계층적 시간 메모리(HTM) 기반의
멜로디 기억 및 생성 기법 연구

(HTM-Based Methods for Melody Recall and Generation)

지도교수 : 장병탁

이 논문을 공학학사 학위 논문으로 제출함.

2011 년 12 월 22 일

서울대학교 공과대학

컴 퓨 터 공 학 부

이 성 민

2012 년 2 월

계층적 시간 메모리(HTM) 기반의 멜로디 기억 및 생성 기법 연구

2008-11686 이성민

서울대학교 컴퓨터공학부

초록

사람의 경우 익숙한 멜로디에 대해서 부분적으로 따라 할 수 있으며 예측할 수 있다. 사람의 이러한 능력을 알고리즘을 사용하여 따라 하려는 연구들이 진행되어 왔다. 본 연구에서도 HTM을 이용하여 알고리즘 작곡에 대한 연구를 진행해보았다. 본 연구에서는 시간적 공간적인 패턴을 처리하는 새로운 기계학습 알고리즘인 Hierarchical Temporal Memory를 사용하여 멜로디를 학습하고 처리하는 모델을 제시한다. 본 연구에서는 음악의 학습 및 추론을 벡터의 순서가 의미가 있는 Classification 문제로 재정의하여 문제를 풀어보았다. 문제를 푸는 방식은 멜로디의 높이와 길이에 대한 양자화된 값을 사용하여 HTM모델에게 이를 학습시키고 학습시킨 모델에게 일부 멜로디를 질의로 주어 멜로디에 대한 순차적인 예측을 반복하여 음악을 생성하도록 하였다. 이 때 잘 학습되고 구성한 HTM네트워크에 대하여 학습된 데이터 셋에서 90 퍼센트를 넘는 예측률을 보여주었다. 이를 통해 HTM이 시간적인 데이터를 처리할 수 있으며 알고리즘 작곡에 사용될 수 있다는 점을 보였다. 본 연구에서는 음의 높이와 길이에 대한 값으로 절대적인 값을 사용하였다. 이는 조성에 독립적이지 못한 모델을 생성하므로 향후 음의 상대적인 변화를 가지고 구성한 모델을 사용하면 좀 더 높은 예측률을 보일 것이라고 생각한다.

I. 서론

음악은 인류의 문화가 시작되면서부터 함께한 가장 오래된 형태의 예술이다. 음악의 역사가 오래되었음에도 불구하고 지금까지도 음악을 만드는 작업은 재능이 있는 특별한 사람이 특수한 영감을 가져야만 가능하다고 여겨진다. 음악을 만드는 작업은 이렇게 기계뿐만 아니라 사람도 어려워하는 작업이지만 음악에도 시대별, 장르별에 따라 공통된 일련의 규칙들과 패턴이 있기 때문에 이 패턴을 이용하여 인공지능분야뿐만 아니라 여러 분야에서 알고리즘을 사용한 작곡을 하려는 시도들이 있어왔다.

인공지능 분야에서는 알고리즘을 이용한 작곡에 대한 연구들은 꾸준히 이루어져 왔다. 규칙 기반 시스템 기반의 알고리즘 작곡과는 달리 기계학습 기반의 알고리즘들은 음악에 대한 전문가가 정의한 규칙들의 도움을 거의 받지 않고서 기계들이 전달받은 음악 데이터를 분석하여 이를 학습하고 학습한 모델을 기반으로 음악을 기억해 내거나 작성하는 접근방식을 취한다. 이러한 접근 방식은 다른 알고리즘 작곡과는 달리 학습시킨 데이터의 종류와 양에 따라 생성된 서로 다른 모델들을 형성할 수 있다. 이를 사용하면 특정 작곡가나 장르와 비슷한 스타일의 곡을 작성하는 것이 가능하며 이론적으로는 음악 전반에 관한 많은 양의 데이터셋을 구축하여 학습시키는 경우 음악에 대한 일반화된 패턴을 사용하여 작곡을 하는 모델을 작성하는 것이 가능하다. 컴퓨터와 인터넷의 발달로 많은 양의 데이터가 인터넷에 쌓여가고 있으며 컴퓨터가 이를 처리하는 속도가 향상되면서 기계학습 및 데이타 마이닝 분야는 많은 발전을 보이고 있고 여러 분야에서 상용적으로 사용할 수 있는 응용 프로그램들이 개발되고 있다. 음악에 대해서도 디지털 형태로 변환된 많은 음악데이터를 누구나 손쉽게 접근할 수 있게 되었기 때문에 음악에 대한 기계학습 또한 가능하게 되었고 발전을 보이고 있다.

음악데이터는 음악을 문자 형식으로 해석해 놓은 Symbolic한 음악 데이터와 신호형태로 되어있는 음악데이터가 있다. Symbolic한 음악 데이터는 악보나 미디 파일 등이 존재하며 신호형식의 음악 데이터는 웨이브(wave)파일이나, mp3 파일 등이 있다. Symbolic한 음악 데이터는 실제 신호형식의 음악을 음표의 흐름으로 한 단계 추상화 되어있는 데이터로 볼 수 있으며 본 연구에서는 실험의 편의와 학습되는 시스템이 음악의 추상화된 패턴을 좀 더 잘 학습하는 모델을 구성하기 위하여 Symbolic한 음악 데이터를 데이터 셋으로 사용한다.

본 연구에서 다루는 문제는 다음과 같다. symbolic한 형태의 음악 데이터들이 저장되어 있는 다수의 corpora가 있는 상황에서 각 corpus를 기계학습 기반의 모델에게 학습시켜 학습된 네트워크를 생성한다. 이렇게 생성된 네트워크 모델에게 음악데이터 일부의 cue를 input으로 주면 각 모델들은 sequential prediction방식으로 멜로디를 생성한다. 이런 식으로 형성된 음악은 사람들이 직접 감상할 수 있는 음악을 작곡하거나 실제적으로 음악을 작곡하는데 도움을 주는 모델을 만들기 보다는 주어진 데이터의 스타일을 따르는 곡을 작성하며 새로운 음악을 질의로 받았을 때 음악의 작성자를 찾는 종류의 일을 할 수 있을 것이다. 또한 음악을 생성하는데 있어서 확률적으로 음악을 생성하는 부분을 추가하고 생성된 음이 사람들이 듣기에 어색할 경우 이를 보정하는 등의

알고리즘을 추가한다면 실제로 사람이 감상할 수 있는 음악을 생성하는데 큰 무리가 없을 것으로 생각된다.

본 연구에서는 HTM(Hierarchical temporal memory) 모델을 사용하여 음악의 작성과 추론 문제를 푼다. HTM은 공간적인 패턴과 시간적인 패턴이 합쳐진 동영상과 같은 데이터들이 실제 뇌의 신피질에서 처리되는 방식을 연구하여 시공간적인 패턴을 학습하고 기억해 낼 수 있도록 신피질을 모델링한 알고리즘이다. HTM은 알고리즘 자체가 내제적으로 기억과 추론을 할 때 temporal한 패턴을 처리하도록 만들어졌기 때문에 하이퍼네트워크[1]와 같은 다른 종류의 기계학습 모델을 사용한 시스템에 비해서 시간적인 음악 데이터를 기억하고 처리하는데 개선이 있을 것으로 기대한다.

본 논문의 구성은 다음과 같다. 2 장에서는 알고리즘 작곡 및 HTM에 대한 배경이론을 소개한다. 3 장에서는 Sequential Prediction방법과 HTM을 사용하여 실제적으로 음악을 학습하고 생성하는 알고리즘을 소개한다. 4 장에서는 구현한 시스템을 사용하여 실험한 결과들을 보여주고 그것들이 갖는 의미에 대하여 소개한다. 5 장은 결론부분으로 실험에 대한 결론 및 향후 더 연구할 수 있는 연구과제들에 대한 내용을 담고 있다.

Ⅱ. 관련연구

A. 알고리즘 작곡

알고리즘 작곡(algorithmic composition)이란 음악을 만들고 합치는 문제에 있어서 특정한 규칙의 sequence를 사용하여 문제를 푸는 것이라고 정의할 수 있다[2]. 알고리즘을 이용한 작곡은 여러 방법이 있을 수 있는데 확률을 사용한 우연성을 이용한 방법, 수학적 모델을 이용하는 방법, 지식 기반 시스템, 문법을 이용한 방법, 진화 알고리즘을 사용하는 방법, 학습 시스템을 사용하는 방법, 하이브리드 시스템을 사용하는 방법 등으로 분류할 수 있다.

확률을 사용한 우연성을 이용한 방법의 알고리즘 작곡 방법 중 가장 오래된 방법은 모차르트(Wolfgang Amadeus Mozart)의 주사위를 사용하여 미뉴에트를 작곡하는 방법이다. 1787 에 모차르트는 음악 작곡 주사위 게임을 만들었는데, 아이디어는 미리 작곡된 음악 마디들이 존재할 때 주사위를 던져서 각 마디를 규칙에 맞게 배열하면 미뉴에트 곡을 작곡 할 수 있다. 이에 대한 예시와 설명은 온라인에 공개되어 있다[3].

수학적 모델을 이용한 방법은 1956 년 미국에서 고안된 자동 작곡기계 '데이터트론(Datatron)' 처럼 학습을 통해 알고리즘 작곡을 하는 시스템이 있다. 수학자 마틴 클라인과 더글라스 보리트가 발전시켜온 것인데 실 수요자들에게는 관심을 얻지 못하였다. 1955-1956 년 르자렌 힐러와 레너드 아이작슨에 의하여 시작되었으며 일리악 컴퓨터를 사용하였다[4].

지식 기반 방법은 인공지능에서 널리 사용되었던 전문가 시스템으로 생각할 수 있다. 규칙과 제약으로 구성된 논리로 구성된 시스템으로 잘 정의된 사전 지식을 이용하여 음악을 보다 사람이 만드는 음악과 유사하게 만드는 방법이다. Ebcioglu이 BSL이라는 언어를 개발하여 그것을 기반으로 CHORAL이라는 규칙기반 시스템을 만들었다[5]. 지식 기반 시스템의 단점은 전문가 시스템이 갖는 공통 문제점인 규칙을 정하는 것이 어렵다는 것에 있다. 지식 기반 시스템을 사용하기 위해서는 지식을 규칙으로 만들어야 하는데 음악과 같은 주관적인 영역의 지식은 특히 더 규칙으로 만들기 어렵다는 점이 있다. 또한 전문가의 지식과 능력에 의존적이며 규칙이 더해질수록 예외를 처리하기 위한 규칙을 추가하면서 규칙의 집합이 매우 커질 수 있다[6].

학습 시스템을 이용하는 방법은 음악에 대한 전문가적인 사전 지식을 최소한으로 줄인 상태에서 음악에 대한 학습 데이터만 사용하여 특성 값을 추출하고 이를 이용해 음악을 생성하는 방법이다. 학습 시스템의 방법 중의 하나로 인공신경망을 들 수 있는데 이를 이용하여 음악을 작곡할 수 있는 많은 응용프로그램들이 개발되었다. Bellgard와 Tsang는 EBM(Effective Boltzmann Machine)을 이용하여 화음을 생성하였다[7]. 본 연구에서 HTM을 사용하여 음악을 만드는 방법도 학습을 이용한 방법이라고 할 수 있다. 학습 시스템을 이용한 방법에는 몇 가지 단점이 있다. 첫째로 충분한 학습데이터가 주어져야 한다. 둘째로 전역적인 일관성을 갖는 모델을 만드는 것이 어렵다. 세 번째로 학습의 일반화와 학습데이터를 잘 학습하는 두 가지 상충적인 목적을 이루기가 어렵다. 만약 아무런 제약 없이 학습데이터를 완벽하게 기억할 수 있다면 이러한 학습모델은 전달받은 데이터에 대한

일반화된 규칙을 학습했다고 볼 수는 없다. 이러한 방식의 학습을 로트 러닝(rote learning)이라고 하는데 이러한 학습을 하였을 때 학습을 한 기계는 학습되지 않은 데이터에서도 적용할 수 있는 일반적인 가설을 학습했다고 보기 어렵다 [6].

B. HTM의 개요

본 논문에서는 음악을 기억하고 생성하는 학습 시스템으로서 HTM(Hierarchical Temporal Memory)를 사용하였다. HTM은 신피질(neocortex)의 구조와 특성을 따르도록 디자인된 기계학습 알고리즘이다. 인간의 신피질은 대뇌의 뉴런으로 구성되어있는 층으로서 대략 1000cm² 넓이와 2mm의 두께를 가지고 있다. HTM은 이름에서도 알 수 있듯이 계층적 구조를 형성하며 알고리즘이 내제적으로 시간적인 데이터를 고려하도록 만들어진 기계학습 기법이다. HTM 네트워크는 시간에 따라 변화되는 많은 양의 데이터를 훈련하며 많은 양의 시퀀스들과 패턴들을 저장하는 네트워크이다. 전통적인 컴퓨터 메모리는 flat organization 구조이며 내제적으로 시간에 대한 개념이 들어가 있지 않다. 하지만 전통적인 컴퓨터 메모리는 어떠한 데이터 구조나 조직도 구현할 수 있다. 대조적으로 HTM 메모리는 제한적이다. HTM 메모리는 계층적 구조를 가지고 있으며 내제적으로 시간에 대한 개념을 가지고 있다. 또한 HTM에 저장되는 정보들은 분산되어있다는 특징을 가지고 있다. 프로그래머는 HTM의 어디에 정보가 저장되고 어떻게 정보가 저장되는지에 대한 조절을 하는 것이 불가능 하다. HTM은 일반적으로 패턴을 저장할 때 Sparse Distributed Representation을 사용하여 데이터를 저장한다.

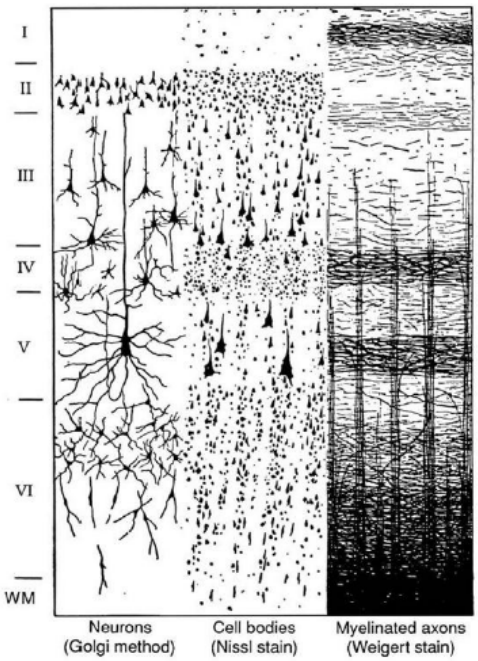


그림 1. 신피질의 계층 구조[8]

다시 말하면 HTM은 크게 두 가지의 특징을 가지고 있다고 볼 수 있다. 첫 번째 특징은 HTM 네트워크가 여러 단계의 층 구조를 가지고 있으며, 이러한 층 구조에서 높은 층은 아래층에서 받은 입력으로 뉴런들이 발화되고 아래층보다 좀 더 추상화된 정보를 나타낸다는 것이다. 두 번째 특징은 HTM이 기둥 구조를 가지고 있으며 이러한 기둥 구조가 패턴을 pooling하는 역할을 한다는 것이다. 이러한 HTM의 HTM의 이 특성들을 음악에 적용할 때 기대되는 효과는 음악을 학습한 HTM 네트워크가 음악을 예측할 때 비슷한 멜로디를 듣고 이전에 학습했던 패턴으로 pooling하여 pattern화 된 음악을 생성할 수 있고 HTM이 음악을 학습을 할 때 음악에 대한 추상화된 모델을 형성할 수 있다는 것과 HTM이 음악을 들을 때 이전에 들었던 음들과 함께 음악의 문맥을 파악할 수 있다는 것이다.

C. HTM 학습 알고리즘

HTM 네트워크는 여러 Region들로 구성된다. 각 Region들은 하위 레벨의 region들과 상위 레벨의 region들과 연결되어 계층적인 HTM네트워크를 이룬다. HTM의 학습과정은 이 Region을 단위로 일어난다. Region들이 전달받는 Input은 수십~수천 길이의 비트이다. 이 입력은 하위레벨의 region들에서 전달 받은 것일 수도 있고 감각기관에서 받은 데이터 수도 있다. Region들은 이 입력들을 통하여 pattern을 학습하고 pattern을 학습한 이후에는 각 region은 예측을 한다. 이 Region이 학습하는 과정은 세 단계로 나눌 수 있다. 먼저 입력으로부터 sparse distributed representation을 구성한다. 두 번째로 이 sparse distributed representation을 이전 단계의 문맥과 함께 고려하여 새로운 representation을 구성한다. 마지막으로 이 representation을 사용하여 예측을 한다. 각 단계에 대해서 좀 더 자세히 설명하면 아래와 같다.

첫 번째 단계인 input으로부터 sparse distributed representation을 구성하는 단계는 다시 말하면 HTM이 입력을 받아서 Spatial 한 패턴을 representation하는 것이라고 생각할 수 있다. HTM의 Region은 여러 뉴런들로 이루어져 있으며 각 뉴런은 일정한 높이의 기둥구조로 이루어져 있다. HTM의 공간적인 패턴 대한 representation은 이 기둥구조의 발화를 기반으로 하여 나타난다. 이 Column구조는 여러 비트의 입력들과 연결되어 있으며 연결된 input들 중 발화된 숫자가 어떠한 threshold를 넘기면 연결된 Column이 발화되게 된다. 이러한 방식의 Column의 발화하는 패턴은 입력이 약간 변화하더라도 input의 발화 패턴보다 더 적게 변하게 된다. 즉 공간적인 패턴에 대하여 입력의 pooling이 가능하게 되며 또한 노이즈에도 강하게 된다. Column들이 발화하게 되면 Column들 중 가장 강하게 발화된 Column이 주위 Column의 발화를 억제한다. 이러한 방법을 사용하면 발화된 Column들의 숫자가 일정한 수준으로 제한할 수 있게 되기 때문에 Sparse Distributed Representation을 가능하게 한다.

두 번째 단계인 시간적인 표현은 공간적인 표현인 기둥 별로 발화된 패턴에서 기둥에 포함된 뉴런들 중 일부의 뉴런들만 발화하는 방식으로 가능하다. 이러한 방식으로 입력 받은 bit가 같더라도 HTM 네트워크 내부의 발화 표현은 달라질 수 있다. 발화된 기둥 중 발화할 뉴런을 선택하는 방법은 기둥에 있는 뉴런들 중 예측 상태에 있는 뉴런들을 선택하여 그 뉴런들만을 발화시킨다. 만약 기둥에 예측상태인 뉴런이 없다면 기둥 전체를 발화시킨다. 이는 예측에 없던 기둥이 발화된 상황이므로 모든 가능한 상황을 고려한다는 뜻으로 해석할 수 있다. 이 과정이 끝나게 되면 뉴런들은 Sparse Distributed Representation한 발화상태가 되어있다. Sparse Distributed Representation으로 패턴을 표현할 시 패턴을 Sub-Sampling할 수 있다는 장점을 가진다. 예를 들어 패턴을 나타내는 정보의 양이 100 이라고 HTM은 그 중 일부인 20~30 만을 네트워크에 기록하고도 충분히 패턴을 기억하고 패턴을 복원할 수 있다. 이러한 Sub-Sampling은 HTM 네트워크가 상황에 따라 유연하게 학습할 수 있게 해준다. HTM 네트워크가 저장해야 할 패턴들의 데이터보다 큰 경우에는 각 패턴들에 대한 자세한 정보를 기억할 수 있어 정확도의 향상을 이끌어 낼 수 있으며 HTM 네트워크가 저장 해야 할 패턴보다 작은 경우에는 각 패턴들을 Sub-Sampling하여 정확도는 약간 떨어지지만 그래도 대부분의 패턴을 기억할 수 있다.

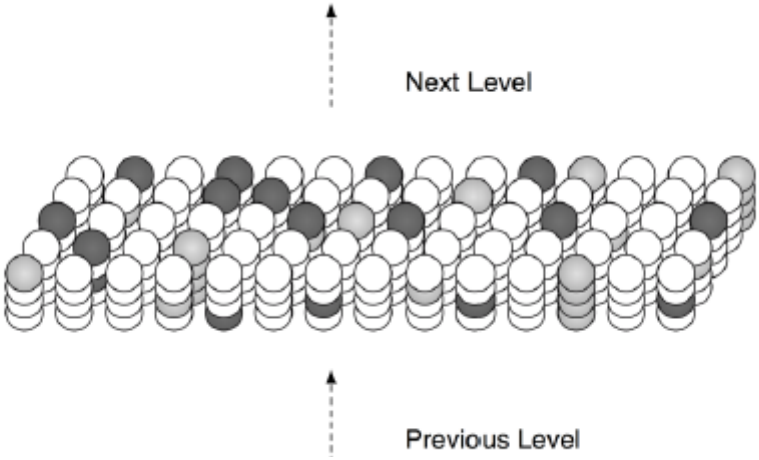


그림 2. HTM region, 옅은 회색은 발화된 뉴런을 나타내며 짙은 회색은 예측상태의 뉴런을 나타낸다.

세 번째 단계인 예측은 시간적인 뉴런의 발화 표현을 형성한 이후에 진행된다. Region의 뉴런은 뉴런 안에서 다른 뉴런들과 수상돌기로 연결되어 있다. HTM에서 예측은 시간적인 발화 표현을 한 이후에 수상돌기가 지나는 뉴런들이 발화하는 수를 세고, 발화한 수가 정해진 Threshold를 넘으면 수상돌기를 가진 뉴런을 예측상태로 넘어가게 한다. 이 뉴런이 예측 상태로 넘어간 이후의 모습은 그림 2 를 보면 볼 수 있다. 그리고 수상돌기에 연결된 뉴런이 예측상태에 넘어가게 되면 수상돌기에 연결된 뉴런들 중 예측상태가 될 수 있도록 기여한 발화해 있던 뉴런들의 Permanence 값을 변경시킨다. 이 Permanence값의 변화를 통해 학습이 일어나는 것이다. Permanence값은 뉴런이 수상돌기에 기여할 수 있는지 없는지를 결정한다. 뉴런의 Permanence값이 일정한 값보다 낮게 내려가면 뉴런은 해당 수상돌기에 대하여 off상태가 되고 발화가 되더라도 해당 수상돌기를 가진 뉴런이 예측하는데 기여를 할 수 없다. 이러한 식으로 학습이 반복되면 수상돌기의 on상태인 뉴런들과 off상태인 뉴런들의 분포가 특정한 상태로 수렴하게 된다. 이 때 각 수상돌기에서 On상태로 연결된 뉴런은 특정한 시간적인 패턴들을 구성하는 뉴런이라고 볼 수 있다. 즉 시간적인 패턴을 학습하는 것으로 볼 수 있다.

III. HTM 기반 멜로디 기억 및 생성

A. 학습 및 추론 알고리즘 개요

본 연구에서 진행하는 음악 멜로디의 학습 및 추론 알고리즘의 개요는 아래의 그림 3 과 같다.

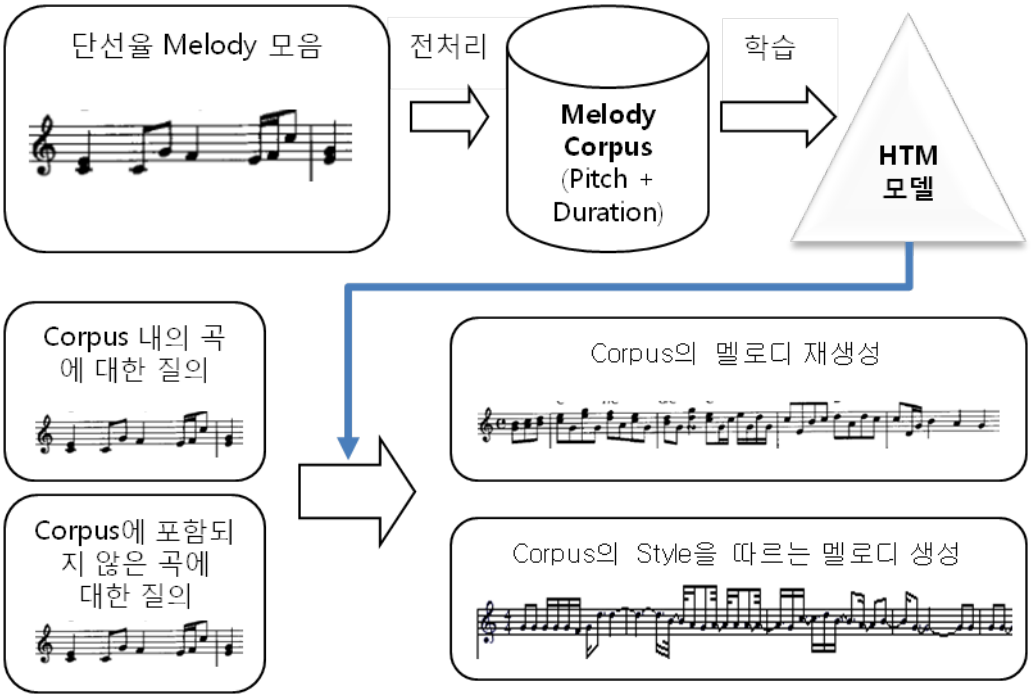


그림 3. 구현한 시스템의 알고리즘 개요. Symbolic한 음악 데이터를 가지고 HTM 예측기를 사용한 음악의 기억 및 생성에 대한 그림

본 연구에서 사용하는 음악 데이터는 단선율의 멜로디로 한정한다. 이 때 단선율 멜로디 데이터는 높이 와 길이의 정보를 가지는 벡터 시퀀스로 생각할 수 있다. 높이와 길이의 정보를 하나의 시퀀스로 처리하거나 또는 높이와 길이 각각을 두 가지의 sequence로 처리하여 이를 합하는 방식으로 처리할 지의 선택에 대한 문제가 있을 수 있다. 정해진 비트의 뼈대 위에 멜로디가 씌워지는 형태로 음악이 구성되어있다는 점을 고려하여 본 연구에서는 높이와 길이를 각각 구분하여 HTM 네트워크에게 학습시키고 이를 통합하여 멜로디를 생성하는 방식으로 문제를 처리하였다.

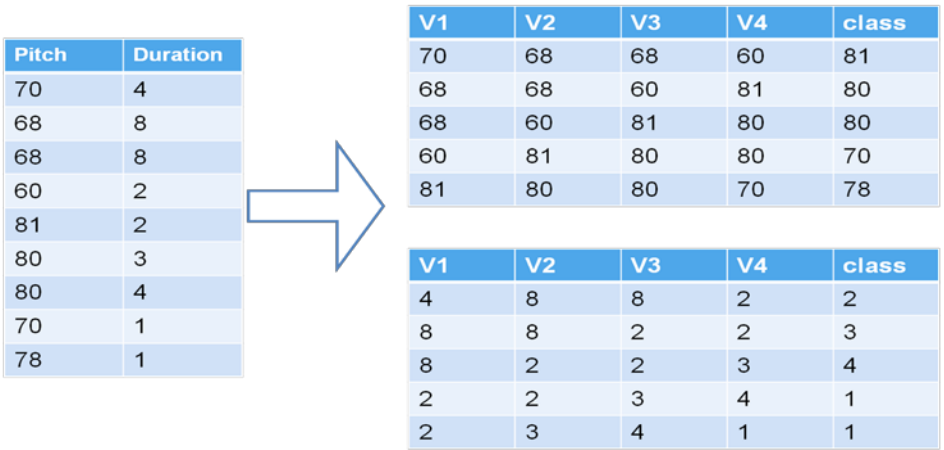
위에서 언급했듯이 이 연구에서 멜로디의 학습 및 추론은 Sequential Prediction의 방식을 이용하여 진행된다. 본 연구의 Sequential Prediction 방법을 간략히 설명하면 다음과 같다. 먼저 prediction의 핵심이 되는 predictor가 있는데 이 predictor는 특정한 벡터가 주어지면 그 벡터를 분석하여 한 원소를 추론하여 결과로 출력하는 역할을 한다. 이 때 결과로 나오는 원소의 차원은 predictor 에게 입력으로 주어진 벡터의 원소들을 이루는 차원과 같은 차원이다. 본 연구에서 predictor에게 입력으로 주어지는 벡터들은 멜로디의 높이와 길이에 대한 벡터이기 때문에 벡터들의 각 원소는

양자화된 높이 또는 길이 값을 나타낸다. 이 때 predictor가 결과로 출력하는 원소의 값은 입력 받은 벡터의 다음 음표에 대한 추론으로서 나온 값이다. 이제 이 값을 predictor가 입력으로 받은 벡터의 마지막 원소로 붙이고 첫 원소를 없애면 새로운 벡터를 생성할 수 있다. 그리고 이 벡터를 사용하여 다시 predictor에게 다음 음을 예측하도록 한다. 이를 계속 반복하면 일정한 멜로디를 얻을 수 있다. Sequential Prediction 방식 멜로디 생성의 중심 부분인 predictor는 HTM을 사용하여 구현하였다.

B. 멜로디 데이터의 전처리

가지고 있는 데이터는 음악들에 대한 미디 데이타로 이 데이타를 가지고 아티스트 별로 코퍼스를 구성한다. 아티스트 별로 코퍼스를 구성하면 한 코퍼스에는 한 아티스트의 곡만이 담겨 있게 되고 따라서 코퍼스가 각 가수의 스타일에 대한 정보를 담고 있다고 기대할 수 있다.

위에서 말한 곡들은 데이터는 미디 데이타이다. 먼저 이 미디 데이타에서 단선율의 멜로디를 추출해야 한다. 이 단선율의 멜로디를 추출하는 과정은 사람이 직접 들어가며 다선율의 멜로디중 중심이 되는 멜로디를 선택하고 이 멜로디의 데이터를 추출하는 방식으로 진행하였다. 또한 이번 프로그램에서는 쉼표를 제거하고 음표의 나열로만 멜로디를 가정하였다. 위와 같은 처리를 하면 음표의 높이와 길이의 양자화된 값을 가지고 있는 심볼릭한 단선율 멜로디 데이터를 생성할 수 있다. 이제 이 데이터를 HTM predictor에게 넘겨주기 위하여 벡터형식으로 바꾸어야 하고 각 벡터에 대하여HTM이 supervised learning을 진행할 수 있도록 class label을 만들어 주어야 한다. 아래의 그림 4 는 단선율 멜로디 데이터로부터 벡터와 class를 얻는 방식을 정리한 그림이다. Symbolic 한 음들의 데이터가 나열되어 있을 때 높이와 길이에 대해서 따로 sliding window 기법으로 벡터들을 생성하고 각 벡터의 다음 음에 대한 정보를 class 값으로 정하게 된다. 좀 더 자세히 설명하기 위해 벡터의 길이가 4 라고 가정해보겠다. 이 때 생성한 첫 번째 벡터는 첫 번째 음부터 네 번째 음까지의 값으로 구성되며 두 번째 벡터는 두 번째 음부터 다섯 번째 음까지의 값이다. 또한 첫 번째 벡터의 class label은 다섯번째 음에 대한 값이다. 두 번째 벡터의 class label은 여섯 번째 음에 대한 값이다.



Pitch	Duration
70	4
68	8
68	8
60	2
81	2
80	3
80	4
70	1
78	1

V1	V2	V3	V4	class
70	68	68	60	81
68	68	60	81	80
68	60	81	80	80
60	81	80	80	70
81	80	80	70	78

V1	V2	V3	V4	class
4	8	8	2	2
8	8	2	2	3
8	2	2	3	4
2	2	3	4	1
2	3	4	1	1

그림 4. Midi data 전처리의 예

위에서 설명한 sliding방식으로 벡터를 나누는 데이터 처리는 학습되는 데이터 형식을 N-gram 형식으로 나누는 것이라고 볼 수 있다. 음악의 구조 또한 자연언어의 구조와 비슷하다고 볼 수 있으며[9] 자연언어에서 Information Retrieval을 할 때 많이 사용하는 방식이 N-gram 기반의 방식이기 때문에 본 연구에서는 음악을 N-gram 기반으로 나누어 처리 하였다. 또한 위의 처리 방식을 보면 음에 대한 정보를 높이와 길이로 나누어 처리한다는 것을 확인할 수 있다. 음악은 beat중심의 뼈대 위에 음높이에 대한 정보가 입혀지는 것이라고 볼 수 있기 때문에 위와 높이와 길이를 따로 처리하였다. 또한 높이 와 길이를 한번에 고려하여 벡터를 생성할 경우 벡터의 class label의 수가 높이와 길이의 조합만큼 많아지게 되고 따라서 정확도에 문제가 있을 수 있다. 이러한 문제들 때문에 실제 음악 생성과 기억에서 높이와 길이를 나누어 처리하게 되었다.

C. 분류기를 이용한 순차적 멜로디 생성

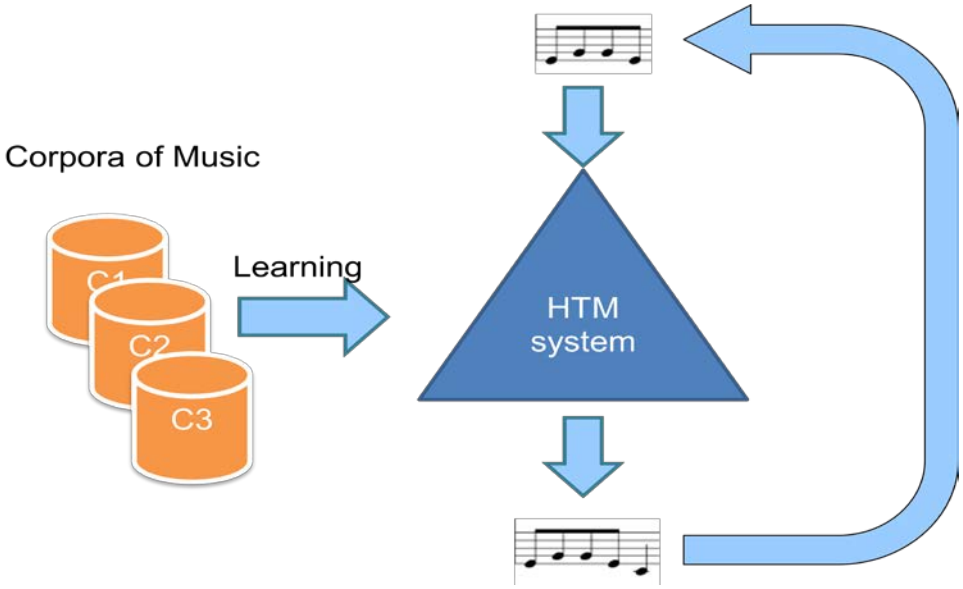


그림 5. HTM 기반의 Sequential Music Prediction 시스템

생성하는 시스템의 목표는 원곡을 최대한 잘 외우고 또한 특정한 코퍼스에 대해서 학습을 하였을 때 학습시킨 데이터들과 비슷한 스타일의 멜로디를 작성하는 모델을 생성하는 두 가지로 생각 해볼 수 있다. 본 연구에서는 위 두 가지의 목표를 이루기 위하여 HTM 을 이용한 sequential prediction 을 사용하여 멜로디를 학습하고 재생성한다.

Sequential Prediction 방식으로 멜로디 재생성을 하기 위하여 먼저 predictor 인 HTM 모델이 코퍼스 내의 데이터를 학습하여 학습된 네트워크를 만들어야 한다. 네트워크의 학습은 위에서 처리한 데이터들을 가지고 이루어지며 네트워크들이 각 코퍼스들을 따르게 하기 위하여 코퍼스들마다 따로 네트워크를 만들어 학습시킨다. 이번 연구에서 사용된 코퍼스는 세 개이므로 각 코퍼스당 높이, 길이

네트워크 두 개씩을 할당해 총 여섯 개의 학습된 HTM 네트워크 모델을 생성하게 된다. 이렇게 생성된 학습이 완료된 네트워크들은 각각이 predictor 로써 작동하게 된다.

학습된 HTM 네트워크로 이루어진 HTM predictor 가 있을 때 멜로디의 생성은 위의 그림 5 처럼 sequential prediction 방식으로 진행된다. 먼저 HTM predictor 에게 일정한 개수의 벡터로 이루어진 melody cue 를 질의로 주게 된다. HTM predictor 는 받은 멜로디 cue 를 분석하여 각 벡터의 class label 을 분석하게 된다. 이중에서 마지막 벡터의 class label 은 주어진 멜로디 cue 의 다음 음에 대한 예측 값이다. 따라서 이 값을 알면 우리는 sliding window 기법을 사용하여 기존의 cue 에 있던 벡터의 다음 벡터를 하나 더 생성할 수 있다. 이 벡터를 원래 predictor 에게 주어졌던 멜로디 cue 에 더하여 새로운 cue 를 생성할 수 있고 생성 된 cue 를 사용하여 위의 과정을 반복하면, 맨 처음 cue 의 종류에 따라 높이 또는 길이의 sequence 를 구할 수 있다. 이제 높이와 길이에 대해서 생성된 sequence 를 다시 합치면 멜로디 sequence 를 얻을 수 있다.

IV. 실험 및 결과

A. HTM 모델 구축

HTM 모델 설정은 구조 결정 단계와 데이터 기반의 학습 단계로 구분된다. HTM의 구조는 II-3 절에서 설명한 region을 노드로 하는 계층적 네트워크로 구성된다. 본 연구에서 사용할 코퍼스를 기준으로 기본적인 예측 실험 결과 그림 6 과 같은 구조를 최종 선택하였다.

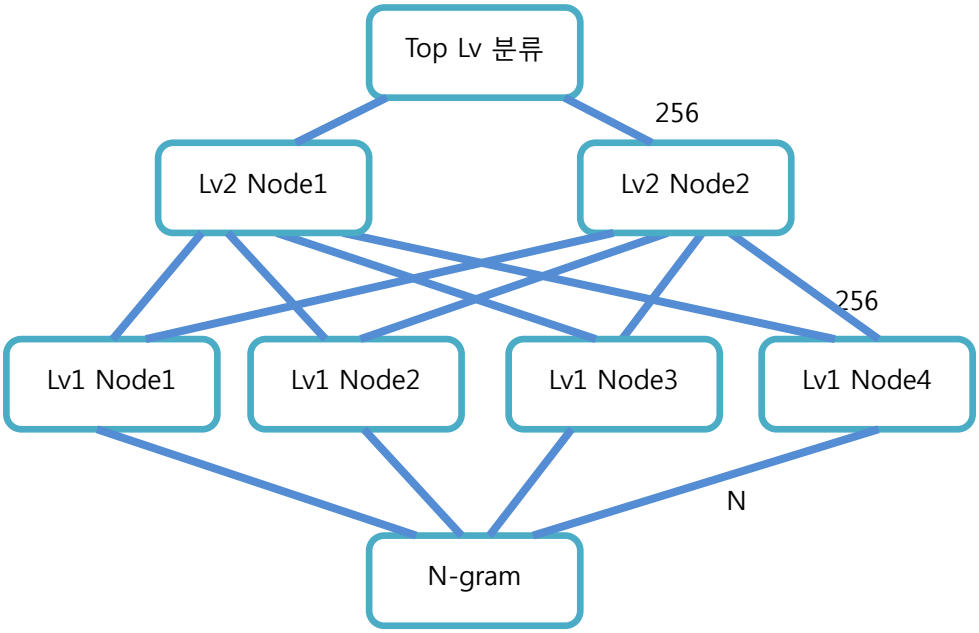


그림 6. 멜로디 학습 및 생성에 사용한 HTM 모델의 계층적 구조.구성한 HTM네트워크는 네 개의 Level 1 Region들과 2 개의 Level2 Region들로 이루어져 있으며 Toplevel에는 Classifier가 연결되어 있다. 노드 간 연결선은 표기한 숫자만큼의 길이를 가지는 벡터이다.

데이터 기반 학습 단계에서는 노드 내부 및 노드 간 연결선에 대한 최적의 가중치를 탐색하여 설정하게 된다.

HTM predictor 모델 구현에는 HTM 네트워크에 대한 설계와 학습 추론 등의 기능을 지원하는 API인 NuPIC[1]을 사용하여 설계하였다. 맨 하위 N-gram Input Node는 Corpus에서 데이터를 읽어 Level 1 Node들에게 전달해 준다. Level 1 노드들은 이 벡터들로부터 내부의 뉴런들의 정보들을 이용하여 node당 각각 길이 256 인 벡터를 만들고 이를 Level2 노드에 전해준다. Level2 의 노드는 전달받은 벡터들을 취합하여 내부의 뉴런들에 대한 정보들을 이용하여 각각 길이 256 의 벡터를 생성한다. 이 벡터들은 Top Level Node에 전달된다. Top Level은 분류기로 Learning mode에서는

[1] **NuPIC (Numenta Platform for Intelligent Computing),** http://www.numenta.com/legacysoftware.php

아래 Layer들이 처리한 벡터와 함께 이 벡터의 Label값을 함께 전달받게 된다. Top Level의 분류기는 기존에 있는 분류기들인 KNN(K-nearest neighbor), SVM(Support Vector Machine), NB(Naïve Bayes) 분류기 세 개의 분류기들 중에 선택할 수 있으며 실험에 따라 서로 다른 분류기를 사용하였다.

B. 학습데이터

학습을 위해 밴드 Beatles의 곡 30 곡과 밴드 Roxette의 곡 30 곡의 midi data를 수집하였다. 이 중에서 Beatles의 곡을 15 곡씩 나누어 Beatles Corpus두 개를 구성하고 Roxette의 곡 30 곡을 가지고 Roxette 의 곡들에 대한 코퍼스를 구성하였다. 그리고 위에서 설명한 방식으로 멜로디 부분을 추출하고 전처리 하여 세 개의 코퍼스를 구성하였다.

코퍼스 명칭	곡 수	특성 / 설명
BC1	15	Beatles곡 사전식 배열로 앞 15 곡
BC2	15	Beatles곡 사전식 배열로 뒤 15 곡
Roxette	30	Roxette 30 곡

표 1. Corpus 구성. Beatles의 곡 30 곡을 15 곡씩 나누어 두 개의 코퍼스를 구성하였고 Roxette의 곡들로 하나의 corpus를 구성하였다.

C. HTM의 학습 효율에 대한 실험

이 실험들은 HTM이 실제적으로 음악에 대한 데이터를 전달받았을 때 음악 데이터에 대한 학습을 할 수 있는지 확인하는 실험이다. 실험은 BC1 와 BC2 에 대하여 학습한 네트워크를 가지고 교차적으로 예측률 확인해 학습한 코퍼스에 대한 예측률이 높은지 확인하였으며 학습곡선을 그려 보았다. 또한 HTM의 분류기중 가장 성능이 좋은 분류기를 찾기 위하여 분류기에 따른 학습 성능을 확인해 보았다. 실험의 결과는 아래의 표 2, 표 3, 표 4 와 그림 7 과 같다.

표 2와 표3의 실험에 쓰인 분류기는 NB 분류기를 사용한 네트워크를 사용하여 실험을 설계하였다. 표 2와 표 3의 결과들은 HTM의 학습이 얼마나 잘 되는지 보여주는지를 보여주는 표이다. 실제적으로

N(입력 길이)	4	8	12	16
BC1/BC1	0.36±0.11	0.59±0.11	0.57±0.09	0.49±0.10
BC2/BC2	0.41±0.09	0.64±0.07	0.59±0.09	0.53±0.11
BC1/BC2	0.21±0.08	0.15±0.05	0.11±0.04	0.14±0.09
BC2/BC1	0.16±0.07	0.10±0.08	0.10±0.06	0.08±0.09

표 2. 음높이(pitch) 기준, 입력 벡터의 길이에 따른 학습/테스트 정확도 (평균±표준편차)

Corpus에 대한 학습률이 높다는 것을 보여주는 표이다. 실제로 BC1네트워크로 BC1곡들에 대한 예측을 한 예측확률이나 BC2네트워크로 BC2의 곡들을 예측한 예측확률이 BC1네트워크로 BC2의 곡들을 예측한 예측확률이나 BC2의 네트워크로 BC1의 곡들에 대한 예측을 수행한 예측확률보다 높게 나오는 것을 확인할 수 있다.

N(입력 길이)	4	8	12	16
BC1/BC1	0.47±0.11	0.60±0.11	0.63±0.08	0.61±0.11
BC2/BC2	0.49±0.09	0.65±0.08	0.70±0.10	0.69±0.06
BC1/BC2	0.21±0.08	0.26±0.15	0.24±0.17	0.18±0.18
BC2/BC1	0.16±0.07	0.40±0.20	0.37±0.20	0.35±0.19

표3. 음길이(duration) 기준, 입력 벡터의 길이에 따른 학습/테스트 정확도 (평균±표준편차)

학습곡선의 측정은 BC1 corpus에 대해서 이루어 졌으며 HTM의 분류기로는 1-NN를 사용하였다. BC1 corpus에 대한 학습곡선은 학습이 진행됨에 따라 약간의 요동치는 모습이 있긴 하지만 전반적으로 학습이 진행되면서 일정한 예측확률에 수렴하는 것을 확인할 수 있다.

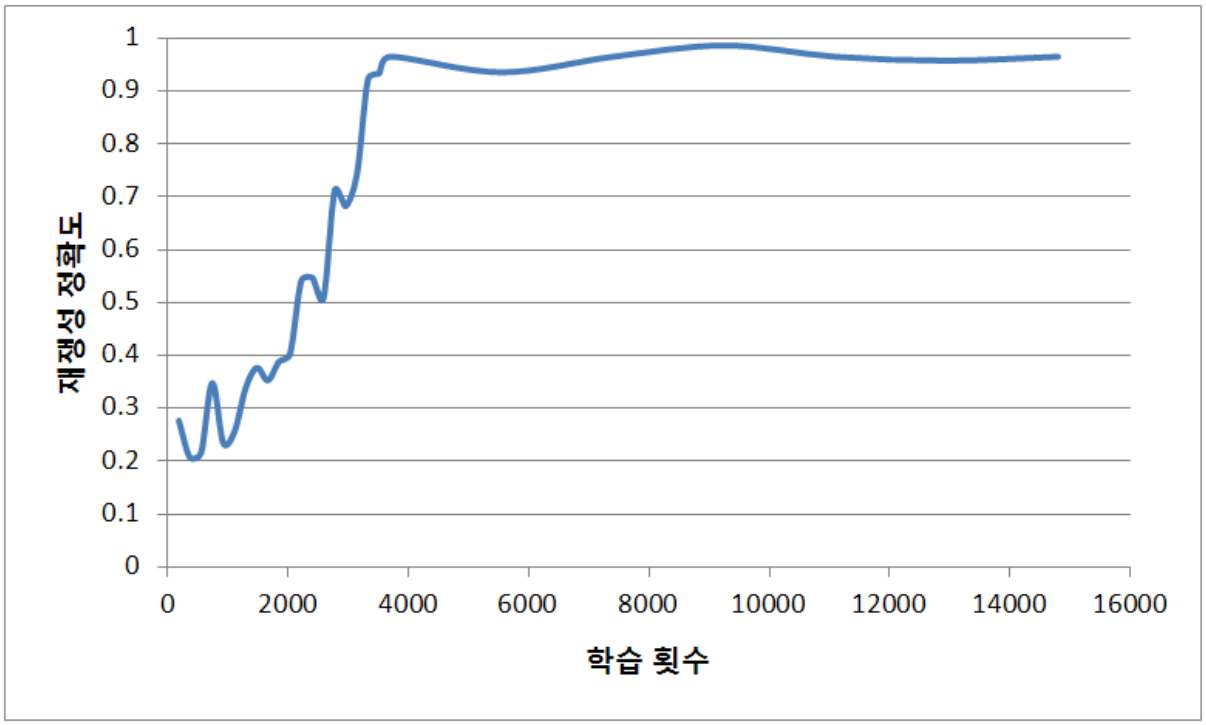


그림 7. BC1에 대한 학습곡선, 1-NN 분류기를 사용하였다.

분류기	1-NN	SVM	NB Classifier
BC1 with BC1	96.4%	59.2%	59.2%
BC2 with BC1	12.7%	0.0%	7.7%

표 4. 8-gram 데이터에 대해서 분류기별 예측률

마지막 표 4 는 8-gram 데이터에 따라서 분류기별 멜로디의 예측률이 어떠한지 보여준다. 표를 보면 1NN 분류기를 사용하였을 때 학습한 데이터와 학습하지 않은 데이터 모두에 대하여 더 좋은 성능을 보인다는 것을 확인할 수 있다.

D. HTM의 사용유무에 따른 예측확률 비교

HTM의 top node는 일반적으로 많이 쓰이는 종류의 classifier이다. 이 때 n-gram 입력이 HTM을 거치는 것이 필요한지에 대하여 체크하기 위하여 n-gram 입력을 HTM을 거쳐 top node로 가는 경우와 HTM을 거치지 않고 직접 classifier로 입력하는 경우를 비교하여 보았다. 실험 결과는 아래 표 5 와 같다.

Setting	Accuracy
HTM_NB_4	**34.5%**
HTM_NB_8	**59.2%**
HTM_NB_12	**57.4%**
NB_4	28.4%
NB_8	26.8%
NB_12	29.4%

(a)

Setting	Accuracy
HTM_1NN_4	**88.6%**
HTM_1NN_8	96.4%
HTM_1NN_12	93.7%
1NN_4	80.4%
1NN_8	**96.5%**
1NN_12	**97.8%**

(b)

Setting	Accuracy
HTM_SVM_4	**33.0%**
HTM_SVM_8	**59.2%**
HTM_SVM_12	**57.4%**
SVM_4	28.9%
SVM_8	25.6%
SVM_12	27.1%

(c)

표 5. HTM사용유무에 따른 정확도. (a) Naïve Bayes (b) 1-Nearest neighbor (c) SVM을 각각 분류기로 적용하였을 때, HTM을 거친 경우 대부분 성능이 향상되었음을 확인할 수 있다. 1-nearest neighbor 모델의 경우가 예외.

표 5 는 생성된 Corpus들에 대한 Accuracy분석을 할 때 HTM을 통과하지 않은 Corpus의 데이터들을 분류기만을 사용하여 분석하였을 때와 HTM을 사용한 후에 분류기를 사용하여 분석한 결과에 대하여 정확도를 비교한 표이다. 분석은 4-gram, 8-gram, 12-gram으로 나누어 분석하였으며 높이데이터에 대해서만 분석하였다. 표를 보면 대부분의 결과에서 HTM을 사용한 결과가 더 높게 나오는 것을 확인할 수 있다. HTM은 네트워크내부에서 입력 받은 정보들과 뉴런에 대한 정보를 사용하여 좀 더 높은 차원의 정보로 전환된다. 이 뉴런에 대한 정보는 이미 학습된 데이터의 패턴들에 대한 정보를 가지고 있으며 또한 시간적인 정보를 함께 고려한 정보이다. 즉 낮은 차원의 N-gram 기반 벡터를 입력 받아서 HTM이 그 벡터를 좀 더 높은 차원의 벡터로 사상해주는 역할을 한다고 해석할 수 있다.

E. 아티스트의 스타일 학습 확인

HTM이 실제로 코퍼스별 아티스트의 스타일을 학습하는지 확인하기 위하여 BC1 을 가지고 BC2 의 곡들과 Roxette의 곡들에 대해서 예측률을 확인해 보았다. 결과는 아래의 표 6 과 같다.

Test Corpus	BC2	Roxette
4-gram	12.9%	0%
8-gram	12.7%	0%
12-gram	10.3%	0%

표 6. 아티스트 별 예측 확률. 음높이 데이터만을 사용하여 정확도를 비교하였다.

표 6 은 BC1 으로 훈련한 HTM 네트워크를 가지고 BC2 의 곡들과 Roxette의 곡들에 대한 예측확률을 비교한 표이다. HTM 네트워크의 분류기로는 KNN 분류기를 사용하였다. 같은 코퍼스내의 음악을 예측한 것이 아니므로 전체적으로 예측확률이 낮은 것을 볼 수 있다. 또한 BC2 의 곡들에 대한 예측확률이 Roxette의 곡들에 대한 예측확률보다 높은 것을 볼 수 있는데 이는 HTM 네트워크가 아티스트의 스타일을 따르도록 훈련되는 것이라고 생각할 수 있다. Roxette의 곡들에 대해서는 전혀 예측을 성공하지 못하였다. 이는 다음과 같이 생각할 수 있다. 본 연구에서는 높이와 길이에 대한 값으로 절대적인 값을 사용한다. 그런데 Roxette의 곡들은 분포되어있는 높이 value가 Beatles의 곡들과는 확연한 차이를 보인다. 따라서 음의 예측을 절대값이 아니라 음악의 첫 음부터 상대적인 음의 변화를 이용하여 예측하였으면 예측확률이 올라갔을 것으로 생각된다.

F. 생성된 음악의 예

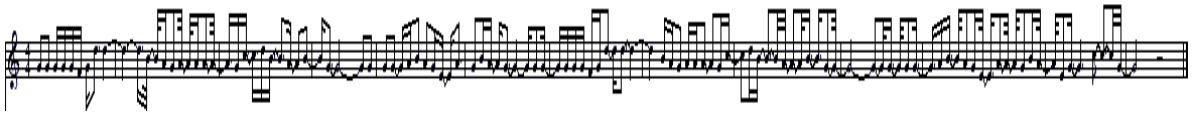


그림 8. BlackBird 원곡

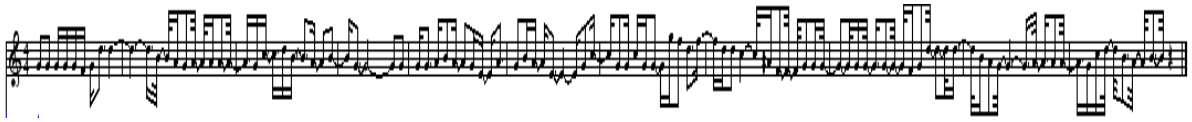


그림 9. BC2로 생성된 BlackBird

그림 10. BC1로 생성된 BlackBird

그림 8-10은 Beatles의 곡 중의 하나인 BlackBird의 음악을 cue로 주었을 때 생성된 악보들이다. 앞의 12음은 cue로 주어진 음으로 세 악보에서 똑같으며 그 후로 이어지는 음표는 50개이다. 악보 2는 BlackBird가 Corpus의 내부에 포함된 BC2로 훈련한 네트워크를 사용하여 생성한 악보이다. 악보를 보면 실제 생성된 곡의 중간부분까지 원곡과 똑 같은 곡을 생성하다가 음이 달라진 이후에 원곡과 다른 곡을 악보3은 BC1으로 훈련한 네트워크를 사용하여 생성한 악보이다.

V. 결론 및 향후 과제

A. 결론

본 연구에서는 음악의 멜로디 예측을 Sequential Prediction, 즉 주어진 멜로디에 대해서 예측한 값을 이용해 다시 예측을 반복하는 문제로 생각하여 주어진 질의에 대한 이어진 멜로디를 예측하는 방식으로 문제를 풀었다. 이 때 문제가 되는 음악의 데이터를 N-gram 기반의 벡터 표현으로 나타내었고 각 벡터에 대하여 다음 음에 해당하는 값을 Class Label로 주었다. 이 때 음악의 예측은 음악에 대한 벡터를 보고 벡터의 class를 유추하는 Classification 문제로 생각할 수 있다. 음악의 예측을 Classification 문제로 생각할 때 기존의 Classification과 다른 점은 기존의 Classification에서 데이터는 벡터의 순서는 상관없이 벡터와 각 벡터의 Class Label만이 의미가 있지만 음악에서는 벡터의 순서가 의미가 있다는 것이다. HTM 모델은 학습된 시간적이고 이러한 벡터의 순서가 의미가 있는 Classification 문제에서 구조적인 패턴들을 인지하여 좀 더 높은 차원의 벡터로 변환하고 이 벡터들을 분류기를 통해 분류한다고 가정하였고, 실제로 HTM을 통과하지 않은 채 분류기만을 사용하여 예측한 정확도가 HTM을 사용하여 예측한 정확도보다 낮은 것을 확인할 수 있었다. 학습곡선을 통해 HTM이 반복적인 학습을 통하여 데이터를 학습할 수 있다는 것을 확인할 수 있었으며 아티스트 별로 분류된 Corpus를 훈련한 HTM 네트워크를 사용하여 HTM 네트워크가 아티스트의 스타일을 따를 수 있다는 것을 보일 수 있었다.

B. 향후 과제

본 연구에서는 음의 높이와 길이를 학습하고 예측하는데 있어서 절대적인 값을 사용하였다. 이 방식으로 접근하면 음악의 조성에 독립적이지 못한 학습을 하게 된다. 이런 식의 학습이 이루어질 경우 조성이 다르거나 템포가 다르다면 멜로디의 흐름이 비슷하더라도 정확한 학습과 예측을 하는데 어려움이 있다. 따라서 향후 연구에서는 음의 높이와 길이에 대하여 절대적인 값이 아니라 상대적인 값을 사용한다면 좀 더 적은 패턴으로 좀 더 다양한 음악 데이터를 더 정확하게 학습하고 추론할 것이라고 생각한다. 또한 HTM은 HTM의 여러 세팅과 HTM에게 주어지는 데이터의 길이에 따라 서로 다른 예

측을 할 수 있다. 이러한 서로 다른 예측을 하는 HTM들을 잘 조합한다면 HTM을 이용한 앙상블 시스템을 구현할 수 있을 것이라고 생각한다.

VI. 감사의 글

먼저 논문 지도교수님인 장병탁 교수님께 감사하다는 말씀을 드립니다. 장병탁 교수님 덕분에 기계학습에 대한 이해와 연구를 할 수 있었던 것 같습니다. 또한 논문을 지도해주신 김병희 선배께서 부족한 저를 이끌어 하나하나 세심하게 가르쳐주셔서 제가 연구에 대한 감을 잡을 수 있었고 이 논문도 완성될 수 있었던 것 같습니다. 감사드립니다.

VII. 참고문헌

[1] B.-T. Zhang, Hypernetworks: A molecular evolutionary architecture for cognitive learning and memory, *IEEE Computational Intelligence Magazine*, vol. **3,** No. 3, pp. 49-63, 2008.

[2] D. Cope. Panel discussion. In *Proceedings of the International Computer Music Conference*, 1993.

[3] http://sunsite.univie.ac.at/Mozart/dice/

[4] 황성호, 전자음악의 이해, 현대음악 출판사, 1993

[5] K. Ebcioglu, An expert system for harmonizing four-part chorales, *Computer Music Journal*, vol. 12, pp. 43-51, 1988.

[6] 김현우, 음악 학습 및 생성을 위한 하이퍼네트워크 연구, 석사학위 논문, 서울대학교 컴퓨터공학부, 2010.

[7] M.I. Bellgard and C.P. Tsang, Harmonizing music the Boltzmann way, *Connection Science*, vol. 6, pp.281-297, 1994.

[8] Numenta, "HTM Cortical Learning Algorithms"

[9] J. Wolokwicz, Z. Kulka and V. Keselj, N-gram-based approach to Composer Recognition, *Archives of Acoustics*, vol. 33, issue 1, pp 43-55, 2008.