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: 16

TITOLO PROGETTO: Come funziona *Shazam*?

AUTORE 1: Liotta Alessandro

AUTORE 2: Martini Miriana

AUTORE 3: Macaluso Roberta

Indice

1. Obiettivi del progetto

Spiegare come avviene il riconoscimento di una canzone: *fingerprint* di una canzone

Lo scopo di questo progetto è spiegare come funzionano le applicazioni di riconoscimento musicale.

Tali app, a seguito di un'acquisizione audio, creano un'impronta digitale (*fingerprint*) per il campione registrato, consultano il database e usano un determinato algoritmo di riconoscimento musicale per dire esattamente quale canzone si sta ascoltando.

Il nostro primo obiettivo è quindi quello di definire il *fingerprint* di una canzone ed evidenziare la sua importanza nel riconoscimento di una canzone.

Definire l'algoritmo di identificazione della musica

Come detto precedentemente, le applicazioni che andremo a considerare fanno uso di particolari algoritmi di identificazione musicale. Il nostro secondo scopo sarà, quindi, quello di definire tali algoritmi e vederne il funzionamento.

Caso particolare: *Shazam*

Come caso particolare dell'intero progetto prenderemo in considerazione l'applicazione di riconoscimento musicale più famosa al mondo: *Shazam*.

Ne vedremo il suo funzionamento facendo riferimento soprattutto all'algoritmo di *Shazam* rivelato al mondo dal suo inventore Avery Li-Chung Wang nel 2003.

2. Riferimenti Bibliografici

https://www.toptal.com/algorithms/shazam-it-music-processing-fingerprinting-and-recognition

Articolo di Jovan Jovanovic su cui ci siamo basati per la spiegazione di questo progetto. Presenta una buona spiegazione del funzionamento di *Shazam* e un'introduzione necessaria per poterlo comprendere.

http://www.ee.columbia.edu/~dpwe/papers/Wang03-shazam.pdf

La spiegazione dell'algoritmo di *Shazam* di Avery Li-Chung Wang a cui abbiamo fatto precedentemente riferimento. Utilizzeremo questo articolo come approfondimento sul funzionamento della corrispondente applicazione.

https://it.wikipedia.org/wiki/Shazam_(software)

Pagina di *Wikipedia* di *Shazam*. Niente da aggiungere. Grazie di esistere.

3. Argomenti Teorici Trattati

Passaggio da analogico a digitale: campionamento di un segnale

A differenza di altri servizi che permettono di identificare una canzone canticchiata, *Shazam* funziona analizzando il suono catturato e cercando il brano uguale sulla base di una impronta acustica in un database di oltre 11 milioni di canzoni. Ma come fa? Andiamo per gradi.

Sappiamo che il suono è una variazione di pressione che si propaga come un'onda meccanica, attraverso un mezzo come l'aria o l'acqua. Quando quella vibrazione arriva alle nostre orecchie, in particolare al timpano, muove piccole ossa (staffa, incudine e martello) che trasmettono la vibrazione a piccole cellule ciliate in profondità nel nostro orecchio interno (coclea). Infine, le piccole cellule cigliate producono impulsi elettrici, che vengono trasmessi al nostro cervello attraverso il nervo uditivo.

I dispositivi di registrazione simulano questo processo, convertendo la pressione dell'onda sonora in un segnale elettrico. Un'onda sonora reale nell'aria è un segnale di pressione continua. In un microfono, il primo componente elettrico che incontra questo segnale (un trasduttore) lo traduce in un segnale di tensione analogica, che è ancora una variabile continua.

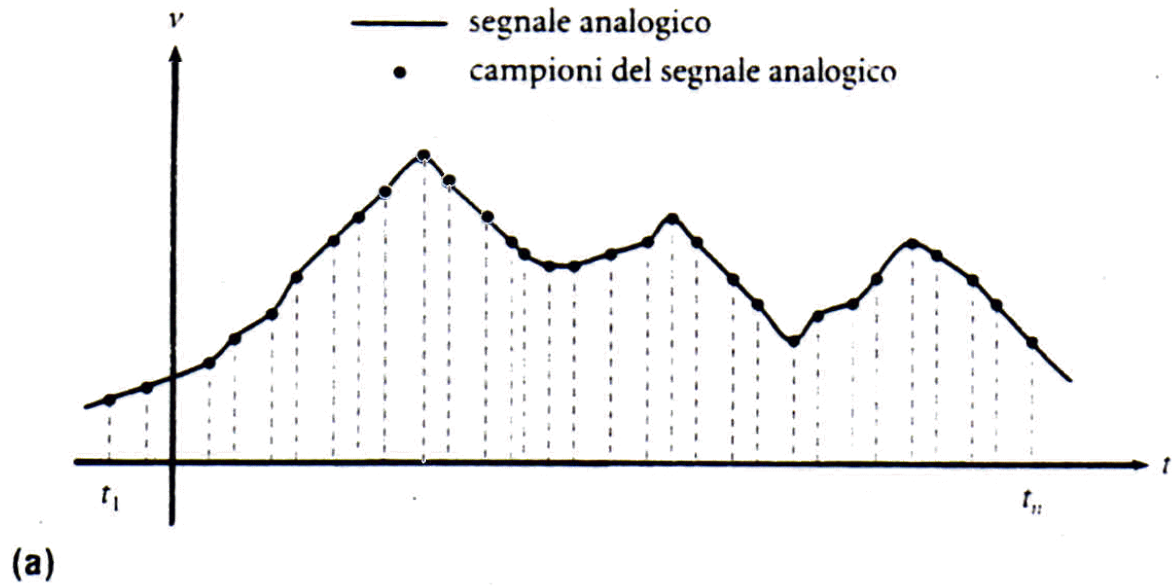
Questo segnale continuo **[figura (a)]** non è così utile nel mondo digitale, quindi prima che possa essere elaborato, deve essere tradotto in un segnale discreto **[figura (b)]** che può essere memorizzato digitalmente. Questo viene fatto catturando un valore digitale che rappresenta l'ampiezza del segnale.

La conversione implica la quantizzazione dell'input e introduce necessariamente una piccola quantità di errore. Pertanto, invece di una singola conversione, un convertitore analogico-digitale esegue molte conversioni su pezzi molto piccoli del segnale - un processo noto come campionamento.

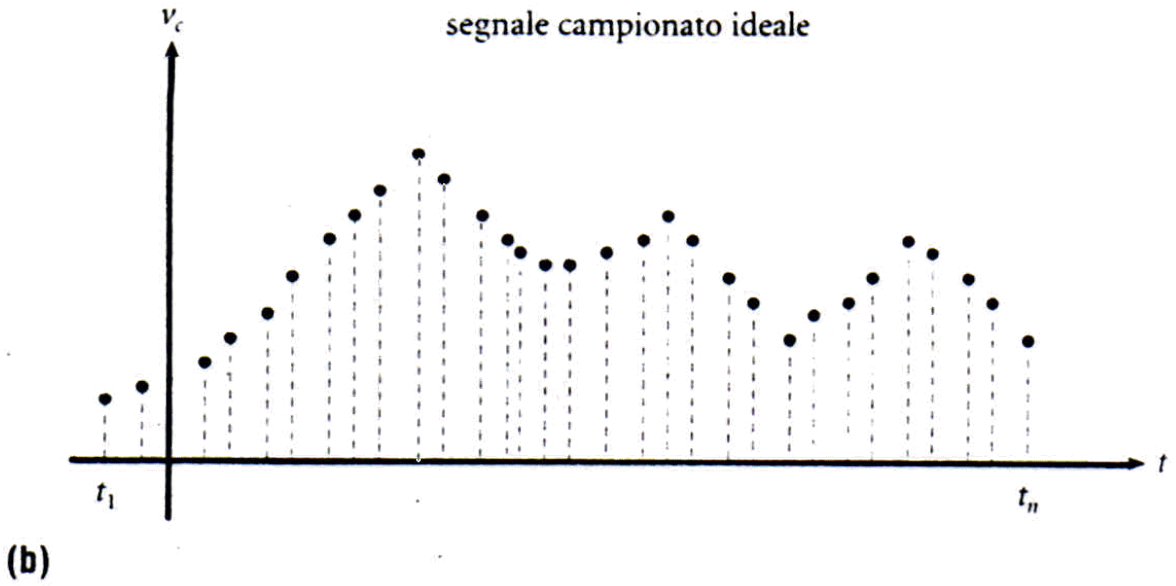
Il *Teorema di Nyquist-Shannon* ci dice quale tasso di campionamento è necessario per catturare una certa frequenza nel segnale continuo. In particolare, per catturare tutte le frequenze che un essere umano può sentire in un segnale audio, dobbiamo campionare il segnale ad una frequenza doppia rispetto a quella del range uditivo umano. L'orecchio umano può rilevare frequenze all'incirca tra 20 Hz e 20.000 Hz. Di conseguenza, l'audio viene registrato più spesso a una frequenza di campionamento di 44.100 Hz.

Questo tasso specifico è stato originariamente scelto da Sony perché poteva essere registrato su apparecchiature video modificate a 25 fotogrammi al secondo (PAL) o 30 fotogrammi al secondo (utilizzando un videoregistratore monocromatico NTSC) e coprire l'ampiezza di banda di 20.000 Hz necessaria per abbinare l'attrezzatura di registrazione analogica professionale del tempo.

Quindi, quando si sceglie la frequenza del campione che è necessario registrare si dovrà probabilmente andare fino a 44.100 Hz.



(a)



(b)

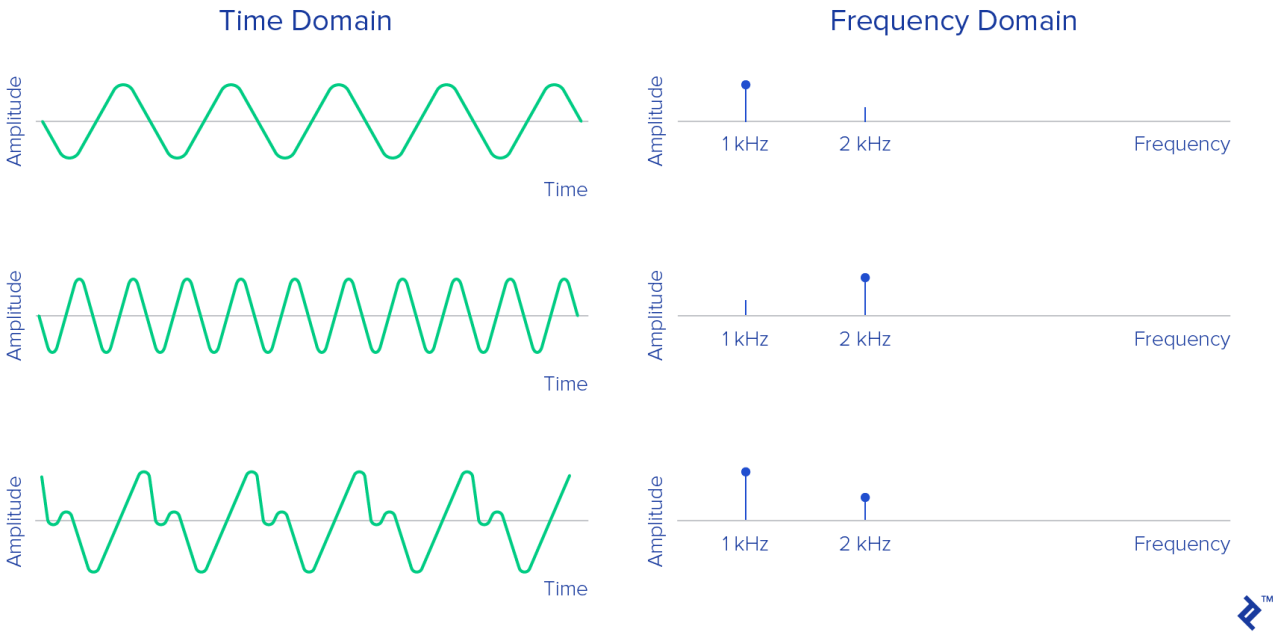
Dominio del tempo e dominio della frequenza

All'inizio del 1800, Jean-Baptiste Joseph Fourier fece la straordinaria scoperta che ogni segnale nel dominio del tempo è equivalente alla somma di un numero (forse infinito) di semplici segnali sinusoidali caratterizzati da una certa frequenza, ampiezza, e fase. La serie di sinusoidi che insieme formano il segnale originale nel dominio del tempo è conosciuta come serie di Fourier.

In altre parole, è possibile rappresentare qualsiasi segnale del dominio del tempo considerando semplicemente l'insieme di frequenze, ampiezze e fasi corrispondenti a ciascuna sinusoide che costituisce il segnale. Questa rappresentazione del segnale è nota come dominio della frequenza.

In qualche modo, il dominio della frequenza agisce come un tipo di *impronta digitale* per il segnale del dominio del tempo, fornendo una rappresentazione statica di un segnale dinamico.

L'analisi di un segnale nel dominio della frequenza è più conveniente nel mondo dell'elaborazione del segnale digitale perché permette all'ingegnere di studiare lo spettro del segnale (la rappresentazione del segnale nel dominio della frequenza) e determinare quali frequenze sono presenti e quali sono mancanti. Successivamente, è possibile filtrare, aumentare o diminuire alcune frequenze o semplicemente riconoscere il tono esatto dalle frequenze fornite.

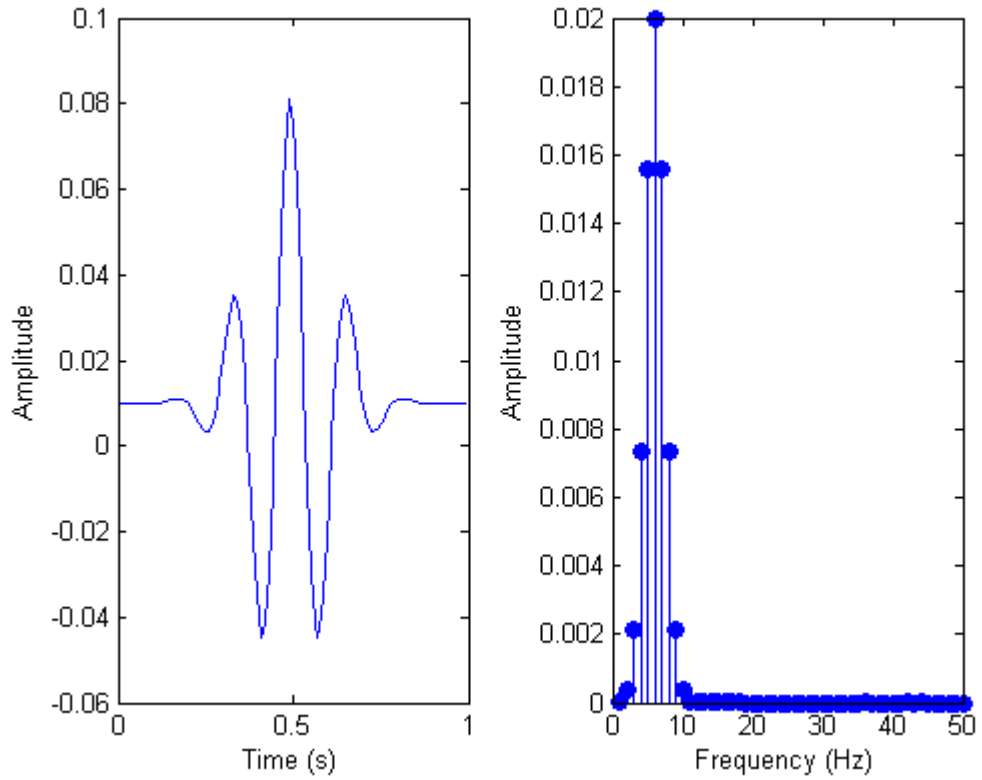


Discrete Fourier Transform (DFT)

Un modo per convertire il nostro segnale dal dominio del tempo al dominio della frequenza è utilizzare la *Discrete Fourier Transform* (DFT). La DFT è una metodologia matematica per l'esecuzione dell'analisi di Fourier su un segnale discreto (campionato). Si tratta di una trasformata che converte una collezione finita di campioni equispaziati di una funzione in una collezione di coefficienti di una combinazione lineare di sinusoidi complesse, ordinate al crescere della frequenza.

Uno degli algoritmi numerici più popolari per il calcolo di DFT è la trasformata Fast Fourier (FFT). Di gran lunga la variazione più comunemente utilizzata di FFT è l'algoritmo di Cooley-Tukey. Questo è un algoritmo che divide ricorsivamente una DFT in molti DFT più piccoli. Mentre la valutazione di un DFT richiede direttamente operazioni $O(n^2)$, con un FFT Cooley-Tukey lo stesso risultato viene calcolato in operazioni $O(n \log(n))$.

Qui di seguito l’esempio di un segnale prima e dopo un’analisi FFT.



Cattura di un suono

Uno sfortunato effetto collaterale della FFT è che perdiamo una grande quantità di informazioni sui tempi. (Anche se teoricamente questo può essere evitato, i costi generali delle prestazioni sono enormi). Per esempio, per una canzone di tre minuti, vediamo tutte le frequenze e le loro ampiezze, ma non abbiamo la minima idea di quando sono comparse nella canzone. Ma questa è l'informazione chiave che rende la canzone quello che è, quindi in qualche modo abbiamo bisogno di sapere quando è apparsa ogni frequenza.

Ecco perché introduciamo una sorta di “finestra scorrevole”, o un chunk di dati, e trasformiamo solo questa parte delle informazioni. La dimensione di ogni blocco può essere determinata in diversi modi. Ad esempio, se registriamo il suono, in stereo, con campioni a 16 bit, a 44.100 Hz, un secondo di tale suono sarà *44.100 campioni · 2 byte · 2 canali ≈ 176 kB*. Se scegliamo 4 kB per la dimensione di un chunk, avremo 44 blocchi di dati da analizzare in ogni secondo della canzone. Questa è una buona densità per un’analisi dettagliata necessaria per l'identificazione audio.

Una volta che abbiamo informazioni sulla composizione delle frequenze del segnale, possiamo iniziare a determinare il *fingerprint* della canzone. Questa è la parte più importante dell'intero processo di riconoscimento audio di *Shazam*. La sfida principale qui è come distinguere, nell'oceano di frequenze catturate, quali frequenze sono le più importanti. Intuitivamente, cerchiamo le frequenze con la massima ampiezza (comunemente chiamate picchi).

Tuttavia, in una canzone la gamma delle frequenze forti potrebbe variare tra i bassi *C - C1 (32,70 Hz)* e gli alti *C - C8 (4,186,01 Hz)*. Essendo questo un intervallo enorme da coprire, invece di analizzare l'intera gamma di frequenze contemporaneamente, scegliamo diversi intervalli più piccoli, scelti in base alle comuni frequenze dei più importanti componenti musicali, e analizzarli separatamente.

Una volta fatto ciò, all'interno di ogni intervallo possiamo semplicemente identificare la frequenza con la massima ampiezza. Questa informazione costituisce una firma per questo intervallo della canzone, e questa firma diventa parte del *fingerprint* nel suo complesso. Per facilitare la ricerca audio, questa firma diventa la chiave in una tabella hash. Il valore corrispondente è il tempo in cui questo insieme di frequenze è apparso nella canzone, insieme all'ID della canzone (titolo del brano e artista). Ecco un esempio di come potrebbero apparire nel database.

Hash Tag	Time in Seconds	Song
`30 51 99 121 195`	53.52	Song A by artist A
`33 56 92 151 185`	12.32	Song B by artist B
`39 26 89 141 251`	15.34	Song C by artist C
`32 67 100 128 270`	78.43	Song D by artist D
`30 51 99 121 195`	10.89	Song E by artist E
`34 57 95 111 200`	54.52	Song A by artist A
`34 41 93 161 202`	11.89	Song E by artist E

Se sviluppiamo un'intera libreria di canzoni attraverso questo processo di identificazione musicale, possiamo creare un database con l'impronta completa di ogni canzone nella libreria.

Per esempio, consideriamo di volere scoprire il titolo di una canzone che è in riproduzione in un locale, prendiamo il nostro telefono ed eseguiamo la registrazione attraverso lo stesso processo di *fingerprinting* audio di cui sopra. Quindi possiamo iniziare a cercare nel database i tag hash corrispondenti.

Come accade, molti dei tag hash corrisponderanno all'identificatore musicale di più brani. Ad esempio, potrebbe capitare che qualche parte della canzone A suoni esattamente come un pezzo della canzone E. Ogni volta che associamo un tag hash, il numero di corrispondenze possibili diventa più piccolo, ma è probabile che queste informazioni da sole non riducano la corrispondenza a una singola canzone. Quindi c'è un'altra cosa che dobbiamo controllare con il nostro algoritmo di riconoscimento musicale, e questa è la tempistica.

Il campione che abbiamo registrato nel locale potrebbe provenire da qualsiasi punto della canzone, quindi non possiamo semplicemente abbinare il *timestamp* dell'hash corrispondente con

il *timestamp* del nostro campione. Tuttavia, con più hash abbinati, possiamo analizzare i tempi relativi degli abbinamenti e quindi aumentare la nostra certezza.

Ad esempio, se si considera la tabella precedente, possiamo vedere che il tag hash 30 51 99 121 195 corrisponde sia alla canzone A che alla canzone E. Ma se, un secondo dopo, la nostra registrazione combacia con l'hash 34 57 95 111 200, che è un'altra combinazione per la canzone A, capiamo che entrambi gli hash e le differenze di tempo combaciano.

Consideriamo i_1 e i_2, momenti nel brano registrato, e j_1 e j_2 momenti nel brano dal database. Possiamo dire che abbiamo due combinazioni tra le differenze di tempo se:

> $$RecordedHash(i_1) = SongInDBHash(j_1)\ AND\ RecordedHash(i_2) = SongInDBHash(j_2)$$
>
> $$AND$$
>
> $$abs(i_1 - i_2) = abs\ (j_1 - j_2)$$

Questo ci dà la possibilità di registrare la canzone dall'inizio, dalla metà o dalla fine.

Infine, è improbabile che ogni singolo momento della canzone che registriamo nel locale corrisponda ad ogni momento della stessa canzone nella nostra libreria, registrato in studio. La registrazione includerà molto rumore che introdurrà qualche errore nelle combinazioni.

Quindi, invece di cercare di eliminare tutto tranne la canzone corretta dalla nostra lista di combinazioni, alla fine, ordiniamo tutte le canzoni abbinate in ordine decrescente di probabilità, cosicché la nostra canzone sarà la prima nella classifica.

Adesso, finalmente, leggendo la frase su *Wikipedia* che descrive il funzionamento di *Shazam*, ne comprendiamo il significato:

"*Shazam* utilizza il microfono incorporato nel telefono cellulare per catturare un breve campione di una canzone in riproduzione. Viene quindi creata un'impronta digitale acustica sulla base del campione e confrontata con un database centrale alla ricerca della somiglianza. Se viene trovata la somiglianza, sul display dell'utente vengono mostrate le informazioni sull'artista e ovviamente il titolo e il testo della canzone sullo sfondo raffigurante il frontespizio dell'album di riferimento."