UNIVERSITÀ DEGLI STUDI DI PADOVA

FACOLTÀ DI INGEGNERIA
Corso di Laurea in Ingegneria Informatica

TESI DI LAUREA

VERIFICA SPERIMENTALE DI UNA METODOLOGIA PER LA SEGMENTAZIONE DELLA MUSICA

Candidato:
Federico Demuro
Matricola 343598

Relatore:
Prof. Nicola Orio

ANNO ACCADEMICO 2008-2009

FEDERICO DEMURO

VERIFICA SPERIMENTALE DI UNA METODOLOGIA PER LA SEGMENTAZIONE DELLA MUSICA

TESI DI LAUREA

Relatore: Prof. Nicola Orio

Università di Padova
Facoltà di Ingegneria
Dipartimento di Elettronica ed Informatica
20 Aprile 2010

Ai miei cari

SOMMARIO

Nell'ambito della *Music Information Retrieval* la segmentazione audio è un passaggio obbligato nella soluzione di importanti problemi. In questa tesi di laurea si presenta un algoritmo mutuato dalla segmentazione testuale ed adattato alla musica, che prevede la costruzione di una matrice di similarità basata sull'energia dei singoli *frames* audio, la sua trasformazione con un procedimento di *ranking* ed infine l'individuazione dei segmenti mediante una tecnica di *clustering*. Vengono inoltre descritti e testati alcuni indici di similarità provenienti da altri ambiti di ricerca. Infine si fornisce una prima valutazione dell'algoritmo in termini di efficacia, basata su un *set-up* sperimentale composto da brani di musica classica.

INDICE

ELENCO DELLE FIGURE

ELENCO DELLE TABELLE

ACRONIMI

MIR *Music Information Retrieval*

MIDI *Musical Instrument Digital Interface*

MIREX *Music Information Retrieval Evaluation eXchange*

HMM *Hidden Markov Models*

STFT *Short Time Fourier Transform*

ISMIR *International Conference on Music Information Retrieval*

TFR *Time Frequency Representations*

DFT *Discrete Fourier Transform*

1 INTRODUZIONE

La musica è il piacere che la mente umana prova,
quando conta senza essere conscia di contare.

— Gottfried Whilhem Leibniz (1646-1716)
in Marcus du Sautoy, *L'enigma dei numeri primi*

Negli anni recenti abbiamo assistito a enormi progressi nella digitalizzazione di documenti di ogni tipo, così come ad una altrettanto vertiginosa crescita nell'utilizzo dell'*information technology* da parte degli utenti che vogliono accedere e fruire di contenuti multimediali. In questo ambito sono stati fatti grandi sforzi verso lo sviluppo di tecniche per la ricerca e l'estrazione di informazioni utili da grandi moli di dati. In particolare per l'informazione testuale sono stati implementati potenti motori di ricerca in grado di garantire la navigazione e il recupero di informazioni tra bilioni di documenti di testo. Per altri tipi di dati multimediali, come la musica, le immagini o il video, le strategie tradizionali si basano su annotazioni testuali o meta-dati collegati ai documenti stessi. Poiché la generazione manuale di tali etichette descrittive è impraticabile per grandi quantità di dati, sono necessarie a tale scopo procedure completamente automatizzate oltre a efficienti metodi di reperimento delle informazioni basati sul contenuto (*content-based*), che accedono direttamente ai dati grezzi senza basarsi sulla disponibilità di meta-dati.

Information retrieval: approccio data-based vs. content-based.

Uno scenario tipico, che polarizza ormai da tempo l'interesse di molti studiosi nel campo dell'*information retrieval*, si basa sul paradigma del *query-by-example*: data una *query* sotto forma di un frammento di dati, l'obbiettivo è di recuperare automaticamente tutti i documenti nella base di dati che contengono parti o aspetti simili alla *query*. Qui, la nozione di similarità, che dipende fortemente dall'ambito di utilizzo e dalla percezione umana, è di cruciale importanza nella comparazione dei dati. Spesso, gli oggetti

Il paradigma query-by-example.

multimediali, anche se simili da un punto di vista strutturale o semantico, possono rivelare significative differenze spaziali o temporali. Questo rende l'*information retrieval* di tipo *content-based* su dati multimediali un campo di ricerca pieno di sfide con ancora molti problemi irrisolti.

1.1 MUSIC INFORMATION RETRIEVAL

L'information retrieval applicato alla musica.

Nell'ambito della musica, il campo di ricerca multidisciplinare nel quale si affronta, tra gli altri, anche questo problema si chiama *Music Information Retrieval* (MIR): il processamento e il *retrieval* di informazioni di natura musicale, il riconoscimento automatico e la classificazione della musica, la definizione e l'estrazione di caratteristiche dell'audio musicalmente rilevanti o lo sviluppo di nuove interfacce per gli utenti sono tipiche sfide affrontate da questa scienza. Grazie alla varietà e alla ricchezza della musica, le ricerche del MIR uniscono studiosi provenienti da molti diversi campi, come l'informatica, l'ingegneria dell'audio, la musicologia, la teoria della musica, con importanti ricadute sulla legislatura (basti pensare al problema della tutela dei diritti intellettuali) e sull'economia.

Alcune letture introduttive.

Per uno sguardo d'insieme sul tipo di ricerche affrontate da questa disciplina ci si può riferire all'articolo di Downie [5]. Recentemente, Pardo [14] ha prodotto una serie di brevi articoli sempre con lo stesso intento. L'articolo di Orio [13] riassume questioni fondamentali sulla rappresentazione della musica, l'interazione con l'utente, l'elaborazione della stessa e la specifica di sistemi MIR. Un dettagliato resoconto su vari problemi di analisi musicale da un punto di vista interdisciplinare, che include aspetti di psicoacustica e di percezione della musica, può essere trovato nella tesi di dottorato di Scheirer [16]. Il libro di Klapuri and Davy [9] affronta un tema centrale per il MIR, la trascrizione automatica della musica, che comprende altri importanti problemi come l'analisi del ritmo, l'analisi della frequenza fondamentale, la separazione delle sorgenti e la classificazione degli strumenti musicali. L'esauriente libro di Mazzola [10] tratta della matematica della teoria musicale e introduce le basi concettuali per la composizione, l'analisi e l'esecuzione della musica.

Inoltre ci si può riferire all'*International Conference on Music Information Retrieval* (ISMIR) che costituisce una piattaforma multidisciplinare per i ricercatori coinvolti; gli atti

di questa conferenza, che sono accessibili a partire da questo indirizzo http://www.ismir.net, contengono un ampio spettro di articoli che riflettono lo stato dell'arte del settore. Infine, da qualche anno un appuntamento stimolante per gli studiosi è costituito dal *Music Information Retrieval Evaluation eXchange* (MIREX), una competizione che serve soprattutto a condividere idee e risultati e che si svolge in categorie che coprono tutti gli ambiti menzionati; chi è interessato può visitare il sito dell'ultima edizione che si trova all'indirizzo: http://www.music-ir.org/mirex/2009/index.php/Main_Page.

1.1.1 *Applicazioni*

Tipicamente, le raccolte di musica digitale contengono un grande numero di documenti relativi alla stessa opera musicale, che sono dati in vari formati digitali e in molteplici realizzazioni. Per esempio, nel caso della Quinta Sinfonia di Beethoven, una raccolta digitale musicale potrebbe contenere le pagine ottenute con uno scanner di alcune particolari edizioni dello spartito. Oppure lo spartito potrebbe essere descritto da un particolare formato digitale che rappresenti la notazione musicale, codificandola in modo da renderla processabile da un computer. Inoltre la raccolta potrebbe comprendere diverse registrazioni, come le interpretazioni di Karajan e Bernstein, alcune storiche esecuzioni dirette da Furthwängler e Toscanini, le trascrizioni per pianoforte di Liszt della Quinta di Beethoven eseguite da Glenn Gould, come pure le versioni sintetizzate di un corrispondente file di tipo *Musical Instrument Digital Interface* (MIDI). Interpretazioni differenti della Quinta di Beethoven spesso rivelano grandi variazioni riguardo il tempo, la dinamica, l'articolazione, l'accordatura o l'orchestrazione.

La grande varietà dei documenti di tipo musicale.

Come illustrato nell'esempio di Beethoven, per la musica esistono una grande quantità di dati digitalizzati come pure una varietà di possibili rappresentazioni ad essa associate, che la descrivono a vari livelli semantici. Nel campo del MIR, grandi sforzi sono stati diretti verso lo sviluppo di tecnologie che permettano all'utente di esplorare e di accedere alla musica in tutte le sue sfaccettature.

Ad esempio, durante l'ascolto di alcuni brani musicali, un ipotetico riproduttore musicale del futuro potrebbe mostrare il corrispondente spartito, evidenziando la posizione raggiunta dall'esecuzione istante per istante. Su richiesta,

Il music player del futuro.

anche ulteriori informazioni sulla melodia e la progressione armonica, o sul tempo e il ritmo potrebbero essere presentate all'ascoltatore. Un'interfaccia apposita potrebbe mostrare la struttura musicale del brano e consentire all'utente di saltare direttamente ad ogni parte chiave della registrazione, senza estenuanti ricerche sequenziali. Inoltre l'ascoltatore sarebbe fornito di un motore di ricerca per esplorare l'intera raccolta musicale secondo diverse modalità: potrebbe creare una *query* specificando una certa sequenza di note o un *pattern* armonico o ritmico, fischiettando una melodia (*query-by-humming*) o più semplicemente selezionando un breve estratto da una registrazione; il sistema quindi fornirebbe una lista ordinata con un *ranking* dei brani musicali disponibili nella raccolta, musicalmente correlati alla richiesta.

Funzionalità avanzate.

Ad esempio, fornendo un estratto di venti secondi di un'interpretazione di Bernstein del tema della Quinta di Beethoven (*query-by-example*), il sistema restituirebbe tutte le altre corrispondenti clip musicali della base di dati. Queste includerebbero la ripetizione del tema nell'esposizione o nella ricapitolazione dell'interpretazione stessa, come pure gli estratti corrispondenti in tutte le registrazioni dello stesso pezzo interpretate da altri direttori d'orchestra. Un motore di ricerca avanzato sarebbe inoltre capace di identificare il tema anche in presenza di variazioni significative, riuscendo così a trattare arrangiamenti, come le trascrizioni per pianoforte di Liszt, versioni sintetizzate, o versioni pop accompagnate ritmicamente. Altre richieste di un utente potrebbero riguardare la ricerca di brani musicali in grado di suscitare certe emozioni, o comunque emotivamente associabili ad altri pezzi musicali.

Anche se sono stati fatti significativi progressi nello sviluppo di riproduttori musicali avanzati, ci sono ancora molti problemi da risolvere nell'effettuare ricerche di tipo *content-based*, dovuti all'eterogeneità e alla complessità dei dati musicali.

Tipiche questioni dell'analisi automatica della musica.

Come dovrebbe essere progettato un tale sistema di ricerca, se le *query* dell'utente consistono in frammenti musicali fischiettati o in un breve estratto di una registrazione? Come si possono confrontare una rappresentazione simbolica della musica, come uno spartito, con la rappresentazione fisica fornita dalla sua forma d'onda memorizzata in un CD audio? Quali sono le nozioni di similarità adatte a catturare certi aspetti musicali specificati da un utente e allo stesso

tempo capaci di ignorare certe variazioni, dovute magari alla strumentazione o al fraseggio? Come si può individuare da una registrazione la struttura musicale, che si riflette nella ripetizione di schemi musicalmente correlati? Come riconoscere lo stesso brano eseguito da interpreti diversi, che fanno uso di diversi arrangiamenti e strumentazioni? Queste domande, strettamente legate all'analisi automatica della musica, rappresentano solo una minima parte dell'attuale area di ricerca di competenza del MIR.

1.1.2 *Il problema del riconoscimento*

La tesi di laurea di Riccardo Miotto.

Il paradigma *query-by-example* introdotto in precedenza può essere sfruttato in diversi contesti. La tesi di laurea di Riccardo Miotto [11] affronta il problema di riconoscere una performance sconosciuta (di cui si ignorano il titolo, l'autore e così via) in un database o in una raccolta di musica digitale (*audio-matching*). Egli prende in considerazione due approcci diversi:

WAVE-TO-MIDI: cerca dei file MIDI in un database che corrispondano ad una data registrazione. L'esecuzione sintetizzata può anche presentare delle differenze nel tempo e nella tonalità, rispetto alla *query*.

WAVE-TO-WAVE: l'obbiettivo è recuperare da un database tutte le registrazioni audio che in qualche modo rappresentano lo stesso contenuto musicale della *query* audio. Questo è tipicamente il caso in cui lo stesso pezzo musicale è disponibile in diverse interpretazioni ed arrangiamenti.

Entrambe le metodologie proposte si basano su un modello matematico noto come *Hidden Markov Models* (HMM): l'idea è di modellare quei processi fisici non osservabili che sono alla base di una performance musicale di un file MIDI o di un file WAVE. Senza entrare nei dettagli, il brano viene scomposto in una successione di frammenti, che corrispondono a una sequenza di stati del modello collegati da transizioni probabilistiche.

1.1.3 *La segmentazione audio*

Nel caso dell'approccio *wave-to-wave* questa operazione non è banale, in quanto i frammenti in pratica devono corrispon-

dere a ciascuna nota suonata e questo a partire da una registrazione nella quale si sovrappongono i suoni di diversi strumenti, che interagiscono tra loro, oltre che con l'inevitabile rumore.

L'algoritmo di Choi.

Il problema della segmentazione audio, viene affrontato da Miotto con un algoritmo proposto da Choi [3] e nato nell'ambito della segmentazione testuale; serve infatti a delimitare in un testo scritto i brani che trattano di argomenti diversi. La versione adattata all'ambito musicale consiste fondamentalmente di quattro passi:

1. La cosiddetta *feature extraction*: il segnale audio viene processato tramite la *Short Time Fourier Transform* (STFT).

2. A partire dai valori ottenuti al passo precedente viene costruita una matrice di auto-similarità (*self-similarity*).

3. La matrice viene modificata con un procedimento di *ranking*, con l'intento di rendere più semplice il passo successivo.

4. La fase finale in cui si applica un algoritmo di clustering per ottenere la segmentazione vera e propria.

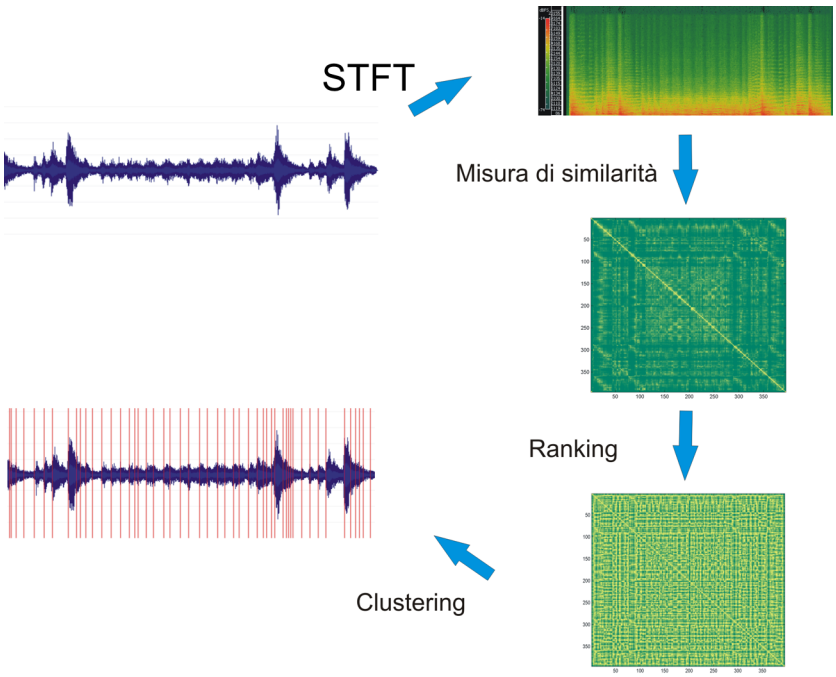


Figura 1: L'algoritmo di segmentazione usato nella tesi.

L'OBBIETTIVO DELLA MIA TESI è di migliorare, se possibile, la costruzione della matrice di auto-similarità testando diversi indici di similarità e di verificare l'efficacia della fase 3 (*ranking*) e dell'algoritmo nel suo complesso.

1.2 STRUTTURA DELLA TESI

La tesi è strutturata nel modo seguente:

- CAPITOLO 2: contiene la descrizione del problema del rilevamento dell'*onset* e della prima parte dell'algoritmo utilizzato da Miotto.
- CAPITOLO 3: descrive la parte sperimentale della tesi, che ha riguardato il comportamento di diversi indici di similarità per la costruzione della matrice di auto-similarità.
- CAPITOLO 4: l'ultima fase dell'algoritmo, il *clustering* ed ulteriori considerazioni di natura sperimentale.
- CAPITOLO 5: le conclusioni e possibili sviluppi futuri.

2

ONSET DETECTION E SIMILARITÀ

I buoni matematici riescono a vedere le analogie.
I grandi matematici riescono a vedere
le analogie tra le analogie

— Stefan Banach (1892-1945)

La musica è un fenomeno basato su una successione di eventi, sono i continui cambiamenti che le danno significato, che sollecitano chi ascolta a muoversi al suo ritmo e chi la esegue a concentrarsi sulla nota che segue. Anche nella musica che non è basata sulle note ci sono transizioni nel timbro e nella tonalità. La sua segmentazione è quindi l'individuazione di questi intervalli temporali successivi, all'interno dei quali la si possa considerare auto-similare rispetto ad una qualche sua proprietà.

La segmentazione audio.

A questo proposito occorre osservare [1] come il termine segmentazione abbia nell'ambito musicale due diversi significati: uno collegato alla musicologia e riferito al suo livello di rappresentazione simbolico (ad esempio la divisione di una composizione nelle sue varie parti), l'altro collegato al livello fisico dell'evento sonoro e quindi utilizzato quando si affronta la musica in termini di segnale acustico. In questo secondo caso, che è quello di nostro interesse, l'obiettivo è quello di segmentare il segnale musicale in frammenti delimitati da due eventi successivi. Gli eventi in questione sono qualsiasi fenomeno che alteri il *pattern* del brano musicale: potrebbe essere l'inserimento di un nuovo strumento, come il suo ridursi a silenzio o il cambiamento di una nota.

Anche la scala della segmentazione può variare molto: può essere molto fine, effettuando la segmentazione a livello di nota, o più generale riferendosi a diverse sorgenti o classi di audio (come la separazione tra il parlato e la musica di sottofondo). Segmentazioni intermedie possono essere quelle che rispondono a criteri di timbro o di strumenti, chiaramente la scala dipende dall'applicazione che richiede

la segmentazione. Nel nostro caso si tratta di individuare l'inizio delle singole note, che possono anche essere multiple, provenendo da uno o più strumenti, come nel caso della musica polifonica. Questo problema in letteratura è noto come *onset detection*.

2.1 ONSET DETECTION

Esistono molti algoritmi studiati per risolvere il problema dell'*onset detection* ed ogni anno ne vengono proposti di nuovi, alcuni hanno raggiunto buoni risultati ma spesso sono troppo dipendenti da alcuni fattori, come il tipo di musica analizzato, e non hanno validità generale. Un'interessante rassegna si può trovare nell'articolo di Bello et al. [2].

2.1.1 *Alcune definizioni: transitorio, onset, attacco.*

E' importante, a questo punto, fare una chiara distinzione tra i concetti correlati di transitorio, *onset* e attacco.[1]

L'attacco.

A. L'attacco della nota è l'intervallo di tempo durante il quale l'inviluppo dell'ampiezza aumenta.

Il transitorio.

B. Il concetto di transitorio è più difficile da descrivere con precisione. Una definizione informale può essere questa: il breve intervallo di tempo durante il quale il segnale evolve velocemente in un modo non banale o relativamente impredicibile. Nel caso degli strumenti acustici, il transitorio spesso corrisponde al periodo durante il quale l'eccitazione (ad esempio il colpo di un martelletto di pianoforte) è applicata e quindi smorzata, lasciando solo il decadimento del suono alla frequenza di risonanza del corpo dello strumento.

   La questione della risoluzione temporale utile è centrale in questo problema: in genere si assume che l'orecchio umano non possa distinguere tra due transitori separati da meno di 10 ms [12]. Da notare che l'estinzione (*release*) di un suono sostenuto può essere considerata come un transitorio.

L'onset.

C. L'*onset* della nota è un singolo istante scelto per deli-

1 Per descrivere l'inviluppo di una nota viene spesso usato anche il modello (A)ttack - (D)ecay - (S)ustain - (R)elease.

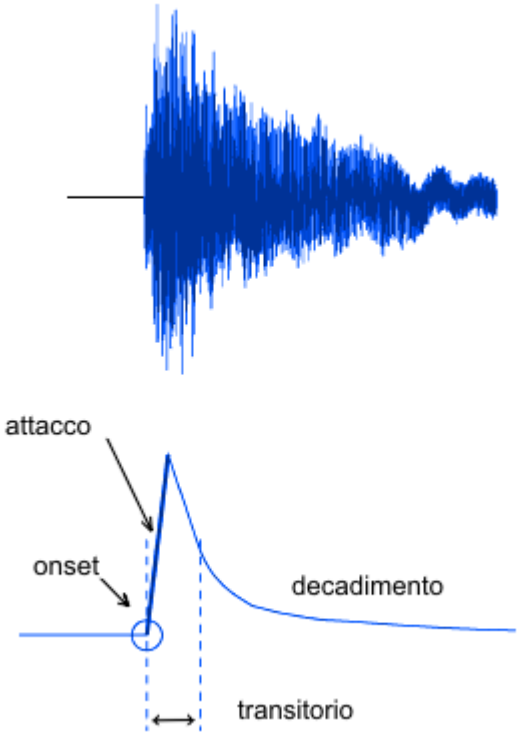


Figura 2: Attacco, transitorio, decadimento, onset nel caso ideale di una singola nota (tratta da [2]).

mitare l'estensione temporale di un transitorio. Nella maggior parte dei casi, coinciderà con l'inizio del transitorio, o con il primissimo istante nel quale il può essere rilevato con attendibilità.

2.1.2 *L'approccio tradizionale*

L'algoritmo tradizionale.

Nel caso più realistico di un segnale polifonico a cui si può anche aggiungere del rumore, le definizioni appena date diventano meno precise. I segnali audio sono sia additivi (gli oggetti musicali nella musica polifonica si sovrappongono e si mascherano l'un l'altro) che oscillatori. Quindi, non è possibile cercare le variazioni nel segnale audio semplicemente differenziando l'originale nel dominio del tempo, bisogna farlo in un segnale intermedio, che rifletta, in modo semplificato, la struttura locale dell'originale. Questa funzione di rilevamento in letteratura è indicata sia come *detection function* che come *novelty function*. Nella figura 3 è illustrato il procedimento utilizzato nella maggioranza degli algoritmi per la rilevazione degli *onset*: dal segnale audio originale, che può essere pre-elaborato per migliorare l'efficacia dei

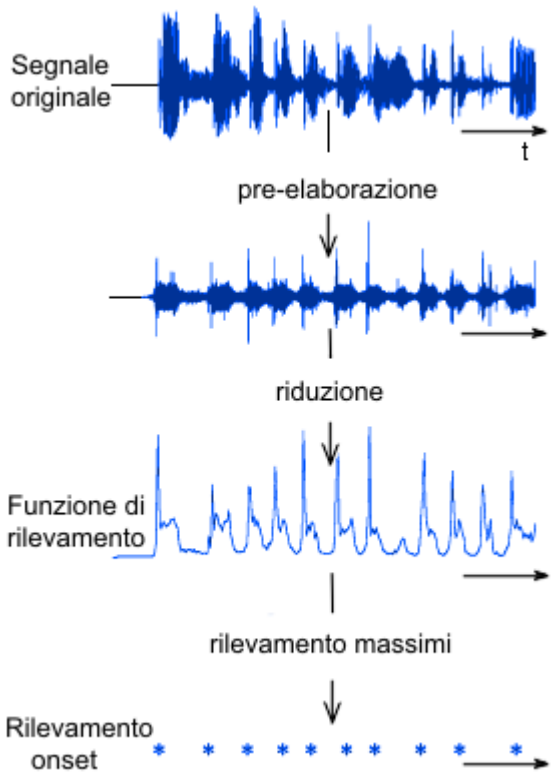


Figura 3: Le fasi di un algoritmo standard per la rilevazione degli *onsets* (tratta da [2]).

passi successivi, si deriva una funzione di rilevamento che ha una velocità di campionamento inferiore, alla quale si applica un algoritmo di ricerca dei massimi per localizzare gli *onsets*.

PRE-ELABORAZIONE: il suo compito è di trasformare il segnale originale per accentuarne o attenuarne vari aspetti in funzione dell'operazione successiva. I due processi più citati in letteratura sono la separazione del segnale in bande multiple di frequenza e la separazione tra la fase transitoria e quella sostenuta del suono (mutuata dalle tecniche di modellazione).

Una classificazione dei metodi di riduzione.

RIDUZIONE: è la parte più difficile e stimolante per gli studiosi. Riporterò una semplice classificazione che certo non può essere onnicomprensiva circa gli sviluppi più recenti, anche le caratteristiche (*features*) del segnale prese in considerazione (come ad es. le *chroma-features*) non sono elencate tutte.

A. Riduzione basata sulle caratteristiche del segnale.

a) Analisi nel dominio temporale: si può osservare come l'apparire di un *onset* è accompagnato di solito da un incremento nell'inviluppo del segnale. I primi metodi usavano una funzione che seguiva l'inviluppo dell'ampiezza del segnale: *Analisi nel tempo.*

$$E_0(n) = \frac{1}{N} \sum_{m=-\frac{N}{2}}^{\frac{N}{2}-1} |x(n+m)|w(m)$$

dove $w(m)$ è una finestra larga N punti (*smoothing kernel*) centrata in $m = 0$. Una variazione è seguire l'energia locale

$$E(n) = \frac{1}{N} \sum_{m=-\frac{N}{2}}^{\frac{N}{2}-1} [x(n+m)]^2 w(m)$$

elevando al quadrato, invece di rettificare, ogni campione. Ci sono stati successivi raffinamenti di questo metodo, come considerare la derivata dell'energia o la derivata di $\log E(n)$ [2], ma si è dimostrato utile solo per certe applicazioni che prevedono forti transitori di tipo percussivo. Può essere comunque usato in combinazioni con altri metodi.

b) Analisi nel dominio spettrale: sono metodi che riducono la necessità di una pre-elaborazione e che si sono dimostrati più efficaci, anche in presenza di segnali polifonici. Alla base vi è il calcolo della STFT: *Analisi in frequenza con il modulo.*

$$X_k(n) = \sum_{m=-\frac{N}{2}}^{\frac{N}{2}-1} x(nh+m)w(m)e^{-\frac{2j\pi mk}{N}} \quad (2.1)$$

dove $w(m)$ è ancora una finestra di N punti e h è lo slittamento nel tempo (*hop-size*) delle stesse. I valori così ottenuti possono, ad esempio, essere sommati dopo essere stati pesati (l'energia è concentrata alle basse frequenze, i cambiamenti dovuti ai transitori sono più rilevabili alle alte

2 La psicoacustica insegna che l'intensità (*loudness*) è percepita logaritmicamente, quindi i cambiamenti della stessa sono percepiti in relazione a tutto il segnale, poiché $d(\log E)/dt = (dE/dt)/E$.

frequenze). Ma un approccio più generale basato sui cambiamenti dello spettro è quello di formulare la funzione di rilevamento come distanza tra successivi spettri di Fourier *short-time*, trattandoli come punti di uno spazio N-dimensionale. A seconda della metrica usata si ottengono diverse differenze spettrali, dette anche *spectral-flux*. Un esempio è la norma L_2 sulla differenza rettificata:

$$SD(n) = \sum_{k=\frac{N}{2}}^{\frac{N}{2}-1} \{H(|X_k(n)| - |X_k(n-1)|)\}^2$$

dove $H(x) = (x + |x|)/2$, vale quindi 0 per gli argomenti negativi. Con la rettificazione si contano solo quelle frequenze per le quali c'è un incremento nell'energia, per enfatizzare di più gli *onsets* rispetto agli *offset*.

Analisi in frequenza con la fase.

c) Analisi nel dominio spettrale utilizzando la fase: invece di considerare solo il modulo dello spettro, prendono in considerazione l'informazione temporale racchiusa nella fase. Alcune funzioni di rilevamento ispirate a questo metodo potete trovarle in Dixon [4].

Trasformata wavelet.

d) Analisi tempo-frequenza e analisi (*Time-Scale*): un'alternativa all'utilizzo dell'inviluppo temporale e dei coefficienti spettrali di Fourier è l'uso delle *Time Frequency Representations* (TFR), in pratica della trasformata *wavelet*.

B. Riduzione basata su modelli probabilistici.

Analisi basata su modelli probabilistici.

Sono basati sull'assunzione che il segnale possa essere descritto da un modello probabilistico. Quindi si può costruire un sistema che, attraverso l'inferenza statistica, stabilisce gli istanti in cui si hanno con maggiore probabilità variazioni brusche nel segnale, date le osservazioni disponibili. Naturalmente il successo di questa strategia dipende dall'aderenza del modello alla reale distribuzione dei dati e può essere quantificato utilizzando misure di verosimiglianza o criteri di selezione del modello di Bayes.

RILEVAMENTO DEI MASSIMI (PEAK-PICKING): se la *function detection* è stata ben progettata, gli *onsets* o altri eventi

simili si rifletteranno in una ben precisa caratteristica della funzione di rilevamento. Di solito, questo consiste nella creazione di massimi locali che variano nel valore e nella forma e sono mascherati da rumore, dovuto a rumore vero e proprio o ad altri aspetti del segnale che non hanno a che fare con l'*onset*, come il vibrato. A volte il rilevamento dei picchi, viene preceduto da una fase di post-elaborazione, come una correzione della forma della funzione per diminuire l'influenza del rumore (*smoothing*), la normalizzazione, la rimozione della componente continua (DC) e la definizione di una soglia fissa o adattativa.

2.2 LA MATRICE DI SIMILARITÀ

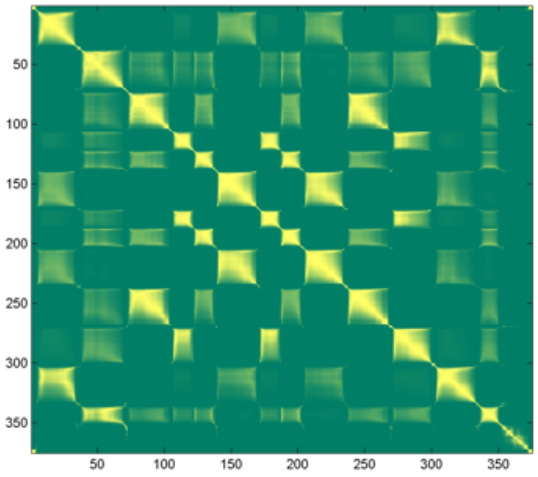
Come anticipato nel capitolo 1, il nostro algoritmo di segmentazione, cioè di rilevamento degli *onsets*, non segue la strategia appena vista ed è derivato da un metodo elaborato per la segmentazione testuale [3], che come primo passo costruisce una matrice di similarità. Infatti la segmentazione consiste nel trovare regioni contigue (nel tempo, per quanto riguarda la musica) che siano simili internamente e differenti dalle zone vicine.

L'idea di Foote.

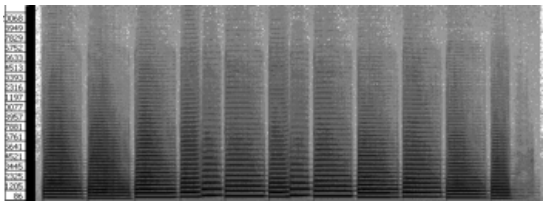
L'idea era già stata applicata alla musica da Foote, che inizialmente [6] la propose per visualizzarla su una scala temporale superiore alla nostra: generalmente la musica è auto-similare nel senso che possiede una struttura e delle ripetizioni. Ad esempio nella musica pop il secondo ritornello è molto simile al primo, come pure nella musica classica il tema assomiglia alle sue variazioni e questa struttura può essere evidenziata con la matrice di similarità. In seguito anche lui [7] ha proposto un metodo di segmentazione che si basa su di essa.

La luminosità è proporzionale alla similarità.

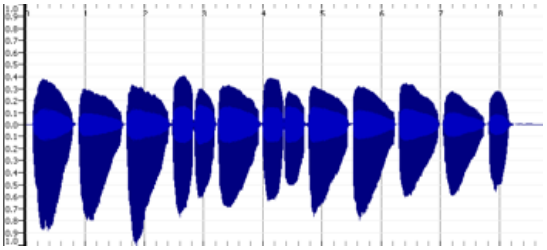
Una tale matrice si presenta come in figura 4: ogni lato del quadrato è proporzionale alla lunghezza del brano (un estratto della durata di alcuni secondi) ed il tempo scorre da sinistra verso destra e dall'alto verso il basso. Il brano è stato suddiviso in brevi intervalli di tempo (*frames*) e nella figura gli assi sono etichettati con il loro numero. L'angolo in alto a sinistra rappresenta l'inizio del brano, quello in basso a destra la fine. La luminosità nel punto (x, y) è proporzionale alla similarità tra il *frame* x e il *frame* y: le regioni simili sono le più luminose, all'aumentare della dissimilarità la luminosità diminuisce. In questo modo compare sempre una diagonale luminosa, in quanto ovviamente ogni frame



(a) matrice di similarità



(b) spettrogramma



(c) forma d'onda

Figura 4: Un breve estratto di 13 note dalla Promenade (*Tableaux d'une exposition*), di M. P. Musorgskij, suonato alla tromba in Si bemolle.

è uguale a se stesso [3] e su di essa si possono vedere le regioni che indicano le note dell'estratto musicale.

Costruzione della matrice.

Per ottenere la matrice si moltiplica il segnale audio (campionato) per una opportuna finestra e si applica la *Discrete Fourier Transform* (DFT) al segnale così ottenuto, poi si continua a traslare la finestra di un certo numero di punti (in modo da sovrapporsi alla precedente) ed a ripetere l'operazione finchè non si arriva alla fine del segnale: si utilizza in pratica la STFT (eq. 2.1 a pag. 13). In questo modo in corrispondenza di ogni avanzamento della finestra si ottiene

3 l'autocorrelazione di un segnale presenta sempre il massimo per $t = 0$

un vettore che rappresenta lo spettro di un breve *frame* del segnale. A questo punto si costruisce la matrice intermedia formata dagli elementi $\{x_{ij}\}$, dove $1 \leqslant i \leqslant S$ e $1 \leqslant j \leqslant F$. I vettori colonna di questa matrice sono l'energia di ogni *frame* ($|X_k|^2$), mentre S è il numero di punti con cui si è calcolata la DFT ed F è il numero dei *frames* ottenuti, che dipende ovviamente dalla lunghezza del segnale e dall'entità della traslazione della finestra (*hop-size*). E' tra questi vettori che si calcola la similarità con delle opportune formule che riflettono la distanza tra di loro.

Si valuta la similarità tra l'energia dei frames.

Quindi anche la matrice di similarità , come lo *spectral-flux*, calcola delle differenze spettrali, ma in un certo senso a livello globale, non si limita cioè a calcolare la differenza tra ciascun *frame* e quello che lo precede, ma tra ciascun *frame* e tutti gli altri.

Nel suo articolo Choi, che utilizzava le frequenze con cui le parole comparivano nel testo, utilizza la distanza coseno:

$$\text{Coseno} \qquad \frac{\sum_{i=1}^{S} x_{ij} x_{ik}}{\sqrt{\sum_{i=1}^{S} x_{ij}^2}\sqrt{\sum_{i=1}^{S} x_{ik}^2}} \tag{2.2}$$

dove x_{ij} e x_{ik} sono i due vettori (*power spectrum*) tra cui si calcola la distanza. Nella prossima sezione elencherò gli indici da me usati.

2.2.1 *Indici di similarità*

La similarità è una quantità che riflette il grado di relazione tra due oggetti o caratteristiche. Non è semplice da misurare e la sua scelta dipende ovviamente dal contesto in cui si opera. Di solito viene espressa con un indice $s_{jk} \in [0, 1]$: un valore pari a 0 indica idealmente una assoluta differenza tra gli oggetti in questione, mentre un valore pari a 1 indica la loro identità (nel caso dell'indice coseno (2.2), utilizzato nell'algoritmo originale, il valore varia tra -1 e 1, con ovvia interpretazione dei valori).

L'indice di similarità.

Un modo diretto per calcolarsi un tale indice è quello di notare che una funzione distanza (d_{jk}) rappresenta una misura della dissimilarità tra due oggetti, quindi se δ_{jk}

è la sua versione normalizzata nell'intervallo $[0,1]$, si ha $s_{jk} = 1 - \delta_{jk}$ [4]. Utilizzando queste tre distanze:

$$\text{MANHATTAN} \quad (L_1) \qquad \sum_{i=1}^{S} |x_{ij} - x_{ik}| \tag{2.3}$$

Indici di similarità ricavati da distanze.

$$\text{EUCLIDEA} \quad (L_2) \qquad \sqrt{\sum_{i=1}^{S} (x_{ij} - x_{ik})^2} \tag{2.4}$$

$$\text{CHEBYSHEV} \quad (L_\infty) \qquad \max_{1 \leqslant i \leqslant S} |x_{ij} - x_{ik}| \tag{2.5}$$

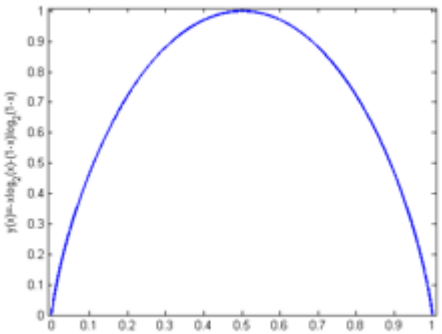
ho ricavato altrettanti indici di similarità con la formula

$$s_{jk} = 1 - d_{ij} / \sqrt{d_{jk}^2 + c} \qquad d = (L_1, L_2, L_\infty), \quad c \geqslant 0.$$

L'indice C-L.

Un altro indice è stato ricavato dalla lettura dell'articolo di Gang et al. [8], che propone una sorta di media pesata tra l'indice coseno, che rende conto della diversa *direzione* dei due vettori e un indice funzione della norma, che rappresenta la loro *lunghezza*.

A partire da $f(x) = -x \log_2 x - (1-x) \log_2 (1-x), x \in [0,1]$, che raggiunge il suo massimo 1 per $x = 0,5$ e il suo minimo 0 per $x = 0$ e per $x = 1$, come illustrato in figura:



definisce in questo modo il nuovo indice di similarità

$$\begin{aligned} d_{log}(a,b) = &- \frac{\|a\|}{\|a\| + \|b\|} \log_2 \left(\frac{\|a\|}{\|a\| + \|b\|} \right) \\ &- \frac{\|b\|}{\|a\| + \|b\|} \log_2 \left(\frac{\|b\|}{\|a\| + \|b\|} \right) \end{aligned} \tag{2.6}$$

4 per la distanza coseno si può usare $s_{jk} = 0,5(1 + \delta_{jk})$

che vale 1 quando la *lunghezza* di a e b sono uguali, mentre la sua prossimità al valor 0 indica una forte dissimilarità; essendo

$$d_{cos}(a, b) = \frac{a \cdot b}{\|a\| \, \|b\|} \tag{2.7}$$

si ottiene infine l'indice di similarità che ho utilizzato:

$$s(a, b) = \frac{\alpha}{2}(1 - d_{cos}(a, b)) + (1 - \alpha)(1 - d_{log}(a, b)) \tag{2.8}$$

dove $\alpha \in [0, 1]$. La norma da me utilizzata è la L2, mentre $\alpha \in \{0,25; 0,5; 0,75; 0,9\}$. Naturalmente nel nostro caso $a = x_{i,j}$ e $b = x_{i,k}$. D'ora in poi mi riferirò a questo indice indicandolo come indice C-L.

Sono stati poi utilizzati anche i seguenti indici di similarità, ricavati da diversi ambiti di studio:

Gli altri indici di similarità usati.

$$\text{Bray Curtis} \qquad \frac{2\sum_{i=1}^{S} \min(x_{ij}, x_{ik})}{\sum_{i=1}^{S}(x_{ij} + x_{ik})} \tag{2.9}$$

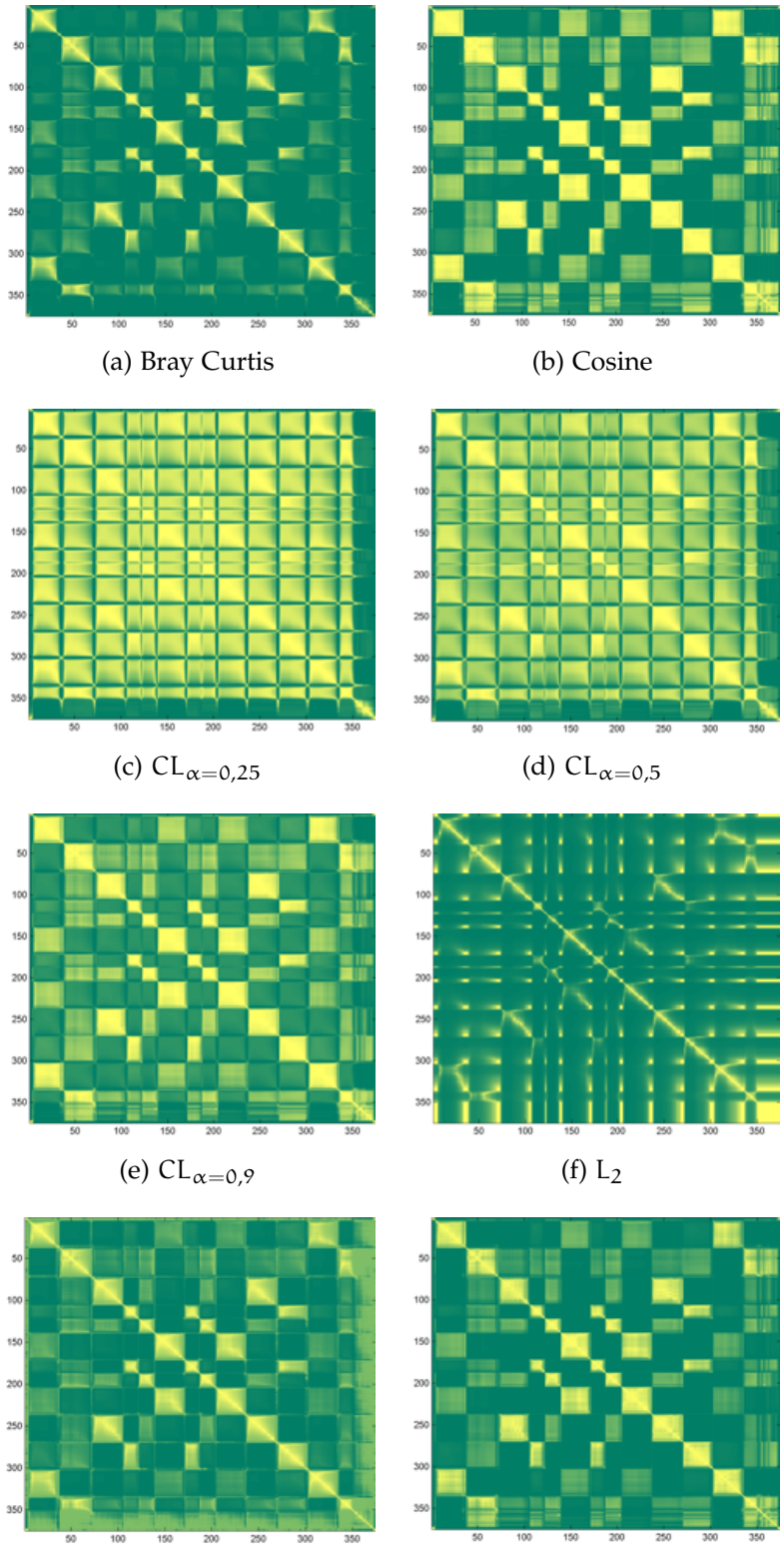
$$\text{Kulczynsky} \quad 0,5\left[\frac{\sum_{i=1}^{S} \min(x_{ij}, x_{ik})}{\sum_{i=1}^{S} x_{ij}} + \frac{\sum_{i=1}^{S} \min(x_{ij}, x_{ik})}{\sum_{i=1}^{S} x_{ij}}\right] \tag{2.10}$$

$$\text{Modified Chord} \qquad 1 - \sqrt{1 - \frac{\sum_{i=1}^{S} x_{ij}x_{ik}}{\sqrt{\sum_{i=1}^{S}(x_{ij})^2 \sum_{i=1}^{S}(x_{ik})^2}}} \tag{2.11}$$

$$\text{Ruzicka} \qquad \frac{\sum_{i=1}^{S} \min(x_{ij}x_{ik})}{\sum_{i=1}^{S} \max(x_{ij}x_{ik})} \tag{2.12}$$

$$\text{Similarity Ratio} \qquad \frac{\sum_{i=1}^{S} x_{ij}x_{ik}}{\sum_{i=1}^{S}(x_{ij})^2 + \sum_{i=1}^{S}(x_{ik})^2 - \sum_{i=1}^{S} x_{ij}x_{ik}} \tag{2.13}$$

E' interessante, a questo punto, un confronto visivo tra le matrici ottenute con diversi indici di similarità.



(a) Bray Curtis (b) Cosine

(c) $CL_{\alpha=0,25}$ (d) $CL_{\alpha=0,5}$

(e) $CL_{\alpha=0,9}$ (f) L_2

(g) Kulczynski (h) Modified Chord

Figura 5: Il breve estratto di 13 note dalla Promenade visualizzato con diversi indici di similarità.

2.3 LA RANKING MATRIX

Nella seconda fase dell'algoritmo si calcola la matrice di *ranking* a partire dalla matrice di similarità. Questo procedimento è giustificato da Choi con il fatto che i parametri di similarità non sono completamente affidabili per dei brevi estratti di testo, ma piuttosto stimino l'ordine di similarità tra di essi, consentendo solo di poter affermare, ad esempio, se a è più simile a b, piuttosto che a c. Anche nell'adattamento proposto da Miotto si segue questa linea, giustificata dalla grande variabilità del segnale nel suo complesso e dalla brevità dei *frames*. Quindi confrontare direttamente i valori di similarità da diverse zone della matrice non è conveniente, in quanto può succedere che due *frames* distanti siano molto simili a differenza di quelli intermedi, creando problemi nel processo di segmentazione.

Il ranking, come nell'analisi statistica parametrica.

Nell'analisi statistica parametrica, quando il comportamento qualitativo è simile, ma le quantità assolute sono inaffidabili si calcola il *rank* degli insiemi di dati. Nello schema utilizzato in questo caso, ogni elemento della matrice di similarità è rimpiazzato dal suo *rank* nella regione locale. Il *rank* è definito come il numero di elementi vicini con un valore di similarità inferiore. La figura 6 illustra un esempio di *ranking* dell'immagine che utilizza una maschera 3x3 con un *range* di valori $\in \{0, \ldots, 8\}$. Per la segmentazione testuale era utilizzata una maschera 11x11, nella segmentazione musicale, dopo alcuni tentativi, si è visto che funziona meglio una maschera 9x9 e questo è il valore anche da me utilizzato nella sperimentazione. Da notare che i valori ottenuti sono

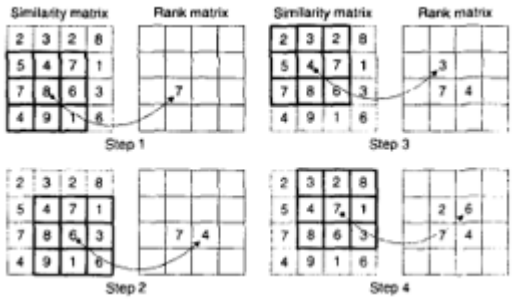


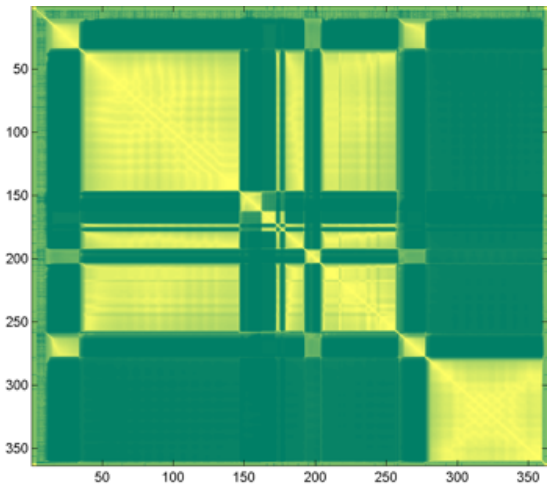
Figura 6: Come si utilizza la *ranking mask* (tratta da [3]).

poi espressi come un rapporto r, per evitare problemi di

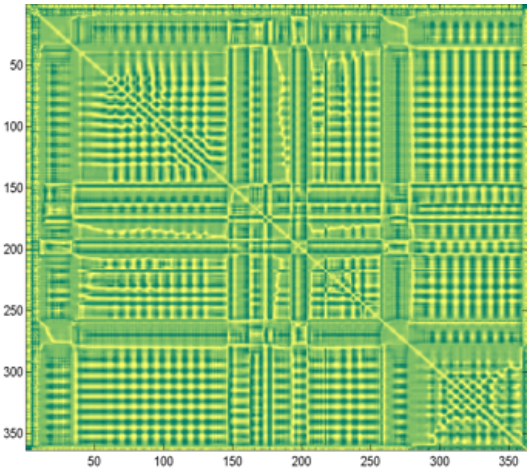
normalizzazione (si consideri il caso in cui la maschera non è contenuta interamente nell'immagine), in questo modo:

$$r = \frac{\text{numero di elementi con un valore inferiore}}{\text{numero di elementi esaminati}}$$

Dalla figura 7 si può notare come il procedimento di *ranking* aumenti il contrasto ed evidenzi i bordi delle regioni di similarità.



(a) matrice di similarità



(b) *ranking matrix*

Figura 7: Matrice di similarità e *ranking matrix* a confronto.

3

VERIFICA SPERIMENTALE

Gli errori commessi usando dati inadeguati
sono molto minori di quelli fatti
senza usare nessun dato.

— Charles Babbage (1792-1871)

Prima di passare all'ultima fase dell'algoritmo, ho cercato di capire quale potesse essere l'indice di similarità migliore per il nostro scopo. Visto che nessuno di essi è stato realizzato specificatamente per la segmentazione musicale è indispensabile trovare una maniera per testarli, magari in modo indipendente dall'algoritmo di *clustering* successivo.

La segmentazione manuale.

Per disporre di valori con cui confrontare l'efficacia degli indici ho deciso di segmentare manualmente gli estratti musicali. Ascoltando i brani ho memorizzato in un file gli istanti degli *onsets* che percepivo e gli ho confrontati con i minimi della *detection function* (2.1.2) ricavata dalla matrice di similarità, ripetendo il procedimento per ciascun indice. Tale funzione infatti indica quanto ogni *frame* sia simile al precedente (nel nostro caso si confronta l'energia), quindi un minimo marcato segnala l'inizio di un nuovo intervallo temporale all'interno del quale i *frames* manifestano più similarità tra di loro, rappresentando ad esempio una nota, nel caso di un brano monofonico.

Come ottenere la detection function dalla matrice di similarità.

La funzione di rilevamento si ottiene in modo immediato dalla matrice di similarità: basta prendere i valori $(i, i+1)$, cioè quelli al di sopra la diagonale principale [1]. Infatti nella matrice di similarità il tempo scorre dall'angolo in alto a sinistra fino in basso a destra, quindi (i, i) rappresenta il grado di similarità dell'i-esimo frame con se stesso, $(i, i+1)$ con il frame immediatamente precedente, $(i, i+2)$ con il frame che lo precede di due posizioni e così via. In questo

1 Si può utilizzare anche la diagonale al di sotto, visto che la matrice di similarità è simmetrica.

modo si formano le strisce verticali che dividono le zone quadrate di similarità lungo la diagonale.

Confrontando i valori di minimo delle *detection functions* ottenute con i vari indici di similarità e la loro distanza temporale dall'*onset* percepito, ho cercato di capire quale fossero gli indici più utili alla segmentazione.

3.1 RACCOLTA DEI DATI

Il software Sonic Visualiser.

Per procedere alla segmentazione manuale mi sono avvalso di *Sonic Visualiser*, un software sviluppato presso il Centro per la Musica Digitale della *Queen Mary University* di Londra; liberamente scaricabile all'indirizzo `http://www.sonicvisualiser.org/` e distribuito con la licenza GNU *General Public License*. E' un applicativo pensato per musicologi e ricercatori, non si tratta di un vero e proprio *editor* audio, ma ha molte funzioni utili (tra cui anche *plugins* che implementano l'*onset detection*...) e io l'ho utilizzato per memorizzare, durante l'ascolto, gli istanti in cui percepivo gli *onsets* premendo un tasto al momento opportuno. Di fondamentale importanza si sono rivelate la possibilità di rallentare i brani a piacimento, per raggiungere una certa precisione e il segnale acustico introdotto dal *software* che si sovrapponeva al brano, per poter poi valutare con vari ascolti l'esattezza della propria scelta.

3.1.1 *Set-up sperimentale*

Per la sperimentazione ho utilizzato estratti musicali monoaurali dalla durata variabile tra i 6 e gli 11 secondi campionati alla frequenza degli audio CD (44100 Hz). Sono stati tratti da brani di musica classica, cercando di ottenere una certa varietà di timbri, registri e velocità. Ecco l'elenco dei 28 estratti monofonici:

Gli estratti dai brani monofonici.

- Sax contralto: 3 estratti dal brano *Il vecchio castello* (*Tableaux d'une exposition*) di M. P. Musorgskij.
- Tuba bassa: 3 estratti dal brano *Bydlo* (*Tableaux d'une exposition*), di M. P. Musorgskij.
- Fagotto: 2 brani di Y. Makau, compositore contemporaneo turco.
- Clarinetto: 4 estratti della *Sonata Op. 120 n.1*, di J. Brahms

- Tromba in Si bemolle: 3 estratti dalla *Promenade*, (*Tableaux d'une exposition*), di M. P. Musorgskij.
- Violoncello: 4 brani estratti dal Preludio della *Suite per Cello Solo n.2*, di J. S. Bach.
- Flauto: 1 cadenza e 3 estratti da *Syrynx*, di C. Debussy.
- Sassofono soprano: 3 estratti dalla *Partita per flauto BWV 1013*, di J. S. Bach, trasposta in sol.
- Violino: 2 estratti da *Aria da Capo*, *Goldberg-Variationen*, di J. S. Bach.

Gli estratti dai brani polifonici.

Questi sono gli estratti dei brani polifonici:

- J. S. Bach: Allegro, *Concerto brandeburghese n.5 in Si Maggiore BWV 1050* (3 estratti).
- C. Debussy: Golliwogg's cakewalk, *Children's Corner* (3 estratti), *Arabesque n.1* (2 estratti).
- L. Francioso: *O' Mar*, arrangiamento per quartetto d'archi ad opera di Filippo Bovo (3 estratti).
- W. A. Mozart: Allegro, *Hornkonzert n.1 K 412* (2 estratti), incipit dall'Allegro del *Konzert für Flöte und Harfe K 299* (1 estratto).
- R. A. Schumann: *Sinfonische Etüden op. 13*, (4 estratti).
- I. F Stravinskij: *Andante della Suite n. 1* per piccola orchestra (3 estratti).
- P. I. Čajkovskij: *Russian Dance - Trepak* (2 estratti), *Dance of the Reedpipes* dalla Suite *Lo schiaccianoci* (2 estratti).

3.1.2 *Sperimentazione*

Per calcolare le matrici di similarità e di *ranking* ho usato Matlab® utilizzando il metodo indicato nelle sezioni 2.2, 2.2.1 e 2.3 [2], i parametri per la STFT sono stati:

Parametri utilizzati per la STFT.

- 4096 punti per la DFT

2 Chi lo desidera, può provare il MIRToolbox scaricabile all'indirizzo https://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/mirtoolbox, contenente molte funzioni già implementate allo scopo.

- 2048 punti per la finestra di Hamming
- 1024 punti di avanzamento della finestra (*hop-size*)

in questo modo si sono sovrapposte le finestre del 50% e considerati *frames* della durata di circa 45,6 millisecondi.

Una prima verifica visiva.

Una volta calcolate le matrici, per prima cosa le ho visualizzate, sovrapponendole all'immagine dei quadrati disegnati dai *bounds* manuali: per i brani monofonici, i più facili, lo scostamento tra i *bounds* percettivi e i bordi evidenziati dalle matrici è minimo per quasi tutti gli indici di similarità. Anche se già a questo punto si possono notare matrici in un certo più promettenti, in quanto delineano zone quadrate più definite lungo la diagonale in corrispondenza dei *bounds* manuali, si è reso necessario un metodo più preciso per confrontare le prestazioni dei vari indici.

Utilizzo delle detection functions.

Si è pensato quindi di ricavare le *detection functions* dalle matrici di similarità nel modo indicato nell'introduzione al capitolo. A questo punto i parametri distintivi per un buon indice di similarità sono i valori dei minimi in prossimità degli istanti indicati dalla segmentazione manuale (assunta come riferimento) e la loro distanza temporale dagli stessi. Naturalmente vorremmo avere dei minimi che siano ben separati dal valore medio della funzione[3] e che si presentino in prossimità, ancora meglio in coincidenza, dell'istante dell'*onset* ricavato dalla segmentazione manuale.

Sono stati utilizzati tutti gli indici di similarità indicati nella sezione 2.2.1, come pure la distanza coseno (utilizzata da Choi) normalizzata tra 0 ed 1, per poterla confrontare con le altre, mentre l'utilizzo della correlazione è stato scartato, visto che i suoi picchi sembravano essere piuttosto imprecisi.[4] Si sono prese in considerazione sia le matrici di similarità che le matrici di *ranking*, mentre per quanto riguarda le distanze dagli *onsets* percepiti si è considerato il loro intorno di 5 *frames*: i valori minimi della *detection function* utilizzati nel calcolo dei parametri statistici sono quelli che cadono entro questo intorno. Quindi si considera al massimo un anticipo o un ritardo di 2 *frames* rispetto all'*onset* percepito.

3 Essendo gli indici di similarità normalizzati nell'intervallo $[0, 1]$, l'ideale sarebbe avere minimi con valori prossimi allo 0 e il valor medio della funzione vicino a 1.

4 Nel caso della correlazione i picchi utili sono i massimi.

3.2 ANALISI DEI DATI

I parametri statistici considerati per ogni indice di similarità e per ogni estratto musicale sono stati questi:

I parametri statistici descrittivi utilizzati.

- per quanto riguarda la *novelty function*:
  1. la media della funzione (μ_{nf}) ;
  2. la media dei minimi (μ_0);
  3. la varianza dei minimi (ν_0);
- riguardo le distanza (espressa in *frames*) tra l'istante in cui si presenta il minimo della funzione e l'istante in cui ho percepito l'*onset*:
  1. la media (mu_{dist});
  2. la varianza (ν_{dist});
  3. la curtosi (k_{dist});
  4. la *skewness* (s_{dist}).

Le medie aritmetiche e le varianze sono state scelte ovviamente per avere una misura della tendenza centrale e della dispersione dei campioni, la curtosi e la *skewness* per valutare la simmetria delle frequenze delle distanze dei minimi dal valore percepito. Infatti le frequenze rappresentano una prima approssimazione della distribuzione di probabilità delle distanze e chiaramente ci aspettiamo un andamento normale, del tipo gaussiano. La segmentazione manuale naturalmente è soggetta ad errori, ma eventuali anticipi o ritardi nel percepire gli *onsets* si compensano e questo vale anche per la segmentazione automatica.

La curtosi e la skewness.

Una curtosi bassa (vicino allo 0) indica che questa distribuzione è unimodale, mentre il suo valore elevato indica la presenza di *outliers*, una situazione che indicherebbe un'anomalia. La *skewness* misura la simmetria della distribuzione: un valore negativo indica uno spostamento verso destra rispetto alla media, quindi una tendenza al ritardo nel segnalare l'*onset*, un valore positivo indica il contrario.

Nelle tabelle 1, 2, 3 e 4 sono indicati i valori medi di questi parametri per ogni indice di similarità. Le medie sono state calcolate rispetto ai brani, separando i monofonici dai polifonici. Invece dei valori medi delle funzioni e dei loro minimi sono indicate la loro differenza percentuale media ($(m_{nf} - m_0)/m_{nf}$), una misura più significativa, visto che quello che interessa non è tanto il valore assoluto dei minimi quanto la loro capacità di staccarsi dal resto della funzione.

Tabella 1: Misure statistiche per i brani monofonici (matrice di similarità).

INDICI	$\frac{\mu_{nf}-\mu_0}{\mu_{nf}}$	ν_0	μ_{dist}	ν_{dist}	k_{dist}	s_{dist}
B. Curtis	0,627	0,018	1,150	0,451	2,164	0,363
$CL_{\alpha=0,25}$	0,287	0,036	1,141	0,518	1,820	0,343
$CL_{\alpha=0,50}$	0,257	0,021	1,167	0,466	1,924	0,292
$CL_{\alpha=0,75}$	0,244	0,014	1,172	0,470	1,995	0,411
$CL_{\alpha=0,90}$	0,243	0,013	1,182	0,484	2,037	0,446
Coseno	0,513	0,055	1,185	0,479	2,072	0,412
Kulczynski	0,477	0,015	1,160	0,470	2,063	0,441
L_1	0,389	0,042	0,960	0,546	2,161	0,315
L_2	0,480	0,049	0,927	0,545	2,188	0,257
L_∞	0,614	0,050	0,965	0,548	2,025	0,240
M. Chord	0,640	0,035	1,185	0,479	2,027	0,370
Ruzicka	0,714	0,010	1,150	0,450	2,164	0,363
S. Ratio	0,762	0,023	1,177	0,454	2,005	0,355

Tabella 2: Misure statistiche per i brani polifonici (matrice di similarità).

INDICI	$\frac{\mu_{nf}-\mu_0}{\mu_{nf}}$	ν_0	μ_{dist}	ν_{dist}	k_{dist}	s_{dist}
B. Curtis	0,300	0,027	1,159	0,498	2,066	0,494
$CL_{\alpha=0,25}$	0,130	0,029	1,098	0,554	1,911	0,332
$CL_{\alpha=0,50}$	0,121	0,019	1,137	0,537	2,001	0,356
$CL_{\alpha=0,75}$	0,119	0,012	1,136	0,540	1,981	0,395
$CL_{\alpha=0,90}$	0,122	0,010	1,165	0,542	1,981	0,396
Coseno	0,271	0,034	1,170	0,545	1,984	0,426
Kulczynski	0,215	0,010	1,155	0,515	2,015	0,471
L_1	0,187	0,036	1,106	0,578	1,913	0,065
L_2	0,239	0,050	1,0867	0,584	1,885	0,053
L_∞	0,361	0,066	1,082	0,568	1,938	0,085
M. Chord	0,383	0,030	1,170	0,545	1,984	0,426
Ruzicka	0,381	0,020	1,159	0,498	2,065	0,494
S. Ratio	0,483	0,046	1,142	0,533	2,000	0,382

Tabella 3: Misure statistiche per i brani monofonici (matrice di *ranking*).

INDICI	$\frac{\mu_{nf}-\mu_0}{\mu_{nf}}$	ν_0	μ_{dist}	ν_{dist}	k_{dist}	s_{dist}
B. Curtis	0,340	0,012	1,164	0,471	1,979	0,494
$CL_{\alpha=0,25}$	0,503	0,023	1,144	0,512	1,794	0,406
$CL_{\alpha=0,50}$	0,447	0,022	1,145	0,505	1,944	0,400
$CL_{\alpha=0,75}$	0,409	0,200	1,169	0,486	2,103	0,521
$CL_{\alpha=0,90}$	0,393	0,019	1,170	0,500	2,059	0,587
Coseno	0,375	0,018	1,194	0,507	2,040	0,520
Kulczynski	0,345	0,014	1,156	0,468	2,055	0,527
L_1	0,315	0,013	0,945	0,551	2,147	0,206
L_2	0,333	0,015	0,908	0,543	2,236	0,204
L_∞	0,342	0,016	0,939	0,578	2,050	0,154
M. Chord	0,375	0,018	1,194	0,507	2,034	0,487
Ruzicka	0,340	0,012	1,164	0,471	1,979	0,494
S. Ratio	0,372	0,015	1,172	0,476	2,004	0,533

Tabella 4: Misure statistiche per i brani polifonici (matrice di *ranking*).

INDICI	$\frac{\mu_{nf}-\mu_0}{\mu_{nf}}$	ν_0	μ_{dist}	ν_{dist}	k_{dist}	s_{dist}
B. Curtis	0,318	0,018	1,133	0,505	2,056	0,525
$CL_{\alpha=0,25}$	0,413	0,027	1,077	0,552	1,9147	0,324
$CL_{\alpha=0,50}$	0,378	0,025	1,117	0,528	2,013	0,411
$CL_{\alpha=0,75}$	0,362	0,024	1,124	0,537	1,986	0,429
$CL_{\alpha=0,90}$	0,368	0,025	1,139	0,534	1,990	0,463
Coseno	0,372	0,025	1,162	0,533	2,020	0,467
Kulczynski	0,332	0,022	1,132	0,526	1,976	0,452
L_1	0,390	0,033	1,062	0,563	1,925	0,011
L_2	0,425	0,030	1,070	0,575	1,893	0,017
L_∞	0,449	0,030	1,056	0,565	1,915	0,037
M. Chord	0,372	0,025	1,162	0,533	2,020	0,467
Ruzicka	0,318	0,018	1,133	0,505	2,056	0,525
S. Ratio	0,366	0,022	1,126	0,529	2,013	0,425

3.3 CONCLUSIONI

Dai valori ricavati si nota una sostanziale omogeneità di comportamento delle distanze dai minimi: i valori medi oscillano attorno a 1 (con una varianza vicina allo 0), la curtosi è bassa e la *skewness* indica un leggero spostamento a sinistra della curva delle frequenze. Questo potrebbe anche indicare un leggero costante ritardo della segmentazione manuale nella rilevazione dell'*onsets*.

Una diversa situazione si delinea nella differenza relativa tra i minimi e il valor medio della funzione, come emerge dai grafici in figura: la *similarity ratio* mostra un comportamento migliore, considerando sia i brani monofonici che quelli polifonici. Nella *ranking matrix* la situazione invece cambia, come mostrato nelle figure 8 e 9. L'effetto del procedimento con cui si ottiene il *ranking* sembra rendere omogenea la *novelty function*, appiattendo i minimi rispetto alla funzione (figure 10 e 11). Bisogna però considerare che questo metodo è utilizzato con una particolare tecnica di rilevamento degli *onsets* che vedremo nel prossimo capitolo e la sua efficacia può essere valutata solo in combinazione con questa.

Quindi dai dati emersi si può affermare che dovendo usare un metodo di rilevazione tradizionale, basato sulla *novelty function*, sembra essere preferibile l'indice *similarity ratio*, ma la valutazione dell'efficacia del *ranking* può essere valutata pienamente solo nell'ambito del particolare algoritmo usato.

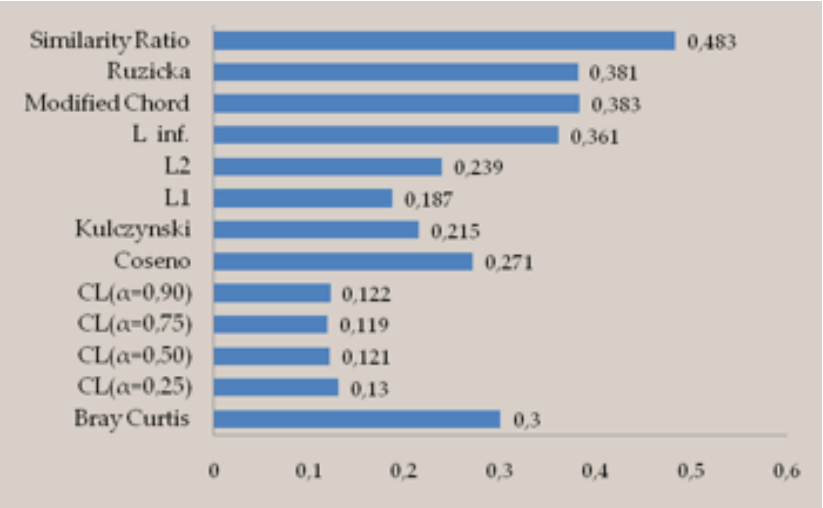


Figura 8: $\frac{\mu_{nf}-\mu_0}{\mu_{nf}}$ per i monofonici (*similarity matrix*).

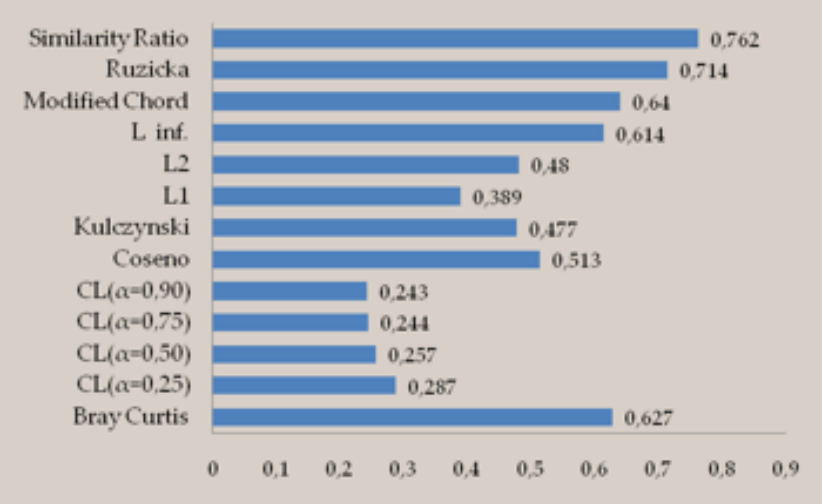


Figura 9: $\frac{\mu_{nf}-\mu_0}{\mu_{nf}}$ per i polifonici (*similarity matrix*).

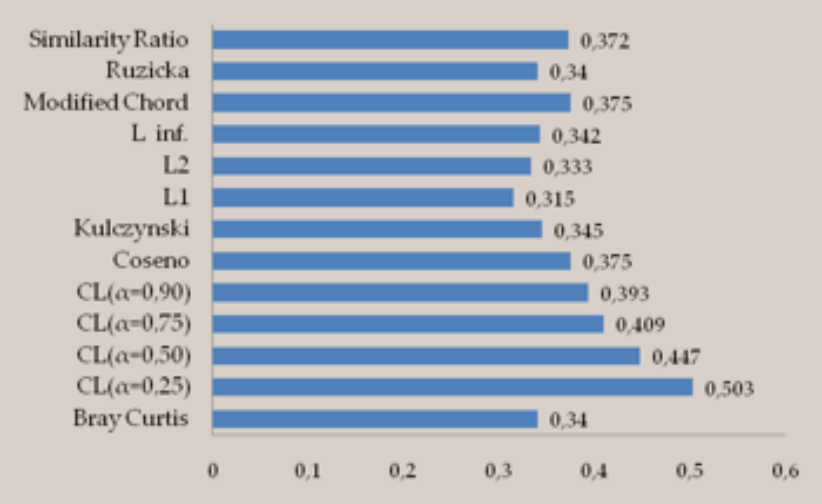


Figura 10: $\frac{\mu_{nf}-\mu_0}{\mu_{nf}}$ per i monofonici (*ranking matrix*).

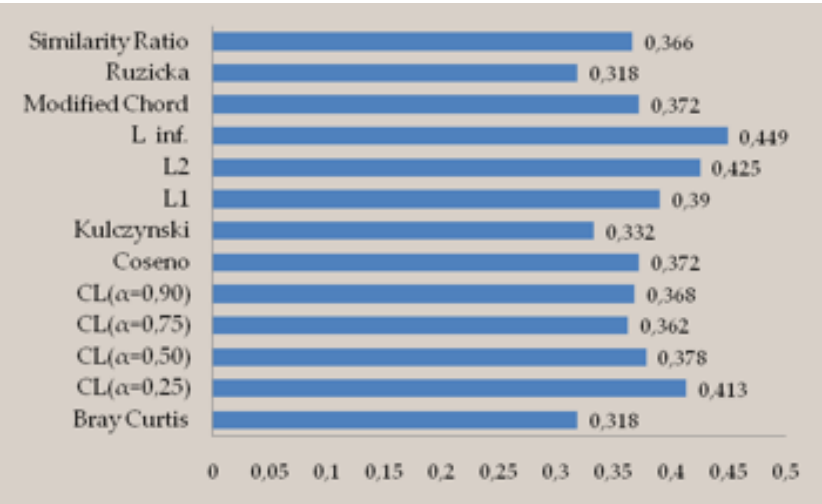


Figura 11: $\frac{\mu_{nf}-\mu_0}{\mu_{nf}}$ per i polifonici (*ranking matrix*).

4 LA SEGMENTAZIONE

La matematica analitica è un brano musicale legato,
che scivola senza crepe tra spazi continui;
la matematica combinatorica è uno staccato,
che salta, senza pensarci su un attimo, da intero a intero.

— John Derbyshire (1945-)
Unknown Quantity, 2007, pag. 284

Negli algoritmi tradizionali la ricerca degli *onsets* viene ricondotta alla rilevazione dei picchi in una funzione, in questo caso invece la si riconduce al *clustering*. Trovare una struttura composta da *clusters* in un insieme di dati è un problema comune a molte discipline, come la linguistica, la biologia molecolare e la psicometria. Spesso questi insiemi di dati sono caratterizzati da una misura della similarità (o dissimilarità) tra coppie di oggetti, non sono date cioè le loro coordinate in uno spazio metrico, non ci sono delle coordinate centrali. La soluzione proposta nell'algoritmo preso in esame è concettualmente molto semplice e si basa sull'osservazione che nella matrice di *ranking* sono già parzialmente visibili lungo la diagonale delle zone di similarità. In questo capitolo verrà descritto questo metodo e ne verrà testata l'efficacia, in particolare si confronteranno i risultati ottenuti con la *ranking matrix* e la distanza coseno (come nell'algoritmo generale) e quelli ottenuti con la matrice similarità e l'indice *Similarity Ratio*.

Il clustering: un insieme di tecniche dell'analisi multivariata dei dati.

4.1 L'ALGORITMO DI MASSIMIZZAZIONE DI REYNAR

Ciò che rende interessanti le matrici di segmentazione e di *ranking* è la loro capacità di visualizzare, in qualche modo, i *bounds*, tanto che si potrebbe pensare di identificarli a partire da queste. In effetti il metodo originale, proposto da Choi, una volta calcolata la matrice di *ranking* partendo dalla distanza coseno, utilizza l'algoritmo di massimizzazione

di Reynar [15]. Quest'algoritmo cerca la segmentazione che massimizza la densità interna dei segmenti.

La segmentazione basata sulla densità della matrice.

Un segmento musicale (ad esempio quello tra da due note) è delimitato da due *frames* ed è rappresentato da una regione quadrata lungo la diagonale della matrice in questione. Sia $s_{i,j}$ la somma dei valori della matrice in un segmento ed $a_{i,j} = (j - i + 1)^2$ l'area all'interno del segmento stesso. Supponiamo che $B = \{b_1, \ldots, b_m\}$ sia un insieme di segmenti coerenti e che s_k ed a_k indichino la somma dei valori e l'area del k-esimo segmento appartenente a B.

D è la densità interna di B:

$$D = \frac{\sum_{k=1}^{m} s_k}{\sum_{k=1}^{m} a_k}$$

All'inizio del processo, l'intera *rank matrix* è considerata un unico segmento. Ad ogni passo l'algoritmo divide uno dei segmenti che appartengono a B e il punto in cui lo divide è un *bound* potenziale che massimizza D. Si tratta di un algoritmo di *clustering* gerarchico, che Reynar sintetizza nei passi seguenti:

L'algoritmo di Reynar.

1. inserisce un *bound* in una certa posizione;
2. calcola la densità totale delle regioni lungo la diagonale, considerando le due nuove regioni individuate da questo *bound* e tutte quelle identificate in precedenza;
3. memorizza il valore di questa densità e la posizione dell'ipotetico *bound*;
4. ripete i passi da 1 a 3 per tutti i possibili *bounds* (quelli che ancora non delimitano segmenti appartenenti a B);
5. sceglie il *bound* per il quale si ha la densità maggiore.

Si ripetono i passi fino a trovare il numero di *bounds* necessario.

Il valore di soglia per fermare la computazione.

Questo numero m, cioè il numero di segmenti da generare è determinato da Choi con un valore di soglia. Sia $D^{(n)}$ la densità interna di n segmenti e $\delta D^{(n)} = D^{(n)} - D^{(n-1)}$ il gradiente. Per un estratto audio di b *bounds* potenziali, dopo b passi del processo di *clustering* si generano i due insiemi di valori: $\{D^{(1)}, \ldots, D^{(b+1)}\}$ e $\{\delta D^{(2)}, \ldots, \delta D^{(b+1)}\}$.

Una riduzione molto marcata del gradiente suggerisce che si è raggiunta la segmentazione ottimale, segnala infatti

il passaggio in una zona in cui la densità aumenta più lentamente (si è entrati in una zona ad alta densità ed è inutile cercare *clusters* più piccoli). Se μ e ν sono la media e la varianza di $\delta D^{(n)}$, con $n \in \{2, \ldots, b+1\}$, il valore m si ottiene applicando la soglia $\mu + c \times \sqrt{\nu}$ a δD [1].

4.1.1 *Ottimizzazione*

Un'altra miglioria introdotta da Choi riguarda il calcolo di s_k , cioè la somma degli elementi della matrice $s_{i,j}$ che fanno parte del k-esimo segmento. Dato che questi sono noti a priori (da quando si è ottenuta la matrice) e costanti, è possibile calcolarne in anticipo i valori. Una procedura efficiente è quella di calcolarli lungo le diagonali, partendo dalla diagonale maggiore e spostandosi verso gli angoli. Nella formula che segue $r_{i,j}$ si riferisce ai valori della matrice (in questo caso quella di *rank*), che si assume abbia dimensioni $n \times n$:

Il calcolo efficiente dei valori di somma.

1. $s_{i,i} = r_{i,i}$
   per $i \in \{1, \ldots, n\}$

2. $s_{i+1,i} = 2r_{i+1,i} + s_{i,i} + si + 1, i + 1$
   $s_{i,i+1} = s_{i+1,1}$
   per $i \in \{1, \ldots, n-1\}$

3. $s_{i+j,i} = 2r_{i+j,i} + s_{i+j-1,i} + s_{i+j,i+1} - s_{i+j-1,i+1}$
   $s_{i,i+j} = s_{i+j,i}$
   per $j \in \{2, \ldots, n-1\}$
   per $i \in \{1, \ldots, n-j\}$

Il metodo ha una complessità di ordine $\Theta(\frac{1}{2}n^2)$.

A proposito di complessità va notato che l'algoritmo cela una piccola insidia nel passo 2: il controllo esaustivo di tutte le segmentazioni possibili ogni volta che si inserisce un nuovo ipotetico *bound* non fornisce una soluzione efficiente[2]. In analogia con il problema della parentesizzazione ottimale di un prodotto matriciale, si può vedere ogni inserimento del *bound* ottimo a dividere un precedente segmento, come la divisione di una sequenza di matrici da moltiplicare nelle sue due sottosequenze ottime (i segmenti audio corrispondono alle singole matrici, la densità da massimizzare al

Un possibile approccio con la programmazione dinamica.

1 Per addolcire il profilo del gradiente si effettua una convoluzione con $[1, 2, 4, 8, 4, 2, 1]$.

2 Si ottiene una equazione di ricorrenza che ha per soluzione la sequenza dei numeri di Catalano.

numero di moltiplicazioni scalari da minimizzare). Si può pensare quindi di calcolare il costo di una soluzione ottima con una modalità *bottom-up* seguendo i principi della programmazione dinamica.

Si può anche osservare che in genere il numero di segmenti è abbastanza elevato, occupa cioè una fascia non molto larga attorno alla diagonale maggiore, quindi una segmentazione che invece di dividere dall'alto la matrice, parta dal basso a fondere i possibili *clusters* si potrebbe fermare prima nella computazione. A parità di complessità nel caso peggiore, si avrebbe una complessità migliore nel caso medio. In effetti un simile tentativo è stato fatto, ma il gradiente scendeva troppo rapidamente e la fusione dei segmenti si fermava troppo presto utilizzando la soglia proposta nell'algoritmo.

4.2 VERIFICA SPERIMENTALE

L'algoritmo di Choi è stato implementato con Matlab® usando gli stessi brani e gli stessi parametri per la DFT della sperimentazione precedente. Per il *ranking* è stata usata una matrice 9×9. L'unico parametro nuovo introdotto è c (vedi 4.1): la sua variazione influisce sul numero di *bounds* finali, quindi può portare ad una sotto-segmentazione, come pure ad una sovra-segmentazione. Dopo alcuni tentativi, ho deciso di assegnare a c questi quattro valori: $\{0,9; 1,2; 1,5; 1,8\}$.

Calcolo dei true positive e dei false positive.

Per cercare di valutare l'efficacia della segmentazione ho utilizzato due parametri spesso citati in questo ambito di ricerca, così definiti:

$$\text{true positive(tp)} = \frac{\text{numero degli onsets esatti}}{\text{numero degli onsets reali}}$$

$$\text{false positive(fp)} = \frac{\text{numero degli onsets sbagliati}}{\text{numero degli onsets rilevati}}$$

dove per *onsets* reali si intendono quelli risultanti dalla segmentazione manuale, mentre gli *onsets* esatti sono quelli che cadono entro un loro intorno di 3 frames (quelli sbagliati cadono fuori) e quelli rilevati sono appunto il totale degli *onsets* rilevati dall'algoritmo.L'intorno è stato scelto per non conteggiare più volte lo stesso *bound*, infatti all'algoritmo è stata aggiunta un'ulteriore soglia finale che cancella i *bounds* più vicini di 3 intervalli temporali (d'altra parte anche quelli manuali rispettano questo vincolo).

Tabella 5: *True positive* e *false positive*: confronto tra *ranking matrix* e *similarity matrix* per la *similarity ratio* (brani monofonici).

coefficiente di soglia	rank. m. t-p(%)	sim. m. t-p (%)	rank. m. f-p (%)	sim. m. f-p (%)
$c = 0,9$	73,2	63,9	77,6	60,0
$c = 1,2$	73,8	61,7	74,1	57,4
$c = 1,5$	71,0	58,4	69,6	56,0
$c = 1,8$	66,7	56,2	67,1	54,5

Tabella 6: *True positive* e *false positive*: confronto tra *ranking matrix* e *similarity matrix* per la similarity ratio (brani polifonici).

coefficiente di soglia	rank. m. t-p(%)	sim. m. t-p (%)	rank. m. f-p (%)	sim. m. f-p (%)
$c = 0,9$	61,9	45,4	65,2	55,3
$c = 1,2$	58,6	41,0	63,2	53,4
$c = 1,5$	54,1	36,6	61,8	51,5
$c = 1,8$	49,5	30,1	58,6	51,3

L'obiettivo di questa sperimentazione era cercare di capire il reale peso del *ranking* e confrontare l'indice di similarità proposto da Choi (il coseno), con quello risultato migliore nella precedente sperimentazione (*similarity ratio*). I risultati non vanno presi quindi per valutare l'algoritmo in assoluto.

4.2.1 *Ranking matrix vs. similarity matrix*

Valutazione dell'efficacia del ranking.

Nelle tabelle 5 e 6 sono riportati i risultati per l'indice *similarity matrix*.

Dai dati si può notare come, a parte il caso dei monofonici con la matrice di similarità, i falsi positivi in genere superino i veri positivi, fatto che ho interpretato come una tendenza dell'algoritmo a sovra-segmentare. Si nota poi come l'ottima performance della matrice di similarità ($c = 0,9$) con i brani monofonici diventi pessima con i brani più impegnativi.

La sensazione che il procedimento di *ranking* sia effetti-

Tabella 7: *True positive* e *false positive*: confronto tra *ranking matrix* e *similarity matrix* per la distanza coseno (brani monofonici).

coefficiente di soglia	rank. m. t-p(%)	sim. m. t-p (%)	rank. m. f-p (%)	sim. m. f-p (%)
$c = 0,9$	65,6	54,7	76,5	51,4
$c = 1,2$	65,6	54,8	75,0	49,1
$c = 1,5$	62,9	54,2	73,0	48,0
$c = 1,8$	61,3	52,0	70,0	46,8

Tabella 8: *True positive* e *false positive*: confronto tra *ranking matrix* e *similarity matrix* per la distanza coseno (brani polifonici).

coefficiente di soglia	rank. m. t-p(%)	sim. m. t-p (%)	rank. m. f-p (%)	sim. m. f-p (%)
$c = 0,9$	63,6	36.2	64,0	48.5
$c = 1,2$	59,1	32.5	60,9	46.7
$c = 1,5$	52,4	29.1	57,6	46.3
$c = 1,8$	48,5	24.7	53,8	47.3

L'importanza del ranking.

vamente utile in questo algoritmo, viene rafforzata dai dati delle tabelle 7 ed 8. In questo caso lo si valuta utilizzando per la segmentazione la distanza coseno: la matrice di similarità illude nei monofonici, con una percentuale di veri positivi maggiore dei falsi, ma nei polifonici le false rilevazioni superano nel caso migliore di un 12% quelle buone, mentre con il *ranking* i due valori rimangono confrontabili (almeno con i c più bassi).

Quest'evidenza sembra contraddire quanto stabilito in 3.3, dove il *ranking* sembra peggiorare la situazione. In realtà bisogna considerare che in questo metodo per la segmentazione non è importante tanto la similarità di un *frame* con il precedente o il successivo, utile per un approccio che consideri lo *spectral-flux*, perchè la matrice contiene un'informazione globale, che bisogna cercare con un buon metodo di *clustering*.

Tabella 9: *True positive* e *false positive*: confronto tra distanza coseno e *similarity ratio* (brani polifonici).

coefficiente di soglia	cosine t-p(%)	sim. rat. t-p (%)	cosine f-p (%)	sim. rat. f-p (%)
$c = 0,9$	63,6	61,9	64,0	65,2
$c = 1,2$	59,1	58,6	60,9	63,2
$c = 1,5$	52,4	54,1	57,6	61,8
$c = 1,8$	48,5	49,5	53,8	58,6

4.2.2 *Distanza coseno vs. similarity ratio*

A questo punto, già osservando le precedenti tabelle si può un confronto tra i due indici di similarità, ma per comodità ho riunito i dati nella tabella 9. La distanza coseno si comporta meglio della *similarity ratio* e vale la stessa considerazione di prima: per rilevare *onsets* con i metodi tradizionali è meglio la seconda, ma utilizzando questa forma di *clustering* sembra essere preferibile quella proposta dall'autore.

CONCLUSIONI

Se torturi i numeri abbastanza a lungo,
confesseranno qualsiasi cosa.

— Greg Easterbrook (1953-)
in Darrell Huff, Come mentire con le statistiche, pag. 181

In questa tesi si è affrontato il problema della segmentazione audio utilizzando un metodo basato sulla costruzione di una matrice di similarità a partire dai *frames* di un brano musicale. Quindi, invece di ridurre il problema alla ricerca dei picchi di una funzione, tipica dei metodi tradizionali, si è fatto ricorso ad una semplice tecnica di *clustering*.

Nella prima parte si sono provati diversi indici di similarità, cercando di testarli in modo indipendente dall'algoritmo proposto, in modo da verificare se potessero essere utili alla segmentazione audio in generale. Un risultato promettente è sembrato offrirlo l'indice *similarity ratio*, una sorta di distanza coseno. Ma utilizzato nell'algoritmo mutuato da Choi, non ha mostrato le stesse buone performance: il motivo potrebbe essere nel tipo globale di informazione contenuto nella matrice di similarità rispetto alle semplici *detection functions* usate nei metodi tradizionali. E' emersa poi l'importanza del procedimento di *ranking* nell'economia di questo algoritmo, che a prima vista non sembrava fondamentale. Un problema emerso è la tendenza alla sovrasegmentazione, testimoniata dall'alta percentuale di falsi rilevamenti di *onsets*, anche se la loro natura è da valutare, in quanto se si trattasse di casi in cui la stessa nota è divisa in più punti, il modello di *audio-matching* per cui si utilizza questo algoritmo potrebbe risentirne meno.

Per migliorare l'efficacia di quest'algoritmo si possono provare nuove *features* audio (anche se a questo livello di segmentazione le alternative sono poche...) e nuove distanze, ma il primo tentativo da fare sarebbe senz'altro un diverso metodo di *clustering*.

BIBLIOGRAFIA

[1] J.J. Aucouturier. Segmentation of Musical Signals and Applications to the Analysis of Musical Structure. Master's thesis, King's College, University of London, 2001. (Citato a pagina 9.)

[2] J.P. Bello, L. Daudet, S Abdallah, C. Duxbury, M. Davies, and M.B. Sandler. A Tutorial on Onset Detection in Music Signals. *IEEE Trans. on Speech and Audio Processing*, 13(5), Settembre 2005. (Citato alle pagine viii, 10, 11 e 12.)

[3] F.Y.Y. Choi. Advances in Domain Independent Linear Text Segmentation. In *Proceedings of NAACL*, Seattle USA, Aprile 2000. (Citato alle pagine viii, 6, 15, 17, 21, 26, 33, 34 e 35.)

[4] S. Dixon. Onset Detection Revised. In 9th *Int. Conference on Digital Audio Effects (DAFx-06)*, Settembre 2006. (Citato a pagina 14.)

[5] J. S. Downie. Music Information Retrieval. *Annual Review of Information Science and Technology*, pages 295–340, 2003. (Citato a pagina 2.)

[6] J. Foote. Visualizing Music and Audio using Self-Similarity. In *Proceedings of ACM Multimedia*, pages 77–80, Orlando FL, 1999. (Citato a pagina 15.)

[7] J. Foote. Automatic Audio Segmentation using a Measure of Audio Novelty. In *Proc. IEEE Int. Conf. Multimedia and Expo (ICME2000)*, volume 1, New York, Luglio 2000. (Citato a pagina 15.)

[8] Chen Gang, Tan Hui, and Chen Xin-meng. Audio Segmentation via the Similarity Measure of Audio Feature Vectors. *Wuhan University Journal of Natural Sciences*, 10(5):833–837, 2005. (Citato a pagina 18.)

[9] A. Klapuri and M. Davy. *Signal processing methods for music transcription*. Springer, New York, 2006. (Citato a pagina 2.)

[10] G. Mazzola. *The topos of music*. Birkauser, 2002. (Citato a pagina 2.)

[11] R. Miotto. A Methodology For The Segmentation And The Recognition Of Digital Music. Master's thesis, Università di Padova, a.a. 2006-07. (Citato alle pagine 5, 6, 7 e 21.)

[12] B. C. J. Moore. *An Introduction to the Psychology of Hearing*. New York: Academic, 1997. (Citato a pagina 10.)

[13] N. Orio. Music Retrieval: A tutorial and review. *Foundation and Trends in Information Retrieval*, pages 1–90, 2006. (Citato a pagina 2.)

[14] B. Pardo. Music Information Retrieval. *Special Issue, Comm. ACM*, pages 28–58, 2006. (Citato a pagina 2.)

[15] J. C. Reynar. *Topic Segmentations: Algorithms and Applications*. PhD thesis, University of Pennsylvania, 1998. (Citato a pagina 34.)

[16] E. D. Scheirer. *Music-Listenings Systems*. PhD thesis, MIT, 2000. (Citato a pagina 2.)

COLOPHON

Questa tesi è stata realizzata con LaTeX 2ε usando lo stile Classic Thesis, di André Miede, ispirato all'opera di Robert Bringhurst *Gli elementi dello stile tipografico* e con il computer portatile prestatomi da Lucrezia, a cui va il mio sincero ringraziamento.