Анализ подходов к извлечению ключевых навыков из вакансий

П. В. Корытов[1], Я. Ю. Грибецкий[2]

Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)

[1]pvkorytov@etu.ru, [2]yanfire-is@yandex.ru

Е. А. Андреева[3], И. И. Холод[4]

Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)

[3]eaandreeva@etu.ru, [4]iiholod@etu.ru

Аннотация. **В рамках работы рассматривается задача извлечения ключевых навыков из вакансий. Исходными данными являются тексты вакансий с портала «Работа России» с разметкой по категориям ключевых навыков. Сравниваются методы на основе извлечения ключевых слов (YAKE, PositionRank и WikiNEuRaL, KBIR-Inspec) и модель для извлечения именованных сущностей, обученная на исходных данных.**

Ключевые слова: извлечение ключевых слов; извлечение именованных сущностей; анализ вакансий

I. Введение

При обучении в ВУЗе у студента формируются определенные компетенции, связанные с теми или иными дисциплинами. Они должны покрывать навыки, требуемые в вакансиях, чтобы выпускники имели возможность продолжать свою профессиональную деятельность сразу после окончания обучения или во время него. В соответствии с этим имеется необходимость автоматически выделять из текстовых описаний вакансий ключевые навыки. В дальнейшем это может помочь в решении следующих задач:

- поиск кластеров похожих вакансий;
- формирование рекомендаций студентам подходящих вакансий;
- валидация реализуемых образовательных программ относительно вакансий, существующих на рынке труда;
- и т. п.

Таким образом, целью работы является разработка решения для выделения ключевых навыков из текстовых описаний вакансий. Для этого в работе производится сравнение различных подходов к решению задачи на наборе данных из аннотированных текстов вакансий (см. п. «описание набора данных»).

II. Описание методов решения задачи

Поставленная задача решается с помощью двух подходов – извлечения ключевых слов и распознавание именованных сущностей (NER). Все методы в качестве выхода имеют множество подстрок из исходного текста; метод NER дополнительно классифицирует подстроки.

A. Случайный метод

В качестве baseline используется «случайный метод» – из текста берется N=15 случайных подстрок длиной от 1 до 4 токенов.

B. Эвристические методы извлечения ключевых слов

Данный метод является ансамблем нескольких эвристических методов извлечения ключевых слов - YAKE [1], PositionRank [2] и SingleRank [3]. YAKE основан на нескольких эвристиках, таких как положение слова в предложении, его капитализация, положение предложения в тексте и т. п. PositionRank и SingleRank используют графовые алгоритмы, где одно слово является вершиной графа, и связи формируются исходя из положения слов в предложениях.

В выводе всех алгоритмов для данного текста сохраняются только ключевые фразы, содержащие хотя бы одно существительное или латинское слово. Затем схожие фразы объединяется с помощью pipeline: стемминг, TF-IDF, DBSCAN. Таким образом, исключаются дублирующие ключевые слова и ключевые слова с пересечениями.

Также возможна вариация данного алгоритма с использованием «окна» на первом шаге, движущегося по предложениям.

В отличие от методов, рассматриваемых далее, этот подход менее требователен к ресурсам и не требует предварительного обучения модели.

C. Нейросетевые методы извлечения ключевых слов

Рассматриваемый нейросетевой метод также является ансамблем нескольких методов:

- англоязычная модель KBIR-Inspec [4], обернутая в русско-английский и англо-русский переводчик. Чтобы обработать варианты плохого перевода, переведенные ключевые слова ищутся в исходном тексте с помощью нечеткого поиска.
- модель WikiNEuRal, обученная на Wikipedia [5] (поддерживает русский язык).

Извлеченные ключевые слова обрабатываются с помощью инженерии подсказок с ruGPT3Large [6], чтобы привести извлеченные ключевые слова к именительному падежу, единственному числу и т.п. Затем удаляются дубликаты с помощью word2vec и DBSCAN.

Более подробное описание метода доступно в [7].

D. Распознавание именованных сущностей (NER)

Задача NER (Named Entity Recognition) заключается в выделении непрерывных фрагментов текста (спанов),

которые обозначают какие-то конкретные объекты, такие как персоны, организации, локации, даты, числа и т. д. Такие фрагменты называются именованными сущностями. Основная цель NER – определить границы и типы именованных сущностей в тексте. На практике такая задача может сводиться к классификации на уровне слов (токенов), которые затем объединяются в спаны.

В данной работе для решения задачи NER используется реализация модели DeepPavlov/rubert-base-cased [8] в библиотеке simpletransformers [9]. Ее применение включает этапы, описанные далее [10].

- Входное предложение разбивается на токены.
- Модель получает на вход последовательность токенов, которая проходит по стеку энкодеров.
- На выходе мы получаем последовательность векторов размера hidden_size (это размерность скрытого слоя сети прямого распространения – 768 для base-версии BERT'a), где каждый вектор соответствует токену из входной последовательности с тем же индексом.

Таким образом, мы получили набор контекстуализированных эмбеддингов – векторных представлений слов или токенов в тексте, которые, в отличие от статических, учитывают контекст, в котором они встречаются.

- Далее результирующий вектор каждого токена пропускается через классификационный слой, благодаря чему, модель предсказывает вероятность принадлежности каждого токена к какому-либо классу NER. В результате работы модели, мы получаем список токенов и соответствующий ему список меток классов.

Затем необходимо интерпретировать и обработать полученные результаты. Стоит упомянуть, что схема разметки, где каждый класс представлен единственной меткой (только название класса), на практике неприменима. Это связано с тем, что при наличии в последовательности токенов, относящихся к одному классу и расположенными непосредственно друг за другом, невозможно будет определить как общее количество сущностей этого класса, так и границы каждой отдельной сущности. Таким образом, задача NER не решается в полной мере.

Поэтому в данной работе, мы также используем BIO-разметку, которая представляет собой специальный способ разметки данных для задачи NER. Эта схема маркировки позволяет указать границы именованных сущностей внутри текста. Акроним BIO означает следующее:

- B (Begin): Слово является началом именованной сущности.
- I (Inside): Слово находится внутри именованной сущности.
- O (Outside): Слово не является частью именованной сущности.

Благодаря использованию такой схемы, осуществляется учет границ сущностей, так как схема позволяет четко указать начало и конец каждой именованной сущности в тексте, что упрощает интерпретацию результатов и обработку текста. Но также это приводит к увеличению общего числа классов, так как каждому классу именованной сущности (например, персона, локация) соответствует несколько подклассов (B-классы для начала сущности и I-классы для внутренних частей сущности). Это может привести к увеличению сложности классификации и уменьшению общей точности модели. Кроме того, использование BIO-схемы может привести к ошибкам в присвоении тегов B и I, что также влияет на качество работы модели.

III. Сравнение методов

A. Описание набора данных

Исходные данные представляют собой 4022 текста технических вакансий с портала «Работа России» [11]. В аннотации вакансий участвовало 56 студентов «Цифровой Кафедры» ЛЭТИ.

Аннотирование заключается в сопоставлении тексту набора триплетов вида [начало, конец, класс сущности], отображающих вхождения сущностей в строку. Например, строка

«...Обязанности: Net Core / .NET 5; EF Core; MSSQL; WPF, EF; Git; JavaScript, HTML, CSS; C#»

делится на следующие триплеты:

...[[Net Core], «Инструмент»], [[.NET 5], «Инструмент»], [[EF Core], «Инструмент»], и т. д., где соответствующие навыки представляются в виде своей начальной и конечной позиции в тексте.

Задачей студентов было по данным вакансиям составить набор таких триплетов, где классами являются: «Технология», «Язык программирования», «Инструмент» и «Soft Skills». Аналогичная задача ставится для рассматриваемых в работе моделей.

B. Обработка данных

Чтобы сравнить методы, описанные в п. «распознавание именованных сущностей (NER)», исходные данные обработаны с учётом особенностей датасета и качества разметки. Во-первых, для каждой сущности установлен самый популярный выбранный класс. Во-вторых, выделенные сущности разделены по знакам препинания по следующим экспериментально подобранным эвристикам:

- по «;» – всегда;
- по «/» и «\» – если в результате более половины сущностей будут на английском;
- по «,» – если в результате число английских сущностей будет превышать 40 %, или если число токенов на сущность после разделения будет менее 2.5.

Таким образом обрабатываются случаи, когда размечающий выделил несколько сущностей (напр. C/C++) как одну.

В-третьих, восстановлены пробелы между знаками пунктуации (после «,», до и после «/»), чтобы облегчить токенизацию. В-четверых, из границ сущностей

исключены пробелы, знаки пунктуации и т.п. В-пятых, в текстах установлены границы предложений исходя из знаков препинания и переносов строк, встречающихся подряд; при необходимости между границами предложения восстановлены «.» (некоторые вакансии в наборе данных вообще не содержат точек).

После приведенных операций создано два набора данных: с полными текстами (датасет I) и только с предложениями, содержащими сущности (датасет II). Последний вариант полезен, т. к. размечающие имеют свойство пропускать некоторые предложения с сущностями, создавая лишние ложно-положительные результаты при распознавании.

C. Методология сравнения

Для оценки качества работы методов вывод методов (получившиеся подстроки из текста) сравнивается с сущностями, выделенными в наборе данных (см. п. «описание набора данных»). Для сравнения выбрана только та часть набора данных, которая не была использована для обучения модели NER (670 вакансий).

Ключевые слова «конвертированы» в сущности следующим образом – в тексте выделяются все вхождения данного ключевого слова. Также, поскольку ключевые слова не имеют класса, в данном сравнении класс не учитывается.

Для результата применения метода к тексту считается F1-score по токенам, т.е. проверяется, выделен ли данный токен методом и выделен ли он в наборе данных.

D. Результаты сравнения

Результаты сравнения методов приведены в табл. 1. Значения F1-score взяты средние по всем документам.

ТАБЛИЦА I. РЕЗУЛЬТАТЫ СРАВНЕНИЯ МЕТОДОВ

Метод	**Avg F1 (I)**	**Avg F1 (II)**
Случайный	0.13	0.25
Ключ. слова	0.19	0.34
Ключ. слова (с окном)	0.28	0.48
Ключ. слова (НС)	0.38	0.53
NER	0.54	0.65

На рис. 1 и рис. 2 приведены гистограммы распределения F1-score по документам для датасета I и датасета II.

Как видно из графиков, распределения для датасета II оказываются «сдвинуты вправо», но общая очередность результатов остается такой же. Все методы показали результаты лучше, чем случайный, но только модель NER смогла показать F1 ∈ [0.8, 1.0] для существенного количества документов на датасете II.

Меньшее количество таких результатов на датасете I не противоречит предположению, что предложения без выделенных сущностей приводят к необоснованному росту ложно-положительных результатов.

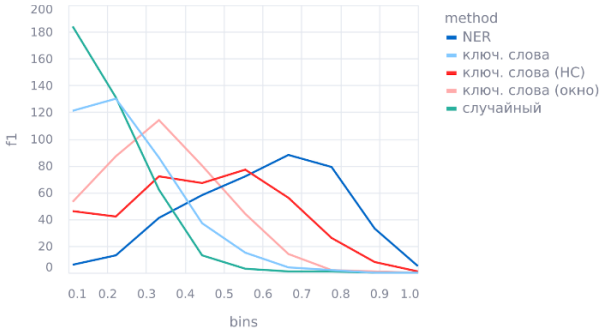

Рис. 1. Распределение F1-score по документам датасета I

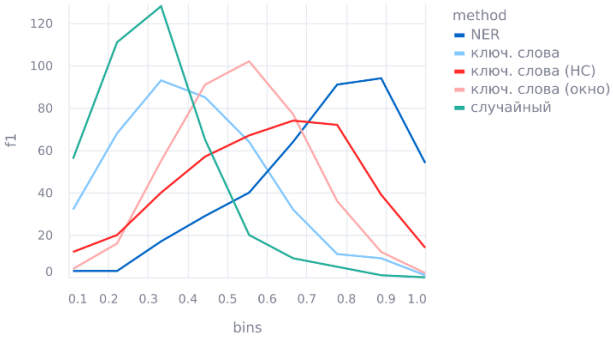

Рис. 2. Распределение F1-score по документам датасета II

Чтобы оценить качество работы метода NER, произведена отдельная оценка с учётом классификации сущностей на наборе данных II. Итоговая матрица ошибок представлена в табл. 2.

ТАБЛИЦА II. ИТОГОВАЯ МАТРИЦА ОШИБОК

Истинный \ Предсказанный класс	**Technology**	**Method**	**Soft Skills**	**O**	**Prog Language**	**Tool**
Technology	1016	293	30	1528	33	201
Method	172	1570	89	3185	15	96
SoftSkills	12	105	670	705	2	3
O	392	1286	486	41561	96	494
Prog Language	3	0	0	109	1220	16
Tool	158	83	26	1122	55	3049

Например, 3 "ProgLanguage" были неверно классифицированы как "Technology"; 109 не были распознаны в принципе.

На основе этих данных можно определить структуру ошибок модели. Результаты приведены в табл. 3.

ТАБЛИЦА III. F1-SCORE ДЛЯ ВЫДЕЛЕННЫХ КАТЕГОРИЙ НАВЫКОВ

Класс	**F1 (класс vs. всё)**	**F1 (всё кроме “O” vs “O”)**	**Разница**
Technology	0,42	0,62	0,20
Method	0,37	0,46	0,09
SoftSkills	0,48	0,57	0,09
O	0,90		
ProgLanguage	0,88	0,92	0,04
Tool	0,73	0,81	0,08

Как видно, лучше всего из значимых классов распознаются "ProgLanguage" и "Tool". Класс "Technology" чаще всего путается с другими классами, в основном с "Tool" и "Method", что привело к снижению F1 по данному классу на 0.2.

IV. Заключение

В ходе работы рассмотрены четыре метода на основе извлечения ключевых слов и модель NER для извлечения именованных сущностей. Можно заметить, что последняя модель выдает лучшие результаты в отличие от остальных методов (F1_NER = 0.65 на имеющихся данных). Однако в отличие от методов с ключевыми словами, модель NER неприменима для вакансий без вхождений известных ей классов. Также то, что F1_NER < 1 показывает, что модель можно улучшить.

Дальнейшая работа с нейронной моделью по извлечению ключевых сущностей может быть связана с улучшением полученных результатов путем использования LLM модели для предобработки данных, а также обработки выходных результатов модели. Еще одним из возможных улучшений может стать расширение работы на тексты рабочих программ.

Итоговое решение будет внедрено в ИС «Индивидуальные образовательные траектории», в рамках которой у студентов будет доступ к просмотру возможных вакансий, содержание которых соответствуют полученным в ходе обучения компетенциям.

Список литературы

[1] Campos R. и др. YAKE! Keyword extraction from single documents using multiple local features // Information Sciences. 2020. Т. 509. С. 257–289.

[2] Florescu C., Caragea C. PositionRank: An Unsupervised Approach to Keyphrase Extraction from Scholarly Documents // Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vancouver, Canada: Association for Computational Linguistics, 2017. С. 1105–1115.

[3] Wan X., Xiao J. CollabRank: Towards a Collaborative Approach to Single-Document Keyphrase Extraction // Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008). : Coling 2008 Organizing Committee, 2008. С. 969–976.

[4] Kulkarni M. и др. Learning Rich Representation of Keyphrases from Text [Электронный ресурс] // arXiv preprint arXiv:2112.08547. 2022. URL: http://arxiv.org/abs/2112.08547 (дата обр. 14.03.2024)

[5] Tedeschi S. и др. WikiNEuRal: Combined Neural and Knowledge-based Silver Data Creation for Multilingual NER // Findings of the Association for Computational Linguistics: EMNLP 2021. Punta Cana, Dominican Republic: Association for Computational Linguistics, 2021. С. 2521–2533.

[6] ai-forever/rugpt3large_based_on_gpt2 Hugging Face [Электронный ресурс]. URL: https://huggingface.co/ai-forever/rugpt3large_based_on_gpt2 (дата обращения: 19.06.2023).

[7] Kholod I.I., Korytov P.V., Sorochina M.V. Application of Neural Network Keyword Extraction Methods for Student's CV Compilation from Discipline Work Programs // 2023 XXVI International Conference on Soft Computing and Measurements (SCM). СПб.: IEEE, 2023. С. 143–146.

[8] DeepPavlov/rubert-base-cased Hugging Face [Электронный ресурс]. URL: https://huggingface.co/DeepPavlov/rubert-base-cased (дата обращения: 16.03.2024).

[9] Simple Transformers [Электронный ресурс]. URL: https://simpletransformers.ai/ (дата обращения: 16.03.2024).

[10] Devlin J. и др. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding [Электронный ресурс] // arXiv:1810.04805v2, 2019. URL: https://arxiv.org/abs/1810.04805 (дата обращения: 16.03.2024).

[11] Работа России. Открытые данные [Электронный ресурс]. URL: https://trudvsem.ru/opendata/ (дата обращения: 16.03.2024)