Google Summer of Code 2024

Dictionary Induction from Parallel Corpora

Chaitanya Gambali

Table of Contents

Contact Information

Name	Chaitanya Gambali
Location	India
University	Indian Institute of Technology (BHU) Varanasi
Email address	gschaitanya2003@gmail.com
IRC	daedealus03 (OFTC) (previously daedalus in Google Code-In 2019) @daedalus2003@matrix.org
Timezone	GMT+5:30
Github	gs-chaitanya
LinkedIn	gs-chaitanya
Timezone	UTC+5:30

Background

I am currently pursuing a Bachelor of Technology Degree in Electronics and Communication Engineering at the highly prestigious Indian Institute of Technology (BHU) Varanasi.

I love learning new languages, and I have a keen interest in Natural Language Processing and Linguistics. I have also contributed to Apertium previously in 2019 under Google Code-In 2019 and I was one of the six Finalists (Daedalus) selected for Google Code-In 2019 from Apertium.

Skills

Languages: Python, C/C++, JavaScript, Bash

Frameworks: Numpy, Pandas, PyTorch, TensorFlow, etc.

Why Am I Interested in Apertium?

In the world of Machine-Learning/Natural Language processing-based translation approaches, Apertium is a breath of fresh air and ingenuity. A rule-based approach to translation is very interesting and more interpretable compared to the data-hungry, uninterpretable black boxes that modern-day machine learning-based systems are.

Apertium's open-source nature, combined with its support for low-resource languages, makes it a promising alternative to NLP-based translation methods that still fail to support a lot of languages.

Which of the published tasks am I interested in? What do I plan to do?

I am interested in the Task: Dictionary Induction from Parallel Corpora. The aim is to construct bidirectional dictionaries for a language pair, given a pair of parallel corpora - i.e., the same content in two different languages using a single script that does the job for the user in a single step.

Coding Challenge

The coding challenge outlined here required me to develop a script that *"reads two parallel corpora, applies the appropriate monolingual taggers and some word-aligner, and then prints a list of paired words."*

To demonstrate the script, we have taken two files, `eng.txt` and `spa.txt`

Ideally, these files should contain the parallel corpora to be used for induction. For the purpose of demonstration, we use a single sentence.

`eng.txt` contains - `The cat ate the fish.`

`spa.txt` contains - `El gato comió el pez.`

First, the script concatenates the corresponding sentences in the source and target language file in the following format, called the fast_text format :

```
source sentence ||| target sentence
another source sentence ||| another target sentence
...
```

This format is required by the aligner that we are using in this script - [eflomal]

```
The cat ate the fish. ||| El gato comió el pez.
```

This is then passed on to eflomal to generate alignment indices. The following code is used.

```
eflomal-align -i eflomal_input.txt -f alignment_indices.txt
```

The alignment indices are generated in the following format :

```
4-1 2-2 3-4
```

To use this in the script, we use the Python subprocess module to run shell commands and capture output.

```
#generating alignment data
alignment_indices = subprocess.run(["eflomal-align -i eflomal_input.txt -f eflomal_output.txt"], \
                                   shell=True, capture_output=True, text=True)
indices = open('eflomal_output.txt', 'r').read().splitlines()
temp = [i.split(' ') for i in indices]
```

Next, both sentences need to be tagged using their respective monolingual word aligners. The monolingual aligner output contains various special characters like '^', '$' , etc, that need to be dealt with. For example, this is the raw output from the monolingual aligner for english.

```
chirag@chirag-linux:~$ echo "The cat ate the fish." | apertium eng-tagger
^The/The<det><def><sp>$ ^cat/cat<n><sg>$ ^ate/eat<vblex><past>$ ^the/the<det><de
f><sp>$ ^fish/fish<n><sg>/fish<n><pl>$^../..<sent>$
```

The following code performs tagging and deals with the special characters, while preserving the tagging data :

```
l_result = subprocess.run([f"echo '{i}' | apertium {l_lang}-tagger"], shell=True, capture_output=True, text=True)
r_result = subprocess.run([f"echo '{j}' | apertium {r_lang}-tagger"], shell=True, capture_output=True, text=True)
l_stems = l_result.stdout.split('^')
r_stems = r_result.stdout.split('^')
l_stems_pruned = [l_stems[i].strip()[:-1] for i in range(1, len(l_stems)-1)]
r_stems_pruned = [r_stems[i].strip()[:-1] for i in range(1, len(r_stems)-1)]
```

Finally, this tagged output is printed according to the alignment indices generated in the previous step.

This is the final tagged and aligned output. The alignment is completely dependent on the aligner itself and

```
cat/cat<n><sg> - el<det><def><m><sg>
The/The<det><def><sp> - gato<n><m><sg>
ate/eat<vblex><past> - comer<vblex><ifi><p3><sg>
fish/fish<n><sg>/fish<n><pl> - el<det><def><m><sg>
the/the<det><def><sp> - pez<n><m><sg>
```

I have **successfully completed** this task, and the source code can be accessed at this repository. I also experimented with other text aligners like efmaral and fast_align. The script was reviewed and approved by Daniel Swanson (@dangswan).

Previous Contributions

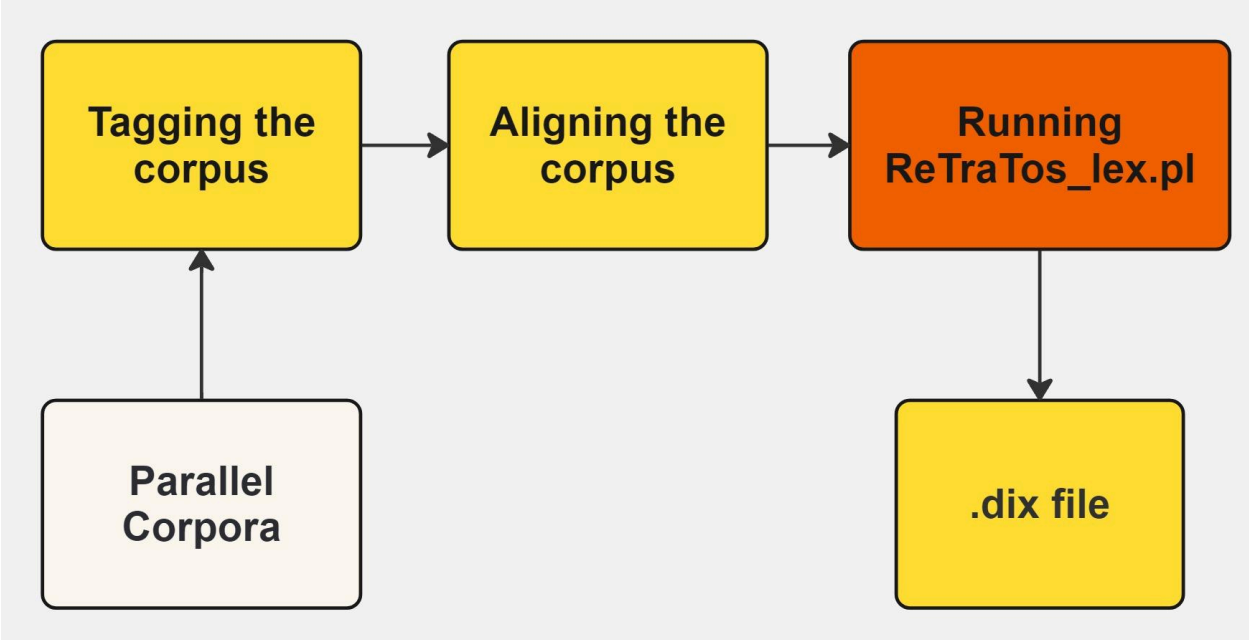
- Worked on the Apertium-English Dictionary [1]
- Disambiguated 500 tokens of text in Apertium-Eng [2]
- Worked on logo Design for UD-Annotatrix [3]
- Documented usage of the Apertium-separable module (Wiki Username: Daedalus) [4]
- Minor contributions to various wiki pages [5]

Proposal

Brief of Deliverables

The aim is to create a general program that can construct bidirectional dictionaries for a language pair, given a pair of parallel corpora - i.e., the same content in two different languages. ReTraTos was developed for this very purpose, but it is not very user-friendly at the moment and works in multiple steps.

The figure below illustrates the current flow of steps required to create a bidirectional dictionary from a set of two parallel corpora.



As we can see, multiple steps are required to build the final bilingual dictionary.

The objective is to build a modern alternative that does this job for the user in a single step and is much easier to use, with the workflow as illustrated below. The current plan is to implement this in python and later convert it into an executable binary, if required, that can be integrated into the rest of the Apertium ecosystem.

Mentors/Experienced members in Contact

Daniel Swanson (@dangswan)

Previously worked with Jonathan Washington and Tino Didriksen in Google Code-In 2019

Why should Google and Apertium sponsor it?

The internet can serve as a huge repository of translated corpora. A tool that can build and update dictionaries using parallel corpora can greatly accelerate the development of language pairs and improve the capabilities of existing language pairs. It eliminates the need for manually developing bidirectional dictionaries to a great extent.

Workplan

Timeline

Phase	Description of Work	Deliverable
Community bonding period: **May 1st -May 26th**	Read carefully through all Apertium docs related to language pair development	Familiarizing myself with the Apertium Environment
Week 1 to Week 3 **May 27th - June 16th**	Add the POS Tagger and the text aligner to the script. Experiment with various text aligners to see which gives the best results, starting with GIZA++. Other candidates include efmeral, elfomal, fast_align, etc.	The script, in its current state should be able to produce an aligned and tagged version of the parallel corpora.
Week 4 to Week 6 **June 17th to July 7th**	Extend functionality to perform Dictionary Induction. Essentially, rewrite the Retratos_lex.pl script and integrate it with the previous script.	A complete script that performs dictionary induction in one go.
Week 7 to Week 9 **July 8th to July 28th**	Perform comprehensive testing of the script. Convert it into an executable binary if required.	The final script and the binary if required.
Week 10 to Week 12 **July 29th to August 18th**	Buffer Period - Incorporate any further changes if required	Final script

Non-Summer of Code Plans

My final exams for the current semester will be completed by the first week of May 2024, and I have no plans other than GSoC for the summer of 2024. I can devote 20-30 hours per week, or more, as required to this project.

References

- https://wiki.apertium.org/wiki/ReTraTos
- http://www.nilc.icmc.usp.br/nilc/projects/retratos.htm
- https://sourceforge.net/p/retratos/code/HEAD/tree/