Interoperability and Data Enrichment of Music Sources

Dimitrios Bountouridis
3570584
Utrecht University
2012

July 25, 2013

Contents

Abstract

This thesis addresses two separate but fundamentally linked topics: interoperability and data enrichment. Interoperability has become a popular concept in the recent years and many systems have aimed at achieving it. However its definition remains vague, since it has been highly dependent on the context of use or the needs of each particular framework. To make things worse, various levels of interoperability between two systems ranging from no interoperability to full interoperability exist. This document's primal goal is to present the different facets of this confusing notion, while focusing on the overlapping concepts across the numerous definitions. In addition, "search" interoperability and record-linkage will be further investigated, since they are highly related to our major topic of interest.

Data enrichment, similar to the previous topic, has a vague definition. Adding data to existing data, as a procedure can vary from visiting the library to developing sophisticated, web harvesting systems. Once again the technical details of data enrichment depend on the context of use. Based on the previous, we have decided for this thesis to limit its focus to data enrichment systems, for music sources, that employ web-mining techniques (Web MIR). We particularly focus on databases containing old, rare and obscure music artists from the first half of the previous century.

Both topics will be particularly investigated inside the context of a real-life scenario; namely the case of interconnecting and enriching the music databases of the Netherlands' Institute of Sound and Vision and Meerten's Institute.

Chapter 1

Introduction

The extreme growth of digital music collections leads to the crucial problem of managing and exploiting their content. Information about the artist, album name and song titles solely has proven unsatisfactory for commercial distributors or even users to exploit. The first require more data in order to provide the correct music to the targeted people, while the latter for generating music playlists, grouping music pieces etc.

Typically such data would be gathered using content-based approaches. In such cases the audio files are processed, analysed and all the resulting high-level information is derived from a set of extracted features such as chroma, MFCC etc.

In the possible case of non-existent audio, typical MIR techniques are futile. In such cases, the available metadata should be employed for further data enrichment. Web based metadata hubs for music entities, such as Last.fm and MusicBrainz, alleviate the problem by offering vast amount of community and editorial metadata. However, old and obscure artists are typically misrepresented and therefore music hubs are rendered also futile.

In the recent years, a novel Music Information Retrieval (MIR) field has emerged. Instead of analysing the audio content, Web-MIR techniques employ the Web to extract the required metadata. By assuming the validity of the "wisdom of the crowd", the Web acts as a vast pool of useful data waiting to be mined. A literature overview is presented in Chapter 3.

Therefore, this thesis investigates the ways that Web-MIR can be employed for data enrichment, especially for pre-1950's or obscure artists (Chapters 5, 6, 7 and 8). We are especially interested in three features related to artists: era of productivity, geographic location and genre. Our experiments show that all of them can be extracted efficiently and reliably using Web-MIR techniques.

However, the higher level concept of artist similarity cannot be computed using the previous methods out of the box. We shall later see that Web-MIR is based on a set of tools that simply cannot work for old and obscure artists, since the way these are documented on the web differs from the normal. Therefore, Chapter 8 presents our approach for computing artist similarity by employing three distinct features, supported by musicological knowledge.

The lack of metadata, for large music collections, is only one of the problems that arise with the growth of digital music. This extreme growth lead also to the need for collaboration and exchange of information between distributed collec-

tions. However, the heterogeneities between data representation and meaning can act as a major drawback for achieving the goal of "interoperability", and therefore surpassing them is crucial. Chapter 3 offers an in depth introduction to the world of interoperability while focusing on music applications. Chapter 4 additionally investigates the sub-problem of record linkage on a real life problem, namely the database of the Netherland's institute for Sound and Vision, as part of the COGITCH project.

1.1 COGITCH

COGITCH's (COGnitive ITCH) general objective is to develop generic techniques to index distributed music sources by developing an interoperable system. The specific case studied in order to reach this general objective involves two Cultural Heritage institutions, Meertens Institute (MI) and the Netherlands Institute for Sound and Vision (S&V). MI and S&V possess two unique collections of Dutch musical heritage that must be made interoperable and accessible, both at the level of metadata and at the level of musical content. For the latter, the phenomenon of the musical "hook" in particular is employed.

Therefore, one of the main goals of the COGITCH project is the development of an audio retrieval framework based on "hooks". Although the definition of "hook" is still abstract the underlying idea remains same: music tracks will be indexed, compared and retrieved based on some feature representation. As a consequence, a form of similarity will be established between each pair of songs in the COGITCH dataset . This service by itself would simply provide a ranked list of audio tracks for each query.

However, the questions that this project should be able to answer are more sophisticated: how did Dutch music evolve from the beginning of the 20th century until the current date? Which phrases/motives or eventually hooks managed to stand the test time? How did the early folk melodies affect the current pop repertoire? How did certain hooks transcend to other countries or even genres? Most of these questions contain the concept of time, genre and others. However, both MI and S&V datasets do not include metadata such as year of release; hence the need for a data enrichment is obvious. Only if the song relationships (encoded as hook similarities) are placed inside a meaningful semantic context, will the COGITCH project manage to answer its research questions.

Chapter 2

Interoperability

2.1 Defining interoperability

As previously mentioned, interoperability lacks universal definition. While it originally emerged from IT systems it quickly spread across different scientific fields (eg. communications, medical industry, public safety etc.). For each of those fields, a different denotion, to what inter-operable system means, was given. However, common ground and fundamental components exist in all definitions, and they are listed below:

1. Ability to exchange information that is well undestood from all parties.
2. Ability to use that information.

Given those two components, more restricting and contex-dependant definitions can be generated. We will later see how this applies for data repositories and web-services, fields related to our work. For now, it is worth discussing the fundamental, theoretic levels of interoperability.

2.1.1 Models for Levels of Interoperability

Unfortunately, there is not a single model of levels that satisfies all needs. LISI (Levels of Information Systems Interoperability)[10] distinguishes between:

1. Isolated Systems: No physical connection exists.
2. Connected Systems: Homogeneous product exchange is possible.
3. Distributed Systems: Heterogeneous product exchange is possible.
4. Integrated Systems: Shared applications and shared data.
5. Universal Systems: Enterprise wide shared systems.

The LCIM (Levels of Conceptual Interoperability Model) [1, 11] aims at going beyond the technical reference models for interoperable solutions to the domain of conceptual modelling and simulations. The whole model is based on the works of [12] in composability, meaning the ability of a system to provide

recombinant components that can be selected and assembled in various combinations to satisfy specific user requirements. The levels are presented in a brief below:

1. Level 0: Stand-alone systems. No interoperability.

2. Level 1: Technical Interoperability assumes the existence a communication protocol for exchanging data between participating parties. A communication infrastructure is established allowing systems to exchange the lowest level of information encoding such as bits and bytes. In addition and the underlying networks and protocols are defined unambiguously.

3. Level 2: Syntactic Interoperability introduces an underlying, shared structure to exchange information, such as a common data format. On this level, a common structure protocol of data is used; in addition the format of the information exchange is unambiguously defined.

4. Level 3: The level of Semantic Interoperability is reached when a common information exchange reference model is used. On this level, not only the structure but the meaning of the data is shared; hence the content of any information exchange requests/responses are unambiguously defined. It should be noted that there is a related but slightly different interpretation of semantic interoperability notion, which roughly corresponds to what is later denoted as Conceptual Interoperability, i.e. information in a form whose meaning is independent of the application generating or using it.

5. Level 4: Pragmatic Interoperability is achieved when the participating systems are aware of the methods and procedures that each system is employing. In other words, the use of the data, or the context of its application, is understood by the participating systems: the context in which the information is exchanged is unambiguously defined.

6. Level 5: The state of dynamics systems may change over time, and this includes the assumptions, constraints and the general context that affect data exchange. Systems with Dynamic Interoperability are able to comprehend the state changes that occur in the assumptions and constraints that each is making over time, while also employ those changes. When it comes to effects of operations, this becomes very important; the effect of the information exchange within the participating systems is unambiguously defined.

7. Level 6: When the conceptual model, meaning the assumptions and constraints of an abstraction of reality, are aligned, then the highest level of interoperability is reached, namely "conceptual interoperability". Conceptual models should be documented by engineering methods that would enable their interpretation and evaluation by other engineers. In other words, fully specified but independent models are required.

In this report, LCIM will be employed, although a rough mapping to the notions of LISI is possible. In addition, for our topic of interest, two are the most important and popular levels corresponding to LCIM: *syntactic* and *semantic*. We will now discuss them further.

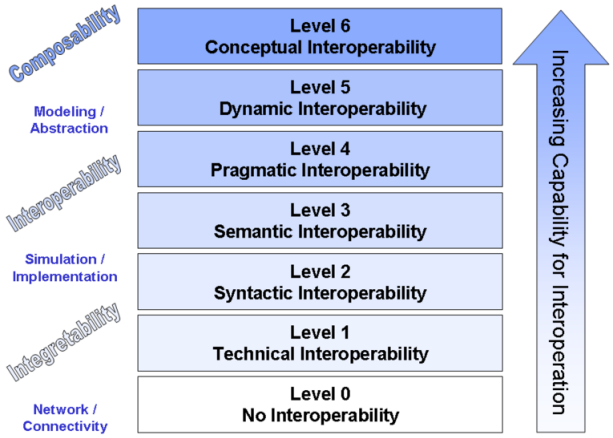

Figure 2.1: The LCIM model

2.1.2 Syntactic Interoperability

In the traditional sense, syntax is defined as the "arrangement of words (in their appropriate forms) by which their connection and relation in a sentence are shown" or "the department of grammar which deals with the established usages of grammatical construction and the rules deduced therefrom" (Oxford English Dictionary). Hence, with syntax in the traditional sense, the challenges of Syntactic interoperability become: a) identifying all the elements in various systems; b) establishing rules for structuring these elements; b) mapping, bridging, creating crosswalks between equivalent elements using schemas etc. c) agreeing on equivalent rules to bridge different cataloguing and registry systems [3].

These challenges, as described by Veltman, can be simplified into the following: if two or more systems are capable of communicating and exchanging data they are exhibiting syntactic interoperability [3]. This simply implies that particular data formats or even protocols should be employed by all participating parties. Common tools for syntactic interoperability are SQL and XML, ensuring that each data source is represented with the same format.

2.1.3 Semantic Interoperability

Semantics is defined as the meanings of terms and expressions. Hence semantic interoperability is "the ability of information systems to exchange information on the basis of shared, pre-established and negotiated meanings of terms and expressions" [3]. To simplify this, semantic interoperability assumes that the content of the information exchanged is unambiguously defined and that all parties interpret it the same. It becomes obvious that syntactic is a prerequisite for semantic interoperability. If the representation of the data is ambiguous then its semantic interpretation and its placing into context becomes impossible.

Now let us assume that the prerequisites are satisfied. Even so, ambiguity would still be an issue since definitions across data sources may differ. In order

to overcome this problem of semantic heterogeneity [5] and achieve an acceptable level of precision and specificity, interoperable systems usually employ a shared, independent vocabulary of concepts to which each data field is linked. The vocabulary in addition to a related ontology describing meanings and First-Order Logic relations between data, provide the fundamental tools for correct semantic interpretation [4].

Currently there's no single ontology that can represent all possible data fields and terms for apparent reasons. As an alternative, foundation ontologies have been suggested. Foundation, or upper ontologies, comprise of quantum elements on which others can be user-generated. However, this topic is still researched and no foundation ontologies have been widely accepted or standardized. As a result, it is quite common for only partial-semantic interoperability to be achieved, meaning that only part of the whole set of definitions are interconnected.

Regarding software application communication, there are at least two different ontology-based types of solutions:

- In the first type of solutions all applications share a common, communication terminology. The semantics of this shared terminology are typically specified by a (meta) standard and all applications employ it in an unambiguous fashion. In the case of an application that internally uses a terminology digressing from the standard, a transformation mapping needs to be established. This is usually achieved via human intervention and specially developed software that perform the transformation.

- In the second case the system's semantics is specified by logic-based ontologies. In order to resolve this, a broader terminology, whose semantics is also specified by a logic-based ontology, is used as a medium or reference terminology. Computer programs can automatically generate transformations between terminology systems. This is possible due the fact that relationships between terminologies can be mapped into or from the reference. As we shall see in later chapters, the computer-based transformations are the core of the Semantic Web.

It becomes quite obvious that the issue of semantic interoperability is multifaceted. Its theoretical aspects and intuition are even rooted to ancient philosophers such as Plato and Aristotle. Therefore, we will end our discussion here, forwarding the reader to the work of [3] for further details. It should be mentioned though that semantic interoperability governs the rest of this thesis, since its importance and practical significance are great.

2.2 Interoperability Into Context

In the previous section we described the different levels or types of interoperability outside any practical context. Therefore in this section, we will focus on its application on certain tasks and frameworks.

- **Software:** Interoperability in software corresponds to the ability of programs to exchange data via a set of data formats, protocols etc. For example two communicating programs running on different platforms (eg. Java, C++) show software interoperability. It is worth refering to the ISO/IEC 2382-01, Information Technology Vocabulary definition which

describes software interoperability as the "the capability of communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units".

- **Medical industry:** Interoperability in medical devices corresponds to the ability of of those devices to communicate and be compatible with others. In simpler words any new device purchased by a medical center, should be able to work with any others already existing.
- **Public safety:** Interoperability in public safety has become a prominent issue (at least in the USA) mainly after 9/11 and Hurricane Katrina, where safety departments (fire fighting, law enforcement etc) presented lack of communication in terms of data representation and hardware.
- **Military:** Ongoing military operations throughout the world have demonstrated an increased reliance on the collaboration of many nations and their corresponding services, disparate systems and procedures. These systems (usually complex), were typically acquired in isolation without considerating interoperability. It wasn't until recently that some nations begun to expend resources on researching the major issue of incompatibility within the battlespace.

2.2.1 Search Interoperability vs Multiple Database Querying vs Enterprise Search

As mentioned in the abstract, the subject of interest of this report is *search interoperability*, referring to the ability of searching and gathering information from multiple, content-wise similar sources using (optimally) a single query. Such sources encode and store semantically similar notions eg. music records, employee profiles etc.

The notion of *multiple database querying* should be separated from the previous, since it refers to combining distributed sources in order to complete a single query. For example, a table containing employees' information may be located in database A and a table with salary information in database B. Multiple database quering aims at combining the two separated sources in order to complete an employee-salary query.

Enterprise search is a widely used concept that needs to be separated from interoperability also. The latter can be considered as a desired or even required characteristic of an enterprise search framework, while its definition is far less solid. Enterprise search actually corresponds to searching, indexing and integrating documents from various sources such as file systems, intranets, databases, the Web and others. The results of a search are usually available within the enterprise but this not a canon. Typical procedures employed by such systems are content processing and ingestion, query parsing, federated and faceted search, and others. In order to be complete, we will later discuss a published approach related to enterprise search [29]. For the time being the reader is forwarded to Hawking's research [28] which nicely summarizes the definition, challenges, problems and solutions of enterprise search.

Returning to search interoperabity, it should be mentioned that during the following paragraphs we will make the assumption that databases, web sources

and everything indexable roughly correspond to the same concept. In other words we assume that each piece of information/entity online has a database representation (set of fields and values). We also assume that the search interoperability problem can be generalized to web services[8], since such services use web sources as input/output.

Problem Definition

Typically, what are called "enterprise" systems, are developed over several periods of time, by diverse organizations and not necessarily with the same metadata schemas, fields and vocabularies. This leads to substantial heterogeneity in syntax, structure and semantics when it comes to interoperation and cross-collection searching between these systems [8]. Figure 2.2 presents some typical heterogeneities.

Schema and field disagreements are sometimes referred to as "metadata schism". The schism suggests that that there is no single metadata scheme and set of fields that satisfies the needs of all applications. Although metadata standardizations exist, this issue remains unsurpassed. On the other hand, generating new schemes for each new framework or application is rather unwise since it propagates the problem.

Given that and if we digress a bit from the "search" problem into the "web search", we will stumble upon the lack of semantic context. Despite the popularity of the standard Web technologies (HTML, XML) it has been obvious they lack expressive power that would enable related entities (e.g. music artists), described in various documents, to be connected. Hyper-links encode relationships between documents, but fail to capture semantic content. In other words, it is very common for the documents to be connected but not the data itself. Taking into consideration the lack of formal specifications and concepts that attach a "meaning" in a given domain, it becomes obvious that Web is currently a source of distributed, heterogeneous data that users struggle to navigate into.

Studying the issues above draws a blueprint of the interoperability levels with respect to distributed sources. Zeng and Chan [14] have clustered interoperability efforts using a practical approach:

- **Schema level**: Efforts are focused on the elements of the schemas, being independent of any applications. The results usually appear as derived element sets or encoded schemas, crosswalks, application profiles, and element registries (all will be discussed later).

- **Record level**: Efforts are intended to integrate the metadata records through the mapping of the elements according to the semantic meanings of these elements. Common results include converted records and new records resulting from combining values of existing records.

- **Repository level**: With harvested or integrated records from varying sources, efforts at this level focus on mapping value strings associated with particular elements (e.g., terms associated with subject or format elements). The results enable cross-collection searching.

Heterogeneities/Conflicts	Examples	
Naming conflicts Two attributes that are semantically alike might have different names (synonyms) Two attributes that are semantically unrelated might have the same names (homonyms)	**Source 1** Student(#id, Name) **Source 1** Student(#id, Name)	**Source 2** Student(SSN, Name) **Source 2** Book(#id, Name)
Data representation conflicts Two attributes that are semantically similar might have different types of representations	**Source 1** Student(#id, Name) #id defined as 4 digit number	**Source 2** Student(#id, Name) #id defined as a9 digit number
Naming conflicts Semantically alike entities might have different names (synonyms). Semantically unrelated entities might have the same name.	**Source 1** EMPLOYEE(#id, Name) **Source 1** TICKET(TicketNo, Movie Name)	**Source 2** WORKER(#id, Name) **Source 2** TICKET(FlightNo, Airport)
Schema isomorphism conflicts Semantically similar entities might have different number of attributes.	**Source 1** PERSON(Name, Address)	**Source 2** PERSON(Name, Address, Phone)
Generalization conflicts Semantically similar entities are represented at levels of generalization.	**Source 1** GRAD-STUDENT(ID, Name, Major)	**Source 2** STUDENT(ID, Name, Major)
Attribute entity conflicts Semantically similar entity modeled as attribute in one service and as an entity in the other.	**Source 1** COURSE(ID, Name, Semester)	**Source 2** DEPT(ID, Course, …, Semester)

Figure 2.2: Heterogeneities

2.3 Achieving Interoperability

2.3.1 Record Level

Semantic Web and the Linking Open Data Project

Linking Open Data (LDO) is a large W3C SWEO[1] project that aims at publishing interlinked datasets on the Web, following a series of Linked Data principles. Technically, links between sources enable the user to navigate from a data item within one document source to associated data items within other sources using a Semantic Web browser (eg. Tabulator [2], Marbles[3]). The aforementioned process is presented in Figure 2.3, where RDFs encode relationships between information described as URI aliases (both notions will be explained later). The use,r by querying for a Band A in a particular database, can navigate to the

[1] www.w3.org/2001/sweo
[2] www.w3.org/2005/ajar/tab
[3] marbles.sourceforge.net/

same band in an another source. From there it is possible to acquire information and relationships not present in the first database, such as the band's location and so on.

It is also possible for links to be followed by the web crawlers, which may provide sophisticated search and query capabilities over crawled data. The functionality and power of such a scheme is enhanced considering that the query results are structured data and not just links to HTML pages therefore can be used by other applications [15]. Currently the number of interlinked datasets is 295, which consist of over 31 billion links.

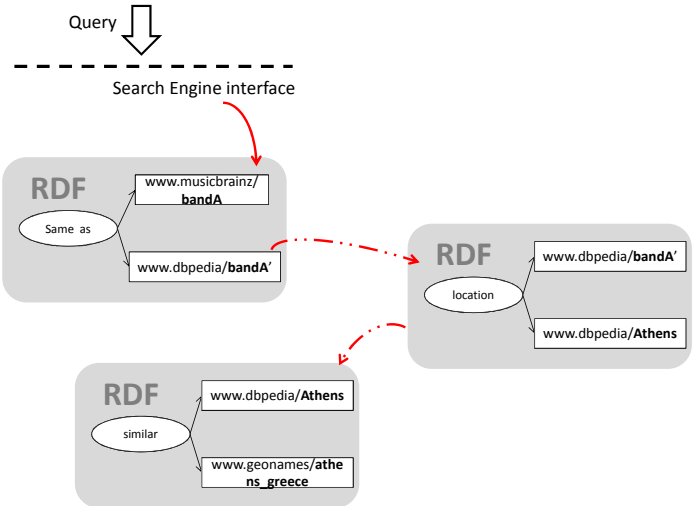

Figure 2.3: Searching through RDF triples

In contrast to HTML and URLs used on the hypertext Web, Linked Data employs the RDF (Resource Description Framework) format and URIs (Uniform Resource Identifiers). In order to continue our discussion, we firstly have to investigate those.

URIs

URLs describe a location/address of a document on the world wide web. URIs on the other hand are more generic and provide an identity to an actual entity. Typically URIs use the *http://* scheme and hence the corresponding entities can be looked up simply by dereferencing the URI over the HTTP protocol. Therefore, the HTTP protocol provides a simple, universal mechanism for retrieving absolute resources:

```
http://example.org/absolute/URI/dog.jpg
```

or retrieving descriptions of entities that cannot themselves be sent across the network in this way:

```
http://en.wikipedia.org/wiki/URI/#dog
```

It should be mentioned that duplicate identities may exist for the same real world concept; for example the URIs *http://data.linkedmdb.org/resource/film/77*

and *http://dbpedia.org/resource/pulp_fiction%28film%29* refer to the same film "Pulp fiction".

Typical URI repositories, or "identifier hubs" as they are usualy denoted, are DBpedia[4] (which extracts sources from Wikipedia pages), Geonames[5] (for geographical information) and MusicBrainz[6][16] (for music related sources eg. songs, albums, artists).

RDF

RDF encodes information using three intuitive attributes: *subject, predicate* and *object*. The subject denotes the resource, while the predicate corresponds to traits or aspects of the resource and expresses a relationship between the subject and the object based on vocabulary of relations, which we have called "ontology" (see figure 2.4). If we imagine that the subject and object correspond to different online resources, we can argue that RDF for datasets is the analogous of hyper-links for documents [18]. A simple example can be very informative:

```
Subject: http://data.linkedmdb.org/resource/film/77
Predicate: http://www.w3.org/2002/07/owl\#sameAs
Object: http://dbpedia.org/resource/Pulp\_Fiction\_\%28film\%29
```

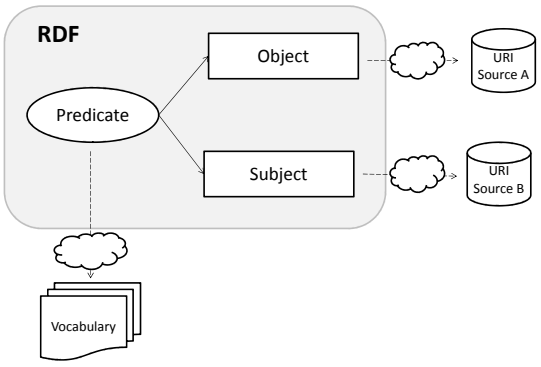

Figure 2.4: The RDF architecture

Once again hubs such as DBpedia, MusicBrainz etc, in addition to URI identifiers, offer millions of RDF triples. Therefore they not only provide information about concepts but also structured data about them. Hence navigation from one source to another is easy and efficient, thus making them the de-facto interlinking hubs.

Vocabularies and Ontologies

During our previous discussion about semantic interoperability (see section 2.3), we mentioned vocabularies and ontologies that provide meaning descriptions, in addition to First Order Logic relations. RDF encodings have a direction relation

[4] dbpedia.org/about
[5] www.geonames.org/ontology
[6] http://musicbrainz.org/

to those notions; for example the previous example's URI-predicate was pointing to an entity of a particular ontology while the subject and object to different vocabularies. It should be noted that the terms"ontology" and "vocabulary" are usually misused and overlapped over the internet, however we shall try separate them. Only "ontologies" describe relations but both can describe meanings.

There are various standardized vocabularies and ontologies that model meaning and relationships over the web, such as:

- RDF Vocabulary Definition Language (RDFS)[7]: a general-purpose language for representing information in the Web.
- Web Ontology Language (OWL)[8]: usually employed to model relationships on the Web, but its applicability can be extended beyond that.
- Friend-of-a-friend (FOAF)[9]: aiming at describing people, the links between them and the things they create and do. Typically employed to model Facebook and MySpace networks.
- Music ontology (MO)[10][8]: aiming at describing main concepts and properties related to music (i.e. artists, albums, tracks, but also performances, arrangements, etc.).
- Semantically-Interlinked Online Communities (SIOC): although not a vocabulary per se, SIOC offers the possibility for shared semantics by combining ontologies such as the example below:

```
:myRadio a mo:Playlist;
   mo:track : song1;
   sioc:has_creator: me;
   sioc:site <http://lastfm.com>;
   dc:tile  'Alex's last.fm playlist';
:song1 a mo:Track;
   dc:title 'Monkey Man;'
   foaf:maker dbpedia:The_Specials.
```

  Here MO, FOAF and Dublin Core are used. The latter will be discussed in later chapters due to its popularity and significance.

- Other vocabularies: GoodRelations for E-Commerce, DOAP for describing open software projects and more.

SPARQL

Given the RDF syntax and vocabularies it is impossible to use conventional querying languages and formats. SPARQL is a query language for data stored in RDF format. A simple example is given below[11].

Data:

[7] http://www.w3.org/TR/rdf-schema/
[8] http://www.w3.org/TR/owl-features/
[9] http://www.foaf-project.org/
[10] http://musicontology.com/
[11] http://www.w3.org/TR/rdf-sparql-query/#basicpatterns

```
{@prefix foaf:  <http://xmlns.com/foaf/0.1/> .
_:a  foaf:name   "Johnny Lee Outlaw" .
_:a  foaf:mbox   <mailto:jlow@example.com> .
_:b  foaf:name   "Peter Goodguy" .
_:b  foaf:mbox   <mailto:peter@example.org> .
_:c  foaf:mbox   <mailto:carol@example.o}
```

Query:

```
{PREFIX foaf:   <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox
WHERE
  { ?x foaf:name ?name .
    ?x foaf:mbox ?mbox }}
```

Query Result:

```
|    name           |             mbox               |
|"Johnny Lee Outlaw"| <mailto:jlow@example.com>      |
|"Peter Goodguy"    |   <mailto:peter@example.org>   |
```

As it becomes obvious SPARQL has many similarities to the typical SQL syntax. However, the reference ontology(s) or vocabulary(s) need to be defined prior to any data storing or querying. SPARQL will not be discussed any further.

2.3.2 Publishing Linked Data on the Web

We now have a clearer understanding of the encoding-information-format employed by the Semantic Web. We are also aware of the increased potential that if offers. However, it remains an issue how to publish our own data; hence the following parts aim to provide a brief guide that would lead through such a process. According to [18] there are three steps involved in the publishing process, if no automated publishing tools are employed:

1. Choose URIs and RDF vocabulary.
2. Generate links.
3. Provide additional metadata in order to increase utility (will not be discussed).

We will discuss each step while focusing on the music-database context.

Choose URIs and RDF vocabulary

As we have previously mentioned the choice of URI repositories (for the Object attribute) and RDF vocabularies/ontologies is highly associated with the context of use. However both can be tailored to the developer's and dataset's needs. For example if we aim to create an online music database, we may choose to link each URI in our set to its corresponding in MusicBrainz, using either the Music Ontology or the OWL. We can also link the band's location to the corresponding URI entities on Geonames. A simple example using both OWL and FOAF is shown below [3]:

```
<http://dbtune.org/jamendo/artist/5>
  foaf:based_near <http://sws.geonames.org/2991627/> ;
  owl:sameAs <http://zitgist.com/music/artist/
        0781a3f3-645c-45d1-a84f-76b4e4decf6d>.
```

Generate Links

Search engines require RDF links in order to efficiently navigate through sources and acquire additional data. Link generation though is a sophisticated procedure, especially when taking into consideration factors such as the size of datasets and possible record ambiguity. Therefore it is common practice to use automated or semi-automated tools.

The most difficult case appears when different naming schemes are employed by both sets. In that case links should be generated based on a distance/similarity metric. This issue has been addressed by many researchers in the general database development area, therefore we will discuss the work of Raimond et al. [3] and Kobilarov et al. [20] which fall into the music context. Before we continue though, it would be wise to formulate the problem of record linkage following Neiling's work [21].

We denote with $(a, b) \in (A \times B)$ an ordered pair of elements of the two populations (datasets) A and B. The crossproduct $A \times B = \{(a, b)|a \in A, B \in B\}$ is the disjoint of two sets:

$$A \times B = M \cup U,\ M \cap U = \emptyset,$$
$$M = \{(a, b) \in A \times B|a = b\},$$
$$U = \{(a, b) \in A \times B|a \neq b\},$$

where M refers to the matched set (common records in A and B) and U to the un-matched set. It is common practice to project both a and b into new data spaces so as to convert the problem of similarity into a simple distance calculation. Hence we introduce two data spaces X, Y and mappings $a : A \to X$ and $b : B \to Y$. Now at the data space X and Y, with $k \in \mathbb{N}$ common dimensions, we define γ the comparison function $\gamma : X \times Y \to R \subset \mathbb{R}^k$. However it becomes obvious that the choice of γ and the decision rule that confirms a matching, are of great importance. We forward the reader to [21] for further investigation of the problem. We will now focus on Raimond's approach, which firstly elaborates on the simplest of methods for achieving linkage.

Naive approach The simplest approach would be to iteratively try to match each record a in A with the ones in B, possibly by employing literal matching (string matching), which is typically supported by databases. Of course the procedure would require to extract literals from the given sources and URIs, but overall this is fairly simple procedure.

However this approach suffices only when the strings themselves provide disambiguation. This is rarely the case, especially if we take into consideration the amount of typos introduced during database creation. Moreover, when it comes to music sources, it is fairly possible for certain song names to belong to more than one artist. Artists with the same or similar name also exist (e.g.. "Iron Maiden" and "The Iron Maidens").

Graph matching We will present Raimond's ideas by employing his simple example, which aims to associate the band "Both" from database A to database B where two instances of "Both" appear (see figure 2.5).

Firstly the two datasets are modelled as graphs, with each edge represented as a triple (s, p, o) corresponding to subject, predicate and object respectively.

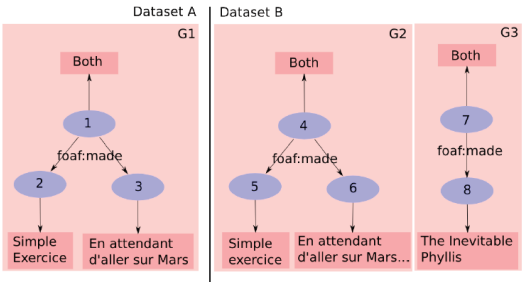

Figure 2.5: Example datasets.

Later initial similarity values between all pairs of resources $(s1, s2)$ and $(o1, o2)$ are computed, such that $(s1, p, o1) \in A$ and $(s2, p, o2) \in B$. Such similarity values might be calculated by a string matching algorithm.

Next, a graph-similarity measure is calculated. In our example, we consider the possible graph mappings as presented in the second column of Figure 2.6. Then, a measure is associated with such mappings: similarity values associated with each pair (x, y) are summed and normalised by the number of pairs in the mapping. In our example, the resulting measures are in the last column of Figure 2.6. Finally, the mapping whose similarity measure is the highest, is chosen, optionally thresholding to avoid making mappings between graphs which are too dissimilar. In our example, $M_{G1:G2}a$ is eventually chosen.

Graphs		Mapping	Measures
G1	G2	$M_{G1:G2}a = \{(1,4),(2,5),(3,6)\}$	0.9
G1	G2	$M_{G1:G2}b = \{(1,4),(2,6),(3,5)\}$	0.4
G1	G3	$M_{G1:G3}a = \{(1,7),(2,8)\}$	0.55
G1	G3	$M_{G1:G3}b = \{(1,7),(3,8)\}$	0.55

Figure 2.6: Possible graph mappings $(x, y) \equiv \{x, owl : sameAs, y\}$, and associated measures.

It should be mentioned that this is an iterative procedure. If the artist names in B were different then the algorithm wouldn't compute the measures associated with the neighbours. However in this example at the first stage (when only "Artist" name is taken into consideration) both sources have a similarity of 1 when compared to A.

BBC approach Kobilarov et al. [20] presented their method for interlinking concepts from various BBC domains (food, music, TV, etc.) using DBpedia as the controlling vocabulary.

It should be firstly noted that BBC used to store data in a CIS format that captured a hierarchy of five terms (proper names, subjects, brands, time periods, places). For example the TV sitcom Marry that appeared in 1985 would be stored as "Mary (1985 sitcom)".

Their algorithm for interlinking CIS data to DBpedia vocabulary consists of two parts: DBpedia look up and Context-based disambiguation. During the

first part the system uses a weighted label lookup and PageRank to identify the more likely to be associated concepts. However, ambiguity could be introduced between retrieved candidate concepts. Therefore, at the second stage the CIS hierarchy is employed to generate clusters (in our particular example "Mary", "1985" and "sitcom"). The algorithm then tries to identify matching DBpedia concepts and employ them for disambiguation.

With the previous example of "Mary (1985 sitcom)" falling into the clusters together with other sitcoms and television shows, it is possible to reject the top ranked label-based result "Mary (Holy Mother)" for the search term "Mary" due to its DBpedia class, reject "Something about Mary" and "Mary Tylor Moore Show", and accept the match dbpedia:Mary_(1985_TV_series) based on the DBpedia category "1980s American television series" (Figure 2.7).

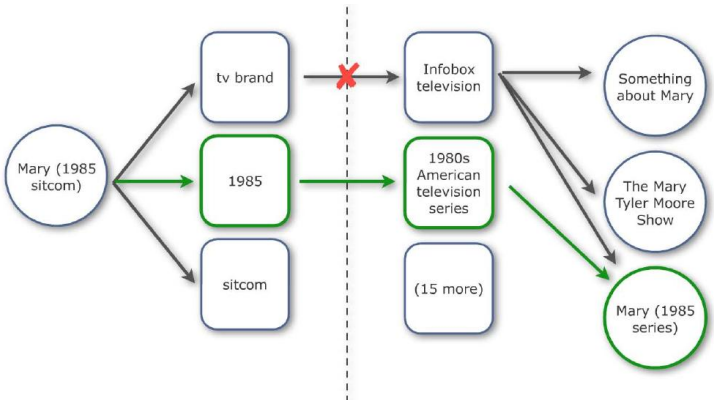

Figure 2.7: Context based disambiguation

To our knowledge, these are the only published approaches for record linkage inside the music database context. However, general record linkage tools have been implemented that can sufficiently interlink music RDF sources, such as SILK [22] and LinQL [23].

2.3.3 Schema Level

Throughout this chapter, it has been stated frequently that no metadata scheme (or "profile") satisfies all needs. Although rather unwise to create new schemes from scratch, the metadata universe has led into metadata schism [7] . There have been many attempts for standardising metadata schemes such as the ISLE Meta Data Initiative (IMDI[12]) and the Open Language Archive Community (OLAC[13]) for describing multi-media/multi-modal language resources. TEI (Text Encoding Initiative) is a successful attempt for text resources. Dublin Core (DC) though is the most popular, at least according to our findings, since it is greatly associated with the OAI-PMH protocol. In addition, various extensions of DC have emerged to cope with specific needs (eg. OLAC). All of these initiatives will be presented in detail later during our "Repository Level" discussion, since their development is mostly associated to cross-repository data

[12]http://www.mpi.nl/IMDI/
[13]http://www.language-archives.org/

harvesting. However, they can find application to simple, schema-level interoperability efforts.

We have already mentioned ontologies that offer relation vocabularies. However, they also aim at addressing this standardized metadata issue, by providing well accepted, domain-specific metadata fields. But specificity and generalized usage are opposite terms, and developers have been fluctuating between both for years. As Broeder et al. [7] state "it started as a move from small sets of descriptors with broad semantics to large sets with highly specific descriptors, which was followed by a backward move". Therefore, even though ontologies exist, the problem remains unsurpassed.

The following paragraphs present typical and popular methods that have been used over the years to address schema-level interoperability.

Application Profiles

One of the simplest ways to achieve interoperability on the schema level is through application profiles (AP). Inside different communities the need of fields provided by certain schema varies. For example, a particular system might require to employ DC's "Creator" but not the "Type" field. With APs, existing metadata schemas are used as the basis for describing a certain group's dataset. Elements from different namespaces can be combined, thus allowing different degrees of depth and detail, while offering optimization for a particular application [13].

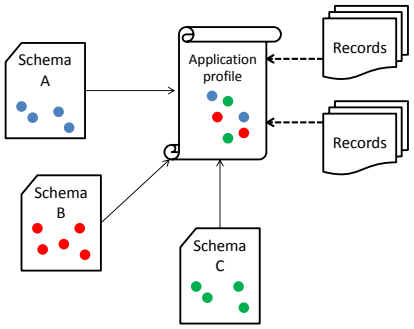

Figure 2.8: A simple illustration of AP generated from three different schemas.

The disadvantage is that APs cannot declare or introduce new elements. Therefore, specific needs may not be met. [17] state the following: "if an implementor wishes to create new elements that do not exist elsewhere (under this model) they must create their own namespace schema, and take responsibility for declaring and maintaining the schema".

Crosswalks

Crosswalk is the procedure of semantically mapping metadata elements from one schema to another. If we want to formalize this procedure a little more, given the sets of metadata fields $A = \{a_1, a_2, ..., a_n\}$ and $B = \{b_1, b_2, ..., b_m\}$, a crosswalk firstly defines a function g such that $g(a_i) = b_j$, and so $g(A) \rightarrow A' \in B$. However, any given query should also be filtered and processed so as

to correspond to the correct fields in both sets, hence a function f is defined such that given a query $Q = \{q_1, q_2, ..., q_k\}$, $f(Q) \rightarrow Q' \in A' \cap B$.

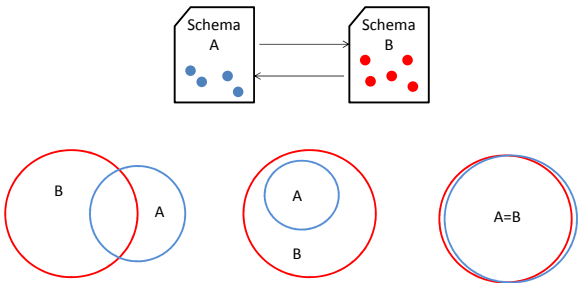

Figure 2.9: Top: An illustration of crosswalk. Bottom: different degrees of element equivalency.

This is by far the most popular approach for interoperability although the disadvantages of such approach are prominent. This one-to-one semantic mapping between metadata schemes suggests that the work overload increases exponentially. Moreover, the overlap of the mapped schemes might not suffice, leaving unmapped elements out of use (see Figure 2.9).

Metadata Registries & ISOcat

Metadata registries aim at gathering metadata schemas. In addition, they offer access, such that the users can search, publish their own metadata while also create APs. Depending on the registry implementation, crosslinking or crosswalking between schemas can be also available. An important functionality of registries is that they assign a unique identifier to various concepts such as elements, schemas, APs etc.

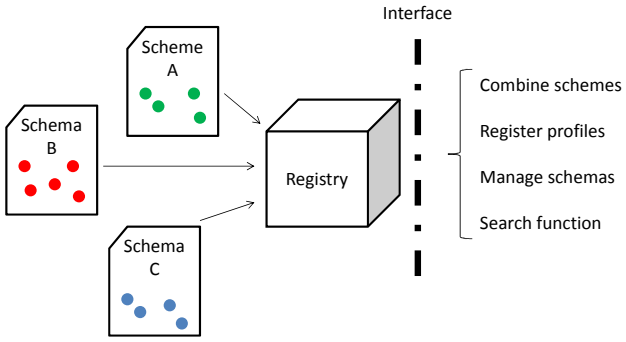

Figure 2.10: An illustration of a metadata registry.

We will later thoroughly discuss Broeder's et al. work on interoperability, which firstly aims at setting a central base of agreed, controlled vocabulary, instead of generating 1-to-1 metadata mappings, via ISOcat.

ISOcat[14] is an implementation of the ISO 12620:2009 standard[15], and is

[14] www.isocat.org

[15] ISO 12620:2009 provides guidelines concerning constraints related to the implementation

used to validate and check the consistency of this standard. Its web interface allows the user to create and combine metadata fields (denoted data categories) into data category registries (DCR), edit, share and export them, while assuring agreement with the ISO standards.

In general each data category field in ISOcat has three attributes (administrative, descriptive, linguistic). The most important, as far as this report is concerned, is the "descriptive", which contains an English name, definition and optional translations. In addition to that, each data category has a unique identifier called PID (persistent identifier), designed to be included in metadata schemas, similar to URIs for records. This is one of the strongest features of ISOcat, offering a common understanding of concepts across databases and therefore communities.

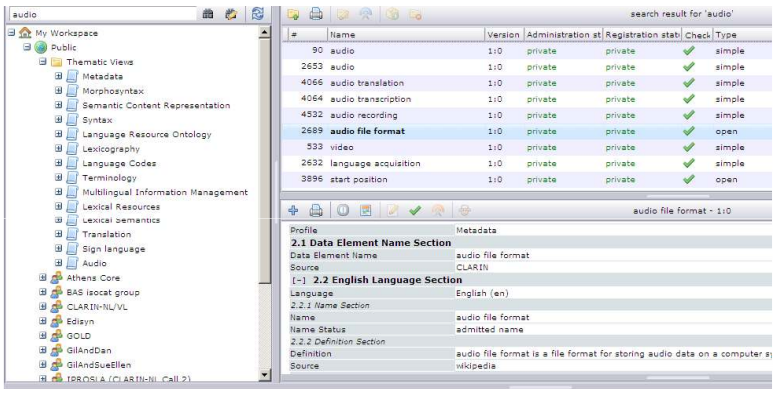

Figure 2.11: ISOcat's interface queried with "audio". From the resulting list the user can pick metadata fields to build his own data category registry.

Federated Search

Federated search falls somewhere between the "Record" and "Repository" levels of interoperability; however, for the sake of convenience we will describe it here. The data flow in a federated search is as follows: the user firstly makes a single query request, which is distributed to the search engines/sources participating in the federation. The federated search then receives the data from the search engines for presentation to the user. The resulting data are aggregated prior to any display thus giving to the user the impression of consistency. This allows a user to search multiple databases at once in real time, arrange the results from the various databases into a useful form and then present them.

Peter Jacso [24] has established a set of criteria that have to be met for a search to be considered federated:

1. Transforming a query and broadcasting it to a group of disparate databases or other web resources, with the appropriate syntax.

of a Data Category Registry (DCR) applicable to all types of language resources, for example, terminological, lexicographical, corpus-based, machine translation, etc. It specifies mechanisms for creating, selecting and maintaining data categories, as well as an interchange format for representing them.

2. Merging the results collected from the databases.

3. Presenting them in a succinct and unified format with minimal duplication.

4. Providing a means, performed either automatically or by the portal user, to sort the merged result set.

The list of criteria is surely debatable; however, it provides a set of fundamental guidelines that can be expanded or modified to fit each framework's needs, and as such we will employ it.

From the developer's point of view it is worth mentioning that recent MySQL versions offer the FEDERATED engine[16]. This storage engine enables data to be accessed from a remote MySQL database on a local server without using replication or cluster technology. When using a FEDERATED table, queries on the local server are automatically executed on the remote tables. No data is stored on the local tables. Although, simple approach, it can be very efficient for small datasets. However, there are some drawbacks:

- Some querying functions can be very expensive for large databases.
- The federated table is not aware of any structural changes on the remote server.
- The remote table must exist in order to create a local one.

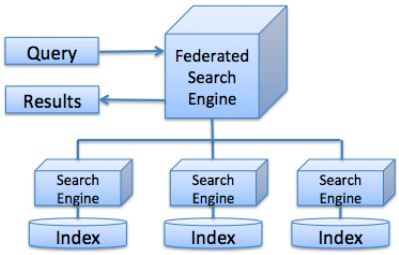

Figure 2.12: An illustration of a federated search. "Search engines" can be generalized to distributes sources.

Besides MySQL there are other, higher level tools such as Sesat [17]. Sesat stands for Sesam Search Application Toolkit, and is a platform that provides the framework and functionality required for handling parallel searches and displaying them elegantly in a user interface, thus allowing engineers to focus on the remaining aspects such as index/database configuration and tuning.

More implementations of federated search can be found online, hence we will end our discussion here.

[16] dev.mysql.com/doc/refman/5.0/en/federated-storage-engine.html
[17] http://sesat.no/

2.3.4 Repository Level

We shall now investigate interoperability at the repository level, aiming at creating a central catalogue covering metadata from various harvestable sources. The next paragraphs present related initiatives.

METS

METS (Metadata Encoding and Transmission Standard)[18] is an XML-schema that was developed as a standard data structure describing complex digital library objects. A METS file comprises of seven sections:

- METS header: It contains metadata describing the METS document itself, including information such as creator, editor, etc.
- Descriptive Metadata: This section can either point to external descriptive metadata or contain internally embedded descriptive metadata. It is possible for multiple instances of both external and internal descriptive metadata to be included in the descriptive metadata section. An example of a METS representation of an audio file is presented in Figure 2.13. The XML section denoted as *dmdSec* corresponds to the Descriptive Metadata section. It includes information such as the title of the song, instruments an others.
- Administrative Metadata: This section contains data regarding the file's creation and storing, intellectual property rights, metadata regarding the original source object from which the digital library object derives and others. Similar to descriptive metadata, administrative metadata may be either external or internal.
- File Section: The file section lists all files containing content which comprise the electronic versions of the digital object. In the example of Figure 2.13 this corresponds to *fileSec*.
- Structural Map: This section outlines a hierarchical structure for the digital library object, and links the elements of that structure to content files and metadata that pertain to each element.
- Structural Links: This section allows to record the existence of hyperlinks between nodes as modeled in the Structural Map.
- Behavioural: The behaviour section associates executable behaviours with content in the METS object.

METS has proven to be a convenient format since its records can be exchanged between different data centers using the popular OAI-PMH protocol (which is described below) [25]. In addition, the Fieldwork Data Sustainability Project (FIDAS) has shown that METS is a suitable format for the metadescription of linguistic resources.

[18] http://www.loc.gov/standards/mets/mets.xsd

Figure 2.13: A METS XML representation of a sound recording. Due to size, a large part of the file is omitted.

OAI-PMH and OAI-ORE

The Open Archives Initiative[19], defines two standards that aim to provide interoperability between heterogeneous repositories.

The first one, OAI-PMH (Protocol for Metadata Harvesting) establishes an XML message format for the exchange of metadata. The OAI-PMH world is devided into two groups: data and service providers. The first correspond to administrative systems that support the OAI-PMH as a means of exposing metadata, while the latter use metadata harvested via the OAI-PMH as a basis for building value-added services. Data providers are able to handle the six possible OAI requests issued by the service providers: *GetRecord, Identify, ListIdentifiers, ListMetadataFormats, ListRecords* and *ListSets*. OAI explicitly requires metadata to be encoded in DublinCore format, hence as we shall see later, loss of information occurs due to the limited number of available DC fields.

OAI-ORE (Object Reuse and Exchange) defines standards for the description and exchange of aggregations of Web resources. These aggregations, usually called "compound digital objects", may consist of distributed resources with multiple media types such as text, images, data, and video. ORE is based on a Web Architecture that assumes that every information object is defined by

[19]http://www.openarchives.org/

a URI. Exchange of compound digital objects is possible individually by direct web access, and via batch discovery mechanisms.

It is worth examining a simple HTTP request and response. Firstly the service provider, by employing the verb GetRecord, asks from the repository arXiv.org to retrieve the record with identifier *oai:arXiv.org:cs/0112017* in DC format.

```
http://arXiv.org/oai2?verb=GetRecord&identifier=oai:arXiv.org:cs/0112017&..
..metadataPrefix=oai_dc
```

The data provider,if it contains the specified record, sends an XML containing the appropriate information. The response is shown in Figure 3.12.

```
▼<OAI-PMH xmlns="http://www.openarchives.org/OAI/2.0/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/
 http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd">
   <responseDate>2002-02-08T08:55:46Z</responseDate>
   <request verb="GetRecord" identifier="oai:arXiv.org:cs/0112017"
   metadataPrefix="oai_dc">http://arXiv.org/oai2</request>
 ▼<GetRecord>
   ▼<record>
     ▼<header>
        <identifier>oai:arXiv.org:cs/0112017</identifier>
        <datestamp>2001-12-14</datestamp>
        <setSpec>cs</setSpec>
        <setSpec>math</setSpec>
      </header>
     ▼<metadata>
      ▼<oai_dc:dc xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/"
       xmlns:dc="http://purl.org/dc/elements/1.1/"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/
       http://www.openarchives.org/OAI/2.0/oai_dc.xsd">
       ▼<dc:title>
          Using Structural Metadata to Localize Experience of Digital Content
        </dc:title>
        <dc:creator>Dushay, Naomi</dc:creator>
        <dc:subject>Digital Libraries</dc:subject>
       ▼<dc:description>
          With the increasing technical sophistication of both information consumers and
          providers, there is increasing demand for more meaningful experiences of digital
          information. We present a framework that separates digital object experience, or
          rendering, from digital object storage and manipulation, so the rendering can be
          tailored to particular communities of users.
        </dc:description>
       ▼<dc:description>
          Comment: 23 pages including 2 appendices, 8 figures
        </dc:description>
        <dc:date>2001-12-14</dc:date>
      </oai_dc:dc>
     </metadata>
    </record>
  </GetRecord>
</OAI-PMH>
```

Figure 2.14: An OAI-PMH response to a GetRecord request.

Dublin Core

The Dublin Core Metadata Initiative started by establishing a small set of 13 elements with semantically broad categories, hoping for high scalability and generalization. The original objective was to define a set of elements that could be used by authors to describe their own Web resources. However, the proliferation of electronic resources and the inability of the library profession to catalogue all them, led to the extension of Dublin Core with elements with precise semantics. The resulting 15 elements are shown in Table 2.1.

Even with the extended element set, the need for further specificity was prominent. Hence, there has been an ongoing procedure to develop exemplary terms extending or refining the Dublin Core set. The additional terms are

Content	Intellectual Property	Instance
Title	Creator	Date
Subject	Publisher	Type
Description	Contributor	Format
Language	Rights	Identifier
Relation		
Coverage		
Source		

Table 2.1: The DC set.

identified, generally in working groups of the Dublin Core Metadata Initiative, and judged by a board to be in conformance with principles of good practice for the qualification of Dublin Core metadata elements[26]. An example of a Dublin Core record is shown in Figure 2.15.

```
<head profile="http://dublincore.org">
<title> … </title>

<link rel="schema.DC" href="http://purl.org/dc/elements/1.1/" />
<link rel="schema.DCTERMS" href="http://purl.org/dc/terms/" />

<meta name="DC.Identifier" schema="DCterms:URI"
content="http://tutorialsonline.info/Common/DublinCore.html" />
<meta name="DC.Format" schema="DCterms:IMT" content="text/html" /> <meta name="DC.Title" xml:lar
<meta name="DC.Creator" content="Alan Kelsey" />
<meta name="DC.Subject" xml:lang="EN" content="Dublin Core Meta Tags" />
<meta name="DC.Publisher" content="Alan Kelsey, Ltd." />
<meta name="DC.Publisher.Address" content="alan@tutorialsonline.info" />
<meta name="DC.Contributor" content="Alan Kelsey" />
<meta name="DC.Date" scheme="ISO8601" content="2007-01-06" />
<meta name="DC.Type" content="text/html" />
<meta name="DC.Description" xml:lang="EN"
content="Learning Advanced Web Design can be fun and easy! Look at a site designed specifically
<meta name="DC.Identifier" content="http://tutorialsonline.info/Common/DublinCore.html" />
<meta name="DC.Relation" content="TutorialOnline.info" scheme="IsPartOf" />
<meta name="DC.Coverage" content="Hennepin Technical College" />
<meta name="DC.Rights" content="Copyright 2011, Alan Kelsey, Ltd.  All rights reserved." />
<meta name="DC.Date.X-MetadataLastModified" scheme="ISO8601" content="2007-01-06" />
<meta name="DC.Language" scheme="dcterms:RFC1766" content="EN" />
```

Figure 2.15: An example of a DC record.

Dublin Core's employment by the OAI-PMH in addition to the various tools provided have made DC the most popular metadata standardization format. It is worth mentioning some of the tools such as Omeka[20] which is a free and open source web publishing system for online digital archives, that uses an unqualified Dublin Core metadata standard. Fedora[21] is a repository architecture compatible with OAI-PMH and therefore Dublin Core. It is built upon the idea that the integration of data, interfaces, and functions as clearly defined modules, ensures interoperability and extensibility.

TEI

The Text Encoding Initiative (TEI) is a popular standard for the representation of text in digital form. The encoding format used to rely on SGML, but later versions employed XML. Similar to DC, metadata can be embedded in the header of a TEI-file, which typically represent fields that correspond to a bibliographic record (e.g. title, distributor). As from the most recent version of

[20] omeka.org
[21] http://fedora-commons.org/

TEI (P5) its schema can be extended via the specification of the newly added elements.

```
<text xmlns="http://www.tei-c.org/ns/1.0" xml:id="d1">
  <body xml:id="d2">
    <div1 type="book" xml:id="d3">
      <head>Songs of Innocence</head>
      <pb n="4"/>
      <div2 type="poem" xml:id="d4">
        <head>Introduction</head>
        <lg type="stanza">
          <l>Piping down the valleys wild, </l>
          <l>Piping songs of pleasant glee, </l>
          <l>On a cloud I saw a child, </l>
          <l>And he laughing said to me: </l>
        </lg>
        <lg type="stanza">
          <l>"Pipe a song about a Lamb!" </l>
          <l>So I piped with merry chear. </l>
          <l>"Piper, pipe that song again;" </l>
          <l>So I piped, he wept to hear. </l>
        </lg>
        <lg type="stanza">
          <l>"Drop thy pipe, thy happy pipe; </l>
          <l>Sing thy songs of happy chear:" </l>
          <l>So I sung the same again, </l>
          <l>While he wept with joy to hear. </l>
        </lg>
        <lg type="stanza">
          <l>"Piper, sit thee down and write </l>
          <l>In a book, that all may read." </l>
          <l>So he vanis'd from my sight, </l>
          <l>And I pluck'd a hollow reed, </l>
        </lg>
        <lg type="stanza">
          <l>And I made a rural pen, </l>
          <l>And I stain'd the water clear, </l>
          <l>And I wrote my happy songs </l>
          <l>Every child may joy to hear. </l>
        </lg>
      </div2>
```

Figure 2.16: An example of William Blake's Songs of Experience and of Innocense. Both songs encoded as *div1* numbered text divisions, with a *type* attribute with value book. Inside these books, all 45 poems are encoded as *div2 type="poem"*. All poems have a title and are subdivided into stanzas *lg type="stanza"*. Due to size issues, a large part of the XML file is omitted.

IMDI

The IMDI Framework for multimedia and multimodal language resources differs from the previously described initiatives; besides a set of XML-based metadata descriptors, IMDI offers a set of tools and an infrastructure to use these. Some of its main characteristics are:

- It allows related resources to be bundled by metadata descriptions. In other words, it allows resources such as two or more audio tracks, texts and others to be grouped into the recording that pertains them.
- By embedding pointers in the XML format metadata description files, records can be linked to form structured virtual organizations facilitating browsing and management. This linking feature offers the ability to create well-designed, hierarchically structured sub-corpora as well as to create some sort of relations.

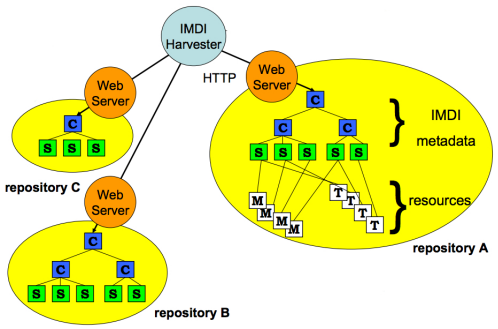

Figure 2.17: A representation of the IMDI architecture. Repositories have a hierarchical data structue (multimedia content deoted as M and text as T). An IMDI harvester can be used to gather data from all sources.

OLAC

The OLAC Metadata Set is yet another set of metadata elements for describing language resources. It is employed by the members of the Open Language Archiving Community so as to harvest data (via OAI-PMH) from all registered repositories and archives.

OLAC can be considered an extension of DC since its set is based on the latter and uses all fifteen elements defined in that standard. However, in order to increase specificity, OLAC follows the DC recommendation for qualifying elements by means of element refinements or encoding schemes [27]. In simple words, OLAC acts as a combination or application profile of the Simple and Qualified DC, with the ability of incorporating other namespaces.

```
▼<olac:olac xmlns:olac="http://www.language-archives.org/OLAC/1.1/" xmlns="http://purl.org/dc/elements/1.1/"
  xmlns:dcterms="http://purl.org/dc/terms/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.language-archives.org/OLAC/1.1/ http://www.language-
  archives.org/OLAC/1.1/olac.xsd">
   ▼<title>
      A grammar of Kayardild. With comparative notes on Tangkic.
    </title>
    <creator>Evans, Nicholas D.</creator>
    <subject>Kayardild grammar</subject>
    <subject xsi:type="olac:language" olac:code="gyd">Kayardild</subject>
    <language xsi:type="olac:language" olac:code="en">English</language>
    <description>Kayardild Grammar (ISBN 3110127954)</description>
    <publisher>Berlin - Mouton de Gruyter</publisher>
    <contributor xsi:type="olac:role" olac:code="author">Nicholas Evans</contributor>
    <format>hardcover, 837 pages</format>
    <relation>related to ISBN 0646119966</relation>
    <coverage>Australia</coverage>
    <type xsi:type="olac:linguistic-type" olac:code="language_description"/>
    <type xsi:type="dcterms:DCMIType">Text</type>
  </olac:olac>
```

Figure 2.18: A simple OLAC record.

MPEG7

MPEG-7 is a multimedia content description standard. The rationale behind it is to allow fast and efficient searching for multimedia material. It uses XML to store metadata, and can be attached to time code in order to tag particular events, or synchronize lyrics to a song, for example. MPEG-7 was never widely used due to its complexity.

Other

Other initiatives and efforts worth mentioning are the Natural Language Software Registry (NLSR)[22], the Association for Computational Linguistics Repositiry (ACL) [23]and the Learning Object Metatada standard (LOM)[24]. The first one corresponds to a summary of a large amount of natural language processing software available to the Natural Language Processing community. The ACL Data and Code Repository is a repository of data (e.g., hand-labeled text, hand-parsed text, feature vectors for machine learning, etc.) and source code (e.g., taggers, parsers, chunkers, etc.) for computational linguistics and natural language processing. The Learning Object Metadata standard was developed to enable the use and re-use of technology-supported learning resources such as computer-based training and distance learning. The LOM defines the minimal set of attributes to manage, locate, and evaluate learning objects encoded in an XML format.

CLARIN

CLARIN's website[25] states as its "main goal": CLARIN is committed to establish an integrated and interoperable research infrastructure of language resources and its technology. It aims at lifting the current fragmentation, offering a stable, persistent, accessible and extendable infrastructure and therefore enabling eHumanities.

- Integrated: the resource and service centres are connected via Grid technology and form a virtually integrated domain.
- Interoperable: the resources and services will be based on Semantic Web technologies to overcome format, structure and terminological differences.
- Stable: the resources and services are offered with a high availability.
- persistent: the resources and services are planned to be accessible for many years so that researchers can rely on them.
- Accessible: the resources and services are accessible via the web; different access methods and training possibilities are offered tailored to the needs of the communities making use of them.
- Extendable: the infrastructure is open so that new resources and services can be added easily.

In general, CLARIN can be considered an open research infrastructure providing facilities for storing data and tools while also browsing and searching. It has been built on top of ISOcat and other trusted registries, meaning that it makes use of their shared data registries as controlled vocabulary to ensure interoperability. However CLARIN uses its own design of metadata descriptions denoted (CMDI). The novel design concept derives from "components". Components act as metadata templates that the local server can use as building

[22]http://registry.dfki.de/
[23]http://aclweb.org/
[24]http://ltsc.ieee.org/wg12/
[25]http://www.clarin.eu/external/index.php?page=about-clarin&sub=0

blocks for more complex schemas. The validity of these schemas can be verified via the web interface; in addition other components can be recommended. According to the developers themselves the re-use of components is highly promoted, although users can generate their own.

CMDI uses existing schemas such IMDI, OLAC and DC, on which more complex schemas can be built, used and registered. Each metadata field is linked to exactly one data category in a data category registry (DCR) using a persistent identifier. The DCR indicates how the content of the field in a metadata description should be interpreted. If the same data category is used in various metadata schemas, the reference to the DCR will still be the same. Figure 2.19 presents an abstraction of the whole CMDI architecture employing CLARIN as the input interface and ISOcat, DCMI as the metadata registries. Metadata components encoded in CMDI format point to ISOcat semantic concepts via PID identifiers.

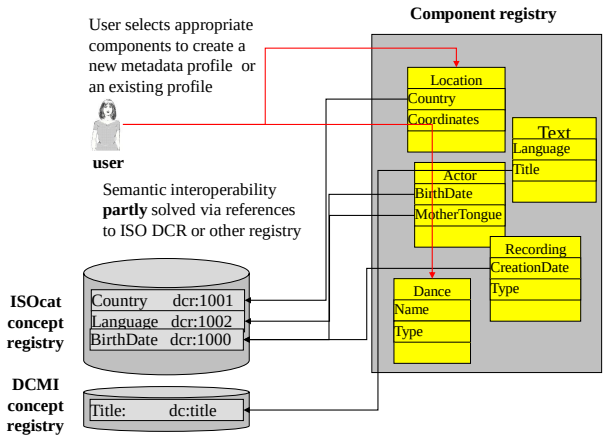

Figure 2.19: A simplified illustration of the CMDI framework using ISOcat, DCMI as registries.

As we've already mentioned CLARIN in addition offers cross-searching between sources. This is achieved via the OAI-PMH protocol and as a result, all repositories need to employ the DC metadata-scheme. Inside The Netherlands many projects have been initiated aiming at employing CLARIN, such as AAM-LR, Adelheid, ADEPT, MIMORE[26] and others.

MuSeUM: Unified Access to the State of the Art

In the previous paragraphs we presented a set of initiatives that aimed at addressing the problem of interoperability at its very core. Most of the tools previously mentioned can be adopted by any developer, archive, enterprise or whoever requires such functionality. However, it is also quite common for specialized systems to employ their own framework that cannot be generalized as easily as CLARIN or DC for example.

Arampatzis et al.[29] addressed the problem of disclosing multiple data and meta-data collections from various cultural heritage insitutes, each with its own

[26] http://www.meertens.knaw.nl/mimore/

characteristics, in a single, unified system. Their approach was based on unconditional merging of all collections and flattening of all metadata structures. The data was later indexed using Apache Lucene. This brute-force process transformed the sophisticated problem of information retrieval into free-text retrieval.

Simple as that, the authors showed the superiority of their approach when compared to the previous search systems used by the institutes. The evaluation was based on two figures: retrieval performance and user satisfaction. For the first one, a set of ad-hoc known topics were used as ground truth. Both systems (old and new) were queried and metrics such as MRR (Mean Reciprocal Rank), were employed to calculate the overall performance. User satisfaction was based on a small user study and questionnaires.

2.4 Conclusions

The previous parts tried to guide the reader through the vast world of interoperability. Various definitions were presented along with relevant tools and initiatives, mostly focusing on the notion of "search" interoperability. It should be now obvious that interoperability cannot be quantified. It is a characteristic that describes an underlying connection between sources and not some kind of input/output function. In other words, interoperable systems don't necesseraly employ a certain framework or protocol. Each enterprise of data sources needs to weigh the pros and cons of all possible interconnection methods, before employing or build on top of one.

Bibliography

[1] A. Tolk, J. A. Muguira, "The Levels of Conceptual Interoperability Model", in 2003 Fall Simulation Interoperability Workshop. 2003.

[2] Interoperability, Wikipedia Entry. en.wikipedia.org/wiki/Interoperability

[3] Syntactic Interoperability, Wikipedia Entry. en.wikipedia.org/wiki/Interoperability#Syntactic_interoperability

[4] Semantic Interoperability, Wikipedia Entry, en.wikipedia.org/wiki/semantic_interoperability

[5] Kim H. Veltman, " Syntactic and Semantic Interoperability: New Approaches to Knowledge and the Semantic Web", The New Review of Information Networking, vol. 7. 2001.

[6] T. Bittner, M. Donnelly, S. Winter, "Ontology and Semantic Interoperability", in: D. Prosperi and S. Zlatanova (ed.), Large-scale 3D Data Integration: Problems and Challenges, London: CRCPress. 2005.

[7] D. Broeder, M. Kemps-Snijders, , D. V. Uytvanck, M. Windhouwer, P. Withers, P., Wittenburg, C. Zinn, "A Data Category Registry and Component-based Metadata Framework", Proceedings BROEDER. 2010.

[8] M. Nagarajan, K. Verma, A. P. Sheth, J. Miller, J. Lathem, "Semantic Interoperability of Web Services Challenges and Experiences", Proceedings of the Fourth IEEE International Conference on Web Services (ICWS). 2006.

[9] J. Searle, J. Brennan, "General Interoperability Concepts"' Simulation and Synthetic Environment Laboratory (SSEL).

[10] "Levels of Information Systems Interoperability", C4ISR Architectures Working Group, 30 March 1998, available at: US DoD, OSD (C3I), CIO, Director for Architecture and Interoperability.

[11] C.D Turnitsa, "Extending the Levels of Conceptual Interoperability Model", Proceedings IEEE Summer Computer Simulation Conference, IEEE CS Press. 2005.

[12] M.D. Petty, E.W. Weisel, "A Composability Lexicon". Proceedings IEEE Spring Simulation Interoperability Workshop, IEEE CS Press, 2003

[13] M. L. Zeng, L. M. Chan, "Metadata Interoperability and Standardization - A study of Methodology", D-Lib Magazine, Volume 12, number 6. 2006.

[14] O. Hassanzadeh, A. Kementsietsidis, Anastasios L. Lim, R. Miller,"A Declarative Framework for Semantic Link Discovery over Relational Data. Engineering", Proceedings of the 18th ACM conference on Information and knowledge management (CIKM). 2009.

[15] Linking open data, Wikipedia entry. en.wikipedia.org/wiki/Linked_data

[16] A. Swartz, " MusicBrainz: a semantic Web service". IEEE Intelligent Systems. 2002.

[17] Heery, Rachel, Manjula Patel, "Application profiles: Mixing and Matching Metadata Schemas." Ariadne 25. http://www.ariadne.ac.uk/issue25/app-profiles/

[18] C. Bizer, T. Heath, T. Berners Lee, " Linked Data - The Story So Far". International Journal on Semantic Web and Information Systems (IJSWIS). 2009.

[19] Y. Raimond, C. Sutton, M. Sandler, "Automatic Interlinking of Music Datasets on the Semantic Web", Proceeding of the Linked Data on the Web Workshop (LDOW). 2008.

[20] G. Kobilarov, T. Scott, Y. Raimond, S. Oliver, C. Sizemore, M., Smethurst, C. Bizer, "Media Meets Semantic Web: How the BBC Uses DBpedia and Linked Data to Make Connections", Proceedings of the 6th European Semantic Web Conference on The Semantic Web: Research and Applications. 2009.

[21] M. Neiling, "Data Fusion with Record Linkage", Presentation at the 3rd Workshop Foderierte Datenbanken. 1998.

[22] [22] J. Volz, C. Bizer, M. Gaedke, G. Kobilarov, " Silk: A Link Discovery Framework for the Web of Data", Proceeding of the 2nd Workshop about Linked Data on the Web (LDOW). 2009.

[23] T. Berners Llee, Y. Chen, L. Chilton, D. Connolly, R. Dhanaraj, J. Hollenbach, A. Lerer, "Tabulator : Exploring and Analyzing linked data on the Semantic Web". In Proceedings of the 3rd International Semantic Web User Interaction Workshop. 2006.

[24] P. Jasco, "Thoughts About Federated Searching". Information Today, Vol. 21, Issue 9. 2008.

[25] R. Tansley, "Building a Distributed, Standards-based Repository Federation". D-Lib Magazine, Volume 12 Number 7/8. 2006.

[26] Dublin Core, Wikipedia entry, wikipedia en.wikipedia.org/wiki/DublinCore

[27] OLAC metada, www.language-archives.org/OLAC/metadata.html

[28] D. Hawking, "Challenges in Enterprise Search", Proceedings of the 15th Australasian database conference (ADC). 2004.

[29] A. Arampatzis, J. Kamps, M. Koolen, N. Nussbaum, "MuSeUM: Unified Access to the State of the Art", Proceedings of the 7th Dutch-Belgian Information Retrieval Workshop (DIR). 2007.

Chapter 3

Data Enrichment

Music Information Retrieval has gained a great amount of popularity over the last years. The reason for this is multifaceted, however it can be mainly ascribed to technological advances (hard drives, portable music players etc.) and the digression of the musical industry to digital rather than physical audio formats.

Considering the above, a lot of the MIR research has focused on the automatic extraction of information about pieces of music or music artists themselves. Early approaches were based on the analysis of audio signal, in order to extract features describing the genre, tonality, tempo and other characteristics of a piece. It wasn't until recently were the focus moved to"cultural" features. These features are based upon the notion of "wisdom of the crowd", meaning:

> "the process of taking into account the collective opinion of a group of individuals rather than a single expert to answer a question. A large group's aggregated answers to questions involving quantity estimation, general world knowledge, and spatial reasoning has generally been found to be as good as, and often better than, the answer given by any of the individuals within the group. An intuitive and often-cited explanation for this phenomenon is that there is idiosyncratic noise associated with each individual judgement, and taking the average over a large number of responses will go some way toward cancelling the effect of this noise" [15].

Depending on the context, cultural features are derived from different sources. It has been common though to use "community metadata", meaning sources related to particular fields. When it comes to music, these are online music stores, collections, music services (Last.Fm, MusicBrainz etc.) and others.

However recently, a lot of research has diverged to exploit the World Wide Web (denoted Web MIR). As Scheld mentions in his PhD thesis, "It is generally assumed that the information provided by the zillions of Web pages and services of the WWW also represents a kind of cultural knowledge and, therefore, can be beneficially used to derive cultural features for MIR tasks". Exploiting the Web is typically achieved via the employment of web crawlers and other web harvesting techniques.

So how does data enrichment for music relate to the MIR automatic extraction of information? To our understanding, there's a thin line separating those notions with regard to the final outcome. However, when it comes to methods,

data enrichment mostly employs already existing data. By combining, filtering and processing it, important conclusions are derived. For example, by exploiting play list data from a single user, someone can derive the era of music that the user mostly enjoys. On the other hand MIR automatic extraction methods aim at getting information from and for one piece of music only. For example, given the audio representation of a song, someone can derive its genre. However, ambiguity between those notions can be still introduced depending on the context. Therefore, for the rest of this report, data enrichment for music and MIR automatic extraction of information will denote the same procedure.

We will now present a brief overview of related published approaches. We will later discuss in detail four of them that show great generalization of concepts, meaning that their methods can be applied to various tasks and topics.

3.1 Literature Overview

Fan and Cohen [21] were among the first to exploit the Web for MIR purposes, namely music recommendation. Their method employed a collaborative filtering web spider that gathered lists of interesting entities (bands, artists) while encoding them as pseudo-users. The recommendation was based on the comparison of the actual user's preferences against the pseudo ones.

Ellis et al [22] evaluated a set of artist similarity scores against a Web-survey "ground truth". The purpose of such an experiment was to assess the quality of automatic rating against the subjective impressions of the users. The most interesting part relates to the "Erdos similarity", which presented the highest correlation to users' opinion. More precisely, the Erdos similarity between two artists a_1 and a_2 is measured as the minimum number of intermediate artists needed to form a path from a_1 to a_2 using the similar artist relationships.

In [1] Whitman and Lawrence extracted various term sets (uni-grams, bi-grams, noun phrases, artist names, and adjectives) from artist-related Web pages. For each artist and based on term occurrences, individual term profiles were created. The"overlap" between the term profiles of two artists was then used to estimate their distance/similarity. Whitman and Lawrence were among the first to query search engines with additional terms such as "music" and "review" rather than just the artist name. Whitman and Smaragdis [12] employed similar method for classifying artists among five genres.

Baumann and Hummel [13], similar to the previous researchers, queried search engines and generated n-grams for each artist. However their method employed term frequency, inverse document frequency weightings (see next section). Their method in addition removed noisy content such as ads from HTML pages.

Geleijnse and Korst [14], on their work on genre and mood classification for artists, employed a"pattern based matching" technique. Instead of simply searching for co-appearances of artists and genres in the same Web page, their method searches for phrases such as "artist A such as artist B". Their work also took into consideration synonyms, while it also employed quering with multiple terms such as"artist+music".

Celma [2] proposed a music search engine that crawls audio blogs via RSS feeds. The feeds contain links to music files in addition to short text descriptions of the pieces. These are combined with metadata extracted from the audio files

(in particular, from the ID3 tags), thus enabling user queries to be matched with music files.

An interesting work by Knees, Pampalk and Widmer [16] employed machine learning approaches for genre classification and artist similarity. Similar to most of the pre-mentioned approaches search engines such as Google and Yahoo were used to download artist-relevant pages. Then a histogram of frequency terms appearing along with the artis name was created. All of the histograms were fed to a support vector machine classifier, k-nearest neighbour classifier and a self-organizing map.

One of the pioneers in the field of MIR information extraction via web mining is Schedl. He has done a large amount of work on that particular field, and some of his papers will be discussed later thoroughly. For now, this brief overview of approaches is sufficient and delineates the fact that most of the methods share a common ground in terms of data retrieval techniques. The next section discusses those techniques.

3.2 Overview of Techniques

This section investigates and discuss some of the most common techniques and methods related to Web content mining and information extraction. Most of these methods are employed by the published approaches discussed in the previous part and are well described in Scheld's and McKay's PhD theses [3, 4]. Therefore, we will discuss them briefly and forward the reader to those theses for further details.

3.2.1 Focused Web-crawlers and Querying Search Engines

Before any similarity, distance, membership and other calculations are performed, a collection of documents must be retrieved from the Web. Two are the most commonly used approaches for such a task:

1. Focused web crawlers take as input a list of seed URLs related to a certain topic. Typically, a probabilistic classifier is trained based on the relatedness of each page to the topic of interest. Therefore starting with the initial seeds, the focused crawler iteratively follows the links that are judged relevant to the topic by the classifier. Considering web pages as nodes in a tree graph and their included links as children nodes, web-crawlers usually employ a best-first exploring method.

2. Querying search engines relies on the search engine's web-crawlers to retrieve a set of related documents. The main difference between the two techniques is that search engines use breadth-first rather than best-first algorithms for visiting web pages. The disadvantage of querying search engines is that the relatedness of the returned documents is not always guaranteed. For that reason, many of the published approaches described above employ queries with additional terms such as "music+review".

3.2.2 Term and Inverse Term Frequency $tf * idf$

The $tf * idf$ weight is a statistical measure that models how important a word is to a document d in a collection of documents. The $tf * idf$ value is proportional to the number of appearance in the document, however this is counterbalanced by the frequency of the word in the corpus.

More specifically, term frequency weight $tf_{t,d}$ corresponds to the times that a term t appears in a certain document d. For a document d , the set of weights determined by the $tf_{t,d}$ weights can be viewed as a quantitative representation of that document. In the literature this view is often denoted as the "bag of words model".

Term frequency lacks discriminating power in determining relevance. For example, when searching for a music artist, the term "music" will appear numerous times across the whole collection of documents while it will have high $tf_{t,d}$ value for each d. However, this doesn't necesserilly mean that it is representative of the documents containing it. Inverse document frequency aims at counterbalancing this effect by reducing the weight of a term by a factor that grows with its collection frequency. Hence inverse document frequency is defined as follows:

$$idf_t = log \frac{N}{df_t} \tag{3.1}$$

where N the total number of documents and df_t the number of documents where the term t appears. Then $td * idf$ is defined as such:

$$tf * idf_{t,d} = tf_{t,d} \times idf_t \tag{3.2}$$

Variations of the $tf * idf$ weighting scheme are typically employed by search engines in order to score and rank documents' relevance based on user queries. For example Apache Lucene employs $tf * idf$ to calculate the query-document similarity as shown in the formula below:

$$sim(q,d) = \sum_{t \in q} \frac{tf_{t,q} \cdot idf_t}{norm_q} \frac{tf_{t,d} \cdot idf_t}{norm_d} \cdot coord_{q,d} \cdot weight_t \tag{3.3}$$

for a collection D, document d, query q, query term t and:

$$tf_{t,X} = \sqrt{freq(t,X)} \tag{3.4}$$

$$idf_t = 1 + log \frac{|D|}{freq(t,D)} \tag{3.5}$$

$$norm_q = \sqrt{\sum_{t \in q} tf_{t,q} \cdot idf_t} \tag{3.6}$$

$$norm_d = \sqrt{|d|} \tag{3.7}$$

$$coord_d = \frac{|q \cap d|}{|q|} \tag{3.8}$$

3.2.3 Collaborative Filtering, Co-occurrence Analysis and Cross-tabulation

Collaborative filtering, as its name implies, refers to a method that links entities (eg. CDs, books, music artists) based on the users' profiles, interests and expressed behaviour. In simpler words, given that two users have expressed similar taste in music by purchasing the same album, then this method can recommend another album purchased by user A to user B. Such a technique is employed by vendors such as Amazon, and its weaknesses are obvious:

- Large amount of user data is required to achieve acceptable confidence values. Otherwise the method models noise.
- Popular entities are usually more favoured than those unpopular which are ignored. This is also known as the "cold start problem".
- Gift items can are a common case that introduces noise.

Co-Occurrence analysis models similarity between two items/terms based on their co-occurrence in the same web context. In other words, if two items appear on the same web page, it is more likely for them to be similar in some sense. However, it becomes quite obvious that the definition of "similarity" in that context is rather vague. Aucouturier and Pachet [5] found co-occurrence analysis generates similarity clusters that mostly model thematic/genre and period relationships. Artists that appear in the the same web source tend to belong to the same genre or artistic era or even year (a result of the "Best of Year X" compilations). Once again though, the problem of labeling the clusters' meaning remains.

It should be noted that there is a distinction between the occurrence of two items from the same and different vocabularies, denoting co-occurrence and *cross-tabulation analysis* respectively. For example, given an set A of artist names and a set B of genres , co-occurrence analysis calculates the co-appearance of items in set A (e.g. Artist i, Artits j), while cross-tabulation of couples from A and B (e.g. Artist i, Genre k).

Whatever the case, calculating co-appearances of strings presents certain weaknesses:

- Synonyms induce noise. For example such methods cannot distinguish between entities with the same or even similar name.
- Variations of an entity's name can be also confusing.
- Names that consist of a set of substrings e.g.. Justin Bieber, induce noise, since each substring can appear on a web page seperately from the rest. Searching for exact string matches e.g. "Justin Bieber" may miss cases such as "Bieber" or "J. Bieber".
- Negative meanings can be confusing. For example the string "Iron Maiden are not like Justin Bieber" presents a dissimilarity between the two artists. However co-occurrence analysis will classify this case as belonging together.

- Finally, as initially stated, the semantic meaning of a co-occurrence is ambiguous. Some artists may appear together but this doesn't ensure significant relatedness or similarity.

Despite the previous, co-occurrence analysis is one of the most popular techniques for information extraction on the Web. Its success is largely based on the fundamental ideas of "wisdom of the crowd", and namely the on the fact that the vastness of the Web filters out noise. We shall now present the different formulations of co-occurrence that aim at counterbalancing weaknesses such as popularity bias. For the remaining of this section $C(x)$ will denote the counts of a term x, and $C(x, y)$ the counts of the co-occurance of terms x, y:

1. Patchet et al.[17] employed co-occurance on radio playlists and CD databases. Their normalized co-occurance is defined as follows:

$$Cooc_{norm}(a, b) = (\frac{C(a, b)}{C(a)} + \frac{C(a, b)}{C(b)})/2 \tag{3.9}$$

   Similarity it is later calculated as such:

$$S_1(a, b) = 1 - Cooc_{norm}(a, b) \tag{3.10}$$

   In addition, in order for indirect links between entities to be taken into account, Patchet introduces:

$$S'_2(a, b) = \frac{Cov(a, b)}{\sqrt{Cov(a, a) \times Cov(b, b)}} \tag{3.11}$$

   where Cov the covariance. The distance is then defined as:

$$S_2(a, b) = 1 - (1 + S'_2(a, b))/2 \tag{3.12}$$

2. Whitman and Lawrence [1] queried search engines with tuples of the form *artits, music+review*. For each web-page retrieved, its content surrounding artist names is divided into n-grams. Based on statistical features regarding various terms, a histogram of term frequencies is generated for each artist. Whitman and Lawrence introduces the following similarity measure:

$$S_3(a, b) = \frac{C(a, b)}{C(b)}(1 - \frac{|C(a) - C(b)|}{C(c)}) \tag{3.13}$$

   where $C(c)$ the counts of the most popular item or artist in tha case.

3. Zadel and Fujinaga[18] employ the following similarity measure, to generate clusters of similar artists, based on Amazon and Google web services:

$$S_4(a, b) = \frac{C(a, b)}{min(C(a), C(b))} \tag{3.14}$$

4. Geleijnse and Korst[19] uses a method derived from information theory in order to classify artists by genre and mood. Each artist a in the set A is assigned a membership score in class g denoted T:

$$T(a,g) = \frac{C(a,g)}{1 + \sum_{i \in A} C(i,g)} \tag{3.15}$$

5. Schedl et al.[20] uses two measures to calculate the membership of an artist a in a genre g using:

$$t_1 = \frac{C(a,g)}{C(a)}, t_2 = \frac{C(a,g)}{C(g)} \tag{3.16}$$

This concludes our brief overview of the different co-occurrence analysis usages. So far none of them is consider optimal, since depending on the context of use, their behaviour may vary.

3.3 Published Approaches

In the following paragraphs we will present a series of published approaches in more detail. Our goal is to discuss all the different steps included in such MIR information extraction systems.

3.3.1 Automatically Extracting, Analysing, and Visualizing Information on Music Artists from the World Wide Web

Schedl's PhD thesis [3] describes the implementation of a series of tools, most of them incorporated inside the CoMIRVA framework (Collection of Music Information Retrieval and Visualization Applications). These tools range from the typical feature extraction to information extraction via web mining techniques and visualization.

AGMIS (Automatically Generated Music Information System) acts as a search engine for music artists with enhanced functionalities. The gathering of artist related data from the Web is performed offline by querying search engines such as Google and Exolead. The queries were of the form *"artist name" NEAR music* and resulted in the huge amount of 732.6 gigabytes of web sources. The pages were then indexed using Apache Lucene.

On the basis of this data, AGMIS calculated artist similarity by calculating a $tf * idf$ vector for each artist. The bins of such vectors corresponded to the elements of a music dictionary (set of music related terms). Regarding genre classification, Schedl queried Exolead with two different term configurations:

- "artist name " AND music
- "artist name " AND "genre name" AND music

The first one captures the genre-independent popularity of an artist pc_a, the second one the popularity with respect to genres $pc_{a,g}$. Eventually the membership r of an artist is calculated using the following equation:

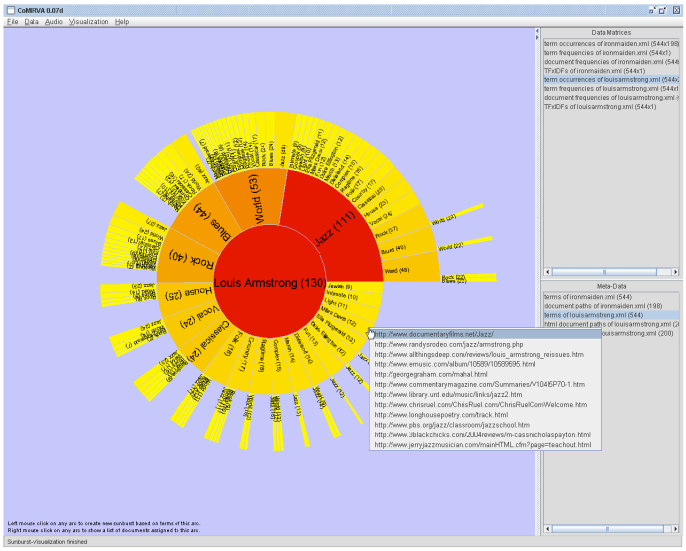

Figure 3.1: A screenshot of the CoMIRVA GUI.

$$r(a,g) = pc_{a,g}(log\frac{1}{pc_a + 1})^2 \tag{3.17}$$

Scheld's thesis also includes extensive experiments aiming at evaluating and comparing $tf * idf$, co-occurrence analysis while also various querying configurations such as : ("artist/band name "+music), ("artist/band name "+music+review) and ("artist/band name "+music+genre+style). The interpretation of the results cannot be summarized in a few lines, hence we forward the reader to the original document for further details. It is our belief though, that the performance of systems like that depends at large on the collection of documents, context and other factors such as the amount of web-pages retrieved for each artist.

3.3.2 A Web-based Approach to Determine the Origin of an Artist

Govaerts and Duval [6] exploited community metadata for determining artists' country of origin. By country of origin the authors denote the geographical locations where an artist started his career.

This approach relies on three methods namely metadata extraction from repositories such as Last.fm, Freebase and biography analysis. Last.fm is one of the most popular music recommender systems while Freebase is semantic database that contains structured data harvested from different sources. Last.fm was harvested using Dapper and Freebase was queried using its own querying language.

One of the most interesting methods described in the original paper, was the analysis of biographies using demonyms. Demonyms correspond to names of the inhabitants of a country or adjectives of geographical locations (e.g. German, Germany). The mapping between demonyms was achieved via Wikipedia. After

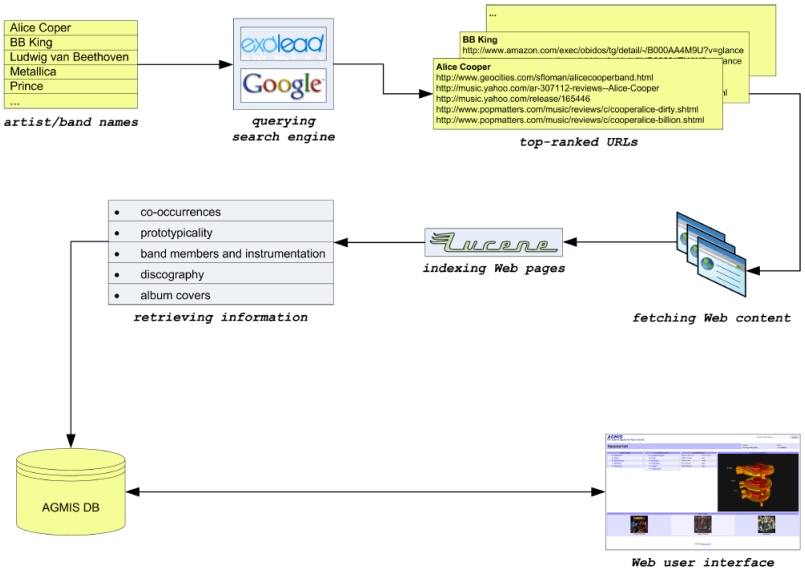

Figure 3.2: Data processing diagram of AGMIS.

that offline procedure each biography was split into natural language sentences, and for every sentence the demonyms and locations were noted and counted. Based on those, three figures well calculated:

- Highest occurrence: the demonym or location that appears most frequently.
- Favor first occurrence: for every country code (cc_i), the sentences containing it are kept (where s_j the sentence number). If s_{tot} is the total number of sentences in the biography then:

$$R_{cc} = \sum_{i=0}^{length(s)} \frac{(s_{tot} + 1) - s_i}{s_{tot} + 1} \tag{3.18}$$

  This scheme gives higher weight to the first sentences in a biography that contain the country of origin. It is based on the author's assumption that the origin of an artist is often mentioned in the first lines.

- Weaker favouring first occurrence: this is similar to the previous but it uses different weighting so as to spread the importance more over the sentences:

$$R_{cc} = \sum_{i=0}^{length(s)} \frac{(s_{tot} + 2) - s_i}{s_{tot} + 1} \tag{3.19}$$

In the experimental section, the authors first evaluated how rich each data source is (denoted coverage). They found that for 62% of the artist's origin can be retrieved with at least one of the three methods (Last.fm, Freebase,

biographies). The accuracy though presents fluctuations; continent determination shows high accuracy with Last.fm achieving 95%. Biographies on the other hand did not perform as expected. The country of origin results show lower performance but with Last.fm and FreeBase around 90%. A combination method, that aimed to combine the results of all methods, was also evaluated but didn't outperform the best.

It is worth mentioning that the authors' assumption that the origin is mentioned in the first sentences in a biography, is proven true by the experiments. Also, each method has its issues. For example Last.fm sometimes uses demonyms instead of country names and ambiguity is introduced in cases where a city/town name appears in multiple countries.

3.3.3 Country of Origin Determination via Web-mining Techniques

Schedl et al. [7] following Govaerts work presented above, aimed at determining the artist's country of origin. In their approach Google was queried with strings of the form "artist music". The first 100 pages returned pages were retrieved and indexed with Apache Lucene, similar to AGMIS.

Schedl's approach uses three different methods for origin determination:

- Page counts approach: this method is based on the assumption that querying search engines with tuples of the form "artist, country" will return more page results for the artist's country of origin. The search engine employed was Google since it has proved to outperform Yahoo! and MSN search (currently renamed to Bing Search) [8].

- Term Weighting approaches: this method is based on the $tf * idf$ weight vector or bag-of-words model as described in 3.2.2. Each document retrieved is assigned a term profile or histogram; each bin corresponds to a term and the value assigned corresponds to its importance in the document. The term weighting measures employed differ slightly from those in 3.2.2 and are presented below:

  - Document frequency: $df_{t,a}$ represents the total number of pages retrieved for an artist a where at least one time t appears.
  - Term frequency $tf_{t,a}$ is the total number of occurrences of term t in the set of retrieved documents for artist a (denoted as "virtual document" for a).
  - Term frequency-inverse document frequency: the classical formulation of $tf * idf$ as described in 4.4.2 performed weakly so Schedl and his colleagues used variations:

$$tf * idf^1_{t,a} = (1 + log_2 tf_{t,a}) log_2(\frac{n}{df_t}) \tag{3.20}$$

    for $tf_{t,a} > 0$, otherwise $tf * idf$ is set to 0.

$$tf * idf^2_{t,a} = ln(1 + tg_{t,a}) ln(1 + \frac{n}{df_t}) \tag{3.21}$$

    In both formulations n denotes the total number of Web pages retrieved and df_t the number of pages containing the term t.

– Text distance approaches: this category of heuristic methods aims at determining the country of origin by calculating the offset distance between key terms such as "born", "origin" etc. and country names. This approach integrates two functions: a distance measure on the document level to determine the distances within a Web page; an aggregation function to combine the document level distances for all pages of a.

The experiments showed that term frequency tf performs slightly better than df (82% and 79% for the f-measure respectively). All the different configurations of the text distance approaches while also page counts performed much worse.

3.3.4 Automatic Music Classification with jMIR

McKay [4], as part of his dissertation, developed a framework of tools (jMIR) that enables users to extract meaningful information from audio recordings (jAudio), symbolic musical representations (jSymbolic) and cultural information relating to music that is available on the Internet (jWebMiner). The functionalities of his implementation vary from automatically building classification models to automatically collecting profiling statistics, to detecting metadata errors in musical collections and others.

The most interesting tool of jMIR, as far as this thesis is concerned is jWebMiner[1] and a relevant sub-tool that is offered online, jSongMiner[2]. In a sense, jWebMiner is nothing more than a set of nicely, gathered tools for Web MIR. It employs the Google and Yahoo! search engines in order to acquire hit counts indicating the number of pages containing one or more search strings. jWebMiner offers the two fundamental types of feature extractions, co-occurrence and cross tabulation, and their set of modifications as described in 3.2.3.

jSongMiner is far more interesting, since it is used for auto-identifying songs and extracting metadata about them from various sources on the web. jSongMiner begins by identifying unknown audio files using fingerprinting. If the audio is absent, it can identify songs using metadata queries, meaning known metadata about a song. Once jSongMiner has identified a song, it can then extract metadata about the song from various sources, such as from The Echo Nest, Last.FM and Music Brainz. From there on it is possible to extract metadata about artists and albums associated with songs as distinct resources.

The whole jMIR framework was employed for a series of tasks and has shown great performance. However, we will not present any results due to the variation and length of those experiments. The reader is forwarded to McKay's thesis for further details.

3.3.5 Mining Microblogs to Infer Music Artist Similarity and Cultural Listening Patterns

One of the latest published approaches by Shedl and Hauger [11] aims at exploiting microblogs, and more specifically Twitter, for music artist similarity. The idea behind the method is to leverage the on growing popularity of such

[1] jmir.sourceforge.net/index_jWebMiner.html
[2] jmir.sourceforge.net/jSongMiner.html

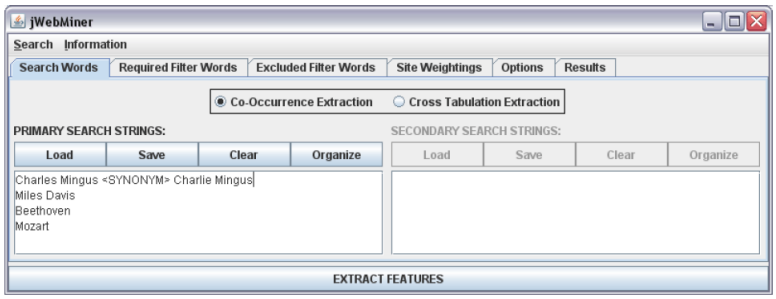

Figure 3.3: Query terms for a sample co-occurrence feature extraction that will measure the similarity between four musical artists. A synonym for Charles Mingus is included.

social media platforms, which offer the possibility of sharing favoured music tracks.

The first dataset of the described system comprised of crawled Twitter messages that contained the hashtag # nowplaying. The second one included that hashtag # iTunes. The text retrieved was processed and matched against common patterns such as "songtitle by artistname", "artistname-songtitle", "#artistname". The potential artist strings were also matched against a list of publicly available, known, artist names.

The experimental section was based on the evaluation of four co-occurrence analysis measures which yielded item-to-item similarity:

1.
$$sim(i,j) = \frac{x(i,j)}{occ(i)} \tag{3.22}$$

2.
$$sim(i,j) = \frac{x(i,j)}{occ(i)}(1 - \frac{|occ(i) - occ(j)|}{max_k occ(k)}) \tag{3.23}$$

3.
$$sim(i,j) = \frac{x(i,j)}{min(occ(i), occ(j))}(1 - \frac{|occ(i) - occ(j)|}{max_k occ(k)}) \tag{3.24}$$

4.
$$sim(i,j) = \frac{x(i,j)}{\sqrt{(occ(i) \times occ(j)}} \tag{3.25}$$

where $x(i,j)$ the co-occurance count between artists i and j, and $occ(i)$ the number of occurances of artist i inside the Twitter-text set. The resulting rank of similar artists for each of the aforementioned methods, was compared to the Last.fm top-ranked similar artists (ground truth). The R-precision measure showed that similarity measures 2 and 4 performed the best.

An interesting part of Schedl's paper investigates geospatial patterns, meaning listening patterns with regard to geographical information. However since this topic exceeds the scope of this report, we will forward the reader to the original publication.

3.3.6 Combining Social Music and Semantic Web for Music-Related Recommender Systems

In our discussion regarding interoperability (see Chapter 2) we thoroughly explained Semantic Web and its relevant notions, such as RDF, URIs etc. We will now see how the semantic web has been exploited in conjuction with social music for music recommendation, by Passant and Raimond [10]. It should be noted that the term "social music" was coined by Last.fm to describe the act of sharing musical tastes, but can be generalized to any platforms rather than Last.fm (eg. blogs, microblogs etc.)

Passant's and Raimond's approach makes use of the large amount of semantic connections, mostly encoded via the FOAF vocabulary, between MySpace, MusicBrainz, DBpedia and Last.fm. Also their system exploits SIOC's inherited functionality that allows new types of user generated data types to be created based on other domain-specfic ontologies.

Their first described method uses social networks and the foaf:topic_interest term that provide a direct link between someone and his interests such as a band or artist. The music ontology MO can be also used to represent someone listening habits as done by Last.fm. A simple example is presented in Figure 3.4.

The second proposed approach uses the MOAT framework which allows people to tag content with URIs rather than simple words. The relationships between URI tags can later be used recommend related items. For example if a user browses a web page pertaining Michael Jackson, the system can provide direct link to a flickr page with his former band "Jackson 5".

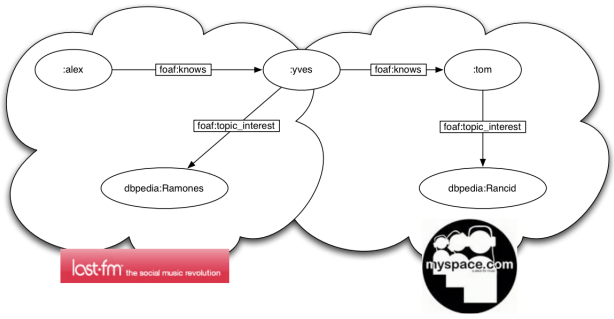

Figure 3.4: Using distributed social-networks in recommendation systems.

The paper describing the aforementioned methods doesn't include any evaluation or experiments. However its importance, as far as this report is concerned, relies on the employment of the semantic web and on the exploitation of its capabilities.

3.3.7 Taking Advantage of Editorial Metadata to Recommend Music

One of the latest approaches for music recommendation using community metadata is proposed by Dmitry Bogdanov and Perfecto Herrera [9]. Their imple-

mentation is based on Discogs.com, a music database that contains information about artists, labels and recordings.

As an initial step, a user profile is generated from a set of provided music tracks and their corresponding metadata in Discogs. Latent semantic analysis is employed to compact the user profile representation into a single vector. The same happens for all artists appearing in Discogs allowing similarity/distance measures to be calculated.

More specifically a tag cloud is initially created for each artist. It contains information such as genre, style, label, country and year of release. Three list of releases are retrieved: "main" where the artist appears as heading the release,"track" where the artist is just credited in the release (e.g. compilation, guest) and "extra" where the artist is mentioned in the record credits. The tag-clouds of each release list are later merged using a weighting function that favors "main" releases. Tags are also propagated using artist relations found on the database. An example of this procedure is shown in Figure 3.5.

The proposed method touches a topic that hasn't been investigated before, namely determining a records's authentic epoch. Authentic epoch, according to the authors, represents the years when the music was firstly recorded, produced and consumed. This task is handled by finding the release with the earliest date and propagate that date with weights as follows:

$$W_{y\pm i} = W_y * 0.75^i, i \in \{1, 2, 3, 4, 5\} \tag{3.26}$$

The evaluation of the whole system is based on questionnaires of 27 voluntary subjects, with the expectation that the data could generalize for a wider population. The proposed approach is compared against methods based on Last.fm tags, black-box similarity by iTunes Genius and content based semantic similarity refined by genre metadata. The results show that the first outperforms the latter, however due to the nature of the evaluation the exact numbers and details will be omitted.

3.4 Conclusion

In the previous paragraphs we described the notion of MIR data enrichment and identified its connection to the MIR information extraction and Web MIR. A series of relevant techniques were presented along with published approaches.

It is our belief that not a single approach can be considered optimal. Depending on the context, a method's efficiency and relevance may vary. However a common set of tools exist, such as querying search engines, indexing, co-occurrence analysis etc. This set can be modified, or even extended, to fit the needs of other tasks rather that artist similarity or genre classification.

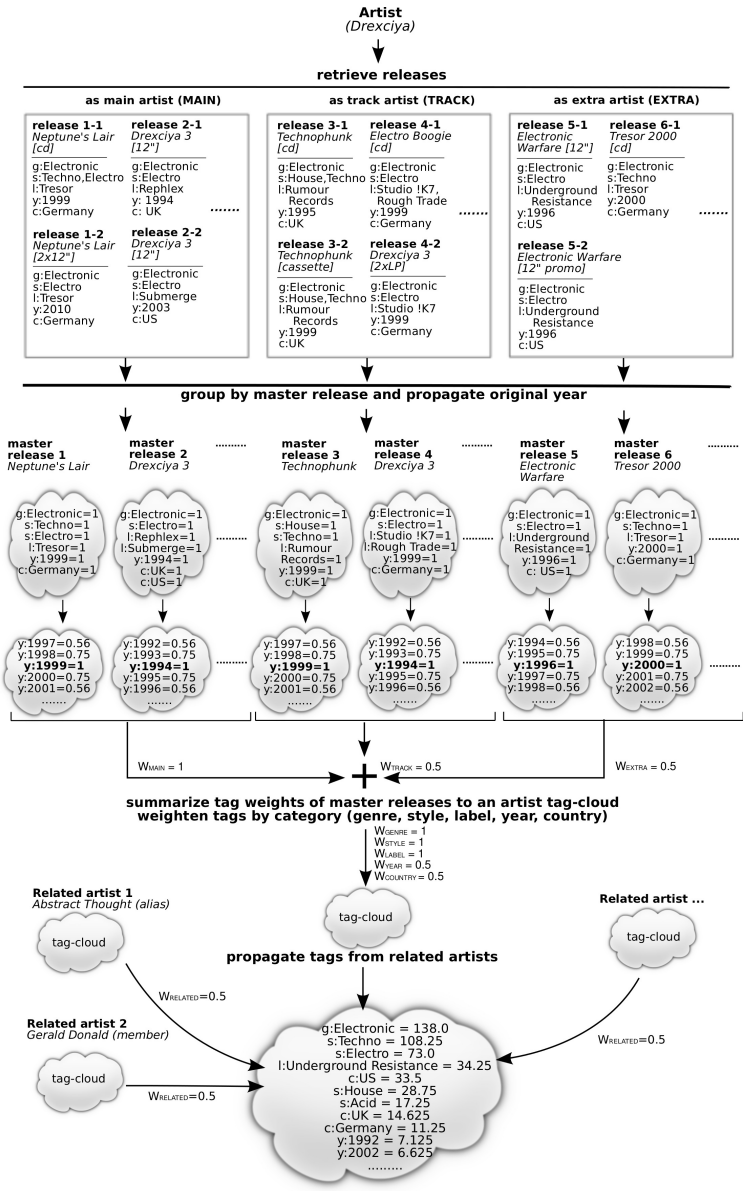

Figure 3.5: An example of the proposed artist annotation based on editorial metadata from Discogs. Three lists of releases (MAIN, TRACK, EXTRA) are retrieved according to an artist's role. Particular releases are summarized into master releases, merging all found genre, style, label, and country tags, and selecting and propagating original year.

Bibliography

[1] B. Whitman, S. Lawrence, "Inferring Descriptions and Similarity for Music from Community Metadata", Proceedings of the 2002 International Computer Music Conference (ICMC). 2002.

[2] O. Celma, P. Cano, P. and Herrera, " SearchSounds: An Audio Crawler Focused on Weblogs", Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR). 2006.

[3] M. Scheld, "Automatically Extracting, Analyzing, and Visualizing Information on Music Artists from the World Wide Web", PhD Thesis. 2008.

[4] C. McKay "Automatic Music Classification with jMIR", PhD Thesis. 2010.

[5] J.J. Aucouturier, F. Pachet, "Representing musical genre: A state of the art", Journal of New Music Research 32 (1). 2003.

[6] S. Govaerts, E. Duval, "A Web-based approach to determine the origin of an artist", Proceedings of 10th International Society for Music Information Retrieval (ISMIR). 2009.

[7] M. Schedl, "Country of origin determination via web mining techiniques", IEEE Proceedingd of the International Conference on Multimedia and Expo (ICME). 2010.

[8] S. Govaerts, N. Corthaut, E. Duval, "Using Search Engine for Classification: Does It Still Work?", Proceedings of the 11th IEEE International Symposium on Multimedia (ISM). 2009.

[9] D. Bogdanov, P. Herrera, "Taking advantage of editorial metadata to recommend music", Proceedings of the 9th International Symposium on Computer Music Modeling and Retrieval (CMMR). 2012.

[10] A. Passant, Y. Raimond, "Combining Social Music and Semantic Web for music-related recommender systems", Proceedings of the Social Data on the Web Workshop (SDoW). 2008.

[11] M. Scheld, D. Hauger, "Mining microblogs to infer music artist similarity and cultural listening patterns", Proceedings of the 21st international conference companion on World Wide Web (WWW). 2012.

[12] B. Whitman, P. Smaragdis, "Combining Musical and cultural Features for Intelligent Style Detection", Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR). 2002.

[13] S. Baumann, O. Hummel, "Using cultural metadata for artist recommendations", In Proceedings of 1st International Conference on Web Delivering of Music (WEDELMUSIC). 2003.

[14] G. Geleijnse, J. Korst, "Web-based Artist Categorization", Proceedings of the Seventh International Conference on Music Information Retrieval (ISMIR). 2006.

[15] Wisdom of the crowds, Wikipedia entry, en.wikipedia.org/wiki/The_Wisdom_of_Crowds

[16] P. Knees, E. Pampalk, G. Widmer, "Automatic Classification of Musical Artists based on Web-Data". OGAI Journal. 2005.

[17] F. Pachet, G. Westerman, D. Laigre, "Musical Data Mining for Electronic Music Distribution", Proceedings of the 1st International Conference on Web Delivering of Music (WEDELMUSIC). 2001.

[18] M. Zadel, I. Fujinaga, " Web Services for Music Information Retrieval", Proceedings of the 5th International Symposium on Music Information Retrieval (ISMIR). 2004.

[19] G. Geleijnse, J. Korst," Web-based Artist Categorization", Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR). 2006.

[20] P. Knees, M. Schedl, T. Pohle, G. Widmer, "An innovative three dimensional user interface for exploring music collections enriched with meta-information from the web", Proceedings of the 14th annual ACM international conference on Multimedia. 2006

[21] W. W. Cohen, W. Fan, "Web-Collaborative Filtering: Recommending Music by Crawling The Web", Computer Networks, Volume 33, Issues 1-6. 2000.

[22] D. P. W. Ellis, B. Whitman, A. Berenzweig, S. Lawrence, " The Quest for Ground Truth in Musical Artist Similarity", Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR). 2002.

Chapter 4

Interoperability: Record Linkage

Two separate and structurally different databases exist inside the COGITCH domain (see section 1.1). Although both of them represent and store semantically the same concepts, syntactic heterogeneities, typos and other factors disallow precise interlinking between musical entities. This chapter presents a simple yet efficient method for locating artist entities inside noisy strings and then linking them to MusicBrainz. Our approach, which relies on the employment of string permutations and metaphones [5], shows great performance and room for further improvement.

4.1 Introduction

The Sound and Vision (S&V) database, although large in terms of number of records, presents various flaws. First, it lacks significant musical information such as genre and year of release; issues to be addressed in the later parts of this thesis. Secondly and most importantly, it stores artist information about a song in a complex string architecture. Although guide rules have been provided to the data loggers, these were not explicitly followed for various reasons. For example, in some cases information about the main artist of a song may not have been clear. It is also possible that the guidelines did not take into consideration special cases, thus the logger might had to improvise when inserting information. When taking into account the existence of typos, it becomes clear that linking each "Artist" field in S&V to distinct artist entities, constitutes a major problem.

But why are we interested in solving this problem? First, consider data enriching functions which can only be applicable on clearly defined music entities such as "Willy Derby"; and not on cases such as"Willy Derby (singer) and the Amsterdam Orchestra". The latter string contains information unrelated to the main artist, and thus any data harvesting techniques would fail to retrieve correct and complete information about the artist. Secondly, consider the case of searching for all songs by Willy Derby. A full-text search optimally would suffice, however for an "Artist" field containing typos and noisy additions, the performance would drop. Finally, linking records within and between databases

is a crucial procedure that requires each record to point to a well defined entity.

4.1.1 Problem Definition and Assumptions

Following Neiling's work [4] the question we are trying to answer is as such: given two overlapping data sets A and B, on the same universe of object, are there any records $a \in A$ that belong to any record $b \in B$? According to [4] the first step for answering this question is object identification. In other words prior to any linkage, the objects of interest should be well-defined. This is typically achieved via cleansing of the data, conversion functions, employment of contextual information and others.

MusicBrainz has well-defined objects and therefore we won't be dealing with that source. However, this is not the case for S&V , since by manually inspecting its contents, we have identified the following cases:

1. Typos e.g. "Willie Derby" instead of "Willy Derby".
2. Name permutations e.g. "Derby Willy" instead of "Willy Derby".
3. Noise additions e.g. "Willy Derby (singer) and the Orchestra of Amsterdam".
4. Name abbreviations e.g "W. Derby" instead of "Willy Derby".
5. Name variations e.g. "Beatles" instead of "The Beatles".
6. Other e.g. "Bob Scholte plays Willy Derby".

Given this specific list of special cases, the problem is transformed into finding a set of functions (conversions, comparisons, etc.) that convert each "Artist" attribute to clearly-defined artist-object.

Before explaining in detail our methodology it is important to discuss two major assumptions that alleviate the difficulty of the whole problem:

1. The main artist appears always first in the "Artist" field. This implies in that cases such as "Orchestra of London conducted by Derby", the main artist is "Orchestra of London".
2. The three first words of an artist's name constitute an adequate identifier. In other words, we assume that there are no two artists with their three first words (in their name) exactly the same. This of course does not hold in cases such as "The band of A" and "The band of B", but this is a sacrifice we are willing to make for the sake of computational efficiency.

4.2 System Description

Our method aims at addressing each and every one of the issues presented in 4.1.1 while at the same time taking into consideration the two major assumptions. For each string of characters S, corresponding to an entry at the "Artist" name field, extracting a unique name-identifier S^* and linking it to MusicBrainz is achieved via a five step function:

Object identification:

1. Locating the main artist.
2. Calculating possible permutations and metaphones.
3. Calculating name abbreviations.

Record linkage:

1. Internal record linkage.
2. Linkage to MusicBrainz.

The first three steps aim at a) locating the main artist's name and b) computing all its possible appearances in the database. The fourth one ensures that any duplicates will be deleted, or will be pointing to the same unique artist name S^*. The final step links any possible unique artist name to a unique MusicBrainz profile.

4.2.1 Locating the Main Artist

By manually inspecting the database we have discovered that the main S string can incorporate the main artist in many ways. Some examples are shown below (the main artist in Italic font):

- *Shepherds, The.*
- *Polland, Pamela* [zang]
- *Hoogoven Harmoni* (Orkest), Steyn, Willem
- *Clark, Gus* [electr. orgel]
- *Life Guards* olv. W. Jackson
- *Shines, Johnny* [zang + gitaar]

Therefore the main artist can be either a band, a person or an orchestra. The role of each entity can vary from being the singer, the guitar player, the main violinist and many more. As a consequence, locating the main artist is a complicated task.

Our method starts by assuming that the singer tag e.g."[zang]" is present and therefore the main artist is a singer. After locating such a tag, the substring on the left would correspond to the main artist, since the singer typically appears first. However, the "[zang]" tag appears in many versions throughout the database; for example as "[zanger]","(zang)","(singer)" etc. Taking into account all the possible versions, our method discards the right substring that corresponds to noise.

This process is successful only when the singer tag is present. In the opposite case, the S string corresponds to either a band e.g. "Shepherds, The" or a sequence of music entities such as "Bulterman, Jack met zijn (Orkest)" and "Mariachi Vargas de Tecalitlan, Fuentes, Ruben". Locating the main artist now, is a matter of segmenting the string at the optimal position. Our manual inspections yielded a set of tags and words that constitute important segmentation points. These are presented below:

```
"gespr  tekst","gitaar", "tenor","a/h","gespr.","Electr.","gespr","citer",
"bas","kwintet","Kwintet","choir","ensemble","Ensemble","kapel",
"Kapel","combo","Combo","kwinter","Kwintet","barrelpiano","e/s",
"Orkest","sextet","Sextet","septet","Septet","Draaiorgel","Orkest",
"orkest","orkesten","Orkesten","accordeon","Accordeon","duo",
"Duo","Kwartet","kwartet","duo", "Trio","trio","Orgel","electr. orgel",
"orgel", "cello","Cello","piano","sopraan","drums","saxofoon",
 "band","Band","koor","Koor","contrabass","trompet", "olv",
 "cineamorgel","y/s","bariton","clarinet","the","The","met"
```

To avoid segmenting in between an artist's name (that includes any of the tags), we segment the string only when the tags are in between white spaces, "( )", "[ ]" and combinations of those. The tag "the" is a special case and does not correspond to a segmentation point. However, it is removed when not in between white spaces. For example, in the cases of "The Beatles" and "Beatles the" it will be discarded but not in the case of "Nick Cave and the Bad Seeds".

The final step of the process employs the second assumption (see section 4.1.1). Each resulting name string containing more than three words is trimmed to three. The purpose of such function is currently not apparent, but will be discussed thoroughly later. For the time being, it is worth presenting some examples of resulting strings after the end of the whole procedure.

- Davidson, Harry met zijn (Orkest) → Davidson Harry
- Hammond Beat Boys, The → Hammond Beat Boys
- Beefheart, Captain a/h Magic (Band) → Beefheart Captain
- Edoardo, Vianello [zang] Morricone, Ennio met zijn [orkest] [Cantori] Moderni → Edoardo Vianello

4.2.2 Computing Permutations and Metaphones

Previous process addresses the issues of locating the main artist by filtering noise in strings and by minimizing name variations of the type "Beatles The", "The Beatles". The process to be discussed deals with typos and random-order, name appearances of the type "Willy Derby", "Derby Willy".

As we have seen, each artist is currently represented by a three-word string at maximum. Out method computes all possible word permutations for each string. Therefore for a two-word name the permutations are two, while for a three-word name the permutations are six. This amount increases exponentially with regard to the words pertained in the name. Now the rational behind our choice of limiting the words to three becomes obvious.

A list of example names and their permutations is shown below:

- Schaik Frans v. → Schaik Frans v.;Schaik v. Frans;Frans Schaik v.;Frans v. Schaik;v. Schaik Frans;v. Frans Schaik
- Jordaan Johnny → Jordaan Johnny;Johnny Jordaan

Computing the permutations covers a large number of the possible appearances of an artist name in the database. However, when using exact-matching string comparisons, typos are a major drawback. Out method assumes that most

frequent typos are of the type "Willie", "Wily" instead of "Willy" or"Peyne" instead of "Payne" or even "Jordan" instead of "Jordaan". In other words our method assumes that typos do not introduce any phonetic variations in the name, therefore the pronunciation is almost identical.

Based on this assumption, our method computes the metaphone [1] correspondence of each permutation. Metaphones, developed by Lawrence Philips, are string keys that are the same for similar sounding words. In other words, names pronounced similarly should produce the same metaphone key. Examples are shown below:

- Low Bruce → LBRS
- Woodhouse John → WTHSJN
- Derby Willy → TRBWL

Eventually each artist name string S is represented by a set of permutations P^S a set of metaphones M^S and a unique name identity I^S. For S ="Willy Derby", P^S = {Derby Willy,Willy Derby}, M^S = {TRBWL;WLTRB} and I^S ="Derby Willy" which simply corresponds to the first permutation.

4.2.3 Computing Name Abbreviations

The procedure to be discussed deals with cases of the form "W. Derby" instead of "Willy Derby", which are quite common in the database. We are not sure if such trimmed name versions are the result of the loggers' mistakes or if the artists themselves were credited in such way. The fact still remains; even after computing all possible permutations and metaphones, "W. Derby" and "Willy Derby" would be still considered different entities.

In an ideal situation, where first names appear before last ones, converting a name string to its abbreviated form would be easy. However, we know that first-last name ordering does not follow any rules. In addition, not all artists are persons. Bands and orchestras also appear in the database and thus converting them would be futile.

Our method makes use of two large sets of possible, English, male and female first names[1] [2]. For each permutation in P^S, the method matches its first word to both name-lists. In case of a match an extra permutation is added to P^S, corresponding to the abbreviated form. For example, for "Willy Derby" our approach would return "W. Derby", since "Willy" appears in the name-lists.

However, there's a downside to that approach. Some artists have last names that can be misinterpreted as first names e.g. "Bob Dylan". Now, imagine that "Dylan Bob" appears as the main artist; then the abbreviated form would be "D. Bob". The confusion and errors that may be introduced are obvious. It is our belief that solving such problem exceeds the scope of this study. We have identified potential solutions, such as using the artist detection function of EchoNest, but we will not pursue them due to time constraints. Our simple solving methodology relies on the observation that artist names are usually

[1] www.infochimps.com/datasets/word-list-3800-common-male-given-names-english-speaking-countrie

[2] www.infochimps.com/datasets/word-list-4900-common-female-given-names-english-speaking-countr

Permutations	Metaphones
Willy Derby$_{id}$	WLTRB
Derby Willy	TRBWL
W. Derby	
D. Willy	

Table 4.1: Permutations and Metaphones for the artist "Willy Derby"

Permutations	Metaphones
Shepherds$_{id}$	XFRTS

Table 4.2: Permutations and Metaphones for the artist "The Shepherds"

inserted in the form "Last name, First name". Given that, we assume that "Bob Dylan" appears significantly less number of times than "Dylan Bob" and therefore any errors will be limited.

The upside of our approach relies on the relation between names and genders. Most names can be employed to directly identify the gender of an artist. This piece of information might be irrelevant to our general goal, but enhances the knowledge we have about the artist in hand.

4.2.4 Internal Record Linkage

The first step in linking "Artist" name fields to unique artist entities is to build a database/index for the latter. In other words, before any linking, the actual target entities must be present. In our case, such source of unique musical entities is either non-existent or insufficient. Therefore we had to build one from the scratch.

Our method sequentially processes each S&V record and adds the corresponding main artist to a unique-artist index Ind in case it hasn't appeared before. The latter is achieved via the employment of permutations, metaphones and name abbreviations.

It is worth describing the process along with a simple example. For each "Artist" name S our method first locates the main artist and then computes a representation comprising of a set of permutations (including the name abbreviation) P_{+}^{S}, a set of metaphones M^S and a unique name id I^S. Let us assume that the first S that appears in the database is "Willy Derby met Lou Bandy [accordeon]". The corresponding representation is shown in Table 4.1.

Unfortunately, both names "Willy" and "Derby" are first names, therefore the incorrect abbreviated version "D. Willy" appears. But we will see later how this affects the matching procedure. Now, since the index Ind is empty, the system adds to it the I^S accompanied by the permutations and metaphones.

Let us assume that the next record in the database contains the "Artist" name S' ="The Shepherds (band)". Its corresponding representation is shown in Table 4.2.

Now, since the index Ind is not empty, the system computes the similarity between the permutations and metaphones. For each permutations p' in $P^{S'}$ and p in P^S, a string similarity function $s(p, p')$ is applied. In our case, for each

Permutations	Metaphones
Derby Willie$_{id}$	TRBWL
Willie Derby	WLTRB
W. Derby	
D. Willie	

Table 4.3: Permutations and Metaphones for the artist "Derby Willie"

of the "Willy Derby", "Derby Willy", "W. Derby" and "D. Willy" the similarity between "Shepherds" is calculated. The algorithm corresponds to the PHP's *similar_text()* function [1]. If the similarity is above 0.96 (with 1 being exact match), our method assumes that the strings are the same and therefore also the same musical entities.

In our case, none of the permutations of P^S yield similarity higher than the threshold. Therefore, our method gives those entities another chance, assuming that the low similarity is a result of typos. The same procedure is now applied on the corresponding metaphones, but with higher threshold of 1; since metaphones are high level abstractions already. In our case, none of the metahophones in M^S are exact matches of $M^{S'}$ and therefore the system concludes that the entities are not the same. The index Ind will now incorporate the entity "Shepherds".

Let us now assume that the third record in S&V contains S'' ="Derby Willie [zang]", with a repesenation as in Table 4.3.

Given the previous process, the entity "Derby Willie" will be matched to "Willy Derby" and therefore it will not be added to the index Ind.

A problem will arise only when an artist string of the form S''' ="Donald Willy" appears. Since "Donald" is a first name also, an abbreviated version "D. Willy" will emerge. As a consequence our method will eventually assert that "Willy Derby" and "Donald Willy" are the same entities, and the latter will not be included in Ind.

It is also worth pointing out that if a typo-version of an artist name appeared before the correct one, then that would be the one stored in the index. For example, if "Derby Willie" appeared before "Willy Derby" then this artist would be represented by the first typo-including version.

4.2.5 Linkage to MusicBrainz

The methods presented above optimally result to an index of unique artist names as they appear in the S&V database. By assuming that the "Artist" field in the Meertens database does not contain any noise, a linkage between the two collections is easily achievable. Although this is not the case, we are interested in a greater problem, namely generalization. Similar to crosswalks (see 2.3.3) for more than two datasets the work overload of record linkage increases exponentially. Therefore, it is wiser to link each artist to a well established, large identifier hub such as MusicBrainz.

Given the precomputed index of unique artists names inside S&V we aim at correctly matching each one of them to a MusicBrainz profile. However, given the possible ambiguities, trusting MusicBrainz for matching each query to the correct artist is ill-advised. MusicBrainz returns a ranked-list XML response

for artist-search query as shown in Figure 4.1. We make use of three pieces of information to link each query to the correct artist in the returned list:

- Relevancy score (see attribute $< ext : score\rangle$) which is computed based on a MusicBrainz's unknown internal function. It should be noted that the top ranked artist is always assigned the maximum value of 1.
- Name (field $< name\rangle$) which corresponds to the artist's name (e.g Don Payne).
- Sort name (field $< sort-name\rangle$) which corresponds to a different ordering of the name (e.g. Payne Don).
- Aliases (field $< alias - list\rangle$ which corresponds different name versions such as abbreviations (e.g. Donald Payne, D. Payne).

Our approach assigns a similarity value between the query and each of the returned artists based on the following formula:

$$sim(q, a_i) = \frac{score_{a_i} + s_1(q, a_i) + s_2(q, a_i)}{3} \tag{4.1}$$

where $s_1(q, a_i)$ the maximum of *similar_text()* similarities between the query's permutations and a_i's name, permutations, sort name and aliases. $s_2(q, a_i)$ denotes the maximum of similarities between the query's and a_i's metaphones, while $score_{a_i}$ the returned MusicBrainz relevancy score.

Our preliminary experiments have shown that the method works really well, except cases where synonymity appears. For example when trying to match "Ken Griffin", MusicBrainz returns a ranked-list where the two top artists are "Ken Griffin" but correspond to different persons. In such a case, our method will pick the first occurrence as the correct match. This example suggests that a more sophisticated algorithm should be employed, probably similar to Raimond's [3]. However such an approach would require additional information to be employed, such as discography.

4.3 Experiments and evaluation

Precise evaluation of our method is not feasible, considering the size of the S&V database. In order to acquire a ground truth, we have to either manually inspect all records or employ the "perfect" algorithm. This "chicken-egg" situation leaves us with no choice but to employ manual labour. Therefore, the idea behind our evaluation is to query for artist names that we know they appear in the database, and manually count the number of true and false positives returned.

We consider as baseline, MySQL's internal, exact matching function. The method that yields the highest positive predictive value ($TP/(TP+FP)$), would be considered the best.

4.3.1 Experimental Setup

For our method, denoted from now on as "PMA" (Permuations Metaphones Abbreviations), given a set of 10 unique artist names, we gather all the matched

```
<metadata xmlns="http://musicbrainz.org/ns/mmd-2.0#" xmlns:ext="http://musicbrainz.org/ns/ext#-2.0"
 <artist-list count="1362" offset="0">
  <artist id="3a2bf254-5a6a-43c9-92ca-32b5ef99850c" type="Person" ext:score="100">
    <name>Jack Payne</name>
    <sort-name>Payne, Jack</sort-name>
    <country>GB</country>
   ▶<life-span>...</life-span>
   ▼<alias-list>
      <alias>J Payne</alias>
    </alias-list>
  </artist>
 ▶<artist id="67262c04-6cc1-4c35-b9a3-cd4a54705cef" ext:score="52">...</artist>
  <artist id="248aeb53-5371-4225-bddf-abc1ffea3948" type="Group" ext:score="44">
    <name>Jack Payne and His Dance Orchestra</name>
    <sort-name>Payne, Jack, and His Dance Orchestra</sort-name>
   ▶<life-span>...</life-span>
  </artist>
  <artist id="a9743cb4-dbea-4f36-9b28-3af46970193b" type="Group" ext:score="44">
    <name>Jack Payne & His BBC Dance Orchestra</name>
    <sort-name>Payne, Jack & His BBC Dance Orchestra</sort-name>
   ▶<life-span>...</life-span>
   ▶<tag-list>...</tag-list>
  </artist>
  <artist id="d8a7ce2d-1a1e-4ea9-953e-10a4a7f4f930" type="Person" ext:score="37">
    <name>Don Payne</name>
    <sort-name>Payne, Don</sort-name>
    <country>US</country>
    <disambiguation>Bass player</disambiguation>
   ▶<life-span>...</life-span>
   ▼<alias-list>
      <alias>Donald Payne</alias>
      <alias>D Payne</alias>
    </alias-list>
  </artist>
  <artist id="a719dafb-b59a-49ef-85c8-a0caffba63b2" type="Person" ext:score="37">
    <name>Joseph Payne</name>
    <sort-name>Payne, Joseph</sort-name>
    <gender>male</gender>
    <country>GB</country>
   ▼<life-span>
      <begin>1941</begin>
      <ended>false</ended>
    </life-span>
   ▼<alias-list>
      <alias>Joseph Payne [h'chd]</alias>
      <alias>J Payne</alias>
    </alias-list>
  </artist>
```


Figure 4.1: An XML sample response for the search term "Jack Payne".

records in the database. However, for MySQL's simple matching function, denoted "MySQL_1", the query terms to be used are really important. For example, searching for "The Beatles" instead of "Beatles" significantly affects the number and type of returned records. Therefore, in order to be fair, our evaluation employs queries of the form "$word^1$ AND $word^2$ .. AND $word^n$", where $word^i$ appears inside the artist's name.

Two experimental configurations are investigated. In the first one, denoted PMA^-, our method does not take into consideration abbreviated name versions, as opposed to second one denoted PMA^+.

4.3.2 Results

The positive predictive value accompanied by the number of returned records for the first ten artists are presented in Table 4.4.

It is worth examining certain thought-provoking cases, starting with "Jack Payne". Our method returns a list of all true-positive records. MySQL's internal function though yields a prediction value of 50%. If we examine the results in detail we observe that MySQL returns also records in which "Jack Payne" is not listed as the main artist (eg. conductor). Since our initial assumption is that the main artist is the one listed first, the low positive predictive value of MySQL makes sense. In cases where the user is interested in all records pertaining the name "Jack Payne", MySQL is more reliable.

Artist	PMA^-	**MySQL_1**	PMA^- **count**	**MySQL_1 count**
Jack Payne	100%	50%	23	36
Jo Vincent	100%	73%	11	15
Willy Derby	99.1%	98.3%	121	119
Duo Hofmann	100%	95.3%	40	43
Louis Noiret	100%	63.3%	11	30
August de Laat	100%	100%	10	9
Kees Pruis	100%	100%	54	53
Lou Bandy	82%	98.1%	62	54
Josef Marais	100%	61%	21	54
William Kimber	100%	100%	8	8

Table 4.4: Positive predictive value and counts for PMA^- and MySQL_1.

Artist	PMA^+	**MySQL_1**	PMA^+ **count**	**MySQL_1 count**
Jack Payne	88%	50%	26	36
Jo Vincent	100%	73%	11	15
Willy Derby	99.1%	98.3%	121	119
Duo Hofmann	100%	95.3%	40	43
Louis Noiret	100%	63.3%	11	30
August de Laat	100%	100%	10	9
Kees Pruis	100%	100%	54	53
Lou Bandy	82%	98.1%	62	54
Josef Marais	100%	61%	21	54
William Kimber	100%	100%	8	8

Table 4.5: Positive predictive value and counts for PMA^+ and MySQL_1.

The second worth-examining case is "Lou Bandy", since our method's predictive value is lower than MySQL's. The reason for this behaviour relies on the fact that "Lou Bennet" yields the same metaphone keys as "Lou Bandy", therefore they are erroneously considered the same artist. This fact suggests that metaphones are too high abstractions to be used as artist name features. We believe that a combination of metaphones and edit distance might work more sufficiently; which remains to be investigated in future work.

Overall, PMA^- yields a larger ratio of true positives (close to 100%) by including typo versions of artist names and by excluding non-main artist listings. MySQL fails to discover typo-versions but can present records where the artist is listed as secondary.

Now let us examine the performance of PMA^+ which takes into consideration abbreviated forms (Table 4.5). It should be noted that PMA^+ enforces abbreviations whenever possible. For example, if "Willy Derby" is already in the index and "Willeke Derby" appears in one of the database's records, then the abbreviated version of the latter will be computed.

By studying Table 4.5 above, it becomes clear that only "Jack Payne" presents different performance from PMA^-. The reason for this behaviour is quite clear; there are no abbreviated forms present in the database for the rest

of the artists. Given this fact, it is wise to examine the "Jack Payne" case more thoroughly. PMA^+ returns three more records that PMA^- which correspond to the artist "Jimmy Payne". The matching occurred between the abbreviated forms "J. Payne". If the framework hadn't enforced the abbreviated form then this mismatch wouldn't had occurred. Therefore, only if the actual artist name was "J. Payne" a link would have been established (see Figure 4.2).

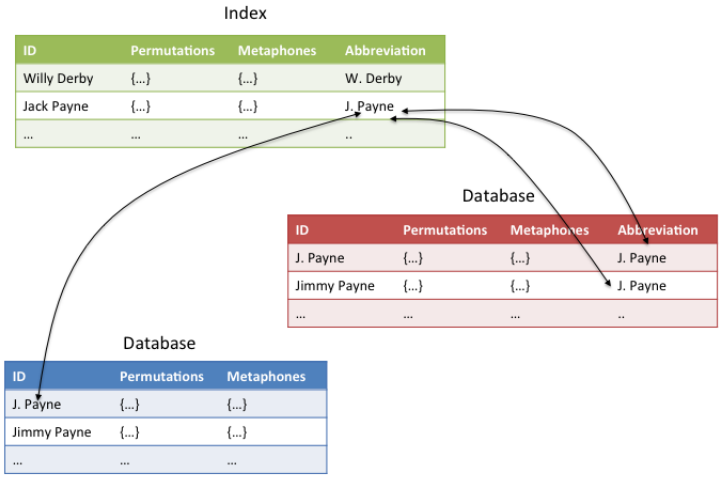

Figure 4.2: Two different cases of enforced and non-enforced abbreviations (red and blue respectively). In the first case a mismatch occurs for "Jimmy Payne". In the second case the mismatch is avoided. In both cases, "J. Payne" is linked to "Jack Payne" in the index, although the abbreviated form "J." is ambiguous.

4.3.3 Discussion and Conclusions

This chapter's aim has been to investigate record linkage between databases, via main-artist-name entity detection. We described certain approaches that employ name permutations, metaphones and naming abbreviations. The results show that a certain amount of compromise is always required; it is extremely difficult to embrace all possible name variations while at the same time, limit ambiguities. However, our method(s) show better performance than MySQL's exact-matching function, as the first is well-tailored to the problem.

Future work should aim at limiting the effect of metaphone and abbreviated forms matching. It is possible for example, to accept a metaphone match only when the edit distance is higher than a certain threshold. Such a scheme would reject linkage between "Lou Bennet" and "Lou Bandy", although determining the threshold might be a complex procedure. Regarding the abbreviated forms, our results imply that non-forced abbreviation works better.

To summarize, even though our method presents room for improvement, the current framework can be successfully applied on the S&V database.

Bibliography

[1] A. Binstock, J.Rex, “Practical Algorithms for Programmers”, Addison Wesley. 1995.

[2] I. Oliver, “Programming Classics: Implementing the World’s Best Algorithms”, 1994.

[3] Y. Raimond, C. Sutton, M. Sandler, “Automatic Interlinking of Music Datasets on the Semantic Web”, Proceeding of the Linked Data on the Web Workshop (LDOW). 2008

[4] M. Neiling, H. J. Lenz, “Data fusion and object identification”, Proceedings of International Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet. 2000.

[5] L. Philips, “Hanging on the Metaphone ”, Computer Language, Vol. 7, No. 12, 1990.

Chapter 5

Data Enrichment: Placing Music Entities in Time

5.1 Introduction

The productive period of a music artist is an important piece of information that is typically highly correlated to his style, influences and similarities to other artists. As such, the productive years constitute a reliable, additional feature for artist-recommendation, similarity calculation, genre classification, etc. A song's original year of release is also extremely valuable music-metadata for users, archivists and musicologists. Both allow placing musical information into a time context, thereby enhancing the semantic quality of that information.

This chapter investigates the novel task of situating music artists and songs in time. The proposed method exploits editorial metadata in conjunction with web mining techniques, in order to determine an artist's productivity over time and estimate the original year of release of a song. Evaluation on a test set of Dutch music shows that the proposed methods are robust and present reliable performance. In addition, the methods offers room for generalization to non-music domains.

5.1.1 Problem definition

We define as a song's *year of release (YoR)*, the year on which it was firstly released in either an album, single, etc. If the song was released in both an album and a single, then it is the earliest release that corresponds to the YoR. (check Dmitry definition).

We define as an artist's *years of productivity (AYoP)*, the years in which the artists was alive, recording and releasing albums, singles, etc. In other words, a compilation release after the death of an artist doesn't contribute to his AYoP. By default we assume that re-releases of songs/albums, while the artist was alive, also do not contribute to the AYoP.

Formulating YoR estimation

Given a tuple of the form $< ArtistName, SongTitle, YoR^* \rangle$ where YoR^* the true year of release, we want to find a function $f(ArtistName, Songtitle) = YoR^{est}$, such that it minimizes $|YoR^{est} - YoR^*|$.

Formulating Artist Years of Productivity estimation

We denote $AYoP^A = \{b_1, b_2, ..., b_{130}\}$ the probability density function (pdf) of an Artist A, where b_i the value of the i^{th} histogram bin corresponding to the productivity of the year $1880 + i$. For example, an artist with high productivity during the 70's will have a AYoP distribution that peaks between $i = 90$ and $i = 100$. We want to find a function $f(A) = AYoP^{est}$ that minimizes the distance $d(AYoP^A, AYoP^{est})$.

5.2 System Description

The YoR and AYoP estimation functions of the system are based on the same set of tools. However, as we will see later, AYoP is a prerequisite for YoR and therefore we will examine it first. Before any of these, we shall discuss the system's main sources of information, from which meaningful data is mined.

5.2.1 Sources

Our system exploits two distinct sources: Music Information Services (MIS) and Search Engines. MIS correspond to (semi) commercial and non-academic online systems and services that utilize or provide technologies related to music databases. These vary from automatic music classification and recommendation to metadata about artist discographies and information. The latter is usually denoted "Editorial Metadata", a convention that this paper will follow from now on. Many companies and initiatives have implemented systems associated with various aspects of music collections, however our approach employs those that are well-established or have been previously employed by MIR researches. These are Last.FM[1], MusicBrainz[2] and EchoNest[3].

Last.FM

Last.FM can be considered a high-profile Internet radio station that encapsulates a music recommendation engine (AudioScrobbler). Last.FM has been exploited by many previous researches since it has been particularly proactive and helpful in providing access to their data through a powerful web API.

Figure 4.1 shows a sample response for the API method *album.getInfo*. Although information about the release date of the album and the included tracks is provided, release type information (e.g. single, album, compilation etc.) is absent. Therefore, in our context of use, Last.FM is unable of providing reliable release dates. If a certain song is matched against the Last.FM database, it is uncertain whether this corresponds to the original or some later release.

[1] www.last.fm
[2] www.musicbrainz.org
[3] the.echonest.com

Figure 5.1: A sample resonse for the API method *album.getInfo*. Apparently, no information regarding the release type is available. The release date is stored in the field $<releasedate\rangle$.

The complement MusicBrainz ID field (<mbid⟩) is an important piece of information in the XML response. As we shall see later, MusicBrainz provides more detailed data about an album or the discography in general of an artist. Consequently, even if Last.FM is fruitless on its own, the mapping/link to MusicBrainz is extremely valuable and hence exploitable.

The EchoNest

EchoNest was founded in 2005 and started as part of a dissertation project at MIT. It currently aims to provide music related services to developers and media companies, including MTV, BBC, Warner Bros and others. Their services are mostly based on a large, automatically generated, dataset of song features. However, similar to Last.FM, EchoNest has been really proactive in providing powerful tools via their API. Although the processing itself is a "black box", it does provide extensive functionality for analysing, retrieving metadata, and processing music.

We are interested in EchoNest's database rather in the provided audio-content analysis. We want to extract as much date information as possible regarding an artist or a song. Figure 5.2 shows the json-format response to an artist-search query, which includes information about the artist's active period (field *years_active*).

Figure 5.2: A sample response for a search-artist query. The returned artist (Radiohead) is accompanied by active-years information. Radiohead began their career in 1985 and are still active.

The same query can include a link to a MusicBrainz profile, via the MusicBrainz ID, as shown in Figure 5.3.

Figure 5.3: A sample response for a search-artist query. The returned artist (Radiohead) is accompanied by a MusicBrainz ID.

The song-related queries are fruitless in terms of date information, therefore the only data relevant to our context is "active years" and MusicBrainz lD.

MusicBrainz

MusicBrainz offers a free fingerprinting (song identification) service that allows users to access its huge recording metadata database using web services in addition to an API. The database contains various pieces of information about music, from artists and their releases to works and their composers. The granularity of MusicBrainz, with regard to discography information, is much higher compared to Last.FM and EchoNest, probably due to MusicBrainz' aim to become the most complete music-metadata collection.

A sample artist-search response is shown in Figure 5.4. Such a response returns a ranked list of artists with name similar to the query. The ranking is based on an internal MusicBrainz function and given possible naming ambiguities, it can be erroneous. However, it is important to notice that information about the life and death of the artist is also provided ($\langle$*life-span*$\rangle$ field). Such

piece of information is really valuable when it comes to placing artists into a time frame. Regarding song data, let us examine a song-search response, as shown in Figure 5.6. The most important fields, in our context of use, are ⟨*date*⟩ and ⟨*release-group*⟩, corresponding the release date and the release type (single, album, compilation etc.) respectively of the song.


```
▼<metadata xmlns="http://musicbrainz.org/ns/mmd-2.0#" xmlns:ext="http://musicbrainz.org/ns/ext#-2.0">
  ▼<artist-list offset="0" count="648">
    ▼<artist ext:score="100" type="Group" id="70f5ff67-59c0-4d04-984b-bf0e1e602ee1">
       <name>Fred</name>
       <sort-name>Fred</sort-name>
       <country>IE</country>
       <disambiguation>Irish 5-piece alternative/indie-pop band</disambiguation>
      ▼<life-span>
         <ended>false</ended>
       </life-span>
     </artist>
    ▼<artist ext:score="100" type="Person" id="e27d48a1-ba25-45b3-a20e-e2377e5f638e">
       <name>Fred</name>
       <sort-name>Fred</sort-name>
       <gender>male</gender>
       <country>FR</country>
       <disambiguation>French singer/songwriter Frédéric Métayer</disambiguation>
      ▼<life-span>
         <begin>1973</begin>
         <ended>false</ended>
       </life-span>
     </artist>
    ▼<artist ext:score="100" type="Group" id="220b8211-cc4f-44dc-8860-d40c4bdeb95a">
       <name>Fred</name>
       <sort-name>Fred</sort-name>
       <disambiguation>80s UK synth pop</disambiguation>
      ▼<life-span>
         <ended>false</ended>
       </life-span>
      ▼<tag-list>
        ▼<tag count="1">
           <name>uk</name>
```


Figure 5.4: A sample MusicBrainz response for an artist-search query.

It becomes quite obvious from the previous, that MusicBrainz constitutes an important source of date information for both artists and songs. Therefore, our system aims at exploiting it as much as possible, even if sometimes the returned ranked lists lack confidence. Given that, our system makes use of the fact that EchoNest and Last.FM can provide links to MusicBrainz profiles, via MusicBrainz IDs, since it is quite common for a song/artist not to appear in all databases (see Figure 5.6).

Search Engines

There are many ways to search through the Web, however dedicated search engines, such as Google Search and Bing Search are the most powerful tools that can be taken advantage of. Different search engines use a variety of algorithms to search the web, ranging from semi to full-automatic web crawlers. The most popular algorithm, for at least Web-MIR research, is Google's PageRank, which favors sites in the search results based on how many other sites link to them. Google in addition employs other criteria to generate results, such as machine learning.

It should be pointed out that such services act as gateways between the user and all that is indexable on the Web. This clearly implies that, what is called the "deep" web (flash pages, on the fly content, etc.), cannot be directly accessed.

Figure 5.5: A sample MusicBrainz response for an song-search query.

5.2.2 General framework

In general, given a set of $\langle Artist, Title \rangle$ tuples, the system's goal to estimate the probability distribution function that best fits the artist's productivity pdf. For the YoR task the input is a single tuple while the output estimate is a single integer number. Both tasks employ the same fundamental procedure which comprises of three major components:

1. Editorial Metadata retrieval
2. Web Mining
3. Post-processing and estimation

We will now examine how these are implemented and vary for both distinct cases.

5.2.3 Editorial Metadata retrieval for Artist Years of Productivity estimation

Given the input $I = \{\{a_1, S^{a_1}\}, \{a_2, S^{a_2}\}, .., \{a_i, S^{a_i}\}\}$ where $a_i \in A$ an artist name and $S^{a_i} \in S$ a set song titles corresponding to the a_i artist's recordings, the system firstly tries to match any of the artists to the databases of Last.FM, EchoNest and MusicBrainz. We want to extract the following information about a_i:

1. Discography (from MusicBrainz)
2. Lifespan (from MusicBrainz)
3. Active years period (from EchoNest)

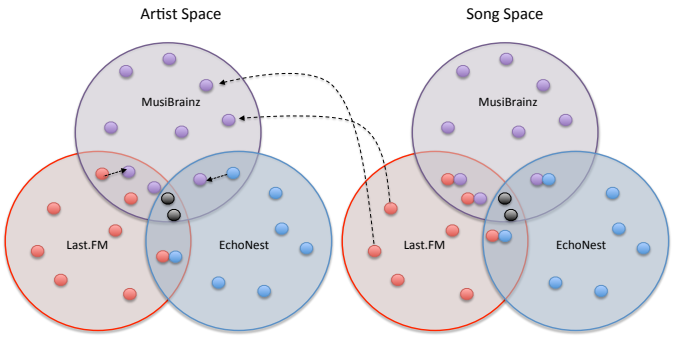

Figure 5.6: A simple Venn diagram representing the song and artist spaces in Last.FM, MusicBrainz and EchoNest. Songs or artists appear in none, one, two or all three databases. Artist profile links, via MusiBrainz IDs (depicted as arrows), may exist for both EchoNest and Last.FM. However, links to MusicBrainz artist profiles, can apper in Last.FM songs only.

The retrieval is based on jSongMiner [1], which requires input of the form $\langle a_i, s_j \rangle$. In reality jSongMiner just passes the query to the MIS systems, retrieves the results, merges and presents them in a comprehensive way. It become obvious that the EchoNest, Last.FM and MusicBrainz perform the actual matching and ranking themselves.

We are only interested in either the MusicBrainz or EchoNest ID of the artist, so as to extract the information in the list above. These can be provided via links from any of the Editorial Metadata sources. For example, if we have a Last.FM match for $\langle a_i, s_k \rangle$, this can provide a link to a_i in MusicBrainz even though s_k is not listed as a recorded song by a_i in the latter.

An issue that arises is that not all pairs can be matched to any of the databases. This is because either no information about an artist a_i exists, or no information about any of his songs in S^{a_i}. In that case the system has to compromise and try to match the artist name solely to the databases. Given that the returned match has now higher probability of being erroneous (artists with same or similar name), the system has to assign a confidence.

Given those factors we have identified four distinct cases:

1. The EchoNest profile exists and also the link to MusicBrainz. This is the best-case scenario. The system then retrieves the "Active Years" data from EchoNest and the "Life span", discography data from MusicBrainz. A confidence value of 1 is set for both sets of data.

2. Only the EchoNest profile exists with no link to MusicBrainz. In that case, we query MusicBrainz ourselves using only $\langle a_i \rangle$. The results of MusicBrainz can be noisy hence even the top ranked results may not be very reliable. Therefore we set the confidence for EchoNest to 1, and for MusicBrainz to 0.6. (Remember that MusicBrainz doesn't offer links to EchoNest, and therefore we cannot ensure that the artists retrieved from both sources match).

3. Only MusicBrainz data exist. In that case, we query EchoNest with $\langle artist \rangle$. Again we assume that the results may not be reliable. Therefore, when the top ranked artists is received, we compare his corresponding MusicBrainz ID (if it exists) we the one already acquired. In case of a match the confidence values for both EchoNest and MusicBrainz are set to 1. In any other case the EchoNest data is discarded. However, if the link between MusicBrainz and Echonest doesn't exist, we set the confidence value of the latter to 0.8.

4. Both MusicBrainz and EchoNest data is absent. In that case we query Echonest with $\langle Artist \rangle$ and then try to find the link to MusicBrainz. If the link exists, we set both the confidence values to 0.8. If the mapping doesn't exist we query MusicBrainz with confidence of 0.6. If the initial EchoNest query doesn't return a match, we simply query MusicBrainz with a confidence of 0.6.

This complex procedure is depicted in the pseudo code below:

```
If (EC and MB) then
    EC_conf=1
    MB_conf=1

If (EC and NOT MB) then
    EC_conf=1
    MB=search_musicbrainz
    if (MB) then MB_conf=0.6

if (MB and NOT EC)
    EC=search_echonest
    if mapping(EC,MB)
        EC_conf=1
        MB_conf=1
    else
        EC_conf=0.8
        MB_conf=1

if (NOT MB and NOT EC)
    EC=search_echonest
    if (EC)
        MB=search_musicbrainz
        if mapping(EC,MB)
            EC_conf=0.8
            MB_conf=0.8
        else
            EC_conf=0.8
            MB_conf=0.6
    else
        MB=search_musicbrainz
        MB_conf=0.6
```

Histogram construction

After the previous process we now acquire a set of confidence values $C = \{c_{mb}, c_{ec}\}$ for the data retrieved from MusicBrainz and EchoNest respectively. EchoNest provides us with a maximum of two values corresponding to the start and end years (st_{a_i}, end_{a_i}) of an artist's activity. Given that, a histogram denoted as H_{ec} is created, with both $H_{ec}(st_{a_i})$ and $H_{ec}(end_{a_i})$ set to c_{ec}.

A histogram $H_{mb,disc}$ is also created for the MusicBrainz data, but populating it is more sophisticated. The discography data comprises of album names, song titles accompanied by their release date and "release group" information. The release group is either "album", "single", "compilation" etc. Our method assumes that some release groups (e.g. singles) are more reliable than others (e.g. compilations) with regard to the original date determination. This is because singles are typically released some time before the actual album, while compilations after the artist has released a considerable amount of works. Therefore for each k date information ($year_{k,a_i}$) retrieved from MusicBrainz, we set:

$$H_{mb,disc}(year_{k,a_i}) = \frac{w_{releasegroup} \times c_{mb}}{N_{releasegroup}} \tag{5.1}$$

where $w_{releasegroup}$ a weight value and $N_{releasegroup}$ a normalization factor corresponding to the number of recordings belonging to that particular release group. Table 5.1 presents all the release groups and their assigned weights.

Release group	**Weight**
Compilation	0.1
Album	0.15
Single	1
EP	0.6
Soundtrack	0.5
Spokenword	0.1
Interview	0.2
Audiobook	0
Live	0.3
Remix	0
Other	0.1

Table 5.1: Release groups and their corresponding weights.

Lifespan data corresponds to the start and end years (st_{a_i}, end_{a_i}) of an artist's life. Similar to the Active Years information we create a histogram $H_{mb,life}$ where both $H_{mb,life}(st_{a_i})$ and $H_{mb,life}(end_{a_i})$ are set to c_{mb}. An example of the resulting histograms is shown in Figure 5.7.

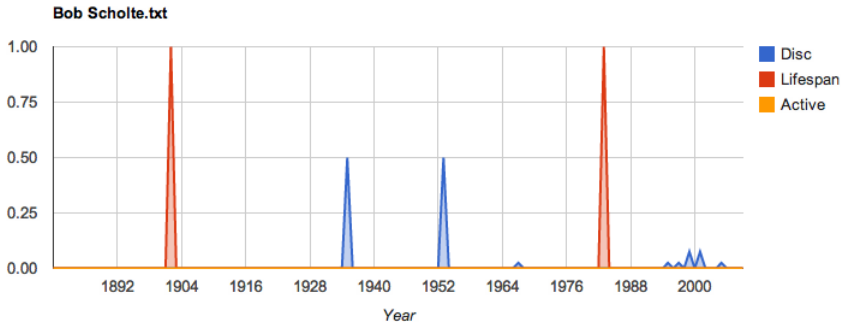

Figure 5.7: The three Editorial Metadata derived histograms for the artist Bob Scholte (1902-1983). Data for the active years of the artist are absent. Even after Scholte's death, albums and compilations of his songs were released which were low-weighted by our method.

5.2.4 Web mining for Artist Years of Productivity estimation

The next step in the process is to identify the Web pages that are related to the artist under consideration. Typically this would include reviews, fan blogs, MySpace or even Last.FM profile pages. As we have discussed previously, our system employs the Google and Bing search engines.

However, as Schedl argues in [2], automatically querying a Web search engine to determine pages related to a specific topic is a common and intuitive task, which is nevertheless prone to a major flaw: when ambiguity is present (e.g. people with common names), a lot of irrelevant pages are returned. Therefore, the main challenge is to somehow, meaningfully restrict the search results to pages related to the desired topic.

This problem, inside the music context, has been addressed by enhancing the search query for the artist name with additional terms, such as "music" and "review". The aforementioned query scheme has been successfully applied to even genre classification tasks [3].

Given that, for each artist $a_i \in A$ we query Google with the scheme "a_i+music" and retrieve the 100 top-ranked URLs, denoted as set G. We query Bing with only a_i, since we have identified that the API returns limited results when queried with complicated terms. It should also be noted that the Bing API results vary significantly when compared to the ones returned by Bing Web interface. In addition, before 01-08-2011 Bing API allowed a maximum number of 80 URLs per query, which was dropped to 50 after that specific date. For the AYoP task, the retrieval was performed before 01-08-2011 and hence 80 URLs per artist were gathered. We denote that set as B.

Since B and G might overlap with eachother, we compute $U_{seed} = B \cup G$. Retrieving the web-pages in U_{seed} is performed by the Apache Nutch web crawler[4]. Although a web crawler, Nutch can also simply gather the web-pages indicated by the URL seed set.

Both Apache Solr[5] and Lucene[6] have been employed by previous researches for indexing web-mined pages. We also employ Apache Solr. The resulting index is now considered as the input for any of the procedures that will be discussed later.

Histogram Construction

The next step in the process is calculating year-histograms for each artist based on the indexed data. We aim at generating a probability distribution that would best model the artist's productivity as it is documented on the Web. We perform that by employing the following tools: page counts, co-occurrence analysis (or cross-tabulation analysis) and $tf * idf$-relevance-scoring as implemented by Lucene.

- **Page counts**: Page counts are usually used in conjunction to web search engines such as Google. The basic idea is to use the search engine's number of indexed Web pages for a given query, to assess its relevance. This

[4] nutch.apache.org
[5] lucene.apache.org/solr
[6] lucene.apache.org/core

number is usually referred to as page counts, and will be denoted as pc_q throughout this report. In our case the number of Solr pages for each artist are usually around 100, however, as we shall see later, this number can be affected by including additional terms to the query, thus retrieving more meaningful values.

- **Cross-tabulation analysis:** Co-occurrence and cross-tabulation analysis were discussed thoroughly in the literature survey. Our approach employs Zadel and Fujinaga's [4] similarity measure (where $C(t)$ the occurrences of the term t and $C(t_1, t_2)$ the number of co-occurrences of the terms t_1, t_2:

$$S(t_1, t_2) = \frac{C(t_1, t_2)}{min(C(t_1), C(t_2))} \tag{5.2}$$

- **Relevance score:** Apache Lucene and Solr employ $tf * idf$ to calculate the query-document similarity as shown in the formula below:

$$sim(q, d) = \sum_{t \in q} \frac{tf_{t,q} \cdot idf_t}{norm_q} \frac{tf_{t,d} \cdot idf_t}{norm_d} \cdot coord_{q,d} \cdot weight_t \tag{5.3}$$

for a collection D, document d, query q, query term t and:

$$tf_{t,X} = \sqrt{freq(t, X)} \tag{5.4}$$

$$idf_t = 1 + log \frac{|D|}{freq(t, D)} \tag{5.5}$$

$$norm_q = \sqrt{\sum_{t \in q} tf_{t,q} \cdot idf_t} \tag{5.6}$$

$$norm_d = \sqrt{|d|} \tag{5.7}$$

$$coord_d = \frac{|q \cap d|}{|q|} \tag{5.8}$$

For each artist a_i and year y_j in $Y = \{1880, 1881, ..., 2010\}$ we query Solr with "$a_i + y_j$". However, our initial experiments showed that such a scheme, although intuitively correct, presents a major flaw. When two or more artists/entities appear on the same web page accompanied with relevant date information, the system might mistakenly relate the wrong dates to the wrong artists. This was a prominent issue in web-pages offering birthdate information for "celebrities" (e.g. www.born-today.com). To overcome this problem we introduce a proximity factor to the query; only pages where the word distance $d(a_i, y_j)$ is smaller than 300 would be returned. In other words, if the artist name a_i appeared more than 300 words later or earlier than the year y_j, then the document pertaining them would be considered irrelevant.

Our approach assigns a score to each query a_i, y_j using a modified cross-tabulation formula:

$$s^*(a_i, y_j) = \frac{score(a_i, y_j)}{min(score(a_i), score(y_j))} \tag{5.9}$$

where:

$$score(a_i, y_j) = max_{1<k<pc_{a_i,y_j}}[sim(\text{“}a_i + y_j\text{”}, d_k)] \times pc_{a_i,y_j} \quad (5.10)$$

Intuitively, the *score*() function corresponds to the product of the page counts and the maximum relevancy. So instead of simply counting to occurrences of a term inside the indexed documents, our method additionally incorporates the relevancy factor to weight that value. We employed *max* instead of *mean* relevancy, since the latter fails to amplify important years and returns a more uniformally-distributed profile.

Now for each a_i we create a histogram H_{web} of 130 bins and set $H_{web}(y_j) = s^*(a_i, y_j)\ \forall y \in Y$. However, date information is not always explicitly stated. For example, it is quite common for an artist that was active during the period 1980-1990, to be considered and identified as an “80’s” artist. Based on this fact we have identified a set of terms T that semantically correspond to date information. These are presented in Table 5.2.

Date correspodence	Term1	Term2	Term3	Term4	Term5	Term6	Term7
1910-1920	1910s	1910’s	tens	10s	10’s	jaren 10	jaren tien
1920-1930	1920s	1920’s	twenties	20s	20’s	jaren 20	jaren twintig
1930-1940	1930s	1930’s	thirties	30s	30’s	jaren 30	jaren dertig
1940-1950	1940s	1940’s	fourties	40s	40’s	jaren 40	jaren veertig
1950-1960	1950s	1950’s	fifties	50s	50’s	jaren 50	jaren vijftig
1960-1970	1960s	1960’s	sixties	60s	60’s	jaren 60	jaren zestig
1970-1980	1970s	1970’s	seventies	70s	70’s	jaren 70	jaren zeventig
1980-1990	1980s	1980’s	eighties	80s	80’s	jaren 80	jaren tachtig
1990-2000	1990s	1990’s	nineties	90s	90’s	jaren 90	jaren negentig

Table 5.2: Semantic terms and their actual time correspondence.

We query Solr with tuple of the form a_i, t_j, where t_j in T, and then increase the value of the corresponding bins by $s^*(a_i, t_j) \times 0.1$. For example if the query “fifties”+“Bob Scholte” returned a score s', then:

$$H_{web}(y_f) = H_{web}(y_f) + s' \times 0.1, \forall y \in \{1950, 1951, .., 1959\} \quad (5.11)$$

The 0.1 factor corresponds to the uniform distribution of the s' across the 10 bins. An example of an artist’s final histogram (on top of the Editorial Metadata histograms) is shown in Figure 2.8.

5.2.5 Post-Processing for Artist Years of Productivity Estimation

After the two previous steps, Editorial Metadata information extraction and Web mining, the system acquired four separate histograms (H_{ec},$H_{mb,disc}$,$H_{mb,life}$ and H_{web}) for each artist. The question now is how to combine those pieces of information in a meaningful way. We solve this issue in a three-step function.

Discography Processing

In many cases, it is very likely for the discography data to be incomplete for apparent reasons. Monitoring music sales or archiving music records have emerged

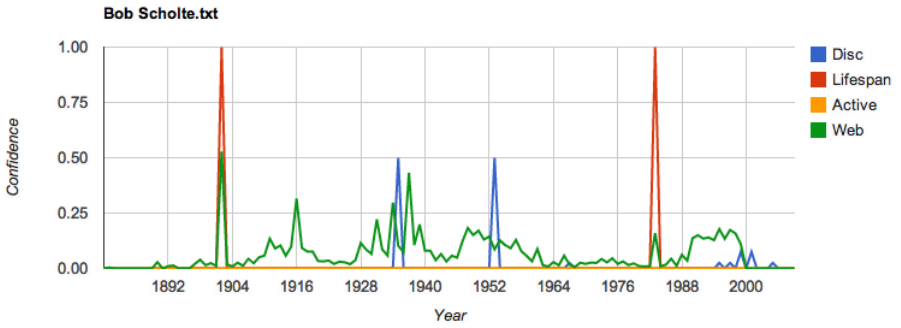

Figure 5.8: The four histograms (Editorial Metadata and Web derived) for the artist Bob Scholte (1902-1983). The scoring function nicely amplified the birth and death date years of the artist.

in the 70's and early 80's when music became easily distributed and more commercialized. Today, artists of low popularity may lack fair representation on MusicBrainz, EchoNest and Last.FM.

Whatever the reason, our method tries to compensate for that fact by the following process; we create an 1-dimensional kernel k of size 10 with decreasing values $\{1, 0.9, .., 0.1\}$. Then we convolve $H_{mb,disc}$ with k:

$$H_{mb,disc} = H_{mb,disc} * k \tag{5.12}$$

This gives us the same effect as Bogdanov's and Herrera's method [5] for determining a records's authentic epoch. Authentic epoch, according to the authors, represents the years when the music was firstly recorded, produced and consumed. Intuitively our process is based on the assumption that the probability of an undocumented record to have been released, is higher in the years after the release of documented record. This assumption is debatable, since nowadays artists tend to release albums and then occupy themselves with other career related tasks, such as touring and promoting the record. Thus the probability of releasing a song/recording immediately after the release of an album is very small. However, for the sake of convenience and without loss of generality, our method doesn't follow this assumption.

Histogram Addition

After we process the discography histogram, we create a histogram H_f whose bin values hold the sum of the the corresponding bins of the normalized H_{web} and $H_{mb,disc}$:

$$H_f(i) = \frac{H_{web}(i)}{max(H_{web})} + H_{mb,disc}(i), \forall i \in 1880, 1881, ..., 2010 \tag{5.13}$$

Noise Removal Part I

Given that the $H_{mb,life}$ and H_{ec} are non-zero, and thus contain some information, out method decreases the values of those bins in H_f that fall outside

the life and active periods of the artist in hand. This agrees with our definition of AYoP that considers an artist active only during his life. Let's assume that $H_{mb,life}$ and H_{ec} contain two values each $\{begin, end\}$ and $\{start, end'\}$ respectively. Noise removal is defined as:

$$H_f(i) = H_f(i) \times c_{mb}, \forall i \in \{begin, begin+1, .., end\} \tag{5.14}$$

$$H_f(i) = H_f(i) \times c_{ec}, \forall i \in \{start, start+1, .., end'\} \tag{5.15}$$

and

$$H_f(j) = H_f(i) \times (1 - c_{mb}), \forall j \in \{1880, .., begin-1\} \cup \{end+1, .., 2010\} \tag{5.16}$$

$$H_f(j) = H_f(i) \times (1 - c_{ec}), \forall j \in \{1880, .., start-1\} \cup \{end'+1, .., 2010\} \tag{5.17}$$

Intuitively this corresponds to the process of minimizing the confidence about the artist's productive period given his lifespan, active period and our confidence on these pieces of information. For example, if the system has identified that the lifespan information corresponds to the correct artist and thus the confidence is 1, then all the bin values in H_f before his birth and after his death will be set to 0. The same idea holds for the cases where only the birthdate is available. Both *end* and *end'* values are set to the year 2010.

Noise Removal Part II

Even after the previous processes, the data inside the lifespan or active period of an artist contain a considerable amount of noise. The noise removal process to be discussed, is based on the heuristic that artists cannot have released any recordings during their early childhood. The exceptions are fairly limited (e.g. Michael Jackson, Jordy) and thus no loss of generality.

However, as the artist approaches adulthood the probability of being musically active is getting higher. Based on our intuition we also assume that an artist achieves his maximum productivity at the ages between 20 and 40 and then slowly fades away.

This is modelled as an envelope-probability density function where the attack and decay-sustain are generated by two Gaussian distributions of size 50 and 120 respectively, while the release by a linear function. The coefficients of the Gaussian window w are computed by the following formula:

$$w(n) = e^{-\frac{1}{2}(a \times \frac{n}{N/2})^2} \tag{5.18}$$

where $a = 5$ and $a = 1.25$ for the attack and decay-sustain respectively.

The final productivity pdf W is then:

$$W(n) = w_{50}(n), \forall n \in \{1, 2, .., 25\} \tag{5.19}$$

$$W(n) = w_{120}(n + 35), \forall n \in \{26, .., 80\} \tag{5.20}$$

$$W(n) = -0.001 \times x + 0.09, \forall n \in \{81, .., 90\} \tag{5.21}$$

Before we continue, it should be noted that the envelope's settings were arbitrarily chosen, based on our early intuition and findings. It is possible for other type of envelopes to be used, which model artists' productivity better.

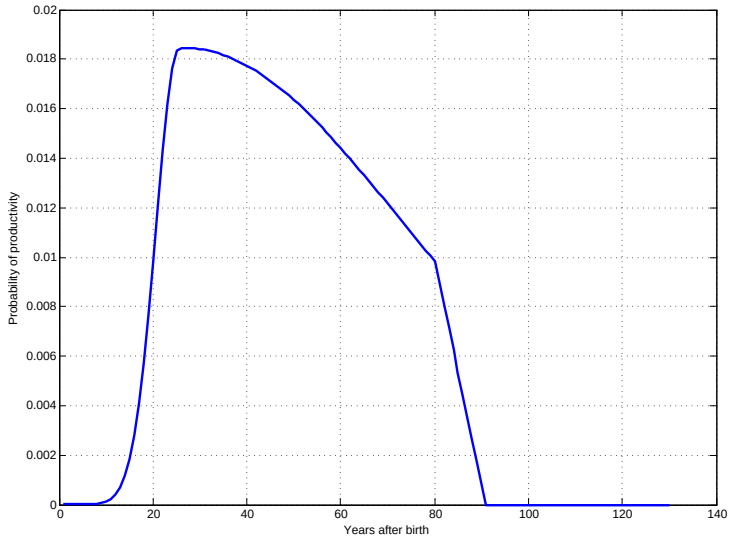

Figure 5.9: The probability density function for artist productivity.

Now given the artist's birth date, as provided by $H_{mb,life}$, we multiply H_f with W, so that the peak of W's distribution to coincide with the artist's 25th year of life. This is formulated as follows, although the details of traversing W are omitted:

$$H_f = H_f \times W^T \tag{5.22}$$

Smooth and Normalize

The final step in the whole process of AYoP estimation is smoothing H_f with a Gaussian window w_5 of size 5 and normalizing it so that its coefficients sum up to 1:

$$H_f = H_f * w_5 \tag{5.23}$$

H_f is now our final estimate for AYoP.

Noise Removal for Absent Birth Date Information

We have previously mentioned that obscure and old artists are usually poorly represented in MusicBrainz, EchoNest and Last.FM. Consequently, it is very likely for birth date information to be missing. This is a major drawback considering that both noise removal processes are based on such information.

In order to deal with this problem, we go back to H_{web} which optimally amplifies important years in the life of an artist. We have seen some cases where sharp peaks perfectly coincide with the birth and death dates of an artist. The idea is to locate these peaks and pick the one that fits our productivity probability assumptions.

Our approach relies on an iterative peak finding function f_p and the probability density function W. $f_p(H_{web})$ return peaks in H_{web} that exceed their neighbours by at least the value of r, where r an iteratively decreasing func-

tion starting from $r = 1$. That scheme ensures that sharpest set of peaks $P = \{p_1, p_2, ..., p_d\}$ will be eventually located.

In order to identify which peak corresponds to the true birthdate, our method computes a set of histograms as such:

$$H_f^p = H_f \times W_p^T, \forall p \in P \tag{5.24}$$

Meaning that the window W is traversed so as its max to coincide with $p+25$ and then multiplied with H_f. The p with the largest product H_f^p is considered the birth date estimate. This whole process is based on the assumption that after a sharp peak corresponding to the birth date, H_{web} will follow a distribution modeled by W.

5.2.6 Editorial Metadata retrieval for Year of Release estimation

As previously mentioned, AYoP and YoR functions are based on the same set of tools. Therefore during the following paragraphs we will only focus on the elements that differ and omit the rest. Firstly, we shall revisit the goal of YoR estimation; given the input $I = \{\{a_1, S^{a_1}\}, \{a_2, S^{a_2}\}, .., \{a_i, S^{a_i}\}\}$ where $a_i \in A$ an artist name and $S^{a_i} \in S$ a set song titles corresponding to the a_i artist's recordings, our aim is to estimate a year value between 1880 and 2010 that minimizes the distance between the truth and the estimate year of release for each song in S.

We have already seen that Editorial Metadata offer discography data, with MusicBrainz being the most sophisticated and complete service. Therefore, similar to AYoP we query Editorial Metadata databases with tuples of the form $\langle a_i, s_j \rangle$ and retrieve the discography, lifespan, and active years, while also assigning two confidence values c_{mb}, c_{ec} for MusicBrainz and Echonest respectively.

It is now fairly easy to search into the discography of a_i for any of the songs in S^{a_i}. If we have a match, the release group is "Single" and the year of release y is available, then a histogram $H_{mb,disc}$ of 130 bins for the song in hand is populated such that $H_{mb,disc}(y) = c_{mb}$. Of course, if not all of the conditions are met, then all bins in the song's histograms are set to 0.

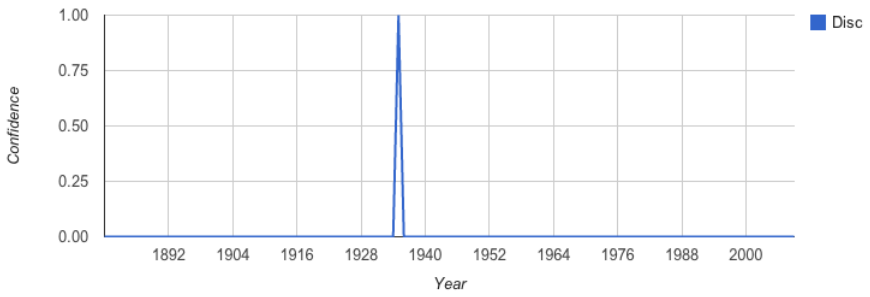

Figure 5.10: The discography based histogram for the song "Margaretta" by Bob Scholte.

5.2.7 Web mining for Year of Release estimation

During this step we query Google with "s_j+$music$" for $s_j \in S$ and retrieve a set of 100 URLs for each song denoted G. Bing is queried with "s_j" and only the 50 top-ranked URLs are gathered for each song (set B). We retrieve 50 instead of 80 since the data for this task were gathered after 1-8-2012. Eventually $B \cup G$ is computed and the corresponding web-pages are retrieved using Apatche Nutch. An index is built on top of the harvested sources using Apache Solr.

Histogram Construction

In the case of AYoP, Solr was queried with "a_i+$year_j$". In contrast for YoR we perform two set of queries for each tuple $\langle a_i, s_j \rangle$, namely "$s_j + year_k$" and "$s_j + a_i + year_k$" where $year_k \in \{1880, 1881, ..., 2010\}$. Similar to AYoP, we additionally employ the term set T and of course the scoring functions 2.9, 2.10.

This results to the generation of two histograms $H_{web,s}$ and $H_{web,s+a}$. The first one represents the frequency/relevancy of the title accompanied by a year value on the Web. The second represents the frequency/relevancy of the title accompanied by a year value in addition to the corresponding artist name.

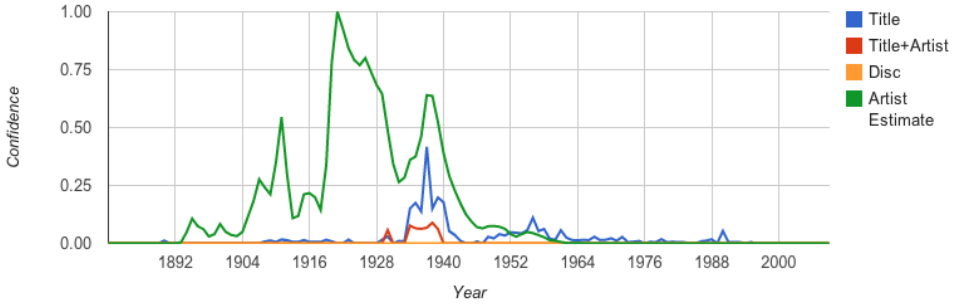

Figure 5.11: The Web-mined derived histograms "Title" and "Tile+Artist" on top of the discography and Artist estimate histograms for "Havenmuziek" by August de Laat.

5.2.8 Post-Processing for Year of Release Estimation

For each song s_j we have acquired three histograms $H_{mb,disc}, H_{web,s}, H_{web,s+a}$ in addition to the AYoP derived histogram for the corresponding a_i, denoted from now on as H_{f,a_i}. We are once again faced with the problem of combining information from these histograms in a meaningful way so as to compute the optimal YoR estimate.

Our method treats the aforementioned histograms as probability density functions, although the sum of each histogram's coefficients is not always 1. However, our method employs a conical combination or conical sum approach that assumes that the input vectors correspond to pdfs or mixture components. More formally, given a finite number of vectors $x_1, x_2, ..., x_n$ in a real vector space, a conical combination or a conical sum of these vectors is a vector of the

form:

$$a_1x_1 + a_2x_2 + ... + a_nx_n, a_i \geq 0 \tag{5.25}$$

In our case the conical sum H_{CS} of the computed histograms is:

$$H_{CS} = w_1 \times H_{web,s} + w_2 \times H_{web,s+a} + w_3 \times H_{f,a_i} + w_4 \times H_{mb,disc} \tag{5.26}$$

with $W_v = [w_1, w_2, w_3, w_4]$ the weight vector. The problem is reduced to finding the optimal W_v. But before we deal with that problem we shall discuss the preprocessing stage of the histograms.

Smoothing and Normalizing

Similar to AYoP, we convolve $H_{web,s}, H_{web,s+a}$ with a Gaussian window of size 5. H_{f,a_i} has been normalized during AYoP, so that $max(H_{f,a_i}) = 1$. We could have normalized the web-mined-derived histograms the same way or such that $\sum_i H_{web,s}(i) = 1$ and $\sum_i H_{web,s+a}(i) = 1$. However, the disadvantages of such process are obvious; we would be only keeping the relative relations between the bins and throw away the actual values assigned by the scoring functions 4.9, 4.10. Therefore at this stage, only Gaussian smoothing is applied.

Conical Sum using a Genetic Algorithm

We are now faced with our original problem of calculating the weight vector W_v that most meaningfully adds the four histograms. Our method employs a genetic algorithm (GA), that iteratively modifies W_v so that the optimal configuration is eventually found based on a training set.

Generally in GAs, a population of strings (binary commonly) encodes some candidate solutions. The algorithm starts from a population of randomly generated individuals and happens in generations N_{gen}. At each generation, the fitness of every individual in the population, usually referred a chromosomes, is evaluated using a fitness function, while multiple individuals are stochastically selected from the current population and modified (recombined and randomly mutated) to form a new population. The new population is then used in the next iteration/generation of the algorithm. The algorithm terminates when the maximum number of generations has been reached, or when the fitness function hasn't increased for fixed amount of iterations. Pseudo code for this algorithm is shown below:

```
1. Choose the initial population of individuals
2. Evaluate the fitness of each individual in that population
3. Repeat on this generation until termination: (Ngen reached.)
       1. Evaluate the individual fitness
       2. Select the best-fit individuals for reproduction
       3. Breed new individuals through crossover and
            mutation operations to give birth to offspring
       4. Replace population with new individuals
```

In order to maximize the fitness function, while at the same time avoid stucking in local maximums, GAs typicaly employ three functions. The crossover function which aims to "mate" candidates, hoping for better offspring, the mutation function that stochastically inverts bits of the candidates and the eliticism function which finds the best candidate in the population and adds it to the population of the offspring.

Discussing GAs in depth exceeds the scope of this report; it is the author's belief that the amount of information provided is more than enough to understand the overall process. Details about the GA's actual configuration will be discussed later.

Final Estimation

After the summing of the histograms, the system now acquires $H_{f,s}$ which can be considered as a probability distribution if we normalize it such that $\sum_i H_{f,s}(i) = 1$. Our YoR estimate is now as follows:

$$YoR^{est} = max_{index, 0<i<130}(H_{f,s}(i)) \tag{5.27}$$

Where max_{index} the index with the greatest value. In simpler words the estimate is simply the actual year correspondence of the coefficient with the maximum value.

5.3 Experiments and Evaluation

5.3.1 Test and training set

There exists no standardized or previously used data set for this kind of task, therefore we had to build one from the scratch. To this end, we manually gathered 2323 Dutch song titles accompanied by original release dates, ranging from the period of 1900 to 1959. Only a subset of 639 titles, corresponding to 7 Dutch artists (see Table 5.3), is used for evaluation due to storage capacity constraints. From those, a set of 100 randomly selected songs were used as input for the genetic algorithm. Overall, 26832 documents were downloaded and indexed for the YoR and 3391 for AYoP including a set of "noise" webpages (irrelevant to the artists themselves).

	Artist Name
1	August De Laat
2	Bob Scholte
3	Eddy Christiani
4	Kees Pruis
5	Lou Bandy
6	Louis Davids
7	Willy Derby

Table 5.3: The set of Dutch artists in the test set.

5.3.2 Ground Truth

The ground truth for the YoR estimation obviously corresponds the year of release, associated with song in hand. For AYoP, generating the ground truth is a more complex procedure. The reason is two-fold: a) we can never be sure that the data for each particular artist is complete and b) we can never be sure that the data is correct, since web sources often disagree with regard to release dates. Point "b" holds for the YoR task also, however we assume that the cases where YoR is debatable are limited.

For each artist a_i in the test set we create a 130-bin histogram H_{gt} corresponding to the actual years 1880 to 2010. For each song $s_j \in S^{a_i}$ we modify the histogram as such:

$$H_{gt}(y_{s_j}) = H_{gt}(y_{s_j}) + 1 \tag{5.28}$$

where y_{s_j} the associated release year of the song s_j. In other words, we simply fill the histogram bins with the counts of release years. Intuitively, we assume that the artist is more productive in the years where the maximum number of songs were released. In many cases the supposition does not hold, since an album A may contain 15 songs and 60 minutes of music and an album B 10 songs and 70 minutes of music. However for the sake of convenience we assume that all songs tend to be of the same length.

In order to deal with point "a", H_{gt} is smoothed using a Gaussian window of size 5. Therefore, we intuitively assume that given a release year y, it is very likely for other releases to have followed or came before y. Eventually we normalize the histogram so that $max(H_{gt}) = 1$. Examples of ground truth for AYoP are showing in Figures 5.12, 5.13.

Figure 5.12: The AYoP ground truth for August De Laat (1882-1966).

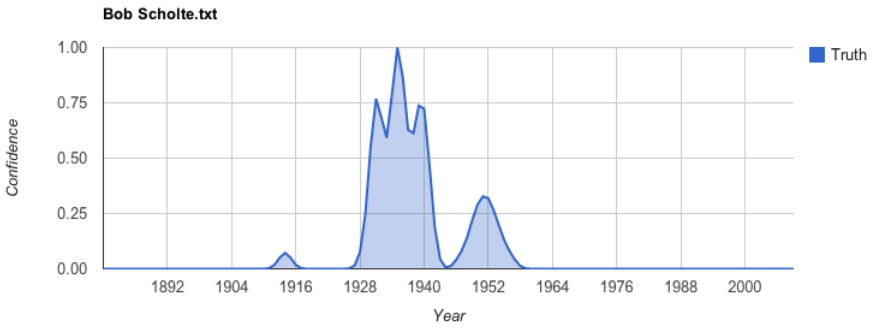

Figure 5.13: The AYoP ground truth for Bob Scholte (1902-1983).

5.3.3 Evaluation Measures

Given N the number of songs in S, $YoR^*_{s_j}$ the true release date of the song s_j and $YoR^{est}_{s_j}$ the estimate. then we define as $MeanError$ the following:

$$MeanError = \frac{\sum_{s_j \in S} |YoR^{est}_{s_j} - YoR^*_{s_j}|}{N} \tag{5.29}$$

"Accuracy" for a year-window of size x is defined by the following formula:

$$Accuracy_x = \frac{\sum_{s_j \in S} f_x(YoR^{est}_{s_j}, YoR^*_{s_j})}{N} \tag{5.30}$$

where

$$f_x(t,e) = \begin{cases} 1 & \text{for} \quad |t-e| \leq x \\ 0 & \text{for} \quad |t-e| > x \end{cases} \tag{5.31}$$

Function f considers as "hits" all the estimates that fall within a certain window around the truth value. For example if the target year is 1950, for window of size 5, every estimate between 1945 and 1955 would be considered as a hit. Consequently $Accuracy_x$ is just the mean value of f for all the songs in S.

The aforementioned measures are employed for the YoR evaluation only. For AYoP the idea is to examine the overlap between the ground truth and estimate distributions. This is achieved by using measures such as precision, recall and their harmonic mean usually denoted F-measure:

$$precision = \frac{tp}{tp + fp} \tag{5.32}$$

$$recall = \frac{tp}{tp + fn} \tag{5.33}$$

$$F = 2 \times \frac{precision \times recall}{precision + recall} \tag{5.34}$$

where tp, tn, fp, fn the number of true positives, true negative, false positives and false negatives respectively.

5.3.4 Training Phase

In order to evaluate the optimal W_v set of weights for the conical sum of the histograms (see 2.8.2), we run a genetic algorithm on the training set of 100 titles. The details of the genetic algorithm are presented in Table 5.4

The best W_v was found after 51 iterations and it is presented in Table 5.5. It is worth discussing the effect and meaning of the weight vector W_v, since our initial experiments have shown that slight modifications to the weights affect accuracy at large. But what do the weights tell us about the usefulness of the gathered information? Firstly, the small weight for the artist histogram H_{f,a_i}, exposes to us its futility; given that the other information is present, the artitst's productivity distribution cannot provide us with an accurate YoR estimation. However, when the rest of the histograms are empty, H_{f,a_i} is our best choice.

Now, assuming that $H_{web,s}$ and $H_{web,s+a}$ roughly correspond to probability density functions $P(s)$ and $P(s,a)$ respectively, it is sensible that $w_2 > w_1$. This

Factor	Value
Initial population size	20
Crossover Fraction	0.8
Elite Count	2-0.05*population
Generations	100
Migration Fraction	0.2
Migration Interval	20
Lower bounds	[0,0,0]
Upper bounds	[1,1,1]
Maximization function	$\frac{\sum_{1 \leq x \leq 5} Accuracy_x}{5}$

Table 5.4: Genetic algorithm setup. Details about each variable's meaning can be found at www.mathworks.nl/help/toolbox/gads/gaoptimset.html

is because $H_{web,s+a}$ models better the co-occurrence of both the song title and the artist name.

Finally, the weight w_4 of the MusicBrainz derived histogram, reveals to us that MusicBrainz can provide valid information but still not as valuable as $H_{web,s+a}$.

Weights	w_1	w_2	w_3	w_4
Values	0.538	0.875	0.08	0.61

Table 5.5: Found weights for the conical sum, using a genetic algorithm on the training set.

To sum up, given that the genetic algorithm locates a local and not the global maximum, while also that the input histograms are not normalized in any way, the aforementioned weight-meanings may lack validity. However, our intuition agrees with the weightings and therefore assumed optimal.

5.3.5 Results

Artist Years of Productivity Results

The results for the AYoP task are presented in Table 5.6. At first glance, given that we are more interested in high F-measure values, the numbers may seem disappointing. However, precision, recall and F-measure fail to capture an important aspect of the method's results. As we shall later, by studying each individual artist case, our approach always agrees with the ground truth with regard to the date boundaries of the productivity. In other words placing the artist into a time context is very successful. The distribution of productivity inside that context, is what draws the numbers down. And since the ground truth distribution depends on manually gathered data, it is safe to state that either a modified or a completely new evaluation metric should be employed. However, for the purposes of this experiment, precision, recall and F-measure will be considered as the optimal measures.

We shall now discuss each individual artist case, since the number of artists is small and valuable conclusions can be derived. Starting with "August De Laat" (see Figure 5.14), the results show the lowest precision values. The artist is well placed into the time context but our approach assumes large productivity from 1920 to 1933. The reason for this misbehavior can be twofold; a) the ground

	Artist Name	*Precision*	*Recall*	*F-measure*
1	August De Laat	0.35692	0.94457	0.51808
2	Bob Scholte	0.66435	0.85348	0.74713
3	Eddy Christiani	0.53249	0.91053	0.67199
4	Kees Pruis	0.70479	0.78344	0.74204
5	Lou Bandy	0.54522	0.83328	0.65916
6	Louis Davids	0.37065	0.89438	0.5241
7	Willy Derby	0.75824	0.93309	0.83662
Mean		**0.56181**	**0.87897**	**0.6713**

Table 5.6: Precision, recall and F-measure for the seven artists in the test set.

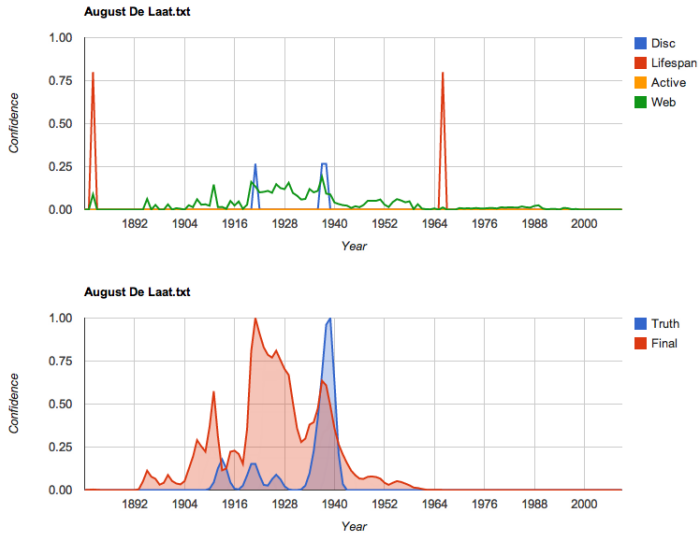

Figure 5.14: (Top)The four input histograms, (bottom) the ground truth against the estimate AYoP for August de Laat.

truth data for August De Laat inside that certain period is disproportionally low compared to the data around 1935, and b) our method is truly incorrect. We believe that point a) has high probability of being true, which faces us with the problem of "ground truth normalization"; a large amount of acquired data for a certain period will attenuate the AYoP values for years with incomplete data. Solving this issue exceeds the scope of this study, so for now we will assume that point b) is valid.

The "Bob Scholte" case (see Figure 5.15) presents some of the best results in the test set. Firstly it should be pointed out that the "Active period" and "Lifespan" information coincide ; an incorrect assertion by default. The active period of an artist should begin immediately after his first release or concert. However, this error does not affect the results. The final AYoP estimate not only places Bob Scholte into the correct time context, but also perfectly models his productivity. An interesting remark: the peak of the Web input histogram at the year 1916 corresponds to the beginning of Scholte's career.

The "Eddy Christiani" case (see Figure 5.16) shows great results in terms of time-context placing and productivity modelling. However, our method assumes

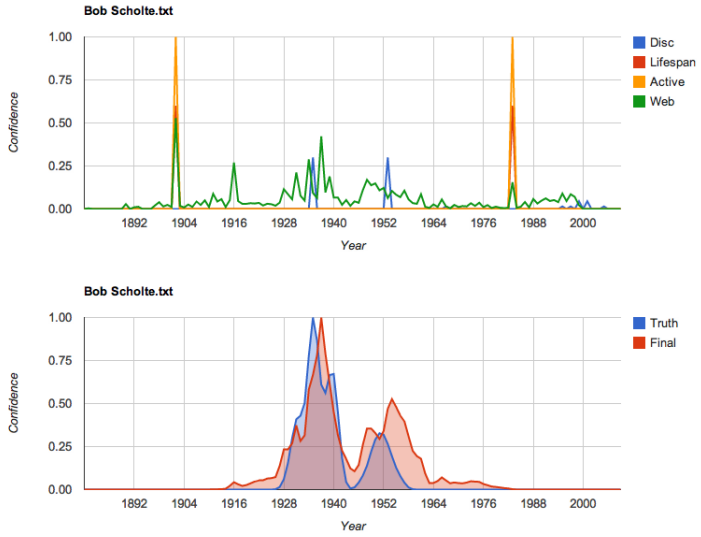

Figure 5.15: (Top)The four input histograms, (bottom) the ground truth against the estimate AYoP for Bob Scholte.

that even after 1964 the artist was still active. This assertion is actually correct (see artist's rateyourmusic.com's profile), but since our ground truth comprises of songs up until 1959 a mismatch occurs. Future implementations of the ground truth should include the complete discography of Eddy Christiani.

The "Kees Pruis" case shows remarkable results and therefore it will not be discussed in detail. In contrast, it is worth examining the case of "Lou Bandy" (1890-1959), where no data about his lifespan or active years were available. In that case our method tried to estimate the birthdate. Then multiplication with the window productivity W, later in the procedure, cleared some of the noise right after the artist's birth and death. Eventually, the final AYoP estimate was close to the ground truth both in terms of time-context placing and distribution modelling.

The same holds for the "Louis Davids" (1883-1939) case (see Figure 5.19). However, the window W proved to be unfit for that particular artist, given that the artist died in 1939. Consequently, we are faced with the problem of death-date absence, which is more difficult to solve than the birth date. As it becomes obvious, from all the previous cases and histograms, the birth date peaks are more easily distinguishable, since the neighboring years do not contain any significant information. This doesn't hold for death dates unless the artist passed away in years of low productivity. This whole issue will be addressed in future implementations of our method.

Finally, the case of "Willy Derby" shows great results and therefore is not worth discussing in detail.

Song Year of Release Results

Table 5.7 and Figure 5.21 present the $Accuracy_x$ for windows ranging from 1 to 10. The Mean Error is 2.892 years. As a general evaluation measure we will

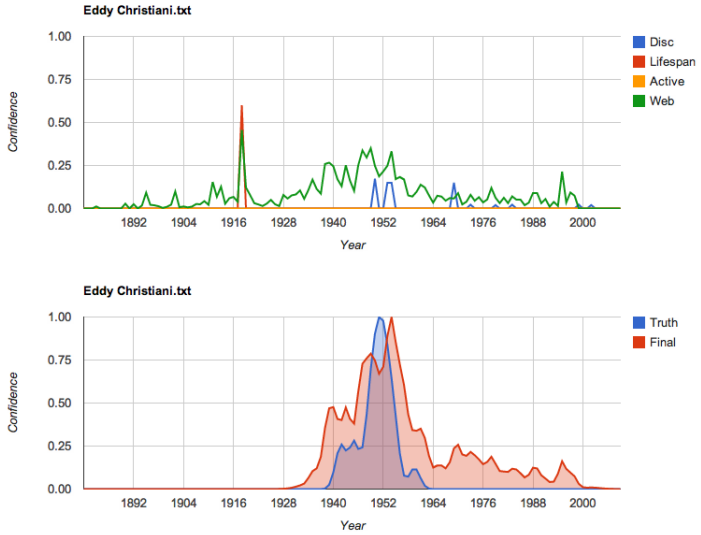

Figure 5.16: (Top)The four input histograms, (bottom) the ground truth against the estimate AYoP for Eddy Christiani.

consider $Accuracy_5$, assuming that this level of detail (or flexibility) is acceptable for artists of the era 1900-1959. However, in a more general scenario, where later artists are considered, a higher level of granularity should be employed.

Therefore, for a 5-year window around 86% of the cases are identified as hits. This high number, in conjunction with the small Mean Error, delineate our method's strength. However, as we shrink the window the accuracy degrades. For example, a 10% decrease is observed when moving from a 5 to a 2-year window, although and accuracy of 76% is still considered high.

Window Size	1	2	3	4	5
Accuracy	0.61972	0.759	0.79499	0.83255	0.86228
Window Size	6	7	8	9	10
Accuracy	0.8748	0.89202	0.91862	0.9374	0.94992

Table 5.7: Accuracy for window size ranging from 1 to 10.

It is worth examining two different representations of the YoR results. Firstly, Figure 5.22 presents the ground truth (axis X) and estimate (axis Y) for each song in the test set. For a perfect algorithm, each song s_j with coordinates (x_j, y_j) should lie on the diagonal. In addition, the regression line (shown in blue) should also coincide with the diagonal. Figure 5.22 additionally depicts the 5 and 10-year window margins. This representation nicely emphasizes the strength of our method; regression line matches nicely the diagonal and the truth-estimate points are distributed along the diagonal with small error.

Figure 5.23 in addition represents the year distribution of the ground truth and estimates. The important thing to notice is the non-uniformal distribution of the test set; most of the date values fall within 1930-1940, which of course slightly reduces the validity of the results. However, we believe that our system can be generalized easily, since no preference towards the 1930-1940 period was

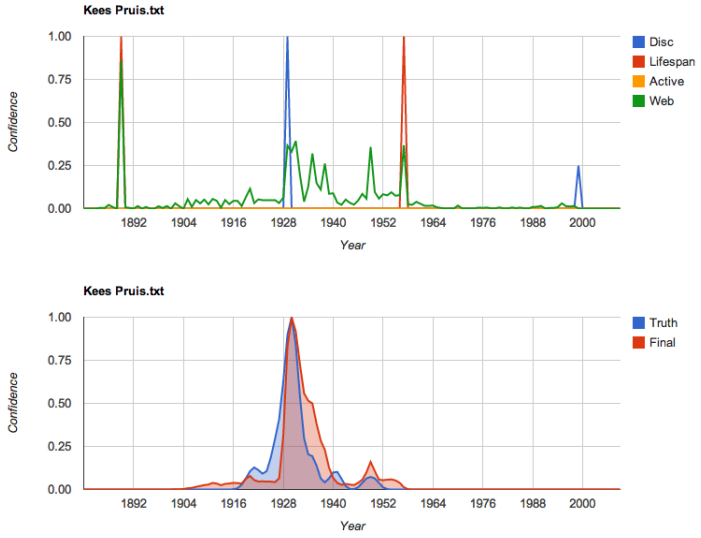

Figure 5.17: (Top)The four input histograms, (bottom) the ground truth against the estimate AYoP for Kees Pruis.

modelled. Whatever the case, future evaluations should include earlier and later dates for the sake of completeness.

It should be mentioned that examples of input and estimate histograms can be found in Appendix.

5.3.6 Discussion and Conclusions

We presented a simple system for determining an artist's era of productivity and a song's year of release. Our approach is based on the exploitation of Editorial Metadata from sources such as MusicBrainz and EchoNest, while also on Web harvested data. The employment of simple heuristics, although debatable, shows room for future improvement and research. The evaluation illuminates the strength of the proposed method for YoR estimation; around 86% of the estimates fall within a $\pm$5-year window, with a mean error of 2.9 years.

The results for determining the artist's productivity are less impressive and the reason is twofold: a) no meaningful evaluation measures were found, b) the ground truth generation assumes complete knowledge of the artist's discography, which is not always the case. However, the results show great potential since artists are well placed into time, and when the discography is complete, modelling the productivity distribution is accurate.

Future implementation will encapsulate larger number of artists, from varying countries and periods of time. The usage of heuristics will be limited by the employment of a well-researched productivity window, that can actually correspond to real life data. The genetic algorithm should be also replaced with a more generalized scheme. Different querying and scoring functions for histogram construction can also be used to improve noise reduction.

To summarise, determining release dates for songs and productivity distributions for artists, is a new MIR task with lots of potential for research and

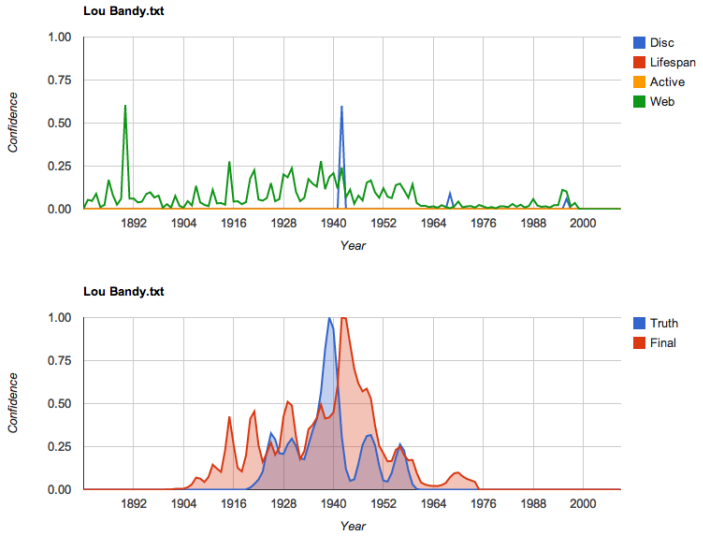

Figure 5.18: (Top)The four input histograms, (bottom) the ground truth against the estimate AYoP for Lou Bandy.

improvement.

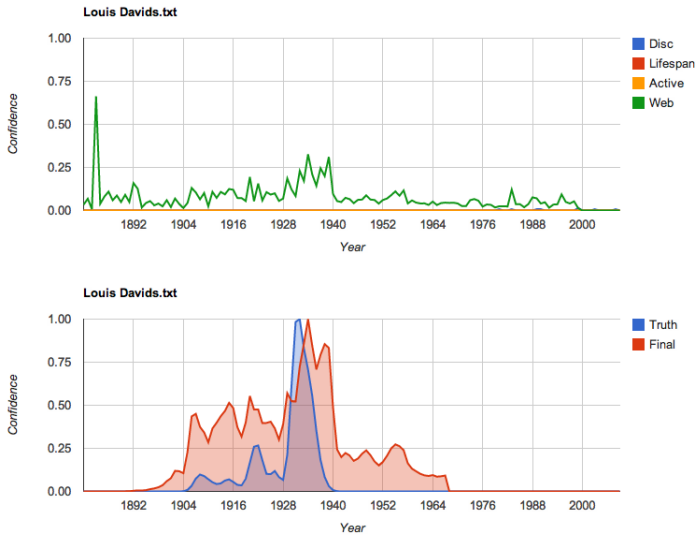

Figure 5.19: (Top)The four input histograms, (bottom) the ground truth against the estimate AYoP for Louis Davids.

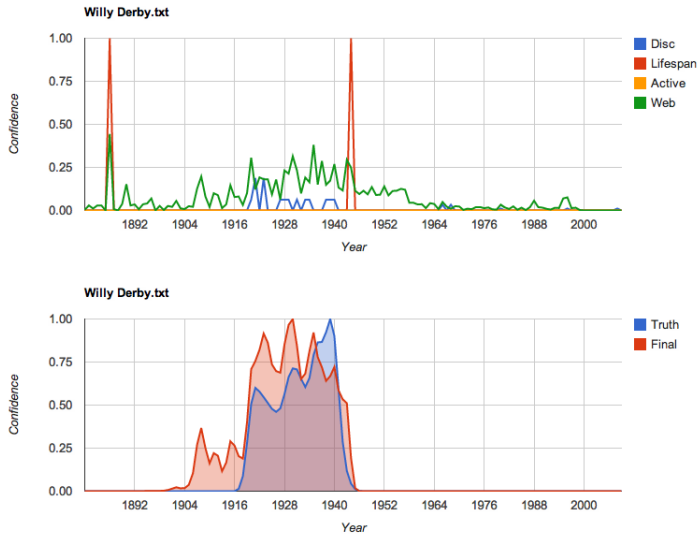

Figure 5.20: (Top)The four input histograms, (bottom) the ground truth against the estimate AYoP for Willy Derby.

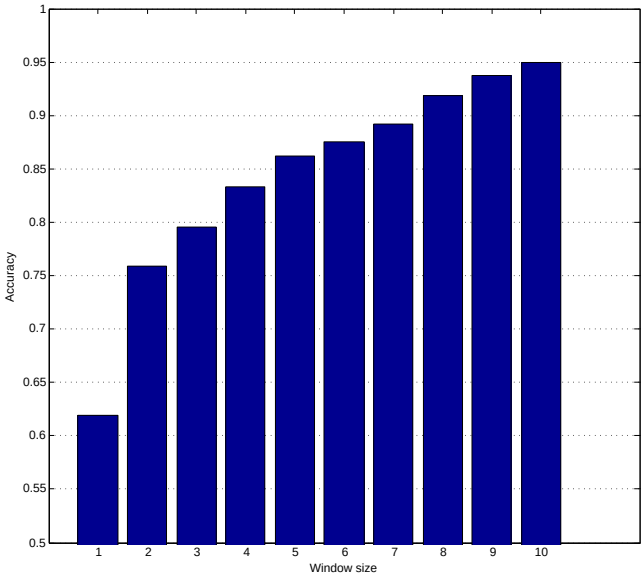

Figure 5.21: Accuracy values for window sizes ranging from 1 to 10.

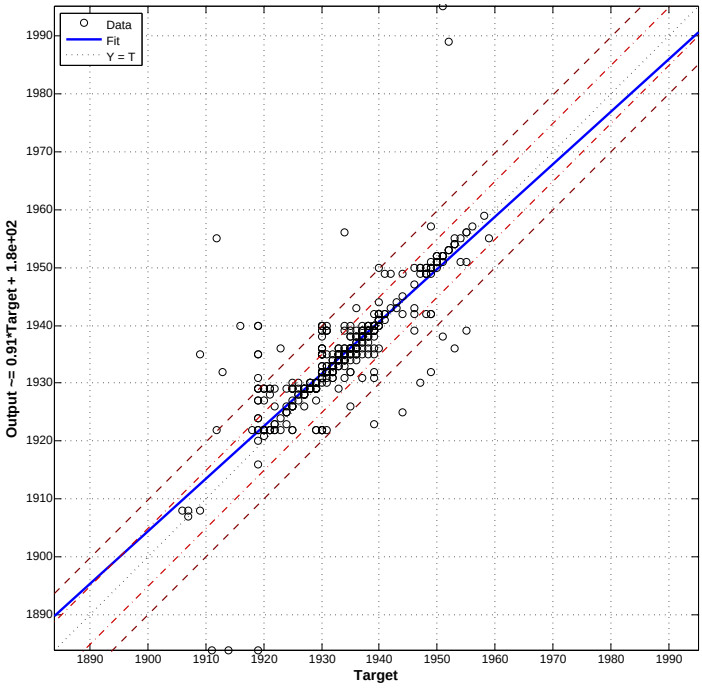

Figure 5.22: The (truth, estimate) couples represented as points on the Cartesian space. A fitted regression line is shown in blue. The dotted red lines represent the 5 and 10-size year windows.

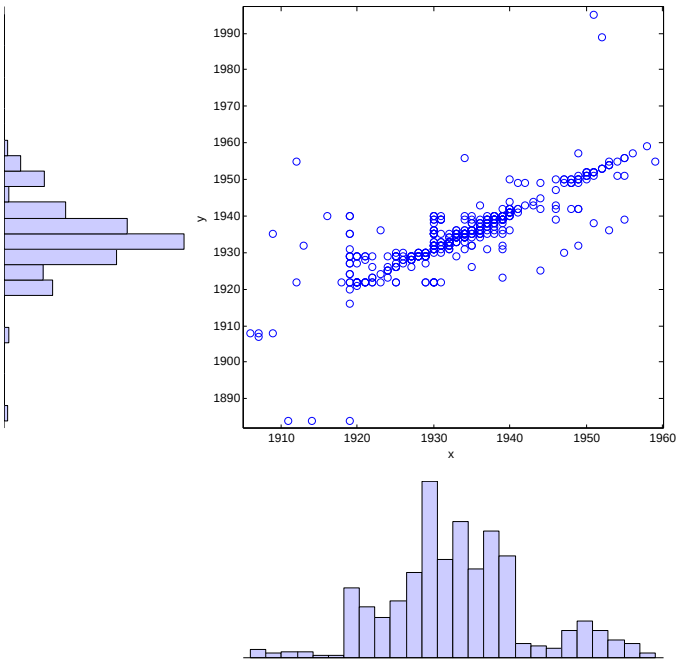

Figure 5.23: The distribution of the truth and estimate values for the whole test set.

Bibliography

[1] C. McKay "Automatic Music Classification with jMIR", PhD Thesis. 2010.

[2] M. Schedl, "Country of origin determination via web mining techiniques", IEEE Proceedingd of the International Conference on Multimedia and Expo (ICME). 2010.

[3] P. Knees, E. Pampalk, G. Widmer, "Automatic Classification of Musical Artists based on Web-Data". OGAI Journal. 2005.

[4] M. Zadel, I. Fujinaga, " Web Services for Music Information Retrieval", Proceedings of the 5th International Symposium on Music Information Retrieval (ISMIR). 2004.

[5] D. Bogdanov, P. Herrera, "Taking advantage of editorial metadata to recommend music", Proceedings of the 9th International Symposium on Computer Music Modeling and Retrieval (CMMR). 2012.

5.4 Appendix

A set of 10 randomly selected examples of input and output histograms for YoR estimation.

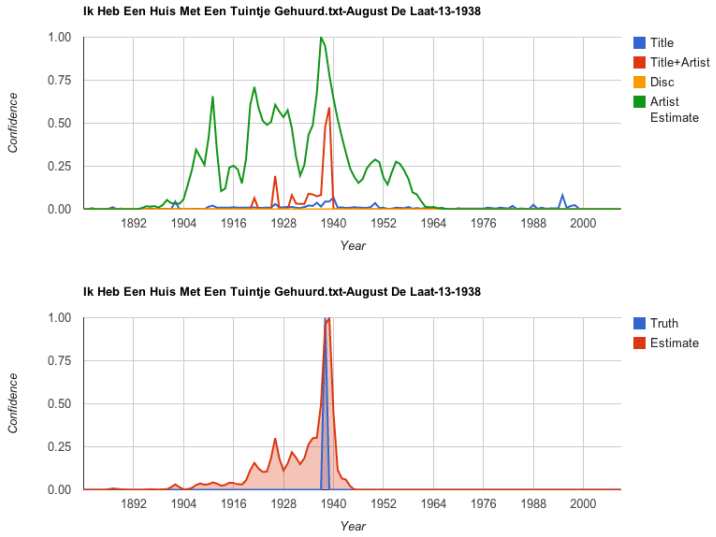

Figure 5.24: (Top)The four input histograms, (bottom) the ground truth against the estimate.

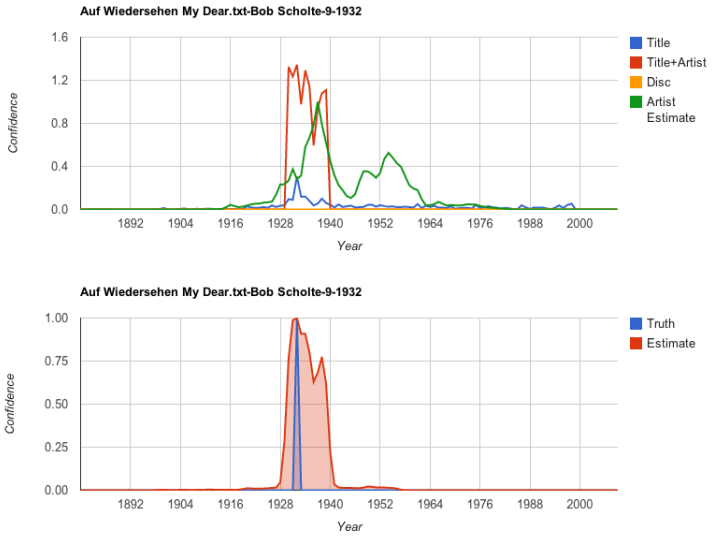

Figure 5.25: (Top)The four input histograms, (bottom) the ground truth against the estimate.

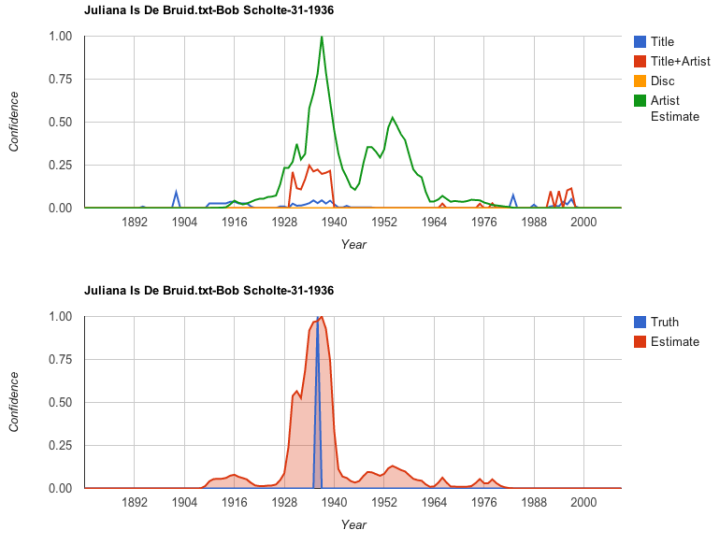

Figure 5.26: (Top)The four input histograms, (bottom) the ground truth against the estimate.

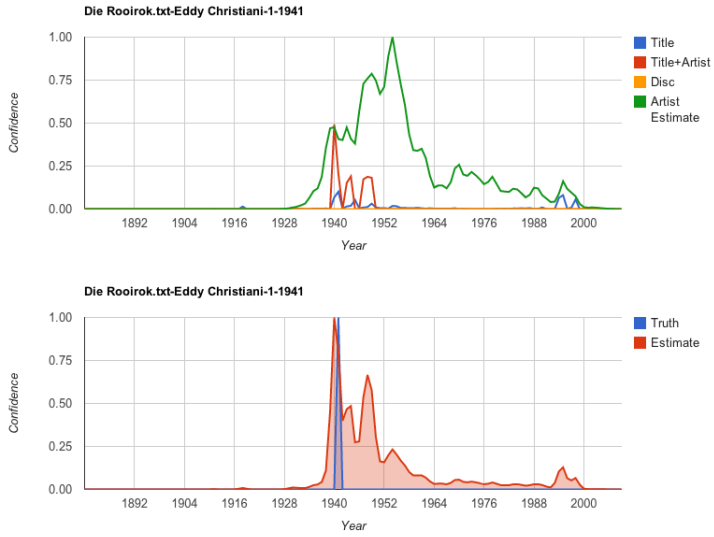

Figure 5.27: (Top)The four input histograms, (bottom) the ground truth against the estimate.

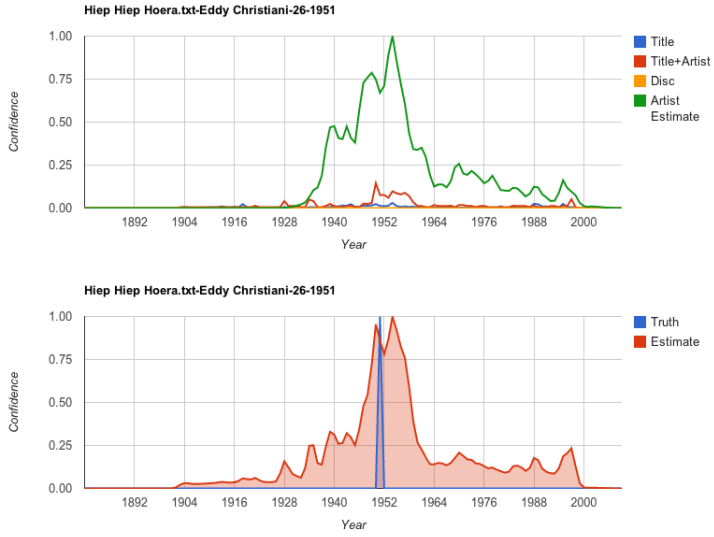

Figure 5.28: (Top)The four input histograms, (bottom) the ground truth against the estimate.

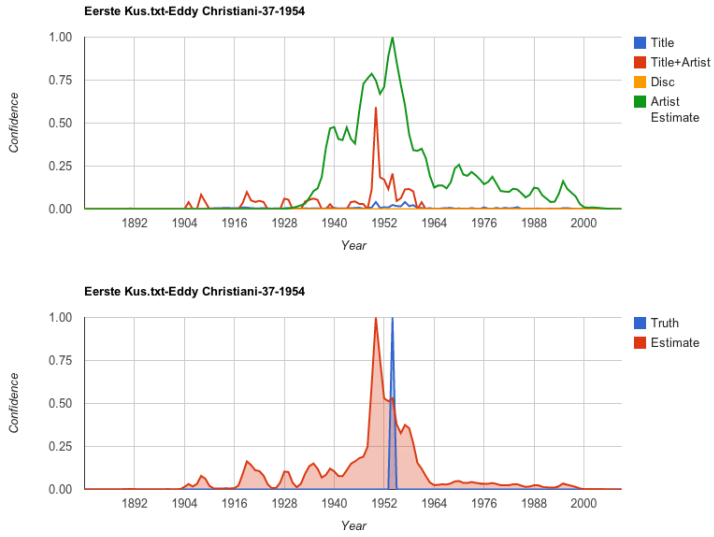

Figure 5.29: (Top)The four input histograms, (bottom) the ground truth against the estimate.

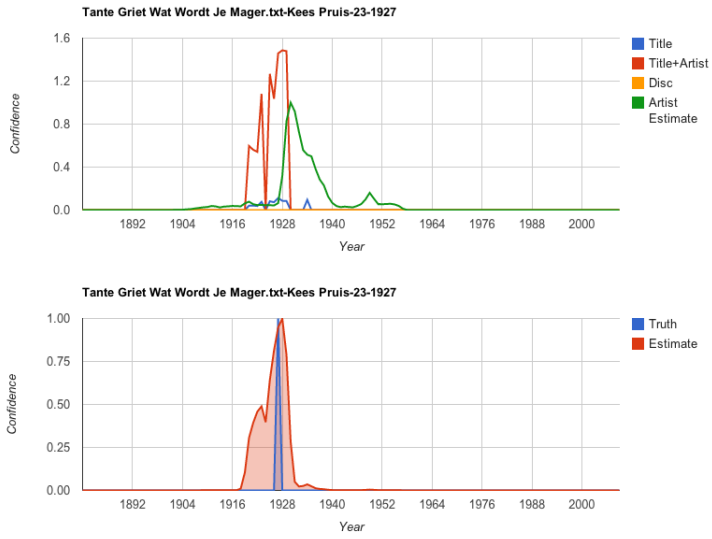

Figure 5.30: (Top)The four input histograms, (bottom) the ground truth against the estimate.

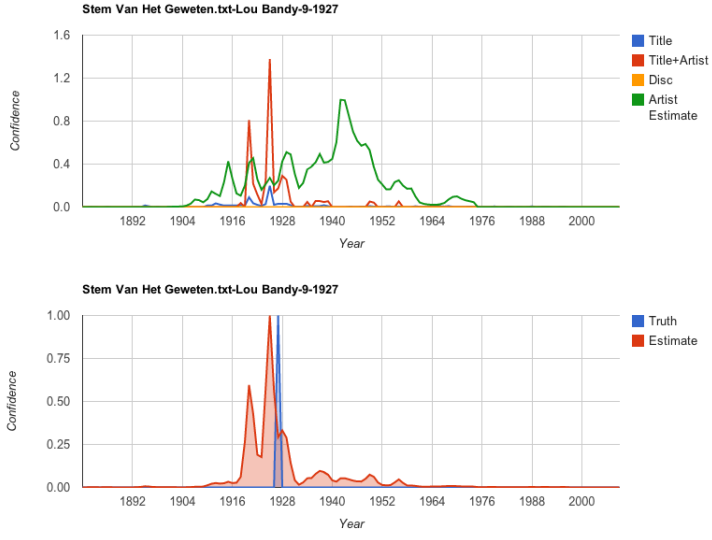

Figure 5.31: (Top)The four input histograms, (bottom) the ground truth against the estimate.

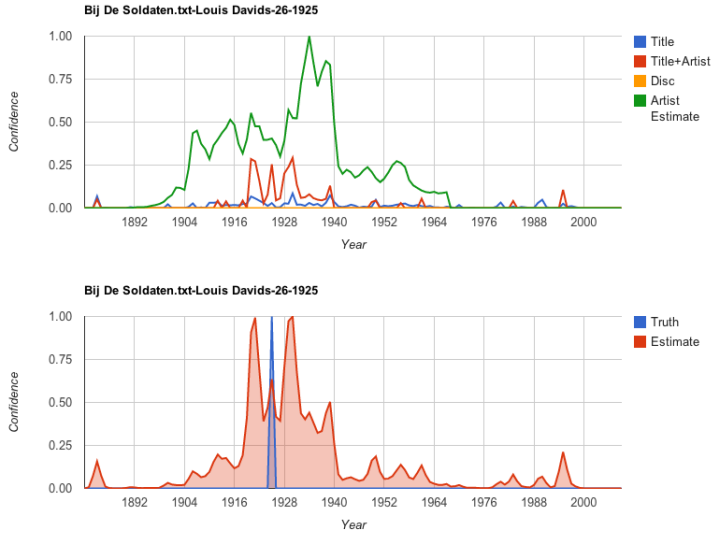

Figure 5.32: (Top)The four input histograms, (bottom) the ground truth against the estimate.

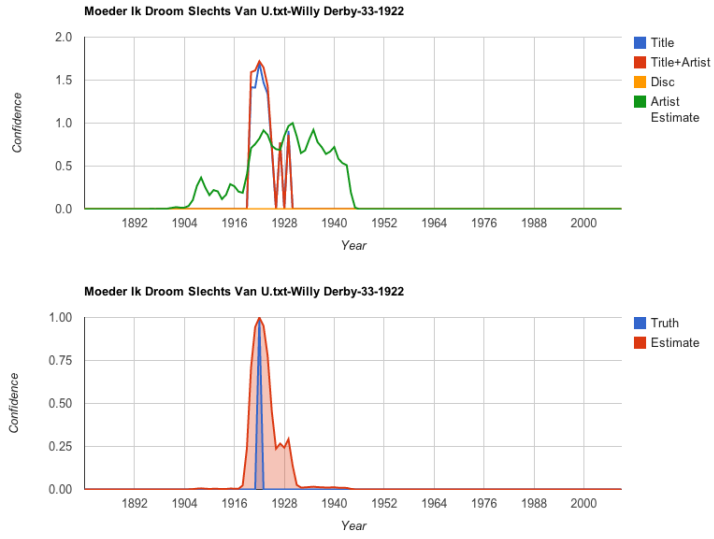

Figure 5.33: (Top)The four input histograms, (bottom) the ground truth against the estimate.

Chapter 6

Data Enrichment: Country of Origin Determination

An artist's country of origin is a substantial piece of information that usualy correlates to his influences and consequently to his style and similarities. The importance of such data becomes more prominent when considering pre-50's eras, when communicating and sharing music outside a country's boundaries was much harder than today. Based on these observations, this report presents a simple method for country of origin determination using editorial metadata and web mining techniques. Evaluating the method on a standardized test-set shows almost 9% higher accuracy than the state of the art.

6.1 Introduction

Similar to the problem of placing music entities in time, country of origin determination allows placing artists in a semantic context. The importance of such a procedure is not apparent for modern music, since currently the distribution of music may exceed a country's boundaries. For instance, people from Turkey or India can be exposed to American music on the same amount as their own. Additionally, searching for artists from less popular countries can be an easy task as a result of the world wide web.

However, music distribution before the existence of the internet and music's commercialization was almost non-existent. Therefore the passing of influences between remote countries was rather difficult. For example, Greek music before the 50's was not influenced by USA, as opposed to early 60's and on. Given these facts, it is safe to state that the country of origin constitutes one of the fundamental factors affecting a pre-50's artist's style and similarities.

6.1.1 Problem Definition

Following Schedl's work [2], we define as "country of origin" the country in which either the performer or musician was born or the band was founded. This definition leads to interesting results considering that band members might have followed a solo career prior the creation or after the disbandment of the band.

The problem of correctly determining the country of origin could have been obsolete if metadata hubs offered complete and reliable information. However, as we have seen before, old and obscure artists are poorly represented in such services. Incorporating the geographical location shows us that artists from non-popular, non-western countries are also misrepresented.

Given that the Web contains vast amount of also non-reliable information, the problem is transformed in finding the optimal way of combining the two sources in a meaningful way.

Formulating CoO Determination

Given a tuple of the form $\langle a_i, c_{a_i} \rangle$ where a_i is an artist and c_{a_i} is his country of origin, we want to find a function f such that $f(a_i, C) = c_{a_i}$, where $C = \{c_1, c_2, .., c_j\}$ a set of countries.

6.2 Method Description

Country of origin determination and time placement rely almost on the same methods. Therefore we will omit any information that has been provided in the previous parts of this thesis. For the sake of completeness we will give brief descriptions of the previously explained notions, however the reader should be forwarded to the initial, corresponding parts for further details.

6.2.1 Sources

Similar to the task of time placement, our system exploits two distinct sources: Music Information Services (MIS) and Search Engines. MIS correspond to (semi) commercial and non-academic online systems and services that utilize or provide technologies related to music databases. Typically, these are Last.FM[1], EchoNest[2] and MusicBrainz[3], however only the latter is employed for CoO determination.

MusicBrainz

A sample artist-search response is shown in Figure 6.1. It is obvious that besides the name, lifespan and disambiguation, MusicBrainz provides the country of origin of the artist. This piece of information is really valuable in our context. However, once again given name ambiguities, it is possible for the top-ranked artist to not correspond to the query. Relying on the MusicBrainz, internal scoring function is not an option, since the top artist is always scored 100. Dealing with that issue will be discussed thoroughly later. It should be noted that EchoNest and Last.fm might also offer such kind of information but not so explicitly. Therefore only MusicBrainz is employed by our method.

[1] www.last.fm
[2] the.echonest.com
[3] www.musicbrainz.org

```
▼<metadata xmlns="http://musicbrainz.org/ns/mmd-2.0#" xmlns:ext="http://musicbrainz.org/ns/ext#-2.0">
 ▼<artist-list offset="0" count="648">
  ▼<artist ext:score="100" type="Group" id="70f5ff67-59c0-4d04-984b-bf0ele602ee1">
     <name>Fred</name>
     <sort-name>Fred</sort-name>
     <country>IE</country>
     <disambiguation>Irish 5-piece alternative/indie-pop band</disambiguation>
   ▼<life-span>
      <ended>false</ended>
     </life-span>
   </artist>
  ▼<artist ext:score="100" type="Person" id="e27d48a1-ba25-45b3-a20e-e2377e5f638e">
     <name>Fred</name>
     <sort-name>Fred</sort-name>
     <gender>male</gender>
     <country>FR</country>
     <disambiguation>French singer/songwriter Frédéric Métayer</disambiguation>
   ▼<life-span>
      <begin>1973</begin>
      <ended>false</ended>
     </life-span>
   </artist>
  ▼<artist ext:score="100" type="Group" id="220b8211-cc4f-44dc-8860-d40c4bdeb95a">
     <name>Fred</name>
     <sort-name>Fred</sort-name>
     <disambiguation>80s UK synth pop</disambiguation>
   ▼<life-span>
      <ended>false</ended>
     </life-span>
   ▼<tag-list>
    ▼<tag count="1">
       <name>uk</name>
```


Figure 6.1: A sample MusicBrainz response for an artist-search query.

6.2.2 General Framework

Given a string representing an artist name a_i, the system's goal is to correctly determine his corresponding country of origin, by just using that string. Similar to the method for time-context placement, the fundamental procedure comprises of three major components:

1. Editorial Metadata retrieval
2. Web Mining
3. Post-processing and estimation

The first step optimally returns a country estimate c_{mb} accompanied by a score value $score_{mb}$, while the second returns ranked list of countries $C_{web} = \{\{c_1, score_1\}, \{c_2, score_2\}, .., \{c_n, score_n\}\}$. During the thirds step those two pieces of information are combined, in order to compute the final country of origin estimate.

6.2.3 Editorial Metadata Retrieval for Country of Origin Determination

The first step in the process is to match a_i to MusicBrainz, in order to extract the $\langle country \rangle$ data. As we have previously seen, MusicBrainz returns a ranked list for search-artist queries. The scores of the list are normalized such that the top ranked artist gets a max score of 100. Optimally, retrieving only the top ranked artist would suffice. However, the MusicBrainz internal name-matching algorithm does not always agree with human intuition. For instance, given the query "Bob Scholten" (which includes a typo), MusicBrainz returns "Grietje

Scholten" at the top and "Bob Scholte" way below with a score of 61 (see Figure 6.2). This fact suggests that the last name is more important; although in terms of complete string distance, "Bob Scholte" is closer to "Bob Scholten" than "Grietje Scholten".

Score	Name	Sort Name
100	Grietje Scholten	Scholten, Grietje
100	Hugo Scholten	Scholten, Hugo
100	Jim Scholten	Scholten, Jim
100	Joop Scholten	Scholten, Joop
100	Joni Scholten	Scholten, Joni
100	Teddy Scholten	Scholten, Teddy
100	Henk Scholten	Scholten, Henk
100	Alexandra Scholten	Scholten, Alexandra
92	Paul Scholten	Scholten, Paul
61	Bob Scholte	Scholte, Bob

Figure 6.2: A returned list for the query "Bob Scholten".

The artist query "Grigor Jeghiasaryan", delineates more vividly the type of erros that may arise using MusicBrainz for name matching. The returned artist names do not agree at all with the query. As a result, the $\langle country \rangle$ field, for any of the top ranked artists, would not be correct.

Our method deals with both aforementioned issues by firstly retrieving all the artists with a score higher than 50. Then, for each artist name, his aliases are gathered and permuted. Each permutation relies on the process of swapping the word order of the alias string. For example given the alias "Willy Derby", its permutations would be {"Derby Willy"}. It is obvious that the number of possible permutations grows exponentially as the number of words in a name increases. Finally, for each alias and its permutations, a string similarity with regard to the query is computed. The similarity function is based on PHP's *similar_text* function, which is described in [1]. Eventually, the maximum of all similarities is considered the confidence value for that artist. For instance, in the previous example "Gerod Grigor" would yield a score of 0.58064, and therefore our confidence that Grigor Jeghiasaryan's country of origin is the same as Gerod Grigor's would be 0.58064.

Score	Name	Sort Name
100	Gerod Grigor	Gerod Grigor
47	Grigory Romanovich Ginzburg	Ginzburg, Grigory Romanovich
47	Grigori Rău	Rău, Grigori
47	Mario Grigorov	Grigorov, Mario
47	Robo Grigorov	Grigorov, Robo
47	Theodore Grigoriu	Grigoriu, Theodore
47	George Grigoriu	Grigoriu, George
47	Grigoras Dinicus	Dinicus, Grigoras
47	Grigoraș Dinicu	Dinicu, Grigoraș

Figure 6.3: A returned list for the query "Grigor Jeghiasaryan".

6.2.4 Web Mining for Country of Origin Determination

This step of the process aims at retrieving all possible web-pages related to an artist a_i and then exploiting them to determine his country of origin. Similar to the time context placement method, we employ the Google Search and Bing API's for retrieving URLs and Apache Nutch for harvesting their content.

Google is queried with string of the form"a_i + music", Bing only with "a_i" while the top 100 and 50 URLs are gathered respectively. After harvesting the

URLs' union for all artists, the gathered web-pages are indexed using Apache Solr. The resulting index is now considered as the input for any of the procedures that will be discussed later.

6.2.5 Generating Country Ranked-List

In order to generate a list containing country name with corresponding scores for each artist, we employ two previously-encountered tools: a) Page counts which correspond to the number of returned indexed Web pages for a given query and b) Solr's relevance score which employs a variation of $tf * idf$ to calculate the query-document similarity $sim(q, d)$.

For each artist a_i and year c_j in C we query Solr with "$a_i + c_j + music$". However, we also introduce a proximity factor of 50, meaning that if the artist name appears more than 50 words later or earlier than the country c_j , then the document pertaining them would be considered irrelevant. Our method assigns a score to each query a_i, c_j using the following two formulas:

$$s^*(a_i, c_j) = mean_{1<k<pc_{a_i,c_j}}[sim(\text{“}a_i + c_j\text{”}, d_k)] \times pc_{a_i,c_j} \tag{6.1}$$

and

$$s^*(a_i, c_j) = pc_{a_i,c_j} \tag{6.2}$$

Intuitively, the first $s^*()$ function corresponds to the product of the page counts and the mean relevancy, while the second one to just the page counts.

However, following Govaerts and Schedl work [2, 3], querying with only country names has proven inadequate. This relies mostly on the fact that stating an artist's origin on the Web is not always explicit. For example, it is quite common to say "Queen is a British band" rather than "Queen is a band from UK". Therefore both researchers have employed demonyms and synonyms, although the distinction between them is not very apparent. Our method querys Solr with "a_i+d_k^j" where $d_k^j \in D^{c_j}$ and D^{c_j} the set of demonyms/synonyms[4] for that particular country. The country's final score is then computed using two different formulas:

$$score(a_i, c_j) = s^*(a_i, c_j) + \sum_{d_k^j \in D^{c_j}} s^*(a_i, d_k^j) \tag{6.3}$$

and

$$score(a_i, c_j) = s^*(a_i, c_j) + \frac{\sum_{d_k^j \in D^{c_j}} s^*(a_i, d_k^j)}{|D^{c_j}|} \tag{6.4}$$

Therefore, (6.3) simply aggregates the scores of each demonym while (6.4) takes their mean.

6.2.6 Post-processing and Estimation

After the previous steps, the system has acquired a country estimate c_{mb} accompanied by a score value $score_{mb}$, and a ranked list of countries $C_{web} = \{\{c_1, score_1\}, \{c_2, score_2\}, .., \{c_n, score_n\}\}$. Before we merge those two, we first

[4] www.cp.jku.at/-people/schedl/music/countries_syn.txt

normalize the latter so that its maximum score is 1. We then find the country in the ranked list that corresponds to c_{mb} and increase its score by $score_{mb}$. The country with the highest score is considered the artist's country of origin.

6.3 Experiments and Evaluation

6.3.1 Test Collection and Evaluation Measures

The problem of determining the country of origin has been addressed by both Govaerts and Schedl. However, only the latter offers a freely available test set. It contains 578 artists from 69 countries, gathered from sources such as Wikipedia, Last.fm and Allmusic.com.

Schedl employs the measures of precision, recall and F-measure. Recall, in our context of use, denotes the percentage of artists for which a country of origin could be determined. However, in both our method and Schedl's, recall is always 100% and therefore, calculating the F-measure is futile. Precision constitutes the main measure of evaluating our method.

6.3.2 Results

In order to examine our method's strength we determine the country of origin using three approaches: a) Web information only, b) MusicBrainz information only and c) the proposed method of fusing both with demonyms score aggregation(*sum*) and mean (*avg*). We also experimented with the two scoring functions 5.9, 5.10 while also with the exclusion of terms such as "tour". The latter is based on the assumption that many country names appear along with artists in documents containing information about the artist's tour.

Method	Scoring	Demonyms score	Excluding terms	Precision
Web	6.1	sum	-	0.6130
MusicBrainz	-	-	-	0.5355
Web+MusicBrainz	6.1	sum	-	0.7833
Web+MusicBrainz	6.2	sum	-	0.7573
Web+MusicBrainz	6.1	avg	-	0.8024
Web+MusicBrainz	6.1	avg	tour	0.7920

Table 6.1: Precision values for the different experimental configurations.

It becomes obvious that the fusion of Web mined and MusicBrainz data yields the highest performance. The precision of the best configuration is actually almost 8% higher than Schedl's best, which delineates out method's strength. However, since Schedl's approach employs only Web mining techniques, its superiority as compared to ours is obvious. In other words, if only the Web techniques were compared, our method would have been ranked second. Taking this a step further, if our method employed Sched's relevancy function, then the fusion of techniques might have yielded much higher precision.

6.3.3 Discussion and Conclusions

We presented a simple method for country of origin determination, that employs a fusion of editorial metadata and Web mining techniques. The evaluation on

a standardized test collection showed that our method performs significantly better than the state of the art. However, there is room for further improvement since our Web mining procedure is inferior to Schedl's. Therefore, future research will investigate the integration of more sophisticated relevancy functions. To summarize, our method's contribution relies on the fusion of sources which proves that the whole is better than the parts it pertains.

Bibliography

[1] I. Oliver, "Programming Classics: Implementing the World's Best Algorithms", 1994.

[2] M. Schedl, C. Schiketanz and K. Seyerlehner. "Country of origin determination via web mining techiniques". In Proceedings of the IEEE International Conference on Multimedia and Expo (ICME-2010): 2nd International Workshop on Advances in Music Information Research. 2010.

[3] S. Govaerts, E. Duval, "A Web-based approach to determine the origin of an artist", Proceedings of 10th International Society for Music Information Retrieval (ISMIR). 2009.

Chapter 7

Data Enrichment: Genre Estimation

Genre constitutes the most fundamental classification feature for music entities and especially artists. A simple visit on the Web would show vividly that most of today's music recommendation systems base their services on the stylistic information about artists. Even though some musicologists and music purists would argue regarding most of genre-related factors (eg. groups, sub-genres, relation to reality), it remains a fact genre is a valuable piece of information. This chapter of the thesis describes our approach not only in terms of tools but also in terms of formulating the problem itself.

7.1 Introduction

Genre-based artist classification and genre estimation are topics well studied inside the audio domain. Describing the published methodologies exceeds the scope and main interest of this study, however it is worth mentioning the related MIREX competition. Music Information Retrieval Evaluation eXchange[2], is an an annual event where various MIR algorithms are evaluated and compared in a series of research domains including genre classification. Its importance, in our context, relies on the publicly available training-set (or ground truth), which includes audio tracks accompanied by genre tags (eg. "rock", "pop"). By going through the audio, it becomes obvious that tags often fail to capture the stylistic variations of a song. Music can be both "pop" and "rock, or even "jazz" and "electronic".

Genre estimation based on metadata is a relatively new field. We have presented certain approaches in the literature survey that employ tools ranging from community metadata to social media. Now, most of the published evaluations employ a ground truth set that is either publicly available or compiled for the particular problem in hand. In an ideal situation the public, well documented benchmarks would suffice for fair evaluation, however certain issues arise:

- Genre tags, across different ground-truth sets, are inconsistent with each other. This makes perfect sense considering the lack of a universal genre-

measure or scale. For example, what is it that makes a band "rock" instead of "pop"? Even though each individual has its own internal genre classification function, it is a fact that most times they don't agree with each other.

- Artists and songs cannot be represented by just a single tag even if it contains more than one term (eg. "alternative rock").
- The orthogonality of genres is debatable. For example, how is "rock" orthogonal to "alternative", or "pop" orthogonal to"rock"?

Based on these observations, we believe that genre estimation has an ill-posed problem definition. Ground truth sets capture just one of the possible perspectives. Genre terms on the other hand are confusing while also many times overlapping with each other. Therefore, increasing the genre classification accuracy of any state-of-the-art methods, is just a matter of over-fitting to the particular problem and its manifestations.

On top of that, it is worth pointing out the inconsistency of genre terms throughout time. In other words, the perception of a term in the current cultural context may disagree with its perception 50 years ago. For example, the music artist Jack Payne (1899 - 1969) is considered a "dance" artist of the "British dance band" era. Does the "dance" term of that era correspond to our current perception? Certainly not, therefore each genre estimation method should take that factor into account.

Genre definition In our context, genre denotes to a set of features, corresponding to typical genre terms. In other words, genre is represented as a profile that encodes the relation of an artist to each particular style. To formulate this a bit more, we denote as G a set of genre terms e.g. "rock", "pop", "jazz" and H^{a_i} the genre profile for each artist a_i, such that:

$$\sum_{g \in G} H^{a_i}(g) = 1 \tag{7.1}$$

An example of a genre-profile, which we will call just profile from now on, is shown in Figure 7.1.

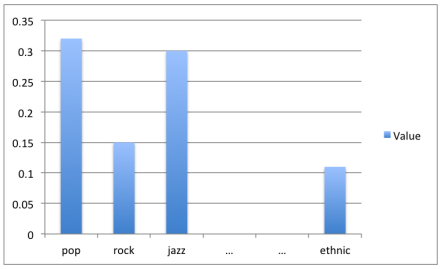

Figure 7.1: An example of a genre-profile.

7.2 Method Description

Similar to time placement and country of origin determination, genre estimation relies on editorial metadata and web-mining techniques. Therefore, once again we will be omitting any information previously provided, and focus on the problem in hand.

7.2.1 Sources

Similar to the previous tasks, our method uses Music Information Systems and Search Engines. The latter correspond to Google's and Bing's search API, while the first to MusicBrainz and Last.fm.

MusicBrainz Figure 7.2 shows an XML sample response for a MusicBrainz artist-search query. Clearly, no genre information is explicitly provided. However, a list of user tags is present. Tags are not picked from a pre-defined pool, but they are rather free-text. They are also accompanied by a number corresponding to counts. In our example, the most popular tag for the band "Metallica" is "thrash metal", which actually provides some kind of genre-information. However, since we are dealing with free-text tags, not all of them are genre-related or even marked by good intentions (see tag "douchebag metal").

Last.fm Last.fm's API, similar to MusicBrainz, does not provide genre - information but offers the query "artist.getTopTags", which returns a list of the top user-defined tags for the artist in hand. An example is shown in Figure 7.3. By examining the XML response it becomes apparent that last.fm, in contrast to MusicBrainz, normalizes the tag counts. Therefore for the artist "Metallica" the top tag is "trash metal", although the precise number of users who tagged Metallica as such is unknown.

7.2.2 General Framework

Given a string representing an artist name a_i, the system's goal is to estimate his genre profile, simply by using that string. Similar to the previous methods, the general procedure comprises of three major components:

1. Editorial Metadata retrieval
2. Web Mining
3. Post-processing and estimation

The first two steps optimally return profiles of the form:

$$H_{a_i} = \{\{genre_1, score_1\}, \{genre_2, score_2\}, .., \{genre_n, score_n\}\}. \quad (7.2)$$

During the third step those two pieces of information are fused, for the final profile to be computed.

```
▼<metadata xmlns="http://musicbrainz.org/ns/mmd-2.0#" xmlns:ext="http://musicbrainz.
 10-30T07:34:12.753Z">
 ▼<artist-list count="1" offset="0">
  ▼<artist id="65f4f0c5-ef9e-490c-aee3-909e7ae6b2ab" type="Group" ext:score="100">
     <name>Metallica</name>
     <sort-name>Metallica</sort-name>
     <country>US</country>
    ▼<life-span>
       <begin>1981-10</begin>
       <ended>false</ended>
     </life-span>
    ▼<alias-list>
       <alias>메탈리카</alias>
       <alias>UN3 (The Unforgiven III).mp3</alias>
       <alias>Flamingo (All Nightmare Long).mp3</alias>
       <alias>German Soup (Cyanide).mp3</alias>
       <alias>メタリカ</alias>
       <alias>Neinteen (The End Of The Line).mp3</alias>
       <alias>Metalica</alias>
       <alias>K2LU (Suicide & Redemption).mp3</alias>
       <alias>Ten (My Apocalypse).mp3</alias>
       <alias>Black Squirrel (Broken, Beat & Scarred).mp3</alias>
       <alias>Gymbag (The Judas Kiss).mp3</alias>
       <alias>Casper (The Day That Never Comes).mp3</alias>
     </alias-list>
    ▼<tag-list>
     ▼<tag count="9">
        <name>thrash metal</name>
      </tag>
     ▼<tag count="1">
        <name>américain</name>
      </tag>
     ▼<tag count="1">
        <name>usa</name>
      </tag>
     ▼<tag count="2">
        <name>speed metal</name>
      </tag>
     ▼<tag count="1">
        <name>douchebag metal</name>
      </tag>
     ▼<tag count="1">
        <name>american thrash metal</name>
      </tag>
     ▼<tag count="3">
        <name>rock</name>
      </tag>
     ▼<tag count="1">
        <name>90s</name>
      </tag>
     ▼<tag count="1">
        <name>80s</name>
      </tag>
     ▼<tag count="1">
        <name>seen live</name>
      </tag>
     ▼<tag count="1">
        <name>rock and indie</name>
      </tag>
```


Figure 7.2: An example of a MusicBrainz artist-search response.

7.2.3 Genres and Related Terms

Before we go deep into the method's details, we present our view on the division of genres. First of all, we consider intra-genre divisions useless, at least in out context. Considering that we are dealing with massive amount of genre-varying artists, a separation of jazz into "jazz dixielan" and "jazz post-bop" is meaningless.

Our genre terms are partially derived from the CODAICH test set [1]. CODAICH contains 15 coarse genre categories, which are subdivided to even more parts. The genre breakdown is shown below.

```
Alternative Pop / Rock
Blues - Contemporary Blues
Blues - Country Blues
Blues - Urban Blues
Classical - 20th Century Classical
Classical - Baroque
Classical - Classical
Classical - Renaissance & Med.
Classical - Romantic
Country
Dance Pop
Electronica
Hip Hop / Rap
Instrumental Pop
Jazz - Acid Jazz
Jazz - Avant-Garde Jazz
Jazz - Bebop
Jazz - Cool Jazz
Jazz - Dixieland
```

```
Jazz - Hard Bop
Jazz - Latin Jazz
Jazz - Post-Bop
Jazz - Soul Jazz
Jazz - Swing
Modern Folk - Alternative Folk
Modern Folk - Singer / Songwriter
R&B - Contemporary R&B 285
R&B - Funk
R&B - Gospel
R&B - Rock & Roll
R&B - Soul
Reggae
Rock - Alternative Metal / Punk
Rock - Classic Rock
Rock - Metal
Rock - Roots Rock
Spoken
World - African
World - Americas
World - Arabic
World - Asian
World - Calypso
World - Celtic
World - Chanson
World - Cuban
World - European
World - Flamenco
World - Fusion
World - Gypsy
World - Indian
World - Klezmer
World - Latin American
World - Mixed Traditional
World - Tango
World - U.S. Traditional
```

```
▼<lfm status="ok">
  ▼<toptags artist="Metallica">
    ▼<tag>
       <name>thrash metal</name>
       <count>100</count>
       <url>http://www.last.fm/tag/thrash%20metal</url>
     </tag>
    ▼<tag>
       <name>metal</name>
       <count>91</count>
       <url>http://www.last.fm/tag/metal</url>
     </tag>
    ▼<tag>
       <name>heavy metal</name>
       <count>72</count>
       <url>http://www.last.fm/tag/heavy%20metal</url>
     </tag>
    ▼<tag>
       <name>hard rock</name>
       <count>39</count>
       <url>http://www.last.fm/tag/hard%20rock</url>
     </tag>
    ▼<tag>
       <name>rock</name>
       <count>32</count>
       <url>http://www.last.fm/tag/rock</url>
     </tag>
    ▼<tag>
       <name>metallica</name>
       <count>10</count>
       <url>http://www.last.fm/tag/metallica</url>
     </tag>
    ▼<tag>
       <name>speed metal</name>
       <count>7</count>
       <url>http://www.last.fm/tag/speed%20metal</url>
     </tag>
    ▼<tag>
       <name>classic rock</name>
       <count>5</count>
       <url>http://www.last.fm/tag/classic%20rock</url>
     </tag>
    ▼<tag>
       <name>american</name>
       <count>5</count>
       <url>http://www.last.fm/tag/american</url>
     </tag>
```

Figure 7.3: An example of a Last.fm response for the query artist.getTopTags.

From those, our approach keeps only the high level categories. Based on our intuition the coarse categories were modified and are as follows: alternative, *pop, rock, blues, classical, electronica & dance, hip-hop & rap, jazz, folk, R & B, funk & soul, reggae, metal, punk, world music.*

As we shall see later, it is quite common for genres categories to not explicitly appear in web-documents or tags. For example, it is uncommon for the term

Genre	Terms
Alternative	brit-pop, british pop, alternative pop, indie, britpop, contemporary pop, alternative dance, alternative rock
pop	ballad, candy-pop, teen pop, pop hit, dance pop, AM pop
rock	hard rock, rock & roll, rock 'n' roll, british psychedelia psychedelic, garage, blues-rock, surf
blues	contemporary blues, country blues, urban blues, modern electric blues, soul-blues, memphis blues, early blues, early r&b
classical	baroque, renaissance, medieval, romantic film score, ensemble, orchestra, conductor, winds, strings, soprano, tenor, ballet
electronic & dance	club, trance, progressive trance, electronic disco, house, acid house, techno, synths, big beat, triip-hop dance, synthesizer, drum beat
hip-hop & rap	hip hop, rap, east cost rap, west coast rap hardcore rap, pop rap, gangsta rap
jazz	acid jazz, avant-garde jazz, bebop, dixieland, hard bop latin jazz, post-bop, soul jazz, swing, trumpet jazz modal jazz, crossover jazz, fusion, guitar jazz
folk	modern folk, celtic folk, irish folk, traditional irish folk contemporary celtic, country music, bluegrass, folk-rock folk-pop
R&B	r'n'b, rhythm and blues, r and b, contemporary r&b
funk & soul	soul, psychedelic soul, chicago soul, blaxploitation uptown soul, funk metal, punk-funk
reggae	political reggae, jamaica, roots reggae, dub, ragga rocksteady, soca
metal	heavy metal, death metal, progressive metal, gothic metal trash metal, doom metal, power metal, riff, lead guitar blastbeat, black metal
punk	hardcore punk, punk/new wave, L.A. punk, new york punk british punk, ska, oi!, power-pop, straight edge
world music	ethnic, african, asia, calypson, celtic, chanson, cuban gypsy, latin, tango, flamenco

Table 7.1: Coarse genres and their related terms. We assume that the latter, when appearing inside a document, highly correlate to the coarse genre.

"world music" to appear in an artist's biography. It is rather more probable that term "ethnic" or "latin" to appear instead. Based on this observation we have identified certain terms, that may appear in a document or tag, and are highly related to the genres themselves. These are presented in Table 7.1.

The terms are a concatenation of CODAICH's sub-genres and AllMusic's genre divisions[1]. By studying the table, it becomes clear that certain terms appear in more that one genres. For example the term "power-pop" contributes to both pop and punk. This is acceptable in our context, since we assume no orthogonality between genres.

7.2.4 Editorial Metadata Retrieval for Genre Estimation

The first step of the whole process is to generate a genre-profile for each artist, based on the tags of MusicBrainz and Last.fm. In order to do that we first have to ensure that the returned tags correspond to the correct artists. This is achieved by using the same procedure as described in 5.2.3. The outcome of

[1] http://www.allmusic.com/

that function is two values c_{mb} and c_{last} representing our confidence that the artist returned and the query-artist are the same (for MusicBrainz and Last.fm respectively). For each artist we merge the corresponding tag lists, while first ensuring that the MusicBrainz one is normalized, similar to Last.fm.

Now, we create an empty genre profile for each artists e.g:

$$H_{a_i} = \{\{alternative, 0\}, \{pop, 0\}, ..., \{worldmusic, 0\}\} \tag{7.3}$$

For each tag $t \in T$ we check whether each of the genre terms $g \in G$ appear inside. If so, then the corresponding genre coefficient is updated as follows:

$$H_{a_i}(genre) = H_{a_i}(genre) + value(t) \times c \tag{7.4}$$

where $value(t)$ the normalized tag counts, and c the confidence value (either c_{mb} or c_{last}). The whole function is represented in Figure 7.4.

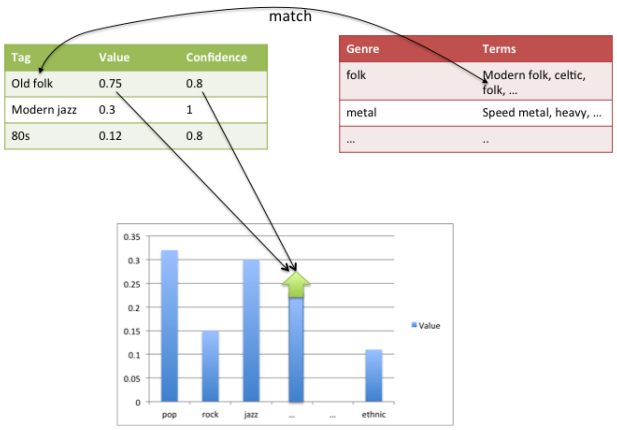

Figure 7.4: An example of profile updating using tags. The genre term "folk" appers in the tag "old folk" therefore the "folk" coefficient of the artists profile is increased.

7.2.5 Web Mining for Genre Estimation

For each artist $a_i \in A$ we query Google with the scheme "a_i+music" and retrieve the 100 top-ranked URLs, denoted as set G. We query Bing with only a_i, and retrieve a set B of 50 URLs per artist. We later compute $U_{seed} = B \cup G$ and retrieve the web-pages in U_{seed} using the Apache Nutch web crawler. Once again an index is built using Apache Solr.

The next step in the process is to calculate genre-histograms for each artist based on the indexed data. We aim at generating a distribution that would best model the artist's genre-variations as they are documented on the Web. We perform that by employing the same set of tools as in the previous tasks: page counts, co-occurrence analysis (or cross-tabulation analysis) and $tf * idf$-relevance-scoring as implemented by Lucene.

For each artist a_i and genre term $g_j \in G$ we query Solr with "$a_i + g_j$", while also introducing a proximity factor of 200 (only pages where the word distance $d(a_i, y_j)$ is smaller than 200 are returned).

Our approach assigns a score to each query a_i, g_j using the following cross-tabulation formula:

$$s^*(a_i, g_j) = (\frac{score(a_i, g_j)}{score(a_i)} + \frac{score(a_i, g_j)}{score(g_j)}) \times 0.5 \tag{7.5}$$

where:

$$score(a_i, g_j) = \frac{\sum_{1<k<pc_{a_i,g_j}} sim(\text{“}a_i + g_j\text{”}, d_k)}{pc_{a_i,g_j}} \times pc_{a_i,g_j} \tag{7.6}$$

Intuitively, the $score()$ function corresponds to the product of the page counts and the mean relevancy of the returned documents. Eventually, the genre profile is computed as such:

$$H_{a_i}(genre) = \frac{\sum_{g_f \in genre} s^*(a_i, g_f)}{|genre|} \tag{7.7}$$

simply meaning that the coefficient value of each coarse genre corresponds to the mean of the scores of the genre-terms comprising it.

7.2.6 Post-processing and Estimation

By the end of the previous process the method acquired two genre profiles; one derived from user tags, denoted H^t and one from the Web denoted H^w. Now we aim at fusing the two profiles into one. Prior to any fusion, both profiles are normalized such that their sum of coefficients is 1. The final genre-profile is just the weighted sum of the profiles:

$$H^f = \frac{(1-w) \times H^w + w \times H^t}{2} \tag{7.8}$$

Examples of estimated genre-profiles are shown in the figures below. It is worth discussing them starting from Figure 7.5, representing the artist Nine Inch Nails (NIN). The estimated profile captures precisely the stylistic variations of the band. NIN are an alternative pop/rock band with many electronic and metal elements. The CODAICH ground-truth for NIN is “Rock - Alternative Metal Punk”, an inadequate representation according to our own perspective.

Figure 7.6 represents the artist Shakira, who frequently uses Latin elements in her music. The user-tags failed to capture that fact, but the final profile incorporates it nicely. Shakira in general is a pop artist with electronic, dance and hip-hop elements. The high value of the “rock” coefficient is surely debatable, but the definition of “rock” varies from person to person. CODAICH defines Shakira as “Dance - Pop” artist.

Figure 7.7 represents the artist “The Prodigy”. According to our own view, Prodigy are an electronic band with lots of rock elements. Although our estimate assumes that the rock part of Prodigy is more prominent, it is still a very good representation of the artist’s genre-variation. CODAICH tags Prodigy as “electronica”.

Finally, let us have a look at another “Dance Pop” artist (according to CODAICH). Rihanna (Figure 7.8) based on our knowledge, is a pop artist with many dance, R& B and hip-hop elements. This is very well captured in the estimate genre-profile. The comparison between Shakira and Rihanna, shows us

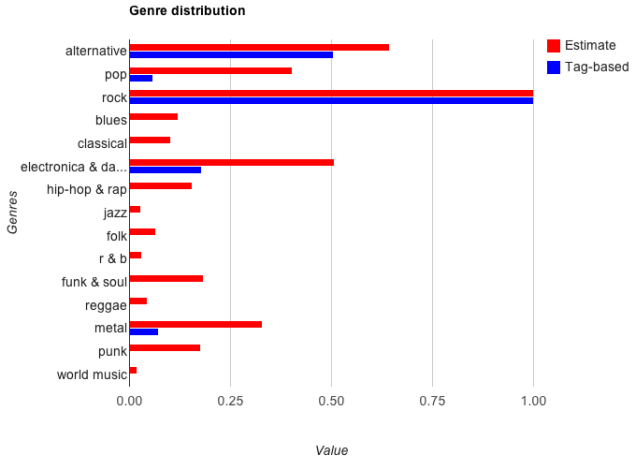

Figure 7.5: The estimated genre-profile for the artist Nine Inch Nails (red columns). The H^t profile based on tags is shown in blue.

that the term "Dance - Pop" is too limited to incorporate the stylistic differences of the two artists.

It is worth pointing out certain issues that arise when looking at the figures. Firstly, the term "alternative" appears prominently in most examples. This was actually expected since the definition of "alternative" is pretty vague. Secondly, for all examples, there are no coefficients with zero values which proves that our method is susceptible to noise. However, in most cases the "signal-to-noise" ratio is high enough to capture the artist's style.

7.3 Experiments and Evaluation

Due to our digression from the typical formulation of the genre-estimation problem, evaluating our method is rendered impossible. This is also supported by the fact that our genre divisions do not agree with any available test-sets. However, in the next paragraphs we will present simple evaluating scheme that helps us understand the method's strength and weaknesses.

7.3.1 Test-set & Evaluation Measures

For our simple experiment we use a subset of 200 random artists from the CODAICH test-set. As we have previously seen, this particular test-set assigns a genre tag of the form "Genre - Sub genre 1, Sub Genre 2..." to each artist. There are even cases, e.g. "Dance Pop", where the main genre can be divided into two. Based on these observations, we generated a mapping from the CODAICH ground truth genre-division to ours. For example an "Alternative Pop/Rock" artist would be assigned three tags, namely "alternative", "pop" and "rock". An "R&B - Funk" artist would be assigned two tags: "R&B" and "Funk".

Our experiment employs the measures of 1st and 2nd Tier. Given c the number of relevant items and v the number of visible or correctly retrieved

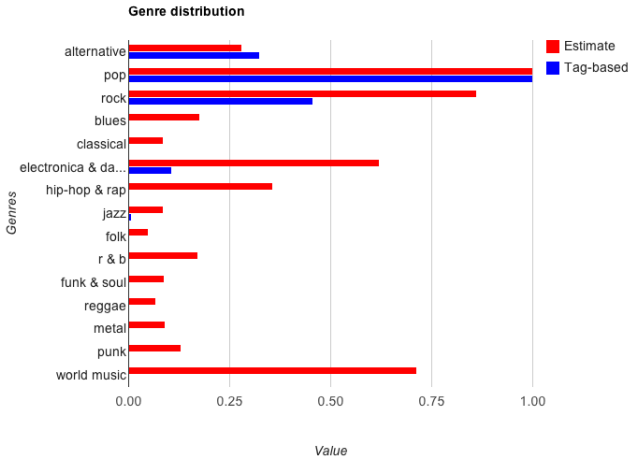

Figure 7.6: The estimated genre-profile for the artist Shakira (red columns). The H^t profile based on tags is shown in blue.

items, k-th Tier is defined as follows:

$$\frac{v}{k(c-1)} \times 100 \tag{7.9}$$

Tiers are typically employed in retrieval tasks, therefore the -1 ensures that the retrieved query will not be taken into consideration. In our scenario though the denominator becomes $k \times c$. For instance, if our method top-ranked the genre terms "alternative" ,"metal" and "jazz" while the ground-truth was "alternative","pop" and"rock", the 1st Tier would be 1/3 or 33%.

7.4 Results

Table 7.2 presents the 1st an 2nd Tier for different profile-addition weightings. At first sight, the 1st-Tier results support the expected; our genre estimation frequently disagrees with CODAICH. As a consequence, it is more meaningful to investigate the effect of summing the individual profiles (Web and Tag based). In both cases, where the Tag and Web profiles are neglected respectively, the results show mediocre performance. In simpler words, if we choose to use either Web mining or user tags, the performance would be lower than a combination of those.

To conclude, although our method lacks substantial evaluation, it yields results that well-agree with our intuition. We believe that the genre-profiles are better representation of an artist, while at the same time offering the possibility of computing genre-distances in a continuous domain. This characteristic is crucial for the next part of our thesis, which aims at computing artist similarity based on era, location and genre features.

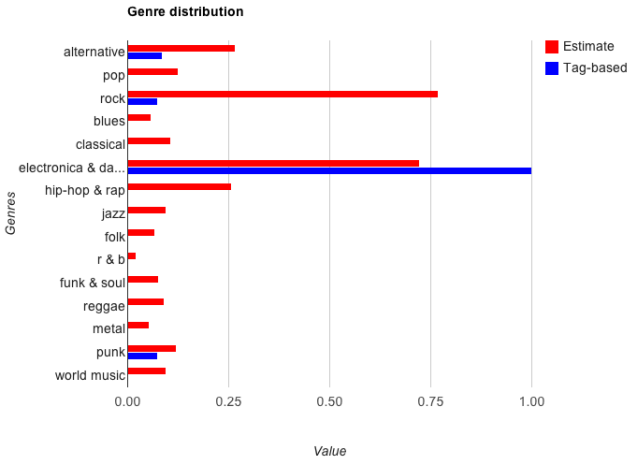

Figure 7.7: The estimated genre-profile for the artist The Prodigy (red columns). The H^t profile based on tags is shown in blue.

Measure	Tag Weight	**Value**
1st-Tier	0	59.58%
2nd-Tier	0	**80.75**%
1st-Tier	0.3	61.04%
2nd-Tier	0.3	80.07%
1st-Tier	0.5	62.08%
2nd-Tier	0.5	79.20%
1st-Tier	0.8	**63.25**%
2nd-Tier	0.8	77.33%
1st-Tier	1	60.45%
2nd-Tier	1	77.29%

Table 7.2: 1st and 2nd Tier.

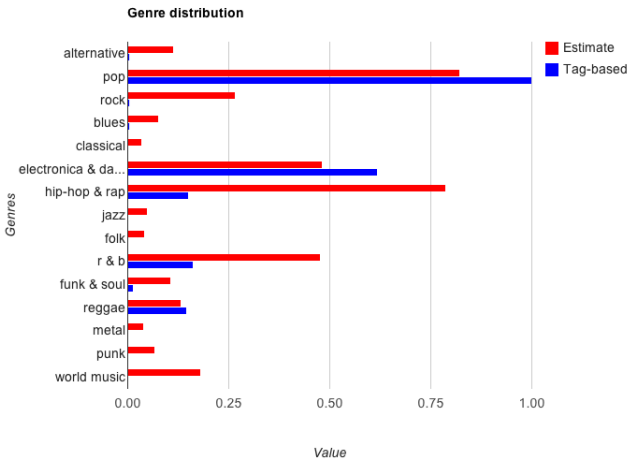

Figure 7.8: The estimated genre-profile for the artist Rihanna (red columns). The H^t profile based on tags is shown in blue.

Bibliography

[1] C. McKay, McEnnis, Fujinaga,"A Large Publicly Accessible Prototype Audio Database for Music Research", Proceedings of the International Conference on Music Information Retrieval (ISMIR). 2010.

Chapter 8

Data Enrichment: Artist Style Similarity based on Time, Geographical Location and Genre

We have previously stated that the lack of information is critical when it comes to placing its music entities in a semantic context. As a consequence, style similarity between artists cannot be efficiently established and therefore music recommendation is rendered impossible. This chapter investigates the novel hypothesis that stylistic similarity can be computed just by the employment of three distinct, music-artist related features: era, geographical location and genre.

8.1 Introduction

Artist similarity, as a typical high-level concept, lacks a precise definition. However, assigning a numerical similarity value for each pair of artists is an extremely useful procedure for music recommendation. Therefore, a lot of MIR research has focused on that particular task, but without taking into consideration the following [1]:

1. **Individual variation**: People's similarity assessment vary to the same extent as their music tastes. Given this fact, it is almost impossible to generate a similarity ground truth that agrees with the general opinion.

2. **Multiple dimensions**: The features that taken into consideration for artist similarity are unclear among people. In some cases that would be the genre and in others the vocal-style or the instrumentation, lyrics etc.

3. **Asymmetry**: Similarity between artists is not symmetric. For instance, people consider the band Blind Guardian to be similar to Queen, but not the other way around. is This raises the issue of using a Euclidean model for similarity, which is the typical case in MIR studies.

4. **Variability and span**: Artists tend to explore different styles and genres throughout their career. Therefore, their similarity to others may vary over time.

All these factors convert the task of assessing artist-similarity into a sophisticated problem. Developing a similarity framework is as difficult as generating a reliable ground-truth; and without a proper ground-truth how can the framework be enhanced? These considerations, although valid, have not stopped the MIR community from working on the subject.

Artist similarity and recommendation have been well researched inside the audio domain. Extracting and comparing audio-derived features has been the typical methodology for many years now. Such an approach relies on the prerequisite of the actual audio track which can not be always satisfied.

The recent field of Web-MIR, has partially solved the problem by utilizing metadata information only, in conjunction to the power of the Web. The fundamental concept behind such an approach is based on the assumption that similar artists will co-appear frequently inside the web-harvested documents. This assumption, although valid at first sight, is subject to a major flaw: old and obscure artists are not clustered based on some particular features when documented on the Web. Or to be more precise, the features used, do not follow any rules. For example, there exist plenty of web-pages containing information about old Dutch artists, where genre variations are not taken into consideration. In other web-pages, containing information about jazz music from the beginning of the previous century until today, the origin of the artists is neglected. In both cases the co-occurrence of artists inside the web-documents does not constitute proof of any similarity.

As a consequence, similarity for old and obscure artists cannot be calculated using any of the aforementioned methods (audio MIR, Web MIR). We aim at surpassing this problem by firstly placing artists in a meaningful context and then using that context to quantify similarities.

8.1.1 Hypothesis

Based on a set of preliminary experiments and our own intuition (although musicological support is also possible), we have identified three features that may define an artist's **style**: genre, location and period of activity or era. Our assumption was initially based on the fact that music distribution prior to 1950's was so limited, that each artist's style was amalgamated by his local and current cultural context. It is important to point out that stylistic similarity differs from genre similarity. Artists that belong to the same genre, don't necessarily share the same style.

We therefore assume that stylistic similarity between pre-1950's artists can be calculated just by using three high level features: genre, era and geographical location. In addition, we hypothesize that such a method would yield more meaningful recommendations than EchoNest for that particular era.

8.1.2 Prerequisites

Our hypothesis assumes that information about the artist's origin, era and location is present. However, since we are dealing with pre-1950's artists, such

information is difficult to acquire but not impossible. Chapters 4, 5 and 6 support, via experiments and evaluation, that good estimates for all three features can be computed via Web mining techniques.

8.2 Computing Similarity

In order to test our hypothesis we first need place artists in a 3-dimensional space. This will convert the high level problem of artists-similarity to the simple geometric problem of computing distances between points in space. Therefore for each artist T in the dataset we compute its distance to the query Q as follows:

$$d(T,Q) = \frac{w_1 * d_y(T_y, Q_y) + w_2 * d_g(T_g, Q_g) + w_3 * d_l(T_l, Q_l)}{3} \tag{8.1}$$

where $d_y(T_y, Q_y), d_g(T_g, Q_g), d_l(T_l, Q_l)$ the distances in terms of year, genre and location.$\mathbf{W}$ correspond to the weight vector that encodes the relative strength of the features. All of the distances are normalized so that the maximum distance is 1 and the lower 0.

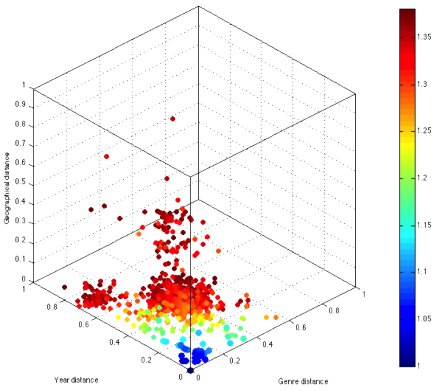

Figure 8.1: An example of artists placed in a 3D space. The coordinates of each artist correspond to its distance in terms of era, genre and location, from the query. The color represents the inverse-similarity or dissimilarity to the query.

Figure 8.1 presents an example of artists placed in the 3D space relative to the query (at the origin). Each artist has four coefficients:

$$\{d_y(T_y, Q_y), d_g(T_g, Q_g), d_l(T_l, Q_l), d(T,Q)\} \tag{8.2}$$

The first three define its location in space while the fourth its artist-distance compared to the query. Artists with small distance to the query have a blueish colour.

8.3 Experiment

The question we are trying to answer is twofold: a) is similarity based on our method meaningful and b) how does it compare to the EchoNest similarity? "Meaningfulness" cannot be computed by evaluation measures but only via means of community opinion. Therefore we created an online audio experiment where subjects are subjected to an audio file for each artist, accompanied by audio files corresponding to its similar artists as computed by our method and Echonest. The subjects are asked to rate the similarity of the audio tracks they are listening (as opposed to the query) in terms of **style**.

8.3.1 Test-set

For our experiments we use the Million Song Dataset, which contains latitude, longitude data and genre tags for artists, while also year of release information for songs. The number of artists whose first song was released before 1950 and the geographical information is present, are 98. In addition to those, we included a set of 46 artists from the period 1950-1960.

Each artist A is represented by a three features: A_y the year of the first documented release, A_g a genre profile as extracted from the tags and A_l a set of two coefficients corresponding to latitude and longitude. The year of first release gives us an indication of an artist's era, as tags an indication of an artist's genre profile. Therefore the artists features are less than precise.

Each artist is also accompanied by a size-10 list of similar artists as compiled by EchoNest itself. The mechanics behind this function are unknown, however, as we mentioned before, the similarity list is employed for comparison.

8.3.2 Online Audio Experiment

Given the two similarity lists for each artist (EchoNest and ours), we want to investigate at which extent both of them agree to human intuition. The online experiment presents a simple interface to the user (Figure 8.2), while asking him to first listen to the query. The user is also presented with four other audio tracks. Two of them are randomly picked from our similarity list and two from EchoNest's (via the 7digital[1] service). The audio tracks correspond to the earliest release of the artist in hand. The order of which they are listed is also random.

The user is asked to rate the similarity between the artists in the tracks in a non-continuous scale of 1 to 5 corresponding to "Not at all", "Rather dissimilar", "Just a bit", "Similar" and "Very similar". The experiment can be performed infinite amount of times and each time the user is presented with a random set of audio tracks.

8.4 Results

After a period of almost one month, during which experiment was online, we gathered around 200 answers mostly from experts (musicologists and music researchers). Given that the similarity rating scale expands from 1 to 5, it is fairly

[1] www.7digital.com

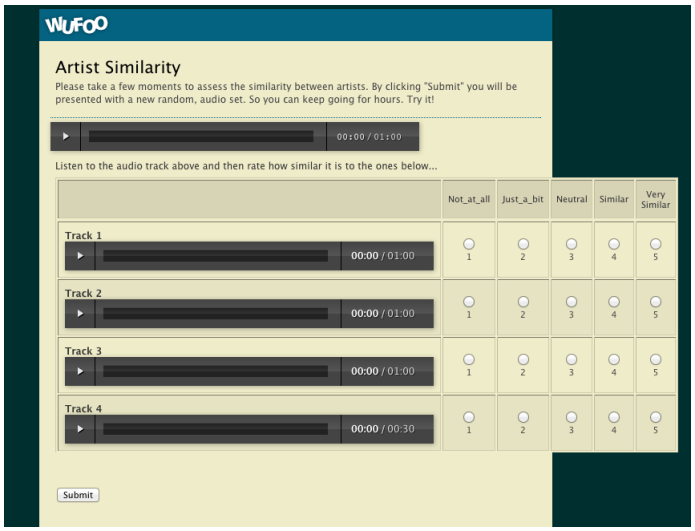

Figure 8.2: The interface for the online audio experiment.

easy to simply compute the mean for both our method and EchoNest's. However such an approach is not informative, considering that the time-context and relative comparison between methods are crucial to our hypothesis. Therefore for each year $y \in \{1930, 1931, ..., 1960\}$ we compute the ratio R such that:

$$R(y) = \frac{mean(M_y(i))}{mean(Ec_y(i))} \tag{8.3}$$

where $M_y(i)$ the mean of the ratings for our method in the period 1930-y, and $Ec_y(i)$ the mean of the ratings for EchoNest in the same period. Therefore, $R(y)$ takes into consideration the accumulative mean of the previous years in addition the current year y. Figure 8.3 presents the plot of R.

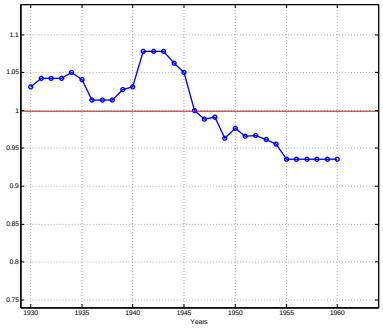

Figure 8.3: The ratio R through time.

Our hypothesis assumes that R should be greater than 1 for years prior to 1950. Therefore, it is clearly obvious the plot supports our assumption. Despite the fluctuations along the curve, it is still clear that as we approach 1945-1950 our similarity method becomes meaningless. We carefully chose the term

"meaningless" instead of "futile", since the R is not drastically dropped; hence our method still produces acceptable similarities, considering that EchoNest constitutes our baseline of "acceptable" recommendation.

The ratio R of the means gives a relative perspective of both methods, but fails to capture how well they agree with human intuition. Do people think that the audio derived from both methods is similar to the original queries and it which extent? This is presented in Figure 8.4 where both $mean(M_y(i)), mean(Ec_y(i))$ are plotted. Ideally the mean of any method should be close to 5 (rating "Very similar"), however, in both the compared methods, the mean is centered around 3 (rating "Just a bit"). For years prior to 1950 both methods are over the 3-threshold , and under it from then on.

What does this say to us? Simply, both methods yield acceptable recommendations for pre-1950's artists, but slowly degrade as we reach the 1970's. We are not sure how the EchoNest similarity will behave for later years (e.g. 1980's, 1990's); for our algorithm though we can safely assume a continuing degradation.

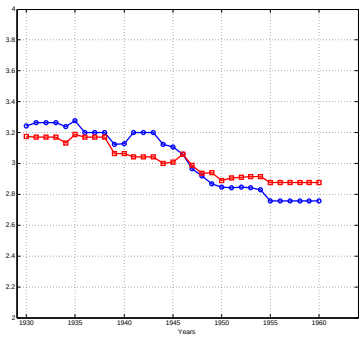

Figure 8.4: The ratios R_{high} (blue) and R_{low} (red) through time.

Figure 8.5 presents the plot of R_{high} and R_{low} corresponding to the R vector for the ratings $\{4, 5\}$ and $\{1, 2\}$ respectively. In other words, instead of just taking the mean of the ratings, we spilt it those corresponding to high ("Very similar", "similar") and low ("Not at all", "Rather dissimilar").

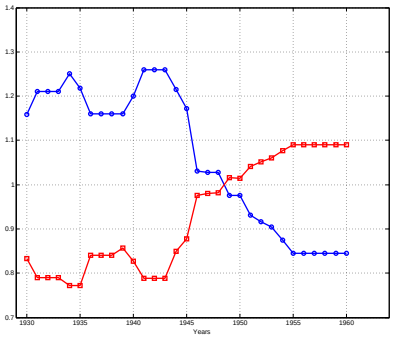

Figure 8.5: The ratios R_{high} (blue) and R_{low} (red) through time.

8.4.1 Issues

So far the results have shown a slight superiority of our method over EchoNest's for pre-1950's artists, and thus proving both our hypotheses. However, certain issues arise when taking into consideration the small size of the user responses. Significance tests (T-test) have shown that the superiority of our method, based on the data, cannot be verified. In other words, both methods seem to show a normal-distribution behaviour with the same means.

On the other hand, it is important to consider the error introduced to our method based on:

1. The approximation of the productivity distribution of an artists, by using his earliest, documented release. The MSD dataset offers release date information for certain artists and for certain songs. Therefore some artists, who might be very similar to the query, may have been neglected due to their lack of release dates. In some cases, the earliest release date might have not corresponded to the actual release date. This would result to an erroneous large distance from the query.

2. The approximation of an artist's genre based on the user tags. Since we are dealing with old artists it is very likely for user tags to be absent or very limited. This results to a meaningless genre profile and consequently to an erroneous artists similarity. In addition, the distribution of genres in the test-set is non uniformal (Figure 8.6). Most of the artists are considered "blues", hence the the genre distance function is rendered meaningless.

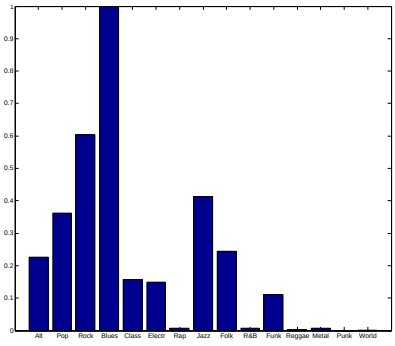

Figure 8.6: The overall distribution of genres in the test-set.

3. The limited amount of geographical information for artists. Most of the MSD data, for which geographical location is present, correspond to American artists (Figure 8.7). Therefore, the effect of geo-graphical distance on the artists similarity function is limited.

4. Finally, audio snippets for every artist cannot be provided. It is possible that EchoNest can not provide a 7digital audio link for the artist's earliest release. Therefore some artists were neglected.

These issues affect the validity of the results presented in the previous section. However, the extend of this effect is rather unknown. Would a richer,

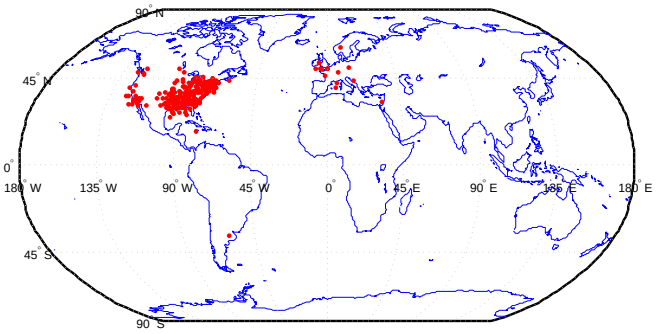

Figure 8.7: The overall distribution of genres in the test-set.

more complete test-set support better our hypothesis? We believe that it probably would, but compiling such a set would require a huge amount of effort. In the context of this experiment, these issues act as pros and cons at the same time, since even with approximate values for genre and eras, our method competes face to face with EchoNest's. However, given that the country of origin is mostly USA, its employment in the distance function is debatable. To sum up, the aforementioned issues are too crucial to ignore and should be investigated further in future implementations.

8.5 Conclusions

This chapter investigated the novel hypothesis that artist similarity for pre-1950's can be computed by means of three distinct features: time, geographical location and genre. The validity of the hypothesis was evaluated using an online experiment where participants were asked to rate similarities between audio tracks. The results support our hypothesis but lack significance. In addition, many issues related to the precision of the test-set arose, while further reducing our confidence on the results. We strongly believe that a more concrete test-set would reveal our method's strength, but this remains to be investigated in future works.

Bibliography

[1] D. Ellis, B. Whitman, S. Lawrence, A. Berenzweig, "The Quest for Ground Truth in Musical Artist Similarity", Proceedings of the International Conference on Music Information Retrieval (ISMIR). 2002.

[2] J. S. Downie, K. West, A. F. Ehmann, E. Vincent, "The 2005 Music Information retrieval Evaluation Exchange (MIREX 2005): Preliminary Overview", Proceedings of the International Conference on Music Information Retrieval (ISMIR). 2005.

Chapter 9

Conclusions

The aim of this thesis has been to investigate two topics: interoperability and data enrichment inside a music context. Although interoperability lacks a precise definition, placing it inside the music domain helped us understand its underlying concepts and possible issues that may arise when trying to achieve it. Furthermore, we examined a specific case of record-level interoperability, namely linking Sound and Vision's music collection to MusicBrainz. Our proposed approach comprising of artist name permutations, metaphones and naming abbreviations (e.g. M. Jackson) shows great potential while room for further improvement.

Current data enrichment for music databases using Web-mining techniques, although thoroughly studied and researched, has been proven futile when dealing with old and obscure artists. As a consequence, we proposed three Web-based techniques tailored to that particular case. We showed that that high-level features such as time, geographical location and genre can be computed using a combination of Web search engines and music hubs such as MusicBrainz, Last.fm and EchoNest.

As a proof of concept we assessed the viability of computing artist similarity based solely on the aforementioned features. Our experiments revealed that such a scheme is more than adequate and can be employed efficiently for artist recommendation. Therefore, we proved that high quality data enrichment is possible simply by using artist names and song titles.

9.1 Putting the Pieces Together

Any of the aforementioned methods would be rendered useless if it wasn't for their applicability. Therefore, as a final demonstration of our methods' usability, we will present a prototype interoperable infrastructure, implemented for the COGITCH project. The general framework employs both the notions of interoperability and data enrichment and more specifically:

1. Interoperability: Federated search, Record linkage.
2. Data Enrichment: Placing music entities in time, Artist country of origin determination, Artist-genre estimation, Artist similarity.

However, before we continue to the details, it is worth describing the remote COGITCH collections.

9.1.1 Collections Description

The S&V song collection comprises of a just one table of around 355000 records. Each record is represented by the following fields: song title, artist, composer, original title, track length and other fields of internal usage. The most important elements in our case are "Song title" and "Artist" , with the latter being the most noisy. It has been previously discussed that it contains tags (e.g. [zanger]. [guitar]), secondary artists (e.g. instrumentalists) with the addition of typos, name variations etc.

The Meertens folksong collection comprises of a set of tables corresponding to "Artists", "Songs", "Sources", "Genres", "Tune families" and many more. All of them create a complex scheme of relations that render any attempt of reorganizing it impossible. However as compared to S&V most of the fields contain noise-free information.

Achieving interoperability is the main goal for COGITCH and as a consequence the most important fields in the Meertens database are those that overlap with S&V, namely "Songs" and "Artists". Therefore, similar to Arampatzis, we performed an unconditional merging of the important tables and flattening of all metadata structure (Figure 9.1). This procedure allowed us to retrieve all the important pieces of information without the employment of complex and time-consuming JOIN queries.

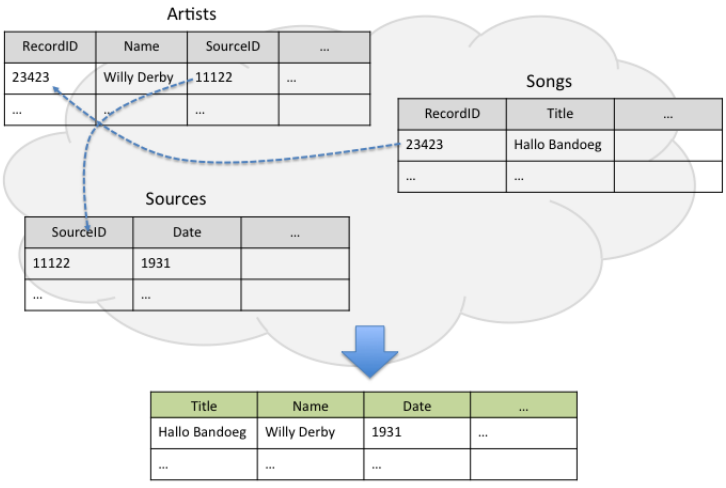

Figure 9.1: A simple representation of the flattening and merging of Meertens' tables. All the important information is stored into a new table.

9.1.2 Interoperability: Federated Search, Record Linkage

Out method for achieving interoperability relies on the notion of federated search as it was described in 2.3.3. Therefore our framework firstly transforms the user query and propagates to the databases. The results of both collections are then merged and sorted based on a string matching function (see Figure 9.2).

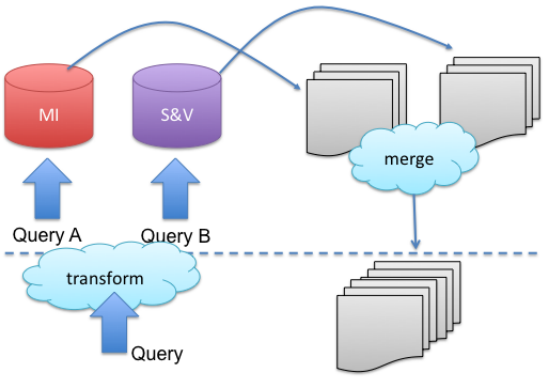

Figure 9.2: A schematic representation of the federated search for the COG-ITCH project.

The transformation of the original query assumes some kind of interoperability at the schema-level. In other words, a semantic mapping between the corresponding fields of both collections should exist. The "Artist" and the "Song Title" fields of both sources are therefore semantically linked.

A screen-shot of the frameworks response to the query "hallo" is shown in Figure 9.3. It can be easily observed that only the overlapping fields are always present (unless their value is NULL). For example, in cases where the "Source" is "S&V", the "Year" field is empty since such information was not provided originally.

Regarding record linkage, our framework incorporates it via links to MusicBrainz and unique artist name identifiers. This is shown in Figure 9.4, where the fields "ArtisID" and "MusicBr" are populated accordingly. The linkage is an off-line process and therefore it has not been applied to all the records and artists of S&V. In our example the "Artist" field values "Hofmann duo zang" and "Willy Derby zang" are linked to the unique artists "Hofmann Duo" and "Willy Derby" and them to MusicBrainz IDs. We shall later see in which way we utilize the latter link.

9.1.3 Data Enrichment: Placing music entities in time, Artist country of origin determination, Artist-genre estimation, Artist similarity.

Data enrichment is an off-line process similar to record linkage. A computed year of release of a song and the country of origin of an artist are placed in the "Year" and "Country" fields respectively (see Figure 9.4). This kind of information lets us generate visualizations as shown on Figure 9.5. The interface presents the

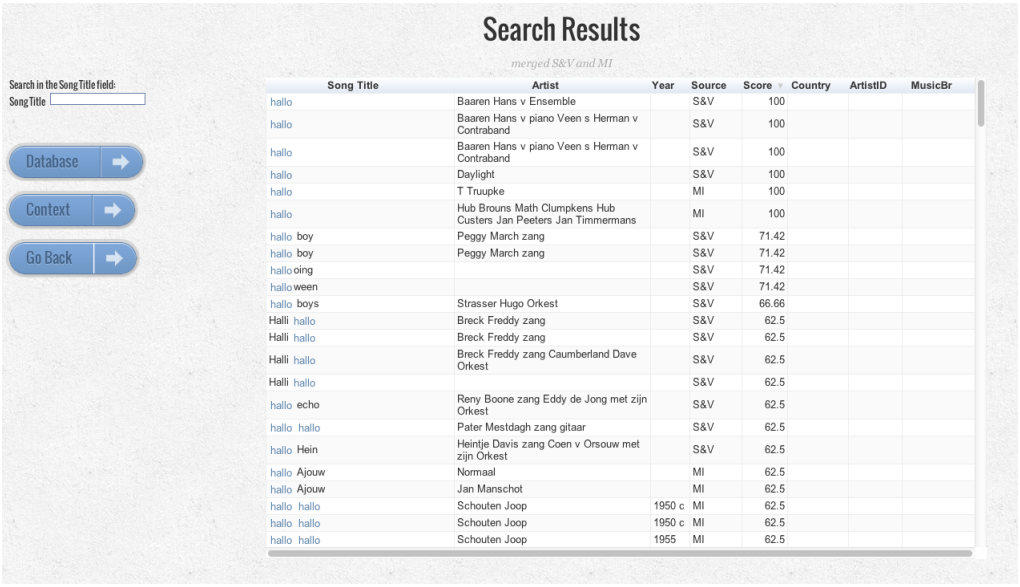

Figure 9.3: An example of implemented interface presenting the results of the query "hallo".

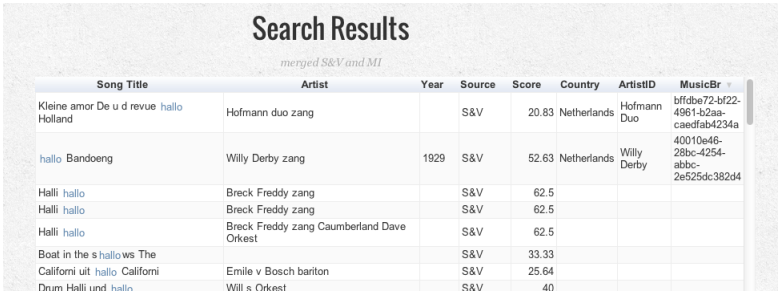

Figure 9.4: An example of implemented interface presenting the results or the query "hallo", focusing on two songs by "Hofmann Duo" and "Willy Derby"

distribution of countries on a world map, in addition to the distribution of years. This gives the user a global perspective of the returned results, and lets him place the information in a strong semantic context. Aside from the estimated data, the visualization presents the distribution of the S&V records in terms of media types (LP, ST, 45 rpm etc.), and also the distribution between sources (S&V and MI).

Clicking on any of the records in the main table presents the user with a new screen, denoted "Overlap". The main idea behind "Overlap" is to show to the user which records from the MI database match the song title he was interested in. For example, when clicking on the record with fields "Hallo Bandoeg" by "Willy Derby zang", the interface will return a list of MI song titles with the highest similarity to "Hallo Bandoeg".

But aside from that, the interface also provides the audio track for that particular song (when available) in addition to the artist's productivity profile,

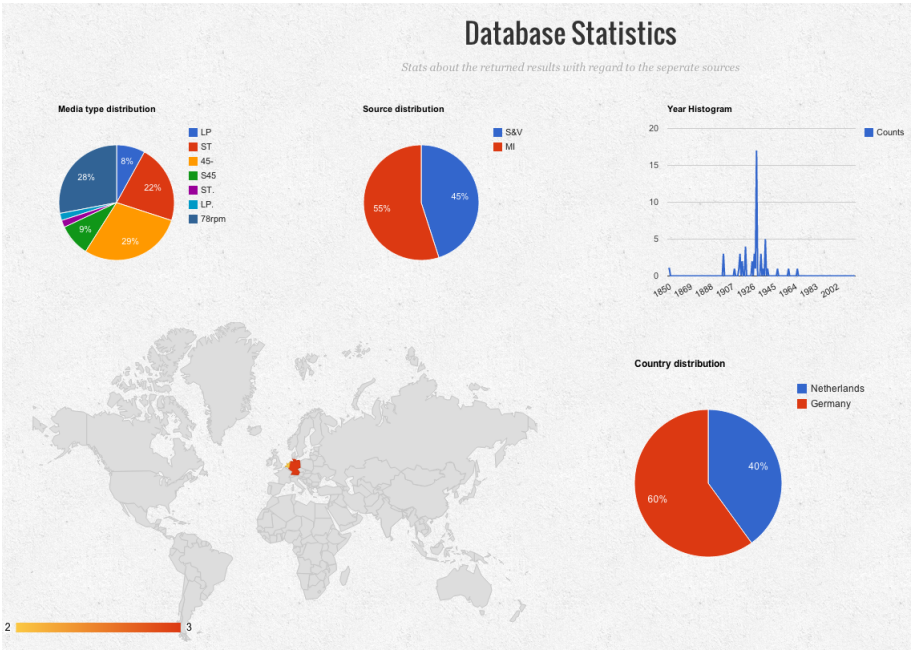

Figure 9.5: A set of data visualization as offered by the framework's interface (query "hallo"). Country of origin has been computed for only three artists (corresponding to five songs). Three out of five songs are from Germany and the rest from Netherlands. Most of the songs were recorded around the 1930's.

and YouTube matches (Figure 9.6). This function lets the user place the artist and song in context. This interface is a work in progress, hence genre profiles have yet to be incorporated.

The linkage of the artists to MusicBrainz is exploited by offering to the user the corresponding MusicBrainz profile page (see Figure 9.7). Therefore, most of the information that is available on the Web regarding the song and artists are provided by our framework.

Finally, since genres profiles have not been incorporated yet, artist similarity cannot be calculated. However, our vision for visualizing that information is presented in Figure 9.8. Artists are represented as spheres in a 3-dimensional space. The axes correspond to the time, genre and geo-location distance from a user-selected or simply the top-ranked artist. The size of each sphere denotes its similarity to the artist of reference. In such an representation the user would be able to interact with the environment (rotate, scale), while also interact with the spheres themselves (play audio, show additional information etc.).

Figure 9.6: Additional information provided for the query "Hallo Bandoeg" by "Willy Derby".

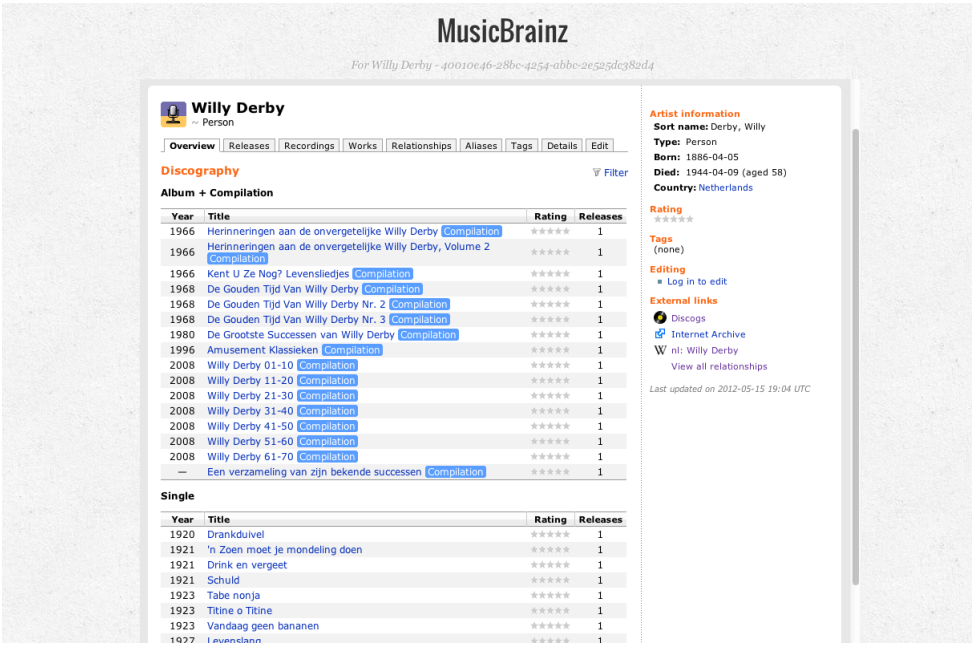

Figure 9.7: The MusicBrainz profile for "Willy Derby" as provided by our interface.

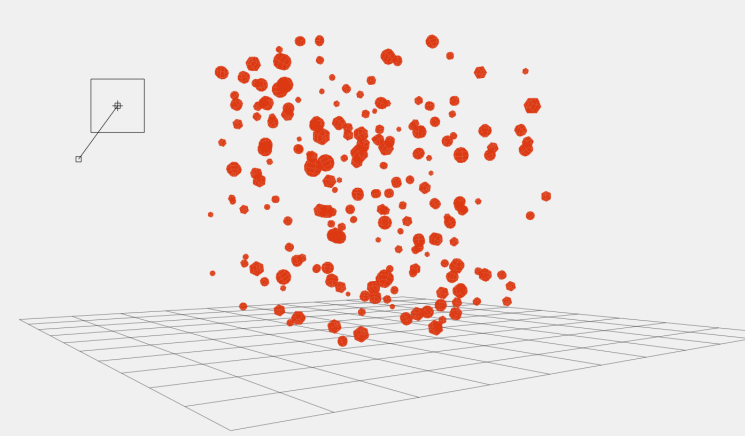

Figure 9.8: Artists represented as points in a 3d-space.