Share a dataset using the CLI

At Hugging Face, we are on a mission to democratize good Machine Learning and we believe in the value of open source. That's why we designed 🤗 Datasets so that anyone can share a dataset with the greater ML community. There are currently thousands of datasets in over 100 languages in the Hugging Face Hub, and the Hugging Face team always welcomes new contributions!

Dataset repositories offer features such as:

- Free dataset hosting
- Dataset versioning
- Commit history and diffs
- Metadata for discoverability
- Dataset cards for documentation, licensing, limitations, etc.
- Dataset Viewer

This guide will show you how to share a dataset folder or repository that can be easily accessed by anyone.

Add a dataset

You can share your dataset with the community with a dataset repository on the Hugging Face Hub.
It can also be a private dataset if you want to control who has access to it.

In a dataset repository, you can host all your data files and configure your dataset to define which file goes to which split.
The following formats are supported: CSV, TSV, JSON, JSON lines, text, Parquet, Arrow, SQLite, WebDataset.
Many kinds of compressed file types are also supported: GZ, BZ2, LZ4, LZMA or ZSTD.
For example, your dataset can be made of `.json.gz` files.

When loading a dataset from the Hub, all the files in the supported formats are loaded, following the repository structure.

For more information on how to load a dataset from the Hub, take a look at the load a dataset from the Hub tutorial.

Create the repository

Sharing a community dataset will require you to create an account on hf.co if you don't have one yet.
You can directly create a new dataset repository from your account on the Hugging Face Hub, but this guide will show you how to upload a dataset from the terminal.

1. Make sure you are in the virtual environment where you installed Datasets, and run the following command:

```
huggingface-cli login
```

2. Login using your Hugging Face Hub credentials, and create a new dataset repository:

```
huggingface-cli repo create my-cool-dataset --type dataset
```

Add the `-organization` flag to create a repository under a specific organization:

```
huggingface-cli repo create my-cool-dataset --type dataset --organization your-org-name
```

Prepare your files

Check your directory to ensure the only files you're uploading are:

- The data files of the dataset
- The dataset card `README.md`

huggingface-cli upload

Use the `huggingface-cli upload` command to upload files to the Hub directly. Internally, it uses the same `upload_file` and `upload_folder` helpers described in the Upload guide. In the examples below, we will walk through the most common use cases. For a full list of available options, you can run:

```
>>> huggingface-cli upload --help
```

For more general information about `huggingface-cli` you can check the CLI guide.

Upload an entire folder

The default usage for this command is:

```
# Usage:  huggingface-cli upload [dataset_repo_id] [local_path] [path_in_repo] --repo-type dataset
```

To upload the current directory at the root of the repo, use:

```
>>> huggingface-cli upload my-cool-dataset . . --repo-type dataset
https://huggingface.co/datasets/Wauplin/my-cool-dataset/tree/main/
```

> [!TIP]
> If the repo doesn't exist yet, it will be created automatically.

You can also upload a specific folder:

```
>>> huggingface-cli upload my-cool-dataset ./data . --repo-type dataset
https://huggingface.co/datasetsWauplin/my-cool-dataset/tree/main/
```

Finally, you can upload a folder to a specific destination on the repo:

```
>>> huggingface-cli upload my-cool-dataset ./path/to/curated/data /data/train --repo-type dataset
https://huggingface.co/datasetsWauplin/my-cool-dataset/tree/main/data/train
```

Upload a single file

You can also upload a single file by setting `local_path` to point to a file on your machine. If that's the case, `path_in_repo` is optional and will default to the name of your local file:

```
>>> huggingface-cli upload Wauplin/my-cool-dataset ./files/train.csv --repo-type dataset
https://huggingface.co/datasetsWauplin/my-cool-dataset/blob/main/train.csv
```

If you want to upload a single file to a specific directory, set `path_in_repo` accordingly:

```
>>> huggingface-cli upload Wauplin/my-cool-dataset ./files/train.csv /data/train.csv --repo-type d
https://huggingface.co/datasetsWauplin/my-cool-dataset/blob/main/data/train.csv
```

Upload multiple files

To upload multiple files from a folder at once without uploading the entire folder, use the `--include` and `--exclude` patterns. It can also be combined with the `--delete` option to delete files on the repo while uploading new ones. In the example below, we sync the local Space by deleting remote files and uploading all CSV files:

```
# Sync local Space with Hub (upload new CSV files, delete removed files)
>>> huggingface-cli upload Wauplin/my-cool-dataset --repo-type dataset --include="/data/*.csv" --d
...
```

Upload to an organization

To upload content to a repo owned by an organization instead of a personal repo, you must explicitly specify it in the `repo_id`:

```
>>> huggingface-cli upload MyCoolOrganization/my-cool-dataset . . --repo-type dataset
https://huggingface.co/datasetsMyCoolOrganization/my-cool-dataset/tree/main/
```

Upload to a specific revision

By default, files are uploaded to the `main` branch. If you want to upload files to another branch or reference, use the `--revision` option:

```
# Upload files to a PR
huggingface-cli upload bigcode/the-stack . . --repo-type dataset --revision refs/pr/104
...
```

Note: if `revision` does not exist and `--create-pr` is not set, a branch will be created automatically from the `main` branch.

Upload and create a PR

If you don't have the permission to push to a repo, you must open a PR and let the authors know about the changes you want to make. This can be done by setting the `--create-pr` option:

```
# Create a PR and upload the files to it
>>> huggingface-cli upload bigcode/the-stack --repo-type dataset --revision refs/pr/104 --create-p
https://huggingface.co/datasets/bigcode/the-stack/blob/refs%2Fpr%2F104/
```

Upload at regular intervals

In some cases, you might want to push regular updates to a repo. For example, this is useful if your dataset is growing over time and you want to upload the data folder every 10 minutes. You can do this using the `--every` option:

```
# Upload new logs every 10 minutes
huggingface-cli upload my-cool-dynamic-dataset data/ --every=10
```

Specify a commit message

Use the `--commit-message` and `--commit-description` to set a custom message and description for your commit instead of the default one

```
>>> huggingface-cli upload Wauplin/my-cool-dataset ./data . --repo-type dataset --commit-message="
...
https://huggingface.co/datasetsWauplin/my-cool-dataset/tree/main
```

Specify a token

To upload files, you must use a token. By default, the token saved locally (using `huggingface-cli login`) will be used. If you want to authenticate explicitly, use the `--token` option:

```
>>> huggingface-cli upload Wauplin/my-cool-dataset ./data . --repo-type dataset --token=hf_****
...
https://huggingface.co/datasetsWauplin/my-cool-data/tree/main
```

Quiet mode

By default, the `huggingface-cli upload` command will be verbose. It will print details such as warning messages, information about the uploaded files, and progress bars. If you want to silence all of this, use the `--quiet` option. Only the last line (i.e. the URL to the uploaded files) is printed. This can prove useful if you want to pass the output to another command in a script.

```
>>> huggingface-cli upload Wauplin/my-cool-dataset ./data . --repo-type dataset --quiet
https://huggingface.co/datasets/Wauplin/my-cool-dataset/tree/main
```

Enjoy !

Congratulations, your dataset has now been uploaded to the Hugging Face Hub where anyone can load it in a single line of code! �

```
dataset = load_dataset("Wauplin/my-cool-dataset")
```

If your dataset is supported, it should also have a Dataset Viewer for everyone to explore the dataset content.

Finally, don't forget to enrich the dataset card to document your dataset and make it discoverable! Check out the Create a dataset card guide to learn more.