ambiguous.

ii. Run the part-of-speech disambiguation. Each word obtains only one tag.

iii. Which tags are correct and non-ambiguous?

iv. Which tags are correct but remain ambiguous?

v. Which tags are incorrect?

vi. Which form has been guessed? Is the guessed tag correct?

Exercice 2. **(Discovering Unitex)**

a. Create a directory called *Unitex* in your usual personal storage area.

b. Run *Unitex2.0* from the desktop and indicate the new directory as the Unitex working directory.

c. Choose the English language.

d. Unitex treats texts in Unicode UTF-16 standard. Use the Unitex text editor to create new texts. Create a new file (*File Edition* → *New File*) containing the following sentence: *He gave her a forget-me-not.* Save the file and close the text editor.

e. **(Tokenization)** Open the same file to process it as a corpus (*Text* → *Open*). Do not preprocess the text. Open the *Token list*. How many tokens were identified and what are their frequencies in the corpus?

f. **(Lexical analysis)** Apply the morphological dictionary to the corpus (*Text* → *Apply Lexical Resources*). Initially only one public dictionary and no private one is accessible. Apply it to the text.

i. Examine the list of morphological interpretations given by the dictionary. Note that annotations are factorized if the lemma and the inflectional class are common.

ii. Which word forms are ambiguous? Which are non ambiguous?

iii. How many different interpretations are given to the word form *gave*?

iv. How many compounds were found?

v. How many word forms were unknown?

g. **(Sentence graph)** Construct the graph showing all morphological interpretations of the sentence (*Text* → *Construct FST-Text*).

i. How many paths exist in the graph?

ii. Which is the correct path?

Exercice 3. **(Processing a real-size corpus)** Copy the Obama.txt file from *P:\public_m2\TAL\savary*. It contains yesterday's Herald Tribune news article copied from the Internet.

a. Open the text, and preprocess it (preprocessing contains tokenization and application of the lexical resources).

b. Which word form is the most frequent one? What is the average percentage of tokens appearing only once in the text?

c. How many sentences were identified?

d. What is the kind of most of the unknown word forms?

e. How many tokens contain the first and the second sentence?

Exercice 4. **(Creating personal dictionaries)** Unitex allows to create personal lexical resources that can complete the public morphological dictionaries. Two forms of textual dictionaries exist:

- Dictionaries of lemmas and their inflection codes (so-called DELAS dictionaries). For instance in the following entry the lemma horse inflects according to the inflection code N3 (i.e. taking an *-es* in plural) :

  bus,N3

- Dictionaries of inflected forms and their annotations (so-called DELAF dictionaries). For instance, the following entry describes the inflectional paradigm of the lemma above.

  bus,bus.N:s
  buses,buses.N:p

a. Copy the N3.grf graph to your personal *Unitex\English\Inflection* directory. Open it (FSGraph → Open) and examine its structure. Labels inside boxes are endings, labels under the boxes are annotations. Compile the graph into a binary form (FSGraph → Tools → Compile FST2).

b. Create a file called *MyDictionary.dic* in your personal *Unitex\English\Dela* directory. Insert the entry **bus,N3**.

c. Open the *MyDictionary.dic* file as a DELA dictionary (*DELA → Open*). Inflect it (*DELA→Inflect*) and examine the resulting DELAF dictionary (called *MyDictionaryflx.dic*). Compile the inflected dictionary (*DELA → Open → MyDictionaryflx.dic; DELA → Compress into FST*). You can now see your new personal dictionary on the list of resources that can be applied to a text (Text → Apply Lexical Resource).

d. The word form *pushback* was unknown in the *Obama.txt* text. Create the inflection graph called N1.grf in *Unitex\English\Inflection* describing the singular and the plural word form for words like *pushback.* (You can use a copy of the *N3.grf* graph. After having selected a box you an edit it in a window above. Push Enter after having modified a box.)

e. Add *pushback* with the correct inflection code to *MyDictionary.dic* file. Inflect the file and check its contents. Compress it.

f. Copy the proper names unrecognized in *Obama.txt* into a text file called *Proper.dic* stored in the *Unitex\English\Dela* directory. Assign them inflection codes, inflect and compress the resulting dictionary. (If you need to modify a lemma in order to create an inflected form, use the L operator which deletes one letter to the left, e.g. *LLas* deletes two final letters and adds the –*as* ending.)

g. Re-apply the lexical resources to *Obama.txt*. You can now choose your two personal dictionaries that complete the public dictionaries. How many unknown words remain? What do they result from? (If you have enough time, examine the graph D:\apps\Unitex2.0\English\Graphs\Preprocessing\Replace. It allows to normalize abbreviated forms like I'm into I am, etc. Copy this graph into your personal Unitex\English Graphs\Preprocessing\Replace directory, and correct it so that it correctly analyses the unrecognized string *wasn*.)

Exercice 5. (Rule-based disambiguation) Let's work with Unitex on a French text.

a. Change the Unitex working language to French (Text → *Change language*)

b. In the Unitex file editor create a new file containing the sentence *Feras-tu bientôt cela?* Open this text as a corpus to be processed. Preprocess it and apply the default linguistic resources. Construct the graph of the sentence.

c. The sentence graph shows that *tu* can b either the pronoun or the past participle of the verb *taire*. A good rule is to says that is a second-person verb is followed by a hyphen (-) and by the second-person pronoun, than this pronoun is not ambiguous (it cannot be interpreted as a past participle). In other words the sequence

{Feras,faire.V:F2s}{-}{tu,taire.V:Kms} is to be eliminated. Such disambiguation rules can be created under Unitex by a system called **ELAG**.

d. Read the Unitex manual (in *D:\apps\Unitex2.0*), page 134 on constructing disambiguation grammars. Create the graph *elag-tu.grf* as shown on figure 7.12, and put it to your personal *Untex\French\Elag* directory. The graph has the following meaning: if a sequence matches the upper path (delimited by <!>) than it must also match the lower path (between <=>), otherwise it is eliminated. Here each path containing a past participle after a verb in 2-nd person singular, and a hyphen, will be eliminated.

e. Compile the graph into an ELAG rule (Text → *Compile Elag Grammars*). Choose the *Unitex/English/Elag* directory, select the *elag-tu.grf* graph for compilation.

f. Then apply the graph to the sentence created above and examine the interpretations that have been eliminated.

Exercice 6. **(Searching with regular expressions)** Read chapter 4 (p.57) of the manual in order to discover the expressive power of regular expressions in searching for sequence occurrences in a corpus. This application is called a concordancer.