Intro

This is GLEM, a lemmatiser.

We present the lemmatizer that we developed for ancient Greek. Not only does it disambiguate between multiple lemmas (with the same or different POS tags), but it also creates new lemmas for unknown words.

Currently available lemmatizers for ancient Greek such as the one developed in the CLTK [**?**] cannot handle such cases.

As the basis for the lemmatizer we used an existing memory-based learning tool, Frog [**?**], that was originally developed for Dutch. We converted Frog to work for ancient Greek. As the results of Frog on ancient Greek were rather modest, we decided to create a smarter lemmatizer named GLEM that combines a lexicon look up with the memory-based tool Frog.

GLEM's look up component overcomes the difficulty of a relative small training set in combination with a morphologically rich language, while the memory-based learning component enables GLEM to handle unknown words.

Usage

Basic Usage

Lemmatising a single file is done as follows.

```
python3 glem.py -f TEST_FILE
```

The input file is expected to contain plain greek text. This produces one output file (and lots of output to the screen). The output file is named as follows: `TEST_FILE.lastrun.wlt.txt`

The `lastrun` marker is appended to the filename automatically, and can be changed with the `-s` option. The following command will produce an output file named `TEST_FILE.run1.wlt.txt`.

```
python3 glem.py -f TEST_FILE -s run1
```

For each word in the input text, the output contains one line containing a word-lemma-tag triplet. An example is shown below.

```
γενόμενα    γίγνομαι    V--papmna-
```

If the `-S` option is added, two extra fields are added to the output. The first one contains the entry as it is contained in the internal lexicon in GLEM, and the second one contains a textual representation of the strategy used by the lemmatizer to reach the answer. The output file is named as follows: `TEST_FILE.lastrun.stats.txt`. An example is shown below.

```
python3 glem.py -f TEST_FILE -s run1 -S
```

And the output looks like this.

```
ιστορίης   ιστορία   N--s---fg-   /ιστορίης/ιστορία/N--s---fg-/2/greek_Haudag/
                                                 multi lemmas, no tag, highest frequency
```

Looking Up

The following example shows how to use the -w option to look up a word from the lexicons loaded by GLEM.

```
python3 glem.py -w τῶν
```

```
LOOKUP WORD τῶν
   τῶν, 14
     τῶν, ὁ, S--p---mg-,  1163 greek_Haudag
     τῶν, ὁ, S--p---qg-,   660 greek_Haudag
     τῶν, ὁ, S--p---ng-,   211 greek_Haudag
     τῶν, ὁ, S--p---fg-,   153 greek_Haudag
```

The next example shows how to use the -l option to look up a lemma from the loaded lexicons.

```
python3 glem.py -l Δάμαρις
```

```
LOOKUP LEMMA Δάμαρις
Δάμαρις
   Δάμαρις, Δάμαρις, N--s---fn-,     1 greek_Haudag
```

Frog

GLEM assumes FROG is available. It expects to be able to initialise FROG with a template named `frog.ancientgreek.template`.

If FROG is unavailable, run GLEM with the -F option to disable it.

Lexicon Files

GLEM reads three different lexicon files.

`greek_Haudag.pcases.lemma.lex.rewrite_new`. This file contains word-lemma-tag-frequency information, one entry per line. The following example shows two entries.

```
αὐτός αὐτός Pd-s---mn- 25
αὐτός αὐτός Pp3s---mn- 11
```

The second file is called `perseus-wlt.txt`. This file contains word-lemma-tag information without the frequency information. The following example shows two entries.

```
ἀναπλέουσι ἀναπλέω V-3ppia---
κατεπάγων κατεπάγω V--sppamn-
```

The third file is an extra file containing punctuation. It is called `extra-wlt.txt` and also contains word-lemma-tag entries. The following example shows two entries from this file.

```
(  (  punct
)  )  punct
```

All Options

GLEM accepts the following command line options.

Input and output

`-f` filename The file (or files, wildcards are allowed) to be lemmatised.

`-s` suffix The output files will be given this suffix.

`-S` Outputs extra information consisting of the full entry from GLEM's dictionary plus a textual representation of the strategy used to come to an answer.

`-W` Assumes the test file contains word-lemma-tag data separated by whitespace. The first word of each line is lemmatised, and the specified lemma and tag are assumed to be correct and compared to GLEM's output.

`-F` The test file will be processed with GLEM only, without using FROG for the unknown words. This means that unknown words remain unhandled.

Look-up

`-l` lemma Looks up the specified lemma in the dictionary.

`-w` word Looks up the specified word in the dictionary.

Lexicons

`-L` filename Loads the specified lexicon file.

`-M` filename Loads another ”merged” file.

`-E` filename Loads another ”extra” file. The default extra file contains punctuation.

Miscellaneous

`-v` Extra verbose output to the screen.

`-D` Prints extra debug information to the screen.

`-R` Remove the token `ROOT` from the lines in the text file.

Strategy

GLEM uses the following strategy when lemmatizing a text.

0.1: Lemmatiser Flowchart

```
Check if word in dictionary.

If it is:
  1) If it has only one tag/lemma entry, return it.
     ("one lemma, same pos tag" / "one lemma, different pos tag")
  2) More than one tag/lemma entry: go through the tag/lemmas, and:
     a) if a lemma with a similar pos tag is found, return it.
        ("multiple lemmas, same pos tag, highest frequency" or
         "multi lemmas, same pos tag, other frequency")
     b) otherwise, return the most frequent tag/lemma.
        ("multi lemmas, different pos tag, highest frequency")

If it is not in the dictionary:
  1) Take Frog entry, and return it.
     ("Frog" / "Frog list")
  2) If this fails:
  return ("unknown")
```