Contents

1 PDF Section Extractor for CouchDB

1.1 Overview

The `PDFSectionExtractor` class provides automated extraction of section headers and paragraphs from PDF attachments stored in CouchDB documents.

1.2 Features

- **Automatic PDF detection**: Finds PDF attachments in CouchDB documents

- **In-memory processing**: Works directly with PDF bytes (no temp files)
- **Text extraction**: Uses PyMuPDF (fitz) for high-quality text conversion
- **Intelligent parsing**: Identifies section headers vs. paragraphs
- **Metadata extraction**: Extracts title, abstract, keywords, authors
- **Content search**: Find sections by keyword
- **Batch processing**: Extract from multiple documents at once

1.3 Installation Requirements

The class requires PyMuPDF (fitz) for PDF text extraction:

```
# Install PyMuPDF
pip install PyMuPDF

# Verify installation
python -c "import fitz; print(f'PyMuPDF version: {fitz.__version__}')"
```

This uses the same `pdf_to_text` function as `jupyter/ist769_skol.ipynb` for consistent text extraction.

1.4 Quick Start

```
from pdf_section_extractor import PDFSectionExtractor

# Initialize (uses environment variables for credentials)
extractor = PDFSectionExtractor()

# Extract sections from a PDF attachment
sections = extractor.extract_from_document(
    database='skol_dev',
    doc_id='00df9554e9834283b5e844c7a994ba5f',
    attachment_name='article.pdf'  # Optional: auto-detects
)

print(f"Extracted {len(sections)} sections")
for i, section in enumerate(sections[:5], 1):
    print(f"{i}. {section[:60]}...")
```

1.5 Usage Examples

1.5.1 Example 1: Basic Extraction

```python
from pdf_section_extractor import PDFSectionExtractor

extractor = PDFSectionExtractor(verbosity=1)

# Extract from specific document
sections = extractor.extract_from_document(
    database='skol_dev',
    doc_id='00df9554e9834283b5e844c7a994ba5f'
)

print(f"Extracted {len(sections)} sections")
```

1.5.2 Example 2: Extract Metadata

```python
# Extract sections
sections = extractor.extract_from_document(
    database='skol_dev',
    doc_id='00df9554e9834283b5e844c7a994ba5f'
)

# Extract metadata
metadata = extractor.extract_metadata(sections)

print(f"Title: {metadata['title']}")
print(f"Keywords: {', '.join(metadata['keywords'])}")
print(f"Abstract: {metadata['abstract'][:200]}...")
print(f"Sections found: {', '.join(metadata['sections_found'])}")
```

Output:

```
Title: A new species of Arachnopeziza from Taiwan
Keywords: discomycetes, Helotiales, Hyaloscyphaceae
Abstract: This paper describes and illustrates a new species...
Sections found: Introduction, Acknowledgments
```

1.5.3 Example 3: Search Sections

```python
# Find all sections mentioning specific terms
ascospore_sections = extractor.get_section_by_keyword(
    sections,
    'ascospores',
    case_sensitive=False
)
```

```
print(f"Found {len(ascospore_sections)} sections about ascospores")

for section in ascospore_sections:
    print(f"- {section[:100]}...")
```

1.5.4 Example 4: Batch Processing

```
# Extract from multiple documents
doc_ids = [
    '00df9554e9834283b5e844c7a994ba5f',
    'another-doc-id',
    'yet-another-doc-id'
]

results = extractor.extract_from_multiple_documents(
    database='skol_dev',
    doc_ids=doc_ids
)

for doc_id, sections in results.items():
    print(f"{doc_id}: {len(sections)} sections")
```

1.5.5 Example 5: List Attachments

```
# See what attachments are available
attachments = extractor.list_attachments(
    database='skol_dev',
    doc_id='00df9554e9834283b5e844c7a994ba5f'
)

for name, info in attachments.items():
    content_type = info.get('content_type', 'unknown')
    size = info.get('length', 0)
    print(f"{name}: {content_type} ({size:,} bytes)")
```

Output:

```
article.txt: text/simple; charset=UTF-8 (7,987 bytes)
article.txt.ann: text/plain (8,371 bytes)
article.pdf: application/pdf (740,079 bytes)
```

1.5.6 Example 6: Custom Credentials

```
# Use custom credentials instead of environment variables
extractor = PDFSectionExtractor(
```

```
    couchdb_url='http://localhost:5984',
    username='admin',
    password='secret',
    verbosity=2  # More detailed logging
)

sections = extractor.extract_from_document(
    database='skol_dev',
    doc_id='00df9554e9834283b5e844c7a994ba5f'
)
```

1.6 Class Methods

1.6.1 Initialization

```
PDFSectionExtractor(
    couchdb_url=None,       # Default: from COUCHDB_URL env var
    username=None,           # Default: from COUCHDB_USER env var
    password=None,           # Default: from COUCHDB_PASSWORD env var
    verbosity=1              # 0=silent, 1=info, 2=debug
)
```

1.6.2 Main Methods

1.6.2.1 `extract_from_document()` Extract sections from a PDF attachment.

```
sections = extractor.extract_from_document(
    database='skol_dev',
    doc_id='document-id',
    attachment_name='article.pdf',  # Optional: auto-detects
    cleanup=True                     # Delete temp files
)
# Returns: List[str] of sections/paragraphs
```

1.6.2.2 `extract_from_multiple_documents()` Extract from multiple documents at once.

```
results = extractor.extract_from_multiple_documents(
    database='skol_dev',
    doc_ids=['id1', 'id2', 'id3'],
    attachment_name=None  # Auto-detect PDF
)
# Returns: Dict[str, List[str]] mapping doc_id to sections
```

1.6.2.3 `extract_metadata()`

Extract common metadata from sections.

```
metadata = extractor.extract_metadata(sections)
# Returns: {
#     'title': str,
#     'authors': List[str],
#     'abstract': str,
#     'keywords': List[str],
#     'sections_found': List[str]
# }
```

1.6.2.4 `get_section_by_keyword()`

Search sections for keywords.

```
matching = extractor.get_section_by_keyword(
    sections,
    keyword='methodology',
    case_sensitive=False
)
# Returns: List[str] of matching sections
```

1.6.3 Utility Methods

1.6.3.1 `list_attachments()`

```
attachments = extractor.list_attachments(
    database='skol_dev',
    doc_id='document-id'
)
# Returns: Dict[str, Dict] of attachment metadata
```

1.6.3.2 `find_pdf_attachment()`

```
pdf_name = extractor.find_pdf_attachment(
    database='skol_dev',
    doc_id='document-id'
)
# Returns: str (attachment name) or None
```

1.6.3.3 `download_pdf()`

```
pdf_path = extractor.download_pdf(
    database='skol_dev',
    doc_id='document-id',
    attachment_name='article.pdf',
    output_path=None  # Uses temp file if None
```

```
)
# Returns: str (path to downloaded PDF)
```

1.7 Parsed Section Types

The extractor identifies:

1. **Headers**: Section titles (Introduction, Methods, etc.)
2. **Paragraphs**: Content blocks
3. **Metadata**: Title, authors, abstract, keywords
4. **Figure captions**: "Fig. 1. Description..."
5. **References**: Bibliography entries

1.7.1 Header Detection

The parser identifies headers by: - Common keywords (Introduction, Abstract, Methods, etc.) - Title case formatting - Short length (<100 characters) - Centered text (lots of leading spaces) - All caps (but not ending with punctuation)

1.7.2 Paragraph Assembly

Paragraphs are assembled by: - Joining consecutive non-blank lines - Breaking on blank lines - Breaking on detected headers - Minimum length threshold (default: 10 characters)

1.8 Configuration Options

1.8.1 Verbosity Levels

```
extractor = PDFSectionExtractor(verbosity=0)  # Silent
extractor = PDFSectionExtractor(verbosity=1)  # Info (default)
extractor = PDFSectionExtractor(verbosity=2)  # Debug
```

Output at verbosity=2:

```
Connected to CouchDB at http://localhost:5984
Extracting sections from document abc123 in skol_dev
Found PDF attachment: article.pdf (application/pdf)
Downloaded PDF: /tmp/tmpXYZ.pdf (740,079 bytes)
Extracted 8402 characters from PDF
Parsed 30 sections/paragraphs
Cleaned up temporary file: /tmp/tmpXYZ.pdf
```

1.8.2 Parsing Options

```
sections = extractor.parse_text_to_sections(
    text,
    min_paragraph_length=10  # Minimum characters per paragraph
)
```

1.9 Environment Variables

The class uses these environment variables by default:

```
export COUCHDB_URL="http://localhost:5984"
export COUCHDB_USER="admin"
export COUCHDB_PASSWORD="your-password"
```

Or set them in your code:

```
import os

os.environ['COUCHDB_URL'] = 'http://localhost:5984'
os.environ['COUCHDB_USER'] = 'admin'
os.environ['COUCHDB_PASSWORD'] = 'secret'

from pdf_section_extractor import PDFSectionExtractor
extractor = PDFSectionExtractor()
```

1.10 Error Handling

The class provides helpful error messages:

```
try:
    sections = extractor.extract_from_document(
        database='skol_dev',
        doc_id='missing-doc'
    )
except ValueError as e:
    print(f"Error: {e}")
    # Error: No PDF attachment found in document missing-doc.
    # Available attachments: ['data.csv', 'image.png']

except ImportError as e:
    print(f"Error: {e}")
    # Error: PyMuPDF (fitz) is required for PDF extraction.
    # Install with: pip install PyMuPDF

except Exception as e:
    print(f"Unexpected error: {e}")
```

1.11 Performance

For typical scientific papers (4-10 pages): - **Download**: < 1 second - **Text extraction**: < 2 seconds - **Parsing**: < 1 second - **Total**: ~3-4 seconds per document

For batch processing:

```
# Process 100 documents
results = extractor.extract_from_multiple_documents(
    database='skol_dev',
    doc_ids=doc_ids  # 100 IDs
)
# Takes ~5-6 minutes (3-4 seconds × 100)
```

1.12 Complete Example Script

See example_pdf_extraction.py for a complete working example.

1.13 Integration with SKOL

Use this extractor to prepare training data for SKOL classifier:

```
from pdf_section_extractor import PDFSectionExtractor
from skol_classifier.classifier_v2 import SkolClassifierV2

# Extract sections from PDF
extractor = PDFSectionExtractor()
sections = extractor.extract_from_document(
    database='skol_dev',
    doc_id='00df9554e9834283b5e844c7a994ba5f'
)

# Save as text file for annotation
with open('article_sections.txt', 'w') as f:
    for section in sections:
        f.write(section + '\n')

# After manual annotation, train SKOL classifier
classifier = SkolClassifierV2(
    spark=spark,
    input_source='files',
    file_paths=['article_sections.txt.ann'],
    model_type='rnn'
)
classifier.fit_and_save_model()
```

1.14 See Also

- **article_sections.py** - Example extracted sections
- **example_pdf_extraction.py** - Usage examples
- **CouchDB MCP Server** - CouchDB integration

Created: 2025-12-20 **File**: pdf_section_extractor.py **Status**: ✅ Tested and working