A contribution to semantic indexing and retrieval based on FCA - An application to song datasets

Víctor Codocedo[1]*, Ioanna Lykourentzou[1,2]**, and Amedeo Napoli[1]

[1] LORIA - CNRS - INRIA - Univesité de Lorraine, BP 239, 54506 Vandœuvre-les-Nancy.
victor.codocedo@loria.fr, amedeo.napoli@loria.fr,
[2] Centre de Recherche Public Henri Tudor - 29, avenue John F. Kennedy L-1855 Luxembourg-Kirchberg, Luxembourg
ioanna.lykourentzou@tudor.lu

Abstract. Semantic indexing and retrieval is an important research area, as the available amount of information on the Web is growing more and more. In this paper, we introduce an original approach to semantic indexing and retrieval based on Formal Concept Analysis. The concept lattice is used as a semantic index and we propose an original algorithm for traversing the lattice and answering user queries. This framework has been used and evaluated on a song dataset.

1 Introduction

Semantic indexing and retrieval refer to organizing a set of information items inside an index according to the semantic relations and concepts that they share, and then searching within this index to identify the items, the context of which matches a given user query [7, 15]. Semantic retrieval is based on flexible and partial matching techniques contrasting exact matching techniques. The organization and retrieval of information based on *context*, if effectively carried out, can significantly improve the understanding of information, as well as to provide users with richer and more meaningful search results, understanding *context* as a set of "external elements" which helps to understand or to manipulate information. For these reasons, context-based methods of classification and retrieval are applied on multiple types of information, ranging from text-based documents to multimedia content.

In the past years, Formal Concept Analysis (FCA [6]) has been applied to document indexation (an Information Retrieval task [10]) since it proposes a robust and formal framework to exploit the relations that documents (objects) have through the terms they share (attributes). For example, the work of Priss [13] uses concept lattices to improve the representation of a document collection by merging it with information from thesauri and thus creating a multi-faceted extended context. In a similar approach, the work of Carpineto et al. [4] presents CREDO, a system that queries Google to construct a faceted browser from a concept lattice to help the user on its search experience.

* Part of the Quaero Programme, funded by OSEO, French State agency for innovation.
** Supported by the National Research Fund, Luxembourg, and cofunded under the Marie Curie Actions of the European Commission (FP7-COFUND).

Different from browsing support and automatic facet construction, some approaches use the concept lattice directly as a document index and propose different strategies to explore it in order to find those documents relevant for a given user query. In general, given a document-term concept lattice and a conjunctive user query, these approaches work by identifying a formal concept in the lattice that best represents the query. This formal concept is then denominated as the "query concept". Two strategies can be distinguished to explore the concept lattice in order to retrieve documents: the work in [2] proposes a neighbourhood expansion strategy where concepts are ranked according to the minimal distance they have from the "query concept". In [11], the strategy used is the exploration of the super-hierarchy (super-concepts) of the "query concept". Other approaches, like the system FooCA presented in [8], while based on the FCA framework, do not rely on the lattice structure for an automated document retrieval. As described in [3], there are few works in the area of concept lattice-based information retrieval, and even though the first approaches were presented more than 40 years ago [18], the state-of-the-art remains little explored.

In this paper we propose a semantic indexation and retrieval technique supported over the FCA framework. It is based on the basic general idea of constructing a document-term concept lattice and identifying a "query concept", where we propose a novel exploration strategy based on the notion of "cousin concepts". We illustrate this technique using a song dataset, where songs are indexed using the terms appearing in their lyrics (songs are documents). The specific goal of this approach is to retrieve relevant songs to a user based on the terms provided in his query using a concept lattice as a semantic index. In order to enrich the song descriptions we use Wordnet (an external knowledge source which describes semantic relations among terms). As an additional characteristic we address the problem of semantic indexing and retrieval as a 3-step knowledge discovery on databases (KDD) process, comprising three main steps: data preparation, discovery and filtering of the results.

The main contributions of this work are the following:

- The notion of cousin concepts.
- Highlighting the capabilities of using FCA and a concept lattice as a semantic index and proposing a novel algorithm that traverses the lattice to retrieve information based on the content of the concepts.
- Proposing the incorporation of semantic indexing to current song retrieval systems and providing initial results, as a proof-of-concept of the potential that such an application can have.

We have selected songs indexing as an application domain since few work have been done in content-based indexation using lyrics. To the authors knowledge, the work in this area mostly focus on using low-level features of songs [19, 5, 9] (such as bit-rate, authors name, length, etc.) while high-level features (such as the semantics of the lyrics) have been used mainly for sentiment analysis classification [12, 17].

The rest of this paper is organized as follows. Firstly, Section 2 introduces the use of FCA for semantic indexing and querying and present our approach according to the main steps of KDD. Next, Section 3 presents the evaluation results obtained for our approach. Finally, Section 4 presents a discussion of the extension of our work and the conclusions of our research.

Table 1: Synsets retrieved for the word "soldier"from Wordnet

Synset	Synonyms	Definition
soldier.n.01	soldier	an enlisted man who serves in an army
soldier.n.02	soldier	a wingless sterile ant or termite having...
soldier.v.01	soldier	serve as a soldier in the military

2 Semantic indexing based on FCA

The problem addressed in this paper can be defined as finding songs the lyrics of which are *related* to a set of user-provided keywords, through a sufficient "closeness of meaning". Subsequently, our goal is to construct a semantic index, from a given set of songs and their lyrics and a relation which will successfully support the context-based song retrieval.

To address the above we first constructed a dataset of songs, where each song is related to a number of semantic meanings as follows. Two sources of data, namely musiXmatch and WordNet were used. MusiXmatch is a lyrics database, recently released as part of the MillionSong Dataset [1]. It was used to provide the lyrics for each song, in the form of a bag-of-words, already preprocessed to eliminate morphological word duplications. WordNet[3] is a well-known semantic dictionary and it was used to associate every word in the lyrics of a song with a set of synonym terms, called synsets, where each synset corresponds to one specific meaning of the word. Synsets also have semantic distances to one another, based on their position within WordNet's semantic hierarchy. The created dataset includes 357 songs, where each song is represented by its title, lyrics (in the form of a bag-of-words) and each word of the lyrics is connected to a set of synsets.

In the following, a song s_i is defined as a pair $\{t_i, L_i\}$ where t_i denotes the title and L_i the lyrics of the song in the form of a bag-of-words, as provided by musiXmatch.

2.1 Task 1: Lyrics Annotation

Given a song $s_i = \{t_i, L_i\}$, let $synsets(L_i)$ be the list of WordNet synsets that are associated with each word w_j in the lyrics L_i. Each retrieved synset has a definition and a set of synonyms. Word w_j is part of the synonyms. As an example, Table 1 illustrates the synsets retrieved from WordNet for the word "Soldier".

From the above example it can be understood that not every synset retrieved through WordNet is valuable in the context of a song. For instance, in the context of a war, a reference to a "soldier" would be clearly related to the definition of an *enlisted man who serves in an army* and not to the definition of a *soldier ant*. The third definition can also be disregarded since it is associated with a verb.

Therefore, to accurately annotate each song, we need to keep only those synsets that correspond to the actual context of the song.

[3] `http://wordnet.princeton.edu`

To address this, a filtering process took place as follows: a well-known similarity metric, namely the Wu-Palmer Similarity Measure [20], was used to measure the semantic similarity between every pair of synsets in the $synsets(L_i)$ set. The Wu-Palmer similarity measure $wp(ss_1, ss_2) = [0,1], ss_1 \in synsets(L_i) \wedge ss_2 \in synsets(L_i)$ is provided by the WordNet API and measures semantic similarity using path distance and the difference of levels in the synset tree. Then for each synset ss_j we calculate the average distance with every other synset $ss_k \in synsets(L_i)$ as defined in equation 1.

$$avg\_sim(ss_j) = \frac{\sum_{j \neq k} wp(ss_j, ss_k)}{|synsets(L_i)|} \tag{1}$$

The synset with the lowest avg_sim is deleted from $synsets(L_i)$. The filtering is repeated until we reach to a threshold of 20 most similar elements in $synsets(L_i)$, which will be considered to constitute the so-called *semantic core* of the song s_i, denoted as $core(s_i)$. The threshold of 20 synsets was selected heuristically, since it was found to represent the semantic core of the songs, in the specific dataset, in a concise and non redundant way. As an example, Table 2 shows some of the synsets selected to describe the song titled “The Green Beret Balad”. As it may be observed, the synsets that do not belong to the song’s context (which is mostly about the notions of war, battle, man, etc.) will be less related to the rest of the song’s synsets, and therefore they will be more likely to be omitted. In this example, the synset “soldier.n.01: an enlisted man or woman who serves in an army” has been selected, instead of the other WordNet’s alternative “soldier ant: soldier.n.02: a wingless sterile ant or termite having a large head and powerful jaws adapted for defending the colony”.

Table 2: Synsets describing the “Green Beret Balad” song

Synset	Mean Similarity	Definition
serviceman.n.01	0.665785412147	someone who serves in the armed forces
young.n.01	0.652204776648	any immature animal
man.n.03	0.652204776648	the generic use of the word to refer to any human being
soldier.n.01	0.629664596273	an enlisted man or woman who serves in an army
brave.n.01	0.622338466487	a North American Indian warrior
green_beret.n.01	0.606135516657	a soldier who is a member of the United States...
back.n.04	0.596281910309	(football) a person who plays in the backfield
wing.n.06	0.596281910309	a hockey player stationed in a forward position on either side
son.n.01	0.594639167088	a male human offspring
wife.n.01	0.594639167088	a married woman; a man’s partner in marriage
valet.n.01	0.569009183037	a manservant who acts as a personal attendant to his employer

The outcome of the filtering process is the set $core(s_i)$, which is considered as the final set of semantic annotations of the song s_i since it refers to a well-defined semantic schema, i.e. WordNet, where each annotation contains a definition and relations with other annotations.

2.2 Task 2: Semantic Index Creation

We built the semantic song index as a concept lattice using Formal Concept Analysis (FCA) following the lines of [11, 4, 13]. The basics of FCA are introduced in [6].

In the formal context of songs considered in this paper $\mathcal{K} = (G, M, I)$, the set of objects G contains the songs of the dataset while the set of attributes M contains all the WordNet synsets included in the semantic cores of the songs in G. The set I contains the relations gIm which stand for "song g has synset m in its semantic core". Table 3 shows an example of a formal context created from 11 songs and 6 synsets. The concept lattice obtained from this example context is illustrated in Figure 1. The concept lattice is presented in its *reduced notation* where objects (songs) and attributes (synsets) are shown only next to their object/attribute-concept, i.e. the most general concept introducing the attribute (which is inherited from higher to lower levels), the most specific concept having the object in its extent (the object being shared from lower to higher levels).

	bolshevik.n.01	son.n.01	man.n.03	white.n.01	serviceman.n.01	buddy.n.01
song1		x	x		x	
song6						x
song10		x	x		x	
song14		x	x		x	
song16	x	x	x	x	x	
song18		x	x		x	
song24			x		x	
song27	x	x				x
song32		x	x		x	x
song33			x	x		
song39	x		x	x	x	x

Table 3: Formal context example.

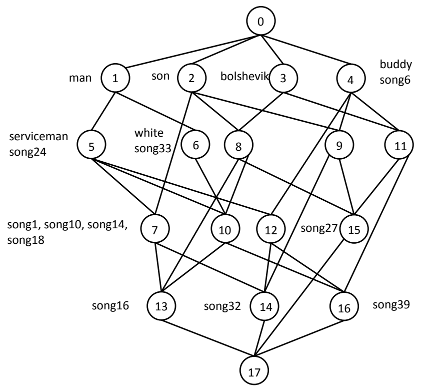

Fig. 1: The semantic index as a concept lattice obtained through FCA. Each concept is labelled with a unique identifier.

2.3 Task 3: Semantic Index querying

A simple query to the constructed semantic index (i.e. the concept lattice) is a pair $q = (A_q, B_q)$ where A_q denotes an empty extent to be filled and $B_q = \{ss\}$ is a synset to be searched for. Actually, the retrieval is based on two steps. The first one corresponds to "exact matching" (as in [11]) and the second corresponds to "partial matching" based on the cousin relation introduced hereafter. The first step consists of searching within the concept lattice for the attribute-concept (A_{ss}, B_{ss}) of attribute ss, i.e. finding the most general concept where ss appears in an intent (also denoted by $\mu(ss)$ in [6]). The extent of A_{ss} contains the list of all songs which are directly associated with the synset ss. This *direct answer* constitutes only a part of the answer. The second step is related to partial matching based on the cousin relation defined in the following.

Definition of cousin concepts. Two concepts (A_1, B_1) and (A_2, B_2) which are not comparable for $\leq_{\mathcal{K}}$ are said to be *cousins* iff there exists $(A_3, B_3) \neq \perp$ such that $(A_3, B_3) \leq_{\mathcal{K}} (A_1, B_1)$ and $(A_3, B_3) \leq_{\mathcal{K}} (A_2, B_2)$ and $d_{\mathcal{K}}((A_2, B_2), (A_3, B_3)) = 1$ (or $d_{\mathcal{K}}((A_1, B_1), (A_3, B_3)) = 1$), where $\perp$ is the bottom concept and $d_{\mathcal{K}}$ measures the minimal distance between two formal concepts in the lattice $\mathcal{K}$. Intuitively, this means that (A_1, B_1) and (A_2, B_2) do not subsume each other and that (A_3, B_3) can be either the lower bound or be subsumed by the lower bound $(A_1, B_1) \sqcap (A_2, B_2)$ (where $(A_1, B_1) \sqcap (A_2, B_2)$ denotes the lower bound of (A_1, B_1) and (A_2, B_2). Actually, (A_3, B_3) represents songs related to both (A_1, B_1) and (A_2, B_2): two songs are related if their semantic cores share some elements, which is the case here, as $A_3 \subseteq A_1 \cap A_2$ and $B_3 \subseteq B_1 \cup B_2$. For example, in Figure 2, concept 3 is a cousin of 2 because of concept 8, concept 11 is a cousin of concept 12 because of concept 16 and so on.

For a given attribute concept (A_{ss}, B_{ss}), the querying algorithm traverses the lattice to extract all *cousin concepts* (A_i, B_i) of the synset ss, and then it moves down the concept lattice, repeating the same extraction level by level. It should be noticed that the original synset query ss is not present in any of the intents of the *cousin concepts* B_i, this is why we can speak of "partial matching". Every cousin concept (A_i, B_i) is ranked according to the intersection that its extent has with the extent of the original attribute concept using the following metric:

$$rank(A_i, A_{ss}) = \frac{|A_i \cap A_{ss}|}{|A_i|} \tag{2}$$

This metric is two-fold since it allows the detection of concepts (A_i, B_i) which are far from the original concept and share no common objects with the extent of (A_{ss}, B_{ss}) ($A_i \cap A_{ss} = \emptyset$ and rank = 0) and those that are too abstract and describe too many objects ($|A_i| \gg |A_{ss}|$ and rank $\sim$ 0).

Hereafter, we give details on the steps of the querying algorithm through the use of an example, graphically illustrated in Figure 2. Let us consider a user query for songs related to the synset "bolshevik.n.01" (concept 3 on Figure 2).

1. Find the attribute-concept (A_{ss}, B_{ss}) for the synset $\{ss\}$: concept 3.
2. Find the sub-hierarchy of (A_{ss}, B_{ss}) in the concept lattice, i.e. all concepts subsumed by (A_{ss}, B_{ss}) and order them by levels: concepts 8, 11, 10, 15, 13, 16, 17 (solid arrows in Figure 2).
3. For each concept in this sub-hierarchy, find the super-concepts which are cousin concepts of (A_{ss}, B_{ss}) (and then for the descendants of (A_{ss}, B_{ss})): concept 2 is a cousin of 3 because of 8, 4 is a cousin of 3 because of 11, 6 is a cousin of 8 because of 10, etc. The final list is ordered by levels: concepts 2, 4, 5, 6, 9, 7, 12, 14 (dashed arrows in Figure 2).
4. Calculate the rank value of each cousin concept (according to Eq. 2) and sort these cousin concepts in descending order: concepts 6, 12, 9, 4, 2, 5, 7, 14.

5. The result is composed of the songs in the extent of the attribute-concept (A_{ss}, B_{ss}) and the extents of the cousin concepts.

The final result, in terms of the retrieved cousin concepts, their rankings and the songs in their extents, is shown in Table 4.

Concept	$rank$	Songs
3 (AC)		16,26,39
6	66%	33
12	50%	32
9	50%	
4	50%	
2	29%	1,10,14,18
5	25%	24
7	17%	
14	0%	

Table 4: Ranked list of retrieved songs for query "bolshevik.n.01"

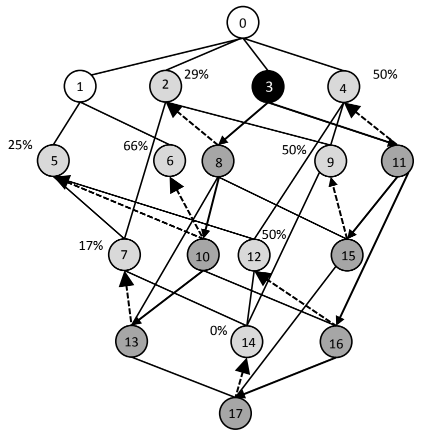

Fig. 2: Querying the semantic index. Starting from attribute concept 3, bold arrows show subhierarchy, dashed arrows show cousin concepts depicted next to their ranking value.

It can be noticed that the basic target of the algorithm proposed above is semantic retrieval. The order in which the algorithm presents the retrieved groups of songs to the user is a "recommendation decision", which could depend on user preferences and imply the use of a threshold to filter the final list of songs. Semantic retrieval does not necessarily imply the use of a threshold, which is why, for the scope of this paper, we present all the songs retrieved.

3 An application to song retrieval

As described in the previous section, the lattice is queried using a synset ss (e.g. *bolshevik.n.01*). This synset is directly related to a set of songs ($direct\_answer_{ss}$), i.e. those which are in the extent of the attribute concept (A_{ss}, B_{ss}) of that synset (in the case of *bolshevik.n.01*, songs 16, 27 and 39). These songs will be retrieved along with the set of songs found in the extents of the *cousin concepts* ($indirect\_answer_{ss}$) of (A_{ss}, B_{ss}) (songs 1, 10, 14, 18, 24, 32 and 33).

Regarding the songs in $direct\_answer_{ss}$, we are interested in examining whether our approach can find them if we apply it on a modified formal context where their relations with the synset ss have been eliminated. Of course, in this case, these songs cannot be retrieved as directly related songs, but only as songs found in the extents of *cousin concepts*. For example, if we eliminate the relation between song 16 and the synset *bolshevik.n.01* we want to know if this song can be retrieved by querying the new lattice using the synset *bolshevik.n.01*.

Regarding the set $indirect\_answer_{ss}$, we are interested in examining how it changes after the application of our approach on the modified formal context since the elimination of a *(song,synset)* relation will affect the structure of the concept lattice and hence the output of the proposed retrieval algorithm. Small variations in the content of this set will indicate robustness.

3.1 Leave-one-out cross validation, precision and recall

To evaluate the above and subsequently the definition of cousin concepts presented in section 2.3 we used and adapted the leave-one-out cross validation (LOOCV) methodology, which is a special type of cross validation [14]. Our adaptation consists of intentionally removing a single *(song,synset)* relation from a formal context (hereafter referred to as the *primary formal context*) and constructing its associated concept lattice (i.e. removing a cross from the primary formal context). The modified formal context is called a *scenario*. Therefore, each scenario is identified by a pair synset (ss_{scn}) and song (s_{scn}), the relation of which was eliminated for the scenario's construction.

For a given scenario, if song s_{scn} can be retrieved by querying for synset ss_{scn} we mark the scenario as *successful* (e.g. querying for *bolshevik.n.01* and retrieving song 16 for scenario with ss_{scn} = *bolshevik.n.01* and s_{scn} = song 16). The number of *successful* scenarios for a synset ss is given by $successful\_scenarios(ss)$. The *total number of scenarios* for a synset ss (given by $total\_scenarios(ss)$) is determined by the number of songs where the synset appears in (i.e. the number of crosses on the synset column in the primary formal context), since we only eliminate at each time a single *(song,synset)* relation from the primary formal context (e.g. for the synset *bolshevik.n.01* we construct 3 scenarios for songs 16, 27 and 39). We can then define $success\_rate(ss)$ as illustrated in equation 3.

$$success\_rate(ss) = \frac{successful\_scenarios(ss)}{total\_scenarios(ss)} \tag{3}$$

Because the relation between a song and a synset is eliminated in a given scenario, the song cannot be retrieved by the "exact matching" of the synset ss (the song will not be contained in $direct\_answer_{ss}$). Instead, the method proposed using cousin concepts has to be used and the song should be found through "partial matching". The test consist on observing if the song can be found in $indirect\_answer_{ss}$. Hence, the measure of *success_rate* represents the tendency of how related are those songs found through "partial matching" with the query and the usefulness of the proposed method.

To evaluate the changes in the set $indirect\_answer_{ss}$, we compare the full set of retrieved songs from each scenario with the respective set of songs retrieved from the primary concept lattice (i.e. the concept lattice constructed from the primary formal context). We calculate precision defined in Eq. 4 as the proportion of true positives over the retrieved list of songs in a scenario. Accordingly, we calculate recall, in Eq. 5, as the proportion of true positives over the retrieved list of songs in the primary concept lattice, i.e. the concept lattice without any removal. The expression $Ret(scn, ss)$ denotes the total set of songs retrieved from scenario scn $(direct\_answer_{ss} \cup indirect\_answer_{ss})$ querying for synset ss while $primary$ denotes the primary concept lattice.

$$precision(scn, ss) = \frac{Ret(scn, ss) \cap Ret(primary, ss)}{Ret(scn, ss)} \tag{4}$$

$$recall(scn, ss) = \frac{Ret(scn, ss) \cap Ret(primary, ss)}{Ret(primary, ss)} \tag{5}$$

3.2 Results

For each synset we calculate the mean precision and recall from all their scenarios. From our test set of 357 songs and 1848 synsets we selected 192 synsets and simulated 1027 scenarios (working with approximately 1000 scenarios allows a lower volume of computation and more different trials). Table 5 shows the values of these measures for 10 synsets. For example, it can be seen that synset anteroom.n.01 has relations with 4 songs. A *success_rate* of 1 means that all simulations were successful. Recall of 0.9 and Precision of 0.96 mean that for an elimination of 25% of the relations for the synset (1 over 4 songs), still 90% of the information was retrieved and 96% of the information was correct. There is a positive relation between the number of songs in which the synset appears and the *success_rate* measure. This is not strange since synsets appearing in a few songs will be in fewer concepts in the lattice and hence the simulation affects them in the worst manner. For example, for the synset bar.n.03, the elimination of one relation with a song leads to the elimination of 50% of its relations (1/2), while for the synset battle.n.01 the elimination of one relation with a song leads to the elimination of only 5% of its relations (1/20).

Figure 3 shows the distribution of *success_rate*, recall and precision (in the interval of $[0, 1]$ in axis y) over the number of songs where synsets appear in (in axis x). The *success_rate* maintains a growing tendency showing that better results are obtained with synsets which appear in a greater number of songs. In a wider sense, precision and recall maintain their values over 70% over all the samples. This is especially important in values of songs per synset below 5 since losing a single connection could disconnect songs more significantly. In the case of the first point (2.5 songs per synset) losing one connection means losing 40% of the connections of the attribute concept, however over 70% of the original set of songs is retrieved.

It should be noticed that a certain degree of bias, caused by the inclusion of the directly related songs in the measures of precision/recall, is to be expected. That is, given that for each scenario we are eliminating only one *(song, synset)* relation, the remaining directly related songs will be present in both sets retrieved when querying the scenario and the primary concept lattice. Therefore, the precision/recall measures are meant to be used, in the context of this paper, as a means of examining how the set of cousin concepts is affected for each synset, and they should not be considered as a medium of comparison with other information retrieval approaches.

Finally, even if more experiments have to be completed, we can conclude that the *definition of cousin concepts* is valuable and allows the use of a concept lattice as a semantic index to retrieve objects not directly related to a query.

synset	songs	success_rate	recall	precision
anteroom.n.01	4	1.0	0.9	0.964
bustle.n.01	3	0.333	0.564	0.611
ambition.n.01	9	0.888	0.888	0.938
child.n.03	13	0.923	0.945	0.982
arrest.n.02	4	0.25	0.75	0.807
battle.n.01	20	0.9	0.956	0.989
champion.n.02	2	0.0	0.083	1.0
better.n.03	3	0.0	0.641	0.694
attack.n.01	2	1.0	0.730	0.791
bar.n.03	2	0.0	0.083	1.0

Table 5: Simulations results.

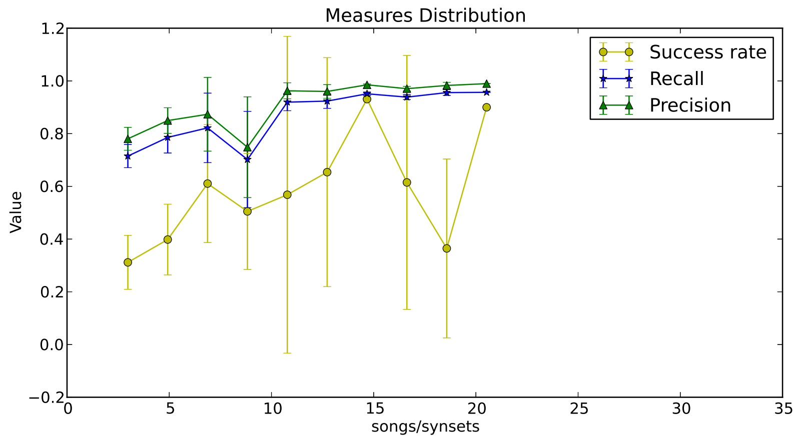

Fig. 3: Distribution of measures over songs per synset.

4 Discussion and Conclusion

In this paper we propose an approach for semantic indexing and retrieval of songs based on Formal Concept Analysis and exploiting the representation of a song's lyrics as a collection of relevant WordNet synsets.

A number of limitations may be found to the present work. First the evaluation focuses, at this stage, mainly on examining the robustness of the cousin concepts. This evaluation choice was made on the basis of two reasons: i) the definition of cousin concepts is the core of the proposed approach and therefore its efficiency is the first parameter that needs to be evaluated and ii) the literature reports an absence of other approaches that perform indexing and retrieval based on the semantics of song lyrics, and which could be used as a comparison benchmark. Given the above, and in view of the positive results that the current evaluation has yielded, the next stage is to proceed with a user evaluation study, which will allow us to examine the relativeness of the retrieved results according to the intended meaning of specific users' queries.

A second limitation refers to the requirements of the proposed approach in case it is intended for a large-scale application. Specifically, at this stage a relatively small dataset of 357 songs was used, however additional experiments would be necessary to examine how the size of the dataset affects the performance of the approach. Another necessity refers to defining in more detail the way that a user query can be matched to a set of synsets, which can then be used to query the lattice-based song index.

Future work includes two directions: i) enriching the semantic song representation with other information resources such as DBpedia and ii) expanding the proposed approach to different data collections.

On the first direction, additional information about the meaning of the songs can be found in external user-contributed sources, such as Wikipedia and its semantic equivalent DBPedia. Therefore, as a first future direction we plan to enrich the constructed semantic song representations with categorical knowledge, i.e. regarding the broader "topic" that each song is about, from DBPedia. To take advantage of this categorical knowledge we plan to extend the proposed FCA-based approach with *Relational Concept Analysis* [16], in order to create a semantic index where songs are related not only through their lyrics but also through their categories.

A second potential extension we consider applying the proposed approach to other types of information content, such as scientific papers or news information, to examine its generalization capability and to facilitate comparison with other benchmark techniques of semantic search and retrieval.

As a conclusion, in this paper we propose a novel contribution to the field of semantic indexing and retrieval, which is based on Formal Concept Analysis. We use the concept lattice as a semantic index and propose a novel algorithm to traverse the lattice in order to match user queries with semantically relevant information items. The approach was tested on a song dataset and the obtained results show good capabilities.

References

1. Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere, 'The million song dataset', in *Proceedings of the 12th International Conference on Music Information Retrieval*, (2011).
2. Claudio Carpineto and Giovanni Romano, 'Order-theoretical ranking', *Journal of the American Society for Information Sciencies*, **51**(7), 587–601, (May 2000).
3. Claudio Carpineto and Giovanni Romano, *Concept Data Analysis: Theory and Applications*, July 2004.
4. Claudio Carpineto, Giovanni Romano, and Fondazione Ugo Bordoni, 'Exploiting the potential of concept lattices for information retrieval with credo', *Journal of Universal Computer Science*, **10**, 985–1013, (2004).

5. Zhouyu Fu, Guojun Lu, Kai Ming Ting, and Dengsheng Zhang, 'A survey of audio-based music classification and annotation', *Multimedia, IEEE Transactions on*, **13**(2), 303 –319, (april 2011).
6. Benrhard Ganter and Rudoph Wille, *Formal Concept Analysis*, Springer, Berlin, 1999.
7. Atanas Kiryakov, Borislav Popov, Ivan Terziev, Dimitar Manov, and Damyan Ognyanoff, 'Semantic annotation, indexing, and retrieval', *Web Semantics: Science, Services and Agents on the World Wide Web*, **2**(1), 49 – 79, (2004).
8. Bjoern Koester, 'Conceptual knowledge retrieval with fooca: Improving web search engine results with contexts and concept hierarchies', in *Industrial Conference on Data Mining*, ed., Petra Perner, volume 4065 of *Lecture Notes in Computer Science*, pp. 176–190. Springer, (2006).
9. Mika Kuuskankare and Mikael Laurson, 'Mir in enp - rule-based music information retrieval from symbolic music notation', in *Proceedings of the 10th International Society for Music Information Retrieval Conference, ISMIR 2009*. International Society for Music Information Retrieval, (2009).
10. Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze, *Introduction to Information Retrieval*, Cambridge University Press, New York, NY, USA, 2008.
11. Nizar Messai, Marie-Dominique Devignes, Amedeo Napoli, and Malika Smal-Tabbone, 'Querying a bioinformatic data sources registry with concept lattices', in *Proceedings of ICCS 2005*, pp. 323–336. LNCS 3596, Springer, (2005).
12. Michael Kai Petersen, Lars Kai Hansen, and Andrius Butkus, 'Computer music modeling and retrieval. genesis of meaning in sound and music', chapter Semantic Contours in Tracks Based on Emotional Tags, 45–66, Springer, (2009).
13. Uta Priss, 'Lattice-based information retrieval', *Knowledge Organization*, **27**, 132–142, (2000).
14. Payam Refaeilzadeh, Lei Tang, and Huan Liu, 'Cross-validation', in *Encyclopedia of Database Systems*, eds., Ling Liu and M. Tamer Özsu, 532–538, Springer US, (2009).
15. Fuji Ren and David B. Bracewell, 'Advanced information retrieval', *Electronic Notes in Theoretical Computer Science*, **225**(0), 303 – 317, (2009).
16. Mohamed Rouane-Hacene, Marianne Huchard, Amedeo Napoli, and Petko Valtchev, 'A proposal for combining formal concept analysis and description logics for mining relational data', in *Proceedings of ICFCA 2007*, pp. 51–65. LNAI 4390, Springer, (2007).
17. Govind Sharma and M. Narasimha Murty, 'Mining sentiments from songs using latent dirichlet allocation', in *Proceedings of the 10th international conference on Advances in intelligent data analysis X*, IDA'11, pp. 328–339. Springer, (2011).
18. Dagobert Soergel, 'Mathematical analysis of documentation systems : An attempt to a theory of classification and search request formulation', *Information Storage and Retrieval*, 129–173, (1967).
19. Rainer Typke, Frans Wiering, and Remco C. Veltkamp, 'A survey of music information retrieval systems', in *Proceedings of the 6th International Conference on Music Information Retrieval*, (2005).
20. Zhibiao Wu and Martha Palmer, 'Verbs semantics and lexical selection', in *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, ACL '94, pp. 133–138, Stroudsburg, PA, USA, (1994). Association for Computational Linguistics.