УДК 004.5, 681.518, 004.032.26

ВЕБ-СЕРВИС ДЛЯ АВТОМАТИЗИРОВАННОЙ ПРОВЕРКИ РЕШЕНИЙ УЧЕБНЫХ ЗАДАНИЙ ПО HTML-ВЕРСТКЕ

Рыженков Артем Дмитриевич[1], Лукьянов Константин Валерьевич[2]

[1]Студент;
Государственный университет «Дубна»;
Россия, 141980, Московская обл., г.Дубна, ул. Университетская, д. 19;
Инженер-программист;
АО Нэксайн;
Россия, 199155, г.Санкт-Петербург, ул. Уральская, д. 4Б;
e-mail: ryzhenkovart@yandex.ru.

[2]Кандидат физико-математических наук, доцент;
Государственный университет «Дубна»,
Россия, 141980, Московская обл., г.Дубна, ул. Университетская, д. 19;
Старший научный сотрудник;
Объединенный институт ядерных исследований;
Россия, 141980, Московская обл., г.Дубна, ул. Жолио-Кюри, д. 6;
e-mail: luky@jinr.ru.

Задача автоматизированной проверки заданий по html-верстке востребована в процессе обучения основам веб-технологий как в учебных заведениях, так и при самоподготовке по соответствующим направлениям. В статье рассматриваются особенности проверки таких заданий, которые необходимо учитывать в случае автоматизированной проверки. Приводится алгоритм системы для реализации в виде веб-сервиса. Особенностью предлагаемой системы является реализация сравнения результата верстки с эталонным образцом различными методами, в том числе с использованием нейросети VGG16. Показано, что при должном задании порога схожести система сравнения изображений дает удовлетворительные результаты и может применяться для автоматизированной проверки заданий по верстке.

Ключевые слова: информационная система, веб-сервис, html-верстка, headless-браузер, сравнение изображений, нейросети.

Для цитирования:
Рыженков А. Д., Лукьянов К. В. Веб-сервис для автоматизированной проверки решений учебных заданий по html-верстке // Системный анализ в науке и образовании: сетевое научное издание. 2023. № 3. С. 71-77. EDN: VAUWXI. URL : https://sanse.ru/index.php/sanse/article/view/591.

WEB SERVICE FOR AUTOMATED VERIFICATION OF SOLUTIONS FOR HTML-LAYOUT EDUCATIONAL TASKS

Ryzhenkov Artem D.[1], Lukyanov Konstantin V.[2]

[1]Student;
Dubna State University;
19 Universitetskaya Str., Dubna, Moscow region, 141980, Russia;
Software engineer;
Nexign;
4B Uralskaya Str., Saint-Petersburg, 199155, Russia;
e-mail: ryzhenkovart@yandex.ru.

[2]PhD in Physical and Mathematical Sciences, associate professor;
Dubna State University;
19 Universitetskaya Str., Dubna, Moscow region, 141980, Russia;
Senior Researcher;
Joint Institute for Nuclear Research;
6 Joliot-Curie Str., Dubna, Moscow region, 141980, Russia;
e-mail: luky@jinr.ru.

The task of automated verification of html layout tasks is in demand in the process of learning the basics of web technologies both in educational institutions and during self-training in the relevant areas. The article discusses the features of checking such tasks, which must be taken into account in the case of automated verification. The algorithm of the system for implementation in the form of a web service is given. A feature of the proposed system is the implementation of comparing the layout result with the reference sample by various methods, including using the VGG16 neural network. It is shown that when the similarity threshold is properly set, the image comparison system gives satisfactory results and can be used for automated verification of layout tasks.

Keywords: information system, web service, html layout, headless browser, image comparison, neural networks.

For citation:
Ryzhenkov A. D., Lukyanov K. V. Web service for automated verification of solutions for html-layout educational tasks. *System analysis in science and education*, 2023;(3):71-77 (in Russ). EDN: VAUWXI. Available from: https://sanse.ru/index.php/sanse/article/view/591.

Особенности выполнения заданий по html-верстке

В Институте системного анализа и управления Государственного университета «Дубна» читается ряд курсов, связанных с веб-разработкой [1]. В некоторые из них включены задания по *html*-верстке разной степени сложности. Одним из типовых заданий является верстка *html*-страницы по заданному макету.

Макет представляет из себя совокупность текстовых и графических объектов. Корректно выполненным заданием считается разработанный студентом *html*- и *css*-код, а также набор изображений, вместе реализующие визуальное размещение текста и изображений в окне браузера в соответствии с макетом. При этом не требуется с точностью до пикселя совпадение визуального результата верстки исходному макету (эталону). Кроме того, допускается замена исходного текста на другой, например, замена контекста флористики на автомобильную. Основным критерием оценки результата верстки является верное взаимное расположение объектов, отступы, примерное соответствие объема текста первоначальному дизайну.

Преподаватель осуществляет проверку индивидуально по каждому выполненному студентом заданию. Сюда входят этапы сохранения на локальный компьютер архива с версткой, распаковка и запуск страницы в браузере, визуальная оценка результата и сравнение с эталоном. Очевидно, что само время оценки чаще всего меньше подготовительных процедур. Кроме этого, возможен субъективный фактор, потенциально влияющий на оценку. Автоматизация проверки решений учебных заданий позволила бы с одной стороны сократить временные затраты преподавателя, а с другой – добавить студентам возможность самоконтроля.

Алгоритм автоматизированной проверки

Выше было указано, что потенциальными пользователями системы являются преподаватели и студенты, т.е. неограниченный круг лиц. В этом случае удобной формой реализации подобной системы может быть веб-сервис.

Основными задачами разрабатываемой в виде веб-сервиса автоматизированной системы, помимо обеспечения взаимодействия с пользователем, являются:

1. создание растрового изображения (скриншота) на основе *html*- и *css*-кода;
2. сравнение полученного изображения с эталоном.

Первая задача имеет достаточный накопленный опыт. К основным решениям можно отнести сторонние веб-сервисы, предоставляющие *API* для генерации скриншотов [2,3]; создание скриншотов с помощью нативных средств браузера [4,5]; наконец, использование *headless*-браузеров [6]. Последний вариант представляет интерес применительно к рассматриваемой задаче, так как может быть использован в областях, где требуется автоматизация взаимодействия с веб-страницами.

Первым *headless*-браузером считается *HtmlUnit* [7], появившийся в 2002 году. Браузер был целиком написан на *Java*, позволял манипулировать веб-сайтами на высоком уровне, включая заполнение и отправку форм, нажатие гиперссылок. Также он обеспечивал доступ к структуре и деталям полученных веб-страниц и частично эмулировал поведение браузера.

С тех пор развитие *headless*-браузеров шло по нарастающей: в 2011 появился *PhantomJS* [8]; в 2017 – *Headless Firefox* [9,10] и *Headless Chrome* [11]. Позже появились и другие. Современные *headless*-браузеры могут использоваться для создания скриншотов веб-страниц, для скрейпинга информации, для тестирования веб-приложений и других задач [12].

На момент написания статьи браузером *Google Chrome* пользуется свыше 60% интернет-пользователей [13]. В связи с этим представляется логичным для задачи создания скриншота использовать *headless*-версию именно этого браузера.

Что касается задачи сравнения изображений, в литературе часто встречаются упоминания таких методов, как метод среднеквадратичной ошибки [14], расчет индекса структурного сходства [15], а также использование нейросетей [16].

Метод среднеквадратичной ошибки сравнивает изображения путем измерения среднеквадратичного отклонения между пикселями этих изображений. Несмотря на простоту и эффективность, метод неустойчив к шуму на изображении, которые могут существенно исказить результаты сравнения, например, наличие артефактов, внезапных изменений освещения, наложения текста.

Индекс структурного сходства сравнивает изображения путем сопоставления их структурных характеристик, таких как контрастность, яркость, текстура и детализация. Этот метод является одним из наиболее точных и широко используемых для сравнения изображений. В то же время, он не учитывает контекст изображений и может не быть достаточно точным при сравнении изображений, содержащих объекты, которые могут влиять на контекст всего изображения.

Нейросетевой подход часто используется для задач распознавания объектов, но также может применяться и для сравнения изображений. Так, нейросеть *VGG16* позволяет точно измерять сходство между двумя изображениями, учитывая глубокие признаки и контекст изображений [17].

Определив возможные технологии и методы для рассмотренных основных задач, составим алгоритм работы системы в целом:

1. Пользователь веб-сервиса загружает код сверстанной страницы, представленной в виде архива с файлами, в систему. Приложение принимает этот код для дальнейшей обработки.
2. Эталон сравниваемого изображения также загружается пользователем в систему либо берется из базы системы.
3. После загрузки файлы архива разархивируются. Происходит создание скриншота страницы с использованием *Headless Chrome*.
4. Происходит сравнение скриншота с эталонным изображением с использованием рассмотренных выше методов.
5. Результаты сравнения представляются в виде метрик схожести, которые помогают пользователю сервиса оценить, насколько близка страница к эталону. Метрики могут быть представлены в виде числовых значений, процентного соотношения или других показателей, которые отражают степень схожести страницы.
6. Определяется значение границы, при котором изображения считаются схожими.
7. На основе полученных данных делается вывод, похожа ли пользовательская сверстанная страница на эталонное изображение.

Реализация веб-сервиса

Для реализации веб-сервиса был обоснован [18] выбор дополнительных технологий и библиотек, таких как *Django* и *Python*, *Selenium* и других. Среди них необходимо отметить *EasyOCR* [19] и *OpenCV* [20], которые будут применяться для распознавания и закраски текстовых фрагментов на сравниваемых изображениях с целью повышения точности сравнения.

Была разработана и описана имплементационная диаграмма классов (рис. 1). Общий интерфейс системы представлен на рис. 2.

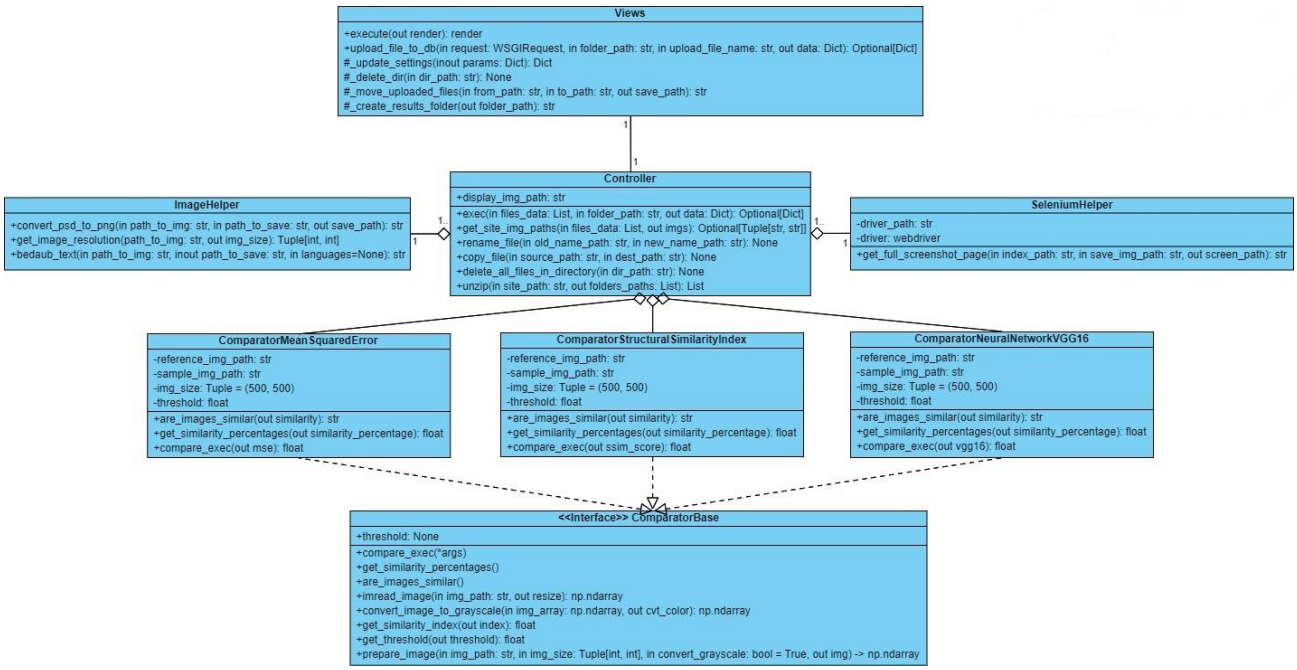

Рис. 1. Веб-сервис в виде имплементационной диаграммы классов

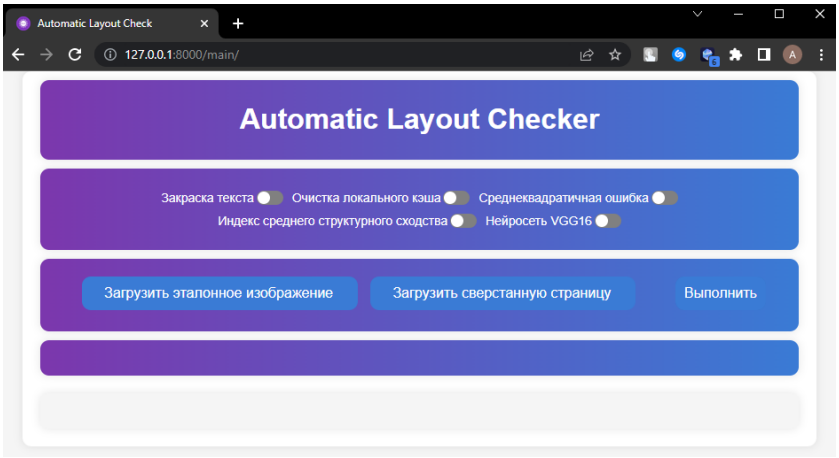

Рис. 2. Интерфейс веб-сервиса

Кроме загрузки сравниваемых изображений пользовательский интерфейс включает следующие опции:

1. Закраска текста – опция, позволяющая закрашивать текст в эталонном и сравниваемом изображениях. Эта настройка может повысить качество сравнения изображений в случае различия текста в эталоне и сверстанной странице.
2. Очистка локального кэша – опция, отвечающая за очистку локальных файлов, хранящихся в папке results. Она необходима по большей части для архивирования результатов загрузки файлов в программу.
3. Среднеквадратичная ошибка – опция, отвечающая за сравнения изображений по методу среднеквадратичной ошибки.
4. Индекс среднего структурного сходства – опция, отвечающая за сравнения изображений по методу индекса среднего структурного сходства.
5. Нейросеть *VGG16* – опция, отвечающая за сравнения изображений с помощью нейросети.

После сравнения эталона и скриншота, сформированного на основе загруженного *html*- и *css*-кода, в интерфейсе отображается таблица результатов, а также два блока со сравниваемыми изображениями (см. рис. 3).

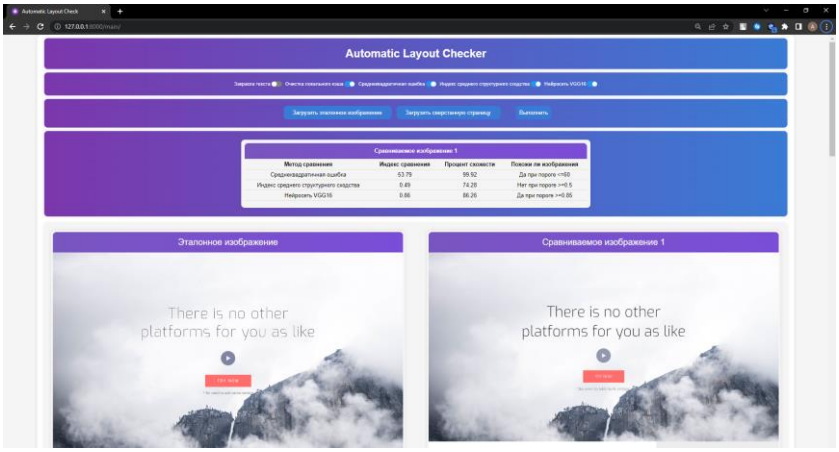

Рис. 3. Результаты сравнения двух изображений

Обоснование порога схожести при сравнении изображений

Порог схожести – это установленное значение, которое определяет похожи ли два изображения. Для определения порога схожести между изображениями использован метод перебора с визуальной экспертной оценкой сходства изображений, который предполагает выполнение следующих шагов:

1. Собрать набор изображений для сравнения. Набор должен содержать изображения, которые можно считать похожими и различными.
2. Произвести сравнение изображений для каждой пары изображений по трем методам с учетом максимального и минимального значений схожести изображений и учетом закраски текста, занести значения индексов в таблицу.
3. Произвести визуальную оценку изображений, определить можно ли изображения считать схожими.
4. Занести данные в таблицу, в которой будет содержаться информация по методу сравнения, полученному значению индекса схожести и утверждению похожи ли изображения.
5. Определить порог на основе данных [21-24].

Набор изображений состоит из 10 различных сверстанных страниц с разной степенью схожести с эталонным образцом. Минимальное значение сходства равняется 0, максимальное – 1.

Результаты расчета значений сходства с применением нейросети *VGG16* приведены в таблицах 1 и 2. Сравнивая значения индексов таблиц с закраской текста и без закраски текста, соответствующих столбцов, можно заметить, что значения для изображений с номерами 2-4 не поменялись, а для остальных значений поменялись, так для изображений с номерами 5 и 6 значение индекса изменилось с 0,81 до 0,78 и с 0,82 до 0,78, что является небольшим колебанием значения, для изображений с номерами 7-10 колебание значение уже больше, значение для 7 изображения поменялось с 0,85 до 0,80, для 8 с 0,20 до 0,26, для 9 с 0,19 до 0,30, для 10 с 0,19 до 0,33.

Табл. 1. Сравнение изображений на основе нейросети VGG16 без закраски текста

Номер изображения	1 (Оригинал)	2	3	4	5	6	7	8	9	10
Значение индекса	1	0,73	0,77	0,77	0,81	0,82	0,85	0,20	0,19	0,19
Изображения похожи	да	да	да	да	да	да	да	нет	нет	нет

Табл. 2. Сравнение изображений на основе нейросети VGG16 с закраской текста

Номер изображения	1 (Оригинал)	2	3	4	5	6	7	8	9	10
Значение индекса	1	0,73	0,77	0,77	0,78	0,78	0,80	0,26	0,30	0,33
Изображения похожи	да	да	да	да	да	да	да	нет	нет	нет

Теперь определим пороги схожести. Для этого используем среднее из наиболее схожих изображений. В случае с *VGG16* возьмем среднее значение изображений 3-7, порог будет равен 0,80. Таблицы с результатами сравнения изображений другими методами и определенными значениями порогов схожести приведены в работе [18].

Заключение

Разработан веб-сервис, позволяющий автоматизировать проверку учебных заданий по *html*-верстке. Код веб-сервиса размещен на платформе *GitHub* [25].

С использованием веб-сервиса проведено тестирование различных методов сравнения изображений на основе реальных макетов и примеров верстки. Определены пороги схожести изображений. Метод на основе нейросети *VGG16* оказался самым показательным и эффективным методом сравнения изображений. Метод среднеквадратичной ошибки и метод нахождения индекса среднего структурного сходства не столь эффективны, они позволяют заметить разницу в изображениях, если те значительно отличаются друг от друга.

Возможным направлением для дальнейшего развития является интеграции веб-сервиса с образовательными платформами и системами для обеспечения более удобного использования.

Список источников

1. Университет «Дубна» : Прикладная информатика в компьютерном дизайне. – URL: https://new2.uni-dubna.ru/directions/ae6b4579-ee8c-477e-b13a-783c33bc9fae (Дата обращения: 02.08.2023).
2. Reliable Screenshot API | Screenshot Machine. – Devtica s.r.o., 2012 - 2023. – URL: https://www.screenshotmachine.com/ (Дата обращения: 02.08.2023).
3. Screenshot API / Competitor Archive, LLC. – URL: https://www.screenshotapi.io/ (Дата обращения: 02.08.2023).
4. Using the Screen Capture API - Web APIs | MDN // MDM Web Docs. – URL: https://developer.mozilla.org/en-US/docs/Web/API/Screen_Capture_API/Using_Screen_Capture (Дата обращения: 02.08.2023).
5. html2canvas - Screenshots with JavaScript / Niklas von Hertzen. – URL: https://html2canvas.hertzen.com/ (Дата обращения: 02.08.2023).
6. Chrome Headless: How to take full page page screenshot // Open source code library - OneLinerHub. – URL: https://onelinerhub.com/chrome-headless/how-to-take-full-page-page-screenshot (Дата обращения: 02.08.2023).
7. HtmlUnit // GitHub : [web platform]. – GitHub, Inc., 2023. – URL: https://github.com/HtmlUnit (Дата обращения: 24.02.2023).
8. PhantomJS - Scriptable Headless Browser. – PhantomJS contributors, 2010-2018. – URL: https://phantomjs.org (Дата обращения: 24.02.2023).
9. История версий Firefox - Firefox version history. – URL: https://ru.abcdef.wiki/wiki/Firefox_version_history (Дата обращения: 02.03.2023).

10. Firefox 55 — Mozilla — Новости. – URL: https://www.linux.org.ru/news/mozilla/13599508/page2 (Дата обращения: 03.03.2023).
11. Truong, T. Headless Chrome - What is it? – Дата публикации: 25.09.2021. – Дата обновления: 7.10. 2021. – URL: https://dev.to/thanhtr99270163/today-i-learned-headless-chrome-what-is-it-1dd.
12. MaxRokatansky. Headless тестирование в браузере. Плюсы и минусы // Хабр :[сайт]. – Habr, 2006–2023. – Дата публикации: 18.03.2019. – URL: https://habr.com/ru/companies/otus/articles/444248.
13. Statcounter Global Stats - Browser, OS, Search Engine including Mobile Usage Share. – StatCounter, 1999-2023. – URL: https://gs.statcounter.com/ (Дата обращения: 02.08.2023).
14. Кацапа Л. Среднеквадратическая ошибка (MSE). – Дата публикации: 03.04.2021. – URL: https://www.helenkapatsa.ru/sriedniekvadratichieskaia-oshibka.
15. Структурное сходство - Structural similarity. – URL: https://ru.abcdef.wiki/wiki/Structural_similarity (Дата обращения: 01.04.2023).
16. Милютин И.VGG16 — нейросеть для выделения признаков изображений // Neurohive – Нейронные сети : [сайт]. – Дата публикации: 23.11.2018. – URL: https://neurohive.io/ru/vidy-nejrosetej/vgg16-model.
17. Вектор средних и матрица ковариации. – URL: https://ab.al-shell.ru/articles/vektor-srednih-i-matritsa-kovariatsii (Дата обращения: 02.08.2023).
18. Рыженков А. Д. Веб-сервис для автоматизированной проверки корректности решения учебных заданий по верстке : Магистерская диссертация. — Дубна, 2023.
19. GitHub - JaidedAI/EasyOCR: Ready-to-use OCR with 80+ supported languages and all popular writing scripts including Latin, Chinese, Arabic, Devanagari, Cyrillic and etc. // GitHub : [web platform]. – GitHub, Inc., 2023. – URL: https://github.com/JaidedAI/EasyOCR (Дата обращения: 18.04.2023).
20. GitHub - opencv/opencv: Open Source Computer Vision Library // GitHub : [web platform]. – GitHub, Inc., 2023. – URL: https://github.com/opencv/opencv (Дата обращения: 18.04.2023).
21. Ермолаев, И. Алгоритм быстрого нахождения похожих изображений // Хабр :[сайт]. – Habr, 2006–2023. – Дата публикации: 22.06.2011. – URL: https://habr.com/ru/articles/122372.
22. Как измерить сходство между двумя изображениями? – URL: https://utyatnishna.ru/info/42419/how-can-i-measure-the-similarity-between-two-images (Дата обращения: 02.08.2023).
23. Сравнение сходства изображений и обнаружение конкретных объектов на изображениях // Русские блоги : [сайт]. – russianblogs.com, 2020-2023. – URL: https://russianblogs.com/article/71441215538 (Дата обращения: 02.08.2023).
24. NewTechAudit. ML-подходы по поиску похожих изображений / NewTechAudit // Хабр :[сайт]. – Habr, 2006–2023. – Дата публикации: 31.03.2023. – URL: https://habr.com/ru/articles/726122.
25. AutomaticLayoutCheck : [Веб-сервис по автоматизированной проверке учебных заданий по верстке] // GitHub : [web platform]. – GitHub, Inc., 2023. – URL: https://github.com/Jaldsky/AutomaticLayoutCheck (Дата обращения: 22.05.2023).