ГЕНЕТИЧЕСКИЙ АЛГОРИТМ ДЛЯ АВТОМАТИЧЕСКОГО РАЗБИЕНИЯ СЛОВ НА МОРФЕМЫ

GENETIC ALGORITHM FOR AUTOMATIC DIVISION OF WORS INTO MORPHEMES*

Гельбух А.Ф.[1] *(www.gelbukh.com),* ***Сидоров Г.О.***[1] *(www.cic.ipn.m/~sidorov),* ***Лара-Рейес Д.***[1]*,*
Чанона-Эрнандес Л.[2]*,* ***Чубукова М.В.***[3] *(licht66@mail.ru)*
[1]*Лаборатория естественного языка и обработки текста,*
Центр Компьютерных Исследований (CIC),
Национальный Политехнический Институт (IPN), г. Мехико, Мексика
[2]*Инженерный факультет (механика, электричество) (ESIME),*
Национальный Политехнический Институт (IPN), г. Мехико, Мексика
[3]*Кафедра филологического образования,*
Московский институт открытого образования,
Москва, Россия

В статье обсуждается независимая от примеров (unsupervised) техника определения морфемной структуры слов во флективных языках на примере испанского языка. Мы используем глобальную оптимизацию, реализованную с применением генетического алгоритма, без каких-либо эвристик или допущений, уменьшающих размерность задачи a priori. Приводится описание алгоритма. Дается предварительная оценка результатов. Данные на входе представляют собой список слов, построенный на основе корпуса или словаря, а данные на выходе – список тех же слов, разделенных на морфемы. Как и многие автоматические методы, такой алгоритм не претендует на нахождение стопроцентно точного решения и требует ручной постобработки. Тем не менее он позволяет быстро обнаружить тенденции в данных и получить предварительные результаты без больших затрат ручного труда.

1. Введение

В статье обсуждается независимая от примеров (*unsupervised*) техника определения морфемной структуры слов во флективных языках на примере испанского языка. Мы используем глобальную оптимизацию, реализованную с применением генетического алгоритма. Отличие от предыдущих подходов состоит в том, что предыдущие подходы основаны на некотором наборе дополнительных эвристик и предположений, посчитанных для условных вероятностях частей слов, для уменьшения размерности задачи (Goldsmith, 2001). Мы же используем более простую модель, в которой не требуются дополнительные эвристики, применение которых отбрасывает многие возможные варианты решения, которые могут быть правильными. Кроме того, с эвристиками существует риск того, что метод окажется «запертым» в локальном максимуме, поскольку другие возможности просто не рассматриваются. Естественно, у такого более универсального подхода есть и свои недостатки, скажем, его труднее сделать применимым к конкретной задаче, или он работает больше времени.

Напомним, что генетический алгоритм это метод оптимизации (часто применяемый, например, в искусственном интеллекте), взявший за основу модель эволюции, когда лушее решение «выводится», комбинируя свойства индивидов и отбирая лучших индивидов для последующей обработки.

На самом деле, применение методов, независимых от примеров (*unsupervised*), оправдано для определенных классов задач, когда нет возможности решить задачу полным перебором, а применение эвристик требует дополнительных допущений. Тогда можно попытаться сформулировать задачу в форме, подходящей для какого-либо метода, независимого от примеров (*unsupervised*), это может быть генетический алгоритм, или моделирование кристаллизации (*simulated annealing*) или оптимизация потока частиц (*particle swarm optimization*), и пр. То есть, речь не о том, что это «модные» методы, или что все задачи должны решаться с их помощью, а о том, что

* Work done under partial support of Mexican Government (CONACyT, SNI) and National Polytechnic Institute, Mexico (SIP, COFAA, PIFI).

для некоторые задачи, которые неразрешимы обычными методами, могут быть разрешены с использованием таких методов оптимизации.

В наших экспериментах мы используем испанский язык, который имеет довольно простую морфологическую структуру, поэтому в алгоритме учитывается не более трех возможных морфем. Окончания мы не рассматриваем, отделяя их на этапе предобработки. С некоторыми изменениями наш алгоритм применим к любому флективному языку, в том числе и к русскому.

Заметим также, что мы работаем в рамках словообразовательного подхода, который отличается от более традиционного словоизменительного подхода, заостряющего внимание только на основах и флексиях.

Основные идеи подхода, основаного на эвристиках, описаны в работе (Goldsmith, 2001). Различные вариации этого метода приведены, например, в (Baroni et al., 2002), (Rehman and Hussain, 2005), (Creutz, 2003), см. также (Creutz and Lagus, 2007). Кроме того, было проведено два конкурса по автоматическому разделению слов на морфемы. В 2005 (Pascal challenge, 2005), вышеупомянутые методы были применены к разбиению слов на морфемы на различных языках (английский, финский язык, турецкий язык); испанский язык не рассматривался. В 2007 году (Pascal challenge, 2007), формулировка проблемы была изменена с целью обнаружения разных аспектов значения морфем, что, естественно, вызвало гораздо большие трудности.

Интересные идеи, связанные с улучшением автоматически построенных моделей, приводятся в (Schone and Jurafsky, 2000). Они основаны на использовании повторяющихся контекстов в корпусе. В нашей работе, поскольку мы работаем со словарем, эти идеи не применимы, однако их полезно учитывать в случае работы с корпусом текстов.

Одна из важных идей при автоматическом разбиении слов на морфемы связана с повторяемостью групп морфем. Часто результаты такой группировки называют сигнатурой (подписью), мы же предпочитаем сразу использовать более лингвистический термин - парадигма; в нашем случае это будет деривационная парадигма. Как это и принято, данный термин указывает на тот факт, что корни могут быть связаны с наборами аффиксальных морфем, и эти наборы повторяются, например, в английском, *high, highly, highness* и *bright, brightly, brightness* имеют общий набор суффиксов {∅, *-ly*, *-ness*}. Необходимо заметить, что использование таких наборов очень существенно сокращает пространство поиска алгоритма. Во всяком случае, это верно для испанского и английского языков. Словообразование русского языка гораздо более прихотливо (Кузнецова, Ефремова, 1986), (Тихонов, 2003) и, вероятно, для русского надо учитывать пересечение таких наборов. Скажем, было не так просто найти примеры полного совпадения парадигм, например, *белый* и *серый* с деривационной парадигмой *{-е(ть), -и(ть), -еньк, -оват, -от, -ёхоньк, -ость}*.

Рассмотрим более подробно подход, основаный на эвристиках (Goldsmith, 2001). Метод состоит из двух фаз: разбиение слов на морфемы без парадигм (signatures) и определение парадигм. В первой фазе используются условные вероятности морфем на основе взятого несколько «с потолка» распределения вероятностей (распределение Больцмана, которое давало результаты, более соответствовавшие интуиции автора) для введения метрики в пространстве поиска. Эта метрика позволяет применить итеративный, быстро сходящийся алгоритм оптимизации. Заметим, что в первой фазе этого метода не используется идея MDL (minimum description length), которая используется в его второй фазе для построения лучшего набора парадигм.

В нашем методе нас интересовало «как можно более *unsupervised*» обучение – мы хотели по возможности избежать подобной подгонки параметров под ответ. Плата за это была двоякой. Во-первых, чем более «*unsupervised*» будет алгоритм, тем менее красивые результаты он даст – однако тем более интересен тот факт, что он все-таки дает хоть какие-то результаты. Во-вторых, по сравнению с (Goldsmith, 2001) мы потеряли столь удобную меру на пространстве поиска, и теперь вынуждены для поиска экстремума честно перебирать все варианты (генетическим алгоритмом или любым другим подобным методом «почти полного перебора»). На этой же фазе мы оптимизируем один из вариантов MDL в качестве функции оценки, которая в указанной работе используется вовсе не для этой цели.

С другой стороны, генетические алгоритмы уже применялись для подобных задач (Kazakov, 1997), (Gelbukh *et al*., 2004). Заметим, что в данной статье мы видоизменяем идею, предложенную в этих работах, учитывая повторения парадигм. Это позволяет сократить пространство поиска, но не за счет введения другой метрики, то есть сама задача не видоизменяется.

Как и многие автоматические методы, рассмотренный алгоритм не претендует на нахождения стопроцентного решения и требует ручной постобработки, но он позволяет быстро обнаружить тенденции в данных и получить предварительные результаты без больших затрат ручного труда.

Далее в статье сначала мы описываем алгоритм и его параметры и представляем предварительные экспериментальные результаты.

2. Алгоритм

В этом разделе статьи мы представляем описание алгоритма и обсуждаем его параметры, используемые в экспериментах. Как во всяком генетическом алгоритме, самым важным является представление данных и реше-

ние о том, что должна вычислять функция оценки (fitness), определяющая, насколько хорош каждый индивид (от этого будет зависеть, останется ли он в следующих популяциях или нет).
Алгоритм содержит следующие шаги:

1. Для каждого слова, мы обнаруживаем все потенциально возможные приставки, начиная от первой буквы и заканчивая предпоследней, включая пустую приставку ∅. В список возможных морфем добавляются возможные приставки без повторений с их частотами.

2. Готовится список возможных корней, начинающихся от первой буквы и кончая последней. Список корней содержит уникальные элементы с их соответствующими частотами и не включает пустой корень ∅.

3. Строится список возможных суффиксов, который представляет собой наборы, организуемые от последней буквы слова до второй буквы от его начала. Этот список содержит пустой суффикс ∅ и также не разрешает повторений. Частоты суффиксов фиксируются.

4. Полученный список фильтруется следующим образом. Для корней обнаруживаются все возможные парадигмы, то есть, наборы морфем, которые повторены несколько раз для различных корней. Например, парадигмы {NULL, *-ism, -iz, -idad*}, {NULL, *-ant, -acion*}. Мы отфильтровываем все некорневые подмножества (потенциальные морфемы), которые содержат только один элемент. Кроме того, мы отфильтровываем морфемы, которые принадлежат только парадигмам с низкой частотой. В нашем случае, мы использовали в качестве порога частоту равную трем, то есть морфема должна входить по крайней мере в одну парадигму с частотой четыре и больше (то есть повторится по крайней мере у четырех слов).

5. Формируются хромосомы в соответствии со следующим правилам:
   a. Хромосомы являются бинарными (двоичными), в том смысле, что они состоят из генов, которые двоичны. А хромосома это набор генов.
   b. Каждый ген (двоичное место в хромосоме) соответствует элементу одного из списков,
   c. Длина хромосомы это сумма числа элементов трех списков.
   d. Значение гена «1» в хромосоме означает, что соответствующий элемент соответствующего списка является частью решения, в то время как значение «0» указывает на то, что этот элемент списка не участвует в словообразовании.

6. Порождается начальная популяция, состоящая из нескольких индивидов (хромосом), в генах которых случайным образом расставляются нули и единицы.

7. Применяется генетический алгоритм с различными значениями параметров (см. ниже). Для каждой вновь полученной хромосомы вычисляется функция оценки для решения о том, останется ли она в следующей популяции или нет. Заметим, что если мутация или скрещивания производят хромосому, которая «неправильна», то есть содержит морфемы, которые не существуют в отфильтрованных списках, тогда хромосома «улучшается» путем добавления случайным способом аффиксов, исправляющих эту «дефектную» хромосому. В худшем случае, такое слово добавляется в список корней с нулевыми аффиксами. Заметим, мутация такова, что нулевые аффиксы никогда не выключаются.

8. Функция оценки каждой хромосомы вычисляется следующим образом:

a. Общее число генов, используемых в решении (гены отличные от нуля), должно быть как можно меньше. Заметим, что мы уже обеспечили покрытие всего списка слов, когда «улучшали» хромосому в случае необходимости.

b. Частоты используемых элементов должны быть как можно больше, то есть высокочастотные элементы премируются. Под частотой здесь имеются в виду участие элемента в разбиении разных слов, то есть чем в большем числе слов он встречается, тем лучше.

В данной версии алгоритма мы делаем это по формуле:

$$fitness = -\log(freq(x0)) - \log(freq(prefix) * freq(stem) * freq(suffix))$$

где

freq(x0)) - сумма всех позиций в хромосоме, где стоят нули,

freq (prefix) - суммарная частота всех префиксов, участвующих в решении (их ген равен 1),

freq (stem) - суммарная частота всех корней, участвующих в решении (их ген равен 1),

freq (suffix) - суммарная частота всех суффиксов, участвующих в решении (их ген равен 1).

Поскольку минимизируется общий размер списка элементов, то можно рассматривать данную функцию как одну из разновидностей идеи minimum length description (MDL).

9. По окончании работы алгоритма из последней популяции выбирается хромосома с наибольшим значением функции оценки. Она и является наилушим, с точки зрения алгоритма, разбиением слов исходного словаря на морфемы.

Необходимо упомянуть следующие традиционные параметры генетического алгоритма:

1. Замещение и выбор родителей. В нашем случае, генетический оператор выбора родителей производится путем использования схемы турнира, а именно, для двух случайно выбранных индивидов сравнивается их пригодность (на основе функции оценки). Это гарантирует, что индивиды конкурируют и что выживают лучшие. Замещение определяет процент индивидов (хромосом), которые должны быть заменены в популяции.

2. Скрещивание. Генетический оператор скрещивания использует в качестве параметра число блоков, на основе которых хромосомы обмениваются своей генетической информацией для создания новых хромосом. Места для скрещивания выбираются случайным образом.

3. Наследование. Оперотор наследования определяет, какие индивиды остаются в популяции. Мы использовали схему наследования с элитизмом, при которой лучшие индивиды всегда сохраняются в популяции.

4. Мутация. Этот генетический оператор изменяет значения случайно выбранных генов с определенной вероятностью. Этот оператор важен, потому что он позволяет рассматривать новые возможности в пространстве решений.

5. Количество поколений. Этот параметр определяет, сколько раз должен применяться генетический алгоритм для популяции (для каждого индивида (хромосомы) в популяции).

6. Размер популяции. Этот параметр определяет, сколько индивидов одновременно существует в популяции. Обычно их должно быть не меньше 100.

Мы провели эксперименты с разными параметрами генетического алгоритма и сравнили их результаты. Выяснилось, что лучшие результаты получаются при использовании трех различных наборов параметров, то есть, алгоритм применяется последовательно три раза с разными параметрами.

Мы экспериментировали в пределах следующего диапазона параметров, см. Таблицу1.

	Минимальные параметры	Максимальные параметры
Размер популяции	50	5,000
Замещение	20%	100%
Мутация	Начинается с 20 %, уменьшается в соответствии с числом поколений	Начинается с 90 %, уменьшается в соответствии с числом поколений
Скрещивание (количество блоков, где происходит скрещивание)	1	20
Поколения	50	10,000

Таблица 1. Диапазоны параметров генетического алгоритма

Лучшие результаты были получены для следующих последовательно примененных наборов параметров генетического алгоритма для популяции в 200 индивидов; см. Таблицы 2, 3 и 4.

Замещение	60%
Мутация	30%, уменьшается в соответствии с числом поколений
Скрещивание	5
Поколения	10,000

Таблица 2. Параметры первого прохода

Замещение	80%
Мутация	80%, уменьшается в соответствии с числом поколений
Скрещивание	20
Поколения	7,000

Таблица 3. Параметры второго прохода

Замещение	80%
Мутация	80%, уменьшается в соответствии с числом поколений
Скрещивание	20
Поколения	7,000

Таблица 4. Параметры третьего прохода

Замещение 40% Мутация 20%, уменьшается в соответствии с числом поколений Скрещивание 4 Поколения 6,000

Цели каждого из проходов различны. При первом проходе популяция подготавливается и упорядочивается. При втором проходе происходит максимальная «встряска» популяции. Наконец, при третьем проходе популяция должна стабилизироваться, например, значительно уменьшена норма мутации.

3. Экспериментальные результаты

В качестве входных данных мы использовали испанский словарь, который содержал более 20,000 значимых слов. Мы игнорировали вспомогательные слова и наречия, работая только с существительными, глаголами и прилагательными. Так как мы интересуемся словообразовательной морфологией, то мы не рассматриваем окончания и заранее их отбрасываем, например, *, trabajar → trabaj- (работа(ть))*, *rojo → roj- (красн(ый))*, и т.д. Кроме того, мы не различали акцентуированные и неакцентуированные гласные, потому что акцент в испанском языке имеет чисто орфографическую функцию. Заключительный входной список содержал 16,849 уникальных слов без окончаний.

Для того, чтобы иметь возможность сравнивать наши данные с одной из самых известных систем - системой Linguistica (Goldsmith, 2001) - в настоящий момент мы провели эксперименты только для отделения суффиксов, хотя алгоритм позволяет производить одновременную обработку суффиксов и приставок.

Обработка наших данных системой (Gelbukh et al., 2004), которая достаточно успешно применяется к словоизменительной морфологии на корпусе, не дала положительных результатов по причине низкой частотности словообразовательных аффиксов. Полученная точность была менее 10%. Это показывает важность применения парадигм для данной задачи.

При подготовке списка начальных наборов для алгоритма были получены следующие результаты. Были выделены 7,747 деривационных парадигм, из которых 6,472 содержали более чем один элемент. Парадигмы, которые содержали ровно один элемент, были отфильтрованы. Стоит упомянуть, что из этих 6,472 парадигм 1,852 парадигмы содержали нулевой суффикс.

Дополнительно, мы использовали порог на повторяемость парадигмы, а именно, мы игнорировали парадигмы, которые повторились меньше чем три раза, то есть, они существуют для трех слов или меньше. Всего было 5,535 парадигм с частотой равной «1», 404 с частотой, равной «2» и 171 с частотой равной «3». В конце концов, только 372 парадигмы обрабатывались алгоритмом, то есть только суффиксы и корни, которые участвовали в них, были использованы для представления хромосом. В результате алгоритм работал с 17,085 корнями и 136 суффиксами. Это очень существенное уменьшение пространства поиска, если учесть, что вначале число потенциальных корней было более 44,000, а потенциальных суффиксов – более чем 15,000.

К сожалению, золотого стандарта разбиения на морфемы для испанского языка не существует (или мы его не нашли), и мы планируем его составить. К моменту написания этой статьи мы смогли провести оценку только на небольшой выборке из полученных результатов (можно считать его крохотным золотым стандартом, который в будущем будет расширен). А именно, мы сравнили наши результаты с результатами, полученными с применением системы Linguistica, для одних и тех же входных данных. Из нашего словаря было взято 5,000 первых слов. Это число связано с тем, что доступная версия системы Linguistica принимает на входе не больше этого количества слов. Обе системы обработали эти данные и получили результаты разбиения слов. В этих результатах мы взяли первые 100 слов из каждого из них и проверили вручную правильность разбиения. Система Linguistica получила 87% точности, а наша система 84%. Заметим, что это предварительные данные которые показывают, что наша система производит сопоставимое с существующими разбиение слов на морфемы, полная оценка может измениться в ту или другую сторону. Мы оценивали только точность, а не точность, полноту и F-меру, поскольку в наших экспериментах мы определяли только суффиксы, то есть точка разбиения была только одна на слово (а не несколько, как, напр., у (Creutz and Lagus, 2007), поэтому все три меры (P, R, и F) совпадают. В будущем мы будем определять как приставки, так и суффиксы, и тогда потребуется применять эти три меры. Конкретные разбиения слов могут не совпадать, например, система Linguistica не находит суффикс *–miento),* который был довольно частотен в списке и был обнаружен нашей системой. Точная оценка полученных результатов дело будущих исследований.

4. Выводы

Мы описали применение достаточно универсального оптимизационного метода, а именно, генетического алгоритма, к проблеме разбиения слов на морфемы на уровне словообразования. Наши эксперименты были проделаны на материале испанского языка.

Полученные результаты сопоставимы с результатами методов, основанных на вычислениях вероятностей с использованием эвристик для уменьшения размерности задачи. Нам представляется важным, что для решения нашей задачи мы не пользуемся дополнительными эвристиками, которые могут давать неплохие результаты, но при этом могут изменить саму формулировку проблемы, как, скажем, предположение о распределении Больцмана для условных вероятностей морфем в наиболее широко используемом подходе.

С другой стороны, очевидно, что решить проблему разбиения слов на морфемы методом полного перебора невозможно. Нам представляется важным представить возможность решения такой проблемы методом «почти полного перебора», с использованием одного из методов обучения без примеров (unsupervised).

Хромосома для алгоритма строится из ряда возможных наборов для корней и аффиксов, отфильтрованных специальным образом. Хромосомы являются двоичными, где «1» отмечает присутствие элементов в возможном решении и «0» - их отсутствие. Алгоритм учитывает деривационные парадигмы, т.е. повторяющиеся наборы аффиксов. Мы использовали в качестве фильтра частотность парадигмы (по крайней мере, четыре слова должны иметь такую парадигму). Также мы игнорировали парадигмы, которые состояли только из одного элемента.

Традиционные операторы генетического алгоритма были применены к обработке хромосом (=индивидов) в популяциях.

В настоящий момент, мы отфильтровываем все некорневые наборы (потенциальные морфемы), которые не являются частью какой-либо дервивационной парадигмы. В будущем, мы собираемся производить обработки этих парадигм иным способом, а именно, включить их в функцию оценки или включить их непосредственно в хромосомы.

Будущие исследования также связаны с выполнением точной оценки результатов разбиения слов на морфемы для испанского языка. Заметим, что для испанского языка не существует золотой стандарт в это области. Мы планируем разработать этот стандарт.

Было бы интересно просести аналогичные исследования для русского языка, где такой стандарт есть (Кузнецова, Ефремова, 1986). В случае русского языка, как мы уже упоминали, видимо, необходимо по-другому трактовать парадигмы, рассматривая их пересечения.

Список литературы

1. Baroni M, Matiasek J, Trost H. Unsupervised discovery of morphologically related words based on orthographic and semantic similarity. // ACL Workshop on Morphological and Phonological Learning. 2002.

2. Creutz M. Unsupervised Segmentation of Words Using Prior Distributions of Morph Length and Frequency. // Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03), Sapporo, Japan, July 2003, pp 280-287.

3. Creutz M., Lagus K. Unsupervised models for morpheme segmentation and morphology learning. // ACM Transactions on Speech and Language Processing (TSLP), January 2007, v.4 n.1, 34 p.

4. Gaussier E. Unsupervised learning of derivational morphology from inflectional lexicons. // Proceedings of the ACLWorkshop on Unsupervised Learning in Natural Language Processing. University of Maryland. 1999. pp 24–30.

5. Gelbukh A., Alexandrov M, SangYong Han. Detecting Inflection Patterns in Natural Language by Minimization of Morphological Model. // A. Sanfeliu, J. F. Martínez Trinidad, J. A. Carrasco Ochoa (Eds.). Lecture Notes in Computer Science N 3287, Springer-Verlag, 2004, pp. 432–438.

6. Goldsmith J. Unsupervised Learning of the Morphology of a Natural Language. // Computational Linguistics 27:2 (2001) pp. 153-198.

7. Haahr P., Baker S. Making search better in Catalonia, Estonia, and everywhere else // Google, 2007 http://googleblog.blogspot.com/2008/03/making-search-better-in-catalonia.htm

8. Kazakov D. Unsupervised learning of naıve morphology with genetic algorithms. // Workshop Notes of the ECML/MLnet Workshop on Empirical Learning of Natural Language Processing Tasks. Prague, Czech Republic, 1997, pp. 105–112.

9. Pascal Morphochallenge // 2005. http://www.cis.hut.fi/morphochallenge2005/

10. Pascal Morphochallenge // 2007. http://www.cis.hut.fi/morphochallenge2007/

11. Rehman Kh, Hussain I. Unsupervised Morphemes Segmentation. // Pascal Morphochallenge, 2005, 5 p.

12. Schone P., Jurafsky D. Knowledge-free induction of morphology using latent semantic analysis. // Proceedings of the Conference on Computational Natural Language Learning (CoNLL-2000), 2000.

13. Snover M.G., Brent M.R. A Bayesian model for morpheme and paradigm identification // Proceedings of the 39th Annual Meeting on Association for Computational Linguistics. Toulouse, France: ACL, 2001, pp. 490 – 498.

14. Snover M.G., Brent M.R.A Probabilistic Model for Learning Concatenative Morphology. // Proceeding of NIPS (Neural Information Processing Systems), 2002.

15. Кузнецова А.И, Ефремова Т.М. Словарь морфем русского языка. // М.: Рус.яз., 1986 -с.1136

16. Тихонов А.Н. Словообразовательный словарь русского языка: В 2 т. // М.: ООО «Издательство Астрель»; «Издательство АСТ», 2003. т1.- 860 ; т.2 -941.