УДК 004.91
DOI 10.34822/1999-7604-2021-4-12-15

ПРИМЕНЕНИЕ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ ДЛЯ АВТОМАТИЗАЦИИ ПРОЦЕССА АНАЛИЗА СОДЕРЖАНИЯ ТЕКСТА

А. И. Валиев, С. А. Лысенкова ✉
Сургутский государственный университет, Сургут, Россия
✉ *E-mail: lsa1108@mail.ru*

Представлены результаты автоматизированной оценки резюме кандидата на соответствие требованиям вакансии с помощью алгоритма, основанного на методах естественной обработки языка, позволяющего трансформировать текстовую информацию в числовые признаки.

Ключевые слова: стемминг, лемматизация, предобработка текста, TF-IDF, косинусное сходство.

APPLICATION OF MACHINE LEARNING METHODS FOR AUTOMATION OF THE PROCESS OF THE TEXT CONTENTS ANALYSIS

A. I. Valiev, S. A. Lysenkova ✉
Surgut State University, Surgut, Russia
✉ *E-mail: lsa1108@mail.ru*

The results of automatic verification of applicant's curriculum vitae for compliance with a job vacancy using the algorithm based on natural language processing techniques are presented. The algorithm makes it possible to transform text information into the numerical features.

Keywords: stemming, lemmatization, text preprocessing, TF-IDF, cosine similarity.

Введение

Рекрутер – сотрудник по подбору персонала – сталкивается со многими рутинными процедурами: описание вакансии, оценка резюме на соответствие вакансии, первичное тестирование и проверка тестового задания, отправка ответа по результатам тестирования, формирование рекомендаций по выбору вакансии в зависимости от навыков кандидата.

Согласно онлайн-мониторингу компании HeadHunter [1] на одну позицию в списке вакансий поступает в среднем от 4 до 12 резюме, часть из которых отклоняется после просмотра и первоначального тестирования. Эта часть работы занимает до 50 %, или около 4 ч, рабочего времени специалиста по подбору персонала ежедневно, поэтому оптимизация процесса поиска и предварительного отбора резюме необходима для перераспределения времени рекрутера на работу с релевантными кандидатами, а также для сокращения временных и финансовых издержек компании при поиске новых сотрудников.

Алгоритм автоматизации состоит из следующих основных этапов: предварительной обработки текста, оценки его информативности и вычисления коэффициента сходства текстов.

Предварительная обработка текста

Основная сложность при работе с текстом связана с количеством слов, часть из которых не относится к полезной информации либо имеет равное значение; их исключение значительно сокращает время анализа. Например, к ним относятся стоп-слова, которые являются вспомогательными (предлоги, союзы, частицы и др.); разные грамматические формы слов. Эта проблема решается методами морфологического анализа: лемматизацией (приведением слова к его первоначальной форме – табл. 1) и стеммингом (нахождением основы слова – табл. 2) [2, 3].

Таблица 1

Пример использования лемматизации

Исходный текст	Лемматизация
У нас амбициозная цель – стать самым инновационным Банком России, и мы быстро движемся в этом направлении	у мы амбициозный цель стать самый инновационный банк россия и мы быстро двигаться в это направление

Примечание: составлено авторами по [2].

Таблица 2

Пример использования стемминга

Исходный текст	Стемминг
У нас амбициозная цель – стать самым инновационным Банком России, и мы быстро движемся в этом направлении	у нас амбициозн цел стат сам инновацион банк росс и мы быстр движ в эт направлен

Примечание: составлено авторами по [2].

Оценка информативности

Для оценки информативности существуют различные статистические меры, самая распространенная из которых – TF-IDF (term frequency-inverse document frequency).

TF измеряет, насколько часто слово встречается в тексте [4]. Так как частота появления слов в длинных текстах может быть значительно выше, чем в малых, применяют относительное значение – количество нужных слов (терминов), деленное на общее количество слов в тексте. Слово, которое встречается в документе чаще, чем в других, важно для этого документа:

$$TF(\text{a}) = \frac{N_a}{N}, \quad (1)$$

где N_a – количество раз, когда термин «а» встретился в тексте;

N – общее число слов в данном документе.

IDF – инверсия частоты, с которой определенное слово встречается [6]. Учет IDF уменьшает вес широкоупотребительных слов. Для каждого уникального слова в пределах рассматриваемого набора текстов существует только одно значение IDF. Значение инверсии частоты слова считается как логарифм частного от общего количества текстов и количества текстов, в которых встречается слово:

$$IDF(\text{a}) = \log\left(\frac{D_{all}}{D_a}\right), \quad (2)$$

где D_{all} – число документов в наборе текстов (коллекции);

D_a – число документов в коллекции, когда слово «а» встретилось в тексте.

Найденные значения TF и IDF перемножаются для определения «веса» слова в текущем тексте (документе):

$$TF - IDF = TF * IDF. \quad (3)$$

Слова, которые не важны для всех документов коллекции, получат низкий вес TF-IDF, а важные – высокий.

Мера TF-IDF используется для представления документов коллекции в виде числовых векторов, которые отражают важность использования каждого слова из некоторого набора слов (количество слов набора определяет размерность вектора) в каждом документе. Получаемая векторная модель позволяет сравнивать тексты, сравнивая представляющие их векторы в какой-либо метрике [5, 6].

Коэффициент сходства

Одна из метрик для сравнения текстов – косинусное сходство. Это мера сходства между двумя векторами, которая использует для измерения косинус угла [6]. В случае информационного поиска косинусное сходство двух текстов изменяется в диапазоне от 0 до 1 [5]:

$$\cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}, \quad (4)$$

где A – векторное представление текста A;

B – векторное представление текста B.

Если значение равно 0, то векторы текстов являются ортогональными и не сходятся между собой. Векторы текстов полностью совпадают, если значение равно 1. Преимущество косинусного сходства – его низкая сложность, так как данный метод исследует только ненулевые значения [7, 8].

Результаты

Для реализации описанных методов использовался язык программирования Python (библиотека docx – для чтения файлов с расширением docx; библиотека scikit-learn – для решения задач классического машинного обучения; pymorphy2 – инструмент для морфологического анализа русского языка; NaturalLanguageToolkit – пакет библиотек и программ для символьной и статистической обработки естественного языка (stopwords, word_tokenize, RussianStemmer, EnglishStemmer)).

На примере текстов вакансий с сайта HeadHunter проведена предварительная обработка по удалению стоп-слов и спецсимволов, в первом случае – по алгоритму стемминга, во втором – лемматизации. С помощью TF-IDF произведен подсчет для обоих случаев и получены числовые векторы (табл. 3, 4).

Таблица 3

Веса при использовании стемминга

Терм	Вес
работ	1,011696
оп	1,066090
дмс	1,130215
требован	1,191667
разработк	1,227514
...	...

Примечание: составлено авторами.

Таблица 4

Веса при использовании лемматизации

Терм	Вес
работы	1,035507
опыт	1,066090
по	1,117057
дмс	1,130215
на	1,163888
...	...

Примечание: составлено авторами.

Для дальнейшего исследования проведена предварительная обработка резюме Python-разработчика (с сайта HeadHunter), содержащего информацию о возрасте, месте рождения, желаемой должности и зарплате, опыте работы, образовании и ключевых навыках *(Python, SQL, RabbitMQ, Redis, DjangoFramework, Flask, Node.js, Celery, DesignPatterns, Docker, DevOps, CI/CD, ML)*, удалены стоп-слова и спецсимволы, произведен стемминг и лемматизация, рассчитан TF-IDF для обоих случаев и получены числовые векторы.

С помощью косинусного сходства проведено сравнение вектора резюме с векторами вакансий и получен список вакансий и значения схожести. В табл. 5, 6 представлены пять самых «схожих» вакансий.

Таблица 5

Список вакансий при использовании стемминга

Название вакансии	Схожесть, %
DevOpsEngineer	13
Backend разработчик	13,8
Ведущий DevOps инженер	15,8
Разработчик моделей машинного обучение (ML Engineer)	16,4
Backend программист (Python)	21,9

Примечание: составлено авторами.

Таблица 6

Список вакансий при использовании лемматизации

Название вакансии	Схожесть, %
Разработчик (Support)	10
DevOpsEngineer	11,3
Ведущий DevOps инженер	12,6
Разработчик моделей машинного обучение (ML Engineer)	14
Backend программист (Python)	16,9

Примечание: составлено авторами.

Заключение

Описанный алгоритм и этапы его использования позволяют оценивать резюме по соответствию заданной вакансии, обеспечивают рациональный автоматизированный поиск, упрощают процесс отбора резюме и могут лечь в основу информационно-аналитической системы поиска и подбора персонала для оптимизации использования времени сотрудников, затрачиваемого на поиск и подбор персонала.

Литература

1. Hh-индекс : статистика по России. URL: https://stats.hh.ru/ (дата обращения: 20.11.2021).

2. Машинное обучение с использованием библиотек Python. URL: https://slemeshevsky.github.io/python-course/ml/html/._ml-flatly003.html (дата обращения: 20.03.21).

3. Вершинин Е. В., Тимченко Д. К. Исследование применения стемминга и лемматизации при разработке систем адаптивного перевода текста // Наука. Исследования. Практика : сб. изб. ст. по материалам Междунар. науч. конф. СПб., 2020. С. 77–79.

4. TF-IDF. URL: https://coolshell.cn/articles/8422.html (дата обращения: 20.11.2021).

5. Белова К. М., Судаков В. А. Исследование эффективности методов оценки релевантности текстов // Препринты ИПМ им. М. В. Келдыша. 2020. № 68. 16 с. DOI: http://doi.org/10.20948/prepr-2020-68.

6. Векторная модель текста. URL: https://habr.com/ru/sandbox/18635/ (дата обращения: 20.11.2021).

7. Сторожук Н. О., Коломойцева И. А. Анализ методов определения текстовой близости документов // Информатика, управляющие системы, математическое и компьютерное моделирование : материалы студ. секции IX Междунар. науч.-технич. конф. (ИУСМКМ-2018). Донецк : ДонНТУ, 2018. С. 43–47.

8. Власенко А. В., Тарасов Е. С., Корх И. А., Мухтаров И. И. Разработка архитектуры блоков фильтрации и нормализации в системе классификации текстовой информации // Научные труды КубГТУ. 2021. № 1. С. 55–65.