ETL_Dataset_Transform_Functions module

Created on Sun Aug 7 14:50:41 2022

@author: Michael || Sebastian

`ETL_Dataset_Transform_Functions.augment_dataset_Frecuency_Mask`()

Augments a dataset by applying frequency masking to the minority class samples.

This function reads a CSV dataset file containing spectrogram images and their associated labels. It identifies the label with the least amount of information (minority class) and augments the spectrogram images belonging to that class by applying frequency masking.

The augmented images are saved with unique names in a specified directory and the corresponding augmented data is added to the original dataset. The resulting balanced dataset is then saved as a new CSV file.

Parameters: **None** (*None.*) –

Returns: **None**

Return type: None.

`ETL_Dataset_Transform_Functions.create_label_df`(*input_folder*)

Creates a label DataFrame based on csv files in the input folder.

Parameters: **input_folder** (*str*) – The path to the folder containing the csv files.

Returns: **label_df** – The created label DataFrame containing the labels of the csv files.

Return type: pandas DataFrame

`ETL_Dataset_Transform_Functions.create_spectrogram_dataframe`(*directory*)

Creates a Pandas DataFrame containing the names and file paths of spectrograms in a directory.

Parameters: **directory** (*str*) – The directory containing the spectrogram files.

Returns: A dataframe containing two columns: ‘NAME’, the basename of the spectrogram file without the ‘.png’ extension, and ‘Path’, the file path to the spectrogram.

Return type: pandas.DataFrame

ETL_Dataset_Transform_Functions.filter_label(*row*)

Filter the dataframe row based on specific labels.

Parameters: **row** (*pd.Series*) – The row of the dataframe that needs to be filtered.

Returns: True if the label in the row is one of the specific labels (‘PHYCUV_M’, ‘PHYCUV_F’,’BOAALB_M’, ‘BOAALB_F’, ‘BOALUN_F’, ‘BOALUN_M’, ‘PHYCUV_M’, ‘PHYCUV_F’), otherwise False.

Return type: bool

ETL_Dataset_Transform_Functions.filter_window(*df, start, end, step*)

Filter a dataframe to get windows of time periods.

This function filters a dataframe based on start, end and step time values, creating a new dataframe for each time window. The resulting dataframes are grouped by the file name and stored in a dictionary.

Parameters:

- **(pandas.DataFrame)** (*df*) –
- **(int)** (*step*) –
- **(int)** –
- **(int)** –

Returns: **tuple** – df_mlabel (list): A list of dataframes containing only rows with time between start and end values. fname_lists (dict): A dictionary of dataframes grouped by the file name.

Return type: A tuple containing:

ETL_Dataset_Transform_Functions.generate_spectrogram(*audio_file_path, output_file_path, sr=22050, n_fft=2048, hop_length=512*)

Generate spectrogram from audio file and save it as an image.

Parameters:

- **audio_file_path** (*str*) – Path to audio file
- **output_file_path** (*str*) – Path to save the generated spectrogram
- **sr** (*int, optional*) – Sampling rate of the audio file, by default 22050
- **n_fft** (*int, optional*) – Length of the FFT window, by default 2048
- **hop_length** (*int, optional*) – Number of samples between successive frames, by default 512

Return type: None

ETL_Dataset_Transform_Functions.generate_spectrograms_from_folder(*input_folder, output_folder*)

Generates spectrograms from WAV files in a given folder and saves the spectrograms in a specified output folder.

Parameters:

- **input_folder** (*str*) – The path to the folder containing the WAV files.
- **output_folder** (*str*) – The path to the folder where the spectrograms will be saved. If the folder doesn't exist, it will be created.

Return type: None

`ETL_Dataset_Transform_Functions.get_labels_from_csv`(*csv_file_path*)

This function reads a csv file and returns a list of labels.

Parameters: **csv_file_path** (*str*) – The path to the csv file that contains the labels.

Returns: A list of strings representing the labels.

Return type: list

`ETL_Dataset_Transform_Functions.get_spectrogram_paths`(*directory*)

Return a list of spectrogram file paths found in the directory.

Parameters: **directory** (*str*) – The directory to search for spectrogram files.

Returns: List of spectrogram file paths.

Return type: list

`ETL_Dataset_Transform_Functions.load_annotations`(*path_annot, nombre*)

Loads annotations from .txt files in the given directory and returns a dataframe.

Parameters:

- **path_annot** (*str*) – The path to the folder containing the annotation files.
- **nombre** (*list*) – A list of strings representing the names of the annotation files (without the '.txt' extension).

Returns:

- **df_mlabel** (*pd.DataFrame*) – The dataframe containing the loaded annotations.
- **fname_lists** (*list*) – A list of filenames corresponding to the loaded annotations.

`ETL_Dataset_Transform_Functions.preprocess_images`(*paths, target_size=(224, 224, 3)*)

Preprocesses a batch of images by resizing them, converting them to NumPy arrays, and normalizing the pixel values.

Parameters:

- **(list)** (*paths*) –
- **(tuple** (*target_size*) –
- **optional)** (*Target size for resizing the images. Defaults to (224, 224, 3).*) –

Returns: **numpy.ndarray**

Return type: Array containing the preprocessed images.

`ETL_Dataset_Transform_Functions.save_fname_lists(`*fname_lists, save_path*`)`

This function saves the data in the *fname_lists* dictionary to csv files.

Parameters:

- **(dict)** (*fname_lists*) – and the value is a list of dataframes.
- **(str)** (*save_path*) –

Returns:

- *None*
- *Example*
- *save_fname_lists(fname_lists, “data/csv_files”)*

`ETL_Dataset_Transform_Functions.save_merged_df(`*merged_df, save_path*`)`

Saves the dataframes to a csv file.

Parameters:

- **merged_df** (*pandas.DataFrame*) – Dataframe to be saved.
- **save_path** (*str*) – The path to save the csv file.

Return type: None

`ETL_Dataset_Transform_Functions.time_stretch(`*file_path, stretch_factor*`)`

Apply time stretching to an audio file.

This function reads an audio file located at the specified file path and applies time stretching to the audio signal. Time stretching modifies the duration of the audio without changing the pitch. The resulting time-stretched audio signal is returned.

Parameters:

- **(str)** (*file_path*) –
- **(float)** (*stretch_factor*) –

Returns: **numpy.ndarray**

Return type: Time-stretched audio signal.

`ETL_Dataset_Transform_Functions.trim_audio_files(`*src_dir, dst_dir, chunk_length*`)`

Trim the audio files in *src_dir* into smaller chunks of *chunk_length* seconds and save them in *dst_dir*.

Each audio file in *src_dir* will be divided into chunks of *chunk_length* seconds and saved in a new folder with the same name as the original file in *dst_dir*. Each chunk will have a unique filename consisting of the original filename followed by an underscore and a chunk index.

Parameters:

- **src_dir** (*str*) – The path to the source directory containing the audio files.

- **dst_dir** (*str*) – The path to the destination directory where the trimmed audio chunks will be saved.
- **chunk_length** (*float*) – The length of each audio chunk in seconds.

Return type: None

Notes

The function assumes that all audio files in *src_dir* are in WAV format.