# Методы предварительной обработки скриншотов десктопных приложений для системы оптического распознавания символов

С.А. Рудь
*Самарский национальный исследовательский университет им. академика С.П. Королева*
Самара, Россия
jin_.96@mail.ru

А.А. Рудь
*Самарский национальный исследовательский университет им. академика С.П. Королева*
Самара, Россия
sasha_.96@mail.ru

М.М. Шушкина
*Поволжский государственный университет телекоммуникаций и информатики*
Самара, Россия
marija.shushkina@yandex.ru

***Аннотация*—В работе рассматриваются типовые методы предварительной обработки изображений перед подачей их в системы машинного зрения для распознавания текста. Решается задача предобработки данных для систем оптического распознавания символов на примере десктопных приложений.**

***Ключевые слова— элементы управления приложения, распознавание текста, оптическое распознавание символов, десктопное приложение, виджеты, YOLO, Tesseract, PaddleOCR, CRAFT.***

## 1. Введение

Задача оптического распознавания символов решается во многих прикладных областях: анализ и распознавание отсканированных документов, распознавание медицинских карт, внедрение электронного документооборота, сбор информации.

Наличие шума, использование различных шрифтов или рукописный текст на изображениях усложняют задачу распознавания текста, что приводит к необходимости разработки неких последовательно применяющихся методов предварительной обработки изображений с целью улучшения их качества.

Вклад авторов в данной работе: собран и размечен набор данных для решения поставленной задачи, решена проблема качественной локализации текстовых блоков.

## 2. Стандартные методы предобработки изображения

В статьях [1-3] рассматриваются такие методы предварительной обработки изображений для улучшения качества локализации и распознавания текста как: пороговая обработка, операции математической морфологии, удаление шума, увеличение разрешения изображения с помощью свёрточных нейронных сетей.

Применение методов предварительной обработки изображений зависит от области задачи, качества изображения, ориентации текста. Изображения, полученные путем снятия экрана десктоп-приложений, отличаются низким разрешением (100-150 dpi), обширной цветовой палитрой (изменяется как задний фон, так и цвет текста), разнообразием интерфейсов, шрифтов и наличием символов, не входящих в заданный алфавит. Облегчает задачу постоянная горизонтальная ориентация текста.

Некоторые элементы управления содержат иконки, схожие с алфавитом распознаваемого языка, которые трудно удалить. Компоненты интерфейса могут находиться в разных состояниях, например, в фокусе, или недоступен для взаимодействия. В последнем случае оттенок текста кнопок сильно тускнеет, ухудшается контрастность. Существуют и иные случаи, когда часть текста выходит за рамки приведения в нужный вид одной лишь пороговой обработкой. В некоторых случаях частично решается сокращением количества цветовых каналов, приведением изображения в оттенки серого. Поэтому имеет смысл создание неких наборов методов предварительной обработки, применяемых к различным видам компонентов [3].

Использование методов предварительной обработки может улучшить качество как обнаружения текста, так и его распознавания (без учёта ошибок детекции текста).

## 3. Системы OCR и типовой алгоритм работы

Tesseract OCR [4] – разработка изначально компании Hewlett-Packard, а в последующем Google, на сегодняшний момент одна из многообещающих реализация технологии оптического распознавания текста. Инструмент является бесплатным, открытым, его исходный код доступен на GitHub [5]. Актуальной, выпущенной на момент написания работы, версией является Tesseract 5.0.

Важными этапами работы системы Tesseract являются: предварительная обработка входящих изображений с помощью библиотеки Leptonika, детектирование и сегментация текста, классификация символов. Для распознавания текста рекомендуется использовать встроенную LSTM-сеть. Tesseract можно интегрировать с внешним механизмом детекции текста. В зависимости от качества изображений авторы рекомендуют использовать собственные методы предварительной обработки. Стоит отметить, что система Tesseract поддерживает множество языков, в том числе и русский.

PaddleOCR – набор инструментов OCR. Основным элементом является система PP-OCR [6], включающая в себя три части: обнаружение текста, коррекция обнаруженного фрейма, распознавание символов с помощью CRNN.

Детекция текста производится в основном с помощью моделей ResNet50 или MobileNetV3. Исходный код с

весами моделей для разных языков доступен на GitHub [7]. Имеется поддержка 80 языков в том числе и русского.

CRAFT [8] – метод обнаружения текста, позволяющий эффективно определять область текста. Состоит из двух компонентов: свёрточная нейронная сеть, оценивающая символьные регионы; связывающая нейронная сеть.

## 4. Локализация текста

Первоначальным этапом большинства технологий OCR является локализация текста. Неточная локализация соответственно приводит к неточному распознаванию и зашумлению результатов.

В контексте решения задачи распознавания текста на изображениях десктоп-данных был собран набор, содержащий 715 вручную размеченных скриншотов, существующих десктопных приложений. Для увеличения набора данных с помощью трёх различных библиотек создания UI-интерфейсов было сгенерировано ещё 548 синтетических изображений. В качестве библиотек UI-интерфейсов были использованы: WPF (C#), Pyside (Python), Java Swing. Использование различных UI-библиотек позволило с наименьшими затратами получить наиболее разнообразные примеры интерфейсов.

Набор данных представлял собой изображения в разрешении 640x640 и файлы разметки в COCO формате. Все элементы в наборе данных представлены одним классом – Text. Разметка ограничивающих рамок производилась на уровне слов. Среднее количество элементов на изображение 32.

Для непредвзятого оценивания качества встроенных методов локализаций Tesseract, PaddleOCR и CRAFT из набора данных была выделена тестовая выборка. Размер выборки составил 190 изображений.

Для улучшения качества локализации была обучена нейронная сеть архитектуры YOLO version 5 [9]. Модель была выбрана из-за высоких значений метрик в решении задачи поиска объектов на общедоступных наборах данных, высокой скорости работы. Обучение производилось на тренировочной части набора данных. Сравнение результатов производилось с помощью метрики IoU (Intersection over Union). Численные значения работы разных методов на тестовой части выборки представлены в таблице 1.

Таблица I. Результаты локализации текста

| Метод | IoU |
|---|---|
| Tesseract | 0.31 |
| Paddle (MobileNetv3) | 0.50 |
| CRAFT | 0.86 |
| YOLO | 0.91 |

На рисунках 1-2 представлены графические примеры разницы детекции текста с помощью Tesseract (рис. 1) и обученной сети YOLO v5 (рис.2).

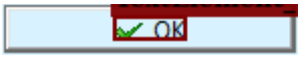


Рис. 1. Локализация текста встроенными методами Tesseract


Рис. 2. Локализация текста YOLO

## 5. Заключение

В работе рассмотрены методы классической предобработки изображений для систем оптического распознавания символов. Обращено внимание на наиболее значимые по мнению авторов недостатки бесплатных OCR-систем, продемонстрированы на примере Tesseract, PaddleOCR, CRAFT.

Решена проблема детектирования текстовых блоков на примере скриншотов десктопных приложений с помощью обучения глубокой нейронной сети YOLO.

Разработанные методы имеют потенциал коммерческого применения в такой области как автоматическое тестирование десктопных и мобильных приложений.

## Благодарности

Авторы выражают благодарность заведующему кафедрой технической кибернетики Самарского национального исследовательского университета имени академика С.П. Королева, доценту, д.т.н., Куприянову Александру Викторовичу, за побуждение и мотивацию для написания, Шутько Вадиму Валерьевичу за идею, без которой не было бы этой работы.

Выражаем благодарность ЛАНИТ Экспертиза за предоставленные вычислительные мощности для обучения моделей.

## Литература

[1] Kshetry, R.L. Image preprocessing and modified adaptive thresholding for improving OCR / R.L. Kshetry // ArXiv, 2021 [Electronic resource]. — Access mode: https://arxiv.org/abs/2111.14075 (15.04.2022).

[2] Peng, X. Building super-resolution image generator for OCR accuracy improvement / X. Peng, C. Wang // International Workshop on Document Analysis Systems. – 2020. – P. 145-160.

[3] Sporici, D. Improving the accuracy of tesseract 4.0 OCR engine using convolution-based preprocessing / D. Sporici, E. Cușnir, C.A. Boiangiu. – Symmetry. – 2020. – T. 12, № 5. – P. 715.

[4] Tesseract User Manual [Electronic resource]. — Access mode: https://tesseract-ocr.github.io/tessdoc/ (15.04.2022).

[5] Tesseract Open Source OCR Engine (main repository) [Electronic resource]. — Access mode: https://github.com/tesseract-ocr/tesseract (07.02.2022).

[6] PP-OCR: A Practical Ultra Lightweight OCR System / Y. Du, C. Li, R. Guo // ArXiv, 2020 [Electronic resource]. — Access mode: https://arxiv.org/abs/2009.09941 (15.04.2022).

[7] PaddleOCR [Electronic resource]. — Access mode: https://github.com/PaddlePaddle/PaddleOCR (15.04.2022).

[8] Character region awareness for text detection / Y. Baek, B. Lee, D. Han // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. – 2019. – P. 9365-9374.

[9] YOLOv5 Documentation [Electronic resource]. — Access mode: https://docs.ultralytics.com/ (15.04.2022).