УДК 004.81

АЛГОРИТМИЧЕСКИЙ ФРЕЙМВОРК ДЛЯ ИЗВЛЕЧЕНИЯ ИНФОРМАЦИОННОГО ЯДРА ВЕБ-СТРАНИЦЫ

Х. Салем[1] [0000-0002-9143-5231], А. С. Тощев[2] [0000-0003-4424-6822]

[1]*Университет Иннополис, г. Иннополис, Россия*
[2]*Казанский федеральный университет, г. Казань, Россия*
[1]h.salem@innopolis.ru, [2]atoschev@kpfu.ru

Аннотация

Представлен новый точный алгоритм MCE извлечения основного содержимого с новостных веб-сайтов. Предложенный алгоритм использует анализ структуры объектной модели документа (DOM) и метрики плотности контента для идентификации и извлечения информационного ядра веб-страницы. Реализованный подход объединяет три ключевые особенности: максимальное количество прямых дочерних элементов с текстом, максимальное текстовое содержимое без дочерних элементов, содержащих текст, и ближайшее расположение к средней глубине узла. Алгоритм продемонстрировал лучшую производительность по сравнению с существующими решениями, такими как Boilerpipe и Readability, достигая 99.96% точности, 99.69% полноты и 99.80% F1-меры на использованном комплексном наборе данных из 500 разнообразных веб-страниц. Языково-независимый дизайн делает алгоритм особенно эффективным для извлечения мультиязычного контента, включая языки со сложной структурой, такие, например, как арабский.

Ключевые слова: *NLP, извлечение данных, языково-независимый алгоритм, RAG (Retrieval-Augmented Generation).*

Введение

Современные веб-страницы содержат обилие разнообразной информации [1], где основной текст окружен шумовыми элементами — меню, рекламой, шапками, подвалами [2]. Задача выделения информационного ядра сводится к отделению тела статьи от незначимых компонентов [3], как показано на рис. 1. Отделение информационного ядра от шаблонных элементов представляет серьезную

проблему [4], особенно критичную для функции browsing (поиск в вебе) в приложениях с использованием какой-либо большой языковой модели LLM (ChatGPT, Perplexity, DeepSeek) [5, 6].

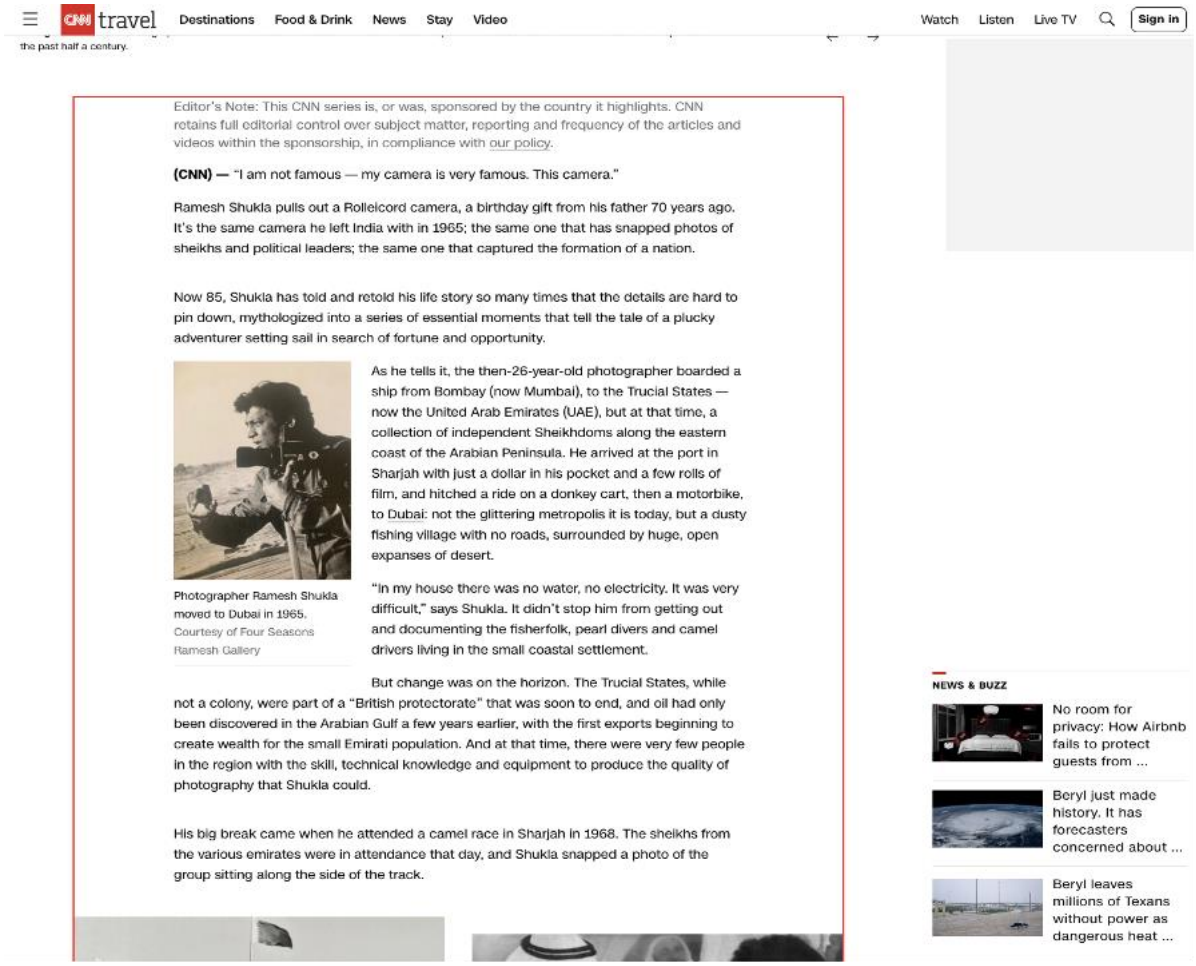


Рис. 1. Пример обнаружения информационного ядра в новостной статье (обведено красным)

Системы RAG (генерации с подключением поиска/извлечения) сначала собирают текст со страниц, а затем передают его языковой модели, чтобы она давала более точные ответы. Однако LLM-сервисы не являются дешевыми: их бизнес-модель основана на оплате токенов, а точность извлечения влияет на стоимость. Популярные библиотеки для извлечения текста с веб-страниц (Boilerpipe, Readability, Trafilatura) нередко захватывают посторонние элементы, что приводит к росту объема данных и увеличению затрат.

Настоящая работа предлагает алгоритм, определяющий информационное ядро веб-страницы и использующий связи в DOM, эвристические методы и анализ плотности текста с фильтрацией ненужных элементов.

ОПИСАНИЕ АЛГОРИТМА

Алгоритм решает задачу идентификации элемента в веб-документе с наибольшим количеством дочерних элементов и слов. Эта идея основана на гипотезе, что именно такие элементы служат основными контейнерами значимого текста для анализа, резюмирования или улучшения контента. Для проверки и практической реализации гипотезы была использована типичная структура сайтов статей и блогов, где основной блок обычно включает несколько дочерних элементов “ ” или “<p>” с текстом. Например, современные CMS с визуальными редакторами генерируют HTML, следующий данной структуре (рис. 2).

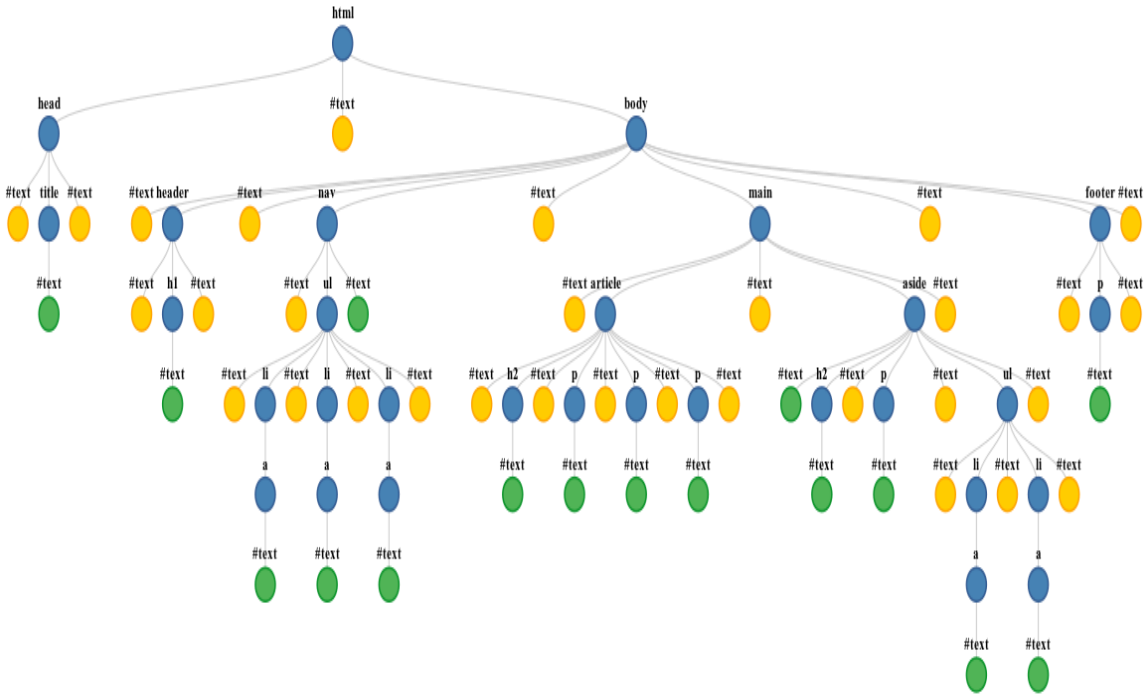


Рис. 2. Максимальное количество прямых дочерних элементов с текстом

Пусть *N* – множество всех узлов в дереве DOM. Для каждого узла $n \in N$ определим следующие величины (1-3):

$C(n)$ = количество прямых дочерних элементов узла *n*, (1)

$T(n)$ = общая длина текста прямых дочерних элементов узла *n*. (2)

Мы стремимся найти узел $\mathrm{n}_{\max}$, который максимизирует комбинацию $C(n)$ и $T(n)$:

$$\mathrm{n}_{\max} = \arg\max_{n \in N} \big(C(n), T(n)\big). \quad (3)$$

Еще одной характерной особенностью информационного ядра является наличие значительного объема текста в узле без текстовых дочерних элементов. Другими словами, элемент html “div” с крупным текстовым блоком рассматривается как более вероятный кандидат на основной контент. Формально для узла $n \in N$ обозначим через $L(n)$ длину текста узла. Пусть $H(n)$ — функция, принимающая значение 1, если у узла есть дочерние элементы с текстом, и 0 — в противном случае. Мы ищем узел $n_{\max}$, максимизирующий $L(n)$ при условии отсутствия дочерних текстовых элементов (4).

$$\mathrm{n}_{\max} = \arg\max_{n \in N} (L(n)\,(1 - H(n))). \quad (4)$$

Последняя выявленная нами особенность связана с расположением узла на глубине, близкой к средней по дереву DOM. Она основана на наблюдении, что глубина HTML-структуры отражает положение элемента на странице: слишком поверхностные узлы обычно содержат навигацию или служебные блоки, а чрезмерно глубокие — второстепенные элементы. Пусть $D(n)$ обозначает глубину узла n в DOM (корневой узел имеет глубину 0). Среднюю глубину M определим как (5).

$$M = |N|/2, \quad (5)$$

где $|N|$ – общее количество узлов. Определим узел n_{closest} с глубиной, ближайшей к M:

$$n_{\mathrm{closest}} = \mathrm{argmin}_{n \in N} |D(n) - M|. \quad (6)$$

Итоговый набор узлов основного содержимого S (7) получается объединением узлов, идентифицированных тремя особенностями:

$$S = \{n_{\max-\mathrm{children}}, n_{\max-\mathrm{text}}, n_{\mathrm{closest}-\mathrm{middle}}\}. \quad (7)$$

Для определения первичного узла основного содержимого используем геометрический подход, основанный на векторах признаков. Каждый узел представим как 3D-вектор [количество дочерних элементов, количество слов, глубина]. Центроид c множества m 3-мерных векторов $x_i = [x_{i1}, x_{i2}, x_{i3}]$ зададим так (8).

$$c_j = \left(\frac{1}{m}\right) \sum_{i=1}^{m} x_{\text{i}} \quad \text{для } j = 1, 2, 3. \tag{8}$$

Для каждой компоненты c_{j} центроида имеем (9)

$$c_{\text{j}} = \left(\frac{1}{m}\right) \sum_{i=1}^{m} x_{ij} \text{ , для } j = 1, 2, 3, \tag{9}$$

где x_i=[Количество дочерних элементов с текстом *i*, Количество слов *i*, Глубина *i*]. Для измерения близости используем манхэттенское расстояние (10)

$$d(a, b) = |a_1 - b_1| + |a_2 - b_2| + |a_3 - b_3|. \tag{10}$$

ЭКСПЕРИМЕНТАЛЬНАЯ ОЦЕНКА

Алгоритм MCE был валидирован на репрезентативной выборке веб-документов [7, 13] размером 500 документов (уровень доверия 95%, предел ошибки 5%). Была также использована стратифицированная случайная техника выборки по языку, типу содержимого и сложности макета. Оценка проводилась с использованием стандартных метрик: точность, полнота, F1-мера и accuracy. Сравнительный анализ с алгоритмами Boilerpipe [8] и Readability [9] показал превосходящую производительность MCE по всем метрикам (табл. 1). Результаты сравнения обобщены и представлены в табл. 1.

Табл. 1. Метрики классификации (основной датасет)

Words Count Text	Words Count MCE	Words Count Readability	Words Count Boilerpipe	TP-C	FP-C	FN-C	TP-R	FN-R
232	232	247	131	232	0	0	232	0
220	220	239	257	220	0	0	220	0
208	208	187	199	208	0	0	208	21
432	434	404	437	434	2	0	434	30
425	425	442	341	425	0	0	425	0
220	220	152	172	220	0	0	220	68

В табл. 2-4 приведены полученные значения метрик Precision, Recall, F1-score и Accuracy соответственно для методов MCE, Readability, Boilerpipe.

Табл. 2. Метрики классификации для метода MCE

Original Text	Precision	Recall	F1-score	Accuracy
232	1	1	1	1
220	1	1	1	1
208	1	1	1	1
432	0.995	1	0.998	0.995
425	0.962	1	0.981	0.962
220	0.764	0.866	0.764	0.764

Табл. 3. Метрики классификации для метода Readability

Original Text	Precision	Recall	F1-score	Accuracy
232	0.939	1	0.969	0.939
220	0.920	1	0.959	0.920
208	0.908	1	0.951	0.908
432	0.935	1	0.967	0.935
425	0.962	1	0.980	0.962
220	0.782	0.878	0.782	0.782

Табл. 4. Метрики классификации для метода Boilerpipe

Original Text	Precision	Recall	F1-score	Accuracy
232	0.565	0.722	0.565	0.565
220	0.874	0.933	0.874	0.874
208	1	0.957	0.978	1
432	0.988	1	0.994	0.988
425	1	0.802	0.890	1
220	1	0.782	0.878	1

Табл. 5 обобщает усредненные по множеству запусков метрики производительности (Precision, Recall, F1-Score, Accuracy) трех алгоритмов классификации: MCE, Readability и Boilerpipe и дает оценку эффективности каждого алгоритма.

Табл. 5. Метрики производительности классификации для датасета

Metric	MCE	Readability	Boilerpipe
Precision	0.9996%	0.9236%	0.9425%
Recall	0.9969%	0.9419%	0.8919%
F1-Score	0.9980%	0.9183%	0.8997%
Accuracy	0.9965%	0.8654%	0.8343%

Дополнительная оценка на наборе данных Webz.io [11], который представляет собой репозиторий публичных новостных статей с еженедельными выпусками тематических датасетов, содержащих около 1000 новостных статей с богатой метаинформацией, подтвердила эффективность всех алгоритмов, перечисленных выше, на различных типах контента (см. табл. 6).

Табл. 6. Метрики производительности классификации для датасета Webz.io

Metric	MCE	Readability	Boilerpipe
Precision	0.9615%	0.9136%	0.9225%
Recall	0.9597%	0.9219%	0.8819%
F1-Score	0.9468%	0.9133%	0.8897%
Accuracy	0.9212%	0.8554%	0.8143%

Как видим, алгоритм MCE сохраняет лидерство с Precision (96.15%), Recall (95.97%) и F1-Score (94.68%), хотя эти значения примерно на 3–4 процентных пункта ниже, чем его производительность на основном датасете. Алгоритмы Readability и Boilerpipe вновь демонстрируют сопоставимые, но более слабые результаты. Примечательно, что все три алгоритма демонстрируют снижение Accuracy на 2–5% на этом датасете, что позволяет предположить наличие более сложных случаев классификации в контенте Webz.io.

Анализ ошибок выявил ложные положительные результаты (неправильная классификация рекламных блоков, навигационных меню, комментариев) и ложные отрицательные результаты (пропуск контента из-за нестандартной структуры HTML, динамического контента, сложных макетов). Языковой анализ показал высокую производительность на латинских языках (> 95%), сопоставимую на кириллических языках, и особую эффективность на арабском языке (94.2%). Алгоритм реализован в расширении Chrome Elkateb[12] и развернут в производстве на https://gptafser.com, где сгенерировал JSON-LD для 90000 страниц за 12 мин, улучшив индексацию и увеличив трафик.

ЗАКЛЮЧЕНИЕ

Предложенный алгоритм MCE продемонстрировал превосходство над существующими решениями с показателями точности 96.15% и полноты 95.97% при организации поиска. Ключевыми преимуществами алгоритма MCE являются языковая независимость, высокая точность, вычислительная эффективность и открытый исходный код. MCE превосходит Readability на 4.79% и Boilerpipe на 3.90% по точности, соответственно на 3.78% и 7.78% по полноте. Ограничения включают затруднения с динамическим контентом JavaScript, снижение точности на нестандартных макетах и оптимизацию только для текстового контента.

MCE находит применение в системах RAG, агрегации новостного контента и мобильной оптимизации. Техническая реализация обеспечивает пропускную способность до 1000 страниц в минуту и использование памяти 50–100 МБ для типичной веб-страницы.

СПИСОК ЛИТЕРАТУРЫ

1. Jach T., Kaczmarek M., Kaczmarek T. Web content extraction: A survey of techniques and applications // Information Sciences. 2021. Vol. 570. P. 378–400. https://doi.org/10.1016/j.ins.2021.04.014

2. Brown K., Davis L. Content density metrics for web page analysis // Information Retrieval Journal. 2020. Vol. 23, No. 4. P. 512–530. https://doi.org/10.1007/s10791-020-09380-4

3. Gottron T. Content extraction from web pages // Proceedings of the 2008 ACM Symposium on Applied Computing. 2008. P. 1160–1164. https://doi.org/10.1145/1363686.1363939

4. Insa D., Silva J., Tomás C. Using content extraction for web page classification // Information Processing & Management. 2013. Vol. 49, No. 1. P. 235–250. https://doi.org/10.1016/j.ipm.2012.05.005

5. Qi X., Zhang Y., Wang L. Investigating the impact of content extraction on sentiment analysis // Information Processing & Management. 2024. Vol. 61, No. 1. 103245. https://doi.org/10.1016/j.ipm.2023.103245

6. Zhang W., Liu X. Machine learning approaches to content extraction // Pattern Recognition. 2022. Vol. 125. 108456. https://doi.org/10.1016/j.patcog.2022.108456

7. Seitl F., Kovarik T., Mirshahi S. et al. Assessing the quality of information extraction // arXiv preprint. 2024. arXiv:2404.04068. https://arxiv.org/abs/2404.04068.

8. Kohlschütter C. Boilerpipe: A Python library for extracting text from HTML // GitHub Repository. 2010. https://github.com/misja/python-boilerpipe

9. Mozilla Foundation. Readability: A Python library for extracting article content from HTML // GitHub Repository. 2020. https://github.com/mozilla/readability

10. Bevendorff J., Gupta S., Kiesel J., Stein B. An empirical comparison of web content extraction algorithms // Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2023. P. 2594–2603. https://doi.org/10.1145/3539618.3591920

11. Webz.io. Webz.io Free News Datasets // Webz.io. 2023. https://webz.io/free-news-datasets

12. Research Team. Elkateb: Browser Extension for Content Extraction // Browser Extension. 2024. https://github.com/elkateb/extension

13. Бобырь М.В., Милостная Н.А., Булатников В.А. The fuzzy filter based on the method of areas' ratio // Applied Soft Computing. 2022. Vol. 117. 108449. https://doi.org/10.1016/j.asoc.2022.108449

AN ALGORITHMIC FRAMEWORK FOR ACCURATELY EXTRACTING MAIN CONTENT FROM NEWS WEBSITES

H. Salem[1] [0000-0002-9143-5231], A. S. Toschev[2] [0000-0003-4424-6822]

[1]*Innopolis University, Innopolis, Russia*

[2]*Kazan Federal University, Kazan, Russia*

[1]h.salem@innopolis.ru, [2]atoschev@kpfu.ru

Abstract

A new precise MCE algorithm for extracting the main content from news websites is presented. The proposed algorithm uses analysis of the Document Object Model (DOM) structure and content density metrics to identify and extract the informational core of a web page. The implemented approach combines three key features: the maximum number of direct child elements containing text, the maximum textual content without child elements containing text, and the closest position to the average node depth. The algorithm demonstrated superior performance compared to existing solutions such as Boilerpipe and Readability, achieving 99.96% precision, 99.69% recall, and 99.80% F1-score on a comprehensive dataset of 500 diverse web pages. Its language-independent design makes the algorithm particularly effective for extracting multilingual content, including languages with complex structures such as Arabic.

Keywords: *NLP, Data Extraction, Language-Independent Algorithm, RAG (Retrieval-Augmented Generation).*

REFERENCES

1. Jach T., Kaczmarek M., Kaczmarek T. Web content extraction: A survey of techniques and applications // Information Sciences. 2021. Vol. 570. P. 378–400. https://doi.org/10.1016/j.ins.2021.04.014

2. Brown K., Davis L. Content density metrics for web page analysis // Information Retrieval Journal. 2020. Vol. 23, No. 4. P. 512–530. https://doi.org/10.1007/s10791-020-09380-4

3. Gottron T. Content extraction from web pages // Proceedings of the 2008 ACM Symposium on Applied Computing. 2008. P. 1160–1164. https://doi.org/10.1145/1363686.1363939

4. Insa D., Silva J., Tomás C. Using content extraction for web page classification // Information Processing & Management. 2013. Vol. 49, No. 1. P. 235–250. https://doi.org/10.1016/j.ipm.2012.05.005

5. Qi X., Zhang Y., Wang L. Investigating the impact of content extraction on sentiment analysis // Information Processing & Management. 2024. Vol. 61, No. 1. 103245.
https://doi.org/10.1016/j.ipm.2023.103245

6. Zhang W., Liu X. Machine learning approaches to content extraction // Pattern Recognition. 2022. Vol. 125. 108456.
https://doi.org/10.1016/j.patcog.2022.108456

7. Seitl F., Kovarik T., Mirshahi S. et al. Assessing the quality of information extraction // arXiv preprint. 2024. arXiv:2404.04068.
https://arxiv.org/abs/2404.04068

8. Kohlschütter C. Boilerpipe: A Python library for extracting text from HTML // GitHub Repository. 2010.
https://github.com/misja/python-boilerpipe

9. Mozilla Foundation. Readability: A Python library for extracting article content from HTML // GitHub Repository. 2020.
https://github.com/mozilla/readability

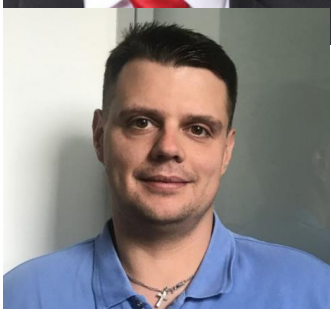
10. Bevendorff J., Gupta S., Kiesel J., Stein B. An empirical comparison of web content extraction algorithms // Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2023. P. 2594–2603. https://doi.org/10.1145/3539618.3591920

11. Webz.io. Webz.io Free News Datasets // Webz.io. 2023.
https://webz.io/free-news-datasets

12. Research Team. Elkateb: Browser Extension for Content Extraction // Browser Extension. 2024.
https://github.com/elkateb/extension

13. Bobyr M.V., Milostnaya N.A., Bulatnikov V.A. The fuzzy filter based on the method of areas' ratio // Applied Soft Computing. 2022. Vol. 117. 108449.
https://doi.org/10.1016/j.asoc.2022.108449

СВЕДЕНИЯ ОБ АВТОРАХ



Хамза САЛЕМ – аспирант, Университет Иннополис / Лаборатория программной инженерии, г. Иннополис.

Hamza SALEM – PhD student, Innopolis University / Software Engineering Lab, Innopolis.

email: h.salem@innopolis.ru

ORCID: 0000-0002-9143-5231

ТОЩЕВ Александр Сергеевич – доцент, к. н., КФУ / Институт информационных технологий и интеллектуальных систем / Кафедра программной инженерии, г. Казань.

Alexander Sergeevich TOSCHEV – Associate Professor, Ph.D., KFU / Institute of Information Technologies and Intelligent Systems / Department of Software Engineering, Kazan.

email:atoschev@kpfu.ru

ORCID: 0000-0003-4424-6822

Материал поступил в редакцию 18 августа 2025 года