# Извлечение лексических единиц с релевантными характеристиками, как основная задача обработки естественного языка

Р. Г. Мифтахова

*Башкирский государственный университет*
*Россия, Республика Башкортостан, 450076 г. Уфа, улица Заки Валиди, 32.*

*Email: miftahovar@yandex.ru*

Корректное выявление и кодирование релевантных характеристик лежит в основе создания текстовых классификаторов. Способность классификатора идентифицировать специфические свойства лингвистических данных в конечном счете позволяет создать определенную форму взаимодействия человека и компьютера.

**Ключевые слова:** обработка естественного языка, текстовый классификатор, NLTK, парсер, извлечение релевантных характеристик.

Обработка естественного языка представляет собой общее направление искусственного интеллекта и математической лингвистики и является важной составляющей в научной, экономической, социальной и культурной жизни. Теоретические и методологические приемы НЛП переживают быстрый рост. Она изучает проблемы компьютерного анализа и синтеза естественных языков. Под анализом имеется в виду понимание языка, а синтез – это генерация грамотного текста. Решение этих проблем позволит создать определенную форму взаимодействия человека и компьютера.

Основной задачей НЛП (Обработка естественного языка) является извлечение лексических единиц с определенными характеристиками. В данном аспекте необходимо идентифицировать специфические свойства лингвистических данных, которые могут служить основой для его классификации. Например, суффикс -л является показателем прошедшего времени глаголов в русском языке, а частое использование глаголов в будущем времени может свидетельствовать о том, что текст новостной. Такие видимые характеристики, как форма слова или частота его встречаемости в тексте, нередко связаны с отдельными аспектами значения, например, времени или тематики. Возникает вопрос, как выявить, какой аспект формы связан с тем или иным аспектом значения; каким образом распознаются специфические свойства языковых данных, которые являются ключевыми для их классификации; как построить такую модель языка, которая бы позволила обрабатывать язык автоматически.

Под классификацией представляется задача присвоения корректной метки класса введенному тексту. В базовой классификации каждый введенный текст рассматривается изолированно от других, а набор меток определен заранее. К такой классифика-

ции относятся, например, задачи определения спама в электронных сообщениях; определение тематики новостной статьи, когда список меток заранее предопределен, например, «спорт», «технологии» или «политика»; определение полярности текста – позитивный текст или негативный; определение жанра текста; а также определение конкретного значения слова в данном контексте, если оно многозначно. Однако базовая классификация имеет ряд особенностей. Так, при мультиклассовой классификации одному и тому же отрывку текста может присваиваться несколько меток; а в классификации открытого типа набор меток не предопределяется заранее.

Рассмотрим гендерную классификацию собственных имен. Даже без подробного анализа можно заметить, что имена русского языка, оканчивающиеся на -а и -я относятся скорее всего к женским, тогда как имена, оканчивающиеся на согласные чаще относятся к мужским. В английском языке большинство мужских имен имеют окончания -k, -o, r, -s, -t, а те, что оканчиваются на -a, -e, -i скорее относятся к женским.

Как построить классификатор, чтобы более четко смоделировать эти различия? Одним из ресурсов для проведения такого рода лингвистических исследований является библиотека с открытым исходным кодом NLTK. Она включает в себя программное обеспечение, базы данных, программное обеспечение, а также документацию.

Итак, на начальном этапе важно определить, какая из характеристик является решающей и как ее закодировать. Допустим, с помощью такой команды можно построить словарь, содержащий релевантную информацию об именах:

```
>>> def gender_features(word):
... return {'last_letter': word[-1]}
>>> gender_features('Nick')
{'last_letter': 'k'}
```

Далее необходима база данных, которая бы содержала список примеров с соответствующими метками. Его можно получить, воспользовавшись ресурсом NLTK:

```
>>> from nltk.corpus import names
>>> labeled_names = ([(name, 'male') for name in names.words('male.txt')] +
... [(name, 'female') for name in names.words('female.txt')])
>>> import random
>>> random.shuffle(labeled_names)
```

Разделив эти данные на обучающую и тестовую части, с помощью наивного байесовского алгоритма получаем желаемый классификатор, способный отличить женские имена от мужских.

```
>>> classifier.classify(gender_features('Neo'))
```

**'male'**

**>>> classifier.classify(gender_features('Trinity'))**

**'female'**

Качество таких классификаторов напрямую зависит от выбора релевантных характеристик и типа их кодирования. Несмотря на то, что часто можно получить достойную производительность, используя довольно простой и очевидный набор функций, в основном значительный результат достигается только через использование тщательно построенных функций, основанных на глубоком понимании поставленной задачи. Задача выделения определенных лингвистических характеристик решается путем проб и ошибок. Деление корпуса на развивающую и тестовую части позволяет генерировать список ошибок, которые допускаются классификатором. Тщательное изучение каждого случая, где модель присвоила ошибочную метку, позволяет определить, какие дополнительные релевантные свойства языка помогут системе принять правильное решение. Например, после анализа ошибок классификатора имен, можно заметить, что к релевантным характеристикам относятся не только последние буквы имен, но и суффиксы. Имена, окачивающиеся на -yn чаще относятся к женским, хотя окончание -n присуще мужским именам. Поэтому на практике классификатору требуется несколько релевантных характеристик. Работа осуществляется на основе системы правил извлечения информации, каждое из которых задает шаблон синтаксической структуры и шаблон формируемого фрагмента формализованного представления информации. При обработке документа просматриваются результаты синтаксического анализа и ведется поиск фрагментов, соответствующих шаблонам из правил извлечения информации. Далее часть слов, подпадающая под указанные правила извлекается из текста и преобразуется в формализованную структуру.

Ключевое отличие такой процедуры от задачи «понимания» текста состоит в том, что происходит работа с информацией из указанной предметной области, для которой четко заданы концептуальная модель данных и правила извлечения. Поскольку описанные принципы извлечения информации применимы для различных предметных областей, то возможно создание универсального программного обеспечения – парсера.

В таких системах анализ текста может проходить в две этапа: поверхностное сканирование и выявление ключевых фрагментов текста, а затем детальный анализ по принципиально другому сценарию, согласно которому вышестоящие модули обращаются к нижестоящим за необходимой уточняющей информацией. Например, в задаче извлечения информации при поверхностном сканировании определяются упоминания людей, а затем семантический модуль обращается к нижележащему синтаксическому с запросом «какие связи есть у каждого найденного упоминания»


В отличие от парсеров для английского языка, для работы с русским языком открытых парсеров, которые можно обучить, всего несколько. Среди них MST Parser, основанный на задаче нахождения минимального остовного дерева, и MaltParser. Они основаны на машинном обучении, но работают по-разному. Обучение MST-парсера очень трудоемкое и результаты невыгодно отличаются от парсера MaltParser.

Для русского языка подобные технологии также разрабатываются компанией Яндекс. Так, для извлечения структурированных данных из текста на естественном языке создан Томита-парсер, где вычленение фактов происходит при помощи контекстно-свободных грамматик и словарей ключевых слов. Парсер позволяет писать свои грамматики и добавлять словари для нужного языка. А исходный код проекта открыт и выложен на GitHub. Томита-парсер позволяет по написанным пользователем контекстно-свободным грамматикам выделять из текста разбитые на поля цепочки слов или факты. Например, можно написать шаблоны для выделения адресов. Здесь фактом является адрес, а его полями – «название города», «название улицы», «номер дома» и т.д. Парсер включает в себя три стандартных лингвистических процессора: токенизатор – разбиение на отдельные лексические единицы, сегментатор – разбиение на предложения и морфологический анализатор – mystem. Основные компоненты парсера: газеттир, набор КС-грамматик и множество описаний типов фактов, которые порождаются этими грамматиками в результате процедуры интерпретации. Томита-парсер позволяет извлекать из текста объекты, такие как даты, адреса, телефоны, ФИО, название товара, действие, тональность; а также связи между объектами: события, мнения и отзывы, контактные данные, объявления. Таким образом, пользователь может получить структурированную информацию о датах рождения личностей, месте рождения, учебных заведениях, в которых они учились и так далее. Томита-парсер представляет собой одну из важнейших отечественных разработок в области обработки естественного языка, несмотря на то, что требует множество идей и доработок для увеличения возможностей целевого анализа текста.

Технологии, основанные на обработке естественного языка, широко применяются в каждодневной практике. Так, смартфоны поддерживают предиктивный ввод текста и распознавание рукописного текста; поисковики позволяют получить доступ к информации, скрытой в неструктурированном тексте; машинный перевод дает возможность извлекать тексты, скажем, на китайском, а читать на английском; а с помощью текстового анализатора можно определить полярность блогов и твитов. Через обеспечение более естественного интерфейса «человек-машина» и более усложненного доступа к базам данных, обработка естественного языка сегодня играет центральную роль в мультиязычном информационном пространстве.

**Литература**

1. Маннинг К., Рагхаван П., Шютце Х. Введение в информационный поиск. М.: Вильямс, 2011. – 528 с.
2. Рассел С., Норвиг П. Искусственный интеллект. Современный подход. М.: Вильямс, 2007. – 1480с.
3. Толдова С. Ю. Оценка методов автоматического анализа текста 2011–2012: синтаксические парсеры русского языка // Диалог-2012: тезисы конференции. Москва, 2012.
4. https://tech.yandex.ru/tomita/
5. https://www.osp.ru/os/2013/04/13035562

# The extraction of lexical units with the relevant features as the main task of natural language processing

R. G. Miftakhova

*Bashkir State University*
*32 Zaki Validi Street, 450074 Ufa, Republic of Bashkortostan, Russia.*

*Email: miftahovar@yandex.ru*

The correct identification and coding of relevant characteristics is the basis for the creation of text classifiers. The ability of a classifier to identify the specific properties of the linguistic data ultimately allows you to create some form of interaction between man and computer.

**Keywords:** natural language processing, text classifier, NLTK, parcer, feature extraction.