1. Leggere articolo TUTTI
2. Power point articolo in inglese G A
3. Progetto

* Studiare il dataset: Cioe capire quanti audio, di che forma, di che durata media ect… (Aldo)

Dataset fatto da 24 audio contenenti la stessa scena acustica con 6 classi da classificare(labels: brakes squeaking, car, children, large vehicle, people speaking, people walking).

L’audio più lungo ha durata 300 secondi, il più breve 181 secondi e hanno una media di 230 secondi.

La cartella evaluation setup contiene dei file txt. Sono di fatto le divisioni del dataset per train,evaluation e test già pronte per la cross-validation. Sono presenti 4 folds.

Formato file .txt per evaluate e train: audio, inizio evento, fine evento, label

Evento2 inizio evento , fine evento, label

Capire se usare più modelli tra i vari (nel caso il processo va fatto N volte)

* Scegliere il modello fra CNN14(Aldo), WavToVec(Gio), AST(Aldo) , HuBERT, VGGish(Ciccio)

1. Studio teorico base
2. Capire che input vuole bene
3. Studiare il modello a livello di implementazione
4. Capire che vantaggi ha e se conviene nel nostro caso

* Cercare possibili pre-trained per il modello scelto
* Scegliere che pre-processing fare e che struttura vogliamo avere il dataset(input per la rete)
* Capire che label abbiamo
* Fare il Pre-processing e avere l’input della rete
* Scegliere la loss function/le loss function e implementarle
* Scegliere gli hyperparametri (lr, ?)
* Implementare il fine tuning: UrbanSound8k
* Allenare la rete
* Fare inferenza (100%,50%,10%)
* Vedere risultati
* Fare modifiche

1. Relazione Progetto in Inglese
2. Power point progetto tarocco e veloce in Inglese

Train a network (can be pretrained, it should be fintuned) to perform acoustic scene classification on TUT-17. You should perform the training 3 times and evaluate how the performance change using: 100% of training data, 50% of training data, 10% of training data. If the whole dataset is computationally expensive using a subset is possible

Models: PANNs model/Wav2Vec2.0/HuBERT/AST

Dataset: TUT-17

**Proprietà modelli**

**CNN14**:Rete convoluzionale creata per la classificazione audio.

* Accetta come input dati in formato: (batch\_size, num\_channels, height, width).
* Accetta come input:

1. Mel\_Spectrogram
2. STFT(ampiezza delle freq nel dominio del tempo)
3. chromagram(energia delle 12 classi di pitch (o cromatiche) in un segmento temporale
4. CQT(freq variabile, più alta per le basse freq. e più bassa per le alte freq, come percepiamo il suono noi)

* Ergo non funzionante con i dati waveform grezzi presenti nel mio codice, tuttavia si può fare un adattamento in tempi brevi (circa 2-3 ore al più).

**AST:** Modello transformer che può dunque catturare le dipendenze nell’audio.

* Accetta come input letteralmente i Mel\_spectrogram, con input dati in formato:(batch\_size,numero\_bande\_mel,numero\_frame temporali).
* Seppur come per CNN14 non è adatto ad avere in input i risultati del mio algoritmo, al tempo stesso è adattabile e ben si confà al mio aver tenuto in considerazione a che evento ci riferiamo e l’ordine dell’evento. Tuttavia potrebbero esserci problemi per il query set avendo considerato eventi di fatto superframe più che eventi in sé.

**Wave2Vec:** Modello di rappresentazione audio che elabora un audio grezzo(waveform) e apprende rappresentazioni in un latent space per riconoscimento vocale.

* Accetta come input gli audio grezzi, cioè i famosi campioni creati dal mio algoritmo, in formato 1D di dimensione dei campioni audio (o nel nostro caso delle frames)