СЕРВИС ДЛЯ АВТОМАТИЗАЦИИ ПРОЦЕССОВ КОНТРОЛЯ И ПРОВЕРКИ НА СООТВЕТСТВИЕ ДОКУМЕНТОВ, ПОСТУПАЮЩИХ В ОРГАНИЗАЦИЮ

В.В. Видман, ассистент ОИТ ИШИТР,
И. В. Федоров, студент гр. 8К81
Томский политехнический университет
E-mail: ivf6@tpu.ru

Введение

В настоящее время различные организации часто сталкиваются с необходимостью в автоматическом контроле документов, поступающих от клиентов, заказчиков, студентов и т.д. Поэтому вопрос об автоматизации процессов работы с документами является актуальным.

Основным инструментом в реализации данного сервиса выступает сравнение двух строк, а именно – оригинала документа в организации и того, что присылается извне. Существует немало программ, предназначенных для так называемого нечеткого сравнения двух строк: ExamDiff Pro, Compare Suite, но все они не предоставляют пользователю возможность узнать процентное соотношение, некий балл, по которому программа сможет определить, насколько похожи рассматриваемые документы, несмотря на то, что происходит сокращение времени анализа текстов с целью выявления различий.

Целью работы является разработка сервиса, предназначенного для проверки документов на соответствие существующему шаблону.

Алгоритм

Для решения поставленной задачи было разработано SWING приложение [6] в среде Eclipse на языке Java. Оно включает в себя реализацию переноса данных документа со скана в текстовый формат с последующим сравнением полученного фрагмента с шаблоном документа, необходимого организации. В случае их несовпадения система отвергает скан, а при положительном ответе – принимает.

Считывание текстовых данных со скана проводится с помощью алгоритма нечеткого сравнения [1] с использованием имплементированной библиотеки Tesseract OCR [2-3] от Google, чтобы разобрать данные из скана в формате jpeg, png.

Наиболее распространенная особенность нечеткого метода – сравнение строк между собой без привязки к регистру букв (верхнему или нижнему). Это называется сравнением по токенам [5]. Парсер иногда может делать ошибки, в том числе путать верхний и нижний регистры.

Основным пунктом в алгоритме нечеткого сравнения с нахождением процента схожести является поиск расстояния Левенштейна [4]. На языке Java сущность нечеткого поиска реализована в несколько шагов, включающих в себя нахождение совпадений по словам с помощью пересечения множеств – первого (распознанного документа клиента) и второго (шаблона) текста, заранее отсортированных. После выполнения операции пересечения отдельной функцией сохраняются оставшиеся части от текстовых блоков и сортируются. Расстояние Левенштейна мы ищем на данном этапе 3 раза попарно, среди трех переменных: начального пересечения, а также его пересечения с невошедшей частью первой, а затем второй строки. Результатом, то есть процентом схожести документа с предоставленным шаблоном, будет являться максимальное из полученных значений.

Система, сравнивая фактическое значение процента схожести с установленным, дает положительный или отрицательный ответ.

Для сравнения необходимо нажать на кнопку «Прикрепить файл» и выбрать нужный документ. После нажатия кнопки «ОК» система выведет результат и совпадение в процентах.

Формирование установленного, минимального значения схожести происходит путем создания системы, основанной на ряде сканов заполненных бумаг. Иными словами, для каждой справки, для каждого документа должна быть найдена своя константа, с которой будет сравниваться результат нечеткого сравнения, представленный в процентах.

Рис.1. Работа программы с заполненной справкой 2-НДФЛ

Средством для разработки этой части проекта будет выступать сама разработанная система, просто без вывода ответа «да, это тот документ» или «нет, это не тот документ». Необходимо подать ей на вход с одной стороны – шаблон для сравнения, с другой – множество заполненных по-разному четких сканов документов, соответствующих ему. В результате должен получиться ряд чисел (конечных процентных показателей сравнения), размер которого будет равняться размеру выборки по заполненным бумагам. Выбрав из него наименьшее число и взяв его в качестве сравнительной константы для вывода положительного или отрицательного ответа, с ростом выборки мы всё точнее формируем верный ответ системы. При этом, разброс числовых значений будет невелик.

Реализация – с использованием циклической программы на языке Java, куда предварительно помещена нужная часть разработанной системы без ответа, а также путь к множеству заполненных файлов.

Тестирование эффективности алгоритма

Для оценки алгоритма нечеткого сравнения распознанных текстов двух документов тестирование было проведено на различных входных данных.

Во время тестирования были выбраны два документа, подача системе которых каждую итерацию была различной: обычный скан, фото скана, инверсия скана, инверсия фото скана. Четыре вида входных данных сравнивались с шаблоном, якобы размещенном внутри системы, то есть с чистым сканом документа.

При тестировании учитывалось не только процентное отклонение от оригинала, но и время, затраченное системой на работу алгоритма.

Сравнение эффективности работы системы при обработке различного типа входных данных на одинаковом шаблоне для двух различных документов представлено в таблице 1.

Таблица 1. Показатели работы алгоритма при различных типах входных данных

Тип входных данных	Справка 2-НДФЛ		Заявление на получение справки об отсутствии судимости	
	Схожесть, %	Время, с	Схожесть, %	Время, с
Обычный скан	97	15,15	97	12,83
Фото скана	31	24,21	27	23,16
Инверсия скана	96	13,39	94	22,15
Инверсия фото	25	23,48	21	22,45

Заключение

В результате тестирования можно сделать вывод, что система хорошо работоспособна при условии, что входными данными будут являться сканы документов. Более того, модификации над сканом практически не портят верный результат на выходе, потому что изображение остается:

1. Ровным.
2. Четким.
3. Без лишних элементов на изображении.

Живя в век информационных технологий, различные организации часто сталкиваются с необходимостью в автоматическом контроле документов, поступающих от клиентов, заказчиков, студентов и т.д.

Налаженная система проверки документов не только будет сводить вероятность оформления не тех документов к минимальному значению, но и позволит сэкономить время работы тех, кто производил контроль над ними вручную. Следовательно, повышается продуктивность работы той или иной системы, в которую внедрена разрабатываемая во время работы над проектом модель.

Сложность процесса оформления и визуальной оценки документов и изнурительные повторные визиты нивелируют лояльность клиента к организации. В результате она несёт репутационные потери и снижение клиентопотока.

Для различных сервисов данная система – это контроль сотен и даже тысяч заявок на соответствие прикрепленных справок, сертификатов или дипломов требованиям по оказанию услуги, по повышению квалификации и т.д.

Список использованных источников

1. CitForum [Электронный ресурс]: Нечеткое сравнение коллекций: семантический и алгоритмический аспекты: 2008 г. URL: http://citforum.ru/SE/project/fuzzy_comp/
2. Tesseract OCR [Электронный ресурс]: Tesseract documentation. URL: https://tesseract-ocr.github.io/
3. Bytespace [Электронный ресурс]: Распознавание текста с помощью OCR. URL: http://bytepace.com/ru/blog/tesseract
4. Реализации алгоритмов [Электронный ресурс]: Расстояние Левенштейна. URL: https://ru.wikibooks.org/wiki/Реализации_алгоритмов/Расстояние_Левенштейна
5. FuzzyWuzzy – нечеткое сравнение строк [Электронный ресурс]. 2021 г. URL: https://egorovegor.ru/fuzzywuzzy-nechyotkoe-sravnenie-strok-rasstoyanie-levenshtejna/
6. Java-Online [Электронный ресурс]: Библиотека Swing. 2016 г. URL: http://java-online.ru/libs-swing.xhtml