МЕТОДЫ АВТОМАТИЗАЦИИ ПОСТРОЕНИЯ И ПОПОЛНЕНИЯ ДВУЯЗЫЧНЫХ СЛОВАРЕЙ С ИСПОЛЬЗОВАНИЕМ КОРПУСОВ ПАРАЛЛЕЛЬНЫХ ТЕКСТОВ

METHODS OF AUTOMATION OF CREATING AND WIDENING OF BILINGUAL DICTIONARIES USING PARALLEL TEXT CORPORA

А.А. Липатов *(anton.lipatov@gmail.com)*
Новосибирский государственный университет

А.А Мальцев *(amalcev@mail.ru.),*
Институт математики СО РАН

В настоящей работе рассматривается подход, позволяющий автоматизировать процесс построения двуязычных словарей. Для построения используются уже затраченный труд переводчиков: двуязычный корпус параллельных текстов.

Введение

В настоящей работе рассматривается подход, позволяющий автоматизировать процесс построения двуязычных словарей. В частности, данный подход оказывается полезным при расширении семантических сетей типа WordNet. Данная работа является продолжением работы [1].

Для решения задачи используется уже затраченный труд переводчиков. Для построения используются двуязычный корпус параллельных текстов. Суть алгоритмов для каждого текста и его перевода состоит в нахождении соответствий между семантическими единицами в парных текстах. Это требует последовательного разбиения и анализа парных текстов. Решение задачи делится на три этапа: разбиение текста на предложения, выравнивание текста, собственно построение словаря. Предлагаются различные алгоритмы для каждого из этапов.

Постановка задачи

Пусть есть некоторый двуязычный корпус параллельных текстов. Рассматривается проблема извлечения из этого корпуса словарной информации и построения межъязыковых словарей. Словари могут быть либо традиционными (межъязыковое сопоставление слов и словосочетаний), либо представлять собой сопоставления семантических единиц тезаурусов типа WordNet для разных языков.

Целью данной работы было более подробно изучить возможности решения данной задачи автоматически или полуавтоматически.

Основные предположения

Для упрощения исходной задачи мы используем некоторые предположения, которые подобраны таким образом, чтобы вместе с упрощением задачи минимально ограничить общность.

Предполагается, что выполнено предположение о монотонности перевода, т.е. при переводе порядок предложений не изменяется. Допускаются пропуски кусков текста.

Предполагается, что мы располагаем достаточно большим корпусом текстов. Исходя из этого, мы не ставим перед собой задачу использования информации из всех предложений. Более приоритетным является качество извлекаемой информации.

Предполагается, что тексты переведены качественно.

Мы перечислили только основные предположения. На каждой стадии решения задачи, для каждого алгоритма, далее будет предложено использовать ещё некоторые предположения и ограничения.

Общий алгоритм решения

При решении поставленной задачи можно чётко выделить три этапа:

1. Разбиение текста на предложения
2. Выравнивание текста
3. Собственно построение словаря

Важно отметить, что для каждого этапа можно выбирать алгоритмы независимо от других. Тем самым, можно учесть специфику текстов и использовать алгоритмы с нужными параметрами.

Разбиение текста на предложения

Цель данного этапа – перейти от текста к упорядоченному списку предложений. В работе [1] был описан простой алгоритм разбиения на предложения, который последовательно идентифицировал концы предложений по знакам препинания. Однако этот алгоритм работает удовлетворительно не всегда. Проблемы чаще всего возникают в предложениях с сокращениями, когда следующее слово начинается с заглавной буквы. Назовём такой случай спорной ситуацией.

Было рассмотрено несколько подходов к решению данной проблемы. Их можно применять в зависимости от того, какая информация у нас есть, а также комбинировать.

1. Создать специальный словарь слов-сокращений и считать в спорных ситуациях сокращениями те слова, которые содержатся в нём.
2. Проверять, содержатся ли в словаре общей лексики все слова из пяти и менее символов, которые встретились в спорных ситуациях. Если не содержатся, считать их сокращениями. Все слова длины более 5 символов в спорных ситуациях не считать сокращениями.
3. Для каждого слова с точкой на конце в спорной ситуации рассмотреть все вхождения этого слова в тексте. Если среди этих вхождений подавляющее большинство таких, что следующий символ – точка, то считать это слово сокращением. Здесь мы используем предположение, что в тексте обычно не встречаются слова, находящиеся исключительно в конце предложений, что может быть и не верно.

Если у нас есть какие-то знания о языке (в частности, словарь слов-сокращений или словарь общей лексики), то разумно использовать подходы 1 и 2. В случаях, когда точно нельзя дать ответ в спорной ситуации (когда слово не содержится ни в одном из словарей или содержится сразу в двух), можно привлекать третий подход. Если никаких данных о языке у нас нет, то остаётся использовать только третий подход.

Кроме того, заметим, что в данном случае мы на самом деле решаем не общую задачу разбиения текста на предложения, а её частный случай. Для каждого текста у нас есть его эквивалент на другом языке. Этим можно воспользоваться, предположив, что если в тексте на одном языке есть сокращения, то они должны быть и в его переводе на другой язык.

Выравнивание текстов

Сначала рассмотрим алгоритмы выравнивания текстов для решения задачи с текстами на неизвестных языках. В настоящее время существуют несколько алгоритмов, которые позволяют производить выравнивание текстов, не обладая никакой информацией о языках.

В работе [2] рассматривается подход к выравниванию текста, основанный на простой статистической модели длин предложений. В качестве главного предположения используется факт существования некоторой положительной корелляции между длинами предложений в исходном тексте и в переводе. В рамках данного подхода каждой паре предложений из текстов на разных языках, т.е. каждому возможному соответствию, соответствует некоторая *характеристика возможности сопоставления*. Эта характеристика вычисляется на основе разницы длин предложений (в символах, включая пробелы) и дисперсии этой разницы. Далее, с помощью методов динамического программирования, находится такое соответствие предложений, при котором характеристики возможности сопоставления максимальны.

В работе [3] рассматривается алгоритм выравнивания предложений в двуязычном корпусе текстов, использующий лексическую информацию. Производится построение простой статистической модели “слово-перевод” в процессе выравнивания. Проводится поиск такого сопоставления предложений, при котором вероятность существования корпуса с такой моделью перевода максимальна. Для начального построения и совершенствования модели перевода используются заранее сопоставленные предложения.

Кроме этих методов, при отсутствии какой-либо информации о языках текстов, можно использовать подходы к выравниванию, описанные в работе [1]. Алгоритмы, основанные на этих подходах, используют «непереведённые» слова (т.е. слова с идентичным написанием в текстах на разных языках, в т.ч. числа, даты и т.д.), некоторые знаки препинания.

Теперь рассмотрим случай, когда у нас имеется достаточно информации о языках текстов. В этом случае мы можем использовать алгоритм с однозначно переводимыми словами, описанный в [1]. Однако он использует только часть словаря общей лексики (однозначно переводимые слова) и не всегда применим. Был разработан другой алгоритм, которой использует словарь общей лексики полностью.

Использование словаря общей лексики

Итак, на входе мы имеем текст на одном языке и его перевод на другой. Необходимо сопоставить предложения в тексте и переводе. Будем использовать двуязычный общелексический словарь (представляет собой, по сути, сопоставление словам и словосочетаниям на одном языке множеств их эквивалентов на другом).

Сразу ограничим область наших поисков – будем рассматривать только следующие типы сопоставлений:

1. Одному предложению в исходном тексте соответствует одно предложение в переводе (соответствие «1-1»).
2. Одному предложению в исходном тексте соответствует два последовательно стоящих предложения в переводе (соответствие «1-2»).

3. Двум последовательно стоящим предложениям в исходном тексте соответствует одно предложение в переводе (соответствие «2-1»).

Введём некоторый **коэффициент сопоставления предложений**, который будет определяться для каждой пары предложений из текста и перевода. Мы хотим получить коэффициент, лежащий в интервале [0,1]. В случае, когда в паре предложения эквивалентны (т.е. одно является переводом другого), значение коэффициента должно быть максимально. Чем больше эквивалентных слов и словосочетаний присутствует в паре предложений на разных языках, тем больше значение коэффициента. Когда мы зададим этот коэффициент для любой пары, будем пытаться найти пары с наибольшим значением этого коэффициента.

Сначала для любого предложения определим понятие **разбиения**. Рассмотрим множество всевозможных разделений предложения на слова и словосочетания, которые входят в словарь. Из всех этих разделений выберем те, которые содержат наиболее длинное из найденных словосочетаний, будем далее рассматривать только их. Далее из них выберем те, которые содержат второе по величине словосочетание. И так далее, продолжаем процесс до тех пор, пока не останется единственное разделение, его назовём разбиением. В случае появления неоднозначностей, когда встречаются различные максимальные по длине словосочетания, выбираем произвольно. Длина словосочетания есть число слов, в него входящих.

Итак, у нас имеется некоторое предложение из текста и предложение из его перевода. Зафиксируем некоторое разбиение предложения из текста. Число элементов в нём обозначим через E. Рассмотрим множества переводов для каждого элемента этого разбиения. Эти множества, как и элементы разбиения, состоят из слов и словосочетаний. Обозначим через E_t число элементов зафиксированного разбиения, множества переводов которых имеют непустое пересечение хотя бы с одним возможным разбиением предложения из перевода. Далее, по аналогии, зафиксируем некоторое разбиение предложения из перевода. Число элементов в нём обозначим через R. Рассмотрим множества переводов в обратную сторону для каждого элемента этого разбиения. Обозначим через R_t число элементов зафиксированного разбиения предложения из перевода, множества переводов (в обратную сторону) которых имеют непустое пересечение хотя бы с одним возможным разбиением предложения из исходного текста. Когда мы говорим «пересечение с разбиением», рассматриваем разбиение как обычное множество.

Теперь мы можем получить формулу для вычисления коэффициента сопоставления предложений, удовлетворяющую указанным выше требованиям:

$$coeff = \frac{E_t + R_t}{E + R}$$

Можно предложить и другие формулы вычисления данного коэффициента.

Рассмотрим теперь **матрицу сопоставления**, которая представляет собой матрицу коэффициентов для всевозможных пар предложений. Кроме собственно предложений, будем рассматривать и пары рядом стоящих предложений. Таким образом, каждой нечётной строке (столбцу) матрицы с номером $2k-1$ (где $k \geq 1$) соответствует предложение из текста (перевода) с номером k. Каждой чётной строке (столбцу) матрицы с номером $2k$ (где $k \geq 1$) соответствует пара рядом стоящих предложений из текста (перевода) с номерами k и $k+1$.

Теперь нам нужно найти некоторое сопоставление предложений (пар) с наибольшими значениями коэффициента сопоставления. В терминах матрицы сопоставления, нам нужно выделить некоторое подмножество клеток, которое будет удовлетворять следующим условиям:

1. В каждой строке и столбце может быть не более одной выделенной клетки. Т.к. каждому предложению (паре предложений) должно сопоставляться не более одного предложения (пары).
2. Для каждой выделенной клетки соседние строки и столбцы не должны содержать выделенных клеток. Каждому предложению из текста, по сути, сопоставлена не только соответствующая строка, но и две соседние, в которые это предложение входит на пару с рядом стоящими. Если выделенная клетка находится на нечётной строке, то выбрано для сопоставления само предложение, и его пары с соседними уже не могут быть сопоставлены, т.е. соседние строки должны быть пусты. Если же выделенная клетка находится на чётной строке, то выбрана для сопоставления пара рядом стоящих предложений, и они сами уже не могут быть сопоставлены, т.е. соседние строки опять же должны быть пусты. Аналогичные рассуждения приводят к тому же выводу в случае со столбцами.
3. Введём отношение порядка между двумя клетками, заданными своими координатами. $(x_1, y_1) \prec (x_2, y_2) \Leftrightarrow (x_1 < x_2) \,\&\, (y_1 < y_2)$. В силу предположения о монотонности перевода множество выделенных клеток с указанным порядком должно быть линейно упорядочено.
4. Значения в выделенных клетках должны быть больше некоторой пороговой величины. Сумма значений в выделенных клетках должна быть максимальна. Таким образом, мы добиваемся наилучшего сопоставления.

Итак, мы перешли к математической формулировке задачи максимизации. Данная задача может быть решена методами динамического программирования. Однако вычисление всей матрицы сопоставления для достаточно больших текстов не представляется возможным вследствие больших временных затрат. Поэтому вместо решения задачи на всей матрице предлагается решать её на некоторой окрестности диагонали.

В силу пункта 4 для попадания в искомое множество необходимо преодоление некоторого порога, поэтому нас не интересуют значения меньше пороговой величины. Для упрощения и наглядности мы можем заменить их

нулями. Эмпирически было установлено, что только порядка 5% клеток преодолевают порог в 30% (значение коэффициента 0,3). В идеальном случае, когда в переводе текста содержатся последовательно переводы всех предложений, искомое множество будет целиком лежать на диагонали.

Для ускорения процесса решения и уменьшения количества ошибок, полезно учитывать некоторое предположение о длине предложения и его перевода: при переводе предложение не может увеличиться в длине или укоротиться более чем в *L* раз.

Для улучшения результатов алгоритм можно разделить на две ступени. Сначала производить сопоставление на уровне абзацев (аналогичным образом, рассматривая не предложения, а абзацы; производить разбиения всё равно внутри предложений), а потом уже на уровне предложений в каждой паре сопоставленных абзацев.

Описанный алгоритм подходит для сопоставления предложений, которые не имеют «явных зацепок», которые можно было бы использовать (знаки препинания, непереведённые слова и т.д.). Кроме того, при сопоставлении с помощью других алгоритмов, полезно вычислять коэффициент сопоставления и проверять, превосходит ли он пороговое значение.

Необходимо отметить, что даже при наличии знаний о языках текстов зачастую данный алгоритм не даёт блестящих результатов. Это связано с тем, что качество работы алгоритма целиком зависит от качества используемого словаря общей лексики.

Пополнение словаря

Для создания (пополнения) словаря на основе множества сопоставленных предложений и их переводов используются статистические методы. Основной алгоритм описан в работе [1], однако он не позволяет автоматически выделять словосочетания. Была выполнена его модификация.

Суть исходного алгоритма состоит в следующем. Рассматривается отдельно каждая сопоставленная пара. Для *i*-й пары определяется множество начальных форм новых слов исходного текста *eWordSet(i)={eWord(i, 1), ..., eWord(i, n(i))}* и множество начальных форм всех слов перевода *rWordSet(i)={rWord(i, 1), ..., rWord(i, m(i))}*. Кроме того, рассматриваются объединения соответствующих множеств: *eWordSet = eWordSet(1)∪ ...∪ eWordSet(n)* и *rWordSet = rWordSet(1)∪ ...∪ rWordSet(m).*

Далее проводится сопоставление слов из множеств *eWordSet* и *rWordSet с помощью* ***функции возможного сопоставления слов:***

$$wordCorresponding(eWord, rWord) = \\ = \frac{|\{i : eWord \in eWordSet(i) \wedge rWord \in rWordSet(i)\}|}{|\{i : eWord \in eWordSet(i)\}|}$$

Последовательно фиксируется каждое слово *eWordSelected∈eWordSet,* для него рассматриваются значения функции *wordCorresponding(eWordSelected, rWord)*, где *rWord∈rWordSet* и у *rWord* подходящая часть речи. Далее перевод *eWord* выбирается из всех слов, на которых достигается максимум функции.

Будем называть **фразой** упорядоченный список начальных форм слов. Зададим функцию **возможного сопоставления фраз**:

$$phraseCorresponding(ePhrase, rPhrase) = \\ = \frac{|\{i : eSub(ePhrase, i) \wedge rSub(rPhrase, i)\}|}{|\{i : eSub(ePhrase, i)\}|}$$

Предикат *eSub(ePhrase,i)* истинен тогда и только тогда, когда словосочетание *ePhrase* может входить в i-тое предложение. То есть последовательность начальных форм слов в *ePhrase* является подпоследовательностью некоторых начальных форм слов в i-том предложении. Предикат *rSub(rPhrase,i)* определяется аналогично.

Заметим, что если фразы *ePhrase* и *rPhrase* состоят из единственных слов *eWord* и *rWord* соответственно, то выполняется равенство:

$$phraseCorresponding(ePhrase, rPhrase) = \\ = wordCorresponding(eWord, rWord)$$

Таким образом, мы можем далее рассматривать везде только функцию *phraseCorresponding*, заменяя, где нужно, каждое отдельное слово фразой, состоящей только из этого слова.

Для каждого слова *eWord∈eWordSet* ищем всевозможные слова и группы слов, которые входят неразрывно вместе с *eWord* в предложения из текста не менее *N* раз (в том числе фраза, состоящая из единственного слова e*Word*). Получаем множество *ePhraseSet*. Аналогично для каждого слова *rWord∈rWordSet* ищем всевозможные слова и группы слов, которые входят неразрывно вместе с *rWord* в предложения из перевода текста не менее *N* раз (в том числе фраза, состоящая из единственного слова *rWord*). Получаем множество *rPhraseSet*. Константу *N* следует выбирать не меньше трёх.

Далее поступаем так же, как и в исходном алгоритме. Последовательно фиксируем каждую фразу *ePhraseSelected∈ePhraseSet,* для неё рассматриваем значения функции *phraseCorresponding(ePhraseSelected,*

rPhrase), где *rPhrase*∈*rPhraseSet*. Перевод *ePhrase* выбираем из всех фраз, на которых достигается максимум функции. Если этот максимум меньше некоторого порогового значения, то пару с *ePhrase* в словарь не добавляем. В этом случае получается, что алгоритм на нашёл перевода для *ePhrase*, либо *ePhrase* не является устойчивым словосочетанием в тексте.Если же максимум больше порогового значения, то преимущество при выборе перевода будет у фраз, которых нет в словаре. Если с учётом этого выбор сделать нельзя, подходящее словосочетание предлагается выбрать вручную.

Сгенерированный таким образом словарь, безусловно, требует ручной проверки человеком. Из-за того, что мы добавляли в *ePhraseSet* всевозможные наборы слов, которые могут не являться словосочетаниями, словарь требует чистки. Если где-то перевод слова или словосочетания был найден некорректно, то можно выбрать другой из тех, что имеют достаточно большие значения функции сопоставления.

Реализация алгоритмов

Описанные алгоритмы были реализованы на языке программирования Visual C#. В качестве англо-русского словаря с общей лексикой использовался словарь Мюллера. Использовался морфологический модуль проекта «Автоматическая Обработка Текста» (http://www.aot.ru/).

Анализ результатов работы

Тестирование алгоритмов проводилось на англо-русском корпусе текстов различных тематик: математика, информационные технологии, биология, химия, физика и др. Кроме этого, рассматривались тексты по гуманитарным наукам (философия и др.) и литературные тексты. В результате изменений в алгоритме разбиения текста на предложения в некоторых текстах стало распознаваться корректно на 1-2% предложений больше по сравнению с исходным алгоритмом. Столь незначительное улучшение результатов объясняется небольшим количеством спорных ситуаций с сокращениями при разделении текста на предложения. При использовании словаря общей лексики сопоставляется в среднем порядка 70% предложений (в исходном алгоритме было 60%). Если перевод близок к тексту (что очень часто встречается в технических текстах), то процент сопоставления может быть свыше 95%. Модифицированный алгоритм сопоставления находит практически все словосочетания, однако из-за его указанных особенностей приходится проводить вручную чистку полученного словаря. Несмотря на это, убрать лишние слова из словаря всё же проще, чем построить его с нуля.

Заключение

При выполнении работы исследовалась возможность автоматизации процесса построения (пополнения) двуязычных словарей с использованием корпуса параллельных текстов. Задача была разбита на подзадачи, для решения каждой из которых были разработаны алгоритмы. В процессе работы были предложены улучшения существующих алгоритмов и предложены новые. Для большинства алгоритмов выполнена программная реализация. Проведен анализ и сравнительная оценка полученных результатов.

Список литературы

1. *Липатов А., Мальцев А., Шило В.* Автоматизация процесса построения и пополнения двуязычных специализированных словарей. // Труды конференции «Диалог». М.: 2005.
2. *Gale W.*, Church K. A program for Aligning Sentences in Bilingual Corpora
3. *Chen S*. Aligning Sentences in Bilingual Corpora using Lexical Information
4. *Гельфейнбейн И.Г., Гончарук А.В., Лехельт В.П., Липатов А.А., Шило В.В.* Автоматический перевод семантической сети WORDNET на русский язык. // Труды конференции «Диалог». М.: 2003.
5. *Lipatov A., Goncharuk A., Helfenbein I., Shilo V., and Lehelt V*. Automatic Creation of Non-English WordNet-like Lexical *Databases // Lecture Notes in Computer Science, Papillon Workshop 2003.*
6. *Липатов А.А., Шило В.В.* Автоматизация процесса создания и пополнения специализированных словарей // Сборник тезисов XLII Международной студенческой конференции «Студент и научно-технический прогресс». Новосибирск: 2004.
7. *Каневский Е.А*. Некоторые вопросы пополнения морфологического словаря терминами предметной области // Труды конференции «Диалог». М.: 2001.
8. *Игитов С.В., Крючкова Е.Н., Старцев П.Л., Чумак М.В.* Проблемы формирования словарей в системах искусственного интеллекта.
9. *Поминов А.В.* Некоторые вопросы построения многоязычных автоматических словарей.