석사학위논문

딥러닝 기반 자동작곡에서 구성을 갖춘 곡 생성방법에 대한 연구

2021년

한성대학교 지식서비스&컨설팅대학원

미래융합컨설팅학과

미래융합전략컨설팅전공

정 석 환

석 사 학 위 논 문
지도교수 정성훈

딥러닝 기반 자동작곡에서 구성을 갖춘 곡 생성방법에 대한 연구

A Study on the Generating Method of Songs with Structure in Automatic Composition Based on Deep Learning

2020년 12월 일

한성대학교 지식서비스&컨설팅대학원

미래융합컨설팅학과

미래융합전략컨설팅전공

정 석 환

석 사 학 위 논 문
지도교수 정성훈

딥러닝 기반 자동작곡에서 구성을 갖춘 곡 생성방법에 대한 연구

A Study on the Generating Method of Songs with Structure in Automatic Composition Based on Deep Learning

위 논문을 컨설팅학 석사학위 논문으로 제출함

2020년 12월 일

한성대학교 지식서비스&컨설팅대학원

미래융합컨설팅학과

미래융합전략컨설팅전공

정 석 환

정석환의 컨설팅학 석사학위논문을 인준함

2020년 12월 일

심사위원장 _____(인)

심 사 위 원 _____(인)

심 사 위 원 _____(인)

국 문 초 록

딥러닝 기반 자동작곡에서 구성을 갖춘 곡 생성방법에 대한 연구

한성대학교 지식서비스&컨설팅대학원
미 래 융 합 컨 설 팅 학 과
미 래 융 합 전 략 컨 설 팅 전 공
정 석 환

최근 딥러닝 기술의 발전으로 인해 거의 모든 분야에서 인공지능을 활용한 방법이 도입되고 있다. 심지어 인공지능은 사람만 할 수 있다고 여겨져 왔던 창작의 영역에서도 활발하게 적용되고 있다. 2018년 뉴욕 크리스티 경매장에서는 최초로 인공지능이 그린 작품인 '에드몬드 벨라미의 초상화(Edmond De Belamy)'가 약 5억원(432,000달러)에 판매되었다. 음악에서도 인공지능은 예외가 아니다. 인공지능을 사용하여 새로운 곡을 작곡하는 많은 시도가 이루어지고 있다. 2016년 구글은 음악과 미술 창작이 가능한 인공지능 알고리즘을 설계하는 마젠타 프로젝트를 발표하였다. 그러나 이러한 여러 가지 시도에도 불구하고 아직도 사람이 작곡한 것과 같은 자연스러운 구성을 갖춘 곡을 생성하는 경우는 찾아보기 어렵다.

일반적으로 곡은 도입부, 전개부, 간주, 후렴부, 후주 등의 일정한 형식을 가지고 있다. 기존의 인공신경망을 이용한 자동작곡 모델들은 학습된 곡의 멜로디를 가지고 새로운 곡을 출력하므로 사람이 작곡한 것과 같은 특정한

구성을 갖춘 곡을 출력하는 데는 어려움이 있다. 본 논문에서는 인공신경망을 이용한 자동작곡에서 음악 구성적으로 부족한 부분을 개선하기 위해 멜로디에 곡 구성 정보를 함께 넣어주는 방법을 제안한다. 이를 위하여 기존 음악 데이터를 시간의 흐름에 따라서 변하는 동적 데이터인 멜로디와 보조적인 정적 데이터인 곡 구성 정보로 나누어 학습을 시키는 방법을 고안하였다. 실험 데이터로는 기승전결의 단순한 구성을 가지는 동요 곡들을 사용하였고, 생성된 곡의 정량적 판단을 위해 METEOR 점수와 BLEU 점수를 사용하여 평가하였다. 실험 결과 제안한 모델을 사용하였을 때 새로운 멜로디와 함께 의도한 대로 곡 구성 순서에 맞춰서 곡이 창작되는 것을 확인할 수 있었다. 또한 METEOR와 BLEU를 사용한 평가에서도 더 우수한 성능을 보이는 것을 확인할 수 있었다.

【주요어】 자동작곡, 딥러닝, 인공신경망, 동적 데이터와 정적 데이터 결합, 곡 구성, METEOR, BLEU

목 차

I. 서 론 ····· 1

1.1 연구 배경 ····· 1

1.2 연구 내용 ····· 2

II. 이론 및 배경 ····· 4

2.1 순환신경망 ····· 4

2.2 LSTM ····· 6

2.3 단어 임베딩 ····· 10

2.4 정적 데이터와 동적 데이터를 함께 학습시키는 방법 ····· 12

2.4.1 간접 입력 방법 ····· 13

2.4.2 직접 입력 방법 ····· 14

2.5 정량적 평가 방법 ····· 14

2.5.1 BLEU ····· 15

2.5.2 METEOR ····· 19

III. 구성을 갖춘 자동작곡 시스템 ····· 21

3.1 데이터 전처리 ····· 21

3.1.1 미디 데이터 ····· 21

3.1.2 곡 구성 정보 ····· 23

3.2 자동작곡 시스템 ····· 24

3.2.1 멜로디 학습 모델 ····· 24

3.2.2 곡 구성 정보 직접 입력 모델 ····· 25

3.2.3 곡 구성 정보 간접 입력 모델 ····· 26

IV. 실험 결과 ····· 28

4.1 실험 데이터 ····· 28

4.2 멜로디 학습 모델 ····· 29

4.3 곡 구성 정보 직접 입력 모델 ····· 30
4.3.1 AAAABBBBCCCCDDDD 형식으로 작곡 ····· 30
4.3.2 AABBCCDDAABBCCDD 형식으로 작곡 ····· 31
4.3.3 AAAAAAAABBBBBBBB 형식으로 작곡 ····· 31
4.3.4 AAAAAAAAAAAAAAAA 형식으로 작곡 ····· 32
4.3.5 DDDDCCCCBBBBAAAA 형식으로 작곡 ····· 33
4.4 곡 구성 정보 간접 입력 모델 ····· 33
4.4.1 AAAABBBBCCCCDDDD 형식으로 작곡 ····· 33
4.4.2 AABBCCDDAABBCCDD 형식으로 작곡 ····· 34
4.4.3 AAAAAAAABBBBBBBB 형식으로 작곡 ····· 35
4.4.4 AAAAAAAAAAAAAAAA 형식으로 작곡 ····· 35
4.4.5 DDDDCCCCBBBBAAAA 형식으로 작곡 ····· 36
4.5 곡 평가 ····· 37
4.5.1 모델별 평가 ····· 37
4.5.2 곡 구성 방식별 평가 ····· 39
V. 결 론 ····· 45
참 고 문 헌 ····· 46
ABSTRACT ····· 50

표 목 차

[표 2-1] 원-핫 벡터 예 ····· 11
[표 2-2] Candidate2의 바이그램 $Count$와 $Count_{clip}$ ····· 17
[표 3-1] 음악 기호를 텍스트로 변환한 기호 ····· 22
[표 4-1] 각 학습 모델로 생성한 100곡의 METEOR 평균 점수 ····· 38
[표 4-2] 각 학습 모델로 생성한 100곡의 BLEU 평균 점수 ····· 38
[표 4-3] 곡 구성에 따른 METEOR 점수 (직접 입력 모델) ····· 39
[표 4-4] 곡 구성에 따른 BLEU 점수 (직접 입력 모델) ····· 40
[표 4-5] 곡 구성에 따른 METEOR 점수 (간접 입력 모델) ····· 41
[표 4-6] 곡 구성에 따른 BLEU 점수 (간접 입력 모델) ····· 42

그 림 목 차

[그림 2-1] 순환 경로를 포함하는 순환신경망 구조 ····· 5
[그림 2-2] 순환신경망의 펼쳐진 구조 ····· 5
[그림 2-3] 네 단어를 처리하는 순환신경망 ····· 6
[그림 2-4] 긴 문장을 처리하는 순환신경망 ····· 6
[그림 2-5] LSTM의 구조 ····· 7
[그림 2-6] LSTM의 셀 상태 전달 ····· 7
[그림 2-7] LSTM의 게이트 구조 ····· 8
[그림 2-8] LSTM의 망각 게이트 ····· 8
[그림 2-9] LSTM의 입력 게이트 ····· 9
[그림 2-10] LSTM의 새로운 셀 상태 업데이트 ····· 9
[그림 2-11] LSTM의 출력 게이트 ····· 10
[그림 2-12] 정적데이터를 다루는 모델과 동적데이터를 다루는 모델 ····· 12
[그림 2-13] 간접 입력 방법 ····· 13
[그림 2-14] 직접 입력 방법 ····· 14
[그림 2-15] METEOR 유니그램 정렬 예 ····· 19
[그림 3-1] 악보를 텍스트로 변환한 예 (고향의 봄) ····· 22
[그림 3-2] 곡 구성 정보를 추가한 음악 데이터 (고향의 봄) ····· 23
[그림 3-3] 마디 임베딩을 사용한 멜로디 학습 모델 ····· 24
[그림 3-4] 곡 구성 정보 직접 입력 모델 ····· 25
[그림 3-5] 곡 구성 정보 간접 입력 모델 ····· 26
[그림 4-1] 멜로디 학습 모델로 생성한 곡 1 ····· 29
[그림 4-2] 멜로디 학습 모델로 생성한 곡 2 ····· 29
[그림 4-3] 곡 구성 정보 직접 입력 모델로 생성한 곡 1 ····· 30
[그림 4-4] 곡 구성 정보 직접 입력 모델로 생성한 곡 2 ····· 31
[그림 4-5] 곡 구성 정보 직접 입력 모델로 생성한 곡 3 ····· 32
[그림 4-6] 곡 구성 정보 직접 입력 모델로 생성한 곡 4 ····· 32
[그림 4-7] 곡 구성 정보 직접 입력 모델로 생성한 곡 5 ····· 33
[그림 4-8] 곡 구성 정보 간접 입력 모델로 생성한 곡 1 ····· 34

[그림 4-9] 곡 구성 정보 간접 입력 모델로 생성한 곡 2 ····· 34
[그림 4-10] 곡 구성 정보 간접 입력 모델로 생성한 곡 3 ····· 35
[그림 4-11] 곡 구성 정보 간접 입력 모델로 생성한 곡 4 ····· 36
[그림 4-12] 곡 구성 정보 간접 입력 모델로 생성한 곡 5 ····· 36
[그림 4-13] 각 학습 모델별 METEOR 점수 그래프 ····· 37
[그림 4-14] 각 학습 모델별 BLEU 점수 그래프 ····· 38
[그림 4-15] 곡 구성에 따른 METEOR 점수 (직접 입력 모델) ····· 40
[그림 4-16] 곡 구성에 따른 BLEU 점수 (직접 입력 모델) ····· 41
[그림 4-17] 곡 구성에 따른 METEOR 점수 (간접 입력 모델) ····· 42
[그림 4-18] 곡 구성에 따른 BLEU 점수 (간접 입력 모델) ····· 43

Ⅰ. 서론

1.1 연구 배경

불과 몇 년 전까지만 하더라도 먼 미래의 이야기로만 여겨졌던 인공지능을 이용한 기술들이 요즘에는 생활 속에 쉽게 찾아볼 수 있게 되었다. 이미지 인식[Krizhevsky, A. 등., 2012], 음성 인식[Gevaert, W. 등., 2010] 또는 번역[Yonghui, W. 등., 2016]과 같은 분류 및 예측 작업에 비약적인 발전을 이루어 음성으로 명령을 실행하는 인공지능 스피커, 문자를 통으로 번역해 주는 인공지능 번역기, 사람 대신 상담해 주는 인공지능 챗봇, 도로상의 정보를 인식하여 자율 주행하는 자동차 등의 서비스가 빠르게 발전하고 있다. 예술 분야에서도 인공지능은 그 능력을 발휘하고 있다. 컴퓨터가 그림을 그리고 시를 쓰며[Tanel, K. 등., 2018] 음악을 작곡한다. 이러한 일 들이 이제는 더 이상 놀라운 미래의 일이 아니다.

작곡을 위한 도구로써 컴퓨터는 오래전부터 사용되어왔다. 1957년 미국 일리노이대의 레자렌 힐러(Lejaren Hiller)와 레너드 아이잭슨(Leonard Issacson)은 일리악(ILLIAC) I 컴퓨터를 이용해 최초로 컴퓨터 알고리즘을 사용한 클래식 곡을 만들었다. 최근에는 음악 분야에서도 인공신경망 알고리즘을 사용하여 자동으로 곡을 생성하는 방법이 사용되고 있다. Bob L. 등., (2016)은 RNN을 사용한 음악 생성을 제시하였고, Hao-Wen, D. 등., (2018)은 적대적 생성망(GAN)을 사용한 MuseGAN을 제안하였다. Kotecha, N., (2018)은 곡 생성에 강화학습을 사용하였다. 이러한 연구들로 인해 점점 컴퓨터에 의해 작곡된 음악의 완성도 또한 높아지고 있다. 그러나 컴퓨터에 의해 자동으로 작곡된 곡들은 멜로디는 멋지게 생성되었으나, 실제로 들어보면 곡이 연주 도중 갑자기 끝나거나, 반대로 언제쯤 곡이 마무리되는지 알 수 없는 경우가 많았다. 이는 작곡을 위해서 인공신경망을 학습시킬 때 단순히 곡의 멜로디만 가지고 학습이 이루어지기 때문에 전반적인 곡 구성을 갖추지 못하기 때문이다.

음악은 음표의 높낮이와 길이 장단의 조합으로 리듬과 멜로디를 만들게 된다. 하지만 단순히 리듬과 멜로디만으로 음악이 이루어지지 않는다. 곡을 작곡하기 위해서는 이러한 리듬과 멜로디뿐만 아니라 주제를 선정하고 전체적인 흐름을 생각하여 곡의 구성을 어떻게 전개할 것인지를 정하는 것 또한 중요하다. 예를 들면 가요나 팝송의 경우 일반적으로 intro - verse - chorus - brigde - verse - chorus - outro 형태의 구성을 가진다. 컴퓨터를 이용한 자동작곡에서 이러한 구성을 가진 곡을 생성하기는 쉽지 않은 일이다. 이러한 문제를 해결하고자 본 논문에서는 곡을 생성할 때 단순히 멜로디뿐만 아니라 생성된 곡이 전체적인 곡 구성을 가질 수 있는 모델을 제시하고자 한다.

1.2 연구 내용

음악곡은 시간에 따라서 계속 변하는 멜로디와 시간의 변화와 상관없이 일정한 값을 가지는 곡 구성 정보를 가지고 있다. 이런 다른 범주에 속하는 데이터를 결합하여 인공신경망의 성능을 향상하려는 연구들이 있었다. Rahman, M., 등., (2020)은 시간이 지남에 따라 값이 변하는 동적인 시계열 데이터와 고정된 값을 가지는 정적 데이터를 순환신경망의 입력으로 직접 사용하는 모델과 정적 데이터는 피드포워드 신경망으로 전달되고 동적 데이터는 순환신경망으로 개별적으로 전달된 후 최종 출력이 생성되기 전에 결합하는 간접 입력 모델을 소개하였다.

본 논문에서는 동적 데이터인 멜로디에 정적 데이터인 곡 구성 정보를 추가하는 방법으로서 순환신경망에 동적 데이터와 정적 데이터를 결합하여 입력하는 직접 입력(Direct input) 모델과 동적 데이터가 순환신경망을 지난 후 정적 데이터와 결합하는 간접 입력(Indirect input) 모델을 사용하여 구성을 갖춘 곡을 생성하는 모델을 제안한다. 순환신경망으로는 LSTM이 사용되었으며, 모델 학습 시 곡 구성 정보 학습을 위한 데이터로는 비교적 간단한 기승전결 구성으로 이루어져 있는 동요를 선택하였다.

본 논문의 구성은 다음과 같다. 1장 서론에서는 본 논문의 배경에 관해

기술하였다. 2장에서는 관련 연구로 순환신경망, LSTM, 단어 임베딩 그리고 정적 데이터와 동적 데이터를 함께 학습시키는 방법과 생성된 곡 평가를 위한 METEOR과 BLEU 점수에 관해서 설명한다. 3장에서는 본 논문에서 제안한 멜로디와 곡 구성 데이터를 함께 학습시키는 구성을 갖춘 자동작곡 시스템에 관해서 설명한다. 4장에서는 제안한 방법으로 작곡한 결과를 평가하여 기술한다. 5장에서는 결론으로 끝을 맺는다.

Ⅱ. 이론 및 배경

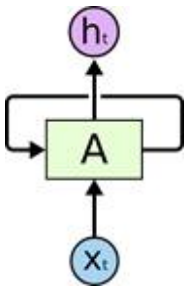
인공신경망을 사용한 자동작곡 모델은 자연어처리 모델과 매우 흡사하다. 우리가 일상생활 속에서 사용하는 언어를 자연어라 한다. 자연어처리란 이러한 인간의 언어를 컴퓨터가 처리할 수 있도록 하는 작업을 말한다.

본 장에서는 자연어처리에 사용되는 순환신경망(RNN, Recurrent Neural Networks), 장단기 메모리(LSTM, Long Short-Term Memory) [Hochreiter, S. 등., 1997], 단어 임베딩(Word Embedding)에 대해서 소개한다. 또한 시계열 데이터 학습 시 정적 데이터를 함께 학습시키는 기존 연구들에 대해서 살펴본다. 마지막으로 언어 모델을 평가하는 데 사용되는 METEOR과 BLEU 점수를 소개한다.

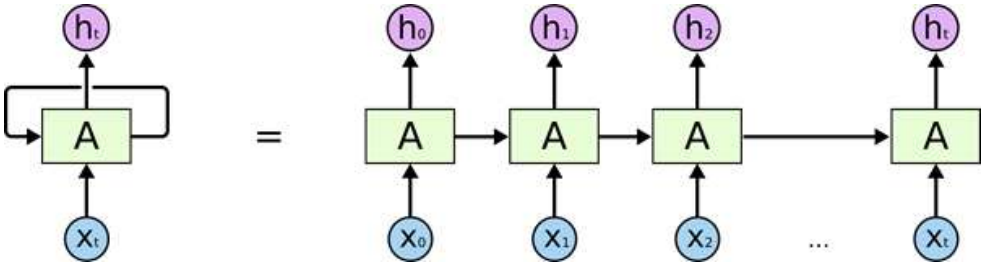
2.1 순환신경망

과거의 데이터가 미래의 데이터에 어떻게 영향을 미치는지를 예측하는 인공신경망 모델 중 순환신경망(Recurrent Neural Networks; RNN)은 일반신경망에서 시계열 개념이 추가된 것으로 은닉계층 이전 정보를 기억시킬 수 있는 장점이 있다. 이런 장점 때문에 지난 몇 년 동안 음성 인식, 언어 모델링, 번역, 이미지 캡션 등 다양한 문제에 순환신경망을 적용하여 놀라운 성공을 거두었다. 데이터의 시간적 순서 관계가 중요한 음악 데이터 학습에도 순환신경망은 적합한 모델이다.

그림 2-1에서 신경망 A는 입력 x_t와 h_{t-1}을 보고 h_t값을 출력한다. 순환 경로를 사용하면 네트워크의 한 단계에서 다음 단계로 정보를 전달할 수 있다. 이러한 순환 경로를 적용한 반복 신경망은 같은 네트워크의 여러 복사본으로 생각할 수 있으며 각 복사본은 데이터를 다음 신경망에 전달한다. 그림 2-2는 순환 경로를 펼친 구조를 보여준다.



[그림 2-1] 순환 경로를 포함하는 순환신경망 구조 (Christopher, 2015)



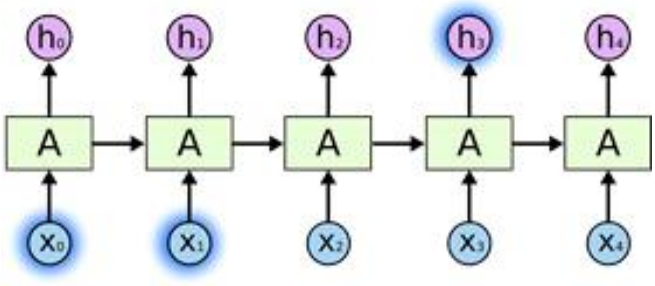
[그림 2-2] 순환신경망의 펼쳐진 구조 (Christopher, 2015)

순환신경망의 식 (2-1)에서 x_t는 입력 데이터, h_{t-1}은 이전 단계의 은닉계층 출력, W는 가중치, B는 바이어스를 나타낸다. t단계의 은닉계층 출력 h_t는 입력 데이터에 가중치를 곱하고 이전 단계의 은닉계층 출력에 가중치를 곱한 값과 바이어스를 더한 후 활성화 함수를 적용하여 산출된다. 이를 통해 순환신경망은 과거의 데이터가 다음 데이터에 어떤 영향을 주는지 학습할 수 있다. 활성화 함수로는 입력을 비선형 관계로 변환하는 활성화 함수인 tanh나 시그모이드 함수 등이 사용된다.

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t + B_h) \quad \text{(식 2-1)(Calvin, 2020)}$$

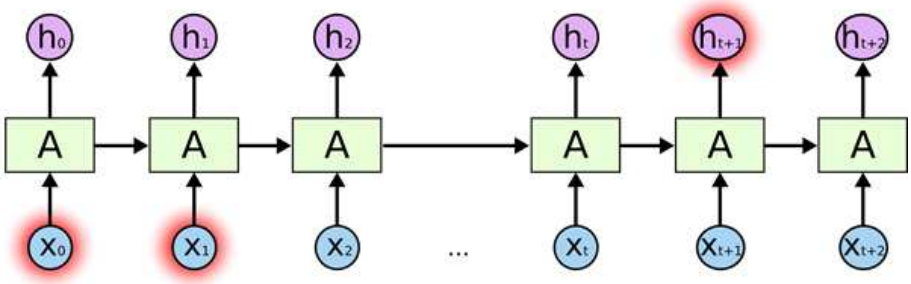
순환신경망 모델에서 현재 작업을 위해서 최근 정보만 볼 필요가 있는 때

도 있다. 예를 들면 그림 2-3과 같이 서너 단어의 짧은 문장에서 마지막 단어를 예측하는 경우에는 작은 순환신경망 모델로도 가능하다.



[그림 2-3] 네 단어를 처리하는 순환신경망 (Christopher, 2015)

그러나 그림 2-4와 같이 긴 문장에서 단어를 예측하는 경우 더 많은 맥락이 있어야 하므로 순환신경망은 정보들을 연결하는 방법을 학습할 수 없게 된다. 이를 장기종속성 문제라고 부른다.

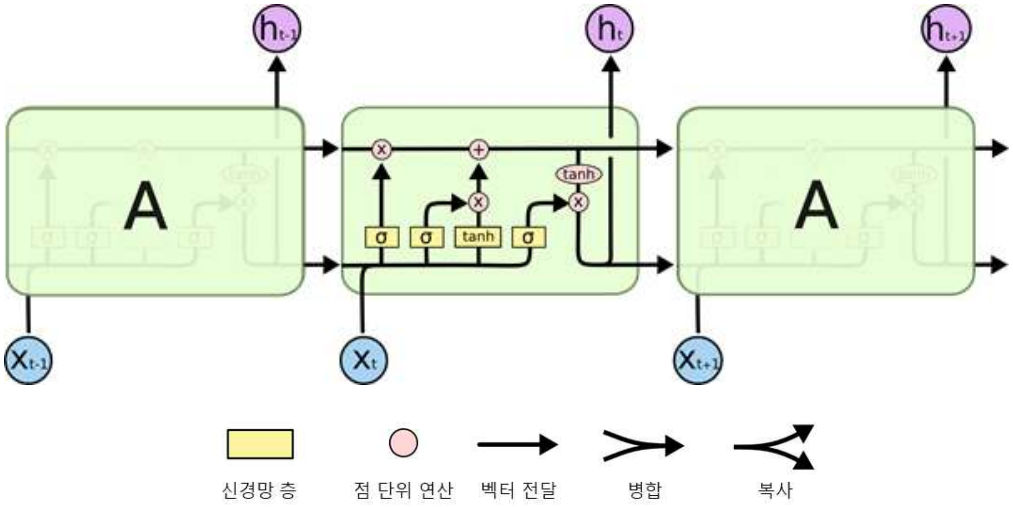


[그림 2-4] 긴 문장을 처리하는 순환신경망 (Christopher, 2015)

2.2 LSTM

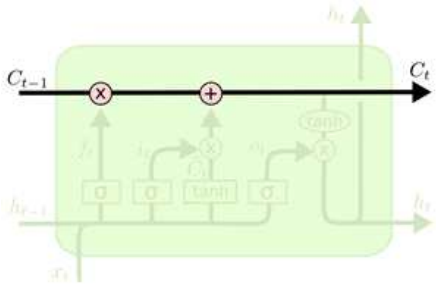
LSTM(Long Short-Term Memory)[Hochreiter, S. & Schmidhuber, J., 1997]은 장기종속성을 학습할 수 있는 특별한 종류의 순환신경망이다. LSTM

은 하나의 신경망 계층을 갖는 대신 특별한 방식으로 상호 작용하는 4개의 층을 가지고 있다.

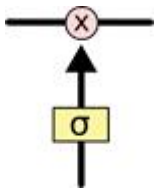


[그림 2-5] LSTM의 구조 (Christopher, 2015)

그림 2-5는 LSTM 구조를 나타내고 있다. 각 라인은 한 노드의 출력에서 다른 노드의 입력까지 전체 벡터를 전달한다. 작은 원은 벡터 추가와 같은 점 단위 연산을 나타내고 네모 상자는 학습된 신경망 계층이다. 병합되는 줄은 연결을 나타내며 분기되는 줄은 내용을 복사하여 다른 위치로 이동하는 것을 나타낸다.



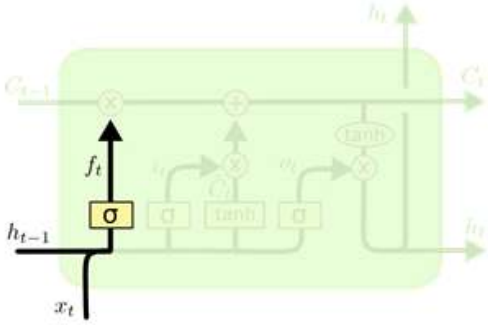
[그림 2-6] LSTM의 셀 상태 전달 (Christopher, 2015)



[그림 2-7] LSTM의 게이트 구조 (Christopher, 2015)

LSTM은 셀 상태에 정보를 제거하거나 추가하는 제어 기능을 가지는 게이트 구조로 되어 있다. 그림 2-6은 셀 상태가 일종의 컨베이어 벨트처럼 선형 상호작용을 통해 LSTM 전체 체인을 따라 똑바로 실행되는 것을 보여준다. 그림 2-7은 게이트 구조를 나타낸다. 게이트는 시그모이드 층을 통해 0과 1 사이의 값을 출력하여 각 구성 요소가 통과해야 하는 양을 정한다.

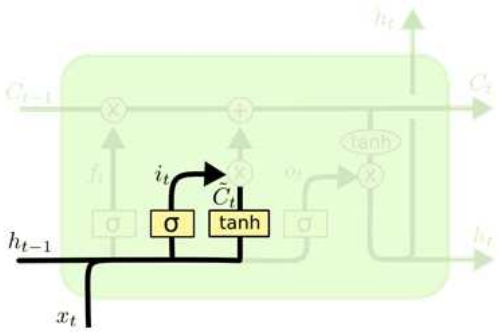
LSTM에는 세 개의 게이트가 있다. 첫 번째 단계는 셀 상태에서 버릴 정보를 정하는 망각 게이트이다. 그림 2-8은 LSTM의 망각 게이트를 보여준다. 식 2-2에 의해 시그모이드 층은 h_{t-1}과 x_t를 보고 셀 상태 C_{t-1}에 대해서 이 0에서 1 사이의 값을 출력한다.



[그림 2-8] LSTM의 망각 게이트 (Christopher, 2015)

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \qquad \text{(식 2-2)}$$

두 번째 단계는 셀 상태에 저장할 정보를 결정하는 입력 게이트이다. 그림 2-9는 LSTM의 입력 게이트를 보여준다. 식 2-3에 의해 시그모이드 층이 업데이트할 값을 결정하고, 식 2-4에 의해 tanh 층은 추가 할 수 있는 새로운 후보 값 $\tilde{C}_t$ 의 벡터를 생성하여 이 두 가지를 결합한다.

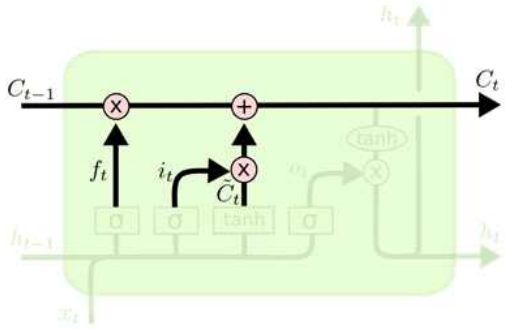


[그림 2-9] LSTM의 입력 게이트 (Christopher, 2015)

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \qquad \text{(식 2-3)}$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \qquad \text{(식 2-4)}$$

새로운 셀 상태 C_t는 식 2-5에 의해 이전 상태 C_{t-1}에 잊기로 결정한 망각 게이트의 출력 f_t를 곱한 값과 기억할 새로운 후보 값 $\tilde{C}_t$에 입력 게이트의 출력 i_t를 곱한 값을 더한다. C_t는 각 상태 값을 조정하여 업데이트하기로 결정한 새로운 값이다.

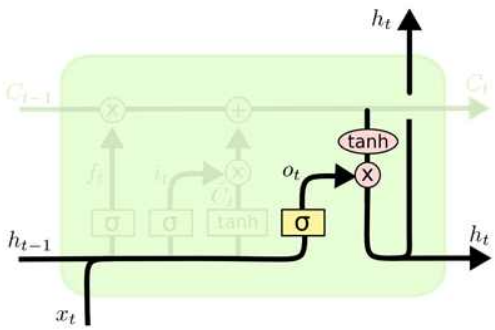


[그림 2-10] LSTM의 새로운 셀 상태 업데이트 (Christopher, 2015)

$$C_t = f_t * c_{t-X} + i_t * \tilde{C}_t \qquad \text{(식 2-5)}$$

마지막 단계는 출력할 내용을 결정하는 출력 게이트이다. 그림 2-11은

LSTM의 출력 게이트를 나타낸다. 먼저 식 2-6에 의해서 시그모이드 층을 통해 셀 상태의 어떤 부분을 출력할지 결정한다. 그런 다음 식 2-7에 의해서 tanh를 통해 셀 상태를 입력하고 출력 게이트의 출력값을 곱한다.



[그림 2-11] LSTM의 출력 게이트 (Christopher, 2015)

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad \text{(식 2-6)}$$

$$h_t = o_t * \tanh(C_t) \quad \text{(식 2-7)}$$

LSTM은 위와 같이 셀 상태 값을 얼마나 잊어버리고 새로운 입력값을 얼마나 받아들일지를 정할 수 있으므로 장기의존성 문제를 해결할 수 있다.

2.3 단어 임베딩

순환신경망 모델을 사용하여 음악 데이터를 학습시킬 때 컴퓨터가 이해할 수 있도록 음악 데이터를 적절히 변환해줘야 한다. 앞서 언급한 것처럼 인공신경망을 사용한 자동작곡 모델은 자연어처리 모델과 매우 흡사하므로 먼저 자연어처리에서 단어를 컴퓨터가 이해할 수 있는 숫자로 표현하는 방법을 살펴보고자 한다.

원-핫 인코딩(One-Hot Encoding)은 단어를 숫자로 표현하는 방법 중에서 가장 기본적인 방법이다. 먼저 학습 데이터에서 단어들을 사전으로 만들어 단어마다 인덱스들 부여한다. 원-핫 인코딩은 이렇게 인덱스가 부여된 단어

사전의 크기를 벡터의 차원으로 하고, 표현하고 싶은 단어의 인덱스에 1을 부여하고 나머지 인덱스에는 0을 부여하는 벡터 표현 방법이다. 이렇게 벡터 대부분의 값이 0으로 표현되는 방법을 희소 표현 이라 한다. 원-핫 벡터는 희소 벡터이다.

단어 사전에 개, 강아지, 소, 송아지가 있는 경우 원-핫 벡터 예는 표 2-1과 같다.

[표 2-1] 원-핫 벡터 예

단어 사전	인덱스	원-핫 벡터
개	1	[1 0 0 0]
강아지	2	[0 1 0 0]
소	3	[0 0 1 0]
송아지	4	[0 0 0 1]

원-핫 인코딩의 경우 단어의 개수가 늘어날수록 벡터 차원의 크기도 같이 커진다는 문제를 가지고 있다. 위 예에서는 단어 사전의 크기가 4이므로, 원-핫 벡터의 차원도 4이다. 하지만 단어 사전의 크기가 백만 개일 경우 원-핫 벡터의 차원 또한 백만이 된다. 이는 컴퓨터의 연산량 증가와 메모리 부족 현상을 초래하게 된다. 또한 원-핫 인코딩으로 표현된 벡터는 단어 간의 유사도, 즉 단어의 의미를 표현하지 못한다는 단점이 있다. 위 예에서는 개와 강아지가 유사하고 소와 송아지가 유사하다는 것을 표현할 수 없다.

이런 단점들을 해결한 방법으로 단어 임베딩(Word Embedding) 기법이 있다. 단어 임베딩은 단어를 밀집 벡터 형태로 표현한다. 밀집 표현은 벡터의 차원을 단어 사전의 크기로 정하지 않고 원하는 크기로 설정된 벡터의 차원에 맞춰 모든 단어를 표현한다. 예를 들어 10만 개의 단어가 있을 때 밀집 벡터 차원을 100으로 설정한다면 모든 단어는 실숫값을 가지는 100차원의 벡터로 표현된다. 개를 100차원의 밀집 벡터로 표현하면 다음과 같이 표시될 수 있다.

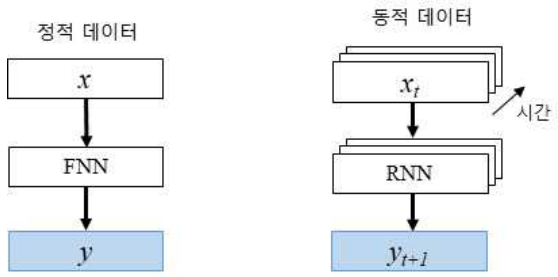
개(100차원 벡터) = [0.0015 0.0146 -0.0077 -0.0025 ... 중략 ...]

이처럼 단어 임베딩 방식은 작은 크기의 벡터로도 단어들을 효율적으로 표시할 수 있으며 또한 벡터 연산을 통해 단어 간 유사도뿐만 아니라 거리 측정과 같은 연산이 가능해지는 장점이 있다.

단어 임베딩 방법론으로는 Word2Vec[Mikolov, T., 등., 2013], FastText[Joulin, A., 등., 2017], Glove[Pennington, J., 등., 2014] 등이 있다. 본 논문에서는 케라스에서 제공하는 Embedding을 사용한다.

2.4 정적 데이터와 동적 데이터를 함께 학습시키는 방법

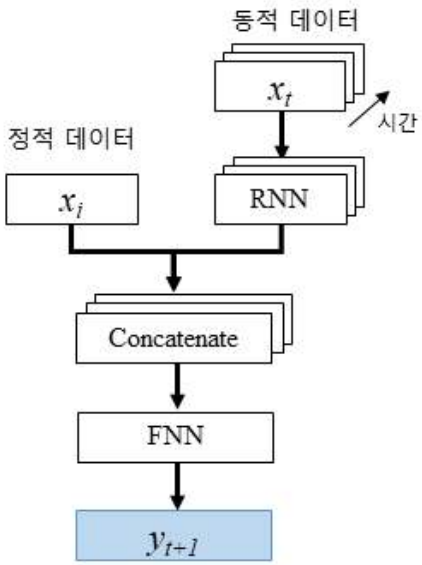
대부분의 인공신경망 알고리즘은 한 가지 유형의 데이터를 다루도록 설계되어 있으므로 학습 데이터의 특징이 순차적인지 아니면 정적인 지에 따라 어떤 모델을 사용할지를 결정하게 된다. 그러나 실제 상황에서는 데이터에 정적 특성과 동적 특성이 모두 존재하는 경우가 많다. 예를 들어 날씨는 같은 시간에도 지역마다 서로 다르다. 여기서 날씨 정보는 시간에 따라 변하는 동적 데이터이고 위치 정보는 그 지역의 정적 데이터이다. 음악에서도 같은 구분 방법을 적용하여 동적 데이터인 멜로디와 정적인 데이터인 곡 구성 정보로 구분할 수 있다. 그림 2-12는 일반적으로 정적 데이터를 다루는 피드포워드 신경망과 동적 데이터를 다루는 순환신경망을 나타내었다.



[그림 2-12] 정적 데이터를 다루는 모델과 동적 데이터를 다루는 모델

2.4.1 간접 입력 방법

다른 특징을 가지는 두 가지 데이터를 딥러닝 네트워크에 통합하는 두 가지 주요 접근방식이 있다. 첫 번째는 동적 데이터를 순환신경망에 공급한 다음 정적 데이터와 연결하는 간접 입력(Indirect input) 방식이다[Zhu, F., 등., 2018 : Leontjeva A., 등., 2016 : Lin C., 등., 2018 : Esteban C., 등., 2016]. 예를 들어 Lin C., 등., (2018)은 정적 정보를



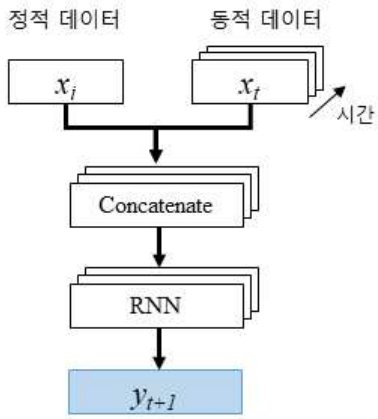
[그림 2-13] 간접 입력 방법

LSTM의 출력과 연결하여 패혈증 쇼크를 조기 진단하는 딥러닝 모델을 제안하였다[Wang, T., 등., 2019].

그림 2-13은 간접 입력 방법의 구조를 보여준다. x_i는 정적 데이터 입력, x_t는 t단계에서의 동적 데이터 입력, y_{t+1}은 $t+1$ 단계에서의 예측을 나타낸다.

2.4.2 직접 입력 방법

두 번째 접근방식은 정적 데이터와 동적 데이터를 순환신경망에 함께 입력하는 직접 입력(Direct input) 방식이다[Donahue J., 등., 2015 : Hsu TC., 등., 2019]. 간접 입력 방법과 비교할 때 직접 입력 방법은 순환신경망이 초기 단계에서 정적 데이터를 포함할 수 있다. 그러나 정적 데이터의 비 시간적 정보로 인해 시간 특징 정보가 오염되는 단점이 있다[Wang, T., 등., 2019].



[그림 2-14] 직접 입력 방법

그림 2-14는 직접 입력 방법의 구조를 보여준다. x_i는 정적 데이터 입력, x_t는 t단계에서의 동적 데이터 입력, y_{t+1}은 $t+1$ 단계에서의 예측을 나타낸다.

2.5 정량적 평가 방법

인공신경망에서 모델을 구현하는 것만큼 그 모델을 어떻게 평가할지도 매우 중요한 문제이다. 이번 절에서는 인공신경망을 통해 생성된 곡을 평가하는 방법으로 BLEU와 METEOR 알고리즘을 살펴본다.

2.5.1 BLEU

BLEU(BiLingual Evaluation Understudy)는 기계가 번역한 내용의 품질을 평가하는 대표적인 알고리즘이다[Papineni, K., 등., 2002]. 기계가 번역한 문장을 인간이 번역한 문장들과 비교하여 번역의 품질을 평가한다. 0과 1 사이의 결괏값을 가지며 1에 가까울수록 참조 문장과 더 유사하다는 것을 의미한다. n-gram을 기반으로 측정하며 언어에 구애받지 않고, 계산 속도가 빠르다는 장점이 있다.

예를 들어보자. 기계가 번역한 문장을 세 명의 사람이 번역한 문장과 비교하겠다. 기계가 번역한 문장을 Candidate이라고 하고 인간이 번역한 문장을 Reference라고 하겠다.

- Candidate: It is a guide to action which ensures that the military always obeys the commands of the party.
- Reference1: It is a guide to action that ensures that the military will forever heed Party commands.
- Reference2: It is the guiding principle which guarantees the military forces always being under the command of the Party.
- Reference3: It is the practical guide for the army always to heed the directions of the party.

Candidate를 Reference와 비교하여 성능을 측정할 때 유니그램 정확도(Unigram precision)를 사용한다. Candidate의 단어 중에 Reference에 등장한 단어의 개수를 센 후 Candidate의 총 단어 수로 나눠준다. 식 2-8은 유니그램 정확도를 나타낸다. N은 Reference에 존재하는 Candidate의 단어 수를 나타내고 W_t는 Candidate를 구성하는 총 단어 수를 말한다. 위 예에서

Candidate의 유니그램 정확도를 계산하면 $\frac{17}{18}$이 된다.

$$Unigram\,Precision = \frac{N}{W_t} \qquad \text{(식 2-8)}$$

두 번째 예를 들어보자.

- Candidate: the the the the the the the
- Reference1: the cat is on the mat
- Reference2: there is a cat on the mat

두 번째 예에서는 Candidate은 'the'만으로 되어 있지만 유니그램 정확도는 $\frac{7}{7} = 1$ 이라는 최고의 평가를 받게 된다. 그러나 이 경우 제대로 번역된 문장이라고 말할 수 없다. 따라서 같은 단어가 반복적으로 나올 때는 이를 보정해 주어야 할 필요가 생긴다. 그러기 위해서 Reference에서 존재하는 Candidate의 단어를 세는 과정에서 Candidate의 유니그램이 이미 Reference에 존재한 적이 있는지를 고려해야 한다. 식 2-9는 수정된 유니그램 정확도를 나타내며 이를 클리핑(clipping)이라 한다. 여기서 $Candidate_{count}$은 Candidate의 단어 수를, $m_{\max}$는 Candidate를 구성하고 있는 단어들과 Reference를 구성하고 있는 단어들이 겹치는 수의 최댓값을, $Count_{clip}$은 $Candidate_{count}$와 $m_{\max}$의 최솟값을, W_t는 Candidate를 구성하는 총 단어 수를 말한다.

$$Unigram\,Precision = \frac{Count_{clip}}{W_t} \qquad \text{(식 2-9)}$$

$$Count_{clip} = \min(Candidate_{count}, m_{\max}) \qquad \text{(식 2-10)}$$

두 번째 예에서 'the'가 7개이므로 $Candidate_{count}$는 7, Candidate을 구성하고 있는 단어는 'the' 1개이고 Reference1에서 'the'가 최대 2개이므로 $m_{\max}$는 2, $Count_{clip}$ 은 $\min(7,2)$ 이므로 2, W_t는 7이다. 따라서 유니그램 정확도는 $\frac{2}{7}$가 된다.

세 번째 예를 들어보자.

- Candidate1: the the the the the the the
- Candidate2: the cat the cat on the mat
- Reference1: the cat is on the mat
- Reference2: there is a cat on the mat

유니그램 정확도는 단어의 순서를 고려하지 않는다는 단점이 있다. 번역 모델에서 일치하는 단어의 빈도수뿐만 아니라 단어의 순서도 중요하다. 세 번째 예는 두 번째 예에 Candidate2를 추가하였다. Candidate2의 바이그램 $Count$와 $Count_{clip}$을 세면 표 2-2와 같다. Candidate2의 바이그램 정확도는 $\frac{4}{6}$이고 Candidate1의 경우 'the the'가 Reference에 없으므로 바이그램 정확도는 0이 된다.

[표 2-2] Candidate2의 바이그램 $Count$와 $Count_{clip}$

바이그램	the cat	cat the	cat on	on the	the mat	합계
$Count_{clip}$	1	0	1	1	1	4
$Count$	2	1	1	1	1	6

바이그램 정확도를 n-gram으로 확장하여 정확도를 표현하면 BLEU의 식은 식 2-11이 된다. p_n은 각 gram의 보정된 정확도를 나타낸다. N은 n의 최대 숫자이다. 보통 4의 값을 가진다. w_n은 각 gram의 보정된 정확도의 가

중치를 말한다. 예를 들어 N이 4인 경우 p_1, p_2, p_3, p_4 에 대해서 동일한 가중치를 사용하면 0.25를 적용할 수 있다.

$$BLEU = \exp(\sum_{n=1}^{N} w_n \log p_n) \qquad \text{(식 2-11)}$$

마지막 예를 들어보자.

- Candidate: the cat
- Reference1: the cat is on the mat
- Reference2: there is a cat on the mat

마지막 예에서 유니그램 정확도를 계산하면 Candidate에 단어가 'the'와 'cat'이 있으므로 $\frac{1}{2}$ + $\frac{1}{2}$ = 1이 되어 최고의 평가를 받게 된다. 이 경우 역시 제대로 번역된 문장이라고 보기 어렵다. 따라서 짧은 문장이 생성될 때도 정확도를 보정해 주어야 한다. 이 보정을 위해 유니그램 정확도에 브레버티 패널티(Brevity Penalty, BP)를 곱해준다. 식 2-12는 브레버티 패널티를 나타낸다. 여기서 c는 Candidate의 길이, r은 Candidate와 가장 길이 차이가 작은 Reference의 길이를 말한다.

$$BP = \begin{cases} 1 & \text{if } c > r \\ \exp^{(1-\frac{c}{r})} & \text{if } c \leq r \end{cases} \qquad \text{(식 2-12)}$$

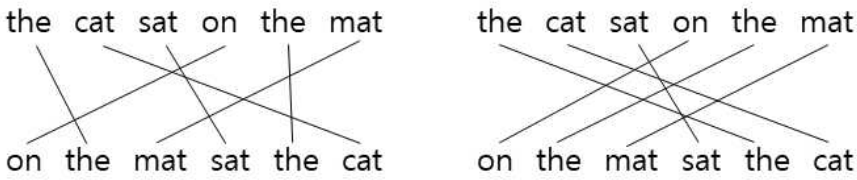
브레버티 패널티를 적용한 최종 BLEU 식은 식 2-13과 같다.

$$BLEU = BP \times \exp(\sum_{n=1}^{N} w_n \log p_n) \qquad \text{(식 2-13)}$$

2.5.2 METEOR

METEOR (Metric for Evaluation of Translation with Explicit ORdering)은 BLEU와 마찬가지로 문장을 기본 단위로 하는 기계가 번역한 내용을 평가하기 위한 알고리즘이다[Banerjee, S., 등., 2005].

기계가 번역한 문장을 Hypothesis이라고 하고 인간이 번역한 문장을 Reference라고 하겠다. METEOR은 먼저 Hypothesis과 Reference 사이에 서로 일치하는 유니그램끼리 연결한 정렬을 만든다. 이때 Hypothesis의 유니그램은 Reference에 최대 1개의 유니그램과 연결될 수 있다. 이렇게 생성된 연결 정렬 중에 연결의 교차가 가장 적은 정렬이 선택된다. 예를 들어 그림 2-15에 표시된 2개의 정렬 중에서는 왼쪽 정렬이 선택된다.



[그림 2-15] METEOR 유니그램 정렬 예

선택된 정렬의 유니그램 정확도(Precision) P는 식 2-14와 같다. m은 Reference에서 발견되는 Hypothesis의 단어 수이고 w_t는 Hypothesis의 단어 수이다.

$$P = \frac{m}{w_t} \qquad \text{(식 2-14)}$$

유니그램 재현율(Recall) R은 식 2-15와 같다. m은 위와 같고 w_r은 Reference의 단어 수이다.

$$R = \frac{m}{w_r} \quad \text{(식 2-15)}$$

정확도와 재현율은 식 2-16과 같이 조화 평균을 사용하여 결합한다. 이때 재현율은 정확도보다 9배의 가중치를 준다.

$$F_{mean} = \frac{10PR}{R+9P} \quad \text{(식 2-16)}$$

유니그램을 n-gram으로 확장해보자. 일치하는 n-gram을 Reference와 Hypothesis 정렬에 사용하는 경우 패널티를 계산해 준다. 이때 Reference에서 Hypothesis와 일치하는 연속된 유니그램 세트, 즉 연결된 단어 묶음을 청크라 한다. Reference와 Hypothesis 사이에 청크 수가 더 적을수록(즉 단어의 긴 연결이 많을수록) 패널티는 낮아진다. 패널티 p는 식 2-17과 같다. 여기서 c는 청크 수이고 u_m은 정렬된 유니그램의 수이다.

$$p = 0.5(\frac{c}{u_m})^3 \quad \text{(식 2-17)}$$

패널티를 적용한 최종 METEOR을 식 2-18에 나타내었다.

$$M = F_{mean}(1-p) \quad \text{(식 2-18)}$$

Ⅲ. 구성을 갖춘 자동작곡 시스템

3.1 데이터 전처리

음악 데이터를 인공신경망에서 사용하기 위해서는 곡을 컴퓨터가 이해할 수 있는 문자 형태로 변경해 주어야 한다. 본 논문에서는 미디 파일의 곡 정보를 부호를 사용하여 텍스트로 변환하는 방식을 사용하였다.

3.1.1 미디 데이터

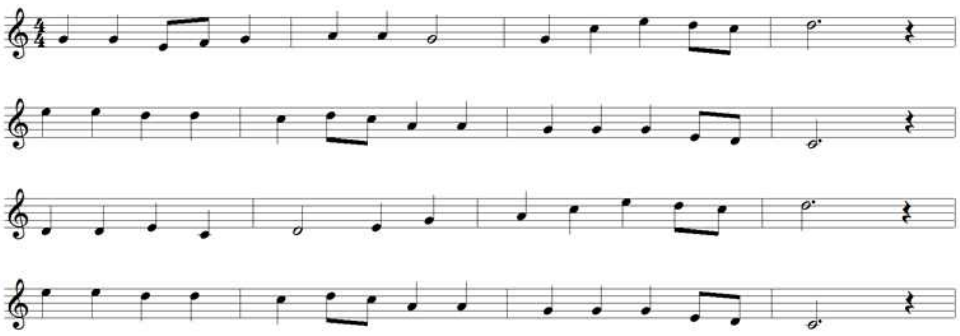
미디(MIDI, Musical Instrument Digital Interface)는 전자악기의 디지털 데이터를 주고받기 위한 표준 규격이다. 본 논문에서는 music21 파이썬 라이브러리를 사용하여 미디 파일 내의 음표, 쉼표, 음표 또는 쉼표의 길이, 박자 등의 정보를 추출하여 텍스트로 변환하는 방식을 사용하였다. 이 중 음표의 높낮이 정보는 music21에서 정의한 숫자 표기 방식을 따랐다. music21 라이브러리의 pitch.Pitch.midi 객체를 사용하면 음의 높낮이를 고정된 숫자로 나타낼 수 있다. 예를 들어 4옥타브 도(가온 다)는 숫자 60, 5옥타브 미는 숫자 76으로 표시된다. 쉼표의 경우에는 rest를 의미하는 'r'로 나타내었고 높낮이를 가지고 있지 않으므로 길이만 표시 하였다. 더불어 노래마다 다른 조성을 가지고 있으므로 조성에 따른 차이를 제거하기 위해서 모든 곡을 장조의 경우에는 다장조로, 단조의 경우에는 가단조로 조 옮김을 한 후 변환하였다.

표 3-1에 본 논문에서 사용된 미디 파일 내의 곡 정보를 추출하여 인공신경망 학습을 위해서 텍스트 형식으로 변환에 사용된 기호들을 나타내었다.

[표 3-1] 음악 기호를 텍스트로 변환한 기호

음악 기호	텍스트 기호
음표	숫자 (가온 다는 60)
쉼표	r
박자표	2/4, 1/3, 4/4 등
음표(쉼표)의 길이	숫자 (1.0은 한 박자)
음표(쉼표)의 길이 구분	_
음표(쉼표)와 음표(쉼표) 구분	,
마디 구분	공백문자
박자와 음표(쉼표) 구분	|

그림 3-1은 미디 파일을 표 3-1의 텍스트 기호를 사용하여 변환한 예를 보여준다.

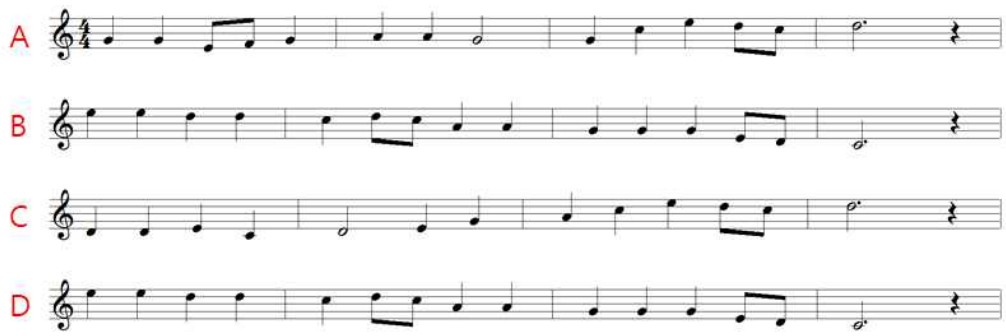


4/4|67_1.0,67_1.0,64_0.5,65_0.5,67_1.0 4/4|69_1.0,69_1.0,67_2.0 4/4|67_1.0,72_1.0,
76_1.0,74_0.5,72_0.5 4/4|74_3.0,r_1.0 4/4|76_1.0,76_1.0,74_1.0,74_1.0 4/4|72_1.0,
74_0.5,72_0.5,69_1.0,69_1.0 4/4|67_1.0,67_1.0,67_1.0,64_0.5,62_0.5 4/4|60_3.0,r_1.
0 4/4|62_1.0,62_1.0,64_1.0,60_1.0 4/4|62_2.0,64_1.0,67_1.0 4/4|67_1.0,72_1.0,76_
1.0,74_0.5,72_0.5 4/4|74_3.0,r_1.0 4/4|76_1.0,76_1.0,74_1.0,74_1.0 4/4|72_1.0,74_
0.5,72_0.5,69_1.0,69_1.0 4/4|67_1.0,67_1.0,67_1.0,64_0.5,62_0.5 4/4|60_3.0,r_1.0

[그림 3-1] 악보를 텍스트로 변환한 예 (고향의 봄)

3.1.2 곡 구성 정보

2.4절에서 언급한 바와 같이 음악은 멜로디 정보인 동적 데이터와 곡 구성 정보인 정적 데이터로 나눌 수 있다. 곡 구성 정보는 미디 파일에 포함되어 있지 않으므로 추가로 작업을 해주어야 한다. 본 논문에서는 곡 구성 정보를 넣어주기 위해서 비교적 쉬운 '기승전결'의 구조를 가지는 동요 300곡을 사용하였다. 300곡을 수작업으로 '기승전결'로 구분하여 각 파트를 A, B, C, D로 표시하였다. 일부 '기결(AD)' 또는 '기승결(ABD)'의 구조를 가지는 곡들도 포함되어 있다. 그림 3-2는 '고향의 봄'을 기승전결(A, B, C, D) 구조로 나누어 곡 구성 정보를 추가하여 미디 파일을 텍스트로 변환한 예를 보여준다.



A4/4|67_1.0,67_1.0,64_0.5,65_0.5,67_1.0 A4/4|69_1.0,69_1.0,67_2.0 A4/4|67_1.0,72_1.0,
76_1.0,74_0.5,72_0.5 A4/4|74_3.0,r_1.0 B4/4|76_1.0,76_1.0,74_1.0,74_1.0 B4/4|72_1.0,
74_0.5,72_0.5,69_1.0,69_1.0 B4/4|67_1.0,67_1.0,67_1.0,64_0.5,62_0.5 B4/4|60_3.0,r_1.
0 C4/4|62_1.0,62_1.0,64_1.0,60_1.0 C4/4|62_2.0,64_1.0,67_1.0 C4/4|67_1.0,72_1.0,76_
1.0,74_0.5,72_0.5 C4/4|74_3.0,r_1.0 D4/4|76_1.0,76_1.0,74_1.0,74_1.0 D4/4|72_1.0,74_
0.5,72_0.5,69_1.0,69_1.0 D4/4|67_1.0,67_1.0,67_1.0,64_0.5,62_0.5 D4/4|60_3.0,r_1.0

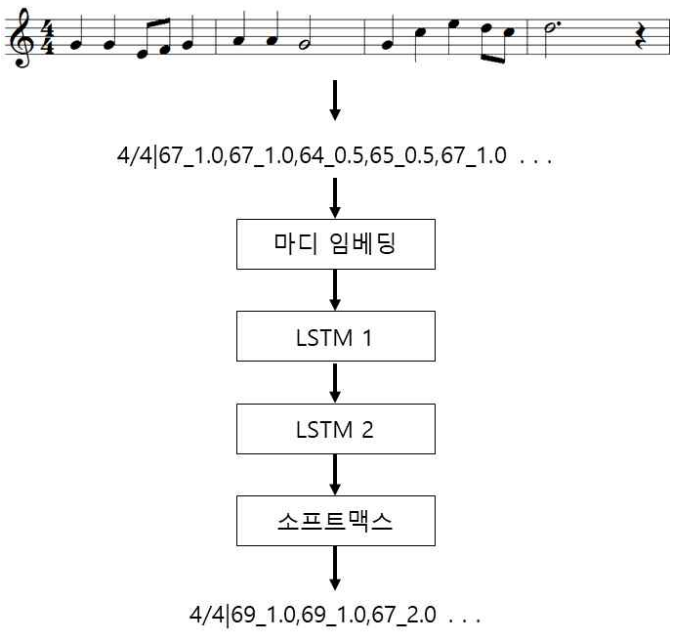
[그림 3-2] 곡 구성 정보를 추가한 음악 데이터 (고향의 봄)

3.2 자동작곡 시스템

이번 절에서는 인공신경망을 사용하여 작곡하는 시스템으로 3가지 모델을 제안한다.

3.2.1 멜로디 학습 모델

첫 번째 모델은 자연어처리에서 자주 사용되는 순환신경망을 사용한 워드 임베딩 모델이다. 악보 데이터는 3.1.1 절에서 설명한 데이터 전처리 방식을 사용하여 텍스트 형태로 변환된다. 변환된 텍스트 데이터는 마디 단위로 나뉘어 모델에 입력으로 들어간다. 입력된 마디는 마디 임베딩 층을 거쳐 벡터화 된다. 순환신경망으로는 LSTM층 2개를 사용하였다. 출력단에서는 소프트맥스 층을 거쳐 최종적으로 모델이 예측한 값이 생성된다. 입력한 정답과 비교

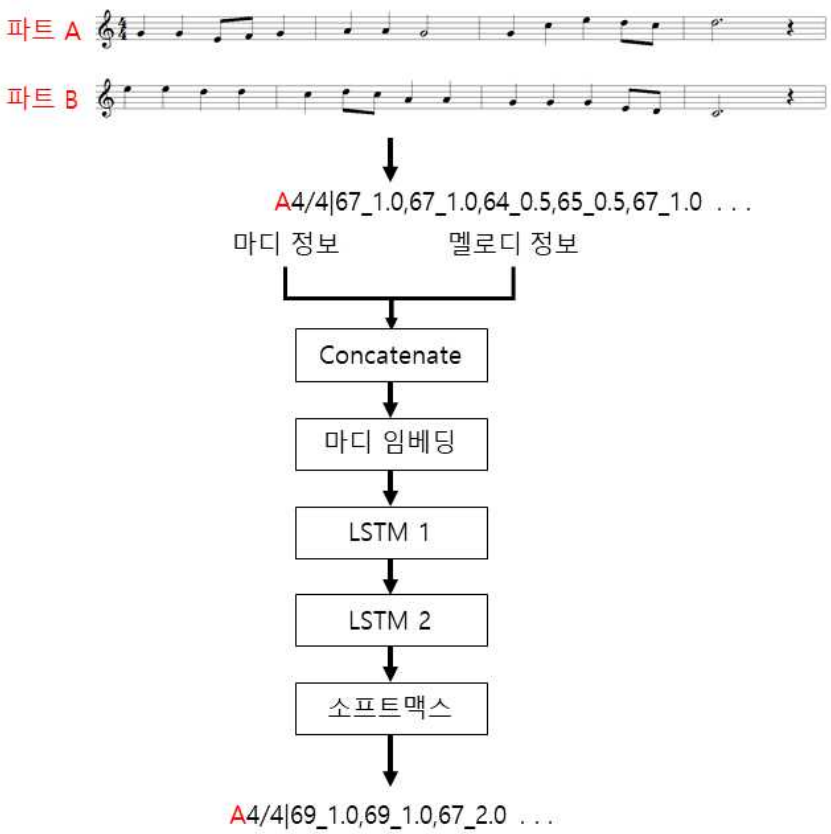


[그림 3-3] 마디 임베딩을 사용한 멜로디 학습 모델

하여 손실을 계산하는 손실 함수로는 교차 엔트로피 오차(Cross Entropy Error) 함수를 사용하였다. 그림 3-3은 마디 임베딩을 사용한 멜로디 학습 모델의 전체 구조를 보여준다.

3.2.2 곡 구성 정보 직접 입력 모델

두 번째 모델은 멜로디 정보에 곡 생성을 위한 보조 데이터인 곡 구성 정보를 결합하여 순환신경망에 대한 단일 입력으로 넣어주는 방법이다. 악보 데이터는 3.1.2 절에서 설명한 데이터 전처리 방식을 사용하여 텍스트 형태로 변환된다. 변환된 텍스트 데이터로부터 파트 정보와 멜로디 정보를 분리하여

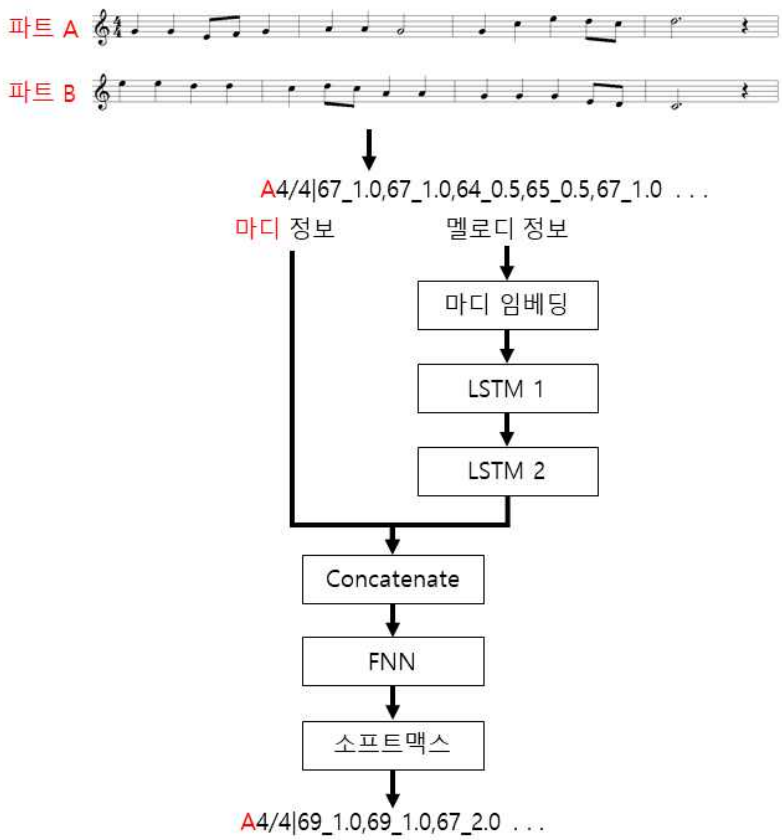


[그림 3-4] 곡 구성 정보 직접 입력 모델

결합 층에 입력으로 넣어준다. 결합 층을 통과한 데이터는 마디 임베딩 층을 거쳐 벡터화된다. 순환신경망과 출력층은 멜로디 학습 모델과 동일하게 LSTM 층 2개와 소프트맥스 층을 사용하였다. 손실 함수도 같은 교차 엔트로피 오차 함수를 사용하였다. 그림 3-4는 곡 구성 정보 직접 입력 모델의 전체 구조를 보여준다.

3.2.3 곡 구성 정보 간접 입력 모델

마지막 모델은 곡 구성 정보를 별도로 사용하는 방식이다. 곡 구성 정보 직접 입력 모델과 동일하게 전처리 된 텍스트 데이터에서 멜로디 정보는



[그림 3-5] 곡 구성 정보 간접 입력 모델

마디 임베딩 층을 거쳐 벡터화되고 2중 LSTM 층에 전달된다. LSTM 층을 통과한 데이터는 텍스트 데이터에서 분리된 파트 정보와 결합 된 후 피드포워드 신경망(FNN)을 거쳐 출력된다. 출력층과 손실 함수는 이전 모델들과 같게 소프트맥스 층과 교차 엔트로피 오차 함수를 사용하였다. 그림 3-5는 곡 구성 정보 간접 입력 모델의 전체 구조를 보여준다.

Ⅳ. 실험 결과

4장에서는 3장에서 제안한 각 모델을 사용하여 곡을 생성하고 분석하였다. 곡 구성 정보를 함께 학습한 모델에서는 곡을 생성할 때도 A-B-C-D의 파트 정보를 함께 입력으로 넣어주어 곡을 생성하도록 하였다. 이렇게 생성된 곡들을 멜로디만 학습하여 생성된 곡들과 비교하였다. 일반적으로 동요는 16마디로 되어 있는 경우가 많으므로 모든 곡은 16마디로 생성하였다. 또한 학습한 적이 없는 D-C-B-A와 같은 순서의 곡 구성을 입력으로 주어 원하는 구성을 가지는 곡이 생성되는지를 실험하였다. 마지막으로 METEOR과 BLEU 점수를 사용하여 작곡가가 작곡한 곡과의 유사도를 모델별로 평가하여 보았다.

4.1 실험 데이터

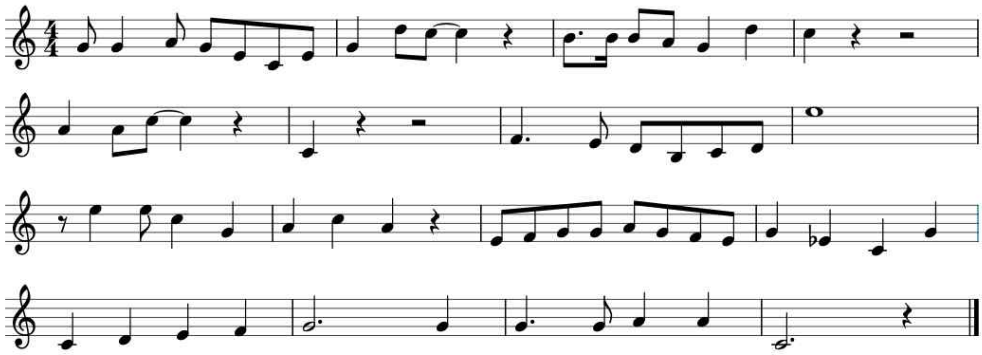
실험 데이터로는 동요 300곡이 사용되었다. 학습 훈련 데이터로 80%(240곡, 2697마디), 평가용 테스트 데이터로 20%(60곡)를 사용하였다.

Keras에서 실행한 학습 조건은 다음과 같다.

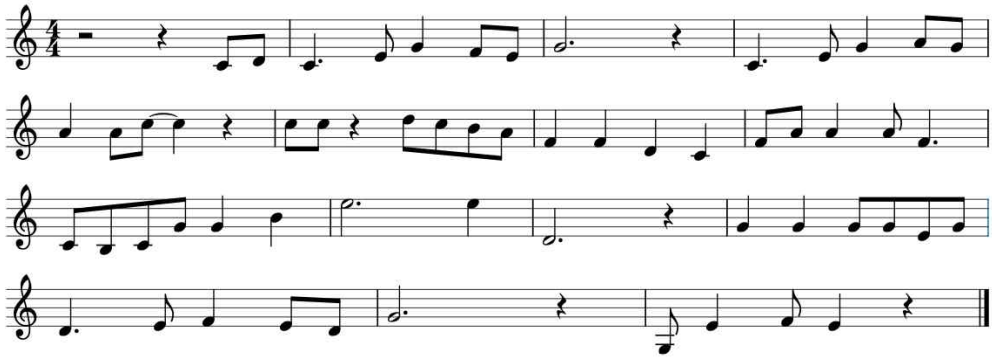
- 옵티마이저 : RMSprop
- 손실 함수 : Cross Entropy Error
- 에폭 : 500
- 학습률 : 0.001
- early stopping : True
- patience : 10
- validation_split : 0.2
- batch size : 128
- embedding size : 100

4.2 멜로디 학습 모델

그림 4-1은 멜로디 학습 모델로 생성한 곡이다. 첫 마디를 입력으로 주고 나머지 마디를 생성하도록 하였다. 멜로디 학습 모델에 사용된 데이터의 경우 미디 파일을 텍스트로 변환 시 곡 구성 정보를 가지고 있지 않으므로 생성된 마디들이 정확히 어느 파트에서 왔는지 알기 어려운 부분이 있다. 곡 구성 정보가 있는 데이터에서 검색하였을 때 대략 [AAB?[1]C??DDDAAACC?] 의 구성을 하고 있었다.



[그림 4-1] 멜로디 학습 모델로 생성한 곡 1



[그림 4-2] 멜로디 학습 모델로 생성한 곡 2

1) '?' 는 해당 마디가 A, B, C, D 파트 중에서 두 개 이상에서 검색된 경우이다.

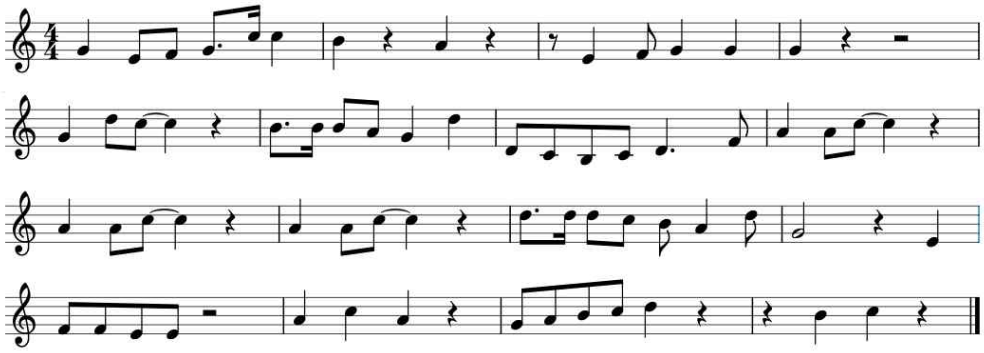
그림 4-2의 두 번째로 생성된 곡은 [AAC?DC?C?B?AAB??] 구성을 하고 있었다. 두 곡 모두 곡 구성이 되어 있지 않음을 볼 수 있다.

4.3 곡 구성 정보 직접 입력 모델

이번 절에서는 직접 입력(Direct input) 모델을 사용하여 멜로디와 곡 구성 정보를 순환신경망에 함께 입력하여 의도한 구성을 가진 곡이 생성되는지를 실험하였다.

4.3.1 AAAABBBBCCCCDDDD 형식으로 작곡

그림 4-3는 곡 구성 정보 직접 입력 모델을 사용하여 생성한 곡이다. 곡 생성 시 곡 구성 정보로 동요에서 가장 많이 사용되는 형식인 기승전결의 [AAAABBBBCCCCDDDD] 형식을 멜로디와 결합하여(Concatenate) 순환신경망에 입력하였다.

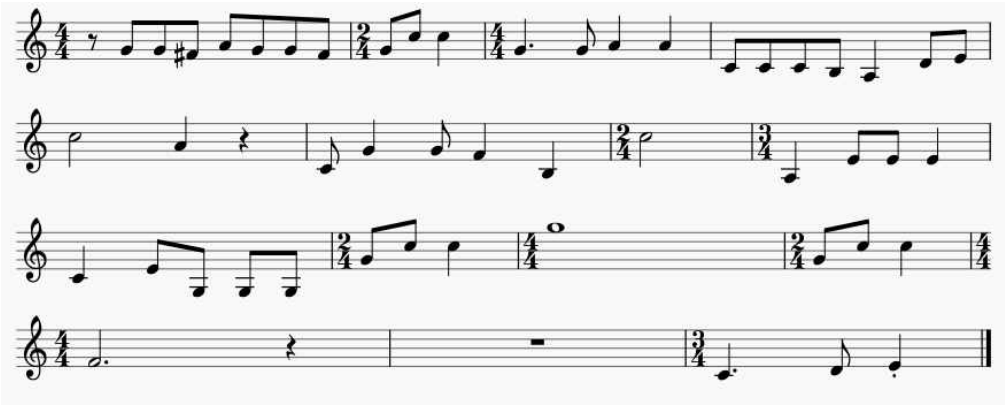


[그림 4-3] 곡 구성 정보 직접 입력 모델로 생성한 곡 1

생성된 각 마디는 실제로 [AAADBBBBCCCCDDDD] 형식으로 구성되어 있었다. 생성을 위해 입력해준 곡 구성 정보와 거의 동일한 구성을 하고 있음을 볼 수 있다.

4.3.2 AABBCCDDAABBCCDD 형식으로 작곡

그림 4-4는 [AABBCCDDAABBCCDD] 형식을 입력하여 빠르게 기승전결이 반복되도록 작곡한 곡이다. 생성된 곡은 [AACACBBCCDDD〈EOS〉[2] AA] 형식으로 되어 있었다. 어느 정도 입력 구성을 따라가는 것을 볼 수 있다. 그러나 악보상 변박자가 심하게 이루어지는 것을 볼 수 있다.

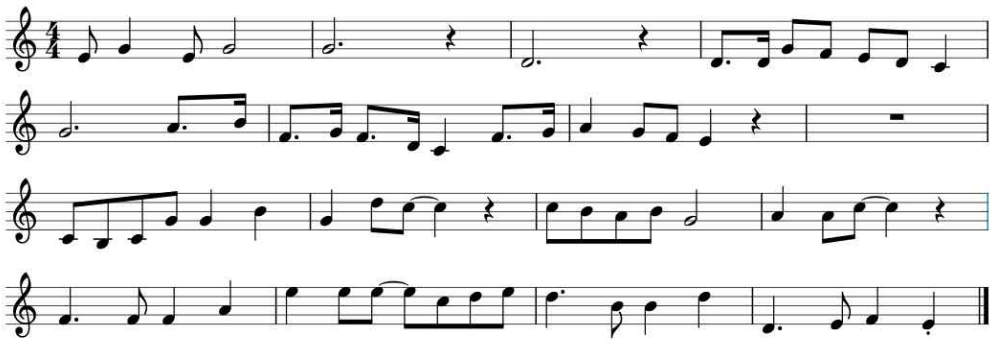


[그림 4-4] 곡 구성 정보 직접 입력 모델로 생성한 곡 2

4.3.3 AAAAAAAABBBBBBBB 형식으로 작곡

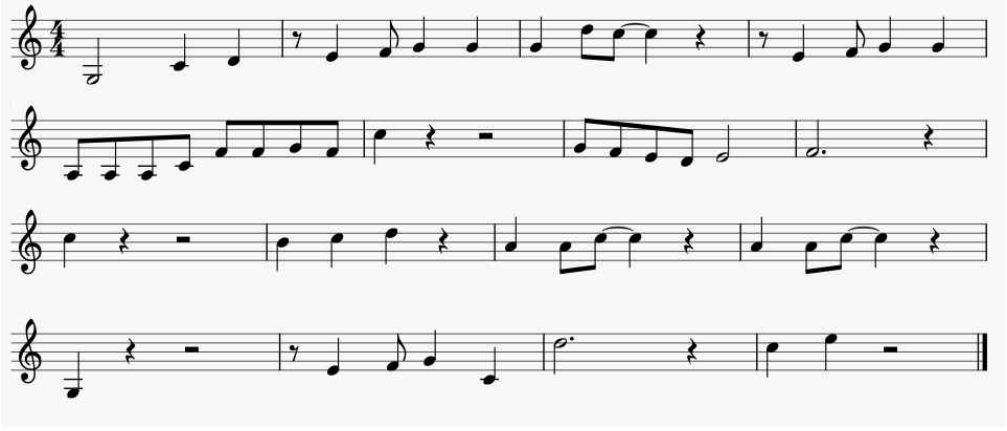
그림 4-5는 [AAAAAAAABBBBBBBB] 형식을 입력하여 구성이 느리게 변하도록 작곡한 곡이다. 생성된 곡은 [AAAAAABB〈EOS〉DBBCCDB] 구성을 하고 있다. 뒷부분으로 갈수록 생성된 곡의 구성이 입력된 구성과 달라짐을 볼 수 있다. 이는 순환신경망의 특성상 시간이 흐를수록 입력된 정보가 약해지는 현상으로 설명된다.

2) <EOS>는 End of Song의 약자로 곡의 끝부분을 나타낸다. 자연어처리에서 End of Sentence와 같은 구분자 역할을 한다.



[그림 4-5] 곡 구성 정보 직접 입력 모델로 생성한 곡 3

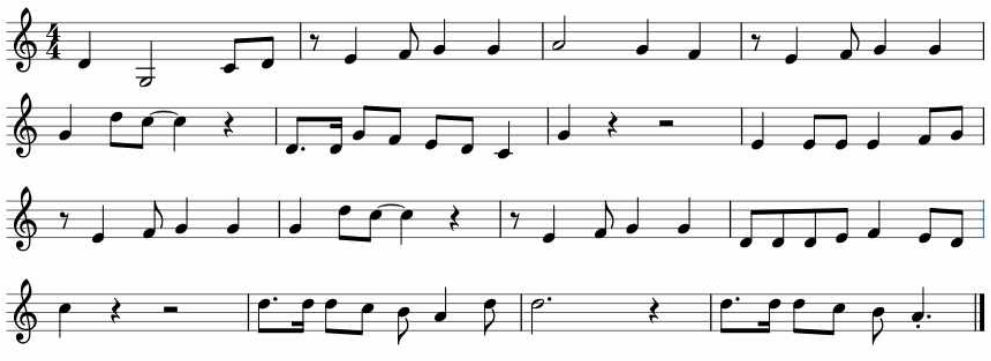
4.3.4 AAAAAAAAAAAAAAAA 형식으로 작곡



[그림 4-6] 곡 구성 정보 직접 입력 모델로 생성한 곡 4

그림 4-6은 [AAAAAAAAAAAAAAAA] 형식을 입력하여 구성이 변하지 않도록 작곡한 곡이다. 생성된 곡은 [AAAAABBBBBADCBD] 구성을 가지고 있었다. 이 역시 시간이 흐를수록 입력된 마디 정보가 약해지는 현상을 보이고 있다.

4.3.5 DDDDCCCCBBBBAAAA 형식으로 작곡



[그림 4-7] 곡 구성 정보 직접 입력 모델로 생성한 곡 5

마지막으로 일반적이지 않은 구성을 입력하였을 때 어떤 곡이 출력되는지를 실험하기 위해 일반적이지 않은 [DDDDCCCCBBBBAAAA] 형식을 입력하였다. 그림 4-7은 이 같은 형식을 입력하여 생성한 곡을 보여준다. 입력된 형식으로 곡을 생성하지 못하고 [DCBBAAAACBBBCCAC] 형식을 가지고 있었다. 단방향 순환신경망을 사용하였기 때문에 역방향 구성을 가지는 곡을 생성하지 못하고 있다.

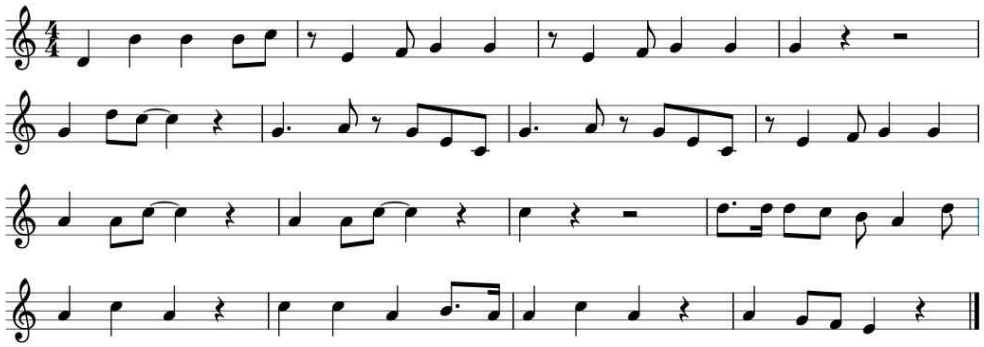
4.4 곡 구성 정보 간접 입력 모델

4절에서는 간접 입력(Indirect input) 모델을 사용하였다. 순환신경망을 통과한 멜로디 데이터를 곡 구성 정보와 합쳐 의도한 구성을 가진 곡이 생성되는지를 실험하였다.

4.4.1 AAAABBBBCCCCDDDD 형식으로 작곡

그림 4-8은 곡 구성 정보 간접 입력 모델을 사용하여 생성한 곡이다. 직접 입력 모델과 마찬가지로, 생성 시 곡 구성 정보로 동요 형식인

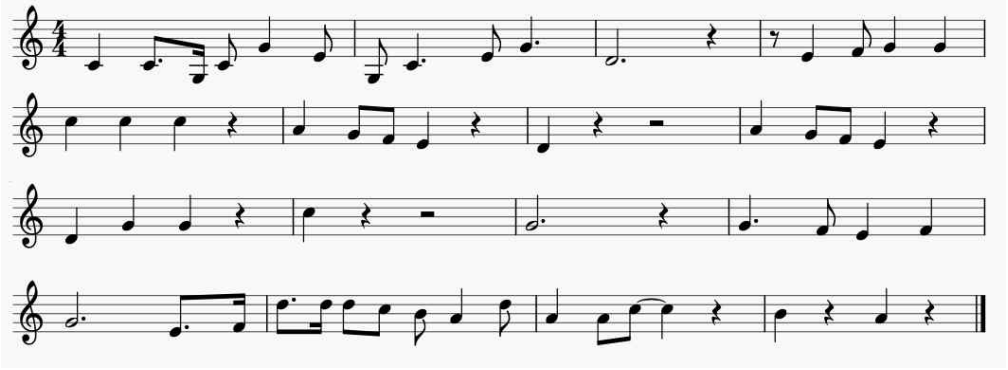
[AAAABBBBCCCCDDDD] 형식을 입력하였다. 멜로디 데이터는 순환신경망을 통과한 다음 [AAAABBBBCCCCDDDD] 형식과 결합(Concatenate) 된다.



[그림 4-8] 곡 구성 정보 간접 입력 모델로 생성한 곡 1

생성된 각 마디는 실제로 [AAAABBBBCCCCDDDD] 형식으로 되어 있었다. 입력된 곡 구성 정보와 동일한 구성을 하고 있음을 볼 수 있다.

4.4.2 AABBCCDDAABBCCDD 형식으로 작곡

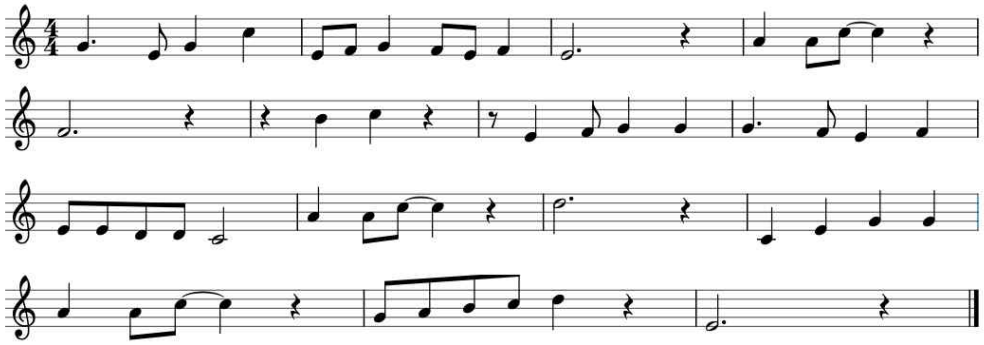


[그림 4-9] 곡 구성 정보 간접 입력 모델로 생성한 곡 2

그림 4-9는 [AABBCCDDAABBCCDD] 형식을 입력하여 기승전결이 반복되도록 작곡한 곡이다. 생성된 곡은 [AACAAABB〈EOS〉ACDDDBD] 형식

으로 되어 있었다. 곡 구성이 원하는 대로 만들어지지 않았음을 볼 수 있다.

4.4.3 AAAAAAAABBBBBBBB 형식으로 작곡

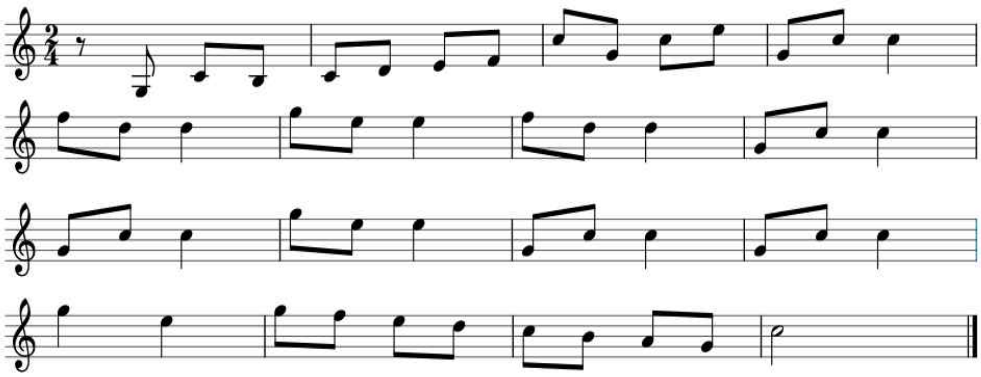


[그림 4-10] 곡 구성 정보 간접 입력 모델로 생성한 곡 3

그림 4-10은 [AAAAAAAABBBBBBBB] 형식을 입력하여 느리게 구성이 변하도록 작곡한 곡이다. 생성된 곡은 [AAAAA〈EOS〉BAACCCDBDD] 구성을 하고 있었다. 직접 입력 모델과 마찬가지로 뒷부분으로 갈수록 생성된 곡의 구성이 입력된 구성과 달라짐을 볼 수 있다.

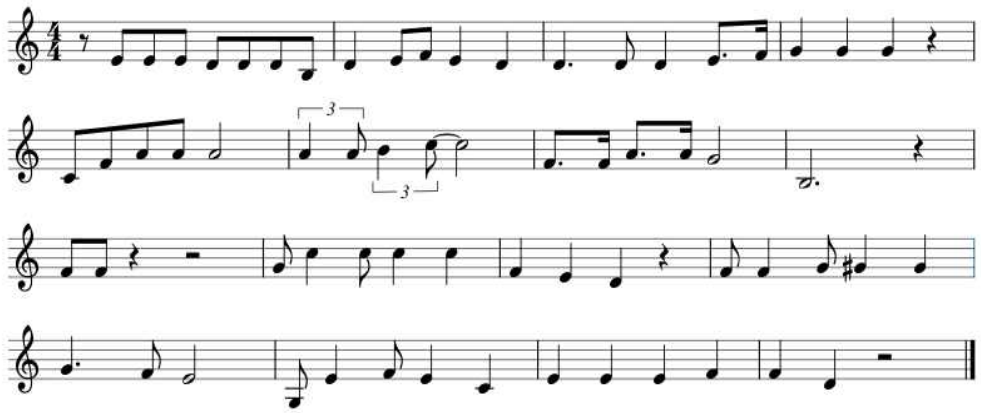
4.4.4 AAAAAAAAAAAAAAAA 형식으로 작곡

그림 4-11은 [AAAAAAAAAAAAAAAA] 형식을 입력하여 구성이 변하지 않도록 작곡한 곡이다. 생성된 곡은 [AAABABBCCCCCDCC] 구성을 가지고 있었다. 이 역시 시간이 흐를수록 입력된 마디 정보가 약해지는 현상을 보이고 있다.



[그림 4-11] 곡 구성 정보 간접 입력 모델로 생성한 곡 4

4.4.5 DDDDCCCCBBBBAAAA 형식으로 작곡



[그림 4-12] 곡 구성 정보 간접 입력 모델로 생성한 곡 5

간접 입력 모델의 곡 생성 마지막으로 [DDDDCCCCBBBBAAAA] 형식을 입력하였다. 그림 4-12는 이 구성으로 생성된 곡을 보여준다. 생성된 곡은 [DCBBABADADDDDCCD] 구성을 하고 있었다. 특별한 형식을 가지고 있지 않다고 볼 수 있다.

4.5 곡 평가

자동작곡 모델의 품질을 평가하는 데 있어서 중요한 것 중 하나는, 생성된 음악이 궁극적으로 얼마나 사람들이 듣기에 좋은가 이다. 음악 평가는 주관적이고 사람에 따라 다르므로 정확한 평가가 어렵다는 문제가 있다. 이를 해결하기 위해 자연어처리 모델에서 사용되는 METEOR과 BLEU와 점수를 사용하여 인공신경망에 의해 생성된 곡이 작곡가가 작곡한 곡과 얼마나 유사한지를 평가하였다. METEOR과 BLEU 점수는 모두 값이 1에 가까울수록 성능이 우수함을 나타낸다. BLEU 점수 측정 시에 n-gram의 수는 4로 기본 가중치 (0.25, 0.25, 0.25, 0.25)를 사용하였다.

4.5.1 모델별 평가

곡 학습과 생성에 사용된 세 가지 모델을 비교하기 위하여 모델별로 [AAAABBBBCCCCDDDD] 구성으로 100곡씩 생성하여 METEOR과 BLEU 점수를 측정하였다.

그림 4-13과 표 4-1은 각 학습 모델이 생성한 100곡에 대한 METEOR 점수의 평균을 나타낸다. 간접 입력 모델이 가장 높은 점수를 보여주고 있다.

[그림 4-13] 각 학습 모델별 METEOR 점수 그래프

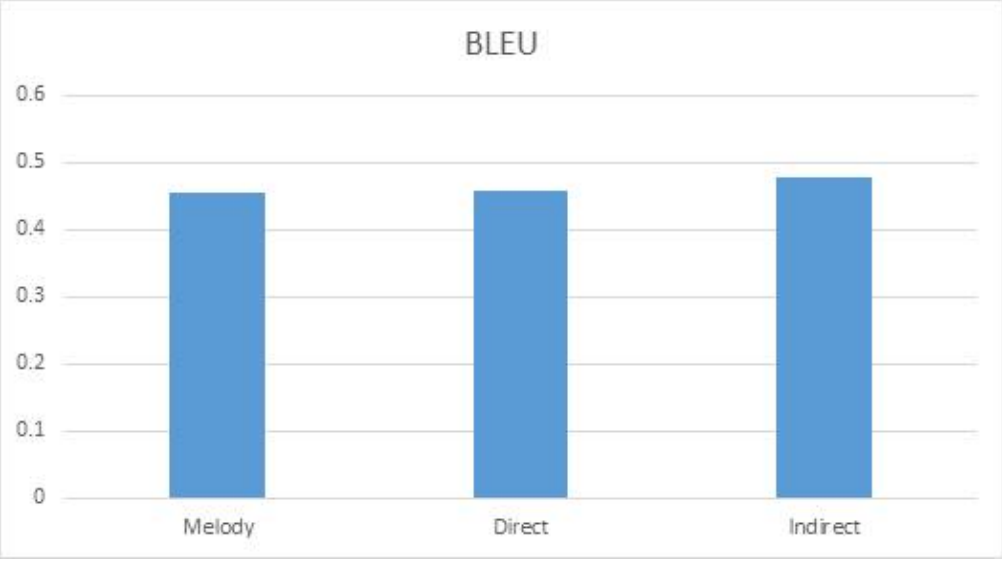
[표 4-1] 각 학습 모델로 생성한 100곡의 METEOR 평균 점수

	멜로디 학습 모델	직접 입력 모델	간접 입력 모델
METEOR 점수	0.257682942	0.365766918	0.388193461

그림 4-14와 표 4-2는 각 학습 모델별로 [AAAABBBBCCCCDDDD] 구성으로 생성한 100곡에 대한 BLEU 점수의 평균을 나타낸다. BLEU 역시 근소하지만 간접 입력 모델이 가장 높은 점수를 보여주고 있다.

[표 4-2] 각 학습 모델로 생성한 100곡의 BLEU 평균 점수

	멜로디 학습 모델	직접 입력 모델	간접 입력 모델
BLEU 점수	0.454006547	0.458130924	0.477583583



[그림 4-14] 각 학습 모델별 BLEU 점수 그래프

곡 구성 정보를 추가한 모델들이 멜로디만 학습한 모델에 비해 METEOR과 BLUE 모두 점수가 높게 나왔다. 특히 METEOR의 경우 멜로디만 학습한

모델에 비하여 직접 입력 모델의 경우 약 0.11, 간접 입력 모델의 경우 약 0.13 정도 높게 나왔다. 이는 신경망 학습 시 곡 구성 정보를 함께 학습하여 생성된 곡이 작곡가가 작곡한 곡에 더 가깝다는 것을 보여준다.

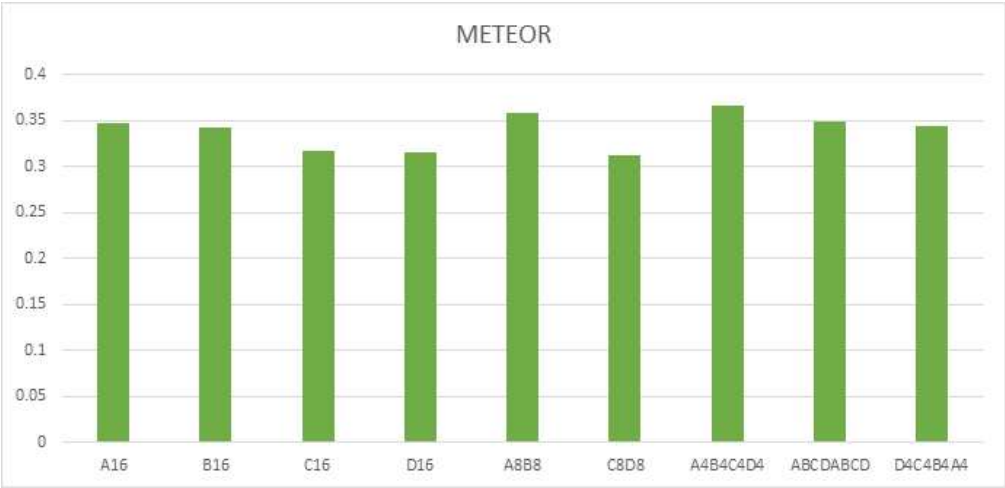
직접 입력 모델과 간접 입력 모델을 비교하였을 때는 간접 입력 모델이 직접 입력 모델에 비하여 METEOR의 경우 약 0.022(6.1%), BLEU의 경우 약 0.019(4.2%) 정도 높은 점수가 나왔다. 2.4.2절에서 언급한 것 같이 직접 입력 방법은 순환신경망 초기 단계에서 정적 데이터를 함께 포함하게 되므로 순환신경망으로 들어가는 시계열 데이터를 오염시키는 단점이 있다. 이로 인해 간접 입력(Indirect input) 모델이 직접 입력(Direct input) 모델보다는 작곡가가 작곡한 곡에 조금 더 가깝다는 것을 보여준다.

4.5.2 곡 구성 방식별 평가

표 4-3과 그림 4-15는 직접 입력 모델로 평가한 여러 곡 구성 형태에 따른 METEOR 점수를 보여준다.

[표 4-3] 곡 구성에 따른 METEOR 점수 (직접 입력 모델)

곡 구성 (라벨)	METEOR 점수
AAAAAAAAAAAAAAAA (A16)	0.3465799
BBBBBBBBBBBBBBBB (B16)	0.3423702
CCCCCCCCCCCCCCCC (C16)	0.3172613
DDDDDDDDDDDDDDDD (D16)	0.3157069
AAAAAAAABBBBBBBB (A8B8)	0.3575316
CCCCCCCCDDDDDDDD (C8D8)	0.3113909
AAAABBBBCCCCDDDD (A4B4C4D4)	0.3657669
AABBCCDDAABBCCDD (ABCDABCD)	0.3482795
DDDDCCCCBBBBAAAA (D4C4B4A4)	0.3444129



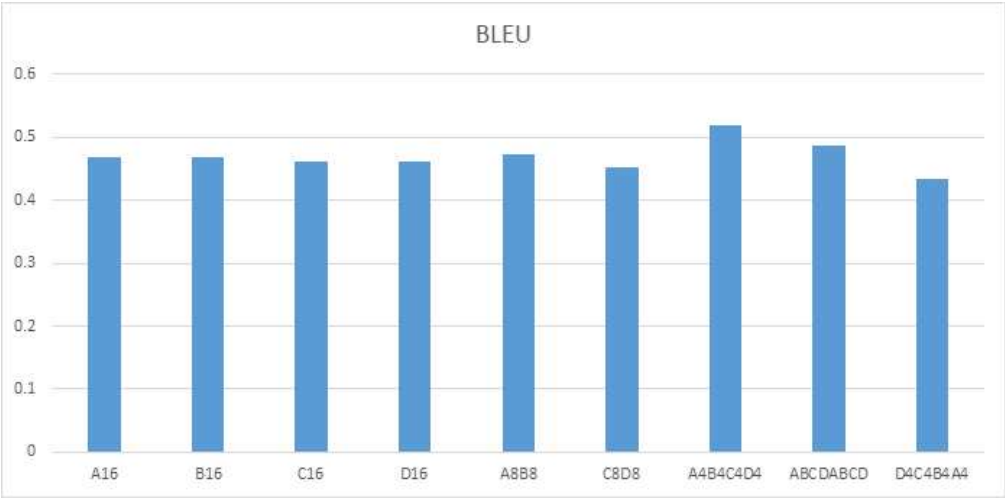
[그림 4-15] 곡 구성에 따른 METEOR 점수 (직접 입력 모델)

일반적인 동요 형식인 기승전결[AAAABBBBCCCCDDDD] 형식과 같은 방식으로 생성한 경우가 점수가 가장 높게 나왔다. 두 번째로는 [AAAAAAAABBBBBBBB] 형식으로 생성한 경우가 높게 나왔다.

표 4-4와 그림 4-16은 직접 입력 모델로 평가한 곡 구성 형태에 따른 BLEU 점수를 보여준다. METEOR과 동일한 경향을 보이고 있다.

[표 4-4] 곡 구성에 따른 BLEU 점수 (직접 입력 모델)

곡 구성 (라벨)	BLEU 점수
AAAAAAAAAAAAAAAA (A16)	0.4679101
BBBBBBBBBBBBBBBB (B16)	0.4678378
CCCCCCCCCCCCCCCC (C16)	0.4622746
DDDDDDDDDDDDDDDD (D16)	0.4624943
AAAAAAAABBBBBBBB (A8B8)	0.4721721
CCCCCCCCDDDDDDDD (C8D8)	0.4532747
AAAABBBBCCCCDDDD (A4B4C4D4)	0.5182774
AABBCCDDAABBCCDD (ABCDABCD)	0.487625
DDDDCCCCBBBBAAAA (D4C4B4A4)	0.4344197

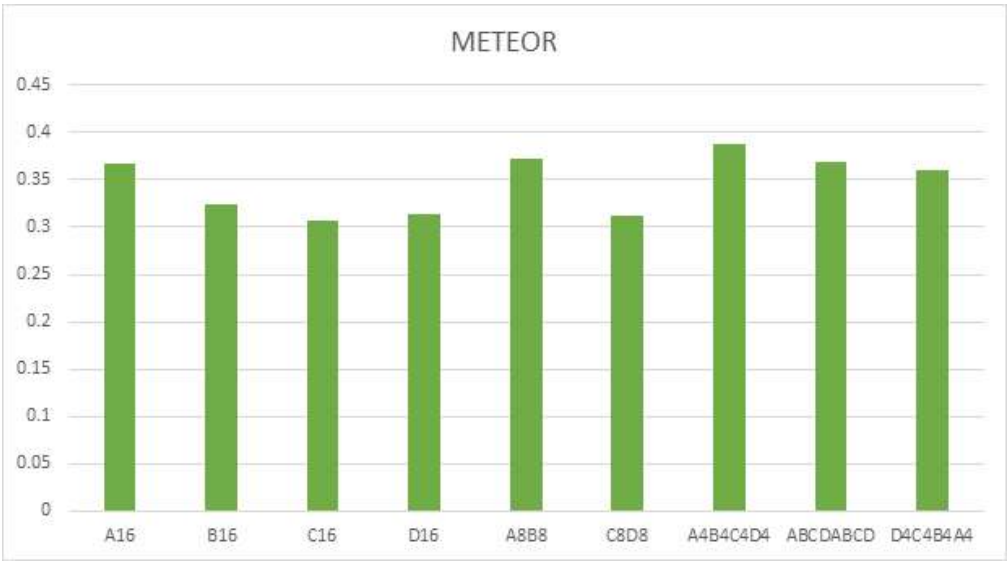


[그림 4-16] 곡 구성에 따른 BLEU 점수 (직접 입력 모델)

표 4-5와 그림 4-17은 간접 입력 모델로 평가한 곡 구성 형태에 따른 METEOR 점수를 보여준다. 직접 입력 모델과 같게 기승전결 [AAAABBBBCCCCDDDD] 형식이 가장 점수가 높게 나왔다.

[표 4-5] 곡 구성에 따른 METEOR 점수 (간접 입력 모델)

곡 구성 (라벨)	METEOR 점수
AAAAAAAAAAAAAAAA (A16)	0.3666107
BBBBBBBBBBBBBBBB (B16)	0.3239054
CCCCCCCCCCCCCCCC (C16)	0.3060361
DDDDDDDDDDDDDDDD (D16)	0.3138062
AAAAAAAABBBBBBBB (A8B8)	0.372547
CCCCCCCCDDDDDDDD (C8D8)	0.3118677
AAAABBBBCCCCDDDD (A4B4C4D4)	0.3881935
AABBCCDDAABBCCDD (ABCDABCD)	0.3685662
DDDDCCCCBBBBAAAA (D4C4B4A4)	0.3607778

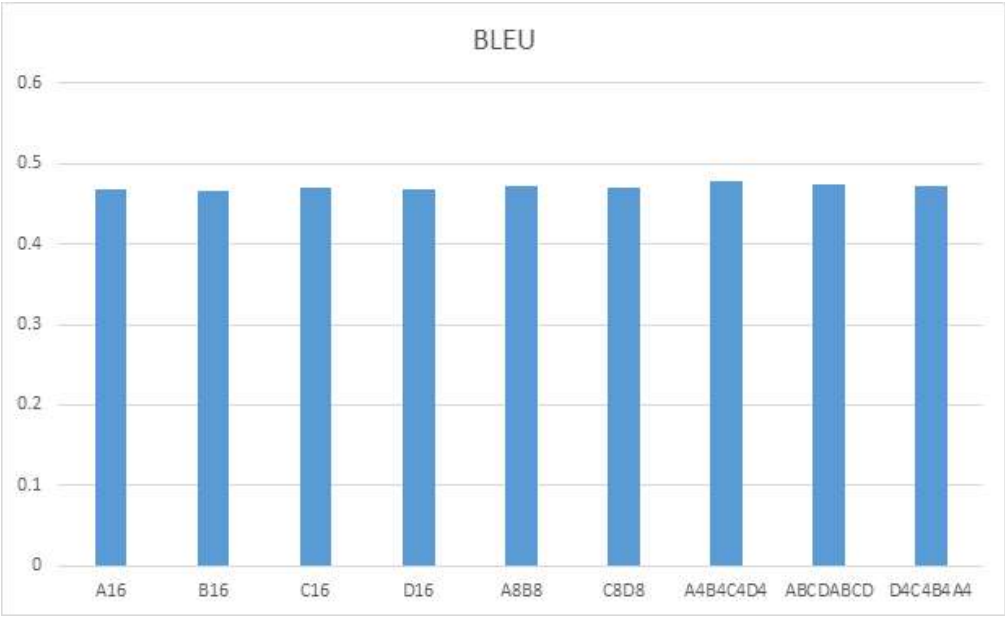


[그림 4-17] 곡 구성에 따른 METEOR 점수 (간접 입력 모델)

표 4-6와 그림 4-18은 간접 입력 모델로 평가한 곡 구성 형태에 따른 BLEU 점수를 보여준다.

[표 4-6] 곡 구성에 따른 BLEU 점수 (간접 입력 모델)

곡 구성 (라벨)	BLEU 점수
AAAAAAAAAAAAAAAA (A16)	0.4677177
BBBBBBBBBBBBBBBB (B16)	0.4666324
CCCCCCCCCCCCCCCC (C16)	0.4698707
DDDDDDDDDDDDDDDD (D16)	0.4686499
AAAAAAAABBBBBBBB (A8B8)	0.4718475
CCCCCCCCDDDDDDDD (C8D8)	0.4695927
AAAABBBBCCCCDDDD (A4B4C4D4)	0.4775836
AABBCCDDAABBCCDD (ABCDABCD)	0.4737288
DDDDCCCCBBBBAAAA (D4C4B4A4)	0.4713638



[그림 4-18] 곡 구성에 따른 BLEU 점수 (간접 입력 모델)

METEOR과 동일하게 [AAAABBBBCCCCDDDD] 형식이 가장 점수가 높게 나왔다.

직접 입력 모델과 간접 입력 모델 모두 원곡과 같거나 비슷한 형태를 가진 구성으로 곡을 생성하였을 때 METEOR과 BLEU 모두 점수가 높게 나오는 것을 볼 수 있다. 예를 들어 표 4-5의 간접 입력 모델의 METEOR 점수를 보면 [AAAABBBBCCCCDDDD], [AAAAAAAABBBBBBBB], [AABBCCDDAABBCCDD], [AAAAAAAAAAAAAAAA] 순으로 점수가 높다. 모두 A로 시작하는 구성들로 되어 있다. 이는 원곡들의 구성인 [AAAABBBBCCCCDDDD] 와 비슷한 형식을 가진 곡들이 높은 점수를 받은 것이다.

BLEU 평가의 경우 점수들이 좁은 구간에 몰려있다. 예를 들어 표 4-6의 간접 입력 모델의 BLEU의 가장 높은 점수와 낮은 점수의 차이는 약 0.01 (0.477584-0.466632=0.011) 밖에 나지 않는다. 이에 비하여 표 4-5의 간접 입력 모델의 METEOR의 가장 높은 점수와 낮은 점수의 차이는 0.082

(0.388194-0.306036=0.082)이다. 이는 METEOR이 BLEU와는 다르게 곡을 평가할 때 생성된 마디들의 존재 여부뿐만 아니라 순서까지도 고려하기 때문이다.

Ⅴ. 결론

지금까지 우리는 순환신경망을 학습시킬 때 동적 시계열 데이터 외에 부가 정보를 가지고 있는 정적 데이터를 추가해 줌으로써 신경망의 성능을 향상시킬 수 있음을 보았다. 자동작곡에서 멜로디 데이터에 곡 구성을 추가하여 학습시킴으로 구성을 갖춘 곡을 생성할 수 있었다. 생성된 곡의 평가를 위해서 METEOR과 BLEU 점수를 사용하여 곡 구성을 갖춘 모델이 그렇지 않은 모델보다 높은 점수를 얻는 것을 보았다. METEOR과 BLEU 점수가 높다고 하여 정성적으로 사람이 듣기에도 좋은 곡을 생성해 낸다고 볼 수는 어렵지만 얼마나 작곡가가 작곡한 곡에 더 가까운 곡을 생성해 내는지 정량적으로 측정할 수 있었다.

또한 METEOR과 BLEU 점수로 각 모델을 평가하였을 때 간접 입력 모델, 직접 입력 모델, 멜로디 학습 모델 순으로 평가가 좋게 나오는 것을 볼 수 있었다. 부가 정보인 정적 데이터는 순환신경망에 함께 입력하여 학습시키는 것보다 순환신경망 출력 이후에 학습에 사용하도록 간접적으로 추가해 주는 것이 좋다는 것을 확인할 수 있었다.

향후 연구로는 더 많은 음악 데이터를 확보하여 장르별로 학습시켜 여러 장르를 합쳐 새로운 곡을 생성해 볼 것이다. 또한 곡의 길이가 짧은 동요뿐만 아니라 가요나 팝송에도 곡 구성을 적용하여 intro – verse – chorus 와 같은 일반적 구성을 갖춘 대중 가요를 생성해 볼 것이다. 마지막으로 순환신경망과 피드포워드 신경망의 결합이 아닌 CCAN (Conditional GAN)을 이용한 자동작곡 모델을 고안하여 더욱 성능을 개선할 것이다.

참 고 문 헌

Allen Huang, Raymond Wu. (2016). *Deep Learning for Music*. arXiv:1606.04930

Andrej Karpathy, The Unreasonable Effectiveness of Recurrent Neural Networks, *Andrej Karpathy blog*, May 21, 2015, http://karpathy.github.io/2015/05/21/rnn-effectiveness

Banerjee, S. and Lavie, A. (2005). METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments, In *Proceedings of Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization at the 43rd Annual Meeting of the Association of Computational Linguistics*, Ann Arbor, Michigan, June 2005

Bob L. Sturm, João Felipe Santos, Oded Ben-Tal, Iryna Korshunova. (2016). *Music transcription modelling and composition using deep learning*. arXiv:1604.08723

Calvin Feng, Vanilla Recurrent Neural Network, *Machine Learning Notebook*, Last modified April, 2020, https://calvinfeng. gitbook.io/machine-learning-notebook /supervised-learning/recurrent-neural-network/recurrent_neural_networks

Christopher Olah, Understanding LSTM Networks, *colah's blog*, August 27, 2015, http://colah.github.io/posts/2015-08- Understanding -LSTMs

Donahue J, Anne Hendricks L, Guadarrama S, Rohrbach M, Venugopalan S, Saenko K, Darrell T (2015). Long-term recurrent convolutional networks for visual recognition and description.

Proceedings of the IEEE conference on computer vision and pattern recognition, 2625-2634.

Esteban C, Staeck O, Baier S, Yang Y, Tresp V (2016). Predicting clinical events by combining static and dynamic information using recurrent neural networks. *2016 IEEE International Conference on Healthcare Informatics (ICHI)*, 93-101 (IEEE).

Gevaert, Wouter & Tsenov, Georgi & Mladenov, Valeri. (2010). *Neural networks used for speech recognition. Journal of Automatic Control*. 20. 10.2298/JAC1001001G.

Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, Yi-Hsuan Yang. (2018). *MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment*. arXiv:1709.06298

Hochreiter, Sepp & Schmidhuber, Jürgen. (1997). Long Short-term Memory. *Neural computation*. 9, 1735-80. 10.1162/neco.1997.9.8.1735.

Hsu TC, Liou ST, Wang YP, Huang YS, et al. (2019). Enhanced recurrent neural network for combining static and dynamic features for credit card default prediction. *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1572-1576 (IEEE).

Joulin, Armand & Grave, Edouard & Bojanowski, Piotr & Mikolov, Tomas. (2017). *Bag of Tricks for Efficient Text Classification*. 427-431. 10.18653/v1/E17-2068.

Kotecha, Nikhil. (2018). *Bach2Bach: Generating Music Using A Deep Reinforcement Learning Approach*. arXiv:1812.01060

Krizhevsky, Alex & Sutskever, Ilya & Hinton, Geoffrey. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Neural Information Processing Systems*. 25. 10.1145/3065386.

Leontjeva A, Kuzovkin I. (2016). Combining static and dynamic features for multivariate sequence classification. *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, 21-30 (IEEE).

Lin C, Zhangy Y, Ivy J, Capan M, Arnold R, Huddleston JM, Chi M (2018). Early diagnosis and prediction of sepsis shock by combining static and dynamic information using convolutional-lstm. *2018 IEEE International Conference on Healthcare Informatics (ICHI)*, 219-228 (IEEE).

Mikolov, Tomas & Chen, Kai & Corrado, G.s & Dean, Jeffrey. (2013). Efficient Estimation of Word Representations in Vector Space. *Proceedings of Workshop at ICLR*. 2013.

Papineni, K.; Roukos, S.; Ward, T.; Zhu, W. J. (2002). BLEU: a method for automatic evaluation of machine translation. *ACL-2002: 40th Annual meeting of the Association for Computational Linguistics*. pp. 311-318. CiteSeerX 10.1.1.19.9416.

Pennington, Jeffrey & Socher, Richard & Manning, Christopher. (2014). Glove: Global Vectors for Word Representation. *EMNLP*. 14. 1532-1543. 10.3115/v1/D14-1162.

Rahman, Molla Hafizur & Yuan, Shuhan & Xie, Charles & Sha, Zhenghui. (2020). Predicting human design decisions with deep recurrent neural network combining static and dynamic data. *Design Science*. 6. 10.1017/dsj.2020.12.

Tanel Kiis, Markus Kängsepp. (2018). *Generating Poetry using Neural Networks*.

Wang, Tong & Jin, Fujie & Yu, & Hu, & Cheng, Yuan. (2019). *Early Predictions for Medical Crowdfunding: A Deep Learning Approach Using Diverse Inputs.*

Won Joon Yoo. (2020). Introduction to Deep Learning for Natural Language Processing: BLEU Score(Bilingual Evaluation Understudy Score). *Wikidocs*. Last modified July 21, 2020, https://wikidocs.net/31695

Wikipedia. (2020). *METEOR*, Last modified June 16, 2020, https://en.wikipedia.org/wiki/METEOR

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, Jeffrey Dean. (2016). *Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*. arXiv:1609.08144v2

Zhu F, Song X, Zhong C, Fang S, Bouchard R, Fontama VN, Singh P, Gao J, Deng L (2018). *Churn prediction using static and dynamic features*. US Patent App. 15/446,870.

ABSTRACT

A Study on the Generating Method of Songs with structure in Automatic Composition Based on Deep Learning

Chung, Suk-Hwan

Major in Futures Convergence Strategy Consulting

Dept. of Futures Convergence Consulting

Graduate School of Knowledge Service Consulting

Hansung University

With the recent development of deep learning technology, methods using AI(artificial intelligence) are being introduced in almost all fields. Even artificial intelligence is being actively applied in the field of creation, which has been believed that only humans can do it. In 2018, at Christie's auction house in New York, the first artificial intelligence-painted 'Edmond De Belamy' was sold for $432,000. AI is no exception to music. Many attempts are being made to compose new songs using AI. In 2016, Google announced the magenta project to design an AI algorithm capable of creating music and art. However, despite these various attempts, it is difficult to find a case that still produces a song with a natural composition like that composed by humans.

In general, a song has a certain structure, such introduction, verse, chorus, bridge and outro. The melody of a song automatically composed using an existing artificial neural network is output according to the

melody of the learned song, and it is difficult to create a song that has a specific musical composition, such as those composed by humans. In this paper, I propose a method of inserting song structure information in order to improve the lack of music composition in automatic composition using artificial neural networks. I devised a method for learning by dividing existing music data into melody, which is dynamic data that changes over time, and song composition information, which is auxiliary static data. Children's songs having a simple composition of songs were used as the experimental data, and the METEOR scores and BLEU scores were used for quantitative evaluation the generated songs. As a result of the experiment, it was confirmed that songs composed using the proposed method generally have a musical structure and show better performance in evaluation as well.

【Keywords】Automatic composition, Deep learning, Artificial neural network, Combining dynamic data and static data, Song structure, METEOR, BLEU