INFORMATICA MUSICALE

UNIVERSITA' DEGLI STUDI DI CATANIA
DIPARTIMENTO DI MATEMATICA E INFORMATICA
LAUREA TRIENNALE IN INFORMATICA
A.A. 2018/19
Prof. Filippo L.M. Milotta

ID PROGETTO: 06

TITOLO PROGETTO: Common Voice e DeepSpeech

AUTORE 1: Camonita Dario

Indice

1. Obiettivi del progetto

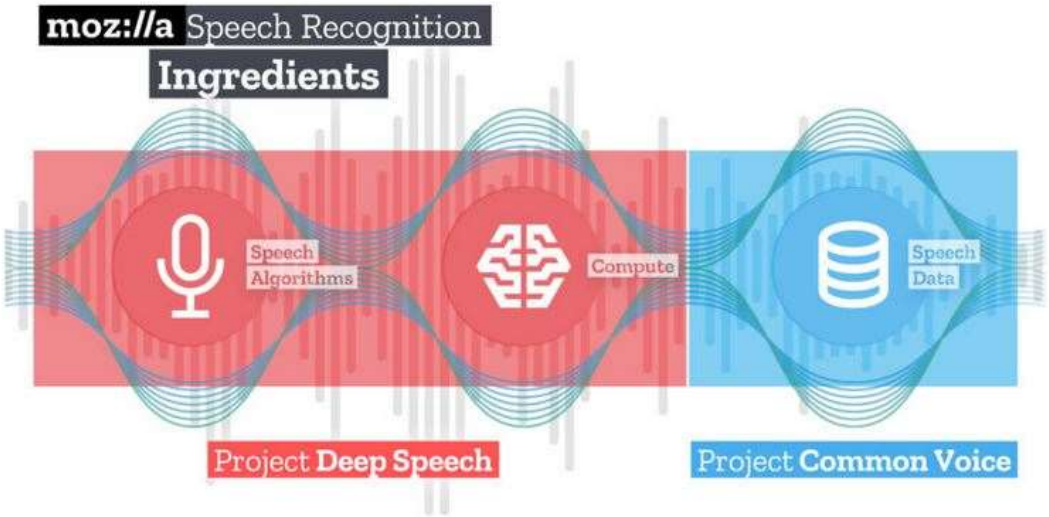
1.1 Introduzione al progetto Common Voice e DeepSpeech

Common Voice è un progetto di crowdfunding di Mozilla incentrato sul raccoglimento di dati testuali e vocali basati su lingue specifiche, in modo da per poter creare un riconoscitore vocale basato su un dataset (per l'appunto, Common Voice) e un modello di pubblico dominio per tutte le lingue che partecipano al progetto. Il modello è implementato dal progetto Open-Source di riconoscimento vocale **DeepSpeech** che funziona grazie alla tecnologia di Machine Learning, imparando a riconoscere le lingue tramite parole e pronunce.

Per poter quindi permettere a Common Voice di progredire, c'è bisogno di quel "nutrimento" composto dalla partecipazione di volontari che leggono del testo (o che revisionano le registrazioni già eseguite) in una specifica lingua. Con centinaia di letture della stessa frase, il software DeepSpeech si occuperà di individuare le somiglianze il quale isolerà le caratteristiche simili nelle frasi registrate da più persone con accenti diversi, tono, velocità e pronuncia.

Tali somiglianze verranno in seguito aggregate dal software in un modello: uno schema di riconoscimento dei dati (vocali) che gli permetterà di riconoscere frasi mai sentite prima. Ovviamente, il modello si basa su una moltitudine di frasi contenenti parole appartenenti anche ad altri vocabolari (per esempio "feedback", usato anche nella nostra lingua seppur originario di quella inglese), sigle (che noi italiani pronunciamo spesso in diverse maniere), parole con lettere accentate e non solo. Il modello è il prodotto di questo motore (e progetto) che viene reso poi pubblico, corredato di tutte le registrazioni audio fornite sotto libera licenza. Una prima beta del dataset per la lingua italiana verrà rilasciata ad inizio gennaio 2019.

Il progetto Common Voice/DeepSpeech è nato principalmente per contrastare la centralizzazione dei motori Speech-To-Text proprietari (come Siri di Apple e l'Alexa di Amazon), contrastarli perché funzionano solo online con problemi legati alla privacy degli utenti e rendere il riconoscimento vocale disponibile per tutte le lingue in maniera Open-Source.



1.2 Cenni sul Machine Learning

"Automazione per capire un'equazione che risolve un problema specifico,

sulla base di alcuni dati di esempio"

Meglio sintetizzarlo in due parole: **Machine Learning**.

Il Machine Learning è quel processo che riguarda lo studio, la costruzione e l'implementazione di algoritmi che permettono ad un qualsiasi calcolatore sul quale sono implementati di imparare e fare previsioni in modo automatico partendo da un insieme di dati in ingresso (files di testo, musicali, immagini, etc…).

Prendendo in esempio Common Voice e DeepSpeech, possiamo descrivere il suo generico funzionamento in sei punti fondamentali:

1) *Raccolta dei dati*

Come scritto nell'introduzione, per permettere al motore di riconoscimento vocale di allenarsi a "capire" quali esatte parole pronunciamo, serve creare un *Training Data* contenente quanti più campioni (dati vocali e testuali) di esempio. Common Voice, nel nostro caso, è quel *Training Data*.

2) *Preparazione dei dati*

Oppure processo di normalizzazione. Ogni dato raccolto si dovrà rivelare utile e sottostare a delle specifiche regole per allenare il riconoscitore vocale. Preparare i dati significa fornirgli dei dati puliti (*clean data*), privi di qualsiasi possibile errore. La revisione delle frasi testuali e delle registrazioni vocali è essenziale per garantire un dato pulito.

3) *Scelta di un modello*

Il dataset è stato creato e i dati sembrano essere pronti per darli "in pasto" al nostro motore di riconoscimento vocale. Occorre scegliere un modello di rete neurale artificiale (ANN) che permetta di analizzare ed elaborare i dati ricevuti in input e mostrare in output dei risultati -inizialmente- accettabili. Inoltre, nonostante il grande lavoro dietro, la revisione dei dati non è abbastanza per ottenere un *clean dataset*. Confrontare i risultati di varie tipologie di reti neurali può rivelarsi vantaggioso per comprendere quale tipo di ANN utilizzare. Ad esempio, DeepSpeech utilizza un modello TensorFlow e le Reti Neurali Ricorrenti Bidirezionali (BRNN) nella sua prima versione 0.1.0, nella 0.3.0 utilizza le Reti Neurali Ricorrenti (RNN) per un migliore indice di VAD (Voice Activity Detection) e rimozione degli errori.

4) *Fase di Training*

È una fase iterativa. Scelto un modello di reti neurali, è il momento di allenare ciclicamente l'algoritmo. In questa fase vengono impostati gli iperparametri, ovvero dei parametri che regolano il processo di Training. Una buona personalizzazione degli iperparametri permette di ottenere dei risultati più accurati. Impostato un set di valori di iperparametri, dal dataset vengono prelevati i dati per poi essere elaborati dalla rete neurale scelta. Ciò avviene per tutte le volte necessarie affinché si arrivi ad un risultato scelto. È la fase più complessa di tutto il processo poiché richiede una certa potenza di calcolo per analizzare accuratamente i dati, motivo per cui solitamente vengono usate le GPUs. Nell'esempio di Common Voice/DeepSpeech verrà utilizzato un modello già allenato (*pre-trained*) di circa 1.4 GB (impreciso, ma stabile).

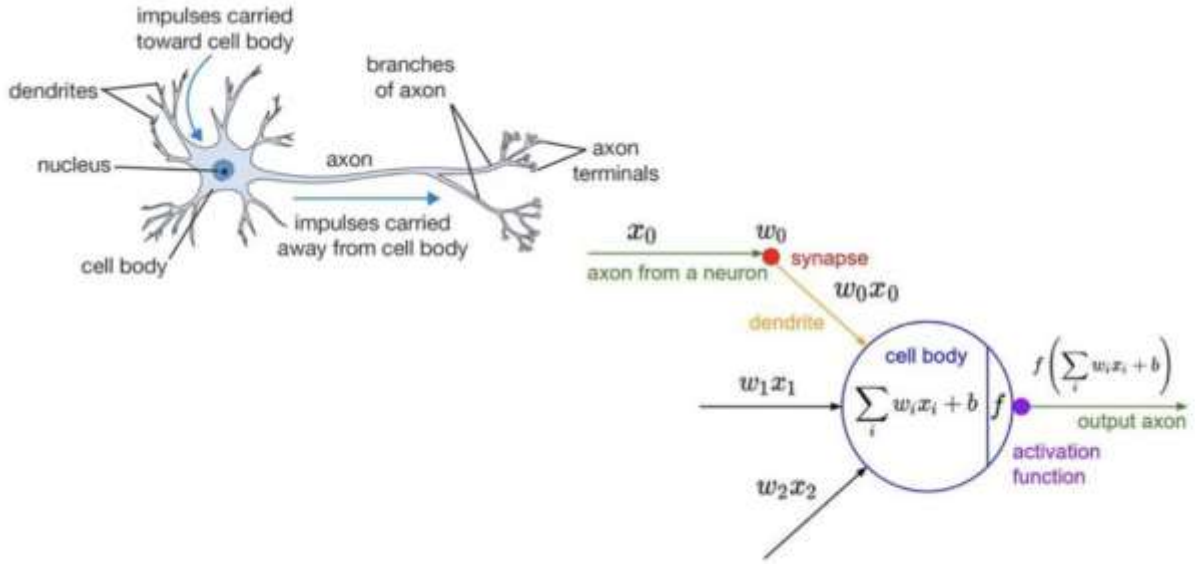
5) *Valutazione dei risultati*

Finita la fase iterativa di Training dell'algoritmo, questi mostra in output dei valori. Se il risultato non corrisponde alle aspettative, bisognerà ri-analizzare e ripetere tutto il processo di elaborazione. Come scritto nel punto 3, nel caso di DeepSpeech, per avere dei miglioramenti è stato cambiato il modello di reti neurali. Ci sarà sempre un margine di errore, uno degli obiettivi principali è quello di ridurlo al minimo possibile.

6) *Predizione*

Risultati finali del processo di Machine Learning. Nel caso di DeepSpeech, l'algoritmo è riuscito a trovare la possibile lettera di una parola che abbiamo pronunciato.

1.3 Approfondimento sulle reti neurali (RNN/BRNN)



Trascrizione matematica di un "semplice" neurone

Nel punto 1.2 sono state accennate le Reti Neurali Artificiali, tuttavia per individuare un modello ottimale di ANN tale che risulti vantaggioso per i nostri obiettivi, con particolare riferimento a DeepSpeech, è necessario chiarire il concetto di ANN.

"In quasi tutti gli organismi viventi sono presenti complesse organizzazioni di cellule nervose, con compiti di riconoscimento delle configurazioni assunte dall'ambiente esterno, memorizzazione e reazione agli stimoli provenienti dallo stesso. Il cervello umano rappresenta probabilmente il più mirabile frutto dell'evoluzione per le sue capacità di elaborare informazioni. Al fine di compiere tali operazioni, le reti biologiche si servono di un numero imponente di semplici elementi computazionali (neuroni) fittamente interconnessi in modo da variare la loro configurazione in risposta agli stimoli esterni: in questo senso si può parlare di apprendimento ed i modelli artificiali cercano di catturare questo tratto distintivo della biologia.

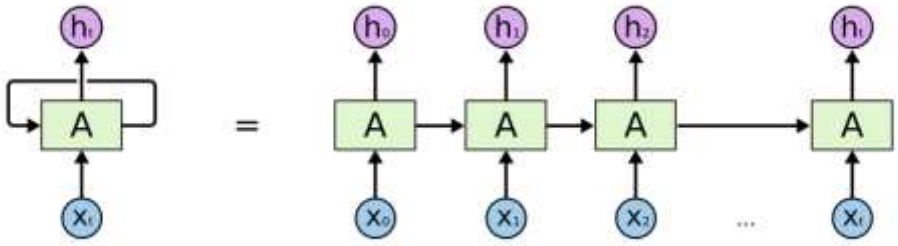
Tipicamente un neurone è costituito di 3 parti principali: il soma (corpo cellulare), l'assone (linea di uscita del neurone unica ma che si può diramare anche in migliaia di rami) e il dendrite (detto anche albero dendritico in quanto la sua estesa ramificazione ne ricorda la forma, il quale funge da linea di entrata del neurone che riceve segnali in ingresso da altri assoni tramite sinapsi). Il corpo cellulare fa una "somma (algebrica) pesata" (integrazione) dei segnali in ingresso. Se il risultato eccede un certo valore di soglia allora il neurone si attiva ed è prodotto un "potenziale di azione" che è mandato all'assone. Se il risultato

non eccede il valore di soglia, il neurone rimane in uno stato di riposo. Una rete neurale artificiale riceve segnali esterni su uno strato di nodi (unità di elaborazione) d'ingresso, ciascuno dei quali è collegato con numerosi nodi interni, organizzati in più livelli. Ogni nodo elabora i segnali ricevuti e trasmette il risultato a nodi successivi." https://it.wikibooks.org/wiki/Intelligenza_artificiale/Reti_neurali

Ponendo a confronto il progetto Common Voice/DeepSpeech con la spiegazione di una rete neurale biologica, teoricamente il funzionamento del motore di riconoscimento vocale avverrebbe attraverso l'ingresso delle informazioni elaborate dal modello di rete neurale scelto (una serie numerica di ingressi moltiplicati per i pesi a cui viene aggiunta una costante di errore, il bias), tale da fornire attraverso una funzione di attivazione (il cui dominio va da 0 a 1, e se 1, l'informazione elaborata diventerà un output, se 0, sarà scartata), un output di informazioni potenzialmente previste.

In tutto questo, però, sussiste un problema: la rete neurale restituisce sempre gli stessi output quando gli si danno gli stessi input. Non c'è progressione poiché è un algoritmo senza memoria (*Stateless Algorithm*). Una rete neurale tradizionale non può mostrare degli output appoggiandosi a degli input utilizzati in altri momenti.

Una soluzione accettabile, utilizzata anche per DeepSpeech, è l'uso delle **Reti Neurali Ricorrenti (RNN)**. Le RNN sono una classe particolare delle ANN le cui connessioni cicliche tra i nodi della rete permettono di dare persistenze alle informazioni ricevute in input. Ciò gli consente di mostrare un comportamento dinamico temporale per una determinata sequenza temporale.



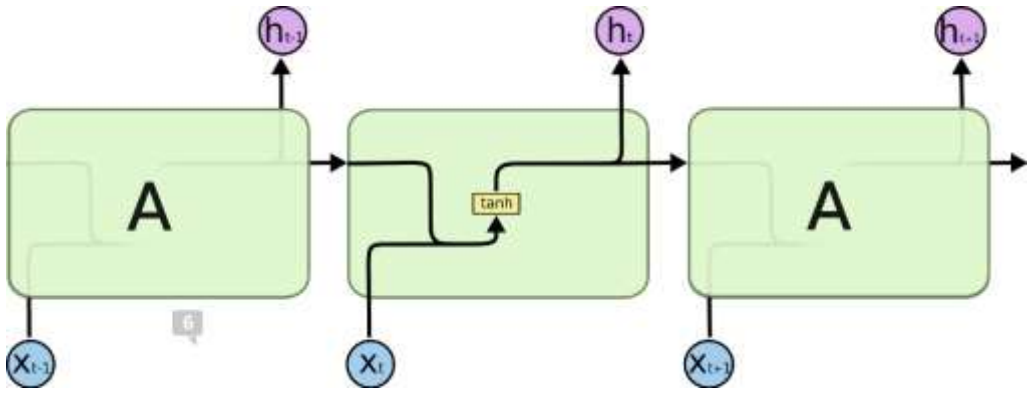
RNN "unrolled"

Una rete neurale ricorrente può essere immaginata come una catena di infinite copie della stessa rete in cui il dato immesso viene continuamente trasferito partendo da una copia iniziale alla successiva, e così via. Nell'ipotesi che si voglia mantenere (e quindi memorizzare) l'informazione in input per ottenere una corretta predizione, subentra un problema contestuale. Prendiamo, ad esempio, una frase simile:

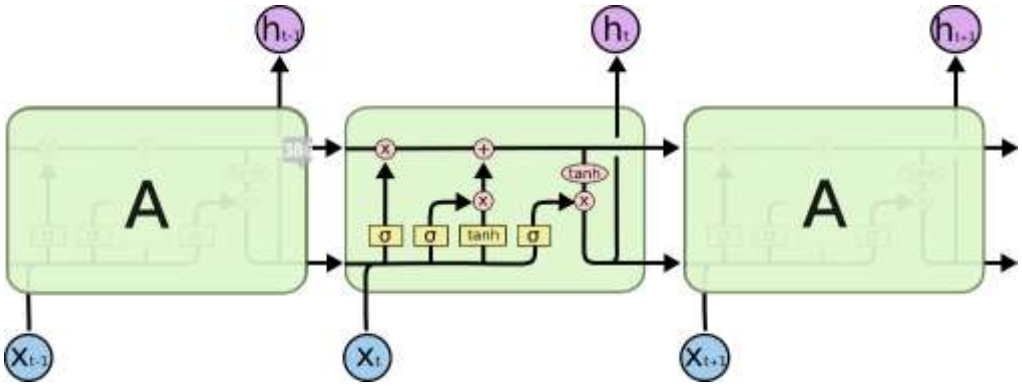
"Sono cresciuto in *Italia*… parlo scorrevolmente l'*italiano*."

L'informazione "Sono cresciuto in *Italia*" suggerisce al lettore un indizio per indovinare la lingua parlata. Esiste una connessione data dalla facilità del contesto. Nel caso di DeepSpeech, come si procede con frasi molto più complesse?

DeepSpeech è un progetto sperimentale, e attualmente seguono benchmarks per individuare la migliore combinazione di RNN e personalizzazione degli iperparametri. Nella versione 0.3.0 vengono utilizzate le RNN LSTM (Long Short Term Memory) poiché in grado di tenere in memoria le informazioni per lunghi periodi di tempo. Fondamentalmente si tratta di aggiornare le informazioni. Solitamente in ogni copia di una RNN è contenuto un solo livello tramite cui viene elaborata l'informazione in ingresso. Le RNN LSTM hanno pure questa struttura a catena, tuttavia contengono quattro livelli che interagiscono tra di loro.



Rappresentazione grafica di una RNN



Rappresentazione grafica di una RNN LSTM

Nel diagramma di sopra, ogni linea riporta un intero vettore, dall'output di un nodo agli input di altri. I cerchi colorati in rosa rappresentano operazioni puntuali (operazioni tra le componenti di vettori in cui l'operatore che prende in input un valore ne restituisce in output uno diverso che dipende esclusivamente dal valore in ingresso) come la somma/moltiplicazione tra vettori, mentre le caselle gialle sono i quattro livelli di rete neurale (con funzione di attivazione tangente iperbolica e sigmoidea). Le linee che si uniscono denotano la concatenazione, mentre una linea che si biforca denota che il suo contenuto viene copiato e procederà verso altri componenti.

Come scritto in precedenza, la differenza fondamentale consiste nell'utilizzo di più livelli di rete neurale per il controllo delle informazioni. La gestione dei livelli di rete neurale è affidata alla **cella di stato** A. La LSTM ha la capacità di rimuovere/aggiungere informazioni alla cella di stato tramite tre "gate". I gate sono un modo per consentire all'informazione di procedere verso altre strutture. Sono composti da un livello di rete neurale con funzione di attivazione sigmoidea e da un'operazione di moltiplicazione puntuale. La funzione di attivazione sigmoidea ha un range di numeri compresi tra 0 e 1, e serve per proteggere e controllare lo stato della struttura: se la quantità dell'informazione ha un valore pari a 1, l'informazione può continuare verso gli altri componenti della struttura, se è 0, viene scartata.

Il processo di aggiornamento della LSTM si articola in questi punti:

1) Quali informazioni eliminare dalla cella di stato:

Questo processo è affidato al (primo) livello di rete neurale con funzione sigmoidea chiamato "*forget gate layer*" o $\mathbf{F_t}$. A partire da $\mathbf{H_{t-1}}$ e $\mathbf{X_t}$, in base all'informazione ricevuta dalla copia precedente, viene restituito un numero compreso tra 0 e 1 per ogni numero presente nella cella di stato $\mathbf{C_{t-1}}$. Se ricevuto 1 l'informazione viene mantenuta, altrimenti viene eliminata.

2) Memorizzazione nella cella di stato delle nuove informazioni ricevute:

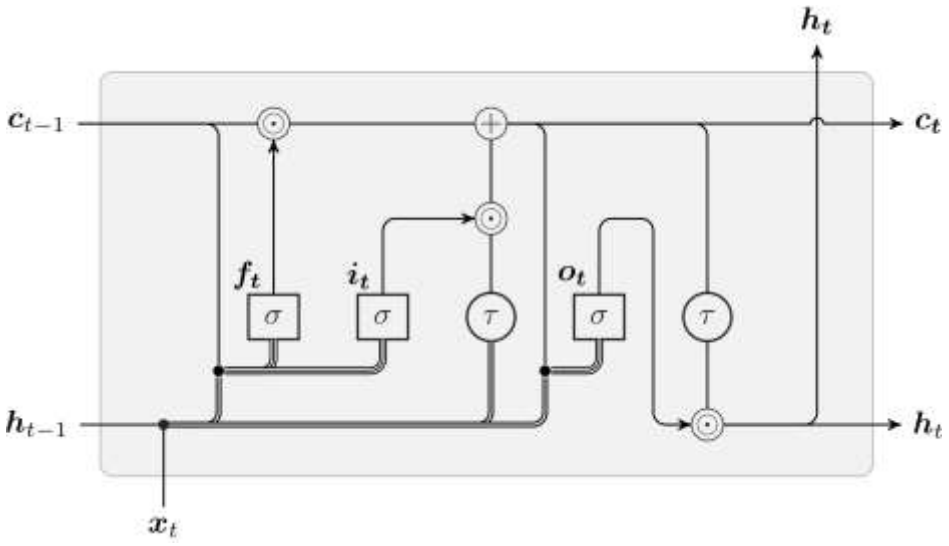
Questa operazione è costituita da due parti. La prima, in cui il (secondo) livello di rete neurale sigmoideo chiamato "*input gate layer*" (oppure $\mathbf{i_t}$) decide quali valori verranno aggiornati. La seconda e successiva, in cui un livello di rete neurale con funzione tangente iperbolica crea un vettore di nuovi valori candidati, che chiameremo $\mathbf{\hat{C}_t}$. La combinazione tra questi due elementi permetterà di aggiornare l'intera cella di stato.

3) Operazione di aggiornamento:

L'operazione di aggiornamento consiste nell'aggiornare il precedente gate con cui sono stati svolti i precedenti punti (1 e 2). Moltiplichiamo il "*forget gate layer*" o $\mathbf{F_t}$ per la "vecchia" cella di stato $\mathbf{C_{t-1}}$ e aggiungiamo ad esso l'"*input gate layer*" (oppure $\mathbf{i_t}$) e il nuovo vettore di candidati $\mathbf{\hat{C}_t}$**.** Il risultato sarà $\mathbf{C_t}$ ovvero il nuovo valore candidato. Nel caso di DeepSpeech, il nuovo valore è la nuova possibile trascrizione della lettera della parola pronunciata.

4) Inizializzazione dell'output per la prossima cella di stato:

L'ultimo punto consiste nel decidere quale output trasferire alla prossima cella di stato A_n. Sarà una versione "filtrata" dell'output dell'attuale cella di stato attraverso l'utilizzo del (terzo) livello di rete neurale sigmoideo che decide quale parte della cella di stato stiamo per mandare in output. L'output del livello sigmoideo viene moltiplicato per la funzione tangente iperbolica in modo tale da avere un potenziale di output $\mathbf{H_t}$ non nullo da inviare alla prossima cella di stato A_n.



Schema completo di un processo di aggiornamento di una cella di stato LSTM

1.4) Dimostrazione sul funzionamento di Deepspeech

Scansionando il QR Code presente nell'immagine è possibile scaricare una semplice interfaccia web e un web server scritto in Node.js che consente di provare DeepSpeech 0.1.0 (con un modello pre-trained in *inglese* di circa 1.4 GB) in locale sul proprio computer. In questa demo sarà possibile registrare la propria voce -obbligatoriamente in inglese- dal proprio browser web, attendere un paio di secondi, e dopodiché leggere la trascrizione della propria registrazione. È consigliabile utilizzare una qualsiasi distribuzione GNU/Linux con Git installato per provare la demo. È necessario installare il programma CLI SoX per poter installare questa demo.

SoX è una utility cross-platform in grado di convertire vari formati audio in altri formati. Nel caso di questa demo, l'applicazione convertirà in automatico la registrazione in un file WAV (con estensione .wav) in quanto formato privo di compressione dati (*lossless*) e supportato pienamente da DeepSpeech.

Installazione su GNU/Linux:

1) Installazione di Git e SoX:

apt install git && apt install sox

3) Creare una cartella "Demo_DeepSpeech" e scaricare la repository della demo:

git clone https://github.com/asciidisco/deepspeech-demo

4) Accedere alla cartella appena scaricata:

cd /home/nome_user/Demo_DeepSpeech

5) Usare il package manager di Node.js per scaricare DeepSpeech ed il modello pre-trained (1.4 GB):

npm install

6) Una volta scaricato DeepSpeech e il modello pre-trained, avviare il server per provare la demo:

npm start

7) Avvio riuscito! Ora è possibile provare la demo su http://localhost:3000 cliccando “Listen” per registrare la propria voce, una volta eseguita la registrazione, occorre cliccare su “Listening” per terminarla e avviare il processo di trascrizione.

1.5) Principali utilizzi di Common Voice e conclusioni

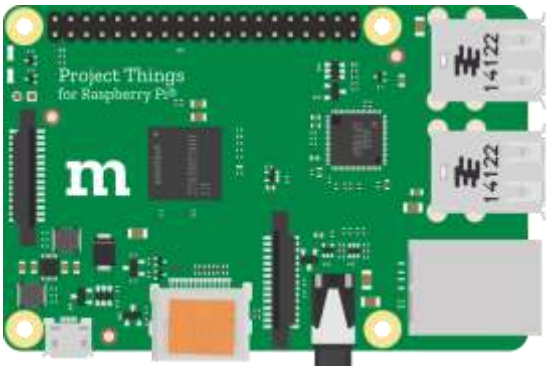
Data la sua natura Open-Source, attualmente Common Voice è utilizzato per la creazione di un Assistente vocale completamente Open-Source, ovvero Mycroft AI (https://mycroft.ai/get-started/). È già possibile utilizzarlo installandolo sul proprio computer con sistema operativo GNU/Linux (anche su Raspberry Pi 3) oppure utilizzando il dispositivo ufficiale Mark 1 (https://mycroft.ai/get-mycroft/).

Dal 2019 sarà presente anche il supporto alla localizzazione in italiano. (Nell’immagine a destra, Mycroft Mark 1)

Altri progetti che mirano ad utilizzare Common Voice sono Xaero (https://xaero.app), per la realizzazione di un correttore grammaticale utile per effettuare controlli di testo accurati.

Mozilla Project Things (https://iot.mozilla.org/) per sfruttare l’attuale ampio ecosistema IoT tramite piattaforma Raspberry Pi 3, proponendosi come complementare e/o alternativa anche ai sistemi di domotica con openHAB 2.

Per ultimo, sono presenti anche progetti sperimentali Text-To-Speech ispirati al Tacotron di Google (https://github.com/mozilla/TTS).



2. Riferimenti Bibliografici

https://arxiv.org/abs/1412.5567
Documento della Baidu Research “Deep Speech: Scaling up end-to-end speech recognition” dove vengono evidenziati i quattro punti fondamentali per un algoritmo speech-to-text:

- Deep Learning attraverso le RNN (Reti Neurali Ricorrenti)

- Parallel processing

- Un grande dataset per il training

- Complessità delle pipelines nulla

http://colah.github.io/posts/2015-08-Understanding-LSTMs/
Articolo utilizzato per approfondire le RNN e le RNN LSTM (contiene ulteriori articoli con cui approfondire l'argomento sulle reti neurali e il machine learning).

https://github.com/mozilla/DeepSpeech
Link alla repository di DeepSpeech su GitHub, permette di introdurre al lato più tecnico del progetto DeepSpeech. Le competenze minime per provare il progetto sono la manualità col package manager di Python/Nodejs e col terminale di una qualsiasi distribuzione GNU/Linux (anche su WSL di Windows 10) o MacOS.

https://www.tensorflow.org/
Framework Open-Source essenziale per il funzionamento di DeepSpeech e, in generale, molto utile per qualsiasi idea di machine learning. Il link indirizza alla sito di TensorFlow, il sito contiene vari riferimenti per prendere praticità con semplici algoritmi di machine learning (soprattutto coi Google Codelabs).

3. Argomenti Teorici Trattati

In questa documentazione sono stati trattati argomenti riguardanti l'Open-Source, gli Open Data, il Machine Learning, il Deep Learning, le Reti Neurali (Ricorrenti), il progetto di Mozilla Common Voice e DeepSpeech, Assistenti vocali, Internet of Things e Algoritmi di ricampionamento.

Tutta la documentazione è basata sui concetti di Open-Source, in particolar modo attenzionando argomenti attuali come il Machine Learning e i suoi sottoinsiemi. Vengono incontro a questi concetti puramente teorici, alcuni progetti della fondazione Mozilla, mirati ad arricchire il panorama Open-Source.