Frog Named Entity Recognition - Training Material | WP3

Author: Bob Boelhouwer INL

Introduction

Frog, formerly known as **Tadpole**, is an integration of memory-based natural language processing (NLP) modules developed for Dutch. All NLP modules are based on **Timbl**, the Tilburg memory-based learning software package. Most modules were created in the 1990s at the **ILK Research Group** (Tilburg University, the Netherlands) and the **CLiPS Research Centre** (University of Antwerp, Belgium). Over the years they have been integrated into a single text processing tool. More recently, a dependency parser, a base phrase chunker, and a named-entity recognizer module were added. In this document we will mainly describe how to apply Frog for named entity recognition. For this document we have liberally borrowed from the original documentation that can be found at http://ilk.uvt.nl/frog/. The software was developed for a single language: modern Dutch. Moreover, it can't be trained or adapted to specific content.

Requirements

The software is developed as a command line tool for Linux. It is therefore not recommended to use it on a Windows environment, since it will require a large amount of additional software and a complex configuration.
Installation on Linux, however, it quite simple because the software has been packaged for Debian, Ubuntu, Mint and Fedora.

Installation

Installation for the Linux versions mentioned above is quite easy, since the software is available in a default repository. It is sufficient to open the software manager and to search 'frog' and 'frogdata'. After installing these two packages we are ready to run.
If these packages are not available, it means that we have to add another repository. In the document http://ilk.uvt.nl/software/build-instructions-for-software-packages/get-install.html there are instructions how to select an additional repository.

Usage

To get an overview of all options available run

```
frog -h
```

The above command will also show where Frog keeps it configuration file. By adapting the configuration file it is possible to change the default settings for e.g. the set of tokenization rules. Below we will discuss the mort useful options that are available.

```
-e <encoding>  specify encoding of the input (default UTF-8)
-t <testfile>      Run frog on this plain text file
-x <testfile>      Run frog on this FoLiA XML file. The description of this format can he
```

Succeed is supported by the European Union under FP7-ICT and coordinated by Universidad de Alicante.

found here.

`-S <port>` Run as server instead of reading from testfile

`--testdir=<directory>` All files in this directory will be tested

`-n` Assume input file to hold one sentence per line

`--max-parser-tokens=<n>` inhibit parsing when a sentence contains over 'n' tokens. (default: no limit)

`--skip=[mptnc]` Skip Tokenizer (t), Chunker (c), Multi-Word Units (m), Named Entity Recognition (n), or Parser (p).

`-o <outputfile>` Output columned output to file, instead of default stdout

`-X <xmlfile>` Output also to an XML file in FoLiA format

`--id=<docid>` Document ID, used in FoLiA output. (Default 'untitled')

`--outputdir=<dir>` Output to directory, instead of default stdout

`--xmldir=<dir>` Use 'dir' to output FoliA XML to.

`--tmpdir=<directory>` (location to store intermediate files. Default /tmp)

`-Q` Enable quote detection in tokeniser.

Example of use

The most simple usage of frog is to run it on a file with plain text.

```
frog -t test.txt -o test.out
```

The output will look like this:

```
1       In      in      [in]    VZ(init) 0.995957 O       B-PP    4       mod
2       februari februari [februari]      SPEC(deeleigen)  1.000000 O       B-NP    1       obj1
3       2004    2004    [2004]  TW(hoofd,vrij)   0.998219 O       I-NP    2       mod
4       diende  dienen  [dien][de]       WW(pv,verl,ev)   0.998970 O       B-VP    0       ROOT
5       Wilders Wilders [Wilders]        SPEC(deeleigen)  1.000000 B-PER   B-NP    4       su
6       weer    weer    [weer]  BW()     0.992612 O       B-ADVP  4       mod
7       een     een     [een]   LID(onbep,stan,agr)      0.996212 O       B-NP    8       det
8       motie   motie   [motie] N(soort,ev,basis,zijd,stan)       0.998566 O       I-NP    4       predc
9       in      in      [in]    VZ(init) 0.500000 O       O       11      None
10      ,       ,       [,]     LET()    0.999925 O       O       9       punct
11      die     die     [die]   VNW(betr,pron,stan,vol,persoon,getal)     0.944056 O       B-SBAR  8
        mod
12      dit     dit     [dit]   VNW(aanw,det,stan,prenom,zonder,evon)     0.965812 O       B-NP    13
        det
13      maal    maal    [maal]  N(soort,ev,basis,zijd,stan)      0.781818 O       I-NP    14      su
14      werd    worden  [word]  WW(pv,verl,ev)   0.999799 O       B-VP    11      body
15      gesteund steunen [ge][steun][d]  WW(vd,vrij,zonder)       0.999705 O       I-VP    14      vc
16      door    door    [door]  VZ(init) 0.987805 O       B-PP    15      mod
17      alle    al      [alle]  VNW(onbep,det,stan,prenom,met-e,agr)      0.998823 O       B-NP    18
        det
18      partijen partij  [partij][en]    N(soort,mv,basis)        0.998313 O       I-NP    16      obj1
19      in      in      [in]    VZ(init) 0.998814 O       B-PP    18      mod
20      de      de      [de]    LID(bep,stan,rest)       0.996522 O       B-NP    21      det
21      Tweede_Kamer    Tweede_Kamer    [Tweede]_[Kamer] SPEC(deeleigen)_SPEC(deeleigen)  1.000000 B-
ORG_I-ORG       I-NP_I-NP       19      obj1
22      .       .       [.]     LET()    0.999956 O       O       21      punct
```

The seventh column specifies the named entities in BIO-format.