KIPS Trans. Softw. and Data Eng.
Vol.10, No.5 pp.195~200
ISSN: 2287-5905 (Print), ISSN: 2734-0503 (Online)

https://doi.org/10.3745/KTSDE.2021.10.5.195

Algorithm to Search for the Original Song from a Cover Song Using Inflection Points of the Melody Line

Bo Hyun Lee† · Myung Kim††

ABSTRACT

Due to the development of video sharing platforms, the amount of video uploads is exploding. Such videos often include various types of music, among which cover songs are included. In order to protect the copyright of music, an algorithm to find the original song of the cover song is essential. However, it is not easy to find the original song because the cover song is a modification of the composition, speed and overall structure of the original song. So far, there is no known effective algorithm for searching the original song of the cover song. In this paper, we propose an algorithm for searching the original song of the cover song using the inflection points of the melody line. Inflection points represent the characteristic points of change in the melody sequence. The proposed algorithm compares the original song and the cover song using the sequence of inflection points for the representative phrase of the original song. Since the characteristics of the representative phrase are used, even if the cover song is a song made by modifying the overall composition of the song, the algorithm's search performance is excellent. Also, since the proposed algorithm uses only the features of the inflection point sequence, the memory usage is very low. The efficiency of the algorithm was verified through performance evaluation.

Keywords : Music Search Algorithm, Inflection Point of Sequence, Sequence Similarity Calculation, Sequence Comparison

멜로디 라인의 변곡점을 활용한 커버곡의 원곡 검색 알고리즘

이 보 현† · 김 명††

요 약

동영상 공유 플랫폼의 발전으로 인해 동영상 업로드 분량이 폭발적으로 증가하고 있다. 그러한 동영상에는 다양한 형태의 음악이 포함되는 경우가 많으며, 그중에는 커버곡이 포함된다. 음악의 저작권을 보호하기 위해서는 커버곡의 원곡을 찾아내는 알고리즘이 필요하지만, 커버곡은 원곡의 조성, 속도와 전체적인 구성이 변형된 것이기 때문에 커버곡의 원곡을 찾기는 쉽지 않다. 이와 같이 변형된 커버곡으로부터 원곡을 검색하는 효율적인 알고리즘은 현재까지 알려진 바가 없다. 이에 본 연구에서는 멜로디 라인의 변곡점들을 활용한 커버곡의 원곡 검색 알고리즘을 제안한다. 변곡점은 멜로디 시퀀스에서 특징적인 변화 지점을 나타낸다. 제안하는 알고리즘은 원곡의 대표 구절에 대한 변곡점 시퀀스를 사용하여 원곡과 커버곡을 비교한다. 원곡의 대표 구절의 특징을 사용하기 때문에 커버곡이 전체적인 곡의 구성을 변형하여 만들어진 곡이라고 해도, 알고리즘의 검색 성능이 우수하다. 또한, 제안한 알고리즘은 변곡점 시퀀스의 특징만을 저장하고 사용하므로 메모리 사용량이 매우 적다. 알고리즘의 효율성은 성능평가를 통해 검증하였다.

키워드 : 음악 검색 알고리즘, 시퀀스의 변곡점, 시퀀스 유사도 계산, 시퀀스 비교

1. 서 론

최근 동영상 공유 플랫폼의 발전으로 폭발적으로 많은 다양한 종류의 영상들이 업로드되고 있다. 이에 따라 영상의 배경음이나 콘텐츠로 쓰이는 음악의 저작권 보호를 위해 음악 검색 알고리즘의 중요성이 대두되고 있다. 특히, 영상에 쓰이는 음악이 원곡 그대로인 경우에는 해당 곡을 식별하기가 쉽지만, 원곡을 변형하여 사용하는 경우에는 원곡의 검색이 용이하지 않다.

원곡을 변형하여 사용하는 대표적인 사례로 커버곡을 들 수 있다. 이는 '덮다, 씌우다'라는 뜻의 영어 단어 'cover'와 노래를 뜻하는 단어인 '곡'을 결합한 합성어로, 이미 있는 곡을 자신의 스타일대로 바꿔서 부르는 것을 뜻하며, 이 과정에서 곡의 조성이나 속도, 리듬, 전체 구성 등이 변하게 된다[1].

커버곡 검색 알고리즘에서 해결해야 할 이슈들은 ① 조옮김으로 인한 변화, ② 전체적인 빠르기 변화, ③ 곡 구성의 변화, 그리고 마지막으로 ④ 매 프레임의 특성값을 저장하는

※ 이 논문은 2020년 한국정보처리학회 추계학술발표대회의 우수논문으로 "멜로디 라인의 변곡점을 활용한 커버곡 검색 알고리즘"의 제목으로 발표된 논문을 확장한 것임.
† 준 회 원 : 이화여자대학교 컴퓨터공학과 석사과정
†† 종신회원 : 이화여자대학교 컴퓨터공학과 교수
Manuscript Received : December 18, 2020
First Revision : February 5, 2021
Accepted : February 26, 2021
* Corresponding Author : Myung Kim(mkim@ewha.ac.kr)

데 필요한 메모리 낭비 문제, 등으로 정리할 수 있다. 기존의 커버곡 검색 알고리즘들은 주로 ①번 문제 해결이 위주였고, ②번 문제를 해결하기 위한 알고리즘도 존재한다. 그러나 ③, ④번 문제에 대해서는 뚜렷한 성능을 내는 알고리즘은 아직 알려지지 않았다.

이에 본 논문에서는 멜로디 라인의 변곡점을 활용한 커버곡의 원곡 검색 알고리즘(Algorithm to Search for the Original song of a Cover song using Inflection points of the Melody line, 이하 SOC-IM)을 제안한다. 이는 멜로디의 절대적인 값이 아닌 상대적인 시퀀스에서 특징적인 부분인 변곡점의 정보를 활용하여 커버곡의 원곡을 검색한다. 또한, 원곡은 대표성을 갖는 구절만 데이터베이스에 저장하고 사용한다. 이로써 앞서 정리한 4가지 문제를 모두 해결할 수 있다.

본 논문은 다음과 같이 구성된다. 2장에서는 기존 커버곡 검색 알고리즘에 쓰이는 기술들에 대하여 설명한다. 그리고 3장에서는 SOC-IM 알고리즘의 핵심 아이디어와 전체적인 흐름을 기술한 후, 4장에서 이에 대한 세부적인 내용과 발전 과정에 관하여 기술한다. 제안한 알고리즘에 대한 성능 평가 결과를 5장에 기술하고, 마지막으로 6장에서 결론을 맺는다.

2. 이론적 배경

기존의 커버곡 검색 알고리즘은 음의 높이 변화나 길이 분석을 중심으로 이뤄지고 있다. 대표적으로 [1]은 옥타브 차이가 나는 음악의 성분들을 계산하여 전체 주파수를 하나의 옥타브 안에 넣어 표현한 크로마그램을 이용하였다. 이러한 방법으로는 계이름 정보는 잘 알 수 있지만, 옥타브 정보가 분실되기 때문에 음정 상호간의 상대적인 높낮이 등의 관계성은 고려하지 못한다. [2]는 음의 높이 변화 방향을 U(up), D(Down), S(same)라는 스트링으로 변환하여 사용하였고, [3]은 음의 높이 변화 정도를 정보에 포함하여 정확도가 향상된 검색 알고리즘을 제안하였다. 그러나 이러한 알고리즘들은 고정된 프레임을 사용하기 때문에 곡의 빠르기 변화에는 취약하다. [4]는 빠르기가 변형된 곡을 검색할 수 있는 비트 동기 크로마 기법을 제안하였으나, 이는 곡 전반의 빠르기가 일정할 때에만 유효하고, 비트 추출의 정확도가 전체 검색 성능에 큰 영향을 미친다는 한계점이 있다[1]. [5]의 Shazam과 같은 음악 검색 알고리즘은 음악 신호를 분석하여 일치하는 곡을 찾아내는 소리 지문 기술을 사용한다. 이는 정확도가 높고 속도가 빠르지만 원곡이 그대로 쓰이지 않는 경우 신호적 특징이 달라져 검색이 어렵다는 단점이 있다.

3. 제안 알고리즘 : SOC-IM

본 논문에서 제안하는 알고리즘인 SOC-IM의 핵심 아이디어는 다음과 같이 3가지로 정리할 수 있다. 첫째, 곡의 멜로디 라인을 사용한다. 이는 사람이 커버곡을 듣고 그의 원곡을 단번에 알 수 있는 이유는 원곡이 커버곡으로 바뀌는 과정에서 멜로디 라인, 즉 선율의 특징이 보존된다는 점에 착안해 고안한 방법이다[1]. 선율은 음의 높낮이와 길이의 조화이며, 흔히 악보에서 볼 수 있는 음표의 흐름으로 표현할 수 있다. 본 연구에서 사용하는 멜로디 라인은 Fig. 1과 같은 악보의 멜로디 라인을 Fig. 2와 같은 시퀀스로 데이터화 시킨 것이다. 음정은 음정을 표현하기 위해 붙이는 미디 채널 메시지 번호인 미디 노트 넘버로 변환하고, 해당 음정의 길이는 16분음표마다 타임스탬프 1로 치환하여 4/4박자를 기준으로 한 마디에 16개의 데이터가 생성되도록 하였다.

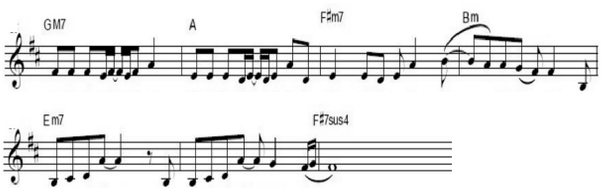

Fig. 1. Sample Music Score

둘째, 멜로디 라인의 변곡점을 선별하여 사용한다. 이는 SOC-IM 알고리즘의 가장 큰 특징이다. SOC-IM 알고리즘은 멜로디 라인에서 변곡점 시퀀스를 찾아내어 사용하기 때문에 곡의 조성, 빠르기 등 다양한 변화에도 곡의 특징을 이해할 수 있어 검색 성능이 보장된다. 또한 곡에 대한 정보를 일정한 간격으로 저장하는 것이 아니라, 변곡점을 사용하여 같은 음이 길게 이어지는 부분이나 변화의 정도가 일정한 부분은 생략하고, 그 시작이나 끝부분만 메모리에 저장하므로, 저장공간을 최소화한다.

멜로디 라인과 이로부터 생성된 변곡점 시퀀스의 관계는 다음과 같다. Fig. 2는 Fig. 1의 악보 데이터를 시각화한 것으로 가로축은 타임 스탬프, 세로축은 멜로디의 미디 노트 넘버를 나타내며, 동그라미로 표시한 부분들은 변곡점들이다. 이는 실제 데이터 입력과는 다르며, 본 논문에서 변곡점이 뜻하는 바와 SOC-IM 알고리즘의 목적을 설명하기 위하여 악보에서 음정이 변하는 부분의 데이터만 표기하고 선으로 이어 이해하기 쉽도록 표현한 것이다. SOC-IM 알고리즘은 변곡점 시퀀스를 추출하여 사용한다. 이와 같이 변곡점을 활용하면 각 변곡점의 앞뒤 기울기와 해당 변곡점과 앞 또는 뒤에 있는 변곡점과의 거리 관계를 고려할 수 있어서 메모리에 저장하지 않고 생략된 정보들까지도 유추할 수 있다. 이는 저장공간을 최소화하면서도 시퀀스의 특징을 놓치지 않는 방법이다.

변곡점 시퀀스를 사용하면 곡의 속도 변화에 유연하게 대처할 수 있다. 그 이유는 변곡점 시퀀스는 멜로디의 상대적 변화에 초점을 맞춰 노래 정보를 표현하기 때문이다. 예를 들어 한 곡과 그 곡을 0.5 배속한 곡이 있을 때, 만약 둘을 타임스탬프별로 프레임을 나누어 표현한다면 두 곡의 유사도는 낮을 것이다. 그러나 두 노래의 변곡점 시퀀스는 거의 일치하므로 두 곡의 유사도는 높게 나온다.

셋째, 원곡은 대표성을 갖는 구절만 사용한다. 원곡을 커버하는 과정에서는 전체적인 조성이나 속도뿐 아니라 아예 곡의 구성 자체가 바뀌는 경우가 있다. 이 경우 원곡과 커버곡

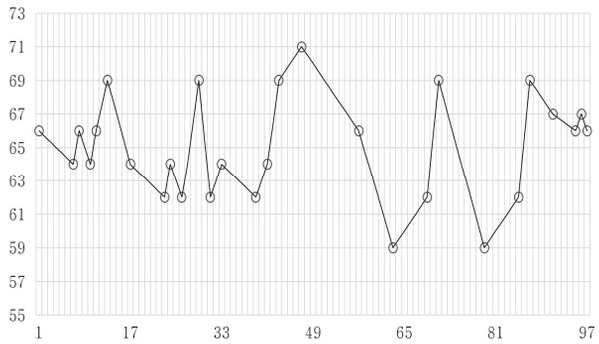

Fig. 2. Inflection Points of the Melody Line

의 유사도를 측정하면 당연히 낮게 나올 수밖에 없다. 이런 점을 개선하기 위해 본 알고리즘에서는 원곡은 그 곡을 대표할 수 있는 구절만 사용한다. 사람들이 도입부나 후렴부 한두 마디 정보만 듣고도 어떤 곡인지 알아차리는 것은 그 구절이 해당 곡의 대표성을 갖는 구절이기 때문이다. 이에 본 연구에서는 곡을 커버할 때는 대표적인 구절은 반드시 들어갈 것이라고 가정하여 계산의 편의와 정확도 향상을 위해 원곡은 대표성을 갖는 구절만 사용하고, 커버곡에 해당 구절이 포함되어 있는지 여부를 판별하는 것으로 알고리즘을 진행한다. 대표성을 갖는 구절을 얻는 방식은 다음 절에서 자세히 기술하였다.

SOC-IM 알고리즘은 다음 사항들을 가정한다. 첫째, 원곡의 대표 구절의 변곡점 정보는 미리 계산되어 있다고 본다. 저작권 보호를 받기 위해 원곡 저작권자가 사전에 등록한다고 가정한다. 반복되는 구절이나 머신러닝 등을 이용하여 음원의 후렴구를 추출하는 것에 관한 연구가 활발히 진행 중이므로, 해당 알고리즘이 발전한다면 원곡의 대표 구절을 자동으로 추출할 수도 있고, 원곡의 대표 구절에서 변곡점 정보를 추출하는 것은 본 연구의 변곡점 추출 과정을 따라 진행하면 된다[6]. 둘째, 제공되는 커버곡 데이터는 커버곡의 멜로디를 미디 노트 넘버로 변환한 정보이다. 여기서 미디 노트 넘버란 비선형 그래프를 이루는 음정 당 주파수를 일정한 수식에 따라 선형으로 변환한 것이다. 현재 MELODIA 등 곡의 멜로디 주파수 추출에 관한 연구 또한 활발히 진행 중이고, 이러한 알고리즘으로 커버곡 음정의 미디 노트 넘버로의 변환도 충분히 대체될 수 있다고 본다[7,8]. 이러한 입력값에 대해 SOC-IM 알고리즘은 변곡점들을 찾아내고, 변곡점의 특징값으로 각 노래와의 유사도를 계산해서 주어진 커버곡의 원곡 번호를 반환한다. 알고리즘은 크게 변곡점 정보 추출과 커버곡의 원곡 검색으로 구성되며, 전체적인 흐름은 Fig. 3과 같다.

SOC-IM 알고리즘은 4단계로 구성된다. [단계 1]에서는 입력된 데이터에 대하여 데이터의 변화 양상이 바뀌는 변곡점 부분을 찾아낸다. [단계 2]에서는 변곡점에 대하여 전후 기울기 변화 양상(이하 타입)과 앞/뒤 변곡점과의 거리 비율(이하 세부 비율)을 이용해 변곡점 정보를 추출한다. [단계 3]에서 주어진 커버곡의 변곡점 정보를 기존에 가지고 있는 노래들의 변곡점 정보와 각각 비교하여 두 변곡점 시퀀스 간의

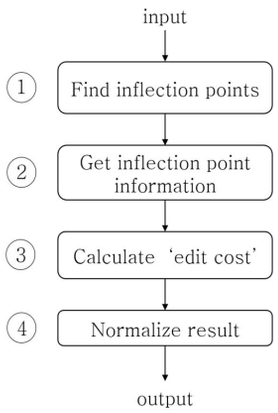

Fig. 3. Flowchart of the Algorithm

유사도를 계산한다. 이때 유사도가 높다는 것은 편집 비용 값이 작다는 것을 의미한다. 여기서 편집 비용이란, SOC-IM 알고리즘에서 시퀀스의 유사도를 측정하기 위해 사용하는 값으로 편집 거리 알고리즘을 변형한 편집 비용 알고리즘으로 계산한다. 편집 비용 알고리즘은 값을 계산할 때 데이터의 전체적인 흐름을 먼저 반영하고 리듬감도 반영하는 방식으로 가중치를 둔다. 편집 비용 계산 알고리즘은 SOC-IM 알고리즘을 4장에서 구체적으로 설명할 때 함께 설명하기로 한다. 마지막으로 원곡의 대표 구절의 길이에 따라 편집 비용이 달라지기 때문에 이를 대표 구절의 길이로 나눠 값을 정규화시키는 [단계 4]를 거친 후, 가장 낮은 값을 갖는 곡의 번호를 반환하며, 이는 커버곡의 원곡의 번호이다.

4. 원곡 검색 알고리즘

여기에서는 본 연구에서 제안하는 원곡 검색 알고리즘 2개를 소개한다. SOC-IM-naive 알고리즘은 변곡점 선별 기준, 변곡점 정보 추출, 유사도 계산 방식에서 상식적인 아이디어를 사용한다. SOC-IM 알고리즘은 SOC-IM-naive 알고리즘에 비해 변곡점 선별 과정과 유사도 계산 측면에서 개선된 방법을 사용한다.

4.1 SOC-IM-naive 알고리즘

변곡점은 음정과 음정의 기울기가 모두 바뀌는 부분을 뜻한다. 음정이 같은 값으로 유지되거나 기울기에 변화가 없는 부분은 변곡점에서 제외된다. 이렇게 선별된 변곡점들에 대하여 변곡점 정보를 추출한다. 이때 추출하는 변곡점 정보에는 타입과 세부 비율이 있다. 타입은 변곡점에 대하여 전후 기울기 변화 양상을 뜻하며, 기울기의 양수, 음수 여부를 문자열 p와 m으로 표현한다. 예를 들어, 변곡점을 기준으로 기울기가 양수에서 음수로 바뀌면 해당 변곡점의 타입은 pm이다. 세부 비율은 앞/뒤 변곡점과의 거리 비율로, 1 이하의 실수로 표현한다. 세부 비율은 해당 변곡점을 기준으로 앞 변곡점부터 뒤 변곡점까지의 전체 길이를 앞 변곡점부터 해당 변

곡점까지의 길이로 나눠 계산한다. 전체 리듬 속에서 해당 음의 리듬을 상대적으로 측정하는 것이다. 이로써 리듬을 절대적인 기준이 아니라 곡에 맞게 상대적으로 반영할 수 있다.

그 다음에는 추출한 변곡점 정보 시퀀스를 사용하여 커버곡 C의 원곡 M을 검색한다. (원곡 가능성이 있는) 노래들 $m_i,\ 1 \le i \le n$, 는 이미 데이터베이스에 저장되어 있다. 원곡 M을 찾기 위해 커버곡 C와 이들 각 노래 m_i 사이의 편집 비용을 계산한다. 편집 비용이 가장 작은 노래가 커버곡 C의 원곡 M이라고 할 수 있다. 커버곡 C와 노래 m_i 사이의 편집 비용을 계산할 때, m_i의 대표 구절 m_{iR}이 커버곡 C에 포함되어 있는지를 점검한다. 이를 위해 커버곡 C를 타임 스탬프 하나씩 줄여가는 방식으로 슬라이딩하면서 편집 비용을 계산한다. 이렇게 계산된 모든 값 중에서 가장 작은 값이 m_i와 C사이의 편집 비용이다. 두 곡 m_i, m_j에 대해, m_i의 대표 구절인 m_{iR}이 m_j의 대표 구절인 m_{jR}에 비해 긴 경우에는 m_i와 C사이의 편집 비용이 m_j와 C 사이의 편집 비용에 비해 상대적으로 크기 때문에, 계산된 편집 비용은 해당 곡의 구절의 길이로 나누어 정규화를 시킨 후 사용한다.

여기에서 사용되는 편집 비용 계산 알고리즘은 두 시퀀스 사이의 유사도를 측정하기 위하여, 기존의 편집 거리 알고리즘을 발전시킨 것이다. 편집 비용 알고리즘의 주요 특징으로 다음 두 가지를 들 수 있다. 첫째, 값을 계산할 때 데이터의 전체적인 흐름인 기울기 변화 양상을 먼저 반영하고, 기울기 변화 양상이 같다면 리듬감도 반영하는 방식으로 가중치를 둔다. 리듬감은 두 변곡점의 타입이 다를 때는 편집 거리 알고리즘의 방법을 그대로 사용하고, 두 변곡점의 타입이 같을 때는 세부 비율도 수식에 더한다. 둘째, 원곡 대상이 되는 곡의 대표 구절의 길이를 기준으로 커버곡과의 편집 비용을 모두 계산한 후 그중에서 가장 작은 값을 결과값으로 취하는 것이다. 이는 편집 비용 알고리즘의 목적이 커버곡에 원곡 대표 구절과 유사한 구절이 있는지를 찾고 그 유사도를 측정하는 것이기 때문이다.

4.2 SOC-IM 알고리즘

앞서 소개한 SOC-IM-naive 알고리즘은 변곡점을 선별하는 부분에 있어서 두 가지 한계점을 가진다. 첫째, 같은 값이 여러 프레임에 거쳐 지속되면 그 시작과 끝부분을 모두 변곡점으로 인식하기 때문에 실제 곡을 들을 때 인식되는 변곡점과는 괴리가 생긴다. 둘째, 값이 일정한 정도로 증가하거나 감소할 때, 프레임 1개당 값이 변화하면 원래 의도대로 변곡점에 포함되지 않지만, Fig. 4의 사각형 부분과 같이 같은 값이 2개 이상의 프레임에 지속되면 값이 일정한 정도로 증가하거나 감소하더라도 값이 바뀔 때마다 모두 변곡점에 포함된다.

이를 해결하기 위해 SOC-IM 알고리즘은 변곡점 선별을 두 단계에 걸쳐 진행한다. Fig. 5는 SOC-IM의 변곡점 선별 알고리즘이다. 알고리즘에서 data 배열은 입력되는 미디 노트 넘버 데이터이고, bp 배열은 선별된 변곡점을 가리킨다. 1~11행에서 일차적으로 값이 변하는 시작 부분이면서 전후로 차이가 같지 않은, 즉 기울기 변화가 있는 부분을 변곡점 후보로 둔다. 이 과정에서 Fig. 4의 파란색 X표로 표시된 부분과 같이 값이 시작하는 부분이 아닌 것, 즉 음정이 유지되는 부분이 변곡점 후보에서 3~4행의 코드를 통해 제외된다. 일차 선별 후 12~20행에서 변곡점 후보들에 대해 이차적으로 해당 변곡점 후보 전후로 기울기가 일정하지 않을 때만 해당 변곡점 후보를 변곡점으로 지정한다. 13, 14행에서는 해당 변곡점과 그 앞/뒤의 변곡점 사이의 기울기를 구한다. 이후 15, 16행에서 해당 변곡점을 기준으로 양쪽의 기울기가 같다면 이는 값이 일정한 정도로 변화하는 것이기 때문에 변곡점에서 제외한다. 이 과정에서 Fig. 4의 빨간색 X표로 표시된 부분이 변곡점에서 제외된다. 이와 같은 과정을 통해 앞서 언급한 변곡점 선별에서의 두 가지 한계를 모두 해결할 수 있다.

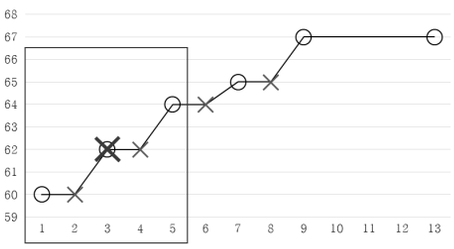

Fig. 4. SOC-IM's Inflection Point Selection Process

SOC-IM-naive 알고리즘은 유사도 계산에서도 주어진 두 변곡점의 타입이 같은지 다른지 여부만 판별할 뿐, 그 정도는 반영하지 못한다는 한계를 보인다. 따라서 변곡점 타입이 다르면 일률적으로 앞선 값들의 최솟값에 같은 값을 더하여 사용한 SOC-IM-naive 알고리즘과는 달리 SOC-IM에서는 앞선 값들의 최솟값에 타입이 다른 정도를 가중치로 더하는 방법을 사용하여 이와 같은 문제를 해결한다.

Fig. 6은 SOC-IM 알고리즘의 편집 비용 계산 알고리즘이다. 변수 d는 계산하는 편집 비용을 저장하는 행렬이다. 두 변곡점의 타입이 같을 때는 13행과 같이 세부 비율도 수식에 반영해 계산하고, 타입이 다를 때는 15행과 같이 타입이 다른 정도에 따라 가중치를 다르게 부여한다. 예를 들어 전후로 기울기가 양성에서 음성으로 변하는 변곡점과 양성에서 양성으로 변하는 변곡점의 가중치는 1, 기울기가 양성에서 음성으로 변하는 변곡점과 음성에서 양성으로 변하는 변곡점의 가중치를 2로 두는 식이다. 그리고 원곡 대상이 되는 노래 m_i의 대표 구절 m_{iR}과 커버곡의 길이가 다르기 때문에 19행과 같이 m_{iR}을 기준으로 하여 그중에서 가장 작은 값을 편집 비용 계산 알고리즘의 결과값인 편집 비용값으로 정한다.

5. 원곡 검색 알고리즘의 성능평가

제안한 원곡 검색 알고리즘 2개의 성능평가 과정과 분석 결과를 설명하기로 한다. 성능평가를 하기 위해 원곡 대상 노래 72곡과 커버곡 23개를 사용하였다. 각 곡은 악보를 사용

```
Procedure dev_select_inflection_points(data) :
1.  bp.append([0, data[0]])
2.  for i in (1, len(data)-1) do
3.      if data[i]-data[i-1] == 0 do
4.          continue
5.      else if (data[i]-data[i-1])==(data[i+1]-data[i]) do
6.          continue
7.      else if abs(data[i]-data[i-1]) >= 1 do
8.          bp.append([i, data[i])
9.      end if
10. end for
11. bp.append([len(data)-1, data[len(data)-1]])

12. for i in range(1, len(bp)-1) do
13.     m=(float)(bp[i][1]-bp[i-1][1])/(bp[i][0]-bp[i-1][0])
14.     n=(float)(bp[i+1][1]-bp[i][1])/bp[i+1][0]-bp[i][0])
15.     if m == n do
16.         continue
17.     else do
18.         bp.append(information of the point)
19.     end if
20. end for

21. for i in range(0, len(bp)) do
22.     cls.append(bp_cls(bp, i))
23. end for
24. Return cls
```

Fig. 5. SOC-IM's Inflection Point Sequence Selection Algorithm

하여 4분음표를 기준으로 한 박자당 4개의 미디 노트 넘버를 추출하여 사용하였다. 원곡 대상 곡들의 대표 구절은 후렴구나 도입부 중에서 대표성이 더 큰 것으로 선택하였고, 커버곡은 원곡 데이터에 포함된 곡을 리메이크한 곡들로 선정하였다.

원곡과 커버곡 데이터는 원곡과 커버곡 사이의 차이가 비교적 큰 것들로 선별함으로써 SOC-IM 알고리즘의 성능을 다방면으로 평가할 수 있도록 하였다. 만약 원곡과 커버곡 사이의 변화가 적은 곡들을 실험데이터로 사용하였다면 두 데이터가 비슷해서 검색 정확도는 더 높게 나왔을 것이다. 그러나 본 연구에서 성능 평가에 사용된 커버곡 데이터들은 조성, 속도, 구성 등 커버곡에서 나타날 수 있는 모든 변화를 포함하도록 구성하였다. 이는 곧 검색이 어려운 여러 상황에서의 검색 성능을 평가한 것이기 때문에 정확도에 대한 신뢰도가 높다고 할 수 있다. 대부분의 곡이 1절과 2절이 반복되기 때문에 연구의 편의를 위해 커버곡은 1절만큼의 길이에 대해서만 데이터를 추출하였다.

본 알고리즘의 목적이 커버곡이 주어졌을 때 그의 원곡을 찾는 것이기 때문에 성능을 평가할 때도 커버곡에 대하여 원곡을 얼마나 맞혔느냐로 평가하는 게 더 목적성에 부합하기는 하지만, 보다 세부적인 분석을 위해 정밀도와 재현율, 그리고 둘의 조화 평균을 이용한 F1 score도 살펴본다. 다만 정확도와는 다르게 다른 척도들은 커버곡을 기준으로는 측정이 어렵기 때문에 원곡을 기준으로 측정한다. 해당 커버곡의 원곡을 찾아내고 이를 제대로 맞췄는지를 평가하는 방식의 모델에서 원곡에 대하여 커버곡들이 해당 원곡의 커버곡인지 아닌지를 분류하는 모델로 시점을 바꿔 성능을 평가하는 것이다.

Fig. 7은 SOC-IM-naive 알고리즘과 SOC-IM 알고리즘에 대하여 각각 커버곡을 기준으로 한 정확도와 원곡을 기준

```
Procedure dev_edit_cost(origin, cover_tmp) :
1.  m = shape(origin feature value matrix)
2.  n = shape(cover_tmp feature value matrix)
3.  d = [[0 for i in range(n+1)] for j in range(m+1)]
4.  for i in range(m+1) do
5.    d[i][0] = i
6.  end for
7.  for j in range(n+1) do
8.    d[0][j] = j
9.  end for
10. for i in range(1, m+1) do
11.     for j in range(1, n+1) do
12.         if 'type' is same do
13.           d[i][j] = min(d[i-1][j]+1, d[i][j-1]+1, d[i-1][j-1])
                    + abs('detail ratio' of cover_tmp[j-1]
                    - 'detail ratio' of origin[i-1])
14.         else do
15.           d[i][j] = min(d[i-1][j], d[i][j-1], d[i-1][j-1]) +
                    'degree of difference between 'type''
16.         end if
17.     end for
18. end for
19. Return min(d[m-1])
```

Fig. 6. SOC-IM's Edit Cost Calculation Algorithm

으로 한 정밀도와 재현율, F1 score를 정리한 것이다. 이때 원곡을 기준으로 정확도를 측정하면 분자 대비 분모가 확연히 크기 때문에 정확도가 모두 높게 나와 알고리즘의 성능을 평가하기에는 적절하지 않다고 판단해 정확도는 커버곡을 기준으로만 측정하였다.

모든 척도에 대해 SOC-IM-naive 알고리즘보다 SOC-IM 알고리즘이 전체적으로 높은 값을 가지며, 특히 커버곡을 기준으로 한 정확도와 원곡을 기준으로 한 재현율은 값이 크게 향상되었다. 이를 통해 수정된 변곡점 선정과 유사도 측정 방법이 알고리즘의 성능에 유의미한 발전을 가져왔음을 알 수 있다. SOC-IM 알고리즘에서 원곡을 제대로 찾아내지 못한 커버곡들은 원곡에 비해 커버곡에서 구절 사이나 끝부분에 애드리브가 섞여 있기 때문으로 추측된다. 애드리브로 인해 원곡과 커버곡의 멜로디 라인이 크게 달라져 변곡점 정보에도 영향을 미친 것이다.

6. 결론 및 향후 연구

본 논문에서는 곡의 멜로디 라인에서 변곡점을 선별하고, 선별된 변곡점들의 정보를 추출해 음악의 유사도를 계산하여 효율적으로 커버곡의 원곡을 검색하는 SOC-IM 알고리즘을 제안하였다. SOC-IM 알고리즘은 곡의 멜로디 라인에서 특징적인 부분인 변곡점을 활용하기 때문에 적은 메모리로도 원곡을 커버하는 과정에서 생기는 전체적인 빠르기 변화, 조옮김으로 인한 멜로디의 높낮이 변화, 곡의 구성의 변화를 감지하여 높은 검색 성능을 보인다. 또한, 원곡 데이터는 대표 구절만을 사용하기 때문에 곡의 구성이 바뀌어도 검색이 용이하다. 마지막으로, 곡 전체에 대해 고정된 프레임을 사용하는 기존 방식들과 다르게 대표 구절에 대한 변곡점 정보, 즉 가변적인 프레임을 사용하기 때문에 한 곡 안에서 빠르기가

Algorithm ver.	Accuracy	Precision	Recall	F1 score
SOC-IM-naive	0.783	0.906	0.75	0.978
SOC-IM	0.913	0.921	0.9	0.981

Fig. 7. Performance Evaluation of SOC-IM-naive and SOC-IM

달라지는 경우에도 검색 성능을 유지할 수 있다.

SOC-IM 알고리즘은 성능 평가 결과 정확도 91.3%의 높은 성능을 보였으나, 애드리브 등으로 인한 변주에는 성능이 다소 감소하는 경향을 보이기 때문에 이를 개선하기 위한 향후 연구가 필요하다. 애드리브는 보통 한 음을 길게 끌고 가면서 거기서 변주를 주는 것이기 때문에 음성 인식을 통해 한 음절에는 한 음정만 추출하는 등의 방법을 사용하면 본 문제를 개선할 수 있을 것으로 보인다. 또한, 변곡점 전후의 기울기 변화를 증가와 감소만이 아니라 수치적으로 표현해 가중치를 두어 계산한다면 더욱 정확한 성능을 가질 수 있을 것으로 보여 이와 관련한 향후 연구가 필요하다.

멜로디 라인의 변곡점을 활용하는 본 연구의 주요 아이디어는 커버곡의 원곡 검색뿐만 아니라 허밍으로 노래 검색[9], 예능을 포함한 방송 콘텐츠에서의 음악 저작권 보호[10], 음악 표절 감지[11] 등 다양한 문제 해결에 활용도가 높다고 판단된다. 또한, 음악 도메인이 아니더라도 범위를 넓혀 시퀀스 유사도 측정이 필요한 여러 분야에서도 활용이 가능할 것으로 본다[12].

References

[1] J. Seo, "Improving Cover Song Search Accuracy by Extracting Salient Chromagram Components," *Journal of Korea Multimedia Society*, Vol.22, No.6, pp.639-645, 2019.

[2] A. Ghias, J. Logan, D. Chamberlin, and B. C. Smith, "Query by humming: musical information retrieval in an audio database," in *Proceedings of the third ACM international conference on Multimedia (MULTIMEDIA '95)*, Association for Computing Machinery, New York, NY, USA, pp.231-236, 1995.

[3] J. Jee and H. Oh, "Design and Implementation of Music Information Retrieval System," *The Transactions of the Korea Information Processing Society*, Vol.5, No.1, pp.1-11, 1998.

[4] D. P. W. Ellis, and G. E. Poliner, "Identifying 'Cover Songs' with Chroma Features and Dynamic Programming Beat Tracking," *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, Honolulu, HI, pp.IV-1429-IV-1432, 2007.

[5] "Cultural Technology (CT) In-depth Report: Music information retrieval technology trend," Korea Creative Content Agency, No.4, pp.1-19, 2012.

[6] J. Choi, J. Yoon, G. Kim, J. Ahn, and J. Jung, "Extraction of Music Highlight via Improved Similarity Anaylysis," *Proceedings of Korean Institute of Information Scientists and Engineers*, pp.1818-1820, 2018.

[7] S. Geum and J. Nam, "Music information retrieval examined by melody extraction algorithm," *The Magazine of the IEIE*, Vol.43, No.5, pp.41-49, 2016.

[8] J. Salamon and E. Gomez, "Melody Extraction from Polyphonic Music Signals using Pitch Contour Characteristics," *IEEE Transactions on Audio, Speech and Language Processing*, Vol.20, No.6, pp.1759-1770, 2012.

[9] D. Oh and H. Oh, "A Similarity Computation Algorithm for Music Retrieval System Based on Query By Humming," *Journal of The Korea Society of Computer and Information*, Vol.11, No.4, pp.137-145, 2006.

[10] W. Heo, B. Jang, H. Jo, J. Kim, and O. Kwon, "Performance of music section detection in broadcast drama contents using independent component analysis and deep neural networks," *Journal of the Korean Society of Speech Sciences*, Vol.10, No.3, pp.19-29, 2018.

[11] J. Park and S. Kim, "Development of a System for Music Plagiarism Detection Using Melody Databases," *Journal of Korea Multimedia Society*, Vol.8, No.1, pp.1-8, 2005.

[12] Y. Fan, Y. Shi, K. Kang, and Q. Xing, "An Inflection Point Based Clustering Method for Sequence Data," In *Web Information Systems and Applications. WISA 2019*. Lecture Notes in Computer Science, Vol.11817. Springer, Cham, 2019.

이 보 현

https://orcid.org/0000-0002-5361-188X
e-mail : leeboh23@ewhain.net
2019년 이화여자대학교 컴퓨터공학과(학사)
2021년 이화여자대학교 인공지능·소프트웨어학부 석사과정
관심분야 : 검색 알고리즘, 추천 알고리즘, 빅데이터 처리, 머신러닝

김 명

https://orcid.org/0000-0003-4981-6345
e-mail : mkim@ewha.ac.kr
1981년 이화여자대학교 수학과(학사)
1983년 서울대학교 계산통계학과(석사)
1990년 미네소타대학교 컴퓨터공학과 (석사)
1993년 캘리포니아주립대학교, 산타바바라(UCSB) 컴퓨터공학과(박사, 박사후 과정)
1995년 ~ 현 재 이화여자대학교 컴퓨터공학과 교수
관심분야 : 고성능컴퓨팅, 추천시스템, 정보 검색, 빅데이터 분석