



# 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.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\nfmmelissadell,jacob carlson  
g@fas.harvard.edu\n4University of  
Washington\nnbcgl@cs.washington.edu\n5University of  
Waterloo\nnw422li@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 investigation. However, various factors like loosely organized codebases and sophisticated model configurations complicate the easy reuse of important innovations by a wide audience. Though there have been on-going efforts to improve reusability and simplify deep learning (DL) model development in disciplines like natural language processing and computer vision, none of them are optimized for challenges in the domain of DIA. This represents a major gap in the existing toolkit, as DIA is central to academic research across a wide range of disciplines in the social sciences and humanities. This paper introduces LayoutParser, an open-source library for streamlining the usage of DL in DIA research and applications. The core LayoutParser library comes with a set of simple and intuitive interfaces for applying and customizing DL models for layout detection, character recognition, and many other document processing tasks. To promote extensibility, LayoutParser also incorporates a community platform for sharing both pre-trained models and full document digitization pipelines. We demonstrate that LayoutParser is helpful for both lightweight and large-scale digitization pipelines in real-world use cases. The library is publicly available at <https://layout-parser.github.io>.

**Keywords:** Document Image Analysis · Deep Learning · Layout Analysis · Character Recognition · Open Source library · Toolkit.

**Introduction**

Deep Learning (DL)-based approaches are the state-of-the-art for a wide range of document image analysis (DIA) tasks including document image classification [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.vectorstores import FAISS
from langchain.embeddings.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 g  
Model 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

## Using MathPix

---

Inspired by Daniel Gross's

<https://gist.github.com/danielgross/3ab4104e14facc12b49200843adab21>

```
from langchain.document_loaders import MathpixPDFLoader
```

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

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

## Using Unstructured

---

```
from langchain.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\nnbcgl@cs.washington.edu\n5 University of  
Waterloo\nnw422li@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-  
portant innovations by a  
wide audience. Though there have been on-going  
efforts to improve  
reusability and simplify deep learning (DL) model  
development in  
disciplines like natural language processing and computer\nvision, none of
```

them are optimized for challenges in the domain of DIA. This represents a major gap in the existing toolkit, as DIA is central to academic research across a wide range of disciplines in the social sciences and humanities. This paper introduces LayoutParser, an open-source library for streamlining the usage of DL in DIA research and applications. The core LayoutParser library comes with a set of simple and intuitive interfaces for applying and customizing DL models for layout detection, character recognition, and many other document processing tasks. To promote extensibility, LayoutParser also incorporates a community platform for sharing both pre-trained models and full document digitization pipelines. We demonstrate that LayoutParser is helpful for both lightweight and large-scale digitization pipelines in real-world use cases. The library is publicly available at <https://layout-parser.github.io>.

**Keywords:** Document Image Analysis · Deep Learning · Layout Analysis · Character Recognition · Open Source library · Toolkit.

**1 Introduction**

Deep Learning (DL)-based approaches are the state-of-the-art for a wide range of document image analysis (DIA) tasks including document image classification [11, arXiv:2103.15348v2 [cs.CV] 21 Jun 2021], 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.document_loaders import OnlinePDFLoader
```

```
loader = OnlinePDFLoader("https://arxiv.org/pdf/2302.03803.pdf")
```

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

```
print(data)
```

[Document(page\_content='A WEAK  $(k, k)$ -LEFSCHETZ THEOREM FOR PROJECTIVE TORIC ORBIFOLDS  
William D. Montoya  
Instituto de Matemática, Estatística e Computação Científica,  
In [3] we proved that, under suitable condition a very general codimension  $s$  quasi-smooth intersection subvariety  $X$  in a projective toric orbifold  $P^d_\Sigma$  with  $d + s = 2(k + 1)$  the Hodge conjecture holds, that is, every  $(p, p)$ -cohomology class, under the Poincaré duality is a rational linear combination of fundamental classes of algebraic subvarieties of  $X$ . The proof of the above-mentioned result relies, for  $p \neq 1 - s$ , on a Lefschetz  
Keywords:  $(1,1)$ -Lefschetz theorem, Hodge conjecture toric varieties, complete intersection Email:

wmontoya@ime.unicamp.br  
theorem ([7]) and the Hard Lefschetz theorem for projective orbifolds ([11]). When  $p = d + 1 - s$  the proof relies on the Cayley trick, a trick which associates to  $X$  a quasi-smooth hypersurface  $Y$  in a projective vector bundle, and the Cayley Proposition (4.3) which gives an isomorphism of some primitive cohomologies (4.2) of  $X$  and  $Y$ . The Cayley trick following the philosophy of Mavlyutov in [7], reduces results known for quasi-smooth hypersurfaces to quasi-smooth intersection subvarieties. The idea in this paper goes the other way around, we translate some results for quasi-smooth intersection subvarieties to  
Acknowledgement. I thank Prof. Ugo Bruzzo and Tiago Fonseca for useful discussions. I also acknowledge support from FAPESP postdoctoral grant No. 2019/23499-7.  
Let  $M$  be a free abelian group of rank  $d$ , let  $N = \text{Hom}(M, \mathbb{Z})$ , and  $N_R = N \otimes \mathbb{Z}_R$ .  
if there exist  $k$  linearly independent primitive elements  $e_1, \dots, e_k \in N$  such that  $\sigma = \{\mu_1 e_1 + \dots + \mu_k e_k\}$ .  
• The generators  $e_i$  are integral if for every  $i$  and any nonnegative rational number  $\mu$  the product  $\mu e_i$  is in  $N$  only if  $\mu$  is integer.  
• Given two rational simplicial cones  $\sigma, \sigma'$  one says that  $\sigma'$  is face of  $\sigma$  ( $\sigma' < \sigma$ ) if the set of integral generators of  $\sigma'$  is a subset of the set of integral generators of  $\sigma$ .  
• A finite set  $\Sigma = \{\sigma_1, \dots, \sigma_t\}$  of rational simplicial cones is called a rational simplicial complete  $d$ -dimensional fan if:  
all faces of cones in  $\Sigma$  are in  $\Sigma$ ;  
if  $\sigma, \sigma' \in \Sigma$  then  $\sigma \cap \sigma' < \sigma$  and  $\sigma \cap \sigma' < \sigma'$ ;  
 $N_R = \sigma_1 \cup \dots \cup \sigma_t$ .  
A rational simplicial complete  $d$ -dimensional fan  $\Sigma$  defines a  $d$ -dimensional toric variety  $P^d_\Sigma$  having only orbifold singularities which we assume to be projective. Moreover,  $T := N \otimes \mathbb{Z}_C \cong (C^*)^d$  is the torus action on  $P^d_\Sigma$ . We denote by  $\Sigma(i)$  the  $i$ -dimensional cones  
For a cone  $\sigma \in \Sigma$ ,  $\hat{\sigma}$  is the set of 1-dimensional cone in  $\Sigma$  that are not contained in  $\sigma$  and  $x^{\hat{\sigma}} := \prod_{\rho \in \hat{\sigma}} x_\rho$  is the associated monomial in  $S$ .  
Definition 2.2. The irrelevant ideal  $\mathfrak{s}$

$B_\Sigma$  is the monomial ideal  $B_\Sigma := \langle x^\sigma \mid \sigma \in \Sigma \rangle$  and the zero locus  $Z(\Sigma) = V(B_\Sigma)$  in the affine space  $A^d := \text{Spec}(S)$  is the irrelevant locus.

**Proposition 2.3** (Theorem 5.1.11 [5]). The toric variety  $P^d_\Sigma$  is categorical quotient  $A^d \setminus Z(\Sigma)$  by the group  $\text{Hom}(\text{Cl}(\Sigma), \mathbb{C}^*)$  and the group action is induced by the  $\text{Cl}(\Sigma)$ -grading of  $S$ .

Now we give a brief introduction to complex orbifolds and we mention the needed theorems for the next section. Namely: de Rham theorem and Dolbeault theorem for complex orbifolds.

**Definition 2.4.** A complex orbifold of complex dimension  $d$  is a singular complex space whose singularities are locally isomorphic to quotient singularities  $\mathbb{C}^d / G$ , for finite subgroups  $G \subset \text{GL}(d, \mathbb{C})$ .

**Definition 2.5.** A differential form on a complex orbifold  $Z$  is defined locally at  $z \in Z$  as a  $G$ -invariant differential form on  $\mathbb{C}^d$  where  $G \subset \text{GL}(d, \mathbb{C})$  and  $Z$  is locally isomorphic to  $\mathbb{C}^d / G$ .

Roughly speaking the local geometry of orbifolds reduces to local  $G$ -invariant geometry.

We have a complex of differential forms  $(A^\bullet(Z), d)$  and a double complex  $(A^\bullet, \bullet(Z), \partial, \bar{\partial})$  of bigraded differential forms which define the de Rham and the Dolbeault cohomology group (for a fixed  $p \in \mathbb{N}$ ) respectively:

**(1,1)-Lefschetz theorem for projective toric orbifolds**

**Definition 3.1.** A subvariety  $X \subset P^d_\Sigma$  is quasi-smooth if  $I_X \subset A_\Sigma(1)$  is smooth outside

**Example 3.2.** Quasi-smooth hypersurfaces or more generally quasi-smooth intersection subvarieties are quasi-smooth subvarieties (see [2] or [7] for more details).

**Remark 3.3.** Quasi-smooth subvarieties are suborbifolds of  $P^d_\Sigma$  in the sense of Satake in [8]. Intuitively speaking they are subvarieties whose only singularities come from the ambient.

**Proof.** From the exponential short exact sequence we have a long exact sequence in cohomology

$$H^2(X, \mathbb{Z}) \rightarrow H^2(0_X) \simeq H^{0,2}(X) \rightarrow H^1(X, \mathbb{Z}) \rightarrow H^1(0_X) \simeq H^{0,1}(X) \rightarrow \dots$$

where the last isomorphism is due to Steenbrink in [9]. Now, it is enough to prove the commutativity of the next diagram

$$\begin{array}{ccc} H^2(X, \mathbb{Z}) & \rightarrow & H^2(0_X) \\ \downarrow & & \downarrow \\ H^2(X, \mathbb{C}) & \rightarrow & H^{0,2}(X) \end{array}$$

where the last isomorphism is due to Steenbrink in [9]. Now,

$$H^2(X, \mathbb{Z}) / \text{torsion} \simeq H^2(X, \mathbb{C}) \simeq \text{Dolbeault } H^2(X, \mathbb{C}) \simeq H^2_{\text{dR}}(X, \mathbb{C}) / H^{0,2}(X)$$

of the proof follows as the  $(1,1)$ -Lefschetz theorem in [6].

**Remark 3.5.** For  $k=1$  and  $P^d_\Sigma$  as the projective space, we recover the classical  $(1,1)$ -Lefschetz theorem.

By the Hard Lefschetz Theorem for projective orbifolds (see [11] for details) we get an isomorphism of cohomologies:

$$H^{1,1}(X, \mathbb{Q}) \simeq H^{\dim X - 1, \dim X - 1}(X, \mathbb{Q})$$

given by the Lefschetz morphism and since it is a morphism of Hodge structures, we have:

**Corollary 3.6.** If the dimension of  $X$  is 1, 2 or 3. The Hodge conjecture holds on  $X$ .

**Proof.** If the  $\dim_{\mathbb{C}} X = 1$  the result is clear by the Hard Lefschetz theorem for projective orbifolds. The dimension 2 and 3 cases are covered by Theorem 3.5 and the Hard Lefschetz.

**Cayley trick and Cayley proposition**

The Cayley trick is a way to associate to a quasi-smooth intersection subvariety a quasi-smooth hypersurface. Let  $L_1, \dots, L_s$  line bundles on  $P^d_\Sigma$  and let  $\pi: P(E) \rightarrow P^d_\Sigma$  be the projective space

bundle associated to the vector bundle  $E = L_1 \oplus \dots \oplus L_s$ . It is known that  $E$  is a  $(d + s - 1)$ -dimensional simplicial toric variety whose fan depends on the degrees of the line bundles and the fan  $\Sigma$ . Furthermore, if the Cox ring without considering the grading, of  $P(d, \Sigma)$  is  $C[x_1, \dots, x_m]$  then the Cox ring of  $P(E)$  is

Moreover for  $X$  a quasi-smooth intersection subvariety cut off by  $f_1, \dots, f_s$  with  $\deg(f_i) = [L_i]$  we relate the hypersurface  $Y$  cut off by  $F = y_1 f_1 + \dots + y_s f_s$  which turns out to be quasi-smooth. For more details see Section 2 in [7].

We will denote  $P(E)$  as  $P(d + s - 1, \Sigma, X)$  to keep track of its relation with  $X$  and  $P(d, \Sigma)$ .

The following is a key remark.

**Remark 4.1.** There is a morphism  $\iota : X \rightarrow Y \subset P(d + s - 1, \Sigma, X)$ . Moreover every point  $z := (x, y) \in Y$  with  $y \neq 0$  has a preimage. Hence for any subvariety  $W = V(I_W) \subset X \subset P(d, \Sigma)$  there exists  $W' \subset Y \subset P(d + s - 1, \Sigma, X)$  such that  $\pi(W') = W$ , i.e.,  $W' = \{z = (x, y) \mid (x, 0) \in W\}$ .

For  $X \subset P(d, \Sigma)$  a quasi-smooth intersection variety the morphism in cohomology induced by the inclusion  $i_* : H^{d-s}(P(d, \Sigma), C) \rightarrow H^{d-s}(X, C)$  is injective by Proposition 1.4 in [7].

**Definition 4.2.** The primitive cohomology of  $H^{d-s}_{\text{prim}}(X)$  is the quotient  $H^{d-s}(X, C) / i_*(H^{d-s}(P(d, \Sigma), C))$  and  $H^{d-s}_{\text{prim}}(X, Q)$  with rational coefficients.

$H^{d-s}(P(d, \Sigma), C)$  and  $H^{d-s}(X, C)$  have pure Hodge structures, and the morphism  $i_*$  is compatible with them, so that  $H^{d-s}_{\text{prim}}(X)$  gets a pure Hodge structure.

The next Proposition is the Cayley proposition.

**Proposition 4.3.** [Proposition 2.3 in [3]] Let  $X = X_1 \cap \dots \cap X_s$  be a quasi-smooth intersection subvariety in  $P(d, \Sigma)$  cut off by homogeneous polynomials  $f_1, \dots, f_s$ . Then for  $p \neq d + s - 1/2, d + s - 3/2$

**Remark 4.5.** The above isomorphisms are also true with rational coefficients since  $H^k(X, C) = H^k(X, Q) \otimes Q \subset C$ . See the beginning of Section 7.1 in [10] for more details.

**Theorem 5.1.** Let  $Y = \{F = y_1 f_1 + \dots + y_k f_k = 0\} \subset P(2k + 1, \Sigma, X)$  be the quasi-smooth hypersurface associated to the quasi-smooth intersection surface  $X = X_{f_1} \cap \dots \cap X_{f_k} \subset P(k + 2, \Sigma)$ . Then on  $Y$  the Hodge conjecture holds.

**Proof.** If  $H^{k,k}_{\text{prim}}(X, Q) = 0$  we are done. So let us assume  $H^{k,k}_{\text{prim}}(X, Q) \neq 0$ . By the Cayley proposition  $H^{k,k}_{\text{prim}}(Y, Q) \simeq H^{1,1}_{\text{prim}}(X, Q)$  and by the  $(1)$ -Lefschetz theorem for projective toric orbifolds there is a non-zero algebraic basis  $\lambda_1 \in H^{1,1}_{\text{prim}}(X, Q)$ ,  $\dots$ ,  $\lambda_n \in H^{1,1}_{\text{prim}}(X, Q)$  with rational coefficients of  $H^{1,1}_{\text{prim}}(X, Q)$ , that is, there are  $n := h^{1,1}_{\text{prim}}(X, Q)$  algebraic curves  $C_1, \dots, C_n$  in  $X$  such that under the Poincaré duality the class in  $\text{hom}([C_i])$  goes to  $\lambda_i$ ,  $[C_i] \mapsto \lambda_i$ . Recall that the Cox ring of  $P(k + 2, \Sigma)$  is contained in the Cox ring of  $P(2k + 1, \Sigma, X)$  without considering the grading. Considering the grading we have that if  $\alpha \in \text{Cl}(P(k + 2, \Sigma))$  then  $(\alpha, 0) \in (P(2k + 1, \Sigma, X))$ . So the polynomials defining  $C_i \subset P(k + 2, \Sigma)$  can be interpreted in  $P(2k + 1, \Sigma, X)$  but with different degree. Moreover, by Remark 4.5 each  $C_i$  is contained in  $Y = \{F = y_1 f_1 + \dots + y_k f_k = 0\}$  and furthermore it has codimension  $k$ .

**Claim:**  $\{C_i\}_{i=1}^n$  is a basis of  $H^{k,k}_{\text{prim}}(Y)$ . It is enough to prove that  $\lambda_i$  is different from zero in  $H^{k,k}_{\text{prim}}(Y)$ .



$\text{prim}(Y, Q)$  or equivalently that the cohomology classes  $\{\lambda C_i\}_{n_i=1}^d$  do not come from the ambient space. By contradiction, let us assume that there exists a  $j$  and  $C \in P^{2k+1}(\Sigma, X)$  such that  $\lambda C \in H^{k,k}(P^{2k+1}(\Sigma, X), Q)$  with  $i_*(\lambda C) = \lambda C_j$  or in terms of homology there exists a  $(k+2)$ -dimensional algebraic subvariety  $V \subset P^{2k+1}(\Sigma, X)$  such that  $V \cap Y = C_j$  so they are equal as a homology class of  $P^{2k+1}(\Sigma, X)$ , i.e.,  $[V \cap Y] = [C]$ . It is easy to check that  $\pi(V) \cap X = C_j$  as a subvariety of  $P^{k+2}(\Sigma)$  where  $\pi: (x, y) \mapsto x$ . Hence  $[\pi(V) \cap X] = [C_j]$  which is equivalent to say that  $\lambda C_j$  comes from  $P^{k+2}(\Sigma)$  which contradicts the choice of  $[C_j]$ .

**Remark 5.2.** Into the proof of the previous theorem, the key fact was that on  $X$  the Hodge conjecture holds and we translate it to  $Y$  by contradiction. So using an analogous argument we have:

**Proposition 5.3** Let  $Y = \{F = y_1 f_s + \dots + y_s f_s = 0\} \subset P^{2k+1}(\Sigma, X)$  be the quasi-smooth hypersurface associated to a quasi-smooth intersection subvariety  $X = X_{f_1} \cdot \dots \cdot X_{f_s} \subset P^d(\Sigma)$  such that  $d + s = 2(k+1)$ . If the Hodge conjecture holds on  $X$  then it holds as well on  $Y$ .

**Corollary 5.4.** If the dimension  $d$  is  $2s-1$ ,  $2s$  or  $2s+1$  then the Hodge conjecture holds on  $Y$ .

**Proof.** Proposition 5.3 and Corollary 3.6.

Angella, D. Cohomologies of certain orbifolds. Journal of Geometry and Physics (2019), 2019-2019.

Batyrev, V. V., and Cox, D. A. On the Hodge structure of projective hypersurfaces in toric varieties. Duke Mathematical Journal (2000), 2000(Aug 2000).

Bruzzo, U., and Montoya, W. On the Hodge conjecture for quasi-smooth intersections in toric varieties. São Paulo J. Math. Sci. Special Section: Geometry in Algebra and Algebra in Geometry (2019).

Caramello Jr, F. Introduction to orbifolds. arXiv:1905.08555 (2019).

Cox, D., Little, and Schenck, H. Toric varieties, vol. 1. American Mathematical Soc., 2011.

Griffiths, P., and Harris, J. Principles of Algebraic Geometry John Wiley & Sons, Ltd, 1978.

Mavlyutov, A. R. Cohomology of complete intersections in toric varieties. Published in Pacific J. of Math. No. 160 (1999), 1999-2000.

Satake, I. On a Generalization of the Notion of Manifold. Proceedings of the National Academy of Sciences of the United States of America (1955), 1955-1955.

Steenbrink, J. H. M. Intersection form for quasi-homogeneous singularities. Compositio Mathematica (1986), 1986-1986.

Voisin, C. Hodge Theory and Complex Algebraic Geometry I, vol. 1. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2003.

Wang, Z. Z., and Zaffran, D. A remark on the Hard Lefschetz theorem for Kähler orbifolds. Proceedings of the American Mathematical Society (2000), 2000(Aug 2000).

[2] Batyrev, V. V., and Cox, D. A. On the Hodge structure of projective hypersurfaces in toric varieties. Duke Mathematical Journal 75, 2 (Aug 1994).

[3] Bruzzo, U., and Montoya, W. On the Hodge conjecture for quasi-smooth intersections in toric varieties. São Paulo J. Math. Sci. Special Section: Geometry in Algebra and Algebra in Geometry (2019).

[3] Bruzzo, U., and Montoya, W. On the Hodge conjecture for quasi-smooth intersections in toric varieties. São Paulo J. Math. Sci.

```
Special Section: Geometry in Algebra and Algebra in Geometry (2021).\n\nA. R
Cohomology of complete intersections in toric varieties. Pub-', lookup_str='
metadata={'source':
'/var/folders/ph/hhm7_zyx4l13k3v8z02dwp1w0000gn/T/tmpgq0ckaja/online_file.pdf
, lookup_index=0)]
```

## Using PyPDFium2

---

```
from langchain.document_loaders import PyPDFium2Loader
```

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

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

## Using PDFMiner

---

```
from langchain.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.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
duplicatess safe to assume that it is redundant info)
```

```

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

semantic_snippets.append(Document(page_content='',metadata=metadata))
    cur_idx += 1
    continue

    # if current snippet's font size <= previous section's content =>
    content belongs to the same section (one can also create
    # a tree like structure for sub sections if needed but that may
    require some more thinking and may be data specific)
    if not semantic_snippets[cur_idx].metadata['content_font'] or s[1] <=
semantic_snippets[cur_idx].metadata['content_font']:
        semantic_snippets[cur_idx].page_content += s[0]
        semantic_snippets[cur_idx].metadata['content_font'] = max(s[1],
semantic_snippets[cur_idx].metadata['content_font'])
        continue

    # if current snippet's font size > previous section's content but less
    than previous section's heading than also make a new
    # section (e.g. title of a pdf will have the highest font size but we
    don't want it to subsume all sections)
    metadata={'heading':s[0], 'content_font': 0, 'heading_font': s[1]}
    metadata.update(data.metadata)
    semantic_snippets.append(Document(page_content='',metadata=metadata))
    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\ncnn [28] and Mask R-CNN [12] are used for identifying document elements [38]\nand detecting tables [30, 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 unified framework to load and use such models.\nThere has been a surge of interest in creating open-source tools for document\nimage processing: a search of document image analysis in Github leads to 5M\nrelevant code pieces 6; yet most of them rely 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 for DIA. However,\nsimilar to the platform developed by Neudecker et al. [21], it is designed for\nanalyzing historical documents, and provides no supports for recent DL models.\nThe DocumentLayoutAnalysis project8 focuses on processing born-digital PDF\ndocuments via analyzing the stored PDF data. Repositories like DeepLayout9\nand Detectron2-PubLayNet10 are individual deep learning models trained on\nlayout analysis datasets without support for the full DIA pipeline. The Document\nAnalysis and Exploitation (DAE) platform [15] and the DeepDIVA project [2]\naim to improve the reproducibility of DIA methods (or DL models), yet they\nare not 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 Detectron2 [35],\n6 The number shown is obtained by specifying the search type as 'code'.  
7 <https://ocr-d.de/en/about>  
8 <https://github.com/BobLd/DocumentLayoutAnalysis>  
9 <https://github.com/leonlulu/DeepLayout>  
10 <https://github.com/hpanwar08/detectron2>  
11 <https://github.com/JaidedAI/EasyOCR>  
12 <https://github.com/PaddlePaddle/PaddleOCR>  
4\nZ. Shen et al.  
Fig. 1: The overall architecture of 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 carefully 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 and 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 community platform inspired by established model hubs such as Torch Hub [23] and TensorFlow Hub [1]. It enables the sharing of pretrained models as well as full document processing pipelines that are unique to DIA tasks. There have been a variety of document data collections to facilitate the development of DL models. Some examples include PRIMA [3] (magazine layouts), PubLayNet [38] (academic paper layouts), Table Bank [18] (tables in academic papers), Newspaper Navigator Dataset [16, 17] (newspaper figure layouts) and HJDataset [31] (historical Japanese document layouts). A spectrum of models trained on these datasets are currently available in the LayoutParser model zoo to support different use cases.

```
{
  'metadata': {
    'heading': '2 Related Work',
    '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.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 Learning Based Document Image Analysis  
Zejiang Shen1 (✉), Ruochen  
Zhang2, Melissa Dell3, Benjamin Charles Germain  
Lee4, Jacob Carlson3, and  
Weining Li5  
1 Allen Institute for AI  
shannons@allenai.org  
2 Brown  
University  
ruochen zhang@brown.edu  
3 Harvard  
University  
{melissadell,jacob carlson}@fas.harvard.edu  
4 University of  
Washington  
nbcgl@cs.washington.edu  
5 University of  
Waterloo  
nw422li@uwaterloo.ca  
Abstract. Recent advances in document image
```

analysis (DIA) have been primarily driven by the application of neural networks. Ideally, research outcomes could be easily deployed in production and extended for further investigation. However, various factors like loosely organized codebases and sophisticated model configurations complicate the easy reuse of important innovations by a wide audience. Though there have been on-going efforts to improve reusability and simplify deep learning (DL) model development in disciplines like natural language processing and computer vision, none of them are optimized for challenges in the domain of DIA. This represents a major gap in the existing toolkit, as DIA is central to academic research across a wide range of disciplines in the social sciences and humanities. This paper introduces LayoutParser, an open-source library for streamlining the usage of DL in DIA research and applications. The core LayoutParser library comes with a set of simple and intuitive interfaces for applying and customizing DL models for layout detection, character recognition, and many other document processing tasks. To promote extensibility, LayoutParser also incorporates a community platform for sharing both pre-trained models and full document digitization pipelines. We demonstrate that LayoutParser is helpful for both lightweight and large-scale digitization pipelines in real-world use cases. The library is publicly available at <https://layout-parser.github.io>.

**Keywords:** Document Image Analysis · Deep Learning · Layout Analysis · Character Recognition · Open Source library · Toolkit.

**1 Introduction**

Deep Learning (DL)-based approaches are the state-of-the-art for a wide range of document image analysis (DIA) tasks including document image classification [11, [arXiv:2103.15348v2](#) [cs.CV] 21 Jun 2021], 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.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.document_loaders import PDFPlumberLoader
```

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

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

```
data[0]
```

```
Document(page_content='LayoutParser: A Unified Toolkit for Deep\nLearnin
Document Image Analysis\nZejiang Shen1 ((cid:0)), Ruochen Zhang2, Melissa De
Benjamin Charles Germain\nLee4, Jacob Carlson3, and Weining Li5\n1 Allen Ins
AI\n1202 shannons@allenai.org\n2 Brown University\nruochen zhang@brown.edu\n
University\nnuJ {melissadell,jacob carlson}@fas.harvard.edu\n4 University of
Washington\nnbcgl@cs.washington.edu\n12 5 University of
Waterloo\nw422li@uwaterloo.ca\n]VC.sc[\nAbstract.
Recentadvancesindocumentimageanalysis(DIA)havebeen\nprimarily driven by the
of neural networks. Ideally,
research\noutcomescouldbeeasilydeployedinproductionandextendedforfurther\nin
However, various factors like loosely organized codebases\nand sophisticated
configurations complicate the easy reuse of im-\n2v84351.3012:viXra
portantinnovationsbyawideaudience.Thoughtherehavebeenon-going\nefforts to im
reusability and simplify deep learning (DL)
model\ndevelopmentindisciplineslikenaturallanguageprocessingandcomputer\nvis
them are optimized for challenges in the domain of DIA.\nThis represents a m
```



the existing toolkit, as DIA is central to academic research across a wide range of disciplines in the social sciences and humanities. This paper introduces LayoutParser, an open-source library for streamlining the usage of DL in DIA research and applications. The core LayoutParser library comes with a set of simple and intuitive interfaces for applying and customizing DL models for layout detection, character recognition, and many other document processing tasks. To promote extensibility, LayoutParser also incorporates a community platform for sharing pre-trained models and full document digitization pipelines. We demonstrate that LayoutParser is helpful for both lightweight and large-scale digitization projects in real-world use cases. The library is publicly available at <https://layout-parser.github.io>.  
**Keywords:** Document Image Analysis · Deep Learning · Layout Analysis · Character Recognition · Open Source library · Toolkit.

# Introduction

Deep Learning (DL)-based approaches are the state-of-the-art for a wide range of document image analysis (DIA) tasks including document image classification [11, 'source': 'example\_data/layout-parser-paper.pdf', 'file\_path': 'example\_data/layout-parser-paper.pdf', 'page': 1, 'total\_pages': 16, 'Author': '', 'CreationDate': 'D:20210622012710Z', 'Creator': 'LaTeX with hyperref', 'Keywords': '', 'ModDate': 'D:20210622012710Z', 'PTEX.Fullbanner': 'This is pdfTeX, Version 3.14159265- (TeX Live 2020) kpathsea version 6.3.2', 'Producer': 'pdfTeX-1.40.21', 'Subject': '', 'Trapped': 'False']