Create a video dataset

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

> [!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.

VideoFolder

The `VideoFolder` is a dataset builder designed to quickly load a video dataset with several thousand videos without requiring you to write any code.

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

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

```
folder/train/dog/golden_retriever.mp4
folder/train/dog/german_shepherd.mp4
folder/train/dog/chihuahua.mp4

folder/train/cat/maine_coon.mp4
folder/train/cat/bengal.mp4
folder/train/cat/birman.mp4
```

If the dataset follows the `VideoFolder` 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 `videofolder` manually in load_dataset() and the directory in `data_dir`:

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

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

```
folder/train/dog/golden_retriever.mp4
folder/train/cat/maine_coon.mp4
folder/test/dog/german_shepherd.mp4
folder/test/cat/bengal.mp4
```

> [!WARNING]
> If all video 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.mp4
folder/train/0002.mp4
folder/train/0003.mp4
```

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

```
file_name,additional_feature
0001.mp4,This is a first value of a text feature you added to your videos
0002.mp4,This is a second value of a text feature you added to your videos
0003.mp4,This is a third value of a text feature you added to your videos
```

or using `metadata.jsonl`:

```
{"file_name": "0001.mp4", "additional_feature": "This is a first value of a text feature you added
{"file_name": "0002.mp4", "additional_feature": "This is a second value of a text feature you adde
{"file_name": "0003.mp4", "additional_feature": "This is a third value of a text feature you added
```

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

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

```
{"input_file_name": "0001.mp4", "output_file_name": "0001_output.mp4"}
{"input_file_name": "0002.mp4", "output_file_name": "0002_output.mp4"}
{"input_file_name": "0003.mp4", "output_file_name": "0003_output.mp4"}
```

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

```
{"videos_file_names": ["0001_left.mp4", "0001_right.mp4"], "label": "moving_up"}
{"videos_file_names": ["0002_left.mp4", "0002_right.mp4"], "label": "moving_down"}
{"videos_file_names": ["0003_left.mp4", "0003_right.mp4"], "label": "moving_right"}
```

Video captioning

Video captioning datasets have text describing a video. An example `metadata.csv` may look like:

```
file_name,text
0001.mp4,This is a golden retriever playing with a ball
0002.mp4,A german shepherd
0003.mp4,One chihuahua
```

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

```
>>> dataset = load_dataset("videofolder", data_dir="/path/to/folder", split="train")
>>> dataset[0]["text"]
"This is a golden retriever playing with a ball"
```

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",
)
```

WebDataset

The WebDataset format is based on TAR archives and is suitable for big video datasets. Indeed you can group your videos in TAR archives (e.g. 1GB of videos per TAR archive) and have thousands of TAR archives:

```
folder/train/00000.tar
folder/train/00001.tar
folder/train/00002.tar
...
```

In the archives, each example is made of files sharing the same prefix:

```
e39871fd9fd74f55.mp4
e39871fd9fd74f55.json
f18b91585c4d3f3e.mp4
f18b91585c4d3f3e.json
ede6e66b2fb59aab.mp4
ede6e66b2fb59aab.json
ed600d57fcee4f94.mp4
ed600d57fcee4f94.json
...
```

You can put your videos labels/captions/features using JSON or text files for example.

For more details on the WebDataset format and the python library, please check the WebDataset documentation.

Load your WebDataset and it will create on column per file suffix (here "mp4" and "json"):

```
>>> from datasets import load_dataset

>>> dataset = load_dataset("webdataset", data_dir="/path/to/folder", split="train")
>>> dataset[0]["json"]
{"bbox": [[302.0, 109.0, 73.0, 52.0]], "categories": [0]}
```