# МЕТОД АВТОМАТИЧЕСКОГО РАСПОЗНАВАНИЯ СИМВОЛЬНО-ГРАФИЧЕСКОГО ТЕСТА *CAPTCHA*

*Д.А. Валяева, Д.С. Салита*
Алтайский государственный университет, г. Барнаул

Все большее применение находит технология двухфакторной аутентификации. В основном эта технология широко используется в сети Интернет, эту технологию используют банки для аутентификации в своих системах. Компании, которые заботятся о сохранности персональных данных, начинают использовать технологию двухфакторной аутентификации для повышения безопасности своих систем аутентификации [1, 2]. Первым фактором, как правило, используется логин/пароль, вторым фактором обычно используется SMS – оповещение с кодом доступа, на мобильный телефон владельца персональных данных. Но SMS – оповещения можно заменить на более старую технологию RFID, где вторым фактором является RFID метка. Поэтому целью данной работы является: Разработка программно-аппаратного комплекса двухфакторной аутентификации с использованием RFID – технологии.

Для защиты от автоматической регистрации и рассылки спам сообщений более чем на пяти миллионах Интернет-ресурсах применяется тест CAPTCHA ("Completely Automatic Public Turing Test to Tell Computers and Humans Apart"), который представляет собой полностью автоматизированный публичный тест Тьюринга для различения компьютеров и людей [3, 4]. Согласно статистике, среди прочих лидирует вид теста, в котором пользователю предлагается распознать символы, изображённые графически с различными искажениями и шумами. В связи с этим актуальной является задача разработки методов автоматического распознавания символьно-графического CAPTCHA с целью анализа эффективности данного способа защиты.

В общем виде, распознавание текста состоит из следующих процедур и методов:

– Процедура предварительной обработки используется практически всегда после получения информации, и представляет собой применение операций усреднения и выравнивания гистограмм, различного типа фильтров для исключения помех, а также подавления внешних шумов.

– Сегментация – процесс разделения изображения на отдельные символы.

– Конечный этап обработки - распознавание. Для этого этапа входными данными являются изображения, полученные в результате шумоподавления и процесса сегментации.

Для того, чтобы подробнее показать разработанный нами метод, будем использовать один пример капча-изображения (рис. 1). Анализируя рис. 1, можно выделить следующие недостатки донного теста для распознавания: поворот символов, фоновый шум в виде символов и символы иногда касаются друг друга. В противовес этому фон значительное светлее символов и символы не искажены. Размер в среднем составляет 120x30 пикселей, но занимаемая площадь символами на изображении меньше и находится по центру. Первым этапом является приведение исходного цветного изображение к бинарному виду и удаление шумов.

Рис. 1. Пример изображения символьно-графического теста CAPTCHA

Цвет для текста генерируется в диапазоне rand(0, 200), 0, rand(0, 200), для R G B соответственно, достаточно выделить цвета только в этом диапазоне, при этом фон с большим количеством разных цветов не сможет повлиять на статистику самого часто используемого цвета.

Теперь на основе этих фактов анализируем цвет каждого пикселя во всем изображении и выделяем самый часто-используемый. Задаем от него небольшую погрешность, выделяем этот цвет и немного похожие на него с учётом погрешности. Инвертируем изображение в многомерный массив состоящий из нулей – белый цвет, и единиц – чёрный. Теперь, когда выделили только символы на изображении, можно обрезать элементы массива, не содержащие единиц, и мы получим максимально обрезанный участок монохромного изображения с текстом (рис. 2).

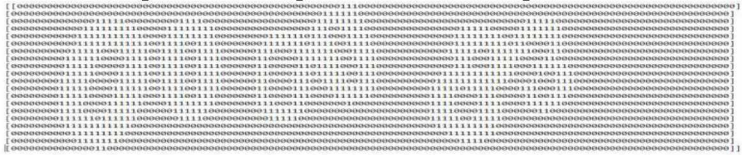

Рис. 2. Пример участка многомерного массива, содержащего символы

Проанализируем полученный многомерный массив (рис. 2). Можно заметить, что между каждым отдельно стоящим символом или парой склеенных символов проходят чёткие линии, не содержащие единиц, это значит, что мы можем перевести строки нашего массива в столбцы. И затем, записать каждый столбец содержащий единицу в отдельный массив, тем самым вырезав наши символы по боковым границам (рис. 3).

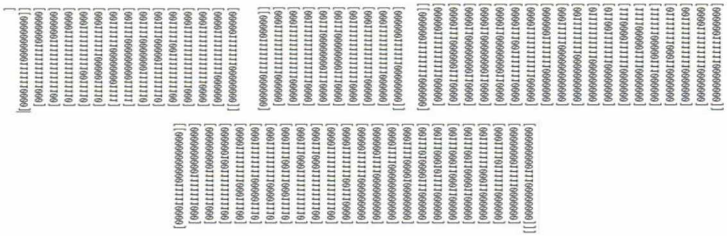

Рис. 3. Пример многомерного массива символов.

Теперь получили четыре многомерных массива, два из которых уже представляют собой отдельные символы, и два массива, которые необходимо разделить ещё на четыре фрагмента, так как данный тест CAPTCHA предполагает шестизначный ответ. Рассмотрим фрагменты, содержащие склеенные символы. Сразу бросается в глаза три точки минимума, соответствующие границам символов. Для того что бы программно разрезать массив в данных точках, нужно найти ключ каждой первой встретившийся единицы в столбце и выбрать точку с максимальным ключом в середине данного массива. По данной точке и будет произведено разделение массива. В результате получим массив, состоящий из двух многомерных элементов, которые необходимо записать в правильном порядке, что бы алгоритм распознавания получал символы последовательно. Каждый символ будет объединен в отдельный массив, и столбцы переведены обратно в строки. Так как, изначально количество строк массива зависело от общей площади, занимаемой символами, теперь становится возможным избавиться от ненужного пространства для каждого из символов (рис. 4).

Таким образом подготовили многомерный массив из отдельно стоящих символов, каждый из которых вырезан строго по своему контуру. Даже если символы частично наложить друг на друга (в разумных пределах), всё равно будут видны точки минимумов, хоть и не так явно, что дает возможность распознавать изображения даже с наложениями символов. Прежде чем передать полученный массив

алгоритму распознавания, необходимо подготовить набор символов, на основе которого будет работать данный алгоритм. В нашем случае мы имеем более сотни примеров монохромных масок на каждый символ.

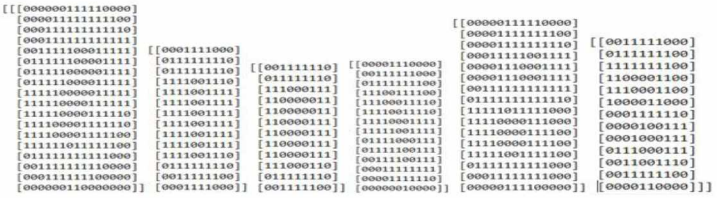

Рис. 4. Пример многомерного массива символов

Так как в выбранном символьно-графическом тесте Тьюринга символы имеют разный размер, то соответственно массив для каждого символа будет содержать разное количество значений. Поэтому необходимо привести все маски к общему виду и дополнить каждый массив символа пустыми строками. PHP-ML — полноценная библиотека для работы с искусственным интеллектом. Для распознавания символов был использован классификатор данной библиотеки Support Vector Classification (SVC).

Необходимый набор данных, который нужно подавать в данный классификатор, представляет из себя многомерный массив, содержащий все маски обучающего набора символов, и массив, содержащий ответы на каждую из этих масок соответственно.

Теперь можно передать массив из примера (рис. 4) на распознавание.

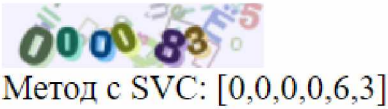


Рис. 5. Результат распознавания примера символьно-графического CAPTCHA

Как видно на рис. 5, полученный результат является ошибочным, алгоритм неверно распознал один символ. К сожалению, мы никак не сможем повлиять на алгоритм работы классификатора SVC для улучшения распознавания, но можно попробовать применить метод не основанный на машинном обучении, чтобы сравнить результаты. Так как есть обучающий

многомерный массив символов и соответственный массив ответов, можем сравнивать каждый массив символа с уже имеющимся набором, и считать пересечение с каждым из них, т.е. количество совпадений нулей и единиц. После подсчёта совпадений получим некоторый набор чисел для шести символов:

1. [227,251,250,239,216,…..,249,286,252,236,230];
2. [198,310,307,312,319,…..,216,215,305,285,335];
3. [200,280,287,296,313,…..,188,203,285,283,301];
4. [207,331,312,315,332,…..,209,218,312,292,318];
5. [216,264,273,266,255,…..,228,247,273,281,259];
6. [197,291,284,295,330,…..,197,206,282,274,332].

После этого необходимо найти максимальное количество совпадений для каждого из распознаваемых символов и выбрать ключ этого числа. За тем можно будет определить ответ из массива по данному ключу, так как набор массива ответов соответствует набору символов обучающего массива (рис. 6):

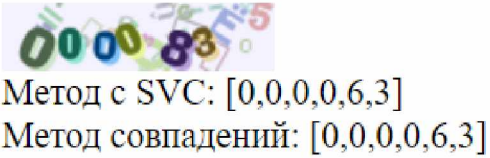


Рис. 6. Результат распознавания примера символьно-графического CAPTCHA

При сравнении полученных результатов видно, что два данных метода распознали символы одинаково, попробуем сравнить результаты работы данных алгоритмов на большем количестве примеров. Анализ будет произведен на тестовом наборе изображений, состоящем из трёхсот картинок символьно-графического теста Тьюринга.

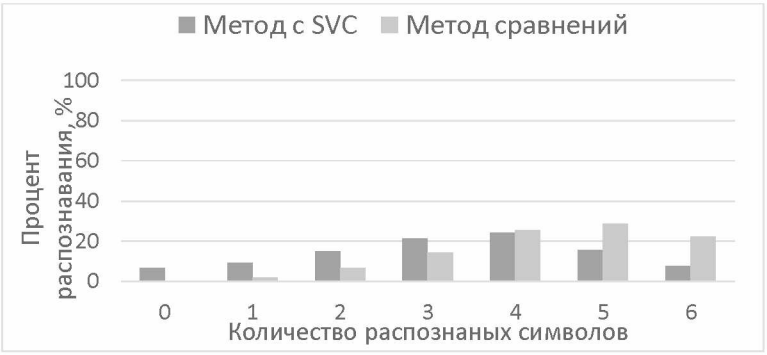


Рис. 7. Гистограмма для количества распознанных символов каждого изображения

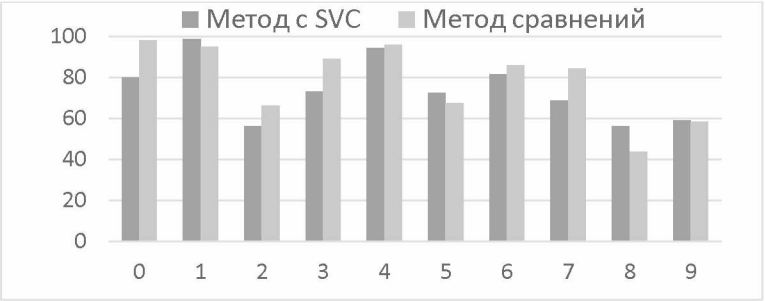


Рис. 8. Гистограмма для р аспознанных символов каждого изображения

При рассмотрении полученных результатов (рис. 7 и 8) можно сделать вывод о том, что разработка методов автоматического прохождения символьно-графического теста Тьюринга не является невыполнимой задачей, и в целом дают неплохой результат, из чего следует, что данный вид защиты не является эффективным на сегодняшний день.

**Библиографический список**

1. Биркун Н.И., Тураров Ж.Ж., Зозуля А.О. Разработка методики оценки уязвимости системы аутентификации на основе технологии CAPTCHA // Труды Международной научно-технической конференции «Перспективные информационные технологии (ПИТ 2014)», 2014. – С. 191-195.

2. Малинин П.В., Поляков В.В. Иерархический подход в задаче идентификации личности по голосу с помощью проекционных методов классификации многомерных данных // Доклады Томского государственного университета систем управления и радиоэлектроники. 2010. № 1-1 (21). С. 128-130.
3. Гуськова А.М., Басараб М.А. Исследование эффективности применения CAPTCHA как средства защиты сайтов // Современные тенденции развития науки и технологий. – 2015. – № 5. – С. 12–17.
4. Попов А.А., Козлов А.Е. Исследование надежности полностью автоматического теста Тьюринга для различения компьютеров и людей (CAPTCHA) для предотвращения несанкционированного ввода информации в интернете // Известия Российского экономического университета им. Г.В. Плеханова. – 2012. – № 3. – С. 80–98.