Extracting Relations from Unstructured Text Sources for Music Recommendation

Mohamed Sordo[1], Sergio Oramas[2] and Luis Espinosa-Anke[2]

[1] Center for Computational Science, University of Miami, Coral Gables, FL, USA
[2] Department of Information and Communication Technologies, Universitat Pompeu Fabra, Barcelona, Spain
msordo@miami.edu, {sergio.oramas, luis.espinosa}@upf.edu

Abstract. This paper presents a method for the generation of structured data sources for music recommendation using information extracted from unstructured text sources. The proposed method identifies entities in text that are relevant to the music domain, and then extracts semantically meaningful relations between them. The extracted entities and relations are represented as a graph, from which the recommendations are computed. A major advantage of this approach is that the recommendations can be conveyed to the user using natural language, thus providing an enhanced user experience. We test our method on texts from *songfacts.com*, a website that provides facts and stories about songs. The extracted relations are evaluated intrinsically by assessing their linguistic quality, as well as extrinsically by assessing the extent to which they map an existing music knowledge base. Finally, an experiment with real users is performed to assess the suitability of the extracted knowledge for music recommendation. Our method is able to extract relations between pair of musical entities with high precision, and the explanation of those relations to the user improves user satisfaction considerably.

1 Introduction

Music consumption has changed dramatically in the last few years. The rise of digital audio and streaming services means users are now one click away from accessing millions of songs by more than a million artists [8]. Yet this vast availability has posed a serious problem: how can a user explore or discover preferred music from all the available content? Traditionally, users have relied on their friends, their favorite music radio host, a music expert in their local retail store, etc. to obtain recommendations on artists or albums they might like. Although this traditional approach is still valid and used by many people, its ability to cover the vast amount of available music nowadays is seriously hindered. Automatic approaches to music recommendation have become necessary. According to [8], there is no clear formula for providing *good* recommendations to a user. There are however some key elements that should be taken into account: *novelty*, *familiarity* and *relevance*. The authors also emphasize the importance of providing an *explanation* to the user, as to why a music item has been recommended. The latter can provide an enhanced user experience, helping the user gain confidence in the recommendation system.

In this paper we propose a method that exploits unstructured text sources from the web to provide music recommendations. It does so by identifying music-related entities in the text (such as *song*, *band*, *person*, *album* and *music genre*) and extracting relations between these entities. The resulting knowledge graph can be used to not only provide recommendations but also to give an explanation of the recommendations to the user by using natural language.

2 Related work

2.1 Music Recommendation

Music Recommendation is a relatively young but continuously growing research topic, in both MIR and RecSys communities [7]. There are many methods for recommending music. In this paper we only concentrate on Context-based filtering methods. Context-based filtering methods use information extracted from text sources to obtain similarity between artists or songs. Approaches based on this technique typically compute some sort of term weighting, like TF-IDF [30], or exploit co-occurrences between musical entities [26,16].

Most research in Music Recommendation has been dedicated to developing algorithms that provide *good* and *useful* recommendations [7], yet very few approaches (at least to our knowledge) provide explanations of the recommendation to the users [25,24]. According to [8], giving explanations of the recommendations provides transparency to the recommendation process and increases the confidence of the user in the system. In [24] Passant proposes a Music Recommendation system that uses the dataset of structured information DBpedia as a backbone for finding similar artists. Explanations are given to the user based on the shared sub-classes of the DBpedia ontology between two artists (e.g.: voice type, instrument, death place, etc.). In this paper we propose a method to extract explanations from unstructured text sources.

2.2 Relation Extraction

Relation Extraction (RE) approaches are often classified according to the level of supervision involved. Supervised learning is a core-component of a vast number of RE systems, as they offer high precision and recall. However, the need of hand labeled training sets makes these methods not scalable to the thousands of relations found on the Web [18]. More promising approaches, called semi-supervised approaches, bootstrapping approaches, or distant supervision approaches do not need big hand labeled corpus, and often rely on existent knowledge base to heuristically label a text corpus (e.g., [6,18]) Open Information Extraction methods do not require a pre-specified vocabulary, as they aim to discover all possible relations in the text [2]. However, these methods have to deal with uninformative and incoherent extractions. In ReVerb [13] part-of-speech based regular expressions are introduced to reduce the number of these incoherent extractions. Less restrictive pattern templates based on dependency

paths are learned in OLLIE [21] to increase the number of possible extracted relations. Unsupervised approaches do not need any annotated corpus. In [12] verb relations involving a subject and an object are extracted, using simplified dependency trees in sentences with at least two named entities. These approaches can process very large amounts of data, however, the resulting relations are hard to map to ontologies [19].

In this paper we use a technique called Dependency Parsing to extract relations from text. Dependency Parsing provides a tree-like syntactic structure of a sentence based on the linguistic theory of Dependency Grammar [29]. One of the outstanding features of Dependency Grammar is that it represents binary relations between words [1], where there is a unique edge joining a node and its parent node (see Fig. 2 for the full parsing of an example sentence). Dependency relations have been successfully incorporated to RE systems. For example, [5] describe and evaluate a RE system based on shortest paths among named entities. [10] focus on the smallest dependency subtree in the sentence that captures the entities involved in a relation, and [14] propose a rule-based dependency-parsing Open IE system.

3 Methodology

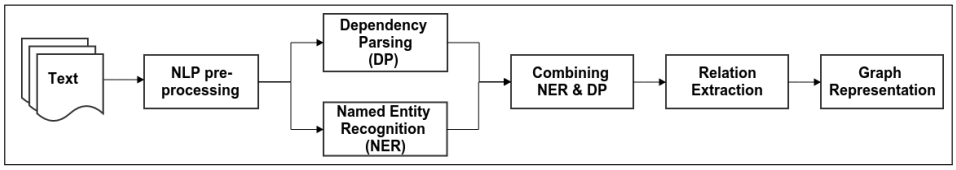

Fig. 1. Workflow of the proposed method.

3.1 NLP Pre-processing

Fig. 1 depicts the work-flow of the proposed method. Given a text input (e.g., a collection of web documents) the pre-processing module segments it into single sentences. Each sentence is subsequently divided into a sequence of words or tokens. In this paper we use the Stanford NLP tokenizer [3].

3.2 Named Entity Recognition (NER)

Although NER is not a solved problem [20], there are many available tools with good enough performance ratios [15]. Among those tools, we tried AIDA [31] and DBpedia Spotlight [22]. Although AIDA has a higher recall [15], in terms of CPU and memory consumption DBpedia Spotlight provided a better performance. DBpedia Spotlight is a system for automatically annotating text

[3] http://nlp.stanford.edu/software/tokenizer.shtml

documents with DBpedia URIs, finding and disambiguating natural language mentions of DBpedia resources. DBpedia Spotlight is shared as open source and deployed as a Web service freely available for public use[4]. It has a competitive performance and evaluations show an F-measure around 0.5 [22].

Our NER module receives a list of sentences as input and uses DBpedia Spotlight to find DBpedia entities in the sentences. The entities are then annotated with their corresponding URI and type. In the current approach we only consider 5 different types that are relevant to the music domain: *song*, *band*, *person*, *album* and *music genre*. The rest of the recognized entities are ignored.

3.3 Dependency Parsing (DP)

Our DP module uses the implementation by [3] and produces a tree for each sentence. Each node in the tree represents a single word of the sentence, together with additional linguistic information like Part-of-Speech[5] and syntactic function. For instance, in Fig. 2 the word *Freedom* is the subject (SBJ) of the root word *was*. The definition of all these syntactic functions is given in [28]. In our case, however, we want to find relations between music-related entities, which can consist of more than one word. The next module takes care of this.

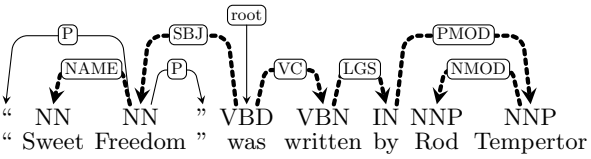

Fig. 2. Example sentence with dependency parsing tree.

3.4 Combining NER & DP

The aim of this module is to combine the output of the two previous modules. For each recognized music-related entity in the NER module (Section 3.2), the combination module merges all the nodes in the dependency tree of the sentence that correspond to that entity into a single node. Fig. 3 shows how the example sentence from Fig. 2 is modified by merging the nodes that correspond to the recognized entities (in this case, the album *“Sweet Freedom”* and the person *Rod Temperton*) into single nodes.

3.5 Relation Extraction (RE)

The Relation Extraction module analyzes the modified dependency trees from the combination module (Section 3.4), and extracts relations between pairs of

[4] https://github.com/dbpedia-spotlight/dbpedia-spotlight/wiki/Web-service
[5] http://ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

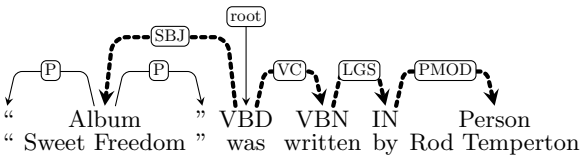

Fig. 3. Example sentence with its modified dependency tree, after merging nodes that correspond to an entity.

recognized music-related entities. Two entities (nodes) in a tree are considered to be related if there is a path between them that does not contain any other entity. Since dependency trees are directed trees, there is no guarantee in finding a path. Therefore we use an undirected version of the tree to obtain the path between the aforementioned pair of entities. Interestingly, the nodes that are part of the path between the two entities are considered by our method to represent the actual relation between the entities. In the example of Fig. 3, the resulting path between *“Sweet Freedom”* and *Rod Temperton* contains the words *was*, *written*, *by*. These words are used to define the relation between *“Sweet Freedom”* and *Rod Temperton*.

By analyzing the output of a subset of the Songfacts dataset (explained in Section 4.1), we observed that not all relations between pairs of music-related entities made sense linguistically. Thus, we empirically introduced a set of rules to filter out irrelevant relations. A rule in our case is defined as a sequence (or a regular expression) of word types that can appear between a pair of music-related entities. The word types are represented by their part of speech tags. For instance, the previous example (Fig. 3) has a relation between an entity of type *Album* (*“Sweet Freedom”*) and an entity of type *Person* (*Rod Temperton*). The relation *Album-Person* in this case contains the terms *VBD*, *VBN* and *IN*, which are part of speech labels meaning verb past tense, verb past participle and preposition, respectively. The complete list of rules is shown in Fig. 4.

3.6 Graph Representation

After the list of relations between entities is filtered, the method creates a graph representation of it, where the nodes are the music-related entities and the edges represent the relations (i.e., the path) between pairs of entities. The graph contains five chosen types of nodes corresponding to the 5 music-related types: *song*, *band*, *person*, *album* and *music genre*.

4 Evaluation

We tested our method against a dataset gathered from *songfacts.com* (Section 4.1). The output of the method has been evaluated from two different standpoints, namely: (1) a linguistically motivated evaluation of the extracted relations and (2) a data-driven evaluation of the extracted knowledge. The linguistic evaluation quantifies the correctness of a relation by comparing it to a reference annotation manually crafted by a Computational Linguistics expert. Data-driven

```
('Album', 'Band'): ['(VBD)(NN)', '(VBZ)(NN)', '(VBD)(NN)(IN)(NN?)', '(NN?)(IN)(NN?)', '(VBD)(VBN)(IN)'],
('Album', 'Person'): ['(NN)(NN?)', '(VBD?)(NN)', '(NN?)(IN)(NN?)', '(VBD)(VBN?)(IN)(NN?)'],
('Album', 'Song'): ['(NN?)(VBD)', '(VBD)(NN)(IN)'],
('Band', 'Album'): ['(VBD?)(NN)', '(VBD)(IN)', '(NN)(VBD)', '(NN)(IN)(NN)', '(NN?)(VBD)(IN)(NN)'],
('Band', 'MusicGenre'): ['(VBZ)(NN)', '(VBD)(NN)', '(VBP)(NN)'],
('Band', 'Person'): ['(NNS)', '(NN)(NN?)', '(NN)(VBD)', '(NN)(CC)(NN)', '(VBD)(IN)(NN?)', '(VBD)(NN)'],
('Band', 'Song'): ['(NN?)(VBD)', '(VBD?)(NN)(IN)', '(NN)(VBD)(NN)(IN)'],
('MusicGenre', 'Band'): ['(NN)(NN?)'],
('MusicGenre', 'Person'): ['(NN)(NN?)', '(NN)(CC)(NN)'],
('Person', 'Album'): ['(NN)(NN?)', '(NN)(IN)(NN?)', '(NN?)(VBD)(IN?)(NN?)'],
('Person', 'Band'): ['(NN?)(IN)', '(VBD)(IN?)(NN?)', '(VBD)(NN)(IN)', '(NN)(IN)(NN)'],
('Person', 'Person'): ['(NN)', '(NN)(CC)(NN)', '(VBD)(IN)(NN?)', '(VBD)(CC)(VBD)'],
('Person', 'Song'): ['(VBD)(IN?)', '(VBD)(IN?)', '(NN)(IN)', '(VBD)(IN)(VBG)', '(NN)(VBZ)(IN)', '(VBZ)(NN)(IN)',
                     '(NN)(VBD)(NN?)(IN?)', '(VBD)(IN?)(NN)(IN?)'],
('Song', 'Album'): ['(VBD)(VBN)', '(VBZ)(IN)(NN)', '(VBD)(VBN?)(IN)(NN?)'],
('Song', 'Band'): ['(VBZ)(NN)', '(IN)(NN)', '(VBD)(VBN?)(IN)(NN?)', '(VBZ)(VBN?)(IN)(NN?)'],
('Song', 'MusicGenre'): ['(VBZ)(NN?)', '(VBZ)(NN)(IN)(NN?)', '(VBD)(VBN)(IN)(NN?)'],
('Song', 'Person'): ['(VBZ)(NN)', '(IN)(NN?)', '(VBD?)(NN)', '(VBZ)(IN)(NN?)', '(VBD)(VBN?)(IN)(NN?)', '(VBZ)(NNS)(IN)',
                     '(VBZ)(NN)(IN)(NN?)', '(NN)(VBD)(VBN)(IN)'],
```

Fig. 4. Part-of-speech rules that represent the relation between music entity types

evaluation compares the extracted knowledge with a reference knowledge-base. A final experiment involving real users was also performed to assess the suitability of the extracted knowledge for music recommendations, and to test the effect produced by textual explanations in the recommendations given to the user. The following subsections provide a detailed description of the dataset and the experimental results.

4.1 Dataset

Songfacts[6] is an online database that collects, stores and provides facts and stories about songs. These stories are collected in a crowd-sourcing way by registered users and they are reviewed by the website staff. The web site contains information about more than 30.000 songs belonging to nearly 6.000 artists. Songfacts tidbits are little pieces of information telling stories about a song, such as what the song is about or who wrote it, who produced it, who collaborated with whom or who directed the video clip, etc. Therefore, a huge amount of information about the actors involved in the creative process of a song is present in the aforementioned tidbits.

We crawled the whole song dataset from Songfacts in mid-January 2014. Before applying our method, we did some adjustments in the NER module due to the specificity of the dataset. Identification of song titles in text is a challenge, since titles are often short and ambiguous [17]. Fortunately, due to the nature of the Songfacts website — which provides a separated web page for each song, with its corresponding metadata and facts — the complexity of the identification process of song titles is reduced considerably, under the assumption that ambiguity is less probable in this scenario. Thus, apart from using the DBpedia Spotlight NER, we searched and matched each song title in the facts. Moreover, further analysis of the facts showed that usually the facts refer to the song in question using expressions such as “the song” or “this song”. Therefore, we looked for these structures and treated them as detected song entities. We chose only the songs whose title had been recognized by our system as a song entity and only those among them who were involved in at least one relation with another recognized entity. Finally, we also used the artist metadata provided by Songfacts

[6] http://www.songfacts.com

to add a relation between entities of type artist and entities of type song with the label "by". After applying all the steps of our method, we obtained 12838 entities and 16341 relations between them. Among the detected entities, 6116 were songs, and those songs were related to 1483 different artists.

4.2 Linguistic Evaluation

We base our evaluation on previous work in Relation Extraction. For example, in [13,21], the general approach is to assess the automatically extracted relations in terms of correctneess according to human judgement. Additionally, [2] describes a finer-grained analysis, adding a prior step in which relations are judged as being *concrete* or *abstract*. Our evaluation sample amounts to 205 relations extracted from 155 randomly selected sentences. Two human judges marked a relation as "correct" if the information contained in the sentence implied or connoted that the relation was true. An "incorrect" label was assigned otherwise.

The results obtained were very high with regard to the observed agreement. Our results indicate that out of 205 relations, both evaluators agreed in judging 146 relations as correct and 23 as incorrect. This means that the overall observed agreement reaches 82.43%, while the agreement only on correct relations is 71.21%. We also computed the Cohen's Kappa [9] agreement measure in order to have an additional viewpoint of the reliability of this evaluation. Our computation of Cohen's Kappa was 40.68, which is a reasonably high value considering that the evaluation only consists of two classes ("correct" and "incorrect"), and this metric strongly punishes the chances of two evaluators to agree by chance.

We illustrate these results with an example that showcases a case of agreement (the first case) and a case of disagreement (the second) in the same sentence.

Sentence: [***Weezer***] frontman [***Rivers Cuomo***] wrote this song for and about Jamie Young , the band's first lawyer.
Entities: Band ↔ Person
Extracted Relation: Weezer (frontman) Rivers Cuomo

Sentence: Weezer frontman [***Rivers Cuomo***] wrote [***this song***] for and about Jamie Young, the band's first lawyer.
Entities: Person ↔ Song
Extracted Relation: Rivers Cuomo (frontman wrote) Weezer_-_Jamie

	P - P	P - S	B - P	B - A	P - A	B - S	S - A	M - B	M - P	S - M
Precision	59.09	95.52	88.88	94.87	81.13	**95.74**	95.55	85.18	62.5	96.55
Recall	48.14	60.95	49.23	31.62	45.74	55.55	47.25	**71.87**	66.66	57.14
F-Score	53.55	74.41	63.36	47.43	58.49	70.36	63.23	**77.96**	64.51	71.79

Table 1. Results per pairs for the token-wise evaluation. The analyzed entities are: P (person), S (song), B (band), A (album), M (Music Genre)

In the first example both evaluators agreed in assigning a "correct" label to the relation. In the second example one evaluator found it to be incorrect. We argue that this can be due to the distracting presence of "frontman", which can be considered to be a property of the first entity, rather than an element of the relation. While this dichotomy has been addressed in previous work ([13] evaluated a relation to be correct where critical information was dropped from the relation but included in the second argument), we propose a lexically motivated approach, which compares the relations extracted by the system with those that would be extracted by a human expert. The idea was to be able to compare the wording of a relation between system and human. Precision and Recall are computed by looking at word-overlap. For instance, in the above case, the relation for the pair Person↔Song would get a score of P=0.5 and R=1 because the human evaluator extracted the relation *Rivers Cuomo (wrote) Weezer_-_Jamie*. "Frontman" would be a false positive. Table 1 provides results for the full evaluation dataset and for each pair of entity types.

It is worth noting how our approach has performed very well in certain pairs, especially in the MusicGenre ↔ Band, Person ↔ Band, Band ↔ Song and Song ↔ MusicGenre pairs. This might be due to the many straightforward one-word relations among these entities, as shown in the following examples:

Sentence: The [***Christian Metal***] band [***Stryper***] recorded this song for their 1990 album Against the Law and made a video for it.
Entities: MusicGenre ↔ Band

Sentence: Jessie Lacey of Brand New's girlfriend cheated on him with [***John Nolan***] of [***Taking Back Sunday***].
Entities: Person ↔ Band

Lower scores were obtained in relations like Person ↔ Album or Band ↔ Album. A closer look at these relations shows that there are many cases where an album is preceded by a number of adjectives and other noun modifiers. These modifiers are often described as sibling nodes of the relation in the dependency tree, and thus do not appear in the path between the two related entities.

4.3 Data-driven evaluation

The output of our system can be regarded as a knowledge base of music related information. This knowledge base consists of entities and relations, two building blocks of a simple , non-taxonomic ontology. According to [11], a learned ontology can be evaluated in three different ways: in the context of an application, by domain experts or by comparing it with a predefined reference ontology (i.e., a Gold Standard). In this section we use the latter approach as it allows a certain amount of automation of the evaluation process [27]. The Gold Standard with which to evaluate our learned knowledge base was obtained from MusicBrainz[7], the most complete and accurate open knowledge base of music information. Instances of musical entities such as Recording, Artist, Release, etc. are identified

[7] `http://musicbrainz.org/`

by a universally unique identifier, a MusicBrainz ID. We extracted a subset of the MusicBrainz database containing all the entities that could be mapped to entities in our knowledge base, along with their corresponding relations. This mapping was accomplished as follows: for those entities in our knowledge base with a DBpedia URI (such as entities of type *person*, *band* and *album*) we obtain their MusicBrainz ID by first mapping the DBpedia URIs to Freebase (another open knowledge-base system) and then mapping Freebase IDs to MusicBrainz. This is because currently MusicBrainz IDs cannot be resolved directly from DBpedia. Regarding entities of type *song*, since we do not have a URI, we query the MusicBrainz API[8] by using song and artist name strings. Entities of type *musicgenre* were not considered for this evaluation as there is no corresponding concept in MusicBrainz. Finally, relations between the mapped entities were obtained using the aforementioned MusicBrainz API.

Of the 12838 entities in our knowledge base we could map 11740 entities in MusicBrainz, which represent a 91.4%. In order to evaluate both knowledge bases we removed those entities that could not be mapped to MusicBrainz. To facilitate the evaluation process we represented both our knowledge base and the Gold Standard as graphs, where nodes correspond to musical entities and edges represent relations between those entities. Some pairs of entities could have more than one relationship. For example, *artist* "Bob Ezrim" is related to *album* "The Wall" as `orchestration` and `producer` in MusicBrainz. In our case we simplified this by merging all these relation terms into a single edge with multiple labels. As a result of this process, our knowledge base and the MusicBrainz Gold Standard contained 13165 and 10595 edges, respectively.

As a first evaluation we calculated the overlap of edges between the two graphs, regardless of the labels (i.e, the relation concepts) of those edges. We obtained an overlap of 5236 edges, which represented a 49.4% of the Gold Standard relations and a 39.8% of our extracted knowledge base. Once this overlap is obtained, the next step is to assess how our knowledge base "fits" the MusicBrainz Gold Standard [4]. Evaluating two ontologies, or in this case two knowledge bases, is an arduous task. Traditional Information Retrieval evaluation measures such as precision and recall cannot be easily used in their strict sense, as there is no clear definition of what knowledge is acquired [4]. The main problem in our case is that the vocabularies used in the two knowledge bases are different. Nevertheless, even though the vocabularies are different, many of their terms refer to similar music-related concepts. Hence, finding a conceptual equivalence between relation terms in our knowledge base and the MusicBrainz Gold Standard is fundamental in order to evaluate our approach in terms of precision and recall. Of the overlapping 5236 edges, we selected all the distinct combinations of the MusicBrainz relation terms (i.e, labels) and our knowledge base relation terms that co-occur in the same edges and grouped them by relation type. A relation type in this case is defined as a relation between a pair of types of entities. For example, *person* $\leftrightarrow$ `member of band` $\leftrightarrow$ *band* is a relation type, where `member of band` is the relation between an entity of type *person*

[8] `http://musicbrainz.org/doc/Development/XML_Web_Service/Version_2`

and an entity of type *band*. A closer look at the Gold Standard graph shows that many relations in this graph do not have labels. For example, many artists are related to recordings in MusicBrainz without any explicit relation concept. Also, as mentioned previously in Section 4.1, our knowledge graph had some artificially added relations (the "by" relation between songs and artists). We thus decided to ignore these relations from our evaluation.

The grouping of the relation terms in relation types resulted in 727 different combinations, for which an equivalence had to be computed. A MusicBrainz relation type is considered to have an equivalent relation type in our knowledge base if their relation terms are conceptually similar. For example, the relation term `married` in MusicBrainz is conceptually implicit in the relation term `husband` in our knowledge base. Futhermore, MusicBrainz also organizes its relation terms in tree-like taxonomies, where conceptually similar terms are grouped in the same tree branch[9]. This can be used to decide whether a term in our knowledge base can be mapped to a term in the MusicBrainz relation taxonomies. In order to compute the equivalence of the 727 combinations we asked three human annotators to vote whether the two relation terms are conceptually similar. Due to lack of space, we made the votings available at `http://goo.gl/uOGjlo`.

Once this equivalence is obtained, precision and recall can be computed at an edge level. For this evaluation we only use a subset of the graphs. The subset is defined by all the overlapping edges in both graphs with at least one relation term. For each edge in the graphs, precision refers to how many relation terms in our knowledge base edge have an equivalence in the Gold Standard edge, whilst recall refers to how many relation terms in the Gold Standard edge have an equivalence in our knowledge base edge. Lets use the previous example of *artist* "Bob Ezrim" related to *album* "The Wall" as `orchestration` and `producer` in MusicBrainz. The relation between "Bob Ezrim" and "The Wall" in our knowledge base is defined by the single term `producer`. In this case, precision will be 1, but recall will be 0.5. We computed the average precision and recall over the 1143 total overlapping edges and obtained a score of 0.74 and 0.72, respectively[10]. These scores show a high correlation between MusicBrainz and our approach, which can confirm the veracity of many relations in the *songfacts.com* website. This could suggest that a combination of both knowledge bases might increase the completeness of metadata in MusicBrainz. The assessment of such assumption is though left for future work.

4.4 Recommendation Experiment

The aim of this experiment is to check the suitability of the extracted knowledge for music recommendation, and test the utility of explaining relations between songs. Although there are several approaches to compute recommendations using knowledge graphs with proven good performance [23], our approach was reduced to finding shortest paths between entities of type *song* in the graph. This baseline

[9] `https://musicbrainz.org/relationships`

[10] the individual precision and recall scores are available at `http://goo.gl/C4Coj3`

approach was selected for simplicity reasons, as the aim of the experiment was not to measure the performance of the recommendation system.

The experiment involved 30 participants, 24 males and 6 females, from 24 to 51 years old and with different musical background and listening habits. Most of the participants affirmed that they had previous experience with recommendation systems. In this experiment a set of recommended songs with textual explanations is presented to the participants. First, the participant is asked to choose a list of 10 songs from different artists she likes among all the songs in our dataset. Then, a new page with a list of the 10 seed songs is displayed, along with one recommended song per seed song and a textual explanation showing the relation between the two songs. The following is an example of how the explanations are given to the users:

Seed song: *Cloud Of Unknowing* by Gorillaz
Explanation:
Bobby Womack (performance on) Cloud Of Unknowing
Bobby Womack (legend played on) Shake
Recommended song: *Shake* by Sam Cooke

Participants could listen to a 30 seconds preview of the songs. They were asked to rate the recommendations and the explanations (with a 1-5 rating scale), and to select whether the explanations influenced their ratings. Finally, participants were asked whether they were familiar with the recommended songs.

		Influence of the explanations		
	Total	Positive	Negative	No influence
Rec.	$3.13_{\pm 1.32}$	$3.95_{\pm 1.02}$	$2_{\pm 0.91}$	$2.74_{\pm 1.23}$
Exp.	$3.18_{\pm 1.21}$	$3.99_{\pm 0.81}$	$2.18_{\pm 1.3}$	$2.74_{\pm 1.02}$

Table 2. Mean and standard deviation of ratings

	Positive	Negative	No influence
Known song	34.12%	15.29%	50.59%
Unknown song	**45.00**%	12.50%	42.50%

Table 3. Percentage of influence of the explanations

A total of 279 answers (corresponding to individual song recommendations) were collected[11], from which the participants knew only 81 recommended songs. The experiment yielded an average recommendation rating score of 3.13 ± 1.12 and an explanation rating score of 3.18 ± 1.21. Recommendation scores around 3 are typical average ratings for unknown recommendations [8]. Interestingly, the authors of [8] emphasize the need for adding context to the recommendations. In this experiment we provide our users with explanations of the recommendations. From the total of 279 answers, 41.22% of the explanations were marked by participants to positively influence their recommendation ratings, while 13.98%

[11] Some participants did not rate all the 10 recommended songs.

were marked as negative influence and 44.48% as to not having influenced the ratings at all. Indeed, Table 2 shows that when the explanations are positively influencing the ratings, the average recommendation rating score increases by 0.82 (from 3.13 to 3.95). Furthermore, we also calculated the correlation between the influence of the explanations in the ratings and the familiarity of the user with the recommended songs, as shown in Table 3. It is interesting to note that the number of ratings with a positive influence is about 10% higher when the recommended song is unknown to the user. This might suggest that explanations are indeed helping users to appreciate the recommendations more.

5 Conclusions

In this paper we presented a method for the creation of datasets for Music Recommendation that exploits information extracted from unstructured text sources. The method identifies music-related entities in the text (such as *songs*, *bands*, *persons*, *albums* and *music genres*) and extracts relations between these entities using an unsupervised rule based approach. The entities and relations are then represented as a graph from where song recommendations can be computed. A good characteristic of our approach is that a recommender system may provide explanations of the recommendations using natural language. We tested our method with a dataset gathered from *songfacts.com*, an online database of facts and stories about songs. We evaluated the extracted relations from a linguistic perspective and the extracted knowledge by comparing it with an existing knowledge base. We also performed a music recommendation experiment based on the extracted knowledge with real users. Evaluation results showed that our method is able to extract relations with a high linguistic and conceptual precision. It also shows that provide explanations with recommendations influence user satisfaction positively, especially when the recommendations are unknown to the user.

Still, there are many avenues for future work. Although the evaluation of our relation extraction system shows good values in terms of precision, recall is low between several pairs. One possible improvement of our approach is to introduce a prior step consisting in syntactic simplification. This would enable capturing potentially noisy relations (which are frequent in text featuring high variability such as the one displayed in Songfacts). Exploring alternative techniques to extract and represent relations between two or more entities is also crucial. In addition, new extracted knowledge could be used to enhance existing ontologies (such as MusicBrainz). All in all, our method provides a first attempt towards exploiting Knowledge Acquisition for Music Recommendation.

6 Acknowledgments

The authors would like to thank Miguel Ballesteros for his valuable advice and the subjects of the online experiment for their feedback.

References

1. Ballesteros, M., Nivre, J.: Going to the roots of dependency parsing. Computational Linguistics 39(1), 5–13 (2013)
2. Banko, M., Cafarella, M.J., Soderland, S., Broadhead, M., Etzioni, O.: Open Information Extraction from the Web. In: International Joint Conferences on Artificial Intelligence. pp. 2670–2676 (2007)
3. Bohnet, B.: Very high accuracy and fast dependency parsing is not a contradiction. In: Proceedings of the 23rd International Conference on Computational Linguistics. pp. 89–97 (2010)
4. Brewster, C., Alani, H., Dasmahapatra, S., Street, P., Wilks, C.B.Y.: Data Driven Ontology Evaluation. International Conference on Language Resources and Evaluation (2004)
5. Bunescu, R.C., Mooney, R.J.: A shortest path dependency kernel for relation extraction. In: Proc. of the conference on HLT/EMNLP. pp. 724–731 (2005)
6. Carlson, A., Betteridge, J., Wang, R.C., Hruschka Jr, E., Mitchell, T.M.: Coupled Semi-Supervised Learning for Information Extraction. In: Proc. of the third ACM WSDM. pp. 101–110 (2010)
7. Celma, Ò.: Music Recommendation and Discovery - The Long Tail, Long Fail, and Long Play in the Digital Music Space. Springer (2010)
8. Celma, Ò., Herrera, P.: A new approach to evaluating novel recommendations. In: Proceedings of the 2008 ACM conference on Recommender systems. pp. 179–186. ACM (2008)
9. Cohen, J.: Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. Psychological bulletin 70(4), 213 (1968)
10. Culotta, A., Sorensen, J.: Dependency tree kernels for relation extraction. In: Proc. of the 42Nd Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics (2004), http://dx.doi.org/10.3115/1218955.1219009
11. Dellschaft, K., Staab, S.: On How to Perform a Gold Standard Based Evaluation of Ontology Learning. Lecture Notes in Computer Science. The Semantic Web ISWC 2006 4273 (2006)
12. Eichler, K., Hemsen, H., Neumann, G.: Unsupervised relation extraction from web documents. In: Proceedings of the 6th International Conference on Language Resources and Evaluation. ELRA (2008)
13. Fader, A., Soderland, S., Etzioni, O.: Identifying Relations for Open Information Extraction. In: Empirical Methods in Natural Language Processing (2011)
14. Gamallo, P., Garcia, M., Fernández-Lanza, S.: Dependency-based open information extraction. In: Proceedings of the Joint Workshop on Unsupervised and Semi-Supervised Learning in NLP. pp. 10–18. ROBUS-UNSUP '12, Association for Computational Linguistics, Stroudsburg, PA, USA (2012)
15. Gangemi, A.: A Comparison of Knowledge Extraction Tools for the Semantic Web. In: The Semantic Web: Semantics and Big Data, pp. 351–366. Springer Berlin Heidelberg (2013)
16. Geleijnse, G., Korst, J.H.: Web-based artist categorization. In: Proc. of ISMIR. pp. 266–271 (2006)
17. Gruhl, D., Nagarajan, M., Pieper, J., Robson, C., Sheth, A.: Context and Domain Knowledge Enhanced Entity Spotting In Informal Text. In: The Semantic Web-ISWC, pp. 260–276. Springer (2009)

18. Hoffmann, R., Zhang, C., Ling, X., Zettlemoyer, L., Weld, D.S.: Knowledge-based weak supervision for information extraction of overlapping relations. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1. pp. 541–550. HLT '11, Association for Computational Linguistics, Stroudsburg, PA, USA (2011), http://dl.acm.org/citation.cfm?id=2002472.2002541
19. Isabelle Augenstein, D.M., Ciravegna, F.: Relation extraction from the web using distant supervision. In: In proceedings of the 19th International Conference, EKAW 2014 (2014)
20. Marrero, M., Urbano, J., Sánchez-Cuadrado, S., Morato, J., Gómez-Berbís, J.M.: Named Entity Recognition: Fallacies, challenges and opportunities. Computer Standards & Interfaces 35(5), 482–489 (2013), http://linkinghub.elsevier.com/retrieve/pii/S0920548912001080
21. Mausam, Schmitz, M., Bart, R., Soderland, S., Etzioni, O.: Open Language Learning for Information Extraction. In: Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (2012)
22. Mendes, P.N., Jakob, M., García-silva, A., Bizer, C.: DBpedia Spotlight : Shedding Light on the Web of Documents. Proc. of the 7th International Conference on Semantic Systems (2011)
23. Ostuni, V., Di Noia, T., Mirizzi, R., Di Sciascio, E.: A linked data recommender system using a neighborhood-based graph kernel. In: E-Commerce and Web Technologies, Lecture Notes in Business Information Processing, vol. 188, pp. 89–100. Springer International Publishing (2014)
24. Passant, A.: dbrec - Music Recommendations Using DBpedia. In: The Semantic Web–ISWC. vol. 1380, pp. 209–224. Springer (2010)
25. Passant, A., Raimond, Y.: Combining Social Music and Semantic Web for music-related recommender systems. In: Social Data on the Web Workshop (2008)
26. Schedl, M., Widmer, G., Knees, P., Pohle, T.: A music information system automatically generated via web content mining techniques. Information Processing & Management 47, 426–439 (2011)
27. Serra, I., Girardi, R., Novais, P.: Evaluating techniques for learning non-taxonomic relationships of ontologies from text. Expert Systems with Applications 41 (2014)
28. Surdeanu, M., Johansson, R., Meyers, A., Màrquez, L., Nivre, J.: The conll-2008 shared task on joint parsing of syntactic and semantic dependencies. In: Proceedings of the Twelfth Conference on Computational Natural Language Learning. pp. 159–177. Association for Computational Linguistics (2008)
29. Tesnière, L.: Elements de syntaxe structurale. Editions Klincksieck (1959)
30. Whitman, B., Lawrence, S.: Inferring descriptions and similarity for music from community metadata. In: Proc. of the 2002 International Computer Music Conference. pp. 591–598 (2002)
31. Yosef, M.A., Hoffart, J., Bordino, I., Spaniol, M., Weikum, G.: AIDA : An Online Tool for Accurate Disambiguation of Named Entities in Text and Tables. Proc. of the 37th International Conference on Very Large Databases pp. 1450–1453 (2011)