Frog & DeepFrog

ID: T139

Author: Maarten van Gompel (original authors of previous version were also: Iris Hendrickx, Antal van den Bosch)

Introduction:

Frog is an integration of memory-based natural language processing (NLP) modules developed for Dutch. Frog performs tokenization, part-of-speech tagging, lemmatization and morphological segmentation of word tokens. At the sentence level Frog identifies non-embedded phrase chunks in the sentence, recognizes named entities and assigns a dependency parse graph.

DeepFrog is a completely new and separate project that aims to train deep learning models on the same data as Frog. We want to see how-state-of-the-art transformer based techniques compare to those in Frog. It aims to deliver output in the same format (FoLiA) and with the same tags sets as Frog. Initial focus will be on only a subset of the modules available in Frog.

What must be adapted / extended / created anew:

This task first and foremost provides for the continued maintenance of Frog, as a popular software package for which users often come to us with questions, bug reports, and feature requests. Also included is maintenance and support for:

- ucto - The tokeniser Frog build upon
  - python-ucto - The Python binding for ucto
- python-frog - The python binding for Frog
- The Frog webservice (CLAM-based)

Further innovative development on Frog, as has been done the past years, has mostly halted now that the lead developer, Ko van der Sloot, has retired. Instead, a new focus is placed on DeepFrog, an attempt to accomplish the same linguistic enrichments with newer techniques, and ideally with better accuracy.

DeepFrog is build on the basis of state-of-the-art deep learning techniques (most notably transformers such as BERT), and existing industry-standard libraries such as torch/pytorch. We fine-tune our models on existing pre-trained models for dutch such as BERTje and RoBBERT.

Unlike Frog, which was fully developed in-house; DeepFrog will mostly build on 3rd party libraries for its cor efunctionality, where necessary, contributions will be made directly to these 3rd part libraries.

Why important for CLARIAH:

Frog is a widely popular and used tool in the Dutch NLP community, it's support and maintenance must remain guaranteed in the future.

DeepFrog development is important if we want a continued scientific impact and align ourselves with current state-of-the-art techiques.

Targeted/Actual users: End-users/developers/researchers

Actual use (quantify!):

The paper describing the core tool and algorithms from 2007 (Van den Bosch et al, 2007) has been cited 127 times in other works. Frog is widely used in the Dutch NLP community. Exact figures are hard to track but an indication is given by visits to its website and github page (see our report).

Lead: Maarten van Gompel (DI, KNAW)

Participants: Maarten van Gompel (DI, KNAW)

Allocated PMs: 6PM (this is insufficient for the remainder of this project and have already been spent)

Deliverables

Frog

1. (T139aD1) Software: Frog
   - Resolution of frog#54 **(COMPLETED)**

  - (T139aD1.1) Documentation
  - (T139aD1.2) Models: frogdata - Trained models for Frog
  - (T139aD1.3) Software: python-frog - A Python binding for Frog
  - (T139aD1.4) Software: Frog webservice (CLAM-based)
2. (T139aD2) Software: ucto - The tokeniser used by frog
  - (T139aD2.1) Documentation
  - (T139aD2.2) Models: uctodata - Rule-based configurations for ucto for various languages
  - (T139aD2.3) Software: python-ucto - A Python binding for ucto
  - (T139aD2.4) Software: Ucto webservice (CLAM-based)
3. (T139aD3) Service: Frog Webservice deployed at CLST
4. (T139aD4) Service: Ucto Webservice deployed at CLST

DeepFrog

1. (T139bD1) Software: DeepFrog
  - (T139bD1.1) Documentation
  - (T139bD1.2) Training pipelines for training the models
  - (T139bD1.3) DeepFrog webservice: An extra webservice and web-interface around the new software, suitable for the CLARIN/CLARIAH infrastructure
2. (T139bD2) Models (also usable independently of the DeepFrog tool in e.g. pytorch)
  - a part-of-speech model for Dutch using the CGN tagset
  - a lemmatisation model for Dutch
  - a Named Entity Recognition model
3. (T139bD3) Paper: Describing the new models and comparing them with the current Frog, to be published either at a conference or journal.

Milestones

Frog

- (T139aM1) Software: Frog v0.20 - Spring 2020 **(COMPLETED)** - Outcome of the redesign effort - This milestone addressed the previous plan of this task in which we planned a fundamental re-design of Frog to reduce complexity, increase maintainability, be more flexible, and speed things up. The implementation has been partially completed and while performance gains have achieved in certain regards, other new developments cancel this out again.

Due to the support character of this task, no further Frog milestones are planned in advance. New releases will be done as bugs emerge and are fixed.

DeepFrog

1. (T139bM1) Exploration stage towards finding possible solutions for each of the components **(COMPLETED)**
2. (T139bM2) Implementation stage (on hold indefinitely)
3. (T139bM3) Evaluation stage (on hold indefinitely)
4. (T139bM4) Revision stage (on hold indefinitely)

Progress Reports

Detailed progression of this task is logged as part of our regular progress reports .

Related tasks

This task heavily depends on T108 FoLiA. It also relates to T096 LaMachine for distribution and T142 CLAM for the webservices.

Changes

- Jan 2021: **The DeepFrog task is on hold as there seems to be no official budget/time for it and other priorities have taken precendence.**

- This task has entered a new stage and has been revised now lead developer Ko van der Sloot has retired. Almost all of the earlier deliverables have been completed and further maintenaince is now guaranteed by Maarten van Gompel.
- Milestone 1 (Frog v0.20) has been completed; this was a main deliverable in the previous version of this task.
- DeepFrog is new
- Since 14 July 2020 - This task has moved from CLST, Radboud University Nijmegen to Digital Infrastructure, Humanities. The webservice deployments, however, stay at CLST.

References

- Canisius, S., and Van den Bosch, A. (2007). Recompiling a knowledge-based dependency parser into memory. In Proceedings of the International Conference on Recent Advances in Natural Language Processing, RANLP-2007, pp. 104-108. Borovets, Bulgaria.
- Van den Bosch, A., Busser, G.J., Canisius, S., and Daelemans, W. (2007). An efficient memory-based morphosyntactic tagger and parser for Dutch. In P. Dirix, I. Schuurman, V. Vandeghinste, and F. Van Eynde (Eds.), Computational Linguistics in the Netherlands 2006: Selected Papers of the Seventeenth CLIN Meeting. Utrecht: LOT, pp. 191-206.