BRIEF DOCUMENTATION FOR TEAM PANIKEE MACHINE LEARNING MODEL

We have created three models or versions of our model for the purpose of detecting specific keywords in order to trigger a popup displayed in Android. Each model has mostly different dependencies and ways to process the audio input, and in the end, our team used the latest version (V.3.0) of our model to deployed to our Android app. Here are some brief explanations of each version of the model.

Model V.1.0 ⟶ Used python_speech_features to extract audio input (.wav) and convert it to an image. After that, we classify the image using CNN architecture whether the extracted feature in the image matches the keyword we looking for. The reason we're not using this model is that there is no python_speech_features library in Android and so we can't implement it.

Model V.2.0 ⟶ Used librosa to replace python_speech_features, and convert audio input into a spectrogram. However, when implemented to Android using Java Programming Language, the only library that is equivalent to librosa is jlibrosa, and it is still in its early development, so we encountered some problems and decide not to use it.

Model V.3.0 ⟶ Used transfer learning method from an existing model, in this case, a custom audio model based on the model topology of the TensorFlow.js Speech Commands model. Just like Model V.2.0, this model converts audio input into a spectrogram, but the model assumes that audio input must be in .wav format. Each .wav file corresponds to an example. Each .wav file is mono (single-channel) and has the typical pulse-code modulation (PCM) encoding. The duration of each wave file should be 1 second or slightly longer. Because of the limited number of our audio dataset, we also used a public audio dataset, Speech Commands Data Set v0.01, which can be downloaded at http://download.tensorflow.org/data/speech_commands_v0.01.tar.gz.

For Model V.3.0, the source code file is in ModelV3 directory, which is Panikee.ipynb. The functions are self-explanatory and some contain docstring. There are comments for blocks of code that are harder to comprehend by just looking at them. So, for this documentation, we wouldn't explain in detail the source code, but we would include important points to replicate our work:

- The audio dataset should match with what mentioned in the previous paragraph.
- Some modules that are used in the source code:
  - glob, for retrieving files/pathnames matching a specified pattern. Default module in python.

    - json. Default module in python.
    - os. Default module in python.
    - random. Default module in python.
    - librosa, for processing audio and music in python. Version 0.8.0.
    - matplotlib.pyplot. Version 3.3.4.
    - numpy. Version 1.18.5.
    - scipy.signal. Version 1.4.1.
    - scipy.io.wavfile. Version 1.4.1.
    - tensorflow. Version 2.3.0 or up.
    - tensorflowjs. Version 1.7.0 or up.
    - tqdm.tqdm, for estimating remaining time and creating a progress bar in python. Version 4.59.0.
- The Speech Commands Dataset is used as negative category classes for our model because of our limited audio dataset and resources. But for further development, we plan to increase the number of our dataset and in return use less of the Speech Commands Dataset.