Binary Relevance Run Through

Data Preprocessing

Main Script:

- data_preprocessing_binary_relevance.py
    - Folders Needed
        - An output folder for each emitter in the dataset
            - This will hold the sorted training, validation and testing sets for each emitter
    - Variables to change
        - path_mat_data: set this to the path where the spectrograms are stored
        - path_data_store: set this to where the spectrogram HDF5 files are going to be stored
        - path_data_store(emitterID): set this to where the training, validation, and test sets will be stored for that particular emitter.
        - positive(emitterID) : Input the list of emitter combinations that will make up the data for the positive training class
        - negative(emitterID): Input the list of emitter combinations that will make up the data for the negative training class

Helper Functions:

- read_mat_binary_relevance in read.py
    - This will read in all the spectrograms in the path_mat_data and then store the HDF5 files made in the file path specified in the path_data_store
- form_binary_relevance_datasets in datasets.py
    - This will form the training, validation and testing sets for each of the emitter in the dataset and then store the HDF5 file in the file path specified by path_data_store(emitterID)
    - Both the target_train_val_test and other_train_val_test variable needs to be updated to match the number of combinations that are specified in the positive(emitterID) and negative(emitterID) variable respectively.
    - This will print to the console which spectrogram combinations were not added to the different datasets

Network Training

Main Script:

- network_train_eval_binary_relevance.py
    - Uses the HDF5 files made in the data preprocessing section
    - Builds and trains the network
    - Variables to change:
        - Target_emitter: Set this to the emitter that the network will be training to recognize
        - Path_data: Set this to the folder path where the _data_notes.txt file is stored

Helper Functions:

- train_eval_binary_relevance in train_eval.py
    - This is the main function that trains the networks. Nothing should need to be changed between runs.

Network Evaluation

Main Script:

- binary_relevance_evaluation.py
    - This script will evaluate the binary relevance network on a set of spectrograms and give a running accuracy as well as an overall accuracy
        - The overall accuracy is a one or none deal. It only counts a prediction as right if all the emitters are identified in the data.
    - Variables to change
        - data_path: Set this to the file path where the test spectrograms are stored
        - Model_paths: Set each item in the list to the file path where each trained emitter network is stored
        - classNames: Set each item in the list to the ID for the emitters. Make sure that the names are in the same order that the networks are in.

Helper Functions:

- evaluate_binary_relevance in eval_bin_rel.py
    - This sorts through the spectrograms in the test folder, formats them, send them through the binary relevance network, and keep track of the results from the network.
- networkDataPreprocessing in utils.py
    - This function formats either a .mat file or a .jpg file into the dimensions and formats needed to be run through the network.
- binary_relevance in binary_relevance_network.py
    - This function is where the binary relevance network is implemented. It takes the data and then runs it through each different binary network and then concatenates the results to get a final classification of the data.

Procedure

1. Update and run data_preprocessing_binary_relevance.py
2. Update and run network_train_eval_binary_relevance.py
3. Update and run binary_relevance_evaluation.py