Create a document dataset

This guide will show you how to create a document dataset with `PdfFolder` and some metadata. This is a no-code solution for quickly creating a document dataset with several thousand pdfs.

> [!TIP]
> You can control access to your dataset by requiring users to share their contact information first. Check out the Gated datasets guide for more information about how to enable this feature on the Hub.

PdfFolder

The `PdfFolder` is a dataset builder designed to quickly load a document dataset with several thousand pdfs without requiring you to write any code.

> [!TIP]
> 💡 Take a look at the Split pattern hierarchy to learn more about how `PdfFolder` creates dataset splits based on your dataset repository structure.

`PdfFolder` automatically infers the class labels of your dataset based on the directory name. Store your dataset in a directory structure like:

```
folder/train/resume/0001.pdf
folder/train/resume/0002.pdf
folder/train/resume/0003.pdf

folder/train/invoice/0001.pdf
folder/train/invoice/0002.pdf
folder/train/invoice/0003.pdf
```

If the dataset follows the `PdfFolder` structure, then you can load it directly with load_dataset():

```
>>> from datasets import load_dataset

>>> dataset = load_dataset("path/to/folder")
```

This is equivalent to passing `pdffolder` manually in load_dataset() and the directory in `data_dir`:

```
>>> dataset = load_dataset("pdffolder", data_dir="/path/to/folder")
```

You can also use `pdffolder` to load datasets involving multiple splits. To do so, your dataset directory should have the following structure:

```
folder/train/resume/0001.pdf
folder/train/resume/0002.pdf
folder/test/invoice/0001.pdf
folder/test/invoice/0002.pdf
```

> [!WARNING]
> If all PDF files are contained in a single directory or if they are not on the same level of directory structure, `label` column won't be added automatically. If you need it, set `drop_labels=False` explicitly.

If there is additional information you'd like to include about your dataset, like text captions or bounding boxes, add it as a `metadata.csv` file in your folder. This lets you quickly create datasets for different computer vision tasks like text captioning or object detection. You can also use a JSONL file `metadata.jsonl` or a Parquet file `metadata.parquet`.

```
folder/train/metadata.csv
folder/train/0001.pdf
folder/train/0002.pdf
folder/train/0003.pdf
```

Your `metadata.csv` file must have a `file_name` or `*_file_name` field which links PDF files with their metadata:

```
file_name,additional_feature
0001.pdf,This is a first value of a text feature you added to your pdfs
0002.pdf,This is a second value of a text feature you added to your pdfs
0003.pdf,This is a third value of a text feature you added to your pdfs
```

or using `metadata.jsonl`:

```
{"file_name": "0001.pdf", "additional_feature": "This is a first value of a text feature you added
{"file_name": "0002.pdf", "additional_feature": "This is a second value of a text feature you adde
{"file_name": "0003.pdf", "additional_feature": "This is a third value of a text feature you added
```

Here the `file_name` must be the name of the PDF file next to the metadata file. More generally, it must be the relative path from the directory containing the metadata to the PDF file.

It's possible to point to more than one pdf in each row in your dataset, for example if both your input and output are pdfs:

```
{"input_file_name": "0001.pdf", "output_file_name": "0001_output.pdf"}
{"input_file_name": "0002.pdf", "output_file_name": "0002_output.pdf"}
{"input_file_name": "0003.pdf", "output_file_name": "0003_output.pdf"}
```

You can also define lists of pdfs. In that case you need to name the field `file_names` or `*_file_names`. Here is an example:

```
{"pdfs_file_names": ["0001_part1.pdf", "0001_part2.pdf"], "label": "urgent"}
{"pdfs_file_names": ["0002_part1.pdf", "0002_part2.pdf"], "label": "urgent"}
{"pdfs_file_names": ["0003_part1.pdf", "0002_part2.pdf"], "label": "normal"}
```

OCR (Optical character recognition)

OCR datasets have the text contained in a pdf. An example `metadata.csv` may look like:

```
file_name,text
0001.pdf,Invoice 1234 from 01/01/1970...
0002.pdf,Software Engineer Resume. Education: ...
0003.pdf,Attention is all you need. Abstract. The ...
```

Load the dataset with `PdfFolder`, and it will create a `text` column for the pdf captions:

```
>>> dataset = load_dataset("pdffolder", data_dir="/path/to/folder", split="train")
>>> dataset[0]["text"]
"Invoice 1234 from 01/01/1970..."
```

Upload dataset to the Hub

Once you've created a dataset, you can share it to the using `huggingface_hub` for example. Make sure you have the huggingface_hub library installed and you're logged in to your Hugging Face account (see the Upload with Python tutorial for more details).

Upload your dataset with `huggingface_hub.HfApi.upload_folder`:

```
from huggingface_hub import HfApi
api = HfApi()

api.upload_folder(
    folder_path="/path/to/local/dataset",
    repo_id="username/my-cool-dataset",
    repo_type="dataset",
)
```