The procedure for preparing the dataset.

DataCreator.ipynb involves all the codes provided sequentially for data preparation. Before running the codes, the following file has to be ready:

/Dataset/youtube_music_links/{split}/links.jsonl

Where {split} is train, validation or other category used during training. The **Dataset** folder is located with the main folder with **MusicGeneration.**

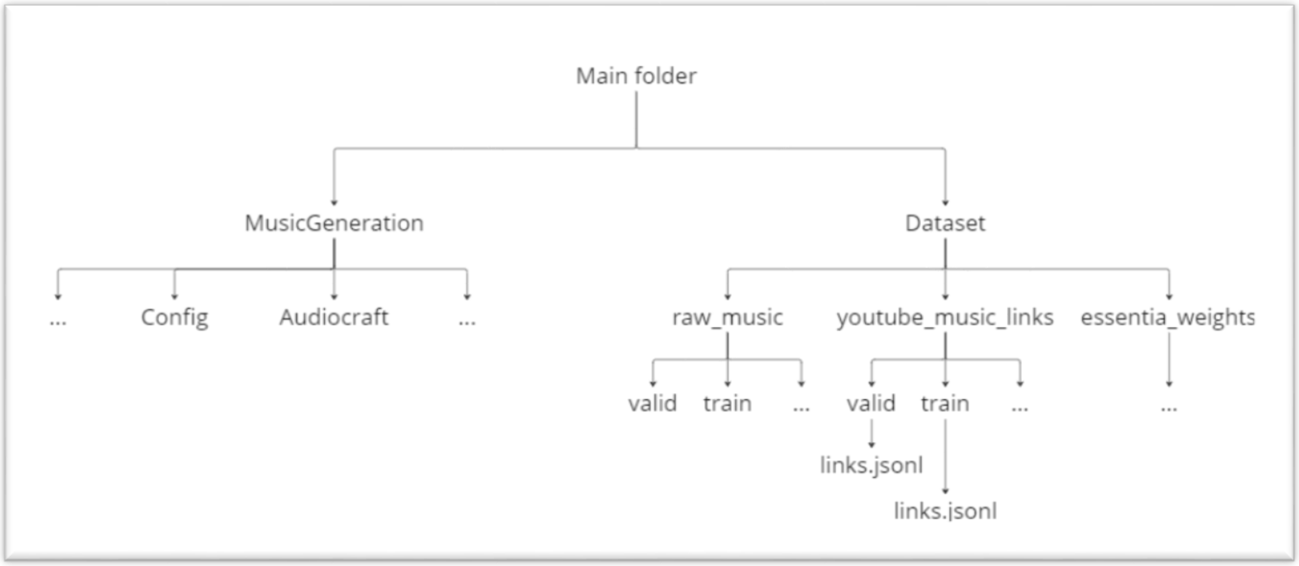

Figure 1: The proper folder architecture for dataset preparation.

Each line represents a music example that should be extracted from YouTube. Attributes of each entry of the **links.jsonl**:

- **“link”:** The link of the YouTube music. Example:
  “https://www.youtube.com/watch?v=rD_GM1cxKLI”

- **“genre”:** The genre of the music piece. Multiple genres can be separated by a comma. Example:
  “Armenian folk, Armenian traditional music”

- **“label”:** The label of the music piece. Represents either the instrument(s) or can be provided the keyword “essentia”. In the second case, the music piece will be automatically labeled using Essentia. Multiple instruments can be separated by slash (/). Example:
  “essentia”; “zurna/drums/percussion/synthesizer”

- **“artist”:** The artist of the music piece. Example:
  Arno Babajanyan

- **“split”:** Defines the interval that should be cut from the YouTube video. An empty string “” means the whole video is taken. Keyword “end” means the end of the YouTube video. “0-end” is equivalent to “”. Example:
  “0-100”; “20-40”; “100-end”; “”

- **“manual_description”:** A manual description. Example:
  “An Armenian folk music, performed on the Armenian duduk, creates a bunch of intense emotions, ranging from dreaming to serenity, weaving a narrative of deep contemplation and nostalgic thoughts.”

Notes on the labeling:

- If multiple segments should be selected from the video, like 20-100 seconds and 150-250, labels corresponding to each segment should be provided with the same order respectively, separated by a comma and a space. For example:

  “label”: “essentia, duduk, violin/piano”, **“split”:** “20-100,150-250,360-400”

- If label is “essentia” then the music piece is cut into 30-second segments and each segment is separately labeled using Essentia. In other words, the two components: moods and instruments are predicted using Essentia’s models.

- If label is other than “essentia”, it represents an instrument(s) separated with slash (/). This means, that the instrument is provided by the “label” attribute, however, it still uses “Essentia” but with higher threshold for predicting instruments, meaning that Essentia should be too confident that the instrument is present. The predicted instrument is added to the instruments provided by the “label” attribute. When the piece is cut into 30-second segments, the instruments are the same for all 30-second pieces, the same for the genre, the only different attribute is the “moods” attribute.

- Mood is always predicted by Essentia.

- Note that you should have **essential_weights** folder with the following weights in it:

  - discogs-effnet-bs64-1.pb
  - mtg_jamendo_instrument-discogs-effnet-1.pb
  - mtg_jamendo_moodtheme-discogs-effnet-1.pb
  - genre_discogs400-discogs-effnet-1.pb (Optional, if you wish to use genre classification also)

  Weights can be downloaded from this link: https://essentia.upf.edu/models.html

The training dataset used in the project can be find in **additional_tools/youtube_music_links** folder.

Data Creator.ipynb

Once the jsonl file with the links annotated is ready, you can proceed to the **DataCreator.ipynb** file.

1) Function **download_split**: Downloads all music files from YouTube from the links provided. The music file is downloaded with the highest available quality and title of the music is used as the name of the wav file. Other attributes from the json file are saved with the same name and .json format.

   Ex. Youtube link with title “Arno Babajanyan – Elegy” will result in two files:

   - Arno Babajanyan – Elegy.wav
   - Arno Babajanyan – Elegy.json

2) Function **divide_into_clips**: Separates the wav file into 30-second portions with 15-second slide. Meaning that 60-second music will be cut into five 30-second portions. The names are saved with the original title with underscore the portion index. Json files are duplicated for each portion with the same title as the wav file with .json format. Thus, 60-second “Arno Babajanyan – Elegy.wav” will be divided into:
   - Arno Babajanyan – Elegy_1.wav, Arno Babajanyan – Elegy_1.json
   - Arno Babajanyan – Elegy_2.wav, Arno Babajanyan – Elegy_2.json
   - Arno Babajanyan – Elegy_3.wav, Arno Babajanyan – Elegy_3.json
   - Arno Babajanyan – Elegy_4.wav, Arno Babajanyan – Elegy_4.json
   - Arno Babajanyan – Elegy_5.wav, Arno Babajanyan – Elegy_5.json

3) Function **prepare_attributes**: Prepares the attributes for labelling for filling them. These are the attributes that are created during this stage from the existing json file:

   "key", "artist", "sample_rate", "file_extension", "description", "keywords", "duration", "bpm", "genre", "title", "name", "instrument", "moods, "label", "manual_description"

   Only the attributes having “description” keyword in the key name are used during the training process. Additionally, “default_description” attribute is added in the stage **fill_descriptions**. So, “description”, “manual_description” and “default_description” in the next steps are processed for training. Some of the attributes are coming from the original MusicGen and are not used or modified in our method. The attributes from the previous json file are transferred to the new one and is overwrote on the same file name.

4) Function **fill_json_split**: Labels the 30-second portion based on the labeling type with the instructions provided above. As a result, duration, mood, genre and instruments are filled. Additionally, data.jsonl file in the **MusicGeneration/egs/{split}**/data.jsonl path is created which is a mandatory file to for the MusicGen model (Several checks for that file is done during the training process, as well as it is needed for training without memory_saver). The data.jsonl file should be given to the **cfg.datasource.{split}**.

5) Function **fill_descriptions**: Fills the “description” and “default_description” attributes with the instructions provided above.

The steps 1-5 are enough for training the model using the original implementation. If you want to go further with our efficient method, proceed with the following steps:

6) Class **PreprocessData** loads the T5 and EnCodec for preprocessing the 30-second wav files and json attributes. PreprocessData.run() encodes the audio file and tokenizes the text (See the paper for more details). The files are then saved in the given directory (default: MusicGeneration/dataset/{split}/{music_name}). Music_name is the folder with the title of the music, inside holds **attributes.pt** and **encodec_encoding.pt**. Attributes.pt represents the tokenized description(s), encodec_encoding.pt represents the tokenized music using EnCodec.

   Please note, that data_split argument given in PreprocessData.run(data_split=’valid’) should be equal to ‘valid’, not ‘validation’ when processing the validation set.