Is that a Snoring Sound?

The University of Texas at San Antonio™

Team 5 – Luis B. Reyes and David Guo

Problem statement - Snoring prevents restful, healthy sleep. Over time, this condition can cause serious complications such as sleep apnea which pauses breathing hundreds of times during the sleep.

Proposal – Deep learning (DL) has been widely used to deal with the unstructured data such as image, sound, text, etc. In this proposal, we will apply a DL algorithm to develop an multi-task model that take the sound recording as input and classify the sound to determine if the sound is a Snoring. We will showcase the application of the model by implementing it in a Raspberry Pi with recording device that will continue to take the sound input and output the binary classification of 0 (no snoring) /1 (snoring). Together with other medication devices, patient can be treated with necessary administration/action.

Ethical Concern – no ethical concern on this application given the implementation of model is on an edge device with no data exchange proposed.

Stakeholder – patient who has medium or severe snoring issue, healthcare providers such as hospital, clinic, nursing house.

Primary Design – Tensorflow will be used as the end-to-end platform to test, fine-tune, and deploy the DL model.

Development Stage -

- Data collection, feature extraction, and model input pre-processing
- Test model building considering the trade-off of model complexity and performance
- Fine-tune model to boost prediction accuracy on binary outcome of snoring or not.
- Test model implementation on edge device (Raspberry Pi) that takes batch/streaming recording for snoring prediction.
- Achieve best model performance with optimized parameters.

Data collection:

1) UrbanSound8K with 10 classese from Kaggle @ https://www.kaggle.com/datasets/chrisfilo/urbansound8k

2) Snoring sound data:

- Shared by Author of "A real-time snore detector using neural networks and selected sound features", by S.A. Mitilineos et al., Presented at 2nd International Electronic Conference on Applied Sciences, 15–31 October 2021, Eng. Proc. 2021, 11(1), 8; https://doi.org/10.3390/ASEC2021-11176
- Wav files characteristics: 48 kHz, 24 kbps, raw (unfiltered) recordings
- 2500 Wav files for snoring and non-snoring, respectively.

3) Merge 2 sets of data to form a 11 classes of training data

Data Pre-processing:

1) Feature extraction
- signal in the time-frequency domain by computing discrete Fourier transforms
- Mel-frequency cepstral coefficients
- chromagram from a waveform
- Melspectrogram
- Energy contrast
- Tonal centroid feature

2) Model Input
- Stack sound features to form a 2-D model input with fixed columns of 161 and varied length of row depending on the length of recording
- Map 2D input with a unique label of length of rows prepared from a single wav.

DL Modeling Approaches to be explored:

1) Multi-Layer Perceptron (MLP)

2) Multi Layer DL

3) Convolution 1D/2D DL

4) RNN model – LSTM / GRU

Optimization Efforts:

- Optuna will be used for fine-tuning hyperparameters, including learning rate, optimizer, loss functions, epoch/step, etc.

Explainable AI Efforts:

- Captum will be used to explain the model

Model Implementation Plan

1) Prepare Raspberry Pi 4B w/ Linux OS Bookwarm (1.1G)
2) Connect Raspberry Pi w/ sound recording device (microphone)
3) Setup virtual environment for sound classification inference
4) Quantize the model to a small model to run inference in an efficient way
5) Convert TensorFlow model to TensorFlow Lite w/ converter Python API on CPU
6) Test runtime on single/streaming sound wave input

Appendix

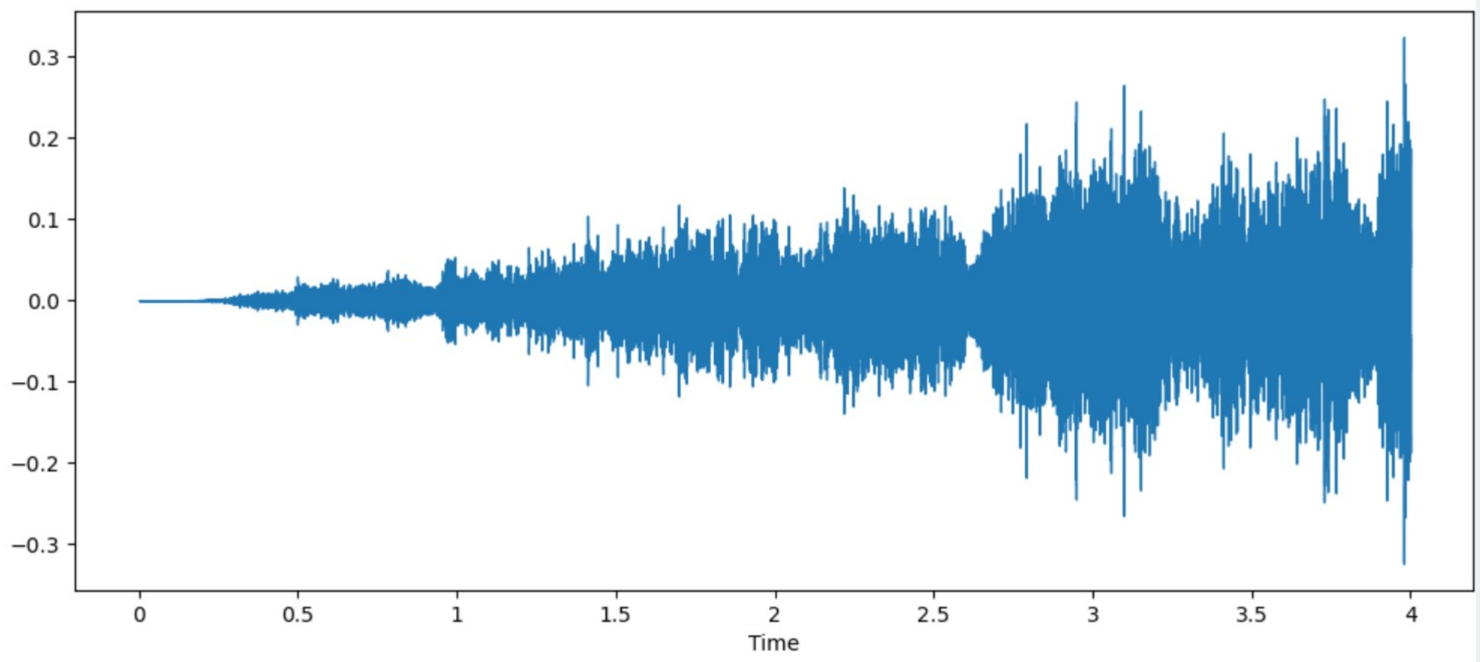
Explore Input Data

```
filepath = "./UrbanSound8K/audio/fold1/99180-9-0-0.wav"
ipd.Audio(filepath)
```

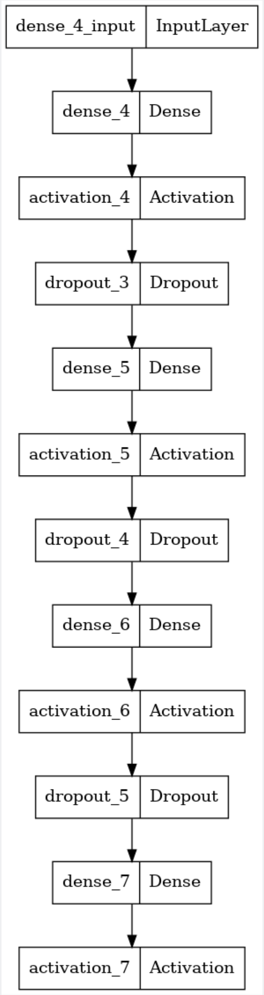
0:00 / 0:04

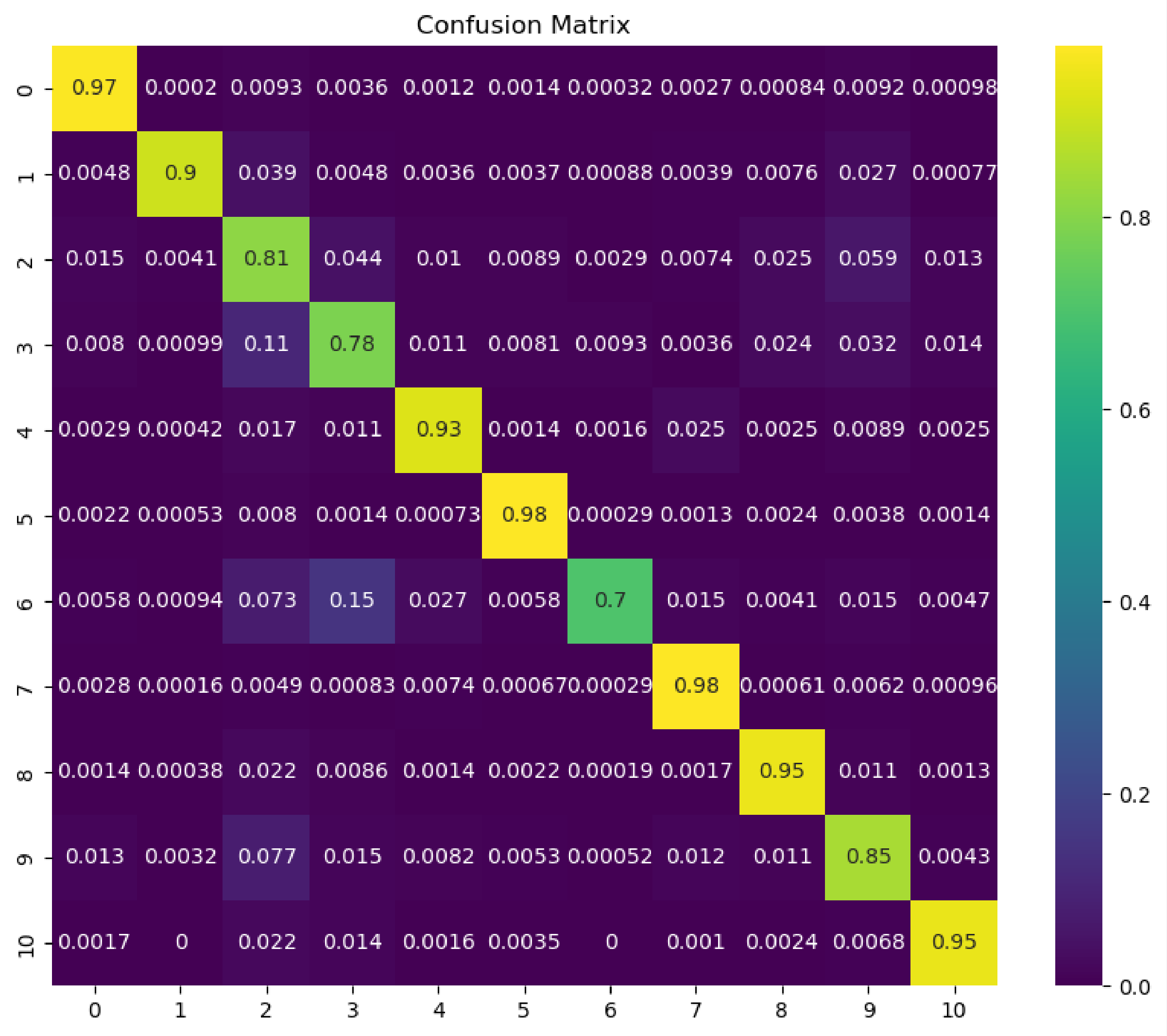
```
data, sample_rate = librosa.load(filepath)
plt.figure(figsize=(12, 5))
librosa.display.waveshow(data, sr=sample_rate)
```

```
<librosa.display.AdaptiveWaveplot at 0x7f5b058d3050>
```



Model Architecture and Performance (Class 10 is Snoring)





Model Inference

```python
def inferenceAudio(wav_file_path):
    global model, sc
    mfccs, chroma, mel, contrast, tonnetz = extract_features(wav_file_path)
    features = np.hstack([mfccs, chroma, mel, contrast, tonnetz])
    cls_prob = list(np.argmax(model.predict(np.array(sc.transform(features))), axis=1))
    major_cls = max(set(cls_prob), key = cls_prob.count)
    print(f"The predicted class is {major_cls} as {cls_annotation[major_cls]}.")
```

```python
inferenceAudio("./UrbanSound8K/audio/fold1/99180-9-0-0.wav")
```

```
6/6 [==============================] - 0s 1ms/step
The predicted class is 9 as street_music.
```

```python
filepath = "./UrbanSound8K/audio/fold1/99180-9-0-0.wav"
ipd.Audio(filepath)
```

0:00 / 0:04