Data Management Plan

Amanjyoti Mridha, ILS 2023 Data Project

Section 1: Types of data

> **1.1** types of data captured
> **1.2** how data will be collected and created
> **1.3** how much data

For this project, the objective is to explore what the most popular themes in the prompts to text-to-image machine learning models are, as well as how these prompts show the relations between artists, different mediums, artistic movements, and artistic themes/flavors. Thus, the raw data (found in the data/raw/ directory) contains a series of txt and parquet files. The txt files will each contain the names of artists, different artistic mediums, historical movements, and flavors/themes in each line as strings. This data is sourced from the CLIP Interrogate dictionary from the public, open source repository AUTOMATIC1111/stable-diffusion-webui, which provides sufficient samples for these four categories. As for the parquet files, these contain tabular textual data where every row under the column text contains the prompt entered by the user into a text-to-image model called MidJourney. This is also sourced from an open sourced dataset in the repository Succinctly/Midjourney Prompts. Links to both original repositories can be found in the README.md file in the main project directory. As for the data created, using the original data, first a cosine similarity between nodes, specifically for the flavors.txt file which contains numerous redundant entries (entries with topics too similar to treat separately but textually slightly different), will be used to create a condensed file containing unique nodes. Then, the text embeddings for every node will be stored along with the corresponding id's, and these embeddings will be constructed using the CLIP text encoder model. From here, using the prompts, a network graph will be computed and the corresponding adjacency matrix will be stored as a csv file for further processing. All processed data will be found in a subdirectory called "processed" and more details on these files can be found in section 2. As for the quantity of data, the raw data contains the names of 5,265 different artists, 100,970 different themes/flavors, 95 artistic mediums, 200 artistic movements, and 246,600 prompts split among three parquet files. For the processed data, the condensed_flavor_nodes file contains 2,048 nodes, the embeddings file and node_id file contains 7,608 text embeddings and ids respectively, and the graph contains an adjacency matrix for these nodes with dimensions of 7,608 by 7,608.

Section 2: 2: Standards for data and metadata

> **2.1** Identifies metadata standards and/or metadata formats that will used for the proposed project
> **2.2** Describes data formats created or used during project
> **2.3** Identifies data formats that will be used for storing data
> **2.4** If the proposed project includes the use of unusual data formats, the plan discusses the proposed solution for converting data into more accessible formats

All details about the data, it's structure and contents, how it was made, its sources, and how to run the python scripts to generate and process the data are explained in the README.md file in the root directory of the repository. There are several formats used for various datatypes, including txt and parquet files for the raw datasets as well as csv, json, and pytorch/pickle serialized dump files for some of the intermediate processing steps. The data for this project has the following structure:

```
data/
|
├──processed/
|   |
|   ├──embeddings.pd
|   |
|   ├──flavors_combine_redundant.json
|   |
```

```
│   ├──graph.csv
│   │
│   └──node_ids.json
│
└──raw/
    │
    ├──artists.txt
    │
    ├──flavors.txt
    │
    ├──mediums.txt
    │
    ├──movements.txt
    │
    └──prompt data/
        │
        ├──data/
        │   │
        │   ├──test-00000-of-00001.parquet
        │   │
        │   ├──train-00000-of-00001.parquet
        │   │
        │   └──validation-00000-of-00001.parquet
        │
        ├──dataset_infos.json
        │
        ├──gitattributes
        │
        └──README.md
```

Here is a breakdown of the files and directories:

- data (directory containint data)
  - processed (directory containing processed network graph data)
    - embeddings.pd (pytorch dump file containing dictionary of the text embeddings for all the data from artists, flavors_combine_redundant, mediums, and movements. Can be opened using torch.load)
    - flavors_combine_redundant.json (file containing a json structure where each node has all the flavors that are very similar combined)
  - raw (directory containing raw prompt and dictionary data)
    - artists.txt (file containing a list of artists)
    - flavors.txt (file containing various prompt modifiers)
    - mediums.txt (file containing various artistic mediums)
    - movements.txt (file containing historical artistic movements)
    - prompt data (directory containing cloned midjourney prompt data)
      - dataset_infos.json (file with info about the data structure from original authors)
      - gitattributes (git log file from original repo)
      - README.md (readme about the prompt dataset from original author)
      - data (directory containing prompt data in train, test, val split, each as a parquet file)
        - test-00000-of-00001.parquet (contains a split of 12.3k prompts)
        - train-00000-of-00001.parquet (contains a split of 222k prompts)
        - validation-00000-of-00001.parquet (contains a split of 12.3k prompts)

Important information on data structure:

txt files:
Every line contains the value (a string) representing the node

parquet files:
These are from the midjourney prompt dataset and contain tabular data. There is a column called "text" and each row in this column contains a prompt. There is a train, test, validation split for the data and I use the train split. This is a commonly used file format for large datasets and can be opened in numerous programs, in this case I use the pandas/pyarrow python libraries and open the dataset using the line: `pandas.read_parquet("parquet_file_name.parquet")` .

flavors_combine_redundant.json:
This json file has the following structure:

```
{
    "name" : "flavors_combine_redundant.json", // this is the name of the file
    "nodes" : [ // this is a list with the nodes
        { // each node is a dictionary containing the values which were deemed redundant and combined
            "n1" : "value 1", // n1 is the first redundant value and so forth
            "n2" : "redundant value 2",
            "n3" : "redundant value 3"
                        ...
        },
        { // the next node and so forth
                        "n1" : "value 1"
        },
        ...
    ]
}
```

embeddings.pd

> ***NOTE:*** This file is too large to upload to github so if you clone this repo from there, you will have to run `python create_embeddings.py` in order to generate this dump file. This file can, however, be found on all other locations where the data is stored (google drive, cloud copies, local backups, etc.)

This is a serialized dump file made with torch.save that contains a dictionary of text embeddings. This dictionary, once loaded using torch.load("embeddings.pd"), has the following structure:

```
{
        "artists.txt" : [ // a list with the nodes
                (name, embedding), // every node is a tuple containing the name (the string value) as well as
the embedding (a torch tensor)
                (name, embedding),
                // ...
        ],
        "mediums.txt" : [],
        "movements.txt" : [],
        "flavors" : []
}
```

node_ids.json
This is a json file that contains the id for the node in the graph as well as a set of dictionaries for which id's correspond to which categories. This file has the following structure:

```
{
    "name": "node_ids.json",
    "ids": {
            "0" : "node name" // this dict contains the node id and the corresponding node text
    },
    "cateogries": {
```

```
        "artists.txt": [0, 1, ...,], // this contains the list of ids which correspond to this category
        "mediums.txt": [...,],
        "movements.txt": [...,],
        "flavors": [...,]
    }
}
```

graph.csv
This is a csv file the contains the adjacency matrix for the network graph (each cell represents an edge between the node represented by the column index and the node represented by the row index). This can be loaded to a numpy array using np.loadtxt("graph.csv", delimiter=","). You can use other programs to edit this (it is just a csv) but because of how large the graph is (~1.3+ GB) it is better to programmatically do it. I provide a script called `network_graph.py` that opens this adjacency matrix and displays a sample of the nodes with the highest connections.

In order to address any potential issues with utilizing some of the more specialized data file formats for the processed data in particular, code is provided to parse the json and pd files, and in case they no longer function, the scripts used to generate them are also provided so support can still exist for newer python or torch versions in case any major changes occur. The embeddings file cannot, unfortunately, be converted to typical formats such as json or yaml for storing dictionary like data because the file contains serialized torch tensors, and while this could be converted to list like data, the resulting files are too large to be opened by typical programs. This could potentially be stored into separate classified csv or parquet files, however it is recommended instead to utilize the provided python scripts to process and regenerate this data.

Section 3: Policies for access and sharing

> **3.1 Public Availability**: Provides details on when the data will be made publicly available
> **3.2 Data Sharing Method**: Describes how the data will be made publicly available
> **3.3 Data Protection**: If the data are deemed to be of a “sensitive” nature, describes what protections will be put into place to protect privacy or confidentiality of research subjects
> **3.4 Intellectual Property Rights**: Describes what intellectual property rights to the data and supporting materials will be given to the public and which will be retained by project personnel (if any)
> **3.5 Data Security**: Describes security measures that will be in place to protect the data from unauthorized access
> **3.6 Data Sharing Limitations**: If there are factors that limit the ability to share data, e.g. commercialization or proprietary nature of the data, plan describes those conditions and describes to whom the data will be made available and under what conditions
> **3.7 Data Retention**: Describes how long the data will be retained and made available to people outside of the project
> **3.8 Data Submission**: If data are going to be submitted to supplementary materials sections of peer-reviewed journals, the plan describes this
> **3.9 Data Formats**: Describes data types or formats that will be used for making data available

The data is already publicly available on GitHub, and subsequent access to other storage locations, specifically google drive, microsoft cloud storage links (OneDrive), and on the public Kaggle dataset storage will be provided by the end of 2023. These sites will allow interested parties to easily download and manipulate the data, as well as browse through the overall structure. Because these datasets contain no private or sensitive information, just being a dictionary of artists and artistic terms/themes/history with publicly provided textual prompt information, the data can be safely provided as is on these platforms. Following the licensing agreements from the original source and following with the open sourced theme, the data is provided under the apache 2.0 and MIT license providing free public access to all the data with the ability to use, modify, and distribute the data as the user sees fit so long as they include the citation of the original source. To protect the data from unauthorized access, the GoogleDrive will be presented with read only permission, as will the public GitHub and Kaggle repositories/datasets meaning only read and fork access is provided without the ability to modify the existing data. Local copies will be kept separated from the internet and backups will be kept of the data. There are no limitations on sharing the data with it being publicly accessible to everyone. Furthermore, Amanjyoti Mridha will be responsible for documenting and explaining the repository, and the data is guaranteed to be preserved for the next 4 years and will continue to be preserved on GitHub and Kaggle indefinitely. To make the data as accessible as possible, the raw data uses only text and parquet file formats which are easily manipulated and used with many different software utilities and programming libraries. Additionally, with regards to the

parquet files, if formats such as csv are desired, a script called `convert_parquet_to_csv.py` is provided which will convert the specified file to a csv format. For the other processed formats, json is a very accessible format and the specialized file can be used with standard python operations (accessed like a dictionary or array), with code provided as well to utilize and generate these files if necessary which can be easily adapted for other needs.

Section 4: Policies and provisions for re-use and redistribution

> **4.1** Describes the policies in place governing the use and reuse of the data
> **4.2** Describes the policies for redistribution of the data
> **4.3** Describes policies for building off of the data, such as through the creation of derivatives
> **4.4*** Describes methods for communicating policies or guidelines for reuse, redistribution, and creation of derivatives

Regarding the use, reuse, and redistribution of the data as well as the creation of derivatives, the repository is provided under the Apache 2.0 and MIT licenses meaning users are free to use and alter the data as they like, so long as the provide citations for the original sources off the data. These licenses are placed directly in the home directory of the repository and are listed in the properties of the README.md file. Thus, these licenses are easily available and listed for users to see.

Section 5: Plans for data archiving and preservation of access

> **5.1** Provides details on how the data will be archived
> **5.2*** Describes how access to the archived data will be maintained
> **5.3*** Describes plans for archiving and preserving digital data
> **5.4*** Describes plans for archiving and preserving physical data
> **5.5*** Identifies a timeframe for how long data will be archived
> **5.6*** Plan discusses the types or formats of data the investigator expects to retain in their possession

As all the data is in a digital form, public access to the data will be provided through several primary means, the GitHub repository, an uploaded kaggle dataset, a public google drive folder, and a Microsoft OneDrive public folder. Users can visit any one of these platforms to acquire the dataset. Furthermore, a local copy of the data will be stored on my computer and my personal cloud storage, alongside one copy on an external drive which is kept safely with me. If necessary, I can be contacted via email for access to the dataset and I will send a copy via Microsoft OneDrive. I plan to maintain the shared folder links to the data for the next four years, and the github repository will remain indefinitely in a frozen state after 2023. The data will be preserved in txt, parquet, json, and csv formats with the embeddings.pd file being open to recreation via the running of the `create_embeddings.py` script.