UDC 330:519.712.1
DOI: 10.34671/SCH.HBR.2019.0303.0026

МЕТОДЫ ОБРАБОТКИ ТЕКСТА ПРИ СОЗДАНИИ ЧАТ-БОТОВ

© 2019
AuthorID: 238830
SPIN: 5167-9840
ResearcherID: G-5698-2015
ORCID: 0000-0002-2872-1008
ScopusID: 56658374000

Бородин Александр Иванович, доктор экономических наук, профессор кафедры «Финансовый менеджмент»

AuthorID: 640058
SPIN: 2816-0073
ResearcherID: F-6867-2018
ORCID: 0000-0001-8021-5738
ScopusID: 55151132000

Вейнберг Роман Рафаилович, кандидат экономических наук, доцент кафедры «Информатика»

AuthorID: 815116
SPIN: 5785-1598
ScopusID: 57000415400

Литвишко Олег Валерьевич, кандидат экономических наук, доцент кафедры Финансового менеджмента

Российский экономический университет имени Г.В. Плеханова
(117997, Россия, Москва, Стремянный переулок, 36, e-mail: Litvishko.OV@rea.ru)

Аннотация. В рамках разработки чат-бота необходимым и достаточным условием работы с текстом является применение различных методик анализа текста, как входного элемента общения с ботом и его обучения. В статье был рассмотрен ряд решений, применяемых для анализа текста и построения моделей анализа текстовых данных: методы лемматизации, векторизации текста, различные модели машинного обучения. Основной упор в статье сделан на рассмотрение методики обработки текста в разном формате и с помощью разных технологий, что обеспечивает масштабируемость и универсальность предлагаемой технологии и эффективность работы будущего чат-бота в целом. Статья будет интересна для программистов, текстовых аналитиков и всех интересующихся работой с текстом и разработкой систем работы с текстовой информацией.

Ключевые слова: анализ текста, чат-боты, разработка, интеллектуальные ассистенты, обучение, векторный анализ слов.

METHODS OF TEXT PROCESSING WHEN CREATING CHATBOTS

© 2019

Borodin Alexander Ivanovich, Doctor of Sciences in Economics, Professor, Department of Financial Management
Veynberg Roman Rafailovich, PhD in Economic Sciences**,** associate professor of the chair of Informatics
Litvishko Oleg Valerevich, PhD in Economic Sciences**,** associate professor of the chair of Financial management

Plekhanov Russian University of Economics
(117997, Russia, Moscow, Stremyanny lane, 36, e-mail: Litvishko.OV@rea.ru)

Abstract. As part of the development of a chatbot, a necessary and sufficient condition for working with text is the use of various methods of text analysis as an input element of communication with the bot and its training. The article deals with a number of solutions used for text analysis and construction of text data analysis models: lemmatization methods, text vectorization, various machine learning models. The main focus of the article is on the methods of text processing in different formats and using different technologies, which provides scalability and versatility of the proposed technology and the effectiveness of the future chatbot as a whole. The article will be interesting for programmers, text analysts and anyone interested in working with text and developing systems for working with text information.

Keywords: text analysis, chat-bots, to develop intelligent assistants, training, vector analysis of words.

ВВЕДЕНИЕ

Внедрение технологии интеллектуальных ассистентов (чат-ботов) уже сейчас позволяет бизнесу экономить существенные средства на поддержке и консультациях пользователей. С развитием искусственного интеллекта коммуникативные возможности чат-ботов начинают приближаться к уровню человека. Однако при наличии большого количества решений на рынке, как правило, подавляющее большинство таких систем работают на достаточно простых принципах, не приспособленных к ответам на сложные вопросы, с учетом всех обстоятельств, озвученных в вопросе.

Для создания бота, способного генерировать уникальные ответы, до сих пор приходится прибегать к самостоятельному созданию моделей машинного обучения, интеграции их в программный код и т.д. Разработку такой программы, можно условно разделить на несколько частей: сбор текстовых данных; формирование размеченного массива данных по принципу «вопрос-ответ»; морфологический анализ слов, содержащихся в выборке (приведение слов к «нормальной» форме); анализ существующих или проектирование качественно новых статистических формул машинного обучения для построения целостной модели; проведение тестирования модели, на выборке данных, не применявшейся при обучении, интеграция разработанной модели в программный код приложения.

МЕТОДОЛОГИЯ

Целью статьи является обзор методики анализа текстовой информации типа «вопрос-ответ» как части семантической базы чат-бота. Для достижения цели в рамках исследования важно решить такие задачи как: анализ существующих алгоритмов и инструментов создания чат-ботов, рассмотрение программной реализации алгоритмов автоматической генерации ответов и поиска наиболее релевантных статей, а также извлечения ответа на поставленный в статье вопрос. Решение поставленных в работе задач осуществлялось на основе применения об-

щенаучных методов исследования. Основополагающим для предпринимаемого исследования является описательный метод, включающий прием наблюдения, интерпретации, сопоставления, обобщения. В исследовании также применены методы синхронного и диахронного анализа языковых и научных данных.

РЕЗУЛЬТАТЫ

Для того, чтобы наиболее эффективно извлечь признаки из слов, по которым текст будет кластеризироваться или классифицироваться, необходимо произвести первичный морфологический анализ. Предварительная обработка текста включает в себя:

1. Токенизацию - разбиение длинных участков текста на более мелкие (абзацы, предложения, слова).
2. Нормализацию - приведение текста к единому регистр слов, отсутствию знаков пунктуации, словесному написание чисел и т.д., что необходимо для применения унифицированных методов обработки текста.
3. Стеммизацию - приведение слова к его корню путем устранения придатков (суффикса, приставки, окончания).
4. Лемматизацию - приведение слова к смысловой канонической форме слова (инфинитив для глагола, именительный падеж единственного числа—для существительных и прилагательных).
5. Чистку - удаление стоп-слов, которые не несут смысловой нагрузки (артикли, междометья, союзы, предлоги и т.д.).

Классификация текста – один из ключевых аспектов работы чат-бота, основной целью которого является нахождение наиболее релевантных ответов на заданные пользователем вопросы. Для того что бы успешно построить модель классификации необходимо собрать большую обучающую выборку, содержащую вопросы и ответы на них. Однако, имея одни необработанные текстовые данные, невозможно консистентно обучить модель машинного обучения, поэтому перед формулированием гипотезы алгоритма аппроксимации, способного качественно обнаруживать зависимости в текстовых данных, необходимо произвести ряд предобработок, в частности: токенизацию, морфологический анализ (стемминг; удаление шумовых слов), векторное представление слов, извлечение признаков из векторного представления слов, обучение модели с помощью machine learning.

Ряд действий (рисунок 1), предваряющих обучение модели machine learning, призван привести тексты к виду, в котором анализ слов становится возможным, поскольку текст в чистом виде не может быть использован в математической модели.

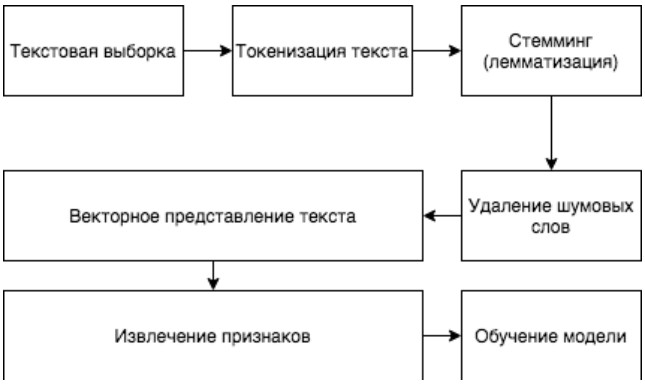

Рисунок 1 - Процесс обучения модели (составлено автором)

Естественно, такой подход можно экстраполировать и на использование модели анализа текста, важно чтобы все описанные этапы выполнялись на тестовой выборке также, как и на тренировочной, в противном случае мы получим некорректное поведение модели, не способной принимать верные решения относительно принадлежности текста к тому или иному классу. Рассмотрим подробнее эффективные инструменты классификации текста, подходящие для каждого этапа классификации:

1. Токенизация текста. Инструменты доступны во встроенных библиотеках в каждом популярном языке программирования, так в языке Python данный процесс может быть реализован следующим образом:

```
import string
import re

def get_tokenized_words(text):

    text_without_punctuation= re.sub('[{}]'.format(string.punctuation),'',  text)

    words = text_without_punctuation.split(" ")
    tokenized_words =  [v.lower() for v in words]

    return tokenized_words
```

В данном примере используются две стандартные библиотеки языка Python: re (модуль, предоставляющий возможность работать с функционалом регулярных выражений), string (модуль, содержащий в себе полезные массивы и функции для анализа Python-строк).

2. Лемматизация слов как правило не предусмотрена в базовом наборе библиотек какого-либо языка программирования, однако существует значительное количество свободно распространяемых библиотек с открытым кодом для целого ряда языков программирования, в частности:

- NLTK для Python:

```
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()

# выведет rocks: rock

print("rocks :", lemmatizer.lemmatize("rocks"))
```

- SpaCy:

```
import spacy

# Инициализация модели на английском языке

nlp = spacy.load('en', disable= ['parser', 'ner'])

sentence = "The striped bats are hanging on their feet for best"

doc = nlp(sentence)
print(" ".join( [token.lemma_ for token in doc])
```

- PyMorphy2 (поддерживает русский язык):

```
import pymorphy2

morph = pymorphy2.MorphAnalyzer()

p = morph.parse('стали')[0]

print(p.normal_form)
```

В данной статье приведены наиболее популярные средства для лемматизации, в действительности список всех доступных средств значительно шире. Соответствующие вычисления могут производится как на локальном компьютере пользователя, так и в облаке, с подключением по API.

3. Удаление шумовых слов (стоп-слов). Данное действие не требует использования функций специальных библиотек, в данном случае достаточно лишь удалить из имеющегося массива слов все слова, идентифицированные как шумовые.

4. Векторное представление слов и текстов. Данный этап – один из важнейших в предобработке текста. Рассмотрим основные подходы и инструменты на примере доступных в языке Python. One-hot Encoding – самый простой способ кодировки слов, при применении данного метода мы исходим из того, что кодировка слова будет производится из заранее определенного и конечного набора возможных значений. Алгоритм подсчитывает количество слов в словаре, например, 1500, и устанавливаете для них порядок от 0 до этого N-1 (в данном случае 1499). Затем вектор i-ого слова определяется как вектор длины N, состоящий из нулевых значений и лишь i-ый элемент вектора равняется единице.

Для того что бы наиболее наглядно проиллюстриро-

вать данный подход, представим ситуацию, при которой наш словарь состоит исключительно из трёх слов: «Red», «Green», «Blue». В результате работы алгоритма векторизации One-Hot Encoding мы получим следующие три вектора (таблица 1).

Таблица 1 - Векторное представление слов*

Слово	0-ой элемент	1-ый элемент	2-ой элемент
Red	1	0	0
Green	0	1	1
Blue	0	0	1

** составлено автором*

Таким образом, предложение, содержащее слова «Red» и «Green» будет представлено в виде вектора [1,1,0]. Такой подход имеет ряд отрицательных аспектов, в частности:

- полученное векторное пространство не дает представление о взаимосвязях между словами;
- работает только при условии, что размер словаря строго ограничен, новые слова будут игнорироваться.

Реализованные модули данной кодировки доступны в ряде библиотек машинного обучения, в частности такая функция доступна в библиотеке Sci-Kit Learn:

```
from sklearn.preprocessing import OneHotEncoder lb =
preprocessing.LabelBinarizer()
lb.fit_transform(['Red','Green','Blie'])
```

```
array([[0, 0, 1],
[0, 1, 0],
[1, 0, 0]])
```

One-Hot Encoding - хорошая отправная точка, но данный подход слишком прост и не позволяет отслеживать важные особенности взаимосвязи слов внутри текстов. Одна проблема с простым подсчетом состоит в том, что некоторые слова, такие как «the», будут появляться много раз, и не будет иметь большого значения в закодированных векторах. Альтернативой является вычисление частот слов и, безусловно, самым популярны методом является TF-IDF. В основе метода лежат два основных показателя:

- TF или частота слова - это отношение количества вхождения конкретного термина к суммарному набору слов в исследуемом тексте (документе). Этот показать отражает важность (весомость) слова в рамках определенной статьи/публикации.
- IDF или обратная (инвертированная) частота документа - это инверсия частотности, с которой определенное слово фигурирует в коллекции текстов (документов). Благодаря данному показателю можно снизить весомость наиболее широко используемых слов (предлогов, союзов, общих терминов и понятий). Для каждого термина в рамках определенной базы текстов предусматривается лишь одно единственное значение IDF.

Данный показатель широко используется в поисковых алгоритмах, поскольку позволяет получать релевантный результат при минимально затраченном машинном ресурсе.

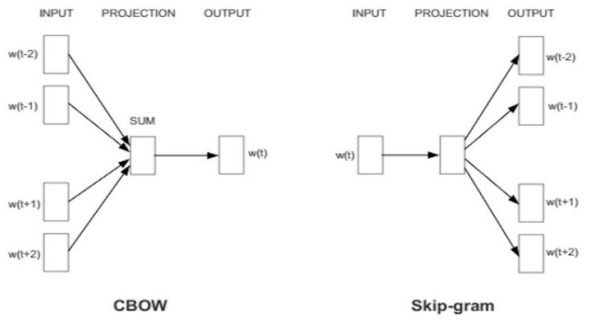

Рисунок 2 - Различия между CBOW и skip-gram (составлено автором)

Инструмент word2vec, созданный группой исследователей Google, реализует модель векторного представления слов и позволяет создавать такого рода распределенные представления.

Алгоритм обучает представления слов, опираясь на модель непрерывного мешка слов (Continious Bag-of-Words, CBOW) или skip-gram, в результате чего слова в пространстве признаков оказываются рядом со схожими словами, в зависимости от их контекста. В целом алгоритмы CBOW и skip-gram не отличаются по наиболее существенным признакам (рис. 2).

Continuous Bag Of Words или CBOW (рис 3) представляет собой простую нейронную сеть с одним скрытым слоем.

Ее цель – угадать вектор слова, получив на вход слова, находящиеся с двух сторон от него (количество слов - настраиваемая величина).

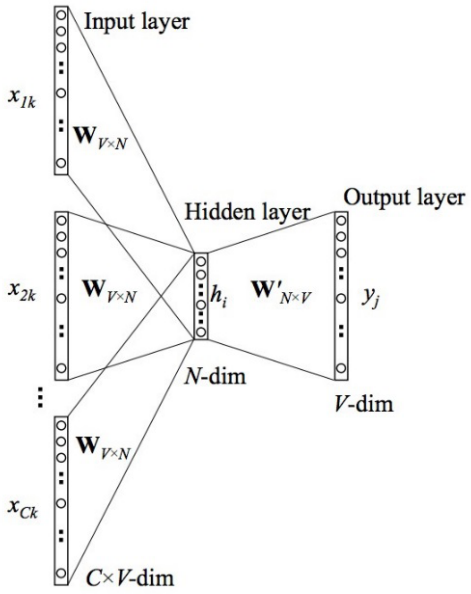

Рисунок 3 - Архитектура CBOW (составлено автором)

На рисунке 4 изображен общий случай CBOW, где $X_{1,2\ldots c}$ – это множество векторных отображений слов, окружающих целевое слово; C – количество окружающих слов; Y – вектор целевого слова; V – длинна векторов; W – вектор весов. В общем случае скрытый слой вычисляется следующим образом:

$$h = \frac{1}{C} W^T (x_1 + x_2 + \cdots + x_C)$$

Для выявления связей между словами необходимо подключить метрику, которая бы отмечала схожесть векторных представлений, для этого используется:

$$p(c|w_I) = \exp(v_c \times v_w) / \sum_{n=1}^{C} \exp(v_n \times v_w)$$

Максимизация данной функции является задачей алгоритма, т.к. максимизировав ее, будут получены контекстуально связанное векторное пространство. На ее основании вычисляется функция потерь, которая в дальнейшем используется для обновления значений векторов слов.

Такой подход позволяет выявлять существенные закономерности в контексте.

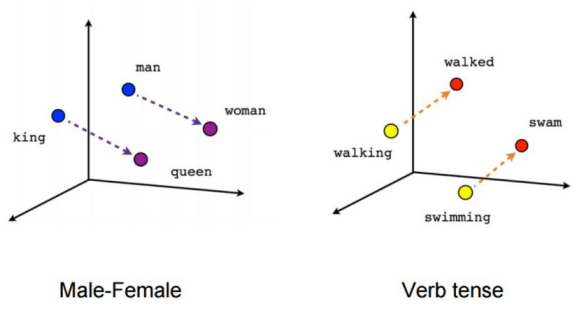

Рисунок 4 - Векторное представление слов (составлено автором)

Среди рассмотренных методов векторизации слов, наибольшей точностью и эффективностью обладает метод распределенного представления слов (табл. 2).

Таблица 2 - Метод распределенного представления слов

Метод векторизации	Функция	Принадлежность
One-hot encoding	Возвращает [0] или [1]	Гауссовых моделей
TF–IDF	Матрица частоты слова	Деревья, нейронные сети
Распределенное представление слов	Вектор слова с учетом контекста	Деревья, нейронные сети

ВЫВОДЫ

В рамках разработки чат-бота необходимым и достаточным условием работы с текстом является применение различных методик анализа текста, как входного элемента общения с ботом и его обучения. В статье был рассмотрен ряд решений, применяемых для анализа текста и построения моделей анализа текстовых данных: методы лемматизации, векторизации текста, различные модели машинного обучения. Основной упор в статье сделан на рассмотрение методики обработки текста в разном формате и с помощью разных технологий, что обеспечивает масштабируемость и универсальность предлагаемой технологии и эффективность работы будущего чат-бота в целом.

В будущем планируется на основе данной методики проанализировать массив тестовой информации и внедрить ее в механизм работы имеющегося в распоряжении авторов чат-бота.

СПИСОК ЛИТЕРАТУРЫ:

1. Ferry Wahyu Wibowo Chatbot Using a Knowledge in Database: Human-to- Machine Conversation Modeling // 7th International Conference on Intelligent Systems, Modelling and Simulation · ISMS 2016, 2016.

2. Richard Csaky Deep Learning Based Chatbot Models // Budapest University of Technology and Economics, 2017.

3. Analytics and Monitoring Tools // AWS URL: https://developer.amazon.com/alexa/agencies-and-tools/tools-analytics (дата обращения: 26.05.2019).

4. Mikhail Burtsev, Alexander Seliverstov,Rafael Airapetyan DeepPavlov: Open-Source Library for Dialogue Systems // ACL 2018, At Melbourne, 2018.

5. Nitin Indurkhya Text Mining: Predictive Methods for Analyzing Unstructured Information. Springer, 2008.

6. Lucie Skorkovská Application of Lemmatization and Summarization Methods in Topic Identification Module for Large Scale Language Modeling Data Filtering // Mapping a Lexical Semantic Resource to a Common Framework of Computational Lexicons, 2012.

7. W. John Wilbur The automatic identification of stop words // Journal of Information Science, 1992.

Статья публикуется при поддержке гранта РФФИ в рамках научного проекта № 18-310-20008