Voice Recognizer

Final Project

Stefano Cortellessa – 254260

L'idea di realizzare un riconoscitore vocale, mi è venuta in mente utilizzando l'assistente intelligente del mio telefono. Dovendo riconfigurare la mia voce in seguito ad aggiornamenti, ho notato che non avendo effettuato tale operazione in un ambiente aperto la mia voce non veniva riconosciuta facilmente e nel modo corretto. Da qui la curiosità di capire cosa ci fosse dietro al suo funzionamento e di come realizzare un sistema del genere con le varie tecniche di Machine Learning.
Per l'implementazione di tale progetto sono state utilizzate diverse librerie, tra cui *pandas* (per la gestione dei dataset)*, sklearn* (per l'utilizzo delle tecniche di Machine Learning) *e pyaudio* (per la registrazione delle voci e gestione delle stesse).

Lo scopo di questo progetto è quello di riuscire a riconoscere chi è la persona che ha parlato al microfono tra quelle salvate nel dataset, dando in output il suo nome.
Il primo passo è stato quello di configurazione di tutte le impostazioni di registrazione. Questo viene fatto dalla funzione *record()* che attiva l'utilizzo del microfono e crea un file di estensione '.wav' della voce registrata all'interno della cartella principale.
Successivamente, bisognava trovare un modo efficace per far in modo da confrontare correttamente la voce registrata al momento dell'esecuzione con le varie voci già presenti nel dataset.
A questo punto, il passo da eseguire è quello di:

- Prendere lo spettrogramma della registrazione e, attraverso la trasformata di Fourier che consente di dividere onde sinusoidali di ogni tipo in più sotto componenti che facilitano l'analisi, trasformare i valori corrispondenti in numeri reali ed eseguire la media tra i valori ottenuti dalla registrazione corrente e quelli delle persone salvate nel dataset (il più vicino sarà quello predetto).

Quello che deve essere fatto è trasformare (attraverso la Trasformata di Fourier) la registrazione creata attraverso la funzione *record()* in un caso più semplice che garantisce un'analisi meno complessa e più accurata.
Questo viene fatto dalla funzione *fourierConversion()* con l'ausilio della funzione fft (Fast Fourier Trasformation), la quale ritorna un array contenente numeri complessi corrispondenti ai valori reali dello spettrogramma in input.
A questo punto è stata effettuata la media tra i valori in output correntemente e quelli presenti nel dataset in quanto, in seguito a ricerche, è risultata la tecnica utilizzata dall'assistente vocale del mio telefono.

A questo punto, una volta implementate la parte di registrazione e campionamento della voce, tali dati sono stati salvati all'interno del dataset che è stato creato man mano attraverso varie esecuzioni del software, il quale ad ogni esecuzione, come specificato successivamente, in base alla funzionalità selezionata si và ad aggiungere al dataset il nome della persona che ha parlato associato ai valori della sua voce campionata e trasformata.

Inoltre, per far si che la predizione di una determinata persona sia più accurata sono state fatte più di una registrazione per ogni persona in quanto la frequenza della voce cambia a seconda della parola pronunciata, e quindi effettuando e campionando più registrazioni si allarga il range delle parole che possono essere pronunciate per far si da essere riconosciuti correttamente.
Successivamente sono stati quindi creati training e test Set splittando il dataset originale in percentuali 80% training e 20% test. Al momento della query inoltre, viene creato un querySet che viene utilizzato appunto per interrogare il dataset e indovinare la persona che ha parlato.

A questo punto viene creata la rete neurale che viene allenata e testata rispettivamente con trainingSet e testSet.

Il software è stato implementato in modo tale da poter inserire una parola chiave che viene stampata e sulla console e in base a quello desiderato è in grado di eseguire le seguenti differenti funzionalità:

- *ADD*: Funzionalità che permette di aggiungere al dataset una nuova persona. Viene inizialmente chiesto di inserire il nome manualmente e successivamente viene attivato il microfono che registra la sua voce che, campionata come spiegato precedentemente, viene inserita nel dataset associata al nome inserito.

- *GUESS*: Funzionalità che permette di indovinare chi parla al microfono. Viene subito attivato il microfono che registra la voce di chi parla. Successivamente con questa voce viene creato il querySet per interrogare il dataset e in output viene fornito il nome che viene predetto.
    - Se il nome predetto è corretto, *il campionamento effettuato viene aggiunto al dataset in modo tale da aumentarne la precisione di predizione, avendo a disposizione più frequenze della voce di una determinata persona*.
    - Se il nome predetto è sbagliato, semplicemente il metodo viene rieseguito.

- *VOCABULARY:* Funzionalità che permette di aggiungere una nuova parola al dizionario, inserendo prima manualmente la parola che si vuole aggiungere, e successivamente pronunciandola al microfono questa viene campionata ed inserita nel dataset delle parole chiamato 'dictionary'.

- *SAID*: Funzionalità che permette di riconoscere la parola pronunciata tra quelle presenti nel 'dictionary' creando il querySet come nel caso di GUESS e interrogando tale dataset.

- *PREDICT*: Funzionalità che permette di calcolare l'accuracy_score della rete neurale.

- *STOP*: Funzionalità che permette di fermare l'esecuzione del metodo.

I metodi che eseguono tali funzionalità nel codice sono rispettivamente *add(), guess(), vocabulary(), said() & predict().*

Al momento del primo allenamento, come visibile in figura 1 è presente una perdita circa uguale a 14.

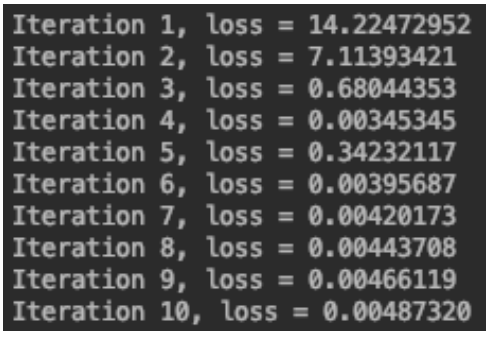

Figura 1: Perdita per ogni iterazione

Esempi di esecuzione di tutte le funzionalità descritte:

ADD:

```
Digit 'Add' or 'Predict' or 'Guess' or 'Said' or 'Vocabulary': add
--------------------------------------------------------------------
Insert the person's name: Stefano
Recording...
Registration made!
------------------
Checking/Preparing trainingSet and testSet...
Done!
------------------
I'm training, just a second...
Done!
```

GUESS:

```
Digit 'Add' or 'Predict' or 'Guess' or 'Said' or 'Vocabulary': guess
--------------------------------------------------------------------
Recording...
Registration made!
------------------
I'm creating the QuerySet, just a second...
Done!
--------------
I'm thinking..
It's Stefano!
Am I rigth? Write 'y' or 'n': y
--------------------------------
I'm updating the dataset, just a second...
The dataset is updated!
```

VOCABULARY:

```
Digit 'Add' or 'Predict' or 'Guess' or 'Said' or 'Vocabulary': vocabulary
--------------------------------------------------------------------
Which word do you want to add? sono io
Recording...
Registration made!
------------------
I'm training, just a second..
Done! wai while I'm creating a network dump
```

SAID:

```
If you want to stop, write 'Stop'
Digit 'Add' or 'Predict' or 'Guess' or 'Said' or 'Vocabulary': said
-----------------------------------------------------------------------
Recording...
Registration made!
------------------
You said ciao!
```

PREDICT:

```
Digit 'Add' or 'Predict' or 'Guess' or 'Said' or 'Vocabulary': predict
-----------------------------------------------------------------------
I'm predicting, just a second...
Accuracy: 60.0%
```

Risultati:

L'implementazione descritta è stata infine testata con diverse reti neurali calcolando il valore di accuracy per ogni struttura differente e contando il numero di volte che effettivamente riconosceva la persona che parlava al microfono su un numero di 5 tentativi per ogni diversa struttura.

- 1 hidden layer da 2000 nodi:
  - Accuracy: 60%
  - Tentativi riusciti/totali: 4/5
- 2 hidden layer da 2000 nodi:
  - Accuracy: 40%
  - Tentativi riusciti/totali: 3/5
- 3 hidden layer da 2000 nodi:
  - Accuracy: 40%
  - Tentativi riusciti/totali: 3/5
- 1 hidden layer da 1000 nodi:
  - Accuracy: 20%
  - Tentativi riusciti/totali: 2/5
- 2 hidden layer da 1000 nodi:
  - Accuracy: 27%
  - Tentativi riusciti/totali: 2/5
- 3 hidden layer da 1000 nodi:
  - Accuracy: 60%
  - Tentativi riusciti/totali: 3/5

Conclusioni:

Quello che può essere notato dalle varie esecuzioni, mostrate e non, del software da me implementato è che non si riesce ad avere una accuratezza che vada oltre il 62% anche cambiando la struttura della rete neurale, aggiungendo strati e numero di nodi. Questo non è un valore ottimo, ma non disastroso se si tengono in considerazione tutti i fattori di inquinamento della voce ad ogni registrazione che viene effettuata come ad esempio il rumore di sottofondo, la qualità del microfono, e le tecnologie a mia disposizione, che, ovviamente, non sono professionali e non mi permettono di effettuare un'esecuzione 'perfetta'.

In generale tornando a quello che era il mio quesito iniziale, ovvero se lo spazio in cui viene effettuata la registrazione per il riconoscimento influisse sulla predizione futura, posso affermare che questo è vero.

Infatti, ho eseguito vari 'tipi' di registrazioni: mentre ero in uno spazio chiuso, in uno spazio aperto e in uno spazio occupato da altre persone notando che quello nello spazio aperto è quello che dà una predizione più precisa in quanto le frequenze delle voci non sono alterate da eco o altri fattori.