

Параллельный корпус

Подготовили студентки
первого курса ОП ФикЛ
Гребнева Валерия,
Худина Янина

Ссылка на репозиторий: <https://github.com/ghjuvas/pykili-project>

Описание проекта

Программа позволяет создать параллельный корпус на основе английских и французских текстов и работать с ним.

- Морфологическая разметка текста;
- Составление соответствий на основе морфологических критериев;
- Создание параллельного корпуса;
- Поиск по словоформе;
- Возможность поиска в собственном корпусе, соответствующем требованиям программы

Используются библиотеки:

- **ufal.udpipe** (english-partut-ud-2.5-191206.udpipe, french-partut-ud-2.5-191206.udpipe)
- **conllu**
- **json**
- **collections**

Начало работы

Приветственная фраза:

```
>>> [evaluate parallel_corpora.py]  
Приветствуем вас в нашей консольной программе!  
Она умеет создавать параллельные корпуса на основе текстов на английском и французском языках и  
осуществлять поиск по словоформе.
```

Выбор пользователя:

```
Вам будет предложено на выбор 3 функции.  
Введите '1', чтобы создать корпус  
Введите '2', чтобы искать по уже созданному корпусу  
Введите '3', чтобы программа закончила работу  
Ваш ответ: |
```

Ваш ответ: 1

Программа произведет поиск соответствий в выбранном текстовом файле и создаст корпус. Для того, чтобы программа прочитала ваш файл, требуется его особое оформление: метаинформация, две пустые строки, первый текст (на английском), две пустые строки, второй текст (на французском).

Введите имя файла или путь к нему: |

```
'results_same_words' - соответствия слов.
```

Формат текста

Текст для проверки:

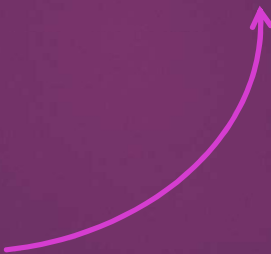
Text / Texte

I live in Paris. It is a beautiful city with a long history. I have many friends here.

J'habite à Paris. C'est une belle ville avec une longue histoire. J'ai beaucoup d'amis ici.

Программа разбивает текст на метаинформацию, текст на английском, текст на французском:

```
{  'full_text': [    'Text / Texte',  
                    'I live in Paris. It is a beautiful city with a long '  
                    'history. I have many friends here.',  
                    "J'habite à Paris. C'est une belle ville avec une longue "  
                    "histoire. J'ai beaucoup d'amis ici.\n"]}]}
```



Обработка данных

```
def tokenize_and_tag_texts(dict_texts):
    eng_model = Model.load('english-partut-ud-2.5-191206.udpipe')
    fr_model = Model.load('french-partut-ud-2.5-191206.udpipe')
    eng_pipeline = Pipeline(eng_model, 'generic_tokenizer', '', '', '')
    fr_pipeline = Pipeline(fr_model, 'generic_tokenizer', '', '', '')
    for language_key, primal_texts in dict_texts.items():
        tokenized_tagged_eng_text = eng_pipeline.process(primal_texts[1])
        tokenized_tagged_fr_text = fr_pipeline.process(primal_texts[2])
        dict_tokenized_tagged_texts = {'eng': tokenized_tagged_eng_text,
                                       'fr': tokenized_tagged_fr_text}
    # print(tokenized_tagged_eng_text)
    # print(tokenized_tagged_fr_text)
    # print(dict_tokenized_tagged_texts)
    return dict_tokenized_tagged_texts
```

```
{  'eng': '# newdoc\n'
    '# newpar\n'
    '# sent_id = 1\n'
    '# text = I live in Paris.\n'
    '1\tI\tI\tPRON\tPE\tNumber=Sing|Person=1|PronType=Prs\t2\tsubj\t_\t'
    '_\n'
    '2\tlive\tlive\tVERB\tV\tTense=Pres|VerbForm=Inf\t0\troot\t_\t_\n'
    '3\tin\tin\tADP\tE\t_\t4\tcase\t_\t_\n'
    '4\tParis\tParis\tPROPN\tSP\t_\t2\tobl\t_\tSpaceAfter=No\n'
    '5\t.\t.\tPUNCT\tFS\t_\t2\tpunct\t_\t_\n'
    '\n'
    ...}
```

Использование модели
english-partut-ud-2.5-191206.udpipe,
french-partut-ud-2.5-191206.udpipe

Получаем большой словарь
(представлен не полностью)

Парсим

```
def parse_conllu(tokenized_tagged_text):
    not_conllu_ud = conllu.parse(tokenized_tagged_text)
    list_tokens_with_tags = list()
    for token_list in not_conllu_ud:
        list_sent = list()
        for tokens in token_list:
            list_sent.append(tokens)
        list_tokens_with_tags.append(list_sent)
    return list_tokens_with_tags

def text_tokenization(dict_tokenized_tagged):
    dict_tokens = collections.defaultdict(list)
    for dict_language_key, dict_tagged_texts in dict_tokenized_tagged.items():
        for list_sent in dict_tagged_texts:
            forms_list = list()
            for ordered_dict in list_sent: #
                form = ordered_dict['form']
                forms_list.append(form)
            dict_tokens[dict_language_key].append(forms_list)
    # print(dict_tokens)
    return dict_tokens
```

На выходе получаем информацию о всех токенах в каждом предложении, а из самих словоформ составляем списки

Поиск соответствий

```
def build_accordance(dict_feat):
    words_alignment = list()
    list_of_sents_eng = dict_feat['eng']
    # print(list_of_sents_eng)
    list_of_sents_fr = dict_feat['fr']
    for id_list_eng, list_ord_dict_eng in enumerate(list_of_sents_eng):
        for id_list_fr, list_ord_dict_fr in enumerate(list_of_sents_fr):
            if id_list_eng == id_list_fr:
                accordance = match_words(list_ord_dict_eng, list_ord_dict_fr)
                words_alignment.append(accordance)
    # print(words_alignment)
    return words_alignment
```

```
def match_words(list_dicts_forms_tags_eng, list_dicts_forms_tags_fr):
    accordance_sent = list()
    for ord_dict_eng in list_dicts_forms_tags_eng:
        for ord_dict_fr in list_dicts_forms_tags_fr:
            equal_words = list()
            if ord_dict_eng['upostag'] == ord_dict_fr['upostag'] and
5             ord_dict_eng['feats'] == ord_dict_fr['feats']:
2             if ord_dict_eng['upostag'] == 'PUNCT' and ord_dict_eng['form']
5             == ord_dict_fr['form']:
                equal_words.append(ord_dict_eng['form'])
                equal_words.append(ord_dict_fr['form'])
                accordance_sent.append(equal_words)
            if ord_dict_eng['upostag'] != 'PUNCT':
                equal_words.append(ord_dict_eng['form'])
                equal_words.append(ord_dict_fr['form'])
                accordance_sent.append(equal_words)
    return accordance_sent
```

Результат поиска:

Выявлено 8 соответствий по словоформам.

Вы хотите увидеть список? Введите 'да' или 'нет'.

Ответ: да

Список соответствий:

```
1  ['in', 'à']
2  ['Paris', 'J']
3  ['.', '.']
4  ['is', 'est']
5  ['with', 'avec']
6  ['here', 'beaucoup']
7  ['here', 'd']
8  ['here', 'ici']
```

Вы хотите отдельно сохранить список соответствий? Введите 'да' или 'нет'.

Ответ: |



Запись корпуса

```
[{'Text / Texte': {'sentences': {'eng': ['I live in Paris.', 'It is a beautiful city with a long history.', 'I have many friends here.'], 'fr': ["J'habite à Paris.", "C'est une belle ville avec une longue histoire.", "J'ai beaucoup d'amis ici."]}, 'tokens': {'eng': [['I', 'live', 'in', 'Paris', '.'], ['It', 'is', 'a', 'beautiful', 'city', 'with', 'a', 'long', 'history', '.'], ['I', 'have', 'many', 'friends', 'here', '.']], 'fr': [['J', '', 'habite', 'à', 'Paris', '.'], ['C', '', 'est', 'une', 'belle', 'ville', 'avec', 'une', 'longue', 'histoire', '.'], ['J', '', 'ai', 'beaucoup', 'd', '', 'amis', 'ici', '.']]}, 'results_same_words': [['in', 'à'], ['Paris', 'J'], ['.', '.'], ['is', 'est'], ['with', 'avec'], ['here', 'beaucoup'], ['here', 'd'], ['here', 'ici']], 'full_text': ['Text / Texte', 'I live in Paris. It is a beautiful city with a long history. I have many friends here.', "J'habite à Paris. C'est une belle ville avec une longue histoire. J'ai beaucoup d'amis ici.\n"]}]}
```

Список, в который добавляются словари:

- ключ: метаинформация
- значения:
 - словарь предложений (на английском и французском отдельно)
 - словарь токенов
 - список найденных соответствий
 - весь текст (метаинформация, часть на английском, часть на французском)

Результат поиска словоформы

Введите '1', чтобы создать корпус

Введите '2', чтобы искать по уже созданному корпусу

Введите '3', чтобы программа закончила работу

Ваш ответ: 2

Выберите файл, в котором хранится ваш корпус. Например, corpora.json (обязательно сохраняя указанное расширение).

Убедитесь, что в нём есть необходимые данные:

'full text' - тексты на двух языках с метайнформацией,

'sentences' - тексты, токенизированные по предложениям,

'tokens' - предложения токенизированные по словам,

'results_same_words' - соответствия слов.

Имя файла: new_corpora.json

На каком языке вы собираетесь вводить словоформу? Введите 'английский' или 'французский'.

Язык: английский

Введите искомую словоформу: is

Найденные соответствия: [['is', 'est']]

Метайнформация текста: Text / Texte

Контекст, в котором встречается искомая словоформа:

I live in Paris.

J'habite à Paris.

Метайнформация текста: Text / Texte

Контекст, в котором встречается искомая словоформа:

It is a beautiful city with a long history.

C'est une belle ville avec une longue histoire.

Пример работы с программой

```
>>> [evaluate parallel_corpora.py]
Приветствуем вас в нашей консольной программе!
Она умеет создавать параллельные корпуса на основе текстов на английском и французском языках и осуществлять поиск по словоформе.
Вам будет предложено на выбор 3 функции.
Введите '1', чтобы создать корпус
Введите '2', чтобы искать по уже созданному корпусу
Введите '3', чтобы программа закончила работу
Ваш ответ: 1
Программа произведет поиск соответствий в выбранном текстовом файле и создаст корпус. Для того, чтобы программа прочитала ваш файл, требуется его особое оформление:
метаинформация, две пустые строки, первый текст (на английском), две пустые строки, второй текст (на французском).
Введите имя файла или путь к нему: test.txt
Выявлено 8 соответствий по словоформам.
Вы хотите увидеть список? Введите 'да' или 'нет'.
Ответ: да
Список соответствий:
1  ['in', 'à']
2  ['Paris', 'J']
3  ['.', '.']
4  ['is', 'est']
5  ['with', 'avec']
6  ['here', 'beaucoup']
7  ['here', 'd']
8  ['here', 'ici']
Вы хотите отдельно сохранить список соответствий? Введите 'да' или 'нет'.
Ответ: нет
Для записи корпуса вы хотите создать новый файл или работать с уже существующим? Введите '1' или '2' соответственно.
Ответ: 1
Выберите имя для создания нового файла. Например, new_corpora.json (обязательно сохраняя указанное расширение).
Имя файла: new_corpora.json
Корпус создан.
Вы хотите начать поиск по созданному корпусу, перейти в главное меню или выйти из программы?
Введите '1', '2' или '3' соответственно
Ваш ответ: 3
Конец работы.
>>> |
```

Что нужно доработать

- Считаем, что предложения по умолчанию выровнены;
- Соответствия слов производятся только на основе морфологии;
- Представлен один вид поиска

Планы на будущее

- Переделать поиск соответствий, используя другую библиотеку (stanza);
- Поработать на выравниванием предложений;
- Добавить другие виды
 - записи корпуса (возможно, csv)
 - поиска в корпусе
 - составления соответствий

Конец работы.