MIREX 2014 SYMBOLIC MELODIC SIMILARITY: EXTRACTING SIMILAR MELODIES BASED ON TOP-N COLOSSAL PATTERN MINING

Yoshiaki OKUBO and Makoto HARAGUCHI
Graduate School of Information Science and Technology
Hokkaido University
N-14 W-9, Sapporo 060-0814, JAPAN
E-mail: {yoshiaki, mh}@ist.hokudai.ac.jp

ABSTRACT

In this report, we outline our system for retrieving similar melodies for a given query melody. We formalize our retrieval problem based on *frequent colossal pattern mining*. In our framework, sequences of pitch-differences with length ℓ extracted from a melody collection are regarded as primitive items. Then a frequent pattern (itemset) corresponds to a set of those sequences which are commonly appeared in at least a pre-defined minimum number of melodies in the collection. Given a query melody, we try to find frequent patterns appeared in both the query and melodies in the collection. Since there are in general a lot of those frequent patterns, we especially try to extract colossal ones with Top-N cardinality each of which provides us a set of candidate melodies similar to the query with higher similarity.

1. INTRODUCTION

For the task of *Symbolic Melodic Similarity* (*SMS*) in MIREX 2014, we propose a retrieval system based on *frequent pattern mining* [1]. In our framework, each melody in a given melody collection is represented as a set of primitive items, where an item is a sequence of pitch-differences with length ℓ extracted from melodies in the collection. Given a query melody q, the set of pitch-difference sequences (items) Q appeared in q is also extracted, and then we try to find an itemset $X \subseteq Q$ which appears in q and at least 10 melodies in the collection. Since q and those 10 (or more) melodies commonly have the itemset X, they are reasonably considered to be similar each other. We, therefore, regard those melodies as candidates to retrieve in our SMS search for the query.

There are in general many itemsets filling the above requirement. In order to obtain candidate melodies with higher similarity to q, we try to extract such itemsets with *Top-N cardinality* according to our previous framework of mining *Top-N colossal patterns* [4].

This document is licensed under the Creative Commons Attribution-Noncommercial-Share Alike 3.0 License.
http://creativecommons.org/licenses/by-nc-sa/3.0/
© 2010 The Authors.

Candidate melodies provided by the extracted patterns with Top-N cardinality are assigned a ranking in descending order of similarity to q, where we evaluate similarity regarding shared items with larger pitch-differences as important. Finally, the top-10 ranked melodies are presented to users as the final result of our SMS search for q.

2. CONVERTING MELODY COLLECTION INTO TRANSACTION DATABASE

2.1 Melody as Sequence of Pitch Differences

It is supposed that we are given a collection, $\mathcal{M}$, of monophonic melodies in Standard MIDI Format. We can extract a sequence of notes from each melody in the collection by picking up only *note-on events*. It is noted that we do not care duration of notes in melodies. Moreover, we regard any consecutive notes with the same pitch as a single note with the pitch. We, therefore, can obtain a sequence of note numbers from each melody in which any adjacent numbers are different.

In order to capture principal (up-and-down) movement of notes in melodies, we ignore any difference of keys in melodies. That is, each melody $x \in \mathcal{M}$ is regarded as a sequence of *pitch-differences*, denoted by $\hat{x}$, between any adjacent notes in the melody.

2.2 Extracting ℓ-Consecutive Pitch-Differences as Primitive Item

For each melody $x_i \in \mathcal{M}$, let us assume we have its corresponding sequence of pitch-differences $\hat{x_i} = \langle a_{i1}, a_{i2}, \ldots, a_{ik_i} \rangle$. Regarding such a sequence as a string, we extract the set of substrings with length ℓ from $\hat{x_i}$, that is, we can obtain at most $k_i - \ell + 1$ substrings. In our framework, such a substring is regarded as a primitive element called an *item*. Technically speaking, each item f can be represented as an ℓ-tuple $f = (d_1, \ldots, d_\ell)$, where d_{ij} is a non-zero integer. Thus, each melody is represented as the set of those items extracted from the melody.

2.3 Melody Collection as Transaction Database

Based on the above, we can easily convert the given melody collection $\mathcal{M}$ into a transaction database $TDB_{\mathcal{M}}$ defined

as

$$TDB_{\mathcal{M}} = \{(ID(x), trans(x)) \mid x \in \mathcal{M}\},$$

where $ID(x)$ is an identification of the melody x and $trans(x)$ the set of items extracted from x.

3. FINDING SIMILAR MELODIES BASED ON TOP-N COLOSSAL PATTERN MINING

Given a transaction database and a minimum frequency threshold σ, a problem of *frequent pattern mining* [1] is to enumerate every set of items, called a *frequent pattern*, which appears as a subset in at least σ transactions in the database. It is noted here that a frequent pattern can be viewed as one of the evidence that the σ (or more) transactions containing the pattern are similar each other. For a query melody q, therefore, if we can find some frequent pattern X in $TDB_{\mathcal{M}}$ which also appears in q, each transaction (melody) with X in $TDB_{\mathcal{M}}$ can be considered to be similar to q. Based on this simple idea, we formalize our task of retrieving similar melodies in terms of frequent pattern mining.

In general, there are many frequent patterns filling the above requirement. Since we have to retrieve melodies with higher similarity to q, it is reasonable to extract a frequent pattern X with as large cardinality as possible. Such a pattern with large cardinality is called a *colossal pattern* [2, 4].

In our previous work, we have proposed an algorithm for efficiently finding frequent patterns with Top-N cardinality [4]. Therefore, our SMS system can be designed based on our previous algorithm.

In the algorithm, the notion of *pattern-graph* [3] plays a very important role for efficient mining of target patterns. The pattern-graph can be constructed before pattern mining process. Although the construction is a bit time consuming task, we need to perform it just once. We, therefore, can construct it in our preprocess.

For the task of SMS, we can efficiently extract frequent patterns with Top-N cardinality which also appear in a query melody. Technically speaking, given a query melody q, we first identify the set of items, Q, which can be extracted from q. Then our original transaction database $TDB_{\mathcal{M}}$ is projected on Q. For the projected database denoted by $TDB_{\mathcal{M}}[Q]$, we try to find frequent patterns with Top-N cardinality. It is emphasized here that only a very small part of the items in the original database appear in $TDB_{\mathcal{M}}[Q]$. It is, therefore, strongly expected that those Top-N colossal patterns can be enumerated efficiently even if a large number of items are in the original database.

Let $\mathcal{X} = \{X_1, \ldots, X_m\}$ be the set of colossal patterns with Top-N cardinality found based on the above procedure, where $X_i \subseteq Q$ for any $i \in \{1, \ldots, m\}$. For each X_i, we can uniquely identify the set of melodies in $\mathcal{M}$ with X_i which is formally defined as

$$M(X_i) = \{x \mid x \in \mathcal{M} \wedge X_i \subseteq trans(x)\}$$

Thus, each colossal pattern X_i can provide a certain viewpoint of similarity. The melodies in $M(X_i)$ are considered similar in the sense that all of them commonly have X_i. That is, each extracted pattern brings us a set of candidate melodies which are similar to the query from the viewpoint. As the result, we can obtain a set of candidate melodies $\mathcal{C}$ expected to be similar to the query, defined as

$$\mathcal{C} = \bigcup_{X \in \mathcal{X}} M(X).$$

4. RANKING CANDIDATE MELODIES

After having a set of candidate melodies $\mathcal{C}$ for a query melody q, the melodies are assigned a ranking based on similarity to q. Although there exist various definitions of similarity, we take a quite simple definition as the first stage of our investigation. More concretely speaking, let Q be the set of items extracted from q and x a melody in $\mathcal{M}$. Then we define a similarity between q and x, denoted by $sim(q, x)$, as the average weight of common items:

$$sim(q, x) = \frac{1}{|Q \cap trans(x)|} \sum_{f \in Q \cap trans(x)} weight(f),$$

where $weight(\cdot)$ is a weight function defined on the items in $TDB_{\mathcal{M}}[Q]$.

Let us assume that each item f is represented as an ℓ-tuple $f = (d_1, \ldots, d_\ell)$. In our current framework, we regard items with larger pitch differences as more important because those items seem to be characteristic in melodies. Then we prefer melodies which share such characteristic items with the query melody. Based on this intuitive idea, we simply define our weight for an item $f = (d_1, \ldots, d_\ell)$ as

$$wight(f) = \sum_{i=1}^{\ell} |d_i|.$$

In order to assign our ranking to melodies in the set of candidates $\mathcal{C}$, for each melody $x \in \mathcal{C}$, we calculate the similarity between q and x, $sim(q, x)$, according to the above definitions. Then the melodies in $\mathcal{C}$ are sorted in descending order of their similarity values. Finally, the first 10 melodies are presented to users as the final output of our SMS system.

It should be noted here that two identical melodies do not necessarily have high similarity. From the definition above, it is observed that common items with small weights decreases their average weight. Hence, for two melodies x and y, if most of their common items have higher weight, we often have $sim(x, y) > sim(x, x)$ (or $sim(y, y)$). In the definition of our similarity, thus, common items with smaller weight could work negatively.

5. PARAMETER SETTING

Our system for SMS search has three parameters, ℓ, σ and N.

The parameter ℓ is for the length of pitch-difference sequences to be extracted as items. If it is too small (short), say 3 or less, it would be difficult to expect those items to present characteristics of individual melodies. Conversely,

if it is too large (long), it would not be probable that several melodies commonly have some of them. Therefore, our current system (tentatively) takes the setting of $\ell = 5$.

As has been mentioned previously, the parameter σ is for the minimum number of transactions in which our target patterns must appear. Practically speaking, the parameter directly affects the cardinality of target patterns to be extracted. As σ becomes smaller, cardinality of extracted patterns tends to be larger. Patterns with larger cardinality is desirable in our SMS search because such a pattern can work as a sufficient basis of high similarity. On the other hand, we have to find at least 10 melodies similar to a given query melody. Therefore, it is reasonable to set the parameter σ to 10 so that we can output at least 10 similar melodies as the minimum requirement of the SMS task.

In order to take various viewpoints of similarity into account, we assume $N = 3$ for Top-N in our current parameter setting. In our preliminary experimentation, we have empirically found that the setting of $N = 2$ or less seems too restrictive because we often obtain a very small number of patterns with the setting. Conversely, if N becomes larger, patterns with the N-th cardinality give us lower similarity and the number of extracted patterns tends to be too large.

6. CONCLUDING REMARKS

We briefly discussed in this report our current system for the task of SMS in MIREX 2014. Since it is at the first stage of our investigation, we have to examine many other ideas. Particularly, design of items would be the most influential in developing an excellent SMS system. In the context of data mining and machine learning, it is well-known as a kind of *"Feature Selection Problem"*. For example, we would have to pay our attention to the following points:

- Is ignoring duration of notes reasonable or not?
- What is an adequate length of pitch-difference sequences we have to extract as an item?

Both questions are deeply concerned with a fundamental theme:

> *Adequate abstraction level (granularity) for representation of symbolic melodies.*

Definitions of similarities and weight functions would also affect the final result of SMS search. Since our current definitions are quite simple, it would be better to take other useful factors into account. Incorporating a TFIDF-based weighting could be a promising approach.

7. REFERENCES

[1] J. Han, H. Cheng, D. Xin and X. Yan, Frequent Pattern Mining - Current Status and Future Directions, Data Mining and Knowledge Discovery, 15(1), 55 – 86, 2007.

[2] F. Zhu, X. Yan, J. Han, P. S. Yu and H. Cheng, Mining Colossal Frequent Patterns by Core Pattern Fusion, Proc. of the 23rd IEEE Int'l Conf. on Data Engineering - ICDE'07, 706 – 715, 2007.

[3] Y. Xie and P. S. Yu, Max-Clique: A Top-Down Graph-Based Approach to Frequent Pattern Mining, Proc. of the 2010 IEEE Int'l Conf. on Data Mining - ICDM'10, 1139 – 1144, 2010.

[4] Y. Okubo and M. Haraguchi: Finding Top-N Colossal Patterns Based on Clique Search with Dynamic Update of Graph, Proc. of The 10th Int'l Conf. on Formal Concept Analysis - ICFCA'12, LNAI-7278, pp. 244 – 259, 2012.