Правительство Российской Федерации

Государственное образовательное бюджетное учреждение

высшего профессионального образования

«Национальный исследовательский университет –

Высшая школа экономики»

Факультет **Гуманитарных наук**

Кафедра **Фундаментальной и компьютерной лингвистики**

**КУРСОВАЯ РАБОТА**

На тему: Формат KWIC для параллельных корпусов:   
пакет search\_kwic для Python.

Студентка группы **№ 152**

Терехина Мария Дмитриевна

Научный руководитель

Старший преподаватель факультета

гуманитарных наук Мороз Г.А.

Москва 2018

# Оглавление

[Оглавление 2](#_Toc513112095)

[1. Введение 3](#_Toc513112096)

[2. Разработка 4](#_Toc513112097)

[2.1. Встроенный переводчик 4](#_Toc513112098)

[2.2. Автоматическое выравнивание текстов 6](#_Toc513112099)

[3. Литература 11](#_Toc513112100)

# ****Введение****

KWIC (Key Word in Context) – это формат представления конкордансов (списка контекстов искомого слова), при котором ключевое слово располагается по центру, слева и справа от него – контекст, помещающийся в строку. Формат KWIC используется многими корпусами, однако ни один из известных нам параллельных корпусов не поддерживает данный формат в полной мере. Например, в параллельном подкорпусе Национального корпуса русского языка в формате KWIC отображаются только контексты на языке запроса[[1]](#footnote-1). Это связано с тем, что нахождение в параллельном тексте фрагмента, соответствующего тексту запроса (далее этот фрагмент текста будет именоваться *переводом*) – достаточно нетривиальная задача.

В 201\_ году в Школе Лингвистики Факультета гуманитарных наук НИУ ВШЭ стартовал проект Lingcorpora, цель которого – создание пакета для языков программирования Python и R, содержащего API для корпусов текстов. На данный момент пакет для Python включает корпуса следующий языков: \_, для R: \_. Пакет находится в открытом доступе. Для Python его можно скачать через pip, для R \_.

Поскольку Lingcorpora включает API для параллельных корпусов, появилась необходимость в формате KWIC и для них, для унификации выдачи. Предварительный поиск литературы показал, что публикаций или проектов на эту тему нет. В связи с этим разработка алгоритма квикизации производилась с нуля. Подробно этапы разработки алгоритма будут описаны в разделе 2.

Поскольку квикизация параллельного текста – не обязательное условие работы параллельных корпусов в Lingcorpora, было решено вынести этот функционал в отдельный пакет, также находящийся в открытом доступе с возможностью установки через pip.

# Разработка

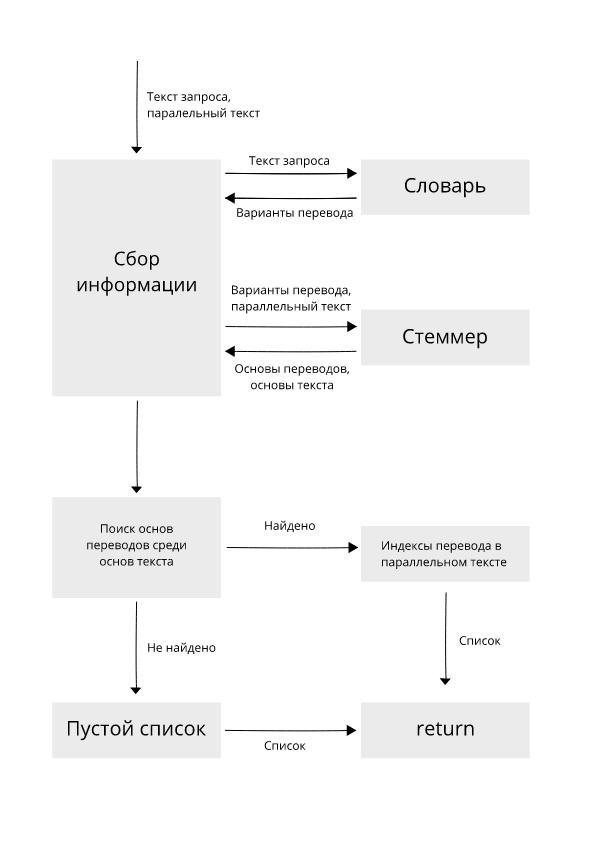
Алгоритм квикизации подразумевает нахождение в параллельном тексте перевода запроса. Архитектура Lingcorpora требует также нахождения индексов перевода, что не составляет труда после того, как сам перевод найден.

В ходе разработки алгоритма было рассмотрено три принципиальных подхода к решению этой задачи: встроенный переводчик, автоматическое выравнивание текстов, программа на основе моделей универсальных зависимостей. Их устройство и причины отказа от первых двух из них будут описаны в этом разделе.

* 1. Встроенный переводчик

Данный алгоритм предельно прост и устроен следующим образом:

1. Программа получает на вход слово, перевод которого нужно найти в параллельном тексте и сам параллельный текст.
2. Происходит обращение к онлай-словарю, в качестве ответа получаем список возможных переводов данного слова.
3. Переводы передаются в стеммер, в результате получаем основы переводов. Все предложение также передается в стеммер.
4. В списке основ предложения производится поиск основ переводов.
5. Если основа найдена в предложении, слово объявляется переводом, возвращаются его индексы (индексы начала и конца) в параллельном тексте. Если не найдена, возвращается пустой список (см. Рисунок 1).

Рисунок 1

В качестве онлайн-словаря был выбран сервис Яндекс.Словари (<https://tech.yandex.ru/dictionary/>). Для этого было несколько причин, в том числе бесплатный доступ к API (с ограничением в 10 000 обращений в сутки) и простота использования. Сервис работает с get- и post-запросами, возвращает автоматически сгенерированную словарную статью в формате JSON, JSONP или XML (более подробно см. <https://tech.yandex.ru/dictionary/doc/dg/reference/lookup-docpage/>). Дальше страница обрабатывается xml-парсером BeautifulSoup, достаются все варианты перевода.

В качестве стеммера использовался nltk.stem.snowball. Это алгоритм, который следуя ряду правил последовательно отсекает от основы аффиксы, принимая во внимание особенности языка. В данном пакете есть стеммеры для 15 языков, включая русский, однако нет стеммера, например, для польского, что вынуждает для некоторых языков искать другие алгоритмы выделения основы.

Отказаться от этого метода было решено по нескольким причинам: во-первых, в такой реализации алгоритм имел очень много внешних зависимостей. Пакет nltk часто вызывает проблемы при установке, поэтому иметь такую зависимость кажется нецелесообразным. Во-вторых, в nltk.stem.snowball сравнительно небольшое количество языков. Это значит, что при добавлении каждого нового языка приходилось бы дополнительно искать алгоритмы стемминга для этих языков. Очевидно, для очень многих языков такого функционала просто не существует. В-третьих, сервис Яндекс.Словари ограничивает число запросов в сутки 10 тысячами, что при интенсивном использовании достаточно малое количество. И наконец, всем эти сложностям можно было бы найти решение, если бы алгоритм эффективно работал. Однако алгоритм давал непустой ответ лишь примерно в 25% случаев[[2]](#footnote-2).

* 1. Автоматическое выравнивание текстов

На втором этапе разработки алгоритма была рассмотрена возможность применения программы Fast Align для автоматического выравнивания текстов. Это алгоритм статистического машинного перевода, основанный на модели «шумного канала» (Noisy Chanel model). Модели такого типа применяются в следующей ситуации: представьте, что вы собирались написать какое-то слово *w*, состоящее из *k* букв. Однако получилось у вас слово *n* из *l* букв (текст как бы прошел через шумный канал» и исказился). Модель шумного канала будет решать задачу предсказания слова *w* по получившемуся слову *n*. Математически модель записывается следующим образом:

Т.е. предсказанное слово – это слово, вероятность которого быть исходным словом при отображении на письме n максимальна.

Применяя теорему Байеса получаем:

Последнее преобразование справедливо, поскольку знаменатель не зависит от w.

Наша модель утроена абсолютно аналогичным образом. Предположим, мы хотим перевести предложение с французского на английский. В таком случае мы будем искать решение следующего уравнения:

.

.

Плюс такой модели в том, что мы можем использовать множитель для увеличения вклада грамматики языка, на который осуществляется перевод. Проблема в том, что определить напрямую невозможно. Для решения этой задачи в модель вводятся дополнительные переменные выравнивания: , которые могут принимать значения , где m – количество слов в предложении (оригинале), – количество слов в предложении (переводе). Такая модель, которая включает дополнительные переменные выравнивания называется IBM model 2 (IBM-M2).

**Определение**. IBM-M2 model состоит из конечного множества английский слов *E*, множества французских слов , переменных *M* и *L* определяющих максимальную длину французских и английских предложений соответственно. Модель описывается следующими параметрами:

* для каждого . Параметр может быть интерпретирован как условная вероятность перевести французское слово как английское слово
* для каждого Параметр может быть интерпретирован как вероятность переменной выравнивания принять значение , учитывая длины и французского и английского предложений.

**Пример**. Чтобы пояснить все вышесказанное, рассмотрим два предложения, где

В таком случае переменные выравнивания задают следующее выравнивание:

*Le ⇒ the   
Programme ⇒ program   
 a ⇒ has   
 ete ⇒ been   
 mis ⇒ implemented   
 en ⇒ implemented   
application ⇒ implemented*

**Предположение о независимости.** Для работы IBM-M2 необходимо сделать два очень сильных предположения о независимости:

* зависит только от переменных *L* и *M* (и не зависит от слов в английском предложении и других переменных выравнивания).
* Значение зависит только от то есть только от английского слова, которому слово сопоставлено.

**Применение IBM-M2**. Рассматриваемая модель может быть применена на корпусе текстов, где нет приписанных переменных выравнивания в тренировочных данных.

Введем параметры:

* – количество раз, когда слово *e* сопоставлено слову *f* в тренировочных данных.
* – количество раз, когда слово e сопоставлено какому-либо слову в тренировочных данных.
* – количество раз, когда нам встретилось английское предложение длины и параллельное ему французское предложение длины , где слово французское слово сопоставлено английскому слову .
* – количество раз, когда нам встретилось английское предложение длины и параллельное ему французское предложение длины m.

Тогда согласно методу максимального правдоподобия[[3]](#footnote-3) наши параметры и (см. выше) будут равны:

POS теги - <http://universaldependencies.org/docs/u/pos/index.html>

Dependency marks - <http://universaldependencies.org/docs/u/dep/index.html>

Статья : <http://ufal.mff.cuni.cz/~straka/papers/2017-conll_udpipe.pdf>

# Литература

Collins 2011 — Collins M. Statistical machine translation: IBM models 1 and 2 //Columbia Columbia Univ. – 2011.

1. Пример запроса в английский параллельный подкорпус НКРЯ, выдача в формате KWIC:  
   <http://search1.ruscorpora.ru/search.xml?mycorp=(lang%3A%22eng%22%20%7C%20lang_trans%3A%22eng%22)&mysent=&mysize=24681277&mysentsize=1608376&dpp=&spp=&spd=&text=lexform&mode=para&sort=gr_tagging&env=alpha&req=linguistics&out=kwic> [↑](#footnote-ref-1)
2. В момент разработки алгоритма не стояло задачи дать точную оценку эффективности его работы, поскольку было очевидно, что он не будет использоваться в качестве финальной версии. [↑](#footnote-ref-2)
3. Метод максимального правдоподобия (maximum likelihood method) [↑](#footnote-ref-3)