

**Audio Recognizer**

*Sistema per il riconoscimento delle emozioni dalla voce*



dipartimento di informatica - sistemi ad agenti 21/22

**team:**

* antonio papeo 111111
* francesco sasso 715742
* cristian riefolo 111111
* rino 111111

**Immagine che contiene luce, sfocatura

Descrizione generata automaticamente**

1. **Introduzione**

AudioRecognizer è un sistema scritto in Python in grado riconoscere le emozioni espresse da un essere umano tramite la voce, utilizzando un modello addestrato per il riconoscimento audio.

Le emozioni rilevate dal sistema sono:

1. Dolore;
2. Sbadiglio;
3. Risata;
4. Respirare;
5. Russare;
6. Pianto di un Bambino;
7. Starnuto
8. Urla
9. Tosse

[INSERIRE FOTO RISULTATO DI UNA PREDIZIONE]

Figura 1: Esempio risultato riconoscimento audio

1. **Implementazione del modello**

Il modello utilizzato all’interno dell’applicativo è scritto in Python e, grazie ad un dataset appositamente suddiviso in Training, Test e Validation, permette di sfruttare il suo addestramento per il riconoscimento di determinate emozioni fornite in input.

Le librerie implementate sono:

* **Pandas**: utilizzata per lavorare con il csv contenente le informazioni del dataset e scaricabile con il comando *<< pip install pandas >>.*
* **Tensorflow**: utilizzata per effettuare operazioni nel modello come l’esecuzione dell’inferenza, scaricabile con *<< pip install tensorflow >>.*
* **Tensorflow\_hub:** utilizzata per caricare il modello ottenuto dal seguente link <https://tfhub.dev/google/yamnet/1>, scaricabile con

*<< pip install tensorflow\_hub >>.*

* **Tensorflow\_io**: utilizzata per impostare il giusto sample\_rate nell’audio caricato, scaricabile con *<< pip install tensorflow\_io >>.*

Immagine che contiene testo

Descrizione generata automaticamente

Figura 2: Funzione caricamento audio

La funzione della figura 2 verrà utilizzata dopo aver definito il modello ed aver inserito l’audio di cui si vuole effetturare il riconoscimento.

Essa permette di caricare il file appena immesso e di modificarne il sample rate, ovvero il numero di cicli in cui viene catturato l'audio (campionato) in un secondo, in modo da poterlo analizzare.

Immagine che contiene testo

Descrizione generata automaticamente

Figura 3: Mappatura della classe ed Esecuzione Inferenza

Nella figura 3 mostriamo come viene mappa la classe audio secondo una decodifica UTF-8 e di come viene effettuata l’inferenza sull’audio inserito in input.

Successivamente abbiamo inserito nel sistema le classi presente nel dataset ed abbiamo associato a ciascuna di essa un id, in modo tale da poter filtrare il dataset per categoria sfruttando pandas.

Le categorie inserite corrispondono alla suddivisione esistente nel dataset:

|  |  |
| --- | --- |
| **CATEGORIA** | **ID CATEGORIA** |
| DOLORE | 0 |
| PIANTO\_BAMBINO | 1 |
| RESPIRO | 2 |
| RISATA | 3 |
| RUSSARE | 4 |
| SBADIGLIO | 5 |
| URLA | 6 |
| TOSSE | 7 |

Dopo aver definito le classi procediamo con l’incorporamento dei dati:

Immagine che contiene testo

Descrizione generata automaticamente

Figura 4: Incorporamento dati

Successivamente procediamo con l’estrazione dei file audio presenti all’interno del dataset utilizzando il modello di estrazione dati presente in Yamnet, come possiamo vedere dalla figura 5.

Immagine che contiene testo

Descrizione generata automaticamente

Figura 5: Estrazione dati dal dataset

Come possiamo vedere dalla figura 6, abbiamo suddiviso i dati in Training, Test e Validation suddividendoli come 70% Training, 20% Validation e 10% Test.

La suddivisione, come vedremo successivamente nella Figura 9 , è stata rappresentata nel csv e successivamente nel codice, come possiamo notare nella figura 6, assegnando un id a ciascuna fold:

1 – Training

2 – Validation

3 – Test

Immagine che contiene testo

Descrizione generata automaticamente

Figura 6: Divisione dei dati nei relativi: Training, Test e Validation

Immagine che contiene testo

Descrizione generata automaticamente

Figura 7:Creazione,Addestramento ed esecuzione del modello

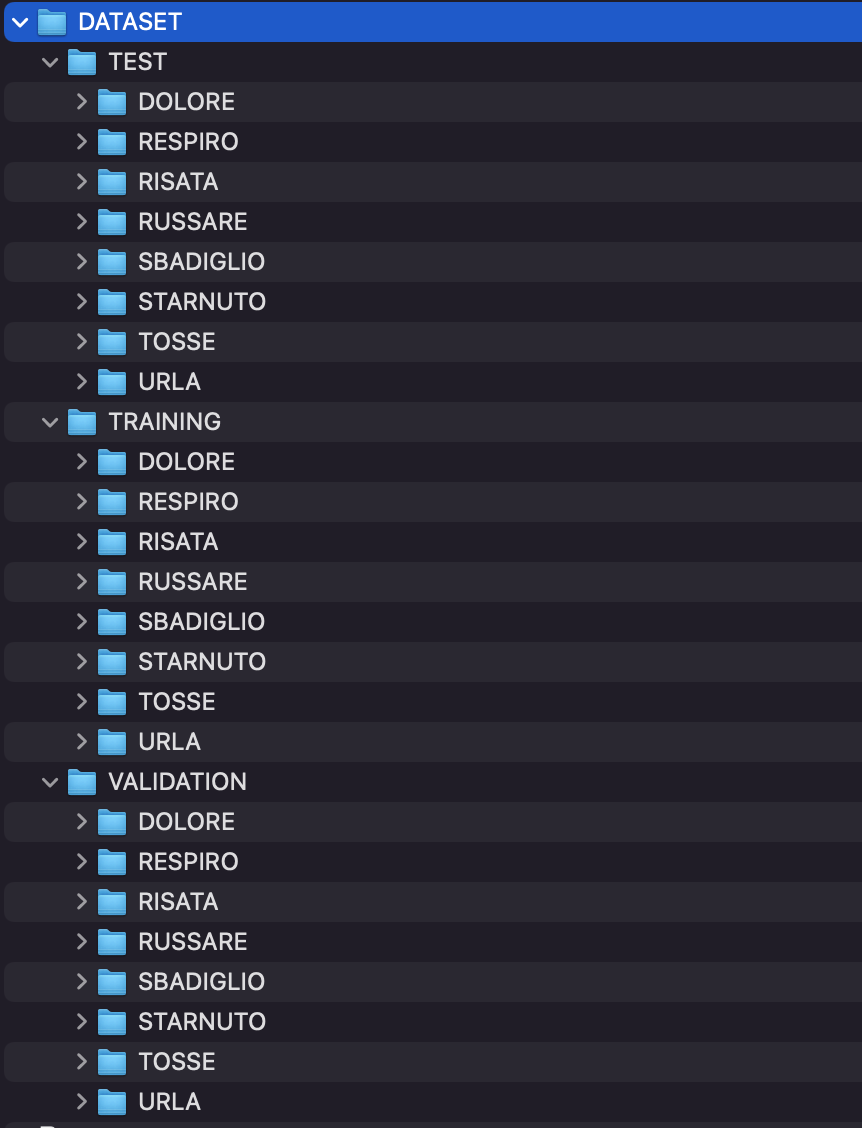
Infine, come possiamo vedere nella figura 7 abbiamo creato il modello sfruttando la libreria di Tensorflow e chiamandolo ‘my\_model’ ed abbiamo verificato quanto stesse migliorando nel tempo il suo apprendimento, dopo almeno 5 avvii, l’accuratezza del sistema.

Ed infine come risultato finale si ottiene la classe rispettiva all’audio inserito in input, permettendo cosi di comprendere la tipologia di emozione emessa dall’essere umano.

1. **Dataset**

Il dataset, come possiamo vedere nella figura 8 e 9, è suddiviso in cartelle in base alla categoria e a sua volta suddiviso internamento in base al relativo utilizzo:

* **Training**
* **Validation**
* **Test**

 Figura 8:Suddivisione cartelle Dataset

[INSERIRE SCREEN CSV]

Figura 9:Suddivisione csv Dataset

1. **Organizzazione Progetto**

L’organizzazione del progetto è avvenuta tramite l’applicativo Microsoft Teams, il quale ci ha permesso di lavorare in gruppo per l’intera fase di progetto nonostante le distanze di abitazioni.

Per la condivisione del progetto abbiamo sfruttato github il quale ci ha permesso di collaborare in maniera omogenea al progetto.

Per la comunicazione giornaliera su idee di progetto e sull’avanzamento delle varie fasi è stata utilizzata la piattaforma Whatsapp mobile.

Come ide per lo sviluppo del codice abbiamo scelto Intellij Idea vista la sua ottima implementazione con github il che ci permetteva un lavoro più semplice ed efficiente.

1. **Conclusioni Finali**

Ringraziamo per l’attenzione tenutasi nei confronti di AudioRecognizer da parte di tutto il Team.

Per poter scaricare il progetto da github è possibile accedere tramite il seguente link:

<https://github.com/AntonioPapeo6/ProgSysAg>