

Познание распознавания-2

PROMETECH

Приветствую всех читателей! Как и договаривались, продолжаем разбирать по полочкам процесс распознавания документов OCR-системой. Переходим, на мой взгляд, к самому «вкусному»...

Продолжение, начало см. в МК, № 52 (483)

Вот это MDA...

Начинается распознавание документа с анализа его структуры. Подлежащий распознаванию документ часто выглядит куда изощреннее, чем белая страница с черным текстом. Иллюстрации, таблицы, колонтитулы, сноски, элементы форматирования, фоновые изображения, применяемые для оформления, усложняют структуру страницы. Для того чтобы корректно воспроизводить в электронном виде такие документы, все современные OCR-программы начинают распознавание именно с анализа структуры. Как правило, при этом выделяют несколько иерархически организованных логических уровней.

Следует отметить, что начиная с новой, девятой версии программы ABBYY FineReader, в основу которой легла технология адаптивного распознавания документов ADRT, наивысшим уровнем этой иерархии является документ в целом, а не отдельные его страницы, как это было ранее.

Технология ADRT — первая попытка распознавать изображения документов целиком и выдавать на выходе логически структурированные, легко редактируемые документы. Традиционные OCR-программы обеспечивают распознавание символов и сохранение компоновки, которые основываются на анализе на уровне символов и страниц. Технология ADRT позволяет программе ABBYY FineReader в дополнение к этим двум уровням выйти на третий уровень анализа — уровень документа, и точно воспроизводить элементы форматирования, сохраняя их первоначальную суть.

Подробнее о том, как работает технология ADRT, вы можете прочитать в моем обзоре ABBYY Finereader 9.0 под названием «Ученый читатель» (см. «МК», №50 (481)). Здесь же скажу лишь, что уровень документа — самый сложный уровень для анализа и распознавания. Ведь при обработке документа как единого целого критерий точности распространяется как на распознавание текста, так и на распознавание элементов форматирования, причем в увязке их с общей структурой всего документа. Кроме того, сам термин «оптическое распознавание символов (OCR)» на этом уровне лишь частично отражает существо задачи, хотя и продолжает использоваться в силу устоявшейся традиции.

В отличие от традиционных технологий постраничного анализа, ADRT анализирует документы целиком: все страницы, компоновку и элементы форматирования одновременно. Строится логическая модель, содержащая информацию о структуре документа, его элементах и их характеристиках, таких как начертание и стиль шрифтов. Далее эта модель используется для точного воспроизведения документа с сохранением его целостности и всех логических связей между элементами.

Благодаря ADRT, OCR не просто имитирует внешний вид исходного документа, а в буквальном смысле стремится понять сущность каждого элемента форматирования и «определить», где именно они должны быть расположены, в каком формате и в каком порядке. К примеру, при преобразовании отсканированного документа в файл Microsoft Word колонтитул с номером страницы будет воспроизведен естественным образом, именно как элемент форматирования Microsoft Word «колонтитул-номер страницы», который можно будет изменить или удалить сразу на всех страницах.

После анализа документа целиком, его логической структуры, следующим уровнем анализа является страница. На

следующей ступени иерархии располагаются таблица и текстовый блок, и так далее (рис. 1) — уровни анализа документа:

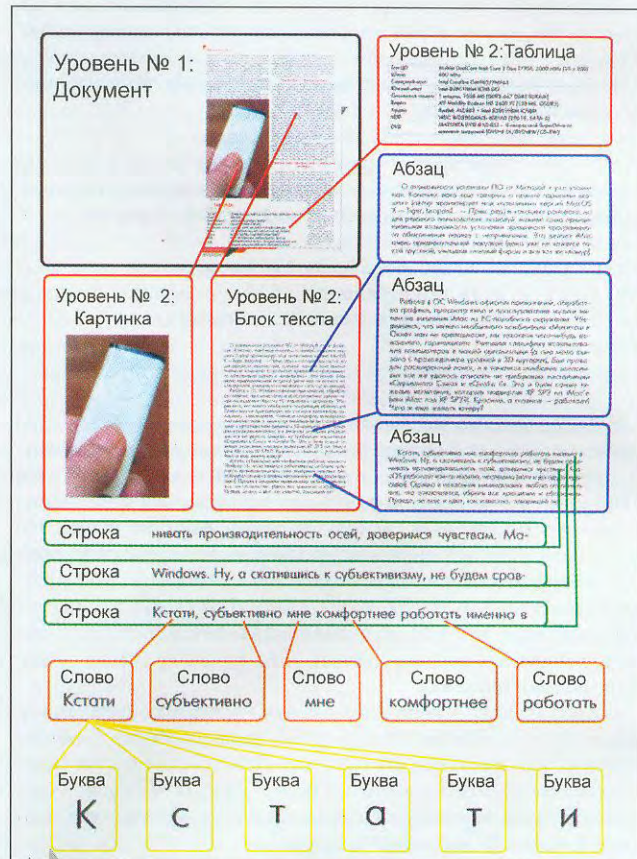


Рис. 1

- ✓ страница;
- ✓ таблица, блок текста;
- ✓ ячейка таблицы;
- ✓ абзац, картинка;
- ✓ строка;
- ✓ слово, картинка внутри строки;
- ✓ буква (символ).

Понятно, что любой высокоуровневый объект может быть представлен как набор объектов более низкого уровня: буквы образуют слово, слова — строки и т.д. Поэтому анализ всегда начинается в направлении сверху вниз. Программа делит документ на страницы, страницу на объекты, их, в свою очередь, на объекты низших уровней, и так далее, вплоть до символов. Когда символы выделены и распознаны, начинается обратный процесс — «сборка» объектов высших уровней — который завершается формированием целой страницы. Такая процедура называется **многоуровневым анализом документа**, или **MDA** (multilevel document analysis).

Объекты любого уровня OCR распознает в полном соответствии с принципами IPA, суть которых мы рассмотрели в прошлый раз. В первую очередь выдвигаются гипотезы относительно типов обнаруженных объектов, затем они целена-

правленно проверяются. При этом система учитывает найденные ранее особенности данного документа, а также сохраняет вновь поступающую информацию (обучается).

Но допустим, все объекты текущего уровня распознаны. OCR переходит к детальному анализу одного из них, определенного, к примеру, как текстовый блок. И вдруг оказывается, что результаты анализа этого блока крайне неубедительны; не удается выделить ни абзацы, ни строки. Система, как мы помним, часто ведет себя подобно живому существу... и потому реакция будет вполне естественна: надо заново определить, что это за блок (причем с учетом уже накопленных знаний). Повторный анализ позволяет внести коррективы: да, это текст, но наложенный на фоновое изображение. После дополнительной обработки распознавание будет продолжено — и уже без ошибок (о распознавании текста на фоновом изображении речь пойдет ниже).

Вероятность грубых ошибок, связанных с неверным распознаванием объектов более высоких уровней, снижает эффективность алгоритма обратной связи в процедуре MDA на всех этапах многоуровневого анализа. (см. таблицу)

ТАБЛИЦА

Процедуры	Уровень	Процедуры
10. Сохранение документа	ДОКУМЕНТ	1. Выделение текстовых блоков
9. "Сборка" документа	БЛОК ТЕКСТА, ТАБЛИЦА	2. Выделение строк
8. "Сборка" строк	СТРОКА	3. Деление на слова
7. Структурирование гипотез	СЛОВО	4. Деление на символы
6. Создание моделей слова	СИМВОЛ	5. Распознавание символов

Мы рассмотрели общий ход процесса распознавания, самое время пройти по этапам.

Вначале была страница...

На первом этапе распознавания система структурирует страницу, выделяет на ней текстовые блоки. Как мы знаем, современные документы часто содержат всевозможные элементы дизайна: иллюстрации, колонтитулы, цветной фон или фоновые изображения и т.д. Основная задача на данном этапе состоит в том, чтобы отделить текст от иллюстраций и «подложенных» текстур.

Здесь уместно будет упомянуть о практикуемых методах подготовки. Все современные системы распознавания начинают процесс «знакомства» с созданием черно-белого изображения документа. При этом подлежащее анализу изображение чаще всего цветное или полутонное (то есть состоящее из разных оттенков серого цвета, подобно картинке на экране черно-белого телевизора). Любая OCR-система прежде всего преобразует такое изображение в монохромное, состоящее только из черных и белых точек. Процесс преобразования называется *бинаризацией*, он всегда предшествует детальной обработке распознаваемой страницы.

Далее — в идеальном варианте — все выглядит достаточно просто. Блок текста, состоящий из строк, должен иметь характерную линейчатую структуру. Разделив этот блок на строки, можем приступить к выделению слов. Однако на практике столь простые варианты встречаются нечасто. Возьмите любой документ, где строки текста наложены на цветной фон, и представьте, как будет выглядеть эта страница в черно-белом варианте. Вокруг каждого символа обнаружатся десятки и сотни «лишних» точек, оставшихся от фона. Лишние точки будут искажать очертания букв и даже границы строк, приводя к ошибкам.

Система не пытается решать задачу бинаризации «в лоб». Принцип целенаправленности диктует иной подход к обнаружению строк в текстовом блоке или слов в строке: они



Рис.2

должны быть где-то здесь, надо только суметь их узнать. Например, ABBYY FineReader для повышения качества поиска использует процедуры *интеллектуальной фильтрации фоновых текстур* и *адаптивной бинаризации*. Первая позволяет уверенно отделять строки текста от сколь угодно сложного фона, вторая — гибко выбирать оптимальные для данного участка параметры бинаризации. Естественно, к этим процедурам система прибегает не всегда, а лишь в тех случаях, когда предварительный анализ указывает на подобную необходимость. В каждом конкретном случае OCR выбирает подходящий «инструмент», опираясь на информацию, накопленную в процессе анализа документа.

Например, идет анализ строки. Система занята поиском объектов уровня «слово». На первый взгляд, проще всего разделить строку на слова по найденным пробелам. Однако первичный анализ показывает, что в конце строки пробелы попадаются заметно чаще, чем в начале. Нет ли тут ошибки, не искажено ли изображение? Процедура адаптивной бинаризации исследует яркость фона и насыщенность черного цвета на протяжении всей строки и подбирает оптимальные параметры бинаризации для каждого фрагмента по отдельности. И вот результат: оказывается, часть символов в конце строки получилась слишком светлой и могла бы быть «потеряна» при обычной обработке, но в результате применения адаптивной бинаризации все слова будут выделены точно (рис. 2).

Однако помните, что программа — всего лишь программа, документ по возможности следует своими силами подготовить к распознаванию, отсканировать его как можно лучше, в сложных случаях самостоятельно разбить изображение на типовые блоки. Это сэкономит время и повысит скорость и качество распознавания. Повторяться не буду, желающие найдут советы по оптимизации распознавания в других материалах. Вообще, по результатам тестов и в ходе практического использования, эффективность процедуры обнаружения строк и слов при использовании интеллектуальной фильтрации фоновых текстур и адаптивной бинаризации оказалась настолько высока, что в отдельных случаях система опережает даже человека!

Разделяй и распознавай

Разделив строку на отдельные слова, система распознавания приступает к наиболее ответственному и кропотливому этапу распознавания — обработке символов.

Сегментация отдельных символов — одна из наиболее сложных задач в OCR. Это объясняется тем, что границы между буквами часто нечеткие, штрихи соседних букв соприкасаются, и это мешает делить слова на символы. Например, две или более буквы распознаются как одна, или одна — как две или три. Существуют шрифты, у которых, несмотря на общее хорошее качество печати, встречаются склейки. Еще одна проблема — грязные или плохо отпечатанные изображения, ведь даже мелкие пятна могут затенять определяющие части символа или преобразовывать один в другой.

Отдельно стоит остановиться и на погрешностях сканирования.

Давайте посмотрим, что происходит с символом при сканировании. Жизненный цикл черно-белого образа буквы при сканировании состоит из следующих этапов:

- ✓ непрерывный черно-белый оригинал (на бумажном носителе);
- ✓ непрерывный серый оригинал — виртуальный образ (после отражения света);

▶ Окончание на стр. 45

▲ Окончание. Начало на стр. 28-29

✓ дискретный серый отсканированный образ (после укладки на сетку сканера);

✓ дискретный черно-белый образ (после бинаризации).

Четвертый пункт отсутствует в случае, если мы сканируем серое изображение. Для первого пункта «непрерывность» и «бинарность» изображения на бумаге — понятия условные, так как на самом деле типографская краска наносится неровно, а бумага не идеально белая, а то и вовсе текст и/или фон — цветные.

Из идеальной буквы А, которая была напечатана на бумаге, получается нечто угловатое, состоящее из квадратов (рис. 3).



Рис.3

Ступенчатая структура возникает из-за наложения символа на сетку сканера, а всевозможные пупырышки — из-за аппаратных погрешностей и некоторых других факторов. В результате из одинаковых символов при сканировании мы получаем множество различных образов, при этом их количество достаточно велико. При таком положении дел задача распознавания усложняется: если бы образы одного символа были бы абсолютно одинаковыми, достаточно было бы объединить их в группу и распознать одного представителя этой группы. При этом мы были бы абсолютно уверены, что все символы в этой группе — это именно определенный при

распознавании символ. В случае большого многообразия изображений символов часто возникают ошибки, когда один символ ошибочно может быть распознан как другой.

Еще одна проблема — страница, расположенная с нарушением границ или перекосом, создает немного искаженные символьные изображения, которые могут путать программное обеспечение распознавания. Даже когда изображения — чистые, странные или декоративные начертания могут вызывать проблемы, деформируя символы для художественного эффекта (яркий пример — букваца). Кроме того, буквы могут иметь вариации в пределах одного шрифта, если они напечатаны на разных принтерах.

Разработка алгоритмов, которые позволяют распознавать символы, несмотря на эти проблемы — трудная задача. Разработчики должны сбалансировать гибкость ПО с его точностью. Если программное обеспечение недостаточно гибко, то оно будет неточно сегментировать символ, когда будет сталкиваться с различными вариациями начертания. С другой стороны, слишком много гибкости может также вызывать ошибки. К примеру, отличие между цифрой «3» и буквой «З» в ряде шрифтов незначительное, и гибкий алгоритм может спутать их.

Следует заметить, что разделение слов на символы и собственно распознавание букв, как и все остальные механизмы многоуровневого анализа документа, реализованы в виде составных частей единой процедуры. Это позволяет в полной мере использовать преимущества принципов ИРА. Выделенные изображения символов поступают на рассмотрение механизма распознавания букв, называемых *классификаторами*.

По моему мнению, это один из наиболее интересных этапов работы OCR-системы, заслуживающий детального рассмотрения. Поэтому о классификаторах мы поговорим в следующий раз.