DOI 10.37882/2223-2966.2023.7-2.25

РАЗРАБОТКА СИСТЕМЫ ОПТИЧЕСКОГО РАСПОЗНАВАНИЯ СИМВОЛОВ НА ИЗОБРАЖЕНИЯХ ДЛЯ АВТОМАТИЗАЦИИ ПРОВЕРОК БЕЗОПАСНОСТИ

DEVELOPMENT OF AN OPTICAL CHARACTER RECOGNITION SYSTEM FOR THE AUTOMATION OF SECURITY CHECKS

A. Rusakov
V. Filatov
S. Dolzhenkov
E. Antonov
D. Andreev
D. Kulikov

Summary: This article deals with different approaches for optical character recognition (OCR) systems on images in order to automate security checks. A review and analysis of current methods and algorithms for character recognition in streaming images is given. The basic technology for optical information processing is considered, and the scope of this technology in the field of information security is determined. The features of CAPTCHA solution automation implementation are shown. OpenCV and Tesseract are considered to be the main libraries. The article develops CAPTCHA analysis software architecture in order to detect the characters depicted on it. The main points in realization of this software solution are given and it is shown that it is possible to apply this software solution in practice thus showing vulnerabilities of CAPTCHA security checking systems.

Keywords: image character recognition, image mining, CAPTCHA recognition, computer vision.

Русаков Алексей Михайлович
старший преподаватель, МИРЭА — Российский технологический университет
rusal@bk.ru
Филатов Вячеслав Валерьевич
доцент, МИРЭА — Российский технологический университет
filv@mail.ru
Долженков Сергей Сергеевич
ассистент, МИРЭА — Российский технологический университет
dolzhenkov@mirea.ru
Антонов Эмиль Петрович
МИРЭА — Российский технологический университет
emillink69@gmail.com
Андреев Дмитрий Игоревич
МИРЭА — Российский технологический университет
ufi-69@mail.ru
Куликов Даниил Михайлович
МИРЭА — Российский технологический университет
daniil.kulikov.00@mail.ru

Аннотация: в статье рассматриваются различные подходы для систем оптического распознавания символов (OCR) на изображении с целью автоматизации проверок безопасности. Приводится обзор и анализ современных методов и алгоритмов распознавания символов в потоковых изображениях. Рассмотрены основные технологии для обработки оптической информации, а также определена область применения данной технологий в сфере информационной безопасности. Показаны особенности реализации автоматизации решений CAPTCHA. В качестве основных библиотек рассмотрены: OpenCV и Tesseract. В статье разработана архитектура программного обеспечения для анализа CAPTCHA с целью определения тех символов, которые на ней изображены. Приводятся основные моменты в реализации данного программного решения, показано, что данное программное решение возможно применить на практике, тем самым, показывая уязвимости систем проверки безопасности CAPTCHA.

Ключевые слова: распознавание символов на изображении, интеллектуальный анализ изображений, распознавание CAPTCHA, компьютерное зрение.

Введение

Для обеспечения безопасности и надежности онлайн-сервисов от кибер-атак были разработаны многочисленные проверки безопасности, и одним из таких решений является CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart). CAPTCHA представляет собой задачу, которая предназначена для того, чтобы определить, является ли пользователь человеком или компьютерной программой [1].

Актуальность данной темы заключается в том, что растущее число проверок безопасности в онлайн-сервисах и приложениях может доставлять неудобства пользователям, делая процесс неприятным и отнимающим много времени. Это может привести к снижению вовлеченности пользователей и негативным отзывам. Также, стоить добавить, что классические решения CAPTCHA становятся все менее эффективными, и пока разработчики пытаются их улучшить, злоумышленники находят новые способы их обойти. Однако, во многих онлайн-сервисах CAPTCHA не обновляется, из-за чего системы,

использующие технологию оптического распознавания символов, могут легко их решать. Целью данной работы является демонстрация того, как технология оптического распознавания символов позволяет обходить проверки безопасности. Для этого выполняется разработка системы для автоматизации таких проверок безопасности, как CAPTCHA.

CAPTCHA и ее роль в обеспечении информационной безопасности

CAPTCHA это метод проверки безопасности, используемый для определения того, является ли пользователь человеком или нет. Эта проверка предназначена для предотвращения несанкционированного доступа к вебсайту или выполнения определенных действий на нем, таких как создание нескольких учетных записей или рассылка спама. Существует несколько типов CAPTCHA, каждая из которых предназначена для противодействия различным типам атак ботов [2]. Наиболее распространенные типы включают:

— CAPTCHA на основе текста
— CAPTCHA на основе изображения
— CAPTCHA на основе звука
— CAPTCHA на основе видео
— CAPTCHA, основанные на головоломке

Основная роль CAPTCHA в обеспечении безопасности заключается в предотвращении автоматизированных атак со стороны ботов. Боты могут быть запрограммированы на повторную отправку запросов на веб-серверы или веб-приложения, что может вызвать атаку типа «отказ в обслуживании» (DoS), скомпрометировать учетные записи пользователей или извлечь конфиденциальную информацию. CAPTCHA может предотвратить эти атаки, заставляя пользователей решать задачу, с которой ботам, в большинстве случаев, не в состоянии справиться.

Методы и алгоритмы распознавания символов

Методы и алгоритмы распознавания символов относятся к методам и математическим моделям, используемым для распознавания и идентификации символов на входном изображении. Эти методы включают предварительную обработку, извлечение признаков (сегментацию) и классификацию.

Предварительная обработка является важным шагом в распознавании символов, поскольку она помогает повысить качество и четкость входного изображения, что может значительно повлиять на точность алгоритмов распознавания. Методы предварительной обработки включают удаление шума и нормализацию.

Сегментация — процесс разделения изображения на множество областей или сегментов, где каждый сегмент соответствует значимому объекту или части изображения. Сегментация является важным шагом в оптическом распознавании символов, поскольку она выделяет отдельные символы или слова из остальной части изображения. Существуют различные методы сегментации изображений, которые включают пороговое значение, обнаружение краев, кластеризацию, увеличение области и морфологические операции.

Классификация — заключительный этап процесса распознавания символов, на котором извлеченные признаки классифицируются по определенным категориям. Процесс классификации обычно включает в себя обучение классификатора с использованием набора помеченных данных, где классификатор учится различать различные классы символов на основе их характеристик. Наиболее часто используемыми классификаторами при распознавании символов являются искусственные нейронные сети, машины опорных векторов, классификаторы *k*-ближайших соседей и деревья решений.

Предварительная обработка входной информации

Для изменения размера изображения CAPTCHA, был выбран алгоритм билинейной интерполяции (bilinear interpolation) [3]. Билинейная интерполяция — это широко используемый метод повторной выборки изображений.

Формула для билинейной интерполяции может быть выражена следующим образом:

Пусть (x, y) представляют исходные координаты пикселя на входном изображении, а (x', y') обозначают новые координаты после изменения размера. Значение пикселя в точке (x', y') может быть вычислено по следующей формуле:

$$F(x', y') = (1-\alpha)(1-\beta)F(x_1, y_1) + \alpha(1-\beta)F(x_2, y_1) + \beta(1-\alpha)F(x_1, y_2) + \alpha\beta F(x_2, y_2),$$

Где $F(x, y)$ — представляет значение пикселя в координатах (x, y) на исходном изображении;

(x_1, y_1), (x_2, y_1), (x_1, y_2), и (x_2, y_2) — обозначают соседние пиксели, окружающие (x', y') на исходном изображении;

α и β — коэффициенты интерполяции, которые зависят от относительного положения (x', y') относительно соседних пикселей.

Чтобы изменить размер изображения в 2 раза, мы определяем новые координаты (x', y') путем умножения исходных координат (x, y) на 2. Затем мы вычисляем коэффициенты интерполяции α и β на основе дробных частей (x', y').

Следующим этапом предварительной обработки является преобразование цветового пространства изображения CAPTCHA в оттенки серого. Чтобы преобразовать изображение RGB в оттенки серого, воспользуемся методом средневзвешенного значения. Формула для преобразования оттенков серого выглядит следующим образом:

$$Y = 0{,}299 * R + 0{,}587 * G + 0{,}144 * B$$

Здесь Y представляет значение интенсивности в оттенках серого, а R, G и B обозначают значения красного, зеленого и синего цветовых каналов соответственно. Коэффициенты 0,299, 0,587 и 0,144 представляют относительную важность каждого цветового канала при определении общей интенсивности.

Заключительным этапом предварительной обработки изображения перед сегментацией является применение к изображению двустороннего фильтра (bilateral filter). Двусторонний фильтр — это нелинейный сглаживающий фильтр с сохранением краев, который уменьшает шум при сохранении важных краев и деталей. Этот шаг помогает устранить высокочастотный шум, такой как мелкие крапинки или артефакты, которые могут мешать последующим процессам обработки.

Математическая формула для двустороннего фильтра выражается следующим образом:

$$I_{filtered}(x,y) = \frac{1}{W(x,y)} * \sum_{(x',y')}^{\Omega(x,y)} (I(x',y') * S\left(\left\|(x,y) - (x',y')\right\|\right) * R(|\, I(x,y) - I(x',y')\,|))$$

Где $I_{filtered}(x, y)$ — отфильтрованное значение интенсивности в пикселе (x, y) изображения после применения двустороннего фильтра;

$\Omega(x, y)$ — обозначает набор всех соседних пикселей внутри окна (window) с центром в точке (x, y);

$I(x', y')$ — значение интенсивности соседнего пикселя (x', y') на изображении;

$S\left(\left\|(x,y) - (x',y')\right\|\right)$ — пространственный вес между пикселями (x, y) и (x', y'), который измеряет пространственное расстояние между ними;

$R(|\, I(x,y) - I(x',y')\,|)$ — вес диапазона, основанный на разнице интенсивности между пикселями (x, y) и (x', y');

W (x, y) — коэффициент нормализации, который гарантирует, что сумма весов равна 1.

Сегментация изображения

После предварительной обработки изображения CAPTCHA, необходимо применить сегментацию изображения с целью извлечения символов для последующей классификации. Выбранным подходом для сегментации в данном случае является алгоритм определения порогового значения (thresholding algorithm) [4].

Данный процесс может быть выражен с помощью следующей формулы:

$$Binary\ Image(x,y) = \begin{cases} 1, if\ Grayscale\ Image(x,y) \geq T \\ 0, if\ Grayscale\ Image(x,y) < T \end{cases}$$

В этой формуле Binary Image (x, y) представляет значение пикселя в бинарном изображении в местоположении (x, y), а Grayscale Image (x, y) представляет значение интенсивности в изображении в оттенках серого в том же местоположении.

Классификация символов на изображении

Один из механизмов OCR — Tesseract, использует статистическую модель, известную как скрытая марковская модель (HMM) [5]. HMM представляет последовательность символов в виде вероятностной модели, где каждый символ считается скрытым состоянием, а наблюдаемые особенности изображения являются информацией, связанной с каждым состоянием.

Обозначим наблюдаемые особенности изображения как $X = \{x_1, x_2, \ldots, x_n\}$, где n — количество символов в сегментированном изображении. Аналогично, обозначим скрытые состояния (символы) как $Y = \{y_1, y_2, \ldots, y_n\}$. Цель состоит в том, чтобы найти наиболее вероятную последовательность скрытых состояний с учетом наблюдаемых особенностей. Используя теорему Байеса, это можно выразить следующим образом:

$$P(Y|X) = \frac{P(X|Y) * P(Y)}{P(X)},$$

Где член правдоподобия (likelihood) P(X|Y) представляет собой вероятность наблюдаемых признаков с учетом скрытых состояний, которые могут быть смоделированы с помощью комбинации методов извлечения признаков изображения и обученных алгоритмов машинного обучения;

предшествующий член (prior) P(Y) представляет собой предшествующую вероятность последовательности символов, которая может быть оценена на основе частотного распределения символов в обучающем наборе данных;

член доказательства (marginalization) P(X) действует как фактор нормализации и может быть вычислен путем

маргинализации по всем возможным последовательностям скрытых состояний:

$$P(X) = \sum_{i=1}^{n} \left(P(X|Y_i) * P(Y_i) \right).$$

Чтобы найти наиболее вероятную последовательность скрытых состояний, механизм OCR Tesseract использует алгоритм Витерби, алгоритм динамического программирования, который эффективно вычисляет максимальную апостериорную оценку максимума (MAP) скрытых состояний с учетом наблюдаемых характеристик. Алгоритм Витерби рекурсивно вычисляет максимальную вероятность каждого состояния в каждой позиции последовательности, принимая во внимание вероятности предыдущих состояний и вероятности перехода между состояниями.

Стек технологий

При разработке серверной части приложения используется следующий стек технологий:

— Язык программирования Golang
— Библиотека компьютерного зрения OpenCV
— Механизм оптического распознавания символов Tesseract

Для пользовательской части:

— Язык программирования JavaScript (фреймворк React)
— Инструмент для настройки среды разработки Vite

Разработка самого приложения производится на операционной системе Linux с использованием текстового редактора VSCode.

Архитектура программного обеспечения

Архитектура программного обеспечения веб-приложения соответствует модели клиент-сервер, при этом серверная часть отвечает за обработку пользовательских запросов и выполнение основных функциональных возможностей системы, в то время как пользовательская часть отвечает за пользовательский интерфейс и взаимодействие с серверной частью.

Серверная часть выступает в качестве основы приложения, реализуя генерацию изображений CAPTCHA, решение CAPTCHA, предварительную обработку изображений, сегментацию изображений, классификацию символов и анализ результатов распознавания.

Пользовательская часть фокусируется на получении изображений с CAPTCHA от пользователей и отправке обратно текстовых ответов на распознанные изображения.

Для облегчения связи между серверной и пользовательской частями реализован API, обеспечивающий беспрепятственный обмен данными и взаимодействие. API обрабатывает передачу изображений с CAPTCHA, а также отправку результатов распознавания обратно в пользовательский интерфейс.

Данная архитектура обеспечивает четкое разделение задач: серверная часть берет на себя вычислительные задачи и обработку данных, в то время как пользовательская часть занимается взаимодействием с пользователем. Такой модульный подход обеспечивает более простое обслуживание, масштабируемость и будущие усовершенствования [6].

Алгоритм работы системы

В качестве примера возьмем исходное изображение CAPTCHA, представленное на рисунке 1.

Рис. 1. Исходное изображение CAPTCHA

Сначала применяется формула билинейной интерполяции. Получаем новое изображение CAPTCHA, представленное на рисунке 2.

Рис. 2. Полученное изображение CAPTCHA после увеличения в 2 раза

Следующим шагом является преобразование цветового пространства изображения в оттенки серого. Результат преобразования, представленное на рисунке 3.

Рис. 3. Полученное изображение CAPTCHA после преобразования в оттенки серого

Применим двусторонний фильтр к предыдущему этапу и получим новое изображение, представленное на рисунке 4.

Рис. 4. Полученное изображение CAPTCHA после применения двустороннего фильтра

Далее, происходит сегментация изображения путем использования алгоритма определения порогового значения. Результат работы данного алгоритма представлен на рисунке 5.

Рис. 5. Полученное изображение CAPTCHA после сегментации

В результате мы получили полностью обработанное изображение CAPTCHA, с выделенными символами, готовыми для дальнейшей классификации.

Заключительным этапом работы системы является классификация полученных символов на изображении CAPTCHA. Результат работы представлен на рисунке 6.

Анализ результатов работы системы

Необходимо проанализировать результаты системы, а именно рассчитать среднюю точность распознавания символов на изображениях CAPTCHA. В качестве примера сгенерируем 10 изображений CAPTCHA и вычислим среднюю точность распознавания символов. На рисунке 7 представлен результат со средней точностью распознавания 86,5 %.

Заключение

Таким образом, была разработана система для автоматизации решений CAPTCHA, способствующая решению проблемы многих онлайн-ресурсов, а именно необходимости обновления методов проверки безопасности, во избежание нарушения целостности своих данных.

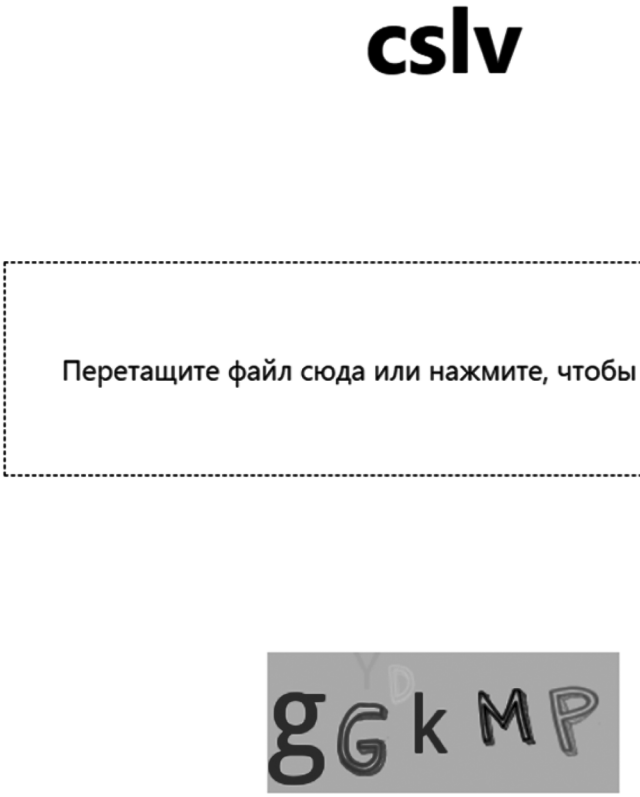

Рис. 6. Результат работы системы

```
{"accuracy":86.50001,"analysis":
[{"id":"SmDxbKJk6B1gnSkDsrkd","answer":"cnek","solution":"cne","rate":75},
{"id":"bscrsXEnSCSxdOsQr4U1","answer":"3ta8","solution":"3ta8","rate":100},
{"id":"K8ron9cmONxcC4yeHvQB","answer":"h326","solution":"h36","rate":75},
{"id":"hnG0jsuzB4oKr9MJdRo3","answer":"mp84o","solution":"mp40o","rate":80},
{"id":"mub2x3oOM6AJ1AMUYH1z","answer":"69xrl","solution":"69xrl","rate":100},
{"id":"fd02TDgR1qaxHZ1Q84GE","answer":"i0m6","solution":"i0mb","rate":75},
{"id":"3RXmgUBCXU0UeEuXEcBV","answer":"mryj","solution":"mryj","rate":100},
{"id":"jrqGU8Bd8EusgFvYRmbb","answer":"v3uv","solution":"v3uv","rate":100},
{"id":"4N69vBDtdaq3Rh8O9xMk","answer":"pj3gq","solution":"pijigq","rate":80},
{"id":"E2r2nvjVoRDxoAj71cb1","answer":"wau25","solution":"way25","rate":80}]}
```


Рис. 7. Анализ результатов распознавания символов CAPTCHA

ЛИТЕРАТУРА

1. von Ahn, L., Blum, M., Hopper, N.J., Langford, J. (2003). CAPTCHA: Using Hard AI Problems for Security. In: Biham, E. (eds) Advances in Cryptology — EUROCRYPT 2003. Lecture Notes in Computer Science, vol 2656. Springer, Berlin, Heidelberg. 10.1007/3-540-39200-9_18
2. Ved Prakash Singh, Preet Pal (2014). Survey of Different Types of CAPTCHA [Электронный ресурс] / Academia. — Режим доступа: https://www.academia.edu/7053243/Survey_of_Different_Types_of_CAPTCHA_international. [Дата обращения 13.03.2023].
3. Parsania, Pankaj & V.Virparia, Dr. (2015). A Review: Image Interpolation Techniques for Image Scaling. International Journal of Innovative Research in Computer and Communication Engineering. 02. 7409–7414. 10.15680/IJIRCCE.2014.0212024.
4. Chaubey, A.K. (2016). Comparison of The Local and Global Thresholding Methods in Image Segmentation [Электронный ресурс] / Semantic Scholar. — Режим доступа: https://www.semanticscholar.org/paper/Comparison-of-The-Local-and-Global-Thresholding-in-Chaubey/8a12c0695eec379f5da8ab28d2277eb243d99f65. [Дата обращения 08.03.2023].
5. R. Smith, (2007). An Overview of the Tesseract OCR Engine, Ninth International Conference on Document Analysis and Recognition, pp. 629–633, 10.1109/ICDAR.2007.4376991.
6. Sharma, Anubha & Kumar, Manoj & Agarwal, Sonali. (2015). A Complete Survey on Software Architectural Styles and Patterns. Procedia Computer Science. 70. 16–28. 10.1016/j.procs.2015.10.019.
7. Антонов Э.П. Исходный код системы оптического распознавания символов на изображении для автоматизации решений CAPTCHA. https://github.com/unemil/cslv. (2023).

Журнал «Современная наука: актуальные проблемы теории и практики»