Greek Music Sentiment Analysis

Michael Zouros

Problem to Solve

- Audio (song) classification to a specific Sentiment according to the Valence-Arousal 2D Model
- Implementation in Python3 with the use of various libraries (OpenCV, Numpy, Matplotlib/Seaborn, Pydub/Librosa, Sklearn/Keras)

- Github repo: https://github.com/mzouros/dl_gmsa

Data Collection

- Approximately 10k Greek songs were collected spanning from the Interwar Period (1920-1940) and Greek Junta / Dictatorship (1967-1974), until today (2021)
- The songs collected were in various forms (.mp3, .wma, .wav) and their aggregated size exceeds 120 GB

- Initial Dataset: https://raw.githubusercontent.com/mzouros/dl_gmsa/main/Greek_Songs_List.txt

Data Preparation

- Data Cleanup:
  - Remove any unnecessary files (.txt, .jpg, .zip, etc)
  - Delete songs with generic names (eg. Track01)
  - Delete duplicate songs
  - Try and narrow down live performance songs
- Data Matching:
  - Create folders for each Singer OR Songwriter OR Lyricist OR Producer (according to Spotify's registries) and assort each song to a single specific folder (locally)
  - Create the same folders on Spotify, then search which of the songs in our DB is a registry on Spotify. For each matching, insert the song to its corresponding folder
  - 1 by 1 matching each song to its Spotify counterpart. Rename each song of the dataset according to Spotify's Track name registries

Data Validation & Feature Extraction

- Data Validation:
  - Deal with bad matching / misspellings
  - Validate Playlist names (Spotify) against Album names (locally)
  - Validate Playlist track names (Spotify) against Album track names (locally)

- Feature Extraction:
  - Feature extraction via Spotify's API and export data to .csv
  - Some of the features extracted were Danceability, Energy, Valence, Liveness and Loudness

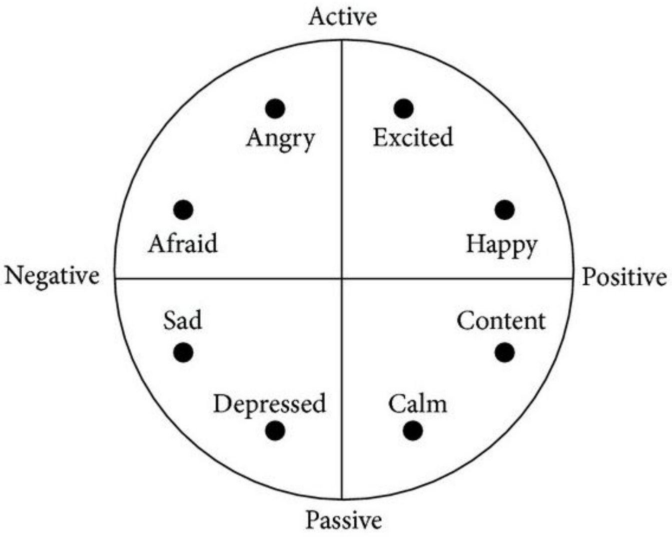
- Song Features: https://github.com/mzouros/dl_gmsa/blob/main/Spotify_Tracks.csv

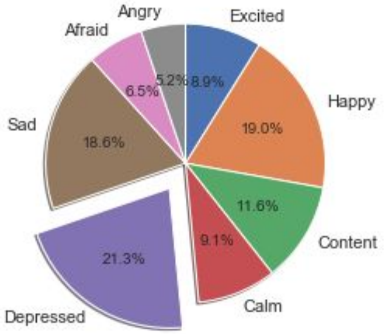
Data Preprocessing

- Transform .mp3 and .wma files to .wav
- Resample all tracks to 8k Hz (or keep their original frequency) and to monophonic (mono) sound
- Create songs' sentiments classification logic, according to Arousal-Valence Model
- Classify each song to a specific sentiment according to its values from the .csv file

Valence-Arousal Model & Classified Songs

We classify each song of our dataset to an emotion, according to the Valence-Arousal dimensional model

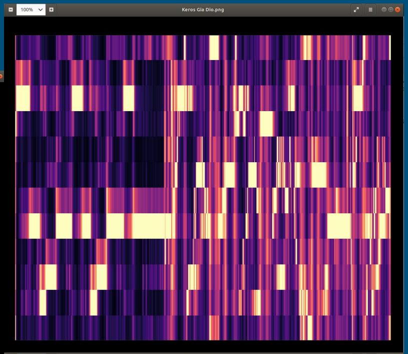


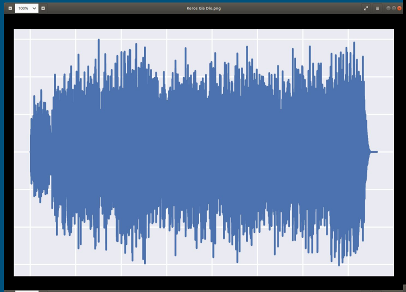


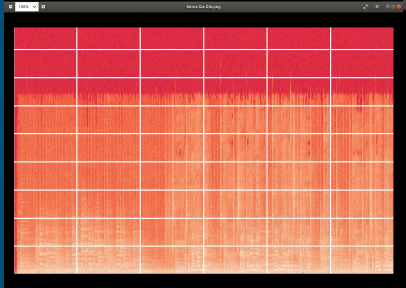
```
Excited: 463
Happy: 991
Content: 605
Calm: 476
Depressed: 1113
Sad: 970
Afraid: 341
Angry: 270
```

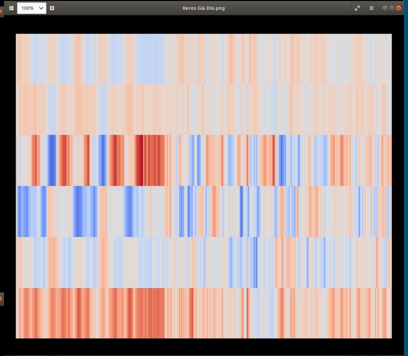
Data Exploration

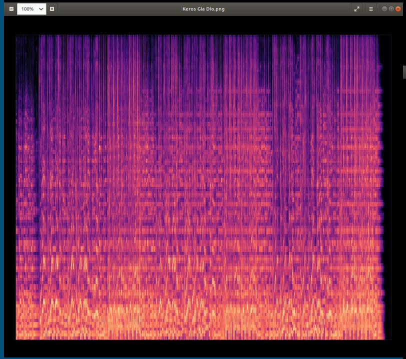
- Optical:
  - Waveforms
  - Spectograms
  - Mel Spectograms
  - Chromas
  - Tonnetz
  - CENS
- Features:
  - MFCCs
  - STFTs
  - ZCRs

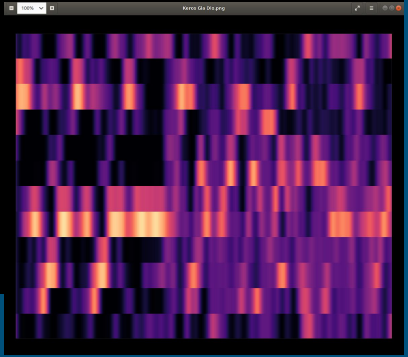








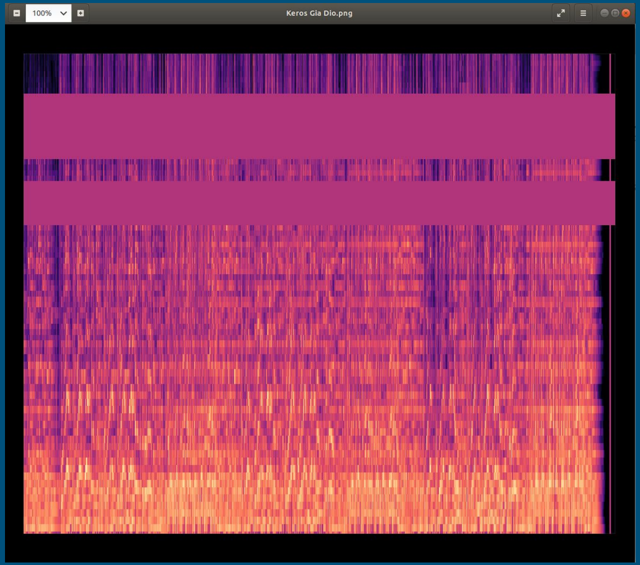




Data Revision & Augmentation

After experimenting with the initial dataset:

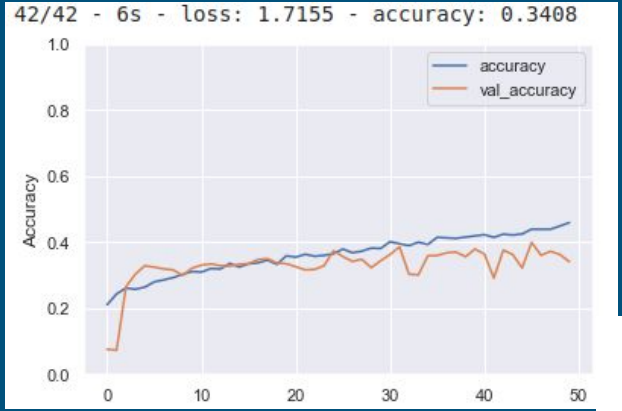
- Consider more data (via SpecAugmentation - Frequency Mask)
- Reconsider total of images per sentiment category (500 images for each category - 4k images), so to have the perfect balanced set
- Reconsider image size from 128x128 to 256x256



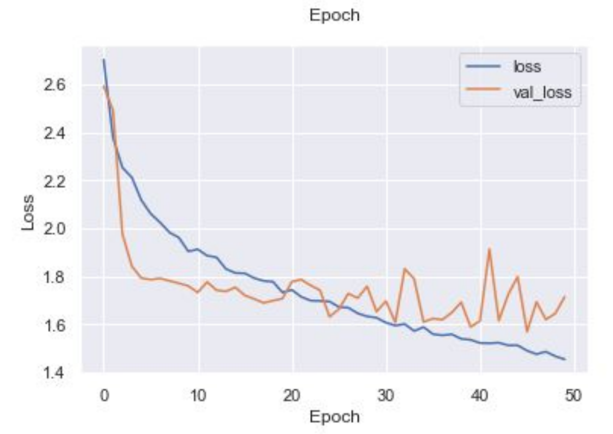
CNN Implementation

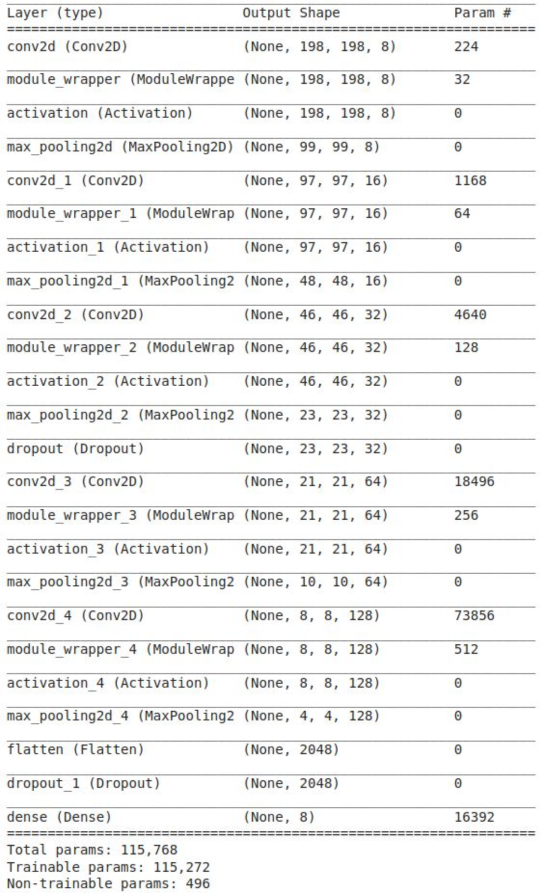
- Lots of different architectures have been tested, resulting on similar results (<0.4 validation accuracy)
- In all the architectures our model seems to not be able to learn after a while
- Tried most of overfitting avoidance techniques (kernel regularization, batch normalization, dropout, ES)
- Tried with different model sizes (layers, nodes), batch sizes, kernel & stride sizes, dropout values, number of epochs
- Tried with a perfect balanced set of 500 samples for each sentiment (4k samples total) - after data augmentation

CNN Results (8 labels)



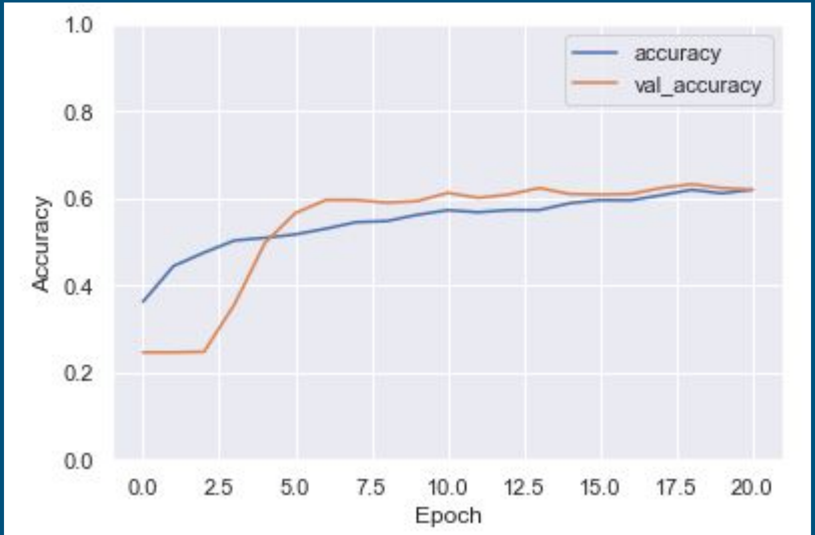
```
42/42 - 6s - loss: 1.7155 - accuracy: 0.3408
```

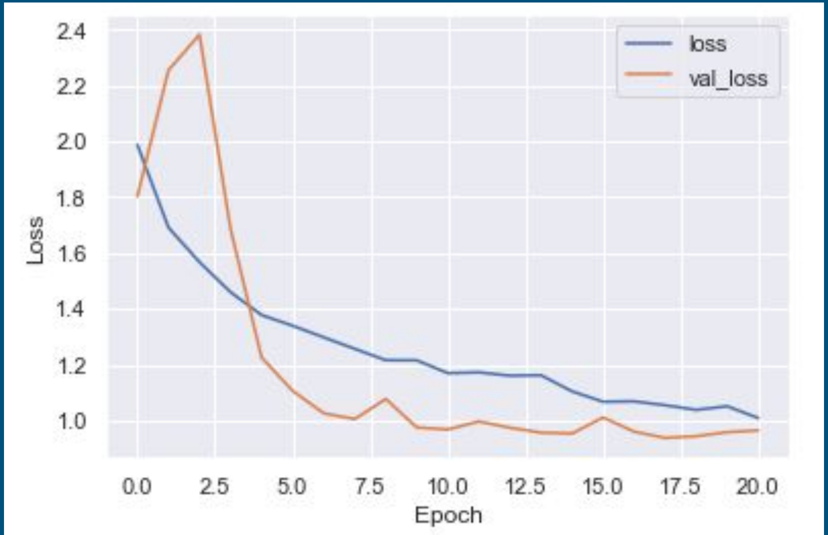




```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 198, 198, 8)       224
_________________________________________________________________
module_wrapper (ModuleWrappe (None, 198, 198, 8)       32
_________________________________________________________________
activation (Activation)      (None, 198, 198, 8)       0
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 99, 99, 8)         0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 97, 97, 16)        1168
_________________________________________________________________
module_wrapper_1 (ModuleWrap (None, 97, 97, 16)        64
_________________________________________________________________
activation_1 (Activation)    (None, 97, 97, 16)        0
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 48, 48, 16)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 46, 46, 32)        4640
_________________________________________________________________
module_wrapper_2 (ModuleWrap (None, 46, 46, 32)        128
_________________________________________________________________
activation_2 (Activation)    (None, 46, 46, 32)        0
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 23, 23, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 23, 23, 32)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 21, 21, 64)        18496
_________________________________________________________________
module_wrapper_3 (ModuleWrap (None, 21, 21, 64)        256
_________________________________________________________________
activation_3 (Activation)    (None, 21, 21, 64)        0
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 10, 10, 64)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 8, 8, 128)         73856
_________________________________________________________________
module_wrapper_4 (ModuleWrap (None, 8, 8, 128)         512
_________________________________________________________________
activation_4 (Activation)    (None, 8, 8, 128)         0
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 4, 4, 128)         0
_________________________________________________________________
flatten (Flatten)            (None, 2048)              0
_________________________________________________________________
dropout_1 (Dropout)          (None, 2048)              0
_________________________________________________________________
dense (Dense)                (None, 8)                 16392
=================================================================
Total params: 115,768
Trainable params: 115,272
Non-trainable params: 496
```

CNN Results (4 labels)





Discussion

- Problems during implementation:
  - 1 by 1 matching very time consuming
  - Lots of duplicates required extra work (eg Stavros Xarxakos and Stavros Ksarhakos, both indicating the same Artist)
  - Spotify classifies as Artists singers, songwriters, lyricists and producers. Need to take all of those into consideration when searching if a specific song in our dataset exists as a registry in Spotify
  - Annotating an audio recording is challenging. How many emotions should we define to recognize?
  - Emotions are subjective, people would interpret it differently. It is hard to define the notion of emotions.
  - Audio Analysis through image classification tend to have bad accuracy results (<0.5) (https://towardsdatascience.com/whats-wrong-with-spectrograms-and-cnns-for-audio-processing-311377d7ccd)

Future Work

Experiment with:

- Bigger Dataset (10k+ images)
- Bigger image size (maybe 512x512)
- Transfer Learning / Pre-trained Classifiers
- Classification in 4 sentiments (Happy, Calm, Sad, Angry) instead of 8
- Different NN architectures (eg CNN-RNN in parallel)
- Time and Frequency Domain features instead of images
- Instead of feature extraction through Spotify's API, use an Annotation App, so volunteers individuals can annotate according to their belief (eg. AnnoEmo app)

Food for Thought:

- How Spotify quantify its track's variables is not clear, but usually in fuzzy logic, a team of specialists define the pertinence degree between qualitative aspects
- A happy song may have sad lyrics and vice versa. A more realistic song sentiment analysis would require both text and sound analysis and classification.

References

- https://github.com/tyiannak/basic_audio_handling/blob/master/notebobok.ipynb
- https://github.com/tyiannak/multimodalAnalysis/tree/master/audio
- https://betterprogramming.pub/how-to-extract-any-artists-data-using-spotify-s-api-python-and-spotipy-4c079401bc37
- https://www.linkedin.com/pulse/how-spotify-knows-your-feelings-almost-exploratory-data-martins/
- https://hackernoon.com/how-to-use-machine-learning-to-color-your-lighting-based-on-music-mood-bi163u8l
- https://link.springer.com/article/10.1007/s11042-019-08192-x
- https://towardsdatascience.com/music-genre-recognition-using-convolutional-neural-networks-cnn-part-1-212c6b93da76
- and many more..

Thank you!