УДК 004

Извлечение изображений из PDF файла

Вихляев Дмитрий Романович
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье содержится описание методов поиска и извлечения изображений из PDF файла. Инструментом программирования является язык python. В результате исследования будут разобраны структурное представление изображений и реализации программ, позволяющих извлекать картинки из PDF документа.
Ключевые слова: PDF, парсинг, fitz.

Extracting images from a PDF file

Vikhlyaev Dmitry Romanovich
Sholom-Aleichem Priamursky State University
Student

Abstract
This article describes methods for searching and extracting images from a PDF file. The programming tool is the python language. As a result of the research, the structural representation of images and the implementation of programs that allow you to extract images in a PDF document will be analyzed.
Keywords: PDF, parsing, fitz.

1 Введение

1.1 Актуальность

PDF-документы могут содержать различные типы данных, включая текст, изображения, графику и даже вложенные файлы. Извлечение изображений позволяет работать с этими графическими данными независимо от остального содержимого. В мире бизнеса, исследований и образования, изображения часто являются важной частью документов. Программы для извлечения изображений обеспечивают удобные средства работы с этими графическими данными. В некоторых случаях, когда изображения важны для отчетности, можно использовать программу извлечения изображений, чтобы автоматически извлекать графическую информацию из PDF и включать ее в отчеты или аналитические документы. Помимо этого, изображения могут быть использованы для создания визуализаций, графиков или слайдов для презентаций. Это особенно важно, если в PDF-документах содержится информация, которую нужно визуализировать.

1.2Обзор исследований

А.С.Ломакин разработал встраиваемую библиотеку для парсинга PDF-отчётов и автоматического интегрирования данных в карту пациента в области офтальмологии [1]. Г.И.Атаева, З.М.К.Адизова реализовали процесс конвертирования изображений в формат PDF с помощью python [2]. Денисович Д.П. описал модифицированный стеганографический алгоритм для скрытия изображения в файле формата PDF [3]. А.С.Бурый исследовал проблему анализа оригинальности изображений документа в формате PDF [4]. М.В.Дагаев осуществил сжатие PDF-файлов, состоящих из растровых изображений, как способ уменьшения объемов данных для оцифрованных документов telegram [5]. А.А.Михайлов автоматизировал разметку данных для сегментации изображений документов с использованием глубоких нейронных сетей [6].

1.3Цель исследования

Цель исследования – описать процессы поиска и извлечения изображений в PDF документе.

2 Материалы и методы

Для реализации используется язык программирования python, модуль «fitz» для парсинга файла и библиотека PIL для обработки изображений.

3 Результаты и обсуждения

Структура PDF-документа может включать изображения в виде графических объектов. Формат PDF поддерживает различные типы изображений и предоставляет структуры данных для их хранения.

В PDF каждое изображение представляется как объект. Этот объект содержит информацию о самом изображении, такую как его размеры, цветовая глубина, цветовое пространство и т.д.

Индексные таблицы могут использоваться для определения цветовых палитр для изображений с использованием цветовых индексов, что может быть полезно, например, для изображений в формате GIF.

PDF поддерживает различные цветовые пространства для изображений, например, RGB, CMYK, Grayscale. Это важно для правильного отображения цветов при просмотре документа.

Изображения могут быть встроены в потоковые объекты, которые содержат байтовые данные изображения. Данные представляют собой фактическое изображение в бинарном формате.

Данный тип документа поддерживает различные форматы изображений, такие как JPEG, PNG, TIFF. Каждый формат имеет свои особенности, и PDF может включать различные типы изображений в одном документе.

Дополнительные метаданные могут быть включены в объект изображения, включая информацию об авторе, дате создания, разрешении и другие параметры.

Примерный вид записи объекта изображения в PDF приведён на рисунке 1.

```
<<
  /Type /XObject
  /Subtype /Image
  /Width 800
  /Height 600
  /ColorSpace /DeviceRGB
  /BitsPerComponent 8
  /Filter /DCTDecode
  /Length 12345  % Длина потока с байтовыми данными изображения
>>
stream
  ... % Байтовые данные изображения
endstream
```

Рис. 1. Структура хранения изображения PDF файла

Здесь представлены ключевые атрибуты, такие как тип, подтип, размеры, цветовое пространство, бит на компонент, алгоритм кодирования и полный размер в байтах. Конкретная структура может различаться в зависимости от того, как было вложено изображение в PDF-документ и каким программным обеспечением оно было создано.

Каждое изображение в PDF представлено в виде объекта изображения (Image Object). Этот объект содержит метаданные и бинарные данные изображения.

Объект изображения имеет тип «/XObject» и подтип «/Image», что указывает на то, что это изображение.

В объекте изображения указываются размеры изображения (ширина и высота в пикселях или единицах измерения) и разрешение (dots per inch - DPI).

Цветовое Пространство (Color Space) указывает на то, в каком цветовом пространстве представлены цвета изображения, например, RGB или CMYK.

Битовая Глубина (Bits Per Component) определяет количество бит на каждый цветовой компонент пикселя (например, 8 бит на красный, 8 бит на зеленый, 8 бит на синий в случае RGB).

Изображения могут подвергаться сжатию с использованием различных фильтров. Например, JPEG-изображения могут использовать фильтр /DCTDecode.

Фактические бинарные данные изображения хранятся в потоковом объекте внутри PDF-файла. Если эти данные скопировать, перенести в другой файл и открыть в программе для просмотра изображения, то скорее всего откроется изображение, даже если у файла небело или было неверно указанно расширение.

В качестве примера для парсинга, использован PDF файл, взятый из интернета [8].

Для упрощения работы была подключена библиотека «PyMuPDF», из которой необходимо подключить модуль «fitz». В примере заданы минимальные размеры для извлекаемых изображений, папка, в которую полученные изображения помещаются и формат, в который они конвертируются.

```
import os
import fitz  # PyMuPDF
import io
from PIL import Image

# Output directory for the extracted images
output_dir = "extracted_images"
# Desired output image format
output_format = "png"
# Minimum width and height for extracted images
min_width = 100
min_height = 100
# Create the output directory if it does not exist
if not os.path.exists(output_dir):
    os.makedirs(output_dir)
```

```
# file path you want to extract images from
file = "G2_arch.pdf"
# open the file
pdf_file = fitz.open(file)
```

Рис. 2. Заданные требования для извлечения изображений

Данный модуль позволяет легко находить нужный данные в PDF фале. Экземпляр представлен как итерируемый объект, состоящий из страниц документа. Обращаясь к каждому элементу по отдельности, осуществляется поиск нужных элементов (рис.3).

```python
# Iterate over PDF pages
for page_index in range(len(pdf_file)):
    # Get the page itself
    page = pdf_file[page_index]
    # Get image list
    image_list = page.get_images(full=True)
    # Print the number of images found on this page
    if image_list:
        print(f"[+] Found a total of {len(image_list)} images in page {page_index}")
    else:
        print(f"[!] No images found on page {page_index}")
    # Iterate over the images on the page
    count=0
    for image_index, img in enumerate(image_list, start=1):
        # Get the XREF of the image
        xref = img[0]
        # Extract the image bytes
        base_image = pdf_file.extract_image(xref)
        image_bytes = base_image["image"]
        # Get the image extension
        image_ext = base_image["ext"]
        # Load it to PIL
        image = Image.open(io.BytesIO(image_bytes))
        # Check if the image meets the minimum dimensions and save it

        if image.width >= min_width and image.height >= min_height:
            image.save(
                open(os.path.join('/content/gpu5/', f"{count}.{output_format}"), "wb"),
                format=output_format.upper())
            count+=1
        else:
            print(f"[-] Skipping image {image_index} on page {page_index} due to its small size.")
```

Рис. 3. Извлечение изображений из PDF файла

Извлечь изображение можно и без дополнительных библиотек или парсинга PDF структуры. Поскольку все изображения в документе хранятся так же как самостоятельный файл. Достаточно осуществить алгоритм поиска сигнатур необходимых форматов. Определения формата файла по сигнатуре используется в операционной системе «linux». На рисунке 4 представлена функция осуществляющая поиск и извлечение изображений указанных форматов.

```python
def extract_images_from_pdf(pdf_path, output_folder):
    with open(pdf_path, 'rb') as file:
        pdf_data = file.read()

        # Поиск сигнатур растровых изображений (PNG, JPEG, GIF)
        image_signatures = [b'\x89PNG', b'\xFF\xD8\xFF', b'GIF']

        for signature in image_signatures:
            start = 0
            while True:
                start = pdf_data.find(signature, start)
                if start == -1:
                    break

                # Найдена сигнатура, записываем данные в файл
                img_start = start
                img_end = pdf_data.find(b'endstream', img_start)
                img_data = pdf_data[img_start:img_end]

                img_path = f"{output_folder}/img_{start}_{signature.hex()}.png"
                with open(img_path, 'wb') as img_file:
                    img_file.write(img_data)

                start = img_end
```

Рис. 4. Извлечение изображений из PDF файла по сигнатуре

В результате обе программы извлекут и сохранят изображения с расширением PNG (рис.5).

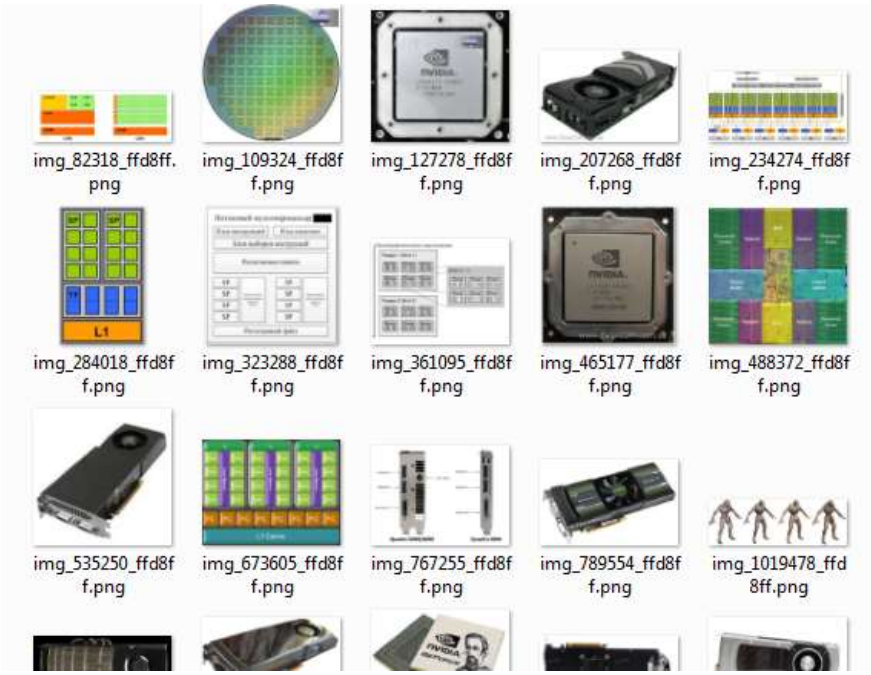

Рис. 5. Извлечённые изображения

Таким образом, была реализована программа, извлекающая картинки из PDF документа. В результате бала рассмотрена структура хранения изображений и приведены примеры методов поиска картинок с описанием технологий и используемых инструментов.

Библиографический список

1. Ломакин А.С. Разработка встраиваемой библиотеки для парсинга PDF-отчётов и автоматического интегрирования данных в карту пациента в области офтальмологии // NovaUm.Ru. 2023. № 44. С. 4-7.
2. Атаева Г.И., Адизова З.М.К. Конвертирование изображений в формат PDF с помощью python // Universum: технические науки. 2022. № 4-1 (97). С. 29-31.
3. Денисович Д.П. Модифицированный стеганографический алгоритм для скрытия изображения в файле формата PDF // В сборнике: Научное сообщество студентов XXI столетия. Технические науки. Сборник статей по материалам LXXX студенческой международной научно-практической конференции. 2019. С. 15-22.
4. Бурый А.С. Проблема анализа оригинальности изображений документа формата PDF // Молодой ученый. 2023. № 22 (469). С. 8-10.
5. Дагаев М.В. Сжатие PDF-файлов, состоящих из растровых изображений, как способ уменьшения объемов данных для оцифрованных документов // Межотраслевая информационная служба. 2011. № 3. С. 28-33.
6. Михайлов А.А. Автоматическая разметка данных для сегментации

изображений документов с использованием глубоких нейронных сетей // Труды Института системного программирования РАН. 2022. Т. 34. № 6. С. 137-146.
7. Горбачев В.Н., Метелёв И.К., Кайнарова Е.М., Полякова М.А. Стеганографическая защита изображений из pdf документов на основе конвертора PDF-SVG // В сборнике: GraphiCon 2017. Труды 27-й Международной конференции по компьютерной графике и машинному зрению. Пермский государственный национальный исследовательский университет. 2017. С. 108-111.
8. http://optic.cs.nstu.ru/files/CC/CompGraph/G2_arch.pdf