Searching and Classifying Affinities in a Web Music Collection

Nicola Orio
Department of Cultural Heritage
University of Padua

Introduction

- Duplicate detection plays an important role in management of large collections
  - Text
    - Similar (or copied) web pages
    - Patents
  - Multimedia
    - Copyright issues
    - Double acquisitions

Fingerprinting

- For multimedia documents a typical approach is based on **fingerprinting**

- A fingerprint
  - Is much **smaller** than the original document
  - Subsumes relevant **perceptual** features
  - Is **robust** to typical modifications
    - Lossy compression
    - Added signals (noise, mix, edits)

Approaches in Music

- **Acoustic fingerprint**

  - Exploited as a service
    - Shazam!
    - Gracenote
    - MusicBrainz

  - Used for internal management of online collections
    - Spotify
    - Last.fm

Affinities in a Music DL

- We define as **affinities** the following cases
  - **Exact** duplicates
    - Same recordings, inaudible compression
  - **Near** duplicates
    - Audible lossy compression, remasters, alternate takes
  - **Far** duplicates
    - Mashups, montages, loops

Applications

- Saving storage space
- Metadata cleanup
- Interactive metadata creation
  - Especially for crowdsourcing
- Improvement of search engines
- Aid for musicologists
- Tool for collaborative compositions

Methodology

- Two-step approach

  - First Step: **Pruning** Candidate Affinities
    - Compute a coarse index of similarity
    - Do not take account temporal features

  - Second Step: **Pairwise Match** between Affinities
    - Find the best temporal match
    - Align the potential affinities
    - Provide a graphical representation

First Step

- A track is represented by a set of its fingerprints

$$s^k = \{h_1^k, \cdots, h_D^k\} \text{ with } h_i^k \neq h_j^k \ \forall \neq j$$

- the affinity between two tracks can be approximated by

$$af(t^h, t^k) = \frac{\|s^h \cap s^k\|}{min(\|s^h\|, \|s^k\|)}$$

- that is computed efficiently from fingerprint indexes

Second Step

- A track is represented by a sequence of fingerprints

$$l^k = \left(h_1^k, h_2^k, \cdots, h_L^k\right) \quad \text{with} \quad h \in \mathbb{N}$$

- the affinity and the alignment are computed with

$$m_j(l^1, l^2) = \max_i \sum_{k=1}^{N} d\left(h_{i+k}^1, h_{j+k}^2\right)$$

$$p_j(l^1, l^2) = \arg\max_i \sum_{k=1}^{N} d\left(h_{i+k}^1, h_{j+k}^2\right)$$

The Test Collection

- Collection of commercial MP3s
  - 350,000 audio tracks
  - 20,000 hours of music
  - Effort more than 20 years long
  - Experts inserted metadata from CD covers
  - Fingerprint already available
  - Indexes routinely used to track TV broadcasts

Results: First Step

- Fingerprint overlaps after first step

Overlap	# song pairs	% song pairs
Complete (*af*=1)	1057	0.3%
High (*af*>0.9)	104	0.03%
Partial (*af*>0.5)	712	0.2%
Low (*af*>0.25)	2098	0.6%
Minimal (*af*>0.1)	1041	0.3%
Total	5012	1.43%

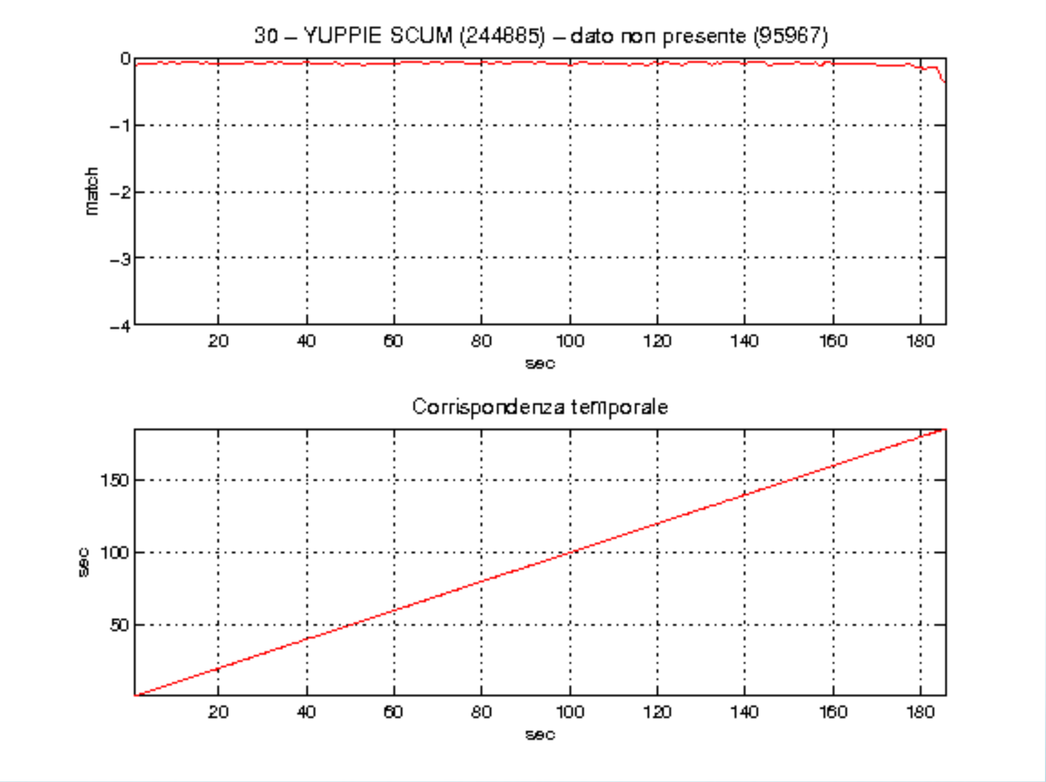
- thresholds have been selected by personnel of the music DL

Results: Second Step

- The goal was to highlight the characteristics of different kind of affinities
  - Graphical tool for the personnel of music DL to:
    - Quickly browse the relevant portions of the audio
    - Possibly identify the kind of affinity without having to listen to the two songs
    - Prioritize the kind of intervention

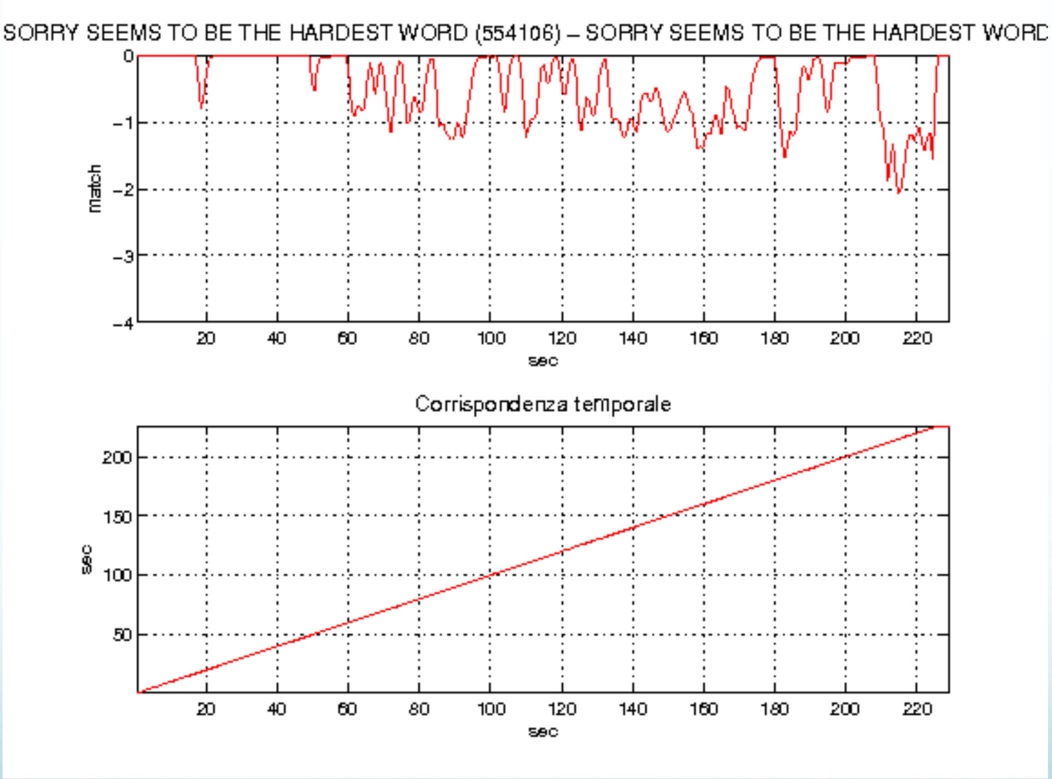
Exact Duplicates

- Different compression algorithms



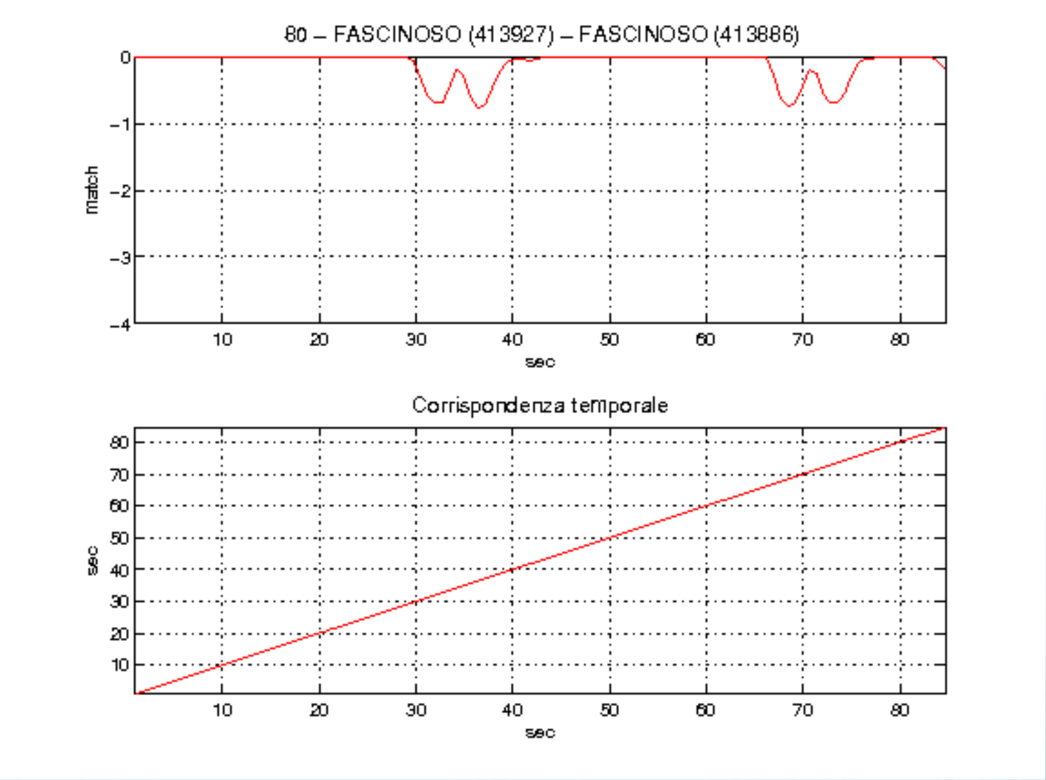
Near Duplicates – 1

- Remasters



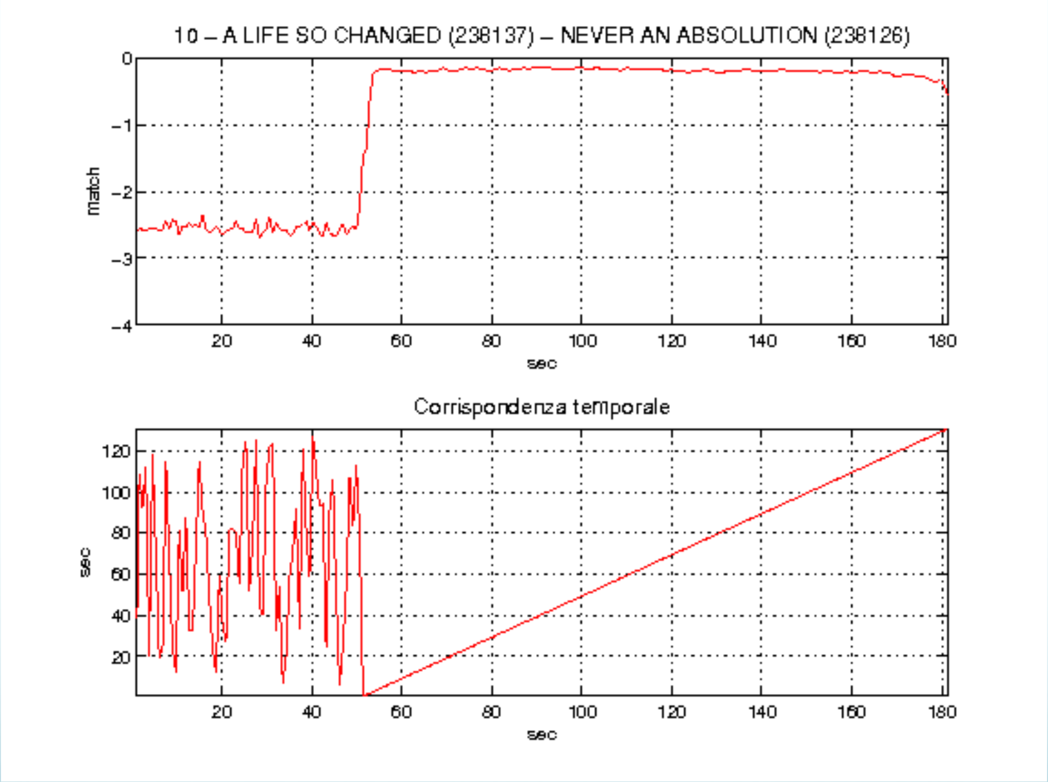
Near Duplicates – 2

- Alternate takes



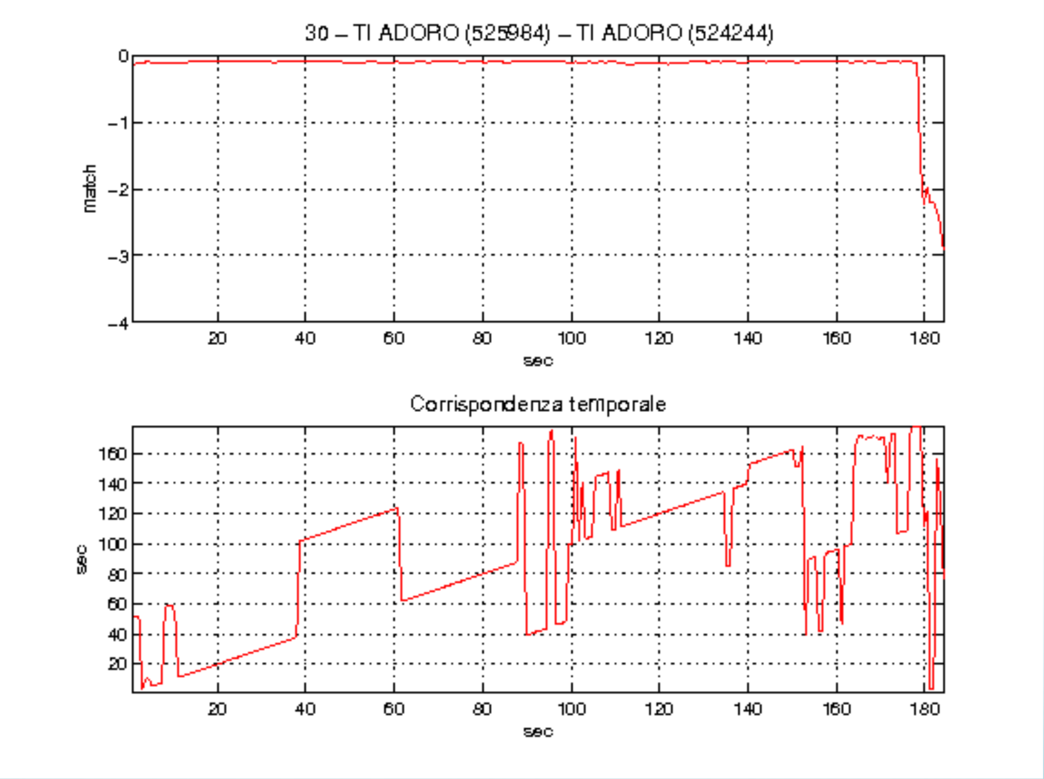
Far Duplicates – 1

- Mashups



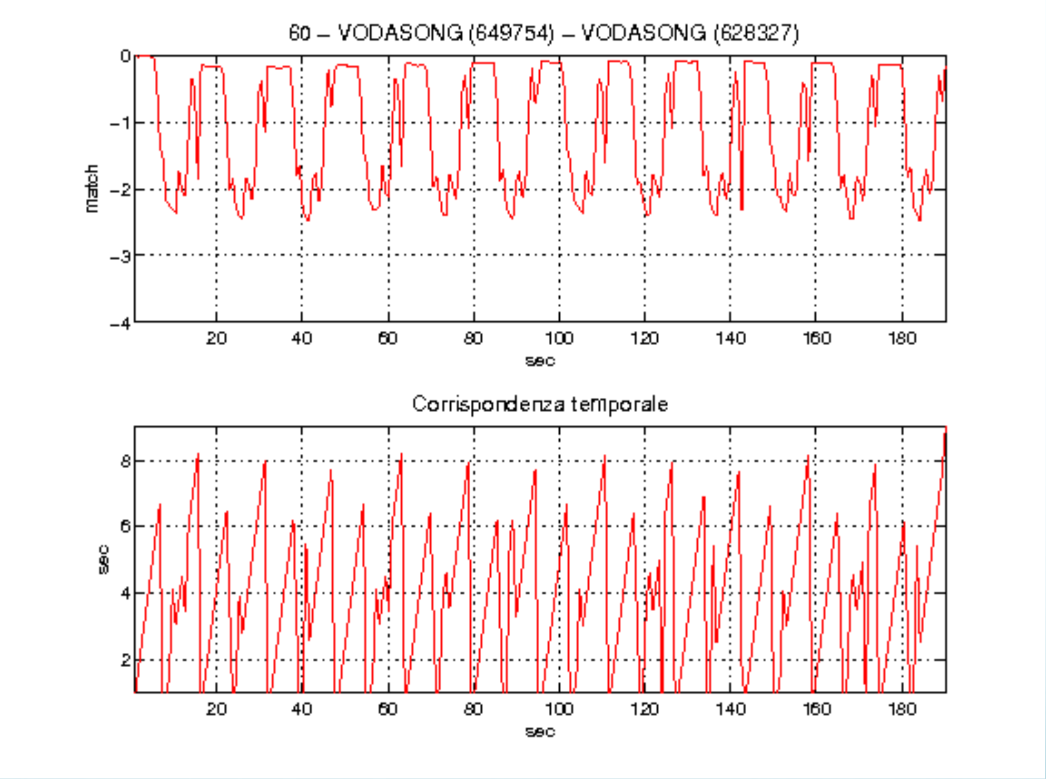
Far Duplicates – 2

- Montages



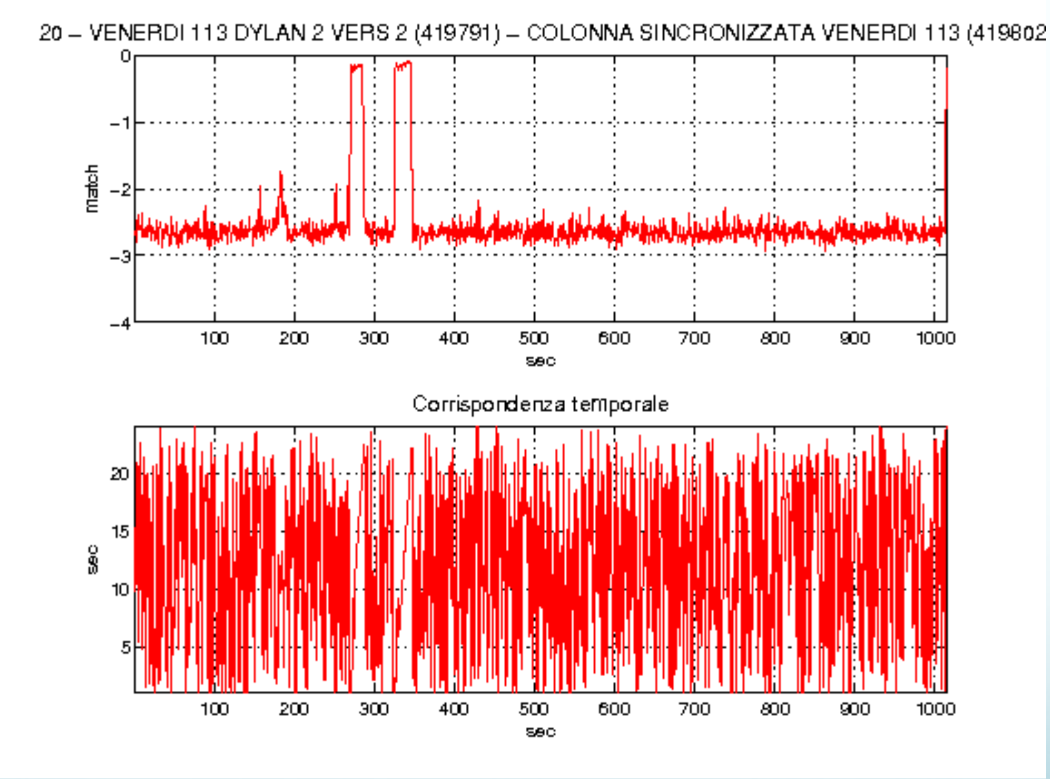
Far Duplicates – 3

- Loops



False positives

- Usage of common sound libraries



Conclusions

- Affinities express tight or loose relationships between audio recordings
  - It is possible to
    - Compute affinities efficiently
    - Represent graphically the kind of affinity
- Future work
  - Analyze the graph trends to extract relevant features for automatic classification
  - Extensive tests also below the 10% threshold

Thanks!