🏠 Modules Retrieval Document loaders PDF

PDF

> Portable Document Format (PDF), standardized as ISO 32000, is a file format developed by Adobe in 1992 to present documents, including text formatting and images, in a manner independent of application software, hardware, and operating systems.

This covers how to load `PDF` documents into the Document format that we use downstream.

Using PyPDF

Load PDF using `pypdf` into array of documents, where each document contains the page content and metadata with `page` number.

```
pip install pypdf
```

```
from langchain_community.document_loaders
import PyPDFLoader
```

```
loader = PyPDFLoader("example_data/layout-parser-paper.pdf")
pages = loader.load_and_split()
```

```
pages[0]
```

```
    Document(page_content='LayoutParser : A Uni\x0ced Toolkit for Deep\nLearning Based Document Image Analysis\nZejiang Shen1( \x00), Ruochen Zhang2, Melissa Dell3, Benjamin Charles Germain\nLee4, Jacob Carlson3, and Weining Li5\n1Allen Institute for AI\nshannons@allenai.org\n2Brown University\nruochen zhang@brown.edu\n3Harvard University\nfmelissadell,jacob carlson g@fas.harvard.edu\n4University of Washington\nbcgl@cs.washington.edu\n5University of Waterloo\nw422li@uwaterloo.ca\nAbstract. Recent advances in document image analysis (DIA) have been\nprimarily driven by the application of neural networks. Ideally, research\noutcomes could be easily deployed in production and extended for further\ninvestigation. However, various factors like loosely organized codebases\nand sophisticated model con\x0cgurations complicate the easy reuse of im-\nportant innovations by a wide audience. Though there
have been on-going\ne\x0borts to improve
reusability and simplify deep learning (DL)
model\ndevelopment in disciplines like natural
language processing and computer\nvision, none
of them are optimized for challenges in the
domain of DIA.\nThis represents a major gap in
the existing toolkit, as DIA is central
to\nacademic research across a wide range of
disciplines in the social sciences\nand
humanities. This paper introduces LayoutParser
, an open-source\nlibrary for streamlining the
usage of DL in DIA research and applica-
\ntions. The core LayoutParser library comes
with a set of simple and\nintuitive interfaces
for applying and customizing DL models for
layout de-\ntection, character recognition, and
many other document processing tasks.\nTo
promote extensibility, LayoutParser also
incorporates a community\nplatform for sharing
both pre-trained models and full document
digiti-\nzation pipelines. We demonstrate that
LayoutParser is helpful for both\nlightweight
and large-scale digitization pipelines in real-
word use cases.\nThe library is publicly
available at https://layout-parser.github.io
.\nKeywords: Document Image Analysis ·Deep
Learning ·Layout Analysis\n·Character
Recognition ·Open Source library ·Toolkit.\n1
Introduction\nDeep Learning(DL)-based
approaches are the state-of-the-art for a wide
range of\ndocument image analysis (DIA) tasks
```

```
including document image classi\x0ccation [
11,arXiv:2103.15348v2  [cs.CV]  21 Jun 2021',
metadata={'source': 'example_data/layout-
parser-paper.pdf', 'page': 0})
```

An advantage of this approach is that documents can be retrieved with page numbers.

We want to use `OpenAIEmbeddings` so we have to get the OpenAI API Key.

```
import os
import getpass

os.environ['OPENAI_API_KEY'] =
getpass.getpass('OpenAI API Key:')
```

```
    OpenAI API Key: ········
```

```
from langchain_community.vectorstores import
FAISS
from langchain_openai import OpenAIEmbeddings

faiss_index = FAISS.from_documents(pages,
OpenAIEmbeddings())
docs = faiss_index.similarity_search("How
will the community be engaged?", k=2)
```

```
for doc in docs:
    print(str(doc.metadata["page"]) + ":", doc.page_content[:300])
```

```
    9: 10 Z. Shen et al.
    Fig. 4: Illustration of (a) the original historical Japanese document with layout
    detection results and (b) a recreated version of the document image that achieves
    much better character recognition recall. The reorganization algorithm rearranges
    the tokens based on the their detect
    3: 4 Z. Shen et al.
    Efficient Data AnnotationC u s t o m i z e d  M o d e l  T r a i n i n gModel Cust omizationDI A Model HubDI A Pipeline SharingCommunity PlatformLa y out Detection ModelsDocument Images
    T h e  C o r e  L a y o u t P a r s e r  L i b r a r yOCR ModuleSt or age & VisualizationLa y ou
```

Extracting images

Using the `rapidocr-onnxruntime` package we can extract images as text as well:

```
pip install rapidocr-onnxruntime
```

```
loader =
PyPDFLoader("https://arxiv.org/pdf/2103.15348.p
extract_images=True)
pages = loader.load()
pages[4].page_content
```

```
'LayoutParser : A Unified Toolkit for DL-Based
DIA 5\nTable 1: Current layout detection
models in the LayoutParser model zoo\nDataset
Base Model1Large Model Notes\nPubLayNet [38]
F / M M Layouts of modern scientific
documents\nPRImA [3] M - Layouts of scanned
modern magazines and scientific
reports\nNewspaper [17] F - Layouts of
scanned US newspapers from the 20th
century\nTableBank [18] F F Table region on
modern scientific and business
document\nHJDataset [31] F / M - Layouts of
history Japanese documents\n1For each
dataset, we train several models of different
sizes for different needs (the trade-off
between accuracy\nvs. computational cost).
For “base model” and “large model”, we refer
to using the ResNet 50 or ResNet
101\nbackbones [ 13], respectively. One can
```

```
train models of different architectures, like
Faster R-CNN [ 28] (F) and Mask\nR-CNN [ 12]
(M). For example, an F in the Large Model
column indicates it has a Faster R-CNN model
trained\nusing the ResNet 101 backbone. The
platform is maintained and a number of
additions will be made to the model\nzoo in
coming months.\nlayout data structures ,
which are optimized for efficiency and
versatility. 3) When\nnecessary, users can
employ existing or customized OCR models via
the unified\nAPI provided in the OCR module .
4)LayoutParser comes with a set of
utility\nfunctions for the visualization and
storage of the layout data. 5)
LayoutParser\nis also highly customizable,
via its integration with functions for layout
data\nannotation and model training . We now
provide detailed descriptions for
each\ncomponent.\n3.1 Layout Detection
Models\nInLayoutParser , a layout model takes
a document image as an input and\ngenerates a
list of rectangular boxes for the target
content regions. Different\nfrom traditional
methods, it relies on deep convolutional
neural networks rather\nthan manually curated
rules to identify content regions. It is
formulated as an\nobject detection problem
and state-of-the-art models like Faster R-CNN
[ 28] and\nMask R-CNN [ 12] are used. This
yields prediction results of high accuracy
```

```
and\nmakes it possible to build a concise,
generalized interface for layout
detection.\nLayoutParser , built upon
Detectron2 [ 35], provides a minimal API that
can\nperform layout detection with only four
lines of code in Python:\n1import
layoutparser as lp\n2image = cv2. imread ("
image_file ") # load images\n3model = lp.
Detectron2LayoutModel (\n4 "lp :// PubLayNet
/ faster_rcnn_R_50_FPN_3x / config
")\n5layout = model . detect ( image
)\nLayoutParser provides a wealth of pre-
trained model weights using various\ndatasets
covering different languages, time periods,
and document types. Due to\ndomain shift [
7], the prediction performance can notably
drop when models are ap-\nplied to target
samples that are significantly different from
the training dataset. As\ndocument structures
and layouts vary greatly in different domains,
it is important\nto select models trained on
a dataset similar to the test samples. A
semantic syntax\nis used for initializing the
model weights in LayoutParser , using both
the dataset\nname and model name
lp://<dataset-name>/<model-architecture-name>
.'
```

Using MathPix

Inspired by Daniel Gross's https://gist.github.com/danielgross/3ab4104e14faccc12b49200843adab21

```
from langchain_community.document_loaders import MathpixPDFLoader
```

```
loader = MathpixPDFLoader("example_data/layout-parser-paper.pdf")
```

```
data = loader.load()
```

Using Unstructured

```
from langchain_community.document_loaders import UnstructuredPDFLoader
```

```
loader = UnstructuredPDFLoader("example_data/layout-parser-paper.pdf")
```

```
data = loader.load()
```

Retain Elements

Under the hood, Unstructured creates different "elements" for different chunks of text. By default we combine those together, but you can easily keep that separation by specifying `mode="elements"`.

```
loader = UnstructuredPDFLoader("example_data/layout-parser-paper.pdf", mode="elements")
```

```
data = loader.load()
```

```
data[0]
```

```
    Document(page_content='LayoutParser: A
Unified Toolkit for Deep\nLearning Based
Document Image Analysis\nZejiang Shen1 (�),
Ruochen Zhang2, Melissa Dell3, Benjamin
Charles Germain\nLee4, Jacob Carlson3, and
Weining Li5\n1 Allen Institute for
AI\nshannons@allenai.org\n2 Brown
```

```
University\nruochen zhang@brown.edu\n3
Harvard University\n{melissadell,jacob
carlson}@fas.harvard.edu\n4 University of
Washington\nbcgl@cs.washington.edu\n5
University of
Waterloo\nw422li@uwaterloo.ca\nAbstract.
Recent advances in document image analysis
(DIA) have been\nprimarily driven by the
application of neural networks. Ideally,
research\noutcomes could be easily deployed
in production and extended for
further\ninvestigation. However, various
factors like loosely organized codebases\nand
sophisticated model configurations complicate
the easy reuse of im-\nportant innovations by
a wide audience. Though there have been on-
going\nefforts to improve reusability and
simplify deep learning (DL)
model\ndevelopment in disciplines like
natural language processing and
computer\nvision, none of them are optimized
for challenges in the domain of DIA.\nThis
represents a major gap in the existing
toolkit, as DIA is central to\nacademic
research across a wide range of disciplines
in the social sciences\nand humanities. This
paper introduces LayoutParser, an open-
source\nlibrary for streamlining the usage of
DL in DIA research and applica-\ntions. The
core LayoutParser library comes with a set of
simple and\nintuitive interfaces for applying
```

```
and customizing DL models for layout de-
\ntection, character recognition, and many
other document processing tasks.\nTo promote
extensibility, LayoutParser also incorporates
a community\nplatform for sharing both pre-
trained models and full document digiti-
\nzation pipelines. We demonstrate that
LayoutParser is helpful for both\nlightweight
and large-scale digitization pipelines in
real-word use cases.\nThe library is publicly
available at https://layout-
parser.github.io.\nKeywords: Document Image
Analysis · Deep Learning · Layout Analysis\n·
Character Recognition · Open Source library ·
Toolkit.\n1\nIntroduction\nDeep Learning(DL)-
based approaches are the state-of-the-art for
a wide range of\ndocument image analysis
(DIA) tasks including document image
classification [11,\narXiv:2103.15348v2
[cs.CV]  21 Jun 2021\n', lookup_str='',
metadata={'file_path': 'example_data/layout-
parser-paper.pdf', 'page_number': 1,
'total_pages': 16, 'format': 'PDF 1.5',
'title': '', 'author': '', 'subject': '',
'keywords': '', 'creator': 'LaTeX with
hyperref', 'producer': 'pdfTeX-1.40.21',
'creationDate': 'D:20210622012710Z',
'modDate': 'D:20210622012710Z', 'trapped':
'', 'encryption': None}, lookup_index=0)
```

Fetching remote PDFs using Unstructured

This covers how to load online PDFs into a document format that we can use downstream. This can be used for various online PDF sites such as https://open.umn.edu/opentextbooks/textbooks/ and https://arxiv.org/archive/

Note: all other PDF loaders can also be used to fetch remote PDFs, but `OnlinePDFLoader` is a legacy function, and works specifically with `UnstructuredPDFLoader`.

```
from langchain_community.document_loaders
import OnlinePDFLoader
```

```
loader =
OnlinePDFLoader("https://arxiv.org/pdf/2302.038
```

```
data = loader.load()
```

```
print(data)
```

```
    [Document(page_content='A WEAK ( k, k ) -LE
TORIC ORBIFOLDS\n\nWilliam D. Montoya\n\nInstit
e Computa¸ c˜ao Cient´ıfica,\n\nIn [3] we proved
```

```
on a very general codimension s quasi- smooth i
projective toric orbifold P d Σ with d + s = 2
holds, that is, every ( p, p ) -cohomology clas
a rational linear combination of fundamental cl
of X . The proof of the above-mentioned result
Lefschetz\n\nKeywords: (1,1)- Lefschetz theorem
varieties, complete intersection Email: wmontoy
and the Hard Lefschetz theorem for projective o
s the proof relies on the Cayley trick, a trick
smooth hypersurface Y in a projective vector bu
(4.3) which gives an isomorphism of some primit
. The Cayley trick, following the philosophy of
results known for quasi-smooth hypersurfaces to
subvarieties. The idea in this paper goes the o
some results for quasi-smooth intersection subv
thank Prof. Ugo Bruzzo and Tiago Fonseca for us
acknowledge support from FAPESP postdoctoral gr
a free abelian group of rank d , let N = Hom (
.\n\nif there exist k linearly independent prim
∈ N such that σ = { μ\n\ne\n\n+ ⋯ + μ k e k }
integral if for every i and any nonnegative rat
is in N only if μ is an integer. • Given two ra
one says that σ ' is a face of σ ( σ ' < σ ) if
of σ ' is a subset of the set of integral gener
σ\n\n, . . . , σ t } of rational simplicial con
simplicial complete d -dimensional fan if:\n\na
;\n\nif σ, σ ' ∈ Σ then σ ∩ σ ' < σ and σ ∩ σ '
t .\n\nA rational simplicial complete d -dimens
dimensional toric variety P d Σ having only orb
assume to be projective. Moreover, T : = N ⊗ Z
action on P d Σ . We denote by Σ ( i ) the i -d
```

```
∈ Σ, ˆ σ is the set of 1-dimensional cone in Σ
σ\n\nand x ˆ σ : = ∏ ρ ∈ ˆ σ x ρ is the associa
2.2. The irrelevant ideal of P d Σ is the monom
> and the zero locus Z ( Σ ) : = V ( B Σ ) in t
is the irrelevant locus.\n\nProposition 2.3 (Th
variety P d Σ is a categorical quotient A d \ Z
) , C ∗ ) and the group action is induced by th
.\n\nNow we give a brief introduction to comple
needed theorems for the next section. Namely: d
theorem for complex orbifolds.\n\nDefinition 2.4
dimension d is a singular complex space whose s
isomorphic to quotient singularities C d / G ,
C ) .\n\nDefinition 2.5. A differential form on a
locally at z ∈ Z as a G -invariant differential
) and Z is locally isomorphic to d\n\nRoughly s
orbifolds reduces to local G -invariant geometr
differential forms ( A • ( Z ) , d ) and a doubl
∂ ) of bigraded differential forms which define t
cohomology groups (for a fixed p ∈ N ) respectiv
for projective toric orbifolds\n\nDefinition 3.1
quasi-smooth if V ( I X ) ⊂ A #Σ ( 1 ) is smoot
smooth hypersurfaces or more generally quasi-sm
3.2 . Quasi-smooth hypersurfaces or more genera
sub- varieties are quasi-smooth subvarieties (s
details).\n\nRemark 3.3 . Quasi-smooth subvarie
the sense of Satake in [8]. Intuitively speakin
only singularities come from the ambient\n\nPro
exact sequence\n\nwe have a long exact sequence
H 2 ( X, Z ) → H 2 (O X ) ≃ H 0 , 2 ( X )\n\nwh
to Steenbrink in [9]. Now, it is enough to prov
diagram\n\nwhere the last isomorphisms is due t
```

```
( X, Z ) / / H 2 ( X, O X ) ≃ Dolbeault H 2 ( X
/ H 0 , 2 ¯ ∂ ( X )\n\nof the proof follows as
in [6].\n\nRemark 3.5 . For k = 1 and P d Σ as
the classical ( 1 , 1 ) - Lefschetz theorem.\n\
for projective orbifolds (see [11] for details)
Theorem for projective orbifolds (see [11] for
cohomologies :\n\ngiven by the Lefschetz morphi
Hodge structures, we have:\n\nH 1 , 1 ( X, Q )
)\n\nCorollary 3.6. If the dimension of X is 1
holds on X\n\nProof. If the dim C X = 1 the res
Lefschetz theorem for projective orbifolds. The
covered by Theorem 3.5 and the Hard Lefschetz.\
proposition\n\nThe Cayley trick is a way to ass
intersection subvariety a quasi- smooth hypersu
line bundles on P d Σ and let π : P ( E ) → P d
associated to the vector bundle E = L 1 ⊕ ··· ⊕
is a ( d + s − 1 ) -dimensional simplicial tori
the degrees of the line bundles and the fan Σ.
without considering the grading, of P d Σ is C
Cox ring of P ( E ) is\n\nMoreover for X a quas
cut off by f 1 , . . . , f s with deg ( f i ) =
hypersurface Y cut off by F = y 1 f 1 + · · · + y
quasi-smooth. For more details see Section 2 in
as P d + s − 1 Σ ,X to keep track of its relati
following is a key remark.\n\nRemark 4.1 . There
s − 1 Σ ,X . Moreover every point z : = ( x, y
Hence for any subvariety W = V ( I W ) ⊂ X ⊂ P
s − 1 Σ ,X such that π ( W ' ) = W , i.e., W '
.\n\nFor X ⊂ P d Σ a quasi-smooth intersection
cohomology induced by the inclusion i ∗ : H d −
C ) is injective by Proposition 1.4 in [7].\n\n
cohomology of H d − s prim ( X ) is the quotien
s ( P d Σ , C )) and H d − s prim ( X, Q ) with
( P d Σ , C ) and H d − s ( X, C ) have pure Ho
i ∗ is com- patible with them, so that H d − s
structure.\n\nThe next Proposition is the Cayle
4.3. [Proposition 2.3 in [3] ] Let X = X 1 ∩· ·
intersec- tion subvariety in P d Σ cut off by ho
f s . Then for p ≠ d + s − 1 2 , d + s − 3 2\n\
isomorphisms are also true with rational coeffici
X, Q ) ⊗ Q C . See the beginning of Section 7.
details.\n\nTheorem 5.1. Let Y = { F = y 1 f 1
Σ ,X be the quasi-smooth hypersurface associate
intersection surface X = X f 1 ∩ · · · ∩ X f k ⊂
conjecture holds.\n\nthe Hodge conjecture holds
) = 0 we are done. So let us assume H k,k prim
proposition H k,k prim ( Y, Q ) ≃ H 1 , 1 prim
Lefschetz theorem for projective\n\ntoric orbif
algebraic basis λ C 1 , . . . , λ C n with rati
( X, Q ) , that is, there are n : = h 1 , 1 pri
. . . , C n in X such that under the Poincar´e
i ] goes to λ C i , [ C i ] ↦ λ C i . Recall t
contained in the Cox ring of P 2 k + 1 Σ ,X wit
Considering the grading we have that if α ∈ Cl
( P 2 k + 1 Σ ,X ) . So the polynomials defining
interpreted in P 2 k + 1 X, Σ but with different
each C i is contained in Y = { F = y 1 f 1 + ···
and\n\nfurthermore it has codimension k .\n\nCl
prim ( ) . It is enough to prove that λ C i is
( Y, Q ) or equivalently that the cohomology cl
come from the ambient space. By contradiction,
j and C ⊂ P 2 k + 1 Σ ,X such that λ C ∈ H k,k
```

```
( λ C ) = λ C j or in terms of homology there e
algebraic subvariety V ⊂ P 2 k + 1 Σ ,X such th
as a homology class of P 2 k + 1 Σ ,X ,i.e., [ 
check that π ( V ) ∩ X = C j as a subvariety of
x . Hence [ π ( V ) ∩ X ] = [ C j ] which is eq
from P k + 2 Σ which contradicts the choice of 
the proof of the previous theorem, the key fact
conjecture holds and we translate it to Y by co
analogous argument we have:\n\nargument we have
= y 1 f s +⋯+ y s f s = 0 } ⊂ P 2 k + 1 Σ ,X be
associated to a quasi-smooth intersection subva
P d Σ such that d + s = 2 ( k + 1 ) . If the Ho
holds as well on Y .\n\nCorollary 5.4. If the d
2 s + 1 then the Hodge conjecture holds on Y .\
Corollary 3.6.\n\n[\n\n] Angella, D. Cohomologi
of Geometry and Physics\n\n(\n\n),\n\n–\n\n[\n\
On the Hodge structure of projective hypersur- 
Mathematical Journal\n\n,\n\n(Aug\n\n). [\n\n] 
the Hodge conjecture for quasi-smooth in- terse
Paulo J. Math. Sci. Special Section: Geometry i
(\n\n). [\n\n] Caramello Jr, F. C. Introduction
a\n\niv:\n\nv\n\n(\n\n). [\n\n] Cox, D., Little
varieties, vol.\n\nAmerican Math- ematical Soc.
Harris, J. Principles of Algebraic Geometry. Jo
Mavlyutov, A. R. Cohomology of complete interse
lished in Pacific J. of Math.\n\nNo.\n\n(\n\n),\
Generalization of the Notion of Manifold. Proce
Sciences of the United States of America\n\n,\n
Steenbrink, J. H. M. Intersection form for quas
positio Mathematica\n\n,\n\n(\n\n),\n\n–\n\n[\n
Complex Algebraic Geometry I, vol.\n\nof Cambri
```

```
Mathematics . Cambridge University Press,\n\n[\
A remark on the Hard Lefschetz theorem for K¨ah
American Mathematical Society\n\n,\n\n(Aug\n\n)
D. A. On the Hodge structure of projective hype
Duke Mathematical Journal 75, 2 (Aug 1994).\n\n
W. On the Hodge conjecture for quasi-smooth in-
S˜ao Paulo J. Math. Sci. Special Section: Geome
Geometry (\n\n).\n\n[3] Bruzzo, U., and Montoya
quasi-smooth in- tersections in toric varieties
Special Section: Geometry in Algebra and Algebr
Cohomology of complete intersections in toric v
metadata={'source':
'/var/folders/ph/hhm7_zyx4l13k3v8z02dwp1w0000gn
lookup_index=0)]
```

Using PyPDFium2

```python
from langchain_community.document_loaders
import PyPDFium2Loader
```

```python
loader =
PyPDFium2Loader("example_data/layout-parser-
paper.pdf")
```

```python
data = loader.load()
```

Using PDFMiner

```
from langchain_community.document_loaders import PDFMinerLoader
```

```
loader = PDFMinerLoader("example_data/layout-parser-paper.pdf")
```

```
data = loader.load()
```

Using PDFMiner to generate HTML text

This can be helpful for chunking texts semantically into sections as the output html content can be parsed via `BeautifulSoup` to get more structured and rich information about font size, page numbers, PDF headers/footers, etc.

```
from langchain_community.document_loaders import PDFMinerPDFasHTMLLoader
```

```
loader = PDFMinerPDFasHTMLLoader("example_data/layout-
parser-paper.pdf")
```

```
data = loader.load()[0]   # entire PDF is loaded as a single Document
```

```
from bs4 import BeautifulSoup
soup = BeautifulSoup(data.page_content,'html.parser')
content = soup.find_all('div')
```

```
import re
cur_fs = None
cur_text = ''
snippets = []   # first collect all snippets that have the same font size
for c in content:
    sp = c.find('span')
    if not sp:
        continue
    st = sp.get('style')
    if not st:
        continue
    fs = re.findall('font-size:(\d+)px',st)
    if not fs:
        continue
    fs = int(fs[0])
    if not cur_fs:
        cur_fs = fs
    if fs == cur_fs:
        cur_text += c.text
    else:
        snippets.append((cur_text,cur_fs))
        cur_fs = fs
        cur_text = c.text
snippets.append((cur_text,cur_fs))
# Note: The above logic is very
straightforward. One can also add more
strategies such as removing duplicate
snippets (as
# headers/footers in a PDF appear on multiple
pages so if we find duplicates it's safe to
assume that it is redundant info)
```

```
from langchain.docstore.document import Documen
cur_idx = -1
semantic_snippets = []
# Assumption: headings have higher font size th
content
for s in snippets:
    # if current snippet's font size > previous
it is a new heading
    if not semantic_snippets or s[1] >
semantic_snippets[cur_idx].metadata['heading_fo
        metadata={'heading':s[0], 'content_font
s[1]}
        metadata.update(data.metadata)
```

```
semantic_snippets.append(Document(page_content=
        cur_idx += 1
        continue

    # if current snippet's font size <= previou
content belongs to the same section (one can al
    # a tree like structure for sub sections if
require some more thinking and may be data spec
    if not semantic_snippets[cur_idx].metadata[
s[1] <= semantic_snippets[cur_idx].metadata['co
        semantic_snippets[cur_idx].page_content
        semantic_snippets[cur_idx].metadata['co
max(s[1], semantic_snippets[cur_idx].metadata['
        continue

    # if current snippet's font size > previous
less than previous section's heading than also 
    # section (e.g. title of a PDF will have th
but we don't want it to subsume all sections)
    metadata={'heading':s[0], 'content_font': 0
s[1]}
    metadata.update(data.metadata)

semantic_snippets.append(Document(page_content=
    cur_idx += 1
```

```
semantic_snippets[4]
```

```
    Document(page_content='Recently, various DL
models and datasets have been developed for
layout analysis\ntasks. The dhSegment [22]
utilizes fully convolutional networks [20] for
segmen-\ntation tasks on historical documents.
Object detection-based methods like Faster\nR-C
[28] and Mask R-CNN [12] are used for identifyi
document elements [38]\nand detecting tables [3
26]. Most recently, Graph Neural Networks [29]
have also\nbeen used in table detection [27].
However, these models are usually
implemented\nindividually and there is no unifie
framework to load and use such models.\nThere h
been a surge of interest in creating open-sourc
tools for document\nimage processing: a search
document image analysis in Github leads to
5M\nrelevant code pieces 6; yet most of them re
on traditional rule-based methods\nor provide
limited functionalities. The closest prior
research to our work is the\nOCR-D project7,
which also tries to build a complete toolkit fo
DIA. However,\nsimilar to the platform develope
by Neudecker et al. [21], it is designed
for\nanalyzing historical documents, and provid
no supports for recent DL models.\nThe
DocumentLayoutAnalysis project8 focuses on
processing born-digital PDF\ndocuments via
analyzing the stored PDF data. Repositories lik
DeepLayout9\nand Detectron2-PubLayNet10 are
individual deep learning models trained
```

```
on\nlayout analysis datasets without support fo
the full DIA pipeline. The Document\nAnalysis a
Exploitation (DAE) platform [15] and the DeepDI
project [2]\naim to improve the reproducibility
of DIA methods (or DL models), yet they\nare no
actively maintained. OCR engines like Tesseract
[14], easyOCR11 and\npaddleOCR12 usually do not
come with comprehensive functionalities for
other\nDIA tasks like layout analysis.\nRecent
years have also seen numerous efforts to create
libraries for promoting\nreproducibility and
reusability in the field of DL. Libraries like
Dectectron2 [35],\n6 The number shown is obtain
by specifying the search type as ‘code’.\n7
https://ocr-d.de/en/about\n8
https://github.com/BobLd/DocumentLayoutAnalysis
9 https://github.com/leonlulu/DeepLayout\n10
https://github.com/hpanwar08/detectron2\n11
https://github.com/JaidedAI/EasyOCR\n12
https://github.com/PaddlePaddle/PaddleOCR\n4\nZ
Shen et al.\nFig. 1: The overall architecture o
LayoutParser. For an input document image,\nthe
core LayoutParser library provides a set of off-
the-shelf tools for layout\ndetection, OCR,
visualization, and storage, backed by a careful
designed layout\ndata structure. LayoutParser
also supports high level customization via
efficient\nlayout annotation and model training
functions. These improve model accuracy\non the
target samples. The community platform enables
the easy sharing of DIA\nmodels and whole
```

```
digitization pipelines to promote reusability a
reproducibility.\nA collection of detailed
documentation, tutorials and exemplar projects
make\nLayoutParser easy to learn and
use.\nAllenNLP [8] and transformers [34] have
provided the community with complete\nDL-based
support for developing and deploying models for
general computer\nvision and natural language
processing problems. LayoutParser, on the
other\nhand, specializes specifically in DIA
tasks. LayoutParser is also equipped with
a\ncommunity platform inspired by established
model hubs such as Torch Hub [23]\nand TensorFl
Hub [1]. It enables the sharing of pretrained
models as well as\nfull document processing
pipelines that are unique to DIA tasks.\nThere
have been a variety of document data collection
to facilitate the\ndevelopment of DL models. So
examples include PRImA [3](magazine
layouts),\nPubLayNet [38](academic paper
layouts), Table Bank [18](tables in
academic\npapers), Newspaper Navigator Dataset
[16, 17](newspaper figure layouts) and\nHJDatase
[31](historical Japanese document layouts). A
spectrum of models\ntrained on these datasets a
currently available in the LayoutParser model
zoo\nto support different use cases.\n', metadat
{'heading': '2 Related Work\n', 'content_font':
9, 'heading_font': 11, 'source':
'example_data/layout-parser-paper.pdf'})
```

Using PyMuPDF

This is the fastest of the PDF parsing options, and contains detailed metadata about the PDF and its pages, as well as returns one document per page.

```
from langchain_community.document_loaders import PyMuPDFLoader
```

```
loader = PyMuPDFLoader("example_data/layout-parser-paper.pdf")
```

```
data = loader.load()
```

```
data[0]
```

```
    Document(page_content='LayoutParser: A Unified Toolkit for Deep\nLearning Based Document Image Analysis\nZejiang Shen1 (�), Ruochen Zhang2, Melissa Dell3, Benjamin Charles Germain\nLee4, Jacob Carlson3, and Weining Li5\n1 Allen Institute for AI\nshannons@allenai.org\n2 Brown
University\nruochen zhang@brown.edu\n3
Harvard University\n{melissadell,jacob
carlson}@fas.harvard.edu\n4 University of
Washington\nbcgl@cs.washington.edu\n5
University of
Waterloo\nw422li@uwaterloo.ca\nAbstract.
Recent advances in document image analysis
(DIA) have been\nprimarily driven by the
application of neural networks. Ideally,
research\noutcomes could be easily deployed
in production and extended for
further\ninvestigation. However, various
factors like loosely organized codebases\nand
sophisticated model configurations complicate
the easy reuse of im-\nportant innovations by
a wide audience. Though there have been on-
going\nefforts to improve reusability and
simplify deep learning (DL)
model\ndevelopment in disciplines like
natural language processing and
computer\nvision, none of them are optimized
for challenges in the domain of DIA.\nThis
represents a major gap in the existing
toolkit, as DIA is central to\nacademic
research across a wide range of disciplines
in the social sciences\nand humanities. This
paper introduces LayoutParser, an open-
source\nlibrary for streamlining the usage of
DL in DIA research and applica-\ntions. The
core LayoutParser library comes with a set of
simple and\nintuitive interfaces for applying
```

```
and customizing DL models for layout de-
\ntection, character recognition, and many
other document processing tasks.\nTo promote
extensibility, LayoutParser also incorporates
a community\nplatform for sharing both pre-
trained models and full document digiti-
\nzation pipelines. We demonstrate that
LayoutParser is helpful for both\nlightweight
and large-scale digitization pipelines in
real-word use cases.\nThe library is publicly
available at https://layout-
parser.github.io.\nKeywords: Document Image
Analysis · Deep Learning · Layout Analysis\n·
Character Recognition · Open Source library ·
Toolkit.\n1\nIntroduction\nDeep Learning(DL)-
based approaches are the state-of-the-art for
a wide range of\ndocument image analysis
(DIA) tasks including document image
classification [11,\narXiv:2103.15348v2
[cs.CV]  21 Jun 2021\n', lookup_str='',
metadata={'file_path': 'example_data/layout-
parser-paper.pdf', 'page_number': 1,
'total_pages': 16, 'format': 'PDF 1.5',
'title': '', 'author': '', 'subject': '',
'keywords': '', 'creator': 'LaTeX with
hyperref', 'producer': 'pdfTeX-1.40.21',
'creationDate': 'D:20210622012710Z',
'modDate': 'D:20210622012710Z', 'trapped':
'', 'encryption': None}, lookup_index=0)
```

Additionally, you can pass along any of the options from the PyMuPDF documentation as keyword arguments in the `load` call, and it will be pass along to the `get_text()` call.

PyPDF Directory

Load PDFs from directory

```
from langchain_community.document_loaders
import PyPDFDirectoryLoader
```

```
loader =
PyPDFDirectoryLoader("example_data/")
```

```
docs = loader.load()
```

Using PDFPlumber

Like PyMuPDF, the output Documents contain detailed metadata about the PDF and its pages, and returns one document per page.

```
from langchain_community.document_loaders
import PDFPlumberLoader
```

```
loader =
PDFPlumberLoader("example_data/layout-parser-
paper.pdf")
```

```
data = loader.load()
```

```
data[0]
```

```
    Document(page_content='LayoutParser: A Unif
Document Image Analysis\nZejiang Shen1 ((cid:0)
Benjamin Charles Germain\nLee4, Jacob Carlson3,
AI\n1202 shannons@allenai.org\n2 Brown Universi
University\nnuJ {melissadell,jacob carlson}@fas
Washington\nbcgl@cs.washington.edu\n12 5 Univer
Waterloo\nw422li@uwaterloo.ca\n]VC.sc[\nAbstrac
Recentadvancesindocumentimageanalysis(DIA)haveb
of neural networks. Ideally,
research\noutcomescouldbeeasilydeployedinproduc
However, various factors like loosely organized
configurations complicate the easy reuse of im-
portantinnovationsbyawideaudience.Thoughtthereha
reusability and simplify deep learning (DL)
model\ndevelopmentindisciplineslikenaturallangu
them are optimized for challenges in the domain
the existing toolkit, as DIA is central to\naca
disciplinesinthesocialsciences\nand humanities.
open-source\nlibrary for streamlining the usage
\ntions. The core LayoutParser library comes wi
and\nintuitiveinterfacesforapplyingandcustomizi
\ntection,characterrecognition,andmanyotherdocu
extensibility, LayoutParser also incorporates a
pre-trained models and full document digiti-\nz
LayoutParser is helpful for both\nlightweight a
real-word use cases.\nThe library is publicly a
parser.github.io.\nKeywords: DocumentImageAnaly
Character Recognition · Open Source library · T
Learning(DL)-based approaches are the state-of-
of\ndocumentimageanalysis(DIA)tasksincludingdoc
{'source': 'example_data/layout-parser-paper.pd
parser-paper.pdf', 'page': 1, 'total_pages': 16
'D:20210622012710Z', 'Creator': 'LaTeX with hyp
'D:20210622012710Z', 'PTEX.Fullbanner': 'This i
(TeX Live 2020) kpathsea version 6.3.2', 'Produ
'Title': '', 'Trapped': 'False'})
```

Using AmazonTextractPDFParser

The AmazonTextractPDFLoader calls the Amazon Textract Service to convert PDFs into a Document structure. The loader does pure OCR at the moment, with more features like layout support planned, depending on demand. Single and multi-page documents are supported with up to 3000 pages and 512 MB of size.

For the call to be successful an AWS account is required, similar to the AWS CLI requirements.

Besides the AWS configuration, it is very similar to the other PDF loaders, while also supporting JPEG, PNG and TIFF and non-native PDF formats.

```
from langchain_community.document_loaders impor
AmazonTextractPDFLoader
loader =
AmazonTextractPDFLoader("example_data/alejandro_
small.jpeg")
documents = loader.load()
```