УДК 81'32 + 811.512.162
DOI 10.25205/1818-7935-2018-16-2-34-47

М. А. Домрачев [1], **С. Н. Судоплатова** [2]

[1] *Компания «2ГИС»*
пл. Карла Маркса, 7, Новосибирск, 630048, Россия

[2] *Новосибирский государственный университет*
ул. Пирогова, 1, Новосибирск, 630090, Россия

m.domrachev.scientist@gmail.com, svetlana.sn21@gmail.com

ТЕСТИРОВАНИЕ МЕТОДОВ АВТОМАТИЧЕСКОГО ОБНАРУЖЕНИЯ ГРАНИЦ МОРФЕМ НА МАТЕРИАЛЕ АЗЕРБАЙДЖАНСКОГО ЯЗЫКА

Статья посвящена методам автоматического извлечения морфологических парадигм и нахождения границ морфем в словах. Подробно описан автоматный подход к обнаружению границ морфем. Этот подход был проверен экспериментально на материале азербайджанского языка, полученные результаты сравниваются с результатами библиотеки Linguistica на тех же данных.

Ключевые слова: извлечение морфологии, извлечение парадигм, обнаружение границ морфем.

Многие задачи современной компьютерной лингвистики так или иначе связаны с морфологией и морфемикой языка и требуют знаний о структуре слов, их морфемном составе и словоизменительных парадигмах. Решение задач оптимизации информационного поиска, проверки правописания, машинного перевода, а также многих других включает в себя работу с морфологическим уровнем каждого конкретного языка. Таким образом, существует потребность в базах данных, содержащих словоизменительные парадигмы. Подобные базы данных используются в таких областях, как автоматический морфологический анализ (POS-tagging, лемматизация), пополнение ресурсов о морфологии (например, Wiktionary), синтез поверхностных форм в машинном переводе (формы вида «лемма + грамматическая информация»).

В данной статье рассматриваются существующие методы автоматического извлечения парадигм и нахождения границ морфем в словах. Некоторые из этих методов были проверены экспериментально на материале азербайджанского языка.

Интерес к автоматическому обнаружению морфологических парадигм возник довольно давно. В 1967 г. вышли в свет две научные работы, затрагивающие эту тему. Одна из них – книга А. А. Зализняка «Русское именное словоизменение», в которой описывается «решение следующей практической задачи: произвести классификацию некоторой точно определенной (притом достаточно большой) совокупности русских именных парадигм и дать основанные на этой классификации правила синтеза именных парадигм, т. е. правила, позволяющие построить, в соответствии с существующей литературной нормой, парадигму любого русского имени (из взятой совокупности)» [2002. С. 3–4]. Еще один труд об автоматическом извлече-

Домрачев М. А., Судоплатова С. Н. Тестирование методов автоматического обнаружения границ морфем на материале азербайджанского языка // Вестн. Новосиб. гос. ун-та. Серия: Лингвистика и межкультурная коммуникация. 2018. Т. 16, № 2. С. 34–47.

ISSN 1818-7935
Вестник НГУ. Серия: Лингвистика и межкультурная коммуникация. 2018. Том 16, № 2

нии морфологии принадлежит Н. Д. Андрееву, известному своими работами в области математической лингвистики. В монографии «Статистико-комбинаторный метод в теоретическом и прикладном языковедении» [1967] Н. Д. Андреев представил алгоритм, который может успешно использоваться для извлечения морфологии языков различных семей и типологий.

В настоящее время в математической и компьютерной лингвистике активно используются технологии машинного обучения, средства теории формальных языков и теории автоматов. В ходе развития этой научной области создано большое количество подходов к решению задачи автоматического извлечения морфологических парадигм и обнаружения границ морфем. Кроме уже названного статистико-комбинаторного метода, существуют также:

1) подход на основе вычисления наибольшей общей подпоследовательности (НОП) слов [Ahlberg et al., 2014];

2) комбинированный подход – вычисление НОП и машинное обучение [Sorokin, Khomchenkova, 2016];

3) автоматный подход с использованием конечных преобразователей [Beesley, 2001];

4) подход на основе принципа минимальной длины описания [Goldsmith, 2001];

5) различные методы автоматического извлечения морфологических парадигм без учителя [Hammarström, Borin, 2011].

В нашем исследовании мы остановились на двух подходах: автоматном и на основе принципа минимальной длины описания. Первый подход мы рассмотрим подробнее в данной статье. Целью исследования была оценка качества определения границ морфем в словах азербайджанского языка при использовании этих подходов. Выбор языка обусловлен отсутствием широко известных аналогичных работ в данной сфере и повышением актуальности изучения этого языка в связи с бурным ростом экономики Республики Азербайджан.

Для экспериментов по извлечению морфологии был создан корпус азербайджанского языка (с письменностью на основе латиницы). Тексты корпуса собраны с различных новостных и информационных сайтов при помощи краулера Apache Nutch 1.3 [1]. Исходные ссылки взяты с сайта проекта An Crúbadán, где в свободном доступе выложена база статистических данных для редких языков, собранная из текстов с веб-сайтов [2]. Для обработки полученных текстов на языке программирования Python 3 был написан парсер, который нормализует текст (удаляет метаданные и пометы, расставляемые краулером, устраняет вкрапления иностранных языков, заменяет некоторые данные – даты, ссылки, телефоны, числа и другое – на спецсимволы, удаляет небуквенные символы, дефисы на месте тире, множественные пробелы и др.).

Таким образом, был получен корпус современного азербайджанского языка, содержащий 100 560 предложений (2 189 398 слов). Также из данного корпуса сформирован список уникальных словоформ (117 152 словоформы). Корпус можно найти в свободном доступе в репозитории на веб-сервисе Github [3].

Для оценки работы алгоритма использовались таблицы словоизменения азербайджанских существительных и глаголов, загруженные из Викисловаря [4]. Данные таблицы содержат следующие грамматические формы:

1) для существительных – формы изменения по падежам и числам, а для отдельных слов также притяжательные формы и их склонение;

2) для глаголов – формы пяти времён (давнопрошедшего, прошедшего категорического, настоящего, будущего категорического, будущего некатегорического), формы императива и других наклонений (желательного, долженствовательного, необходимого, условного) и формы изменения глагола по лицам.

Необходимо заметить, что таблицы глагольного словоизменения являются неполными, так как не включают в себя многих грамматических форм, например форм прошедшего повествовательного времени, настоящего длительного времени, отрицательных форм и др.

[1] Nutch Wiki [сайт]. URL: https://wiki.apache.org/nutch/.
[2] An Crúbadán [сайт]. URL: http://crubadan.org/.
[3] URL: https://github.com/svetlana21/Nutch_parser/.
[4] Vikilüğət [сайт]. URL: https://az.wiktionary.org/.

В результате мы получили таблицы словоизменения для 433 глаголов (28 578 словоформ) и 1 286 существительных (19 776 словоформ). Они были записаны в файлы формата json в виде словаря, где ключами являются лексемы в начальной форме, а значениями – списки словарей вида «форма: грамматическая информация».

Критерием оценки качества являлась правильность определения границы корня и суффикса. По этой причине файлы json с морфологической информацией были отредактированы: ключи в виде начальных форм лексем были заменены на наибольшую общую подпоследовательность всех форм парадигмы. Поскольку азербайджанский язык – язык агглютинативного строя, то в большинстве случаев корень в действительности совпадает с наибольшей общей подпоследовательностью форм парадигмы.

Кроме того, оценивалось количество словоизменительных парадигм, в которых все формы (из тех, что встретились в корпусе) были разобраны правильно.

Первый рассмотренный нами подход – автоматный. Для нахождения границ морфем использовался набор скриптов на языке Python 3 под общим названием FST_morphology [5], частично реализующий алгоритм нахождения морфемных границ внутри слова посредством анализа статистики вершин конечного автомата, построенного на словаре какого-либо языкового корпуса. Данный алгоритм является разработкой компании «ДГ-Софт».

Рассмотрим общую идею данного алгоритма нахождения морфемных границ. Словарь, сформированный на основе корпуса, компилируется в виде конечного автомата. После операций детерминации и минимизации автомат приводится к оптимальному виду (для компиляции словаря и операций над ним использовалась библиотека OpenFST [6]). Затем, анализируя количество предков и потомков каждой вершины, а также частоты, можно определить границы корня слов и морфем. Если количество предков вершины велико (наблюдается пик), но далее следует плато без резких скачков степеней вершины, то, вероятно, здесь и будет находиться граница корня и суффиксов. Еще один способ разделения слов на морфемы – разбиение по наиболее частотным вершинам в определенной позиции. Оба этих способа обнаружения границ морфем могут использоваться по отдельности или объединяться.

Рассмотрим подробнее все шаги алгоритма. Для примера возьмем тестовый корпус, в котором имеются только формы именительного и родительного падежей единственного и множественного числа имен существительных *almaz*, *bazar* и *bant*.

I. Создание списка уникальных словоформ из корпуса и файлов для компиляции автомата на его основе. Для компиляции необходимы 2 (в отдельных случаях 3) текстовых файла.

1. Файл в особом формате AT&T для описания автоматов, задающий переходы и содержащий строки вида <состояние-источник> <следующее состояние> <входной символ> <выходной символ> [<вес>] [7].

2. Внутреннее представление входных и выходных символов является целым числом. Поэтому необходимо явно указать отображение символов в целые числа с помощью файлов с таблицей символов, также в формате AT&T.

Таких файлов может быть два, если входные и выходные символы отличаются друг от друга. В нашем же случае моделируется не преобразователь, а обычный автомат, поэтому символы отображаются сами в себя, и нет необходимости в двух разных таблицах символов.

Все файлы были автоматически сформированы при помощи скриптов. В текстовом автомате каждому символу словоформ соответствует отдельное состояние автомата.

II. Необходимо скомпилировать автомат с помощью консольных команд OpenFST, затем детерминировать и минимизировать его.

После компиляции автомат будет выглядеть как на рис. 1. Затем производится детерминация автомата, т. е. приведение автомата к такому виду, что возможен только один переход по одному символу из одного состояния (тогда как в недетерминированном конечном автомате возможно несколько переходов). Автомат после детерминации представлен на рис. 2. Последний шаг на этом этапе – минимизация. Эта операция позволяет получить оптимальный автомат с минимально возможным числом состояний (рис. 3).

[5] URL: https://github.com/Ulitochka/FST_morphology/.

[6] URL: http://www.openfst.org/.

[7] FSM file formats [сайт]. URL: http://web.eecs.umich.edu/~radev/NLP-fall2015/resources/fsm_archive/fsm.5.html/.

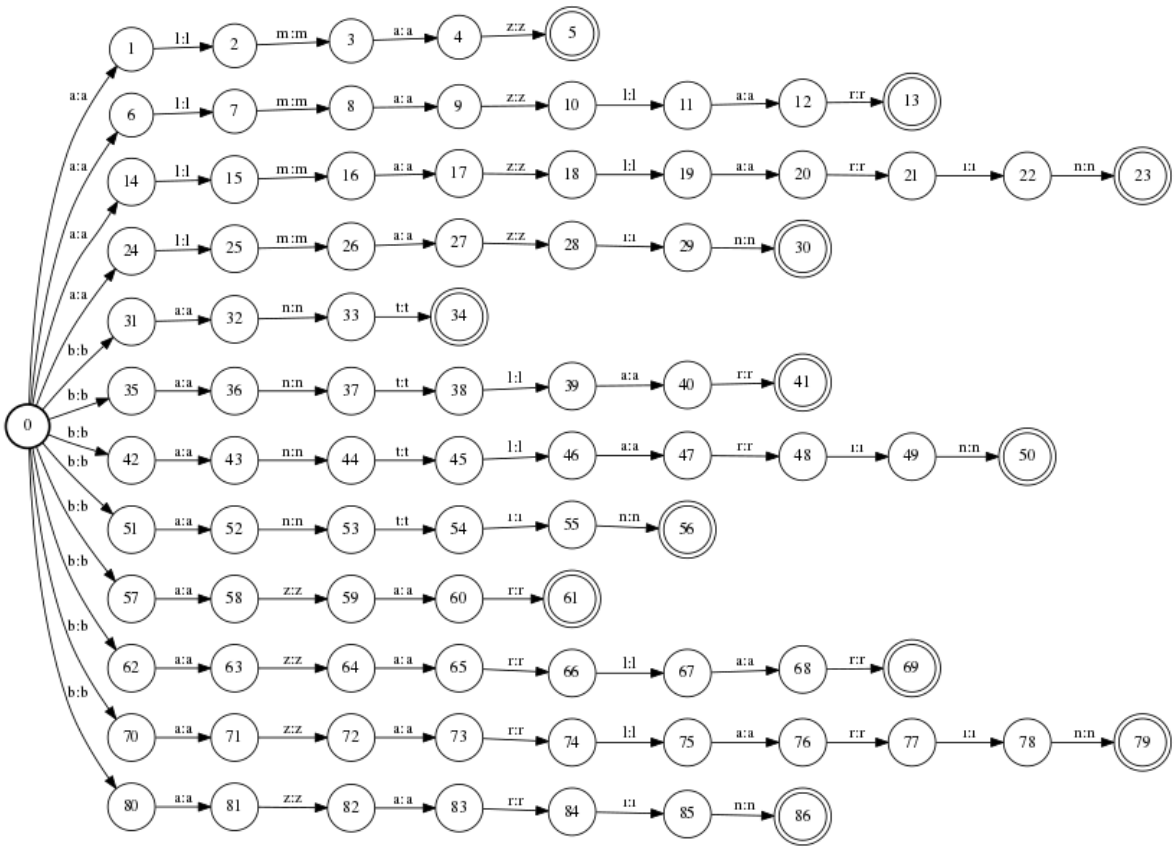

Рис. 1. Тестовый корпус в виде недетерминированного конечного автомата

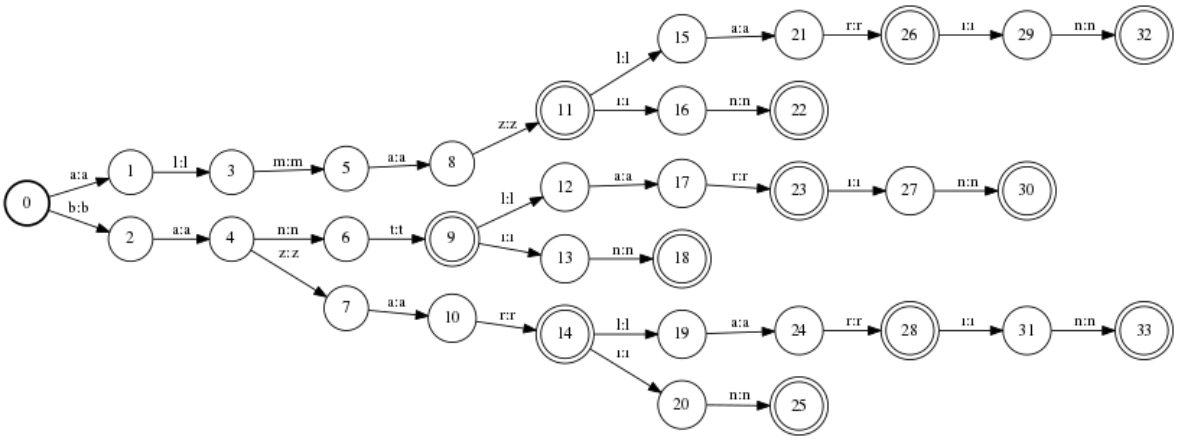

Рис. 2. Тестовый корпус в виде детерминированного конечного автомата

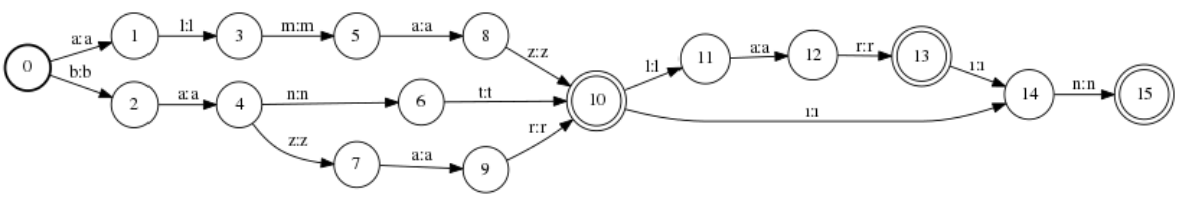

Рис. 3. Тестовый корпус в виде минимального конечного автомата

Получившийся минимальный автомат выводим в виде текстового файла в том же формате, из которого производилась компиляция исходного автомата. Дальнейшие операции будут производиться над этим текстовым автоматом с помощью скриптов.

III. Для разбиения словоформ на морфемы необходимо предварительно извлечь информацию из текстового автомата и создать файлы с данными. Первый шаг на этом этапе – получение из текстового автомата списка словоформ и переходов, т. е. всех состояний автомата, которые необходимо пройти, чтобы распознать слово. Для рассматриваемого примера эти данные будут выглядеть следующим образом:

bant 0-2-4-6-10
bazar 0-2-4-7-9-10
almaz 0-1-3-5-8-10
bantın 0-2-4-6-10-14-15
...

IV. Подсчет статистики степеней вершин автомата.

Одним из представлений конечных автоматов является диаграмма переходов – граф, вершины которого соответствуют состояниям автомата, а рёбра – переходам между состояниями. В дальнейшем будем рассматривать автомат в этом представлении и пользоваться некоторыми терминами теории графов.

Входящей степенью вершины *v* будем называть количество предков вершины *v*, исходящей степенью – количество потомков вершины *v*. Для каждой вершины находим ее входящую и исходящую степени. Далее анализируем степени, используя стратегию пика и плато [Hammarström, Borin, 2011. P. 325].

На этом этапе создаются 2 файла с данными.

1. Файл, содержащий информацию по каждой вершине: ее входящая степень, входящие переходы, номер самой вершины, исходящие переходы, исходящая степень.

2. Файл, содержащий информацию о состояниях (вершинах) с пиками. Если степень вершины превышает некоторый эвристически заданный порог, то вершина добавляется в список состояний с пиками (проверка производится и для входящей, и для исходящей степени). Если входящая и исходящая степени равны друг другу и меньше этого порога, то вершина добавляется в список состояний с плато.

В случае с примером, представленном на рис. 3, данный файл будет содержать список, содержащий кортеж [(3, '10', 2)] (при пороге, равном 2), так как для вершины 10 наблюдается пик: входящая степень этой вершины равна 3.

V. Сбор информации о позициях вершин в словах и длинах слов, сохранение списка с этой информацией в файл.

VI. Сбор статистики по вершинам: их степени и символы переходов.

На этом этапе создаются следующие файлы: статистика по степеням вершин с сортировкой по входящим степеням и по исходящим. В качестве примера приведем часть содержимого первого файла:

Indegree symbols aim arc Outdegree symbols
---
3 10 2
2 14 1
... ...

3. Данные по символам переходов:

Indegree symbols aim arc Outdegree symbols

... ...
['r', 't', 'z'] 10 ['l', 'ı']
['a'] 9 ['r']
... ...

VII. Нахождение самой частотной позиции вершины в слове. Для этого используется файл, созданный на шаге V, и статистика по степеням вершин с сортировкой по входящим степеням (шаг VI). Данные по каждой вершине вносятся в список.

VIII. Подсчет частот вершин, т. е. количества вхождений вершин в кратчайшие пути между двумя вершинами. Для этого загружается список слов с их переходами, созданный на шаге III, и из него формируется словарь частот переходов, где ключ – это номер вершины, а значение – количество вхождений вершины в переходы.

Затем в список, полученный на шаге VII, добавляется частота вершин.

На этом подготовка информации для нахождения границ морфем заканчивается.

IX. Нахождение границ морфем.

В самом начале необходимо задать пороги длин слов, которые будут рассматриваться. Для примера слова были ограничены длинами 4–8. Для слов каждой из допустимых длин производятся подготовка данных и разбиение слов на морфемы. Рассмотрим подробнее оба этапа.

Подготовка данных включает следующие шаги.

1) загрузка характеристик вершин (степени, средняя позиция, частота) из файла, подготовленного на шаге VIII, и слов с переходами и вершинами;

2) фильтрация слов (слова одной длины обрабатываются отдельно, так как одна и та же вершина может занимать разные позиции в словах разной длины, при этом являясь границей морфем для всех этих слов);

3) объединение данных в один список.

Для поиска границ морфем используются три способа: по частотам, по степеням вершин и объединение этих двух способов.

Для обнаружения границ морфем также необходимо задать пороги и константы:

1) порог самой частотной позиции в слове – задает минимальную длину корня, т. е. обозначает позицию в слове, с которой может начинаться суффикс;

2) порог частоты вершины;

3) количество морфем, которые следует выделить (т. е. количество наиболее частотных вершин в словах или вершин с пиками, которые рассматриваются как место возможной границы морфемы).

Обозначим эти константы как C_1, C_2, C_3.

Далее производится поиск границ морфем. Рассмотрим сначала подход с использованием частот вершин в определенной позиции.

Для каждого слова заданной длины производятся следующие действия:

1) фильтрация вершин: если самая частотная позиция вершины больше или равна порогу C_1 и ее частота больше или равна порогу C_2, то вершина и ее частота добавляются в отдельный список;

2) формирование списка кортежей «символ + переход»;

3) выбор C_3 самых частотных вершин;

4) на месте этих вершин ставится символ границы морфемы <>;

5) слово добавляется в список слов с расставленными границами морфем.

Рассмотрим эти шаги на примере формы almazın. Константы C_1, C_2, C_3 установим равными 3, 2 и 1 соответственно.

1. После фильтрации вершин были отобраны вершины, которые проходят пороги C_1 и C_2.

2. Составлен список кортежей «символ + переход».

3. Самой частотной является вершина 10 с частотой 12.

4. На месте нее ставится символ границы: almaz<>ın.

После обработки всех слов был сформирован следующий список (из десяти словоформ, так как формы almazların, bazarların и bantların были отфильтрованы по длине):

1) bant ban<>t
2) bazar baz<>ar
3) almaz alm<>az

4) bantın bant<>ın
5) bantlar bant<>lar
6) bazarın bazar<>ın
7) almazın almaz<>ın
8) bazarlar bazar<>lar
9) almazlar almaz<>lar

Здесь можно увидеть, что граница корня и суффикса в словоформах 4–9 обозначена верно, тогда как в формах 1–3, где границы не должно быть, есть ошибки. К сожалению, алгоритм не предусматривает возможности определения нулевых морфем, поэтому здесь граница оказалась на месте той вершины, которая удовлетворяла условиям.

Важно отметить еще одну проблему алгоритма: он стремится выделить ровно C_3 морфем, что может приводить к ошибкам. Рассмотрим пример работы алгоритма на том же тестовом корпусе, оставив слова длин 4–10 и установив $C_3 = 2$:

1) bant ban<>t
2) bazar baz<>a<>r
3) almaz alm<>a<>z
4) bantın bant<>ı<>n
5) bantlar bant<>l<>ar
6) bazarın bazar<>ı<>n
7) almazın almaz<>ı<>n
8) bazarlar bazar<>l<>ar
9) almazlar almaz<>l<>ar
10) bantların bant<>l<>arın
11) bazarların bazar<>l<>arın
12) almazların almaz<>l<>arın

Видно, что количество ошибок увеличилось. Словоформы almazların и bazarların действительно содержат две морфемы: показатель множественного числа -lar и показатель родительного падежа -ın. Но границы были расставлены иначе, что отражает еще один нюанс: если частоты всех отобранных вершин были одинаковыми (в данном случае частоты всех вершин, кроме вершины 10, были равны 6), то граница будет поставлена сразу после позиции C_1.

Рассмотрим подход с использованием степеней вершин и стратегии пиков и плато. Здесь будет использоваться файл, содержащий номера вершин с пиками, созданный на шаге IV. Для каждого слова заданной длины производятся следующие действия:

1) фильтрация вершин: если самая частотная позиция вершины больше или равна порогу C_1 и вершина содержится в списке вершин с пиками, то она добавляется в отдельный список;

2) формирование списка кортежей «номер вершины + степени»;

3) сортировка вершин от наибольшего перепада степеней к наименьшему и выбор топа-C_3 вершин;

4) на месте этих вершин ставится символ границы морфемы <>;

5) слово добавляется в список слов с расставленными границами морфем.

Рассмотрим эти шаги на примере формы almazın. Константы C_1, C_3 установим равными 3 и 2 соответственно (C_2 для этого подхода значения не имеет).

1. В файл, содержащий вершины с пиками, была записана только вершина 10. После фильтрации вершин она же осталась в списке: [('10', ['3', '2'])].

2. На месте вершины 10 ставится символ границы: almaz<>ın.

После обработки список словоформ выглядит так:

1) bant
2) bazar
3) almaz

4) bantın bant<>ın
5) bantlar bant<>lar
6) bazarın bazar<>ın
7) almazın almaz<>ın
8) bazarlar bazar<>lar
9) almazlar almaz<>lar
10) bantların bant<>ların
11) bazarların bazar<>ların
12) almazların almaz<>ların

Сразу можно отметить очевидное преимущество этого подхода. Алгоритм не стремится выделить ровно C_3 морфем, следовательно, ошибок становится меньше. Действительно, в данном случае границы корней и суффиксов были определены верно. Однако в словах с двумя суффиксами второй суффикс не был обнаружен.

Третий подход, представляющий собой объединение первых двух, дает те же результаты, что и второй подход.

Отметим, что приведенные выше примеры не отражают реального качества работы алгоритма. Для получения достойных результатов необходим большой корпус текстов, на котором можно собрать корректную статистику.

Также был проведен ряд экспериментов по нахождению границ морфем с помощью библиотеки Linguistica [8]. Linguistica – это программа, предназначенная для извлечения морфологии языка без учителя. Она находит вероятности морфемных комбинаций на текстовом корпусе, причем какие-либо дополнительные знания о языке, из которого это множество слов взято, не требуются. Программа определяет, где границы морфем находятся в словах, а затем решает, какие из них являются корнями, суффиксами и т. д. Алгоритм программы Linguistica основан на принципе минимальной длины описания и подробно представлен в статье Дж. А. Голдсмита «Unsupervised Learning of the Morphology of a Natural Language» [Goldsmith, 2001]. Программа работает под Windows, Mac OS X и Linux и написана на C++. Также существует обертка для Python, которая и была использована в нашей работе.

Далее рассмотрим результаты, полученные с помощью набора скриптов FST_morphology (с использованием трех подходов) и библиотеки Linguistica.

I. Результаты, полученные с помощью набора скриптов FST_morphology.

Рассмотрим 3 различных способа обнаружения границ морфем.

1. Способ с использованием частот вершин в определенных позициях.

В ходе экспериментов регулировались допустимые длины слов и параметры C_1, C_2, C_3 соответственно.

Всего было проведено 16 экспериментов. В табл. 1 приведены результаты (лучший результат выделен цветом) некоторых экспериментов, иллюстрирующие влияние изменения параметров на результат.

В столбце с результатами перечислено:

1) общее количество словоформ, которые подвергались оценке, – те словоформы из корпуса, для которых были найдены границы морфем и для которых имеются данные из Викисловаря;

2) количество форм, для которых граница корня и суффиксов была определена верно;

3) количество парадигм, в которых все формы были разобраны верно;

4) процент ошибочно разобранных слов.

Длины слов было решено практически не менять и не ограничивать, так как для азербайджанского языка характерны как короткие существительные (*ana*, *eq*), так и глагольные формы с большим числом морфем и, соответственно, букв. К тому же эксперимент со значительным изменением допустимых длин слов не дал ни сильного ухудшения результатов, ни заметного улучшения (ср. эксперименты 4 и 5).

[8] Linguistica 5 [сайт]. URL: https://linguistica-uchicago.github.io/lxa5/index.html/.

Таблица 1

Результаты, полученные с помощью набора скриптов FST_morphology
с разными наборами параметров
(подход с использованием частот вершин в определенных позициях)

№ эксперимента	Длина слова	C_1	C_2	C_3	Результат
4	3–12	3	100	2	All: 4 149 Correct forms: 730 Correct paradigms: 15 Errors: 82.40539889129911 %
5	3–22	3	100	2	All: 4 383 Correct forms: 734 Correct paradigms: 15 Errors: 83.25347935204198 %
6	3–22	3	150	2	All: 3 593 Correct forms: 556 Correct paradigms: 13 Errors: 84.52546618424715 %
9	3–22	3	100	3	All: 4 383 Correct forms: 796 Correct paradigms: 18 Errors: 81.83892311202374 %
10	3–22	3	80	3	All: 4 727 Correct forms: 872 Correct paradigms: 21 Errors: 81.55278189126295 %
11	3–22	3	70	3	All: 4 937 Correct forms: 892 Correct paradigms: 26 Errors: 81.93234757950172 %
13	3–22	3	80	1	All: 4 727 Correct forms: 660 Correct paradigms: 18 Errors: 86.03765601861646 %
14	3–22	3	80	2	All: 4 727 Correct forms: 797 Correct paradigms: 16 Errors: 83.13941188914745 %
15	3–22	4	80	3	All: 3 136 Correct forms: 204 Correct paradigms: 19 Errors: 93.49489795918367 %
16	3–22	2	80	3	All: 6 539 Correct forms: 343 Correct paradigms: 5 Errors: 94.75454962532497 %

Примечание: C_1 – минимальная длина корня; C_2 – порог частоты вершины; C_3 – количество морфем.

По всей видимости, результаты сильно коррелируют с выбранной минимальной длиной корня, причем для азербайджанского языка оптимальная длина равна 3. При выборе такой

длины корня результаты улучшались примерно на 10 % в сравнении с результатами с длинами 2 и 4 (эксперименты 10, 15, 16).

Большинство азербайджанских глагольных форм и именные формы множественного числа содержат две суффиксальные морфемы. Однако при выборе константы $C_3 = 2$ результат получается несколько хуже, чем при C_3 равной 3 или 1 (эксперименты 10, 13, 14).

Порог частоты вершины C_2, по-видимому, следует определять экспериментально. Для данного корпуса он составил 80.

Полученные результаты также проиллюстрировали существенные недостатки этого подхода, уже упоминавшиеся выше: обнаружение ровно C_3 морфем и отсутствие возможности определения нулевых морфем. Оба этих нюанса приводят к большому числу ошибок.

2. Метод пиков и плато.

В экспериментах с этим подходом константы C_1, C_3 выбраны в соответствии с наилучшим результатом предыдущей серии экспериментов и были равны 3 (C_2 в этом подходе не используется). Менялся порог степени вершины, по которому вершина могла быть отнесена к пикам. Результаты приведены в табл. 2 (лучший результат выделен цветом).

Таблица 2

Результаты, полученные с помощью набора скриптов FST_morphology с разными наборами параметров (метод пиков и плато)

№ эксперимента	Порог	Результат
1	3	All: 6 223 Correct forms: 1 690 Correct paradigms: 59 Errors: 72.84268037923832 %
2	4	All: 5 935 Correct forms: 1 724 Correct paradigms: 43 Errors: 70.9519797809604 %
3	5	All: 5 519 Correct forms: 1 562 Correct paradigms: 41 Errors: 71.69777133538685 %
4	6	All: 4 989 Correct forms: 1 379 Correct paradigms: 33 Errors: 72.35919021848065 %

Результаты показывают, что метод пиков и плато справляется с задачей обнаружения границ морфем значительно лучше, чем метод с использованием частот. Действительно, по сравнению с предыдущим подходом у метода пиков и плато есть очевидное преимущество: алгоритм выделяет не ровно C_3 морфем, а не больше чем C_3, и допускает наличие слов, в которых не выделяются аффиксы.

3. Объединенный подход.

Параметры, подобранные в первых двух сериях экспериментов, не менялись. При этом результат оказался хуже, чем при использовании метода пиков и плато:

All: 4 502
Correct forms: 1 002
Correct paradigms: 23
Errors: 77.74322523322968 %

II. Результаты, полученные с помощью библиотеки Linguistica.

Библиотека позволяет регулировать параметры, перечисленные в табл. 3, со значениями по умолчанию. В ходе экспериментов регулировались такие параметры, как *max_word_types*, *min_stem_length*, *max_affix_length*, *min_sig_count*. В табл. 4 приведены некоторые результаты (лучший результат выделен цветом), полученные с различными наборами параметров, включая значения по умолчанию из табл. 3 (default).

В первых экспериментах параметры менялись по одному с целью оценить степень влияния каждого из них на результат. На результат никак не влияет параметр *max_word_types*, что можно заметить по экспериментам 1 и 2. Поэтому во всех остальных экспериментах этот параметр имел значение по умолчанию. При изменении длины корня результат также практически не меняется (эксперименты 1, 6, 7), однако оптимальной минимальной длиной также остается 3. Большинство азербайджанских аффиксов содержат 2–3 буквы, поэтому максимальные длины аффиксов регулировались в этих пределах. Лучший результат был достигнут при выборе максимальной длины аффикса, равной 2 (эксперименты 1, 8, 9). Что касается параметра *min_sig_count* (количество корней, необходимых для признания сигнатуры валидной), то он также не слишком влияет на результат. Лучшие показатели, как ни странно, были достигнуты при наименьшем количестве корней (см. эксперименты 1, 10–13). Наилучший результат был достигнут при объединении всех подобранных параметров (эксперимент 14).

При сравнении результатов обеих систем можно отметить, что процент ошибочно разобранных форм ниже у FST_morphology, тогда как у Linguistica больше количество верных парадигм. В целом результаты оказались довольно низкими у обеих систем. Это можно объяснить несколькими причинами.

1. Недостатки в системе оценки:

а) оценивается только граница корня и суффиксальных морфем;

б) на стыке корня и суффиксов встречаются некоторые чередования; они не учитываются.

Последнюю особенность следует прокомментировать отдельно. В случае более высокоуровневых задач, таких, например, как автоматическое объединение слов в парадигмы, определение границы морфем является предварительным шагом. Даже если на этом шаге граница

Таблица 3

Параметры класса Linguistica в библиотеке Linguistica
и их значения по умолчанию

Параметр		Значение по умолчанию
max_word_tokens	максимальное количество токенов (неуникальных слов) для обработки	0 (= все)
max_word_types	максимальное количество тайпов (уникальных слов) для обработки	1 000
min_stem_length	минимальная длина корня	4
max_affix_length	максимальная длина аффикса	4
min_sig_count	минимальное количество корней для валидной сигнатуры	5
min_context_count	минимальное количество появлений для валидного контекста	3
n_neighbors	число «соседей» слова	9
n_eigenvectors	число собственных векторов	11
suffixing	есть ли в обрабатываемом языке тенденция к суффиксации	1 (= да)
keep_case	учитывать ли различия в регистре («the» vs «The»)	0 (= нет)

Примечание: в терминологии библиотеки Linguistica сигнатура – это отсортированный по алфавиту список суффиксов, которые встречаются с определенным корнем в корпусе. Сигнатуры содержат только ссылки на корни и суффиксы, причем один корень соединен только с одной сигнатурой.

Таблица 4

Результаты, полученные с помощью библиотеки Linguistica
с разными наборами параметров

№ эксперимента	Параметр				Результат
	max_word_types	min_stem_length	max_affix_length	min_sig_count	
1	default	default	default	default	All: 3 694 Correct forms: 205 Correct paradigms: 99 Errors: 94.45046020573903 %
2	10 000	default	default	default	All: 3 694 Correct forms: 205 Correct paradigms: 99 Errors: 94.45046020573903 %
6	default	3	default	default	All: 3 682 Correct forms: 204 Correct paradigms: 99 Errors: 94.45953286257469 %
7	default	5	default	default	All: 3 563 Correct forms: 175 Correct paradigms: 83 Errors: 95.08840864440079 %
8	default	default	3	default	All: 4 933 Correct forms: 325 Correct paradigms: 107 Errors: 93.41171700790593 %
9	default	default	2	default	All: 6 477 Correct forms: 751 Correct paradigms: 96 Errors: 88.4051258298595 %
10	default	default	default	6	All: 3518 Correct forms: 193 Correct paradigms: 95 Errors: 94.51392836839113%
11	default	default	default	4	All: 3 889 Correct forms: 217 Correct paradigms: 102 Errors: 94.42015942401646 %
12	default	default	default	3	All: 4 235 Correct forms: 222 Correct paradigms: 105 Errors: 94.75796930342385 %
13	default	default	default	2	All: 4 905 Correct forms: 290 Correct paradigms: 120 Errors: 94.08766564729868 %
14	default	3	2	2	All: 7 494 Correct forms: 1 038 Correct paradigms: 126 Errors: 86.14891913530825 %

морфемы проведена не там, где ее принято проводить в соответствии со сложившейся в языкознании традицией, это не помешает правильному объединению словоформ в парадигму. Мы же решаем задачу, пытаясь понять, насколько хорошо алгоритм сможет определить традиционную границу морфем и какие проблемы могут возникнуть в решении этой задачи.

2. Азербайджанский язык имеет довольно богатую морфологию, поэтому, для того чтобы учесть все морфемы и собрать достаточную статистику по ним, необходим большой корпус текстов. По всей видимости, размер корпуса, на котором проводились эксперименты, не является достаточным, поскольку обе системы показывали достойные результаты на других языках (например, на русском).

3. Рассматриваемые системы извлечения морфологии несовершенны. Некоторые проблемы и недочеты были описаны выше.

Что касается дальнейших перспектив, то прежде всего нужно собрать корпус текстов большего объема. Кроме того, следовало бы внести в набор скриптов FST_morphology ряд доработок:

1) добавить возможность определения нечленимых слов или слов с нулевыми морфемами;

2) создать эвристики, позволяющие работать со словами с разным количеством морфем.

Необходимо доработать и систему оценки:

1) оценивать правильность нахождения не только границы корня и суффиксов, но и границы между суффиксальными морфемами;

2) учитывать чередования на стыке корня и суффиксов;

3) собрать больше таблиц словоизменения.

Скрипты для обработки корпуса, сбора таблиц словоизменения и оценки, написанные на языке программирования Python 3, а также готовые материалы (корпус, таблицы словоизменения и др.) собраны в репозитории на веб-сервисе github.com [9].

Список литературы

Андреев Н. Д. Статистико-комбинаторные методы в теоретическом и прикладном языковедении. Л.: Наука, 1967. 403 с.

Зализняк А. А. Русское именное словоизменение с приложением избранных работ по современному русскому языку и общему языкознанию. М.: Языки славянской культуры, 2002. 752 с.

Ahlberg M., Forsberg M., Hulden M. Semi-supervised learning of morphological paradigms and lexicons // Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics. Gothenburg, Sweden, 2014. P. 569–578.

Beesley K. R. Finite-State Morphological Analysis and Generation of Arabic at Xerox: Status and Plans in 2001 // EALC 2001. Workshop Proceedings on Arabic Language Processing: Status and Prospects. Toulouse, France, 2001. P. 1–8.

Goldsmith J. A. Unsupervised Learning of the Morphology of a Natural Language // Computational Linguistics. Massachusetts, USA: MIT Press, 2001. Vol. 27 (2). P. 153–198.

Hammarström H., Borin L. Unsupervised learning of morphology // Computational Linguistics. 2011. Vol. 37 (2). P. 309–350.

Sorokin A. A., Khomchenkova I. A. Automatic Detection of Morphological Paradigms Using Corpora Information // Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference «Dialogue 2016». M., 2016.

Материал поступил в редколлегию 29.01.2018

[9] URL: https://github.com/svetlana21/az_morphology/.

Mikhail A. Domrachev [1], **Svetlana N. Sudoplatova** [2]

[1] *Company «2GIS»*
7 Karl Marx Square, *Novosibirsk*, *630048*, *Russian Federation*

[2] *Novosibirsk State University*
2 Pirogov Str., *Novosibirsk*, *630090*, *Russian Federation*

m.domrachev.scientist@gmail.com, *svetlana.sn21@gmail.com*

TESTING METHODS FOR AUTOMATIC DETECTION OF MORPHEME BOUNDARIES IN THE AZERBAIJANI LANGUAGE

The paper is devoted to unsupervised learning and computer analysis of a natural language morphology, including automatic paradigm extraction and detection of morpheme boundaries. It describes in detail a finite-state approach implemented here. This approach has been tested on the material of the Azerbaijani language; the article discusses its results and compares them with those available in the Linguistica library on the same language.

Keywords: unsupervised learning of morphology, paradigm extraction, detection of morpheme boundaries.

References

Ahlberg M., Forsberg M., Hulden M. Semi-supervised learning of morphological paradigms and lexicons. *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*. Gothenburg, Sweden, 2014, p. 569–578.

Andreev N. D. Statistiko-kombinatornye metody v teoreticheskom i prikladnom yazykovedenii. [Statistical methods in theoretical and applied linguistics]. Leningrad, Nauka, 1967, 403 p. (in Russ.)

Beesley K. R. Finite-State Morphological Analysis and Generation of Arabic at Xerox: Status and Plans in 2001. *EALC 2001 Workshop Proceedings on Arabic Language Processing: Status and Prospects*. Toulouse, France, 2001, p. 1–8.

Goldsmith J. A. Unsupervised Learning of the Morphology of a Natural Language. *Computational Linguistics.* Massachusetts, USA, MIT Press, 2001, vol. 27 (2), p. 153–198.

Hammarström H., Borin L. Unsupervised learning of morphology. *Computational Linguistics*, 2011, vol. 37 (2), p. 309–350.

Sorokin A. A., Khomchenkova I. A. Automatic Detection of Morphological Paradigms Using Corpora Information. *Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference "Dialogue 2016"*. Moscow, 2016.

Zaliznyak A. A. Russkoe imennoe slovoizmenenie [The Russian nominal inflection]. Moscow, Nauka, 1967, 370 p. (in Russ.)

For citation:

Domrachev Mikhail A., Sudoplatova Svetlana N. Testing Methods for Automatic Detection of Morpheme Boundaries in the Azerbaijani Language. *Vestnik NSU. Series: Linguistics and Intercultural Communication*, 2018, vol. 16, no. 2, p. 34–47. (in Russ.)

DOI 10.25205/1818-7935-2018-16-2-34-47