Hand Craft Model

Yeong-Min Ko & Min-Seo Park

2023.12.06

_INDEX

Ⅰ. How to evaluate

i. Evaluation Metrics

I. How to evaluate

i. Evaluation Metrics

- **음정 & 옥타브(그루핑 필요)**
  - 다음 그림은 음정과 옥타브를 판단하는 의사 코드

```python
# 음정 판단(note)
def number_to_note(number):
    notes = ['c', 'c#', 'd', 'd#', 'e', 'f', 'f#', 'g', 'g#', 'a', 'a#', 'b']
    return notes[number%12]
```

```python
# 옥타브 판단
def number_to_note_octave(number):
    notes = ['c', 'c#', 'd', 'd#', 'e', 'f', 'f#', 'g', 'g#', 'a', 'a#', 'b']
    octave = (number // 12) + 1
    note = notes[number % 12]
    return octave
```

- **음정과 옥타브가 모두 일치하는 경우**
  - 음정이 일치하므로 일정한 점수를 부여
  - 옥타브가 일치하지 않으므로 일치하지 않는 정도에 따라 감점
- **음정이 일치하는데 옥타브가 일치하지 않는 경우**
  - 음정이 일치하므로 일정한 점수를 부여
  - 옥타브가 일치하지 않으므로 일치하지 않는 정도에 따라 감점
- **음정이 일치하지 않지만 옥타브가 일치하는 경우**
  - 점수를 추가하지 않음
- **음정과 옥타브가 모두 일치하지 않는 경우**
  - 점수를 추가하지 않음

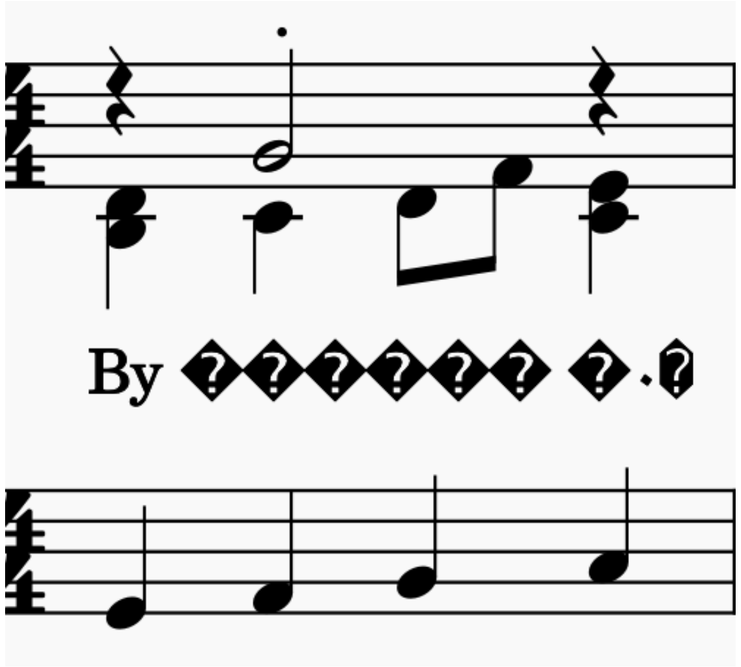
	평가 기준	매트릭
1	음정	pitch
2	셈여림	decibel
3	셈여림의 변화	change of decibel
4	빠르기	speed
5	빠르기의 변화	change of speed
6	붙임줄, 스타카토, 테누토, 늘임표	duration of note
7	악센트	decibel
8	옥타브	note
9	꾸밈음, 반복 기호	note

공식화 가능한 기준

I. How to evaluate

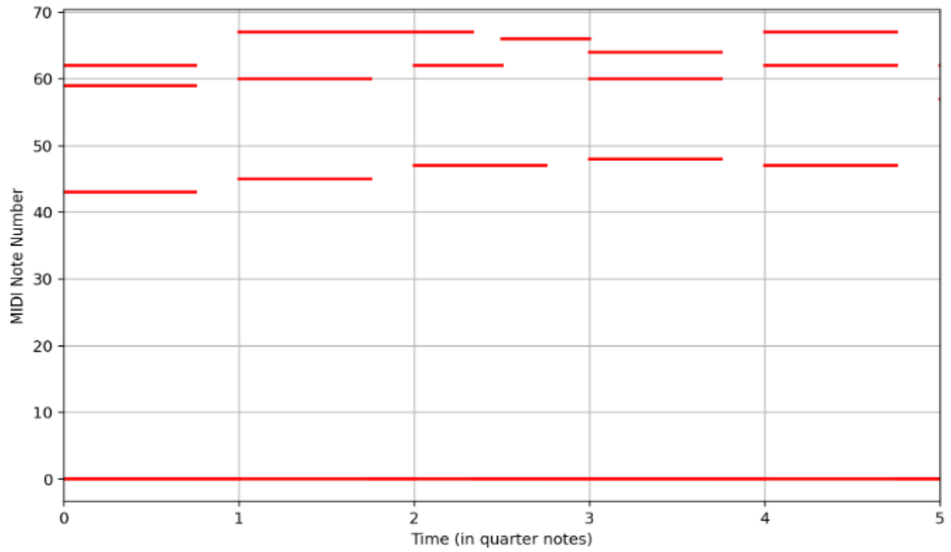
i. Evaluation Metrics

- 음정 & 옥타브(그루핑 필요)
  - 같은 시작 시간에 누른 코드가 일치하는지로 판단해야할 것 같음




```
[<music21.chord.Chord B3 D4>, <music21.note.Rest quarter>, <music21.note.Note G>, <music21.note.Rest whole>, <music21.note.Rest 16th>, <music21.note.Rest 16th>, <music21.note.Note C>, <music21.note.Note G>, <music21.note.Note A>, <music21.note.Rest 16th>, <music21.note.Rest 16th>, <music21.note.Note D>, <music21.note.Note B>, <music21.note.Rest 5/3ql>, <music21.note.Note F#>, <music21.note.Rest 16th>, <music21.chord.Chord C4 E4>, <music21.note.Note C>, <music21.note.Rest 16th>, <music21.note.Rest 16th>, <music21.chord.Chord D4 G4>, <music21.note.Note B>, <music21.note.Rest whole>, <music21.note.Rest 16th>, <music21.note.Rest 16th>, <music21.chord.Chord A3 D4>, <music21.note.Note F#>, <music21.note.Rest 16th>, <music21.note.Rest 16th>, <music21.chord.Chord G3 B3>]
[0.0, 0.0, 0.0, 0.0, 0.0, 0.75, 0.75, 1.0, 1.0, 1.0, 1.75, 1.75, 2.0, 2.0, Fraction(7, 3), 2.5, 2.75, 3.0, 3.0, 3.0, 3.75, 3.75, 4.0, 4.0, 4.0, 4.0, 4.75, 4.75, 5.0, 5.0, 5.0, 5.75, 5.75, 6.0, 6.0, 6.0, 6.75, 6.75, 7.0, 7.0, 7.0, 7.75, 7.75, 8.0, 8.0, 8.0, 8.0, 8.75, 8.75, 9.0, 9.0, 9.0, 9.5, 9.5, 9.5, 10.0, 10.0, 10.0, 10.75, 10.75]
[59, 62, 0, 43, 0, 0, 0, 60, 67, 45, 0, 0, 62, 47, 0, 66, 0, 60, 64, 48, 0, 0, 62, 67, 47, 0, 0, 0, 57, 62]
[0.75, 0.75, 1.0, 0.75, 4.0, 0.25, 0.25, 0.75, Fraction(4, 3), 0.75, 0.25, 0.25, 0.5, 0.75, Fraction(5, 3), 0.5, 0.25, 0.75, 0.75, 0.75, 0.25, 0.25, 0.75, 0.75, 0.75, 4.0, 0.25, 0.25, 0.75, 0.75]
```

- 테두리 부분이 건반 번호(pitch) 속성 활용
- octave = (pitch_midi // 12) - 1



코드는 깃허브(Research_and_Experimental_Results/Piano Performance Evaluation)/MinseoPark/)에 업로드

I. How to evaluate

i. Evaluation Metrics

- **셈여림 및 셈여림의 변화**
  - 음표 간의 간격, 강세 및 강약의 변화를 통해 판단해야 할 것 같음(추가 고민 필요)

I. How to evaluate

i. Evaluation Metrics

- **빠르기(Velocity) 활용**
  - **빠르기 및 빠르기의 변화**
  - 악센트
    - 기존의 방법에는 음을 하나씩 눌러 이전 음과 다음 음 사이의 시간을 기반으로 빠르기를 구하였음
    - 하지만 실제 피아노는 동시에 여러 음을 연주하기에 위 방법은 옳지 않음
    - 따라서, 이 문제는 동시에 치는 음들에 대한 그루핑으로 해결함 ~ 이전과 같이 리듬게임처럼 실시간으로 노트를 치는 방식이라면 해당 노트 연주시 미세한 오차가 있어서 불가능할 것 같고 midi파일과 midi 파일의 단순 비교만 가능할 것 같음(이 경우, 곡 전체의 tempo를 비교하여 점수를 매기는 것도 하나의 방법)

```python
def calculate_bpm(first_timestamp, last_timestamp, beat_count):
    duration = last_timestamp - first_timestamp
    if duration == 0:
        return 0
    beats_per_second = beat_count / (duration / 1000)  # 밀리초를 초로 변환
    return beats_per_second * 60

while True:
  # 노트 집합 누를 때마다? 초 단위로?
  beat_count += 1
  if beat_count > 4:  # 예시로 4박자마다 BPM을 출력하도록 설정
     bpm = calculate_bpm(first_timestamp, prev_end_timestamp, beat_count)
     print(f'Estimated BPM: {bpm}')
     beat_count = 0
```

(그림) 기존의 접근 방식

I. How to evaluate

i. Evaluation Metrics

- **빠르기(Velocity) 활용**
  - 빠르기 및 빠르기의 변화
  - **악센트**: 곡의 연주 도중 특정한 음을 세게 낼 때 사용하는 기호

- velocity 값이 64 이상인 경우를 강세(악센트)로 간주
  - 근거: velocity 값의 범위는 0~127, 그 중앙값인 64를 넘어서면 보통은 세기가 강한 것으로 인식
  - 테스트 결과: 실제 건반에 악센트를 주려면 키를 강하게 눌러야 하는데 velocity 값이 높게 출력됨
  - 다른 방안: 음악 전체의 velocity를 구하고 이를 기준으로 특정 비율 이상 높은 값을 악센트로 간주

```
# 악센트 판단
is_accent = velocity >= 64
return 'accent'
```

```
# 악센트
def detect_accent(velocities, threshold_ratio=1.5):
    # 벨로시티의 평균 계산
    average_velocity = sum(velocities) / len(velocities)

    accents = []
    for velocity in velocities:
        # 상대적 벨로시티 계산
        relative_velocity = velocity / average_velocity

        # 특정 비율 이상인 경우를 악센트로 간주
        if relative_velocity >= threshold_ratio:
            accents.append(True)
        else:
            accents.append(False)

    return accents
```

- **악센트 시 높은 점수**
  - 악센트 부분에서 정확한 강세를 표현하면 높은 점수를 부여
- **특정 벨로시티 이상에 따른 추가 점수**
  - 일정 벨로시티 이상의 음표를 악센트로 간주하고,
    해당 음표에서 정확한 강세를 표현하면 추가 점수를 부여

I. How to evaluate

i. Evaluation Metrics

6	붙임줄, 스타카토, 테누토, 늘임표	duration of note

- 붙임줄(legato), 스타카토(staccato), 테누토(tenuto), 늘임표(fermata)
  - 6번 항목을 판단하기 위해서는 연속된 음표 간의 간격을 살펴볼 필요가 있음
  - 붙임줄(legato): 음과 음이 서로 이어지게 연주하도록 지시하는 기호
  - 스타카토(staccato): 레가토와는 반대로 음표를 짧게 끊어서 연주하도록 지시하는 기호
  - 테누토(tenuto): 음표가 가지고 있는 길이를 충분히 지켜서 폭넓게 연주하도록 지시하는 기호
  - 늘임표(fermata): 음표나 쉼표가 가지고 있는 길이의 2~3배로 늘려서 연주하도록 지시하는 기호

```python
def detect_staccato(current_timestamp, prev_timestamp, staccato_threshold=50):
    time_difference = current_timestamp - prev_timestamp
    return time_difference < staccato_threshold

def detect_legato(current_timestamp, prev_end_timestamp, legato_threshold=100):
    time_difference = current_timestamp - prev_end_timestamp
    return time_difference < legato_threshold

def detect_tenuto(current_timestamp, prev_end_timestamp, tenuto_duration=500):
    time_difference = current_timestamp - prev_end_timestamp
    return time_difference > tenuto_duration

def detect_fermata(current_timestamp, prev_end_timestamp, fermata_duration=2000):
    time_difference = current_timestamp - prev_end_timestamp
    return time_difference > fermata_duration
```

- 최근 노트와 이전 노트간의 간격을
  이용해서 각각을 판단하고 정확하면 점수 추가
- 각각의 threshold를 정하는 기준을 고민해볼 필요있음

I. How to evaluate

i. Evaluation Metrics

- 꾸밈음, 반복 기호
  - 꾸밈음과 반복 기호는 MIDI 데이터에서 직접적으로 확인하기 어려워 추가적인 접근법 고민이 필요

	평가 기준	매트릭
1	음정	pitch
2	셈여림	decibel
3	셈여림의 변화	change of decibel
4	빠르기	speed
5	빠르기의 변화	change of speed
6	붙임줄, 스타카토, 테누토, 늘임표	duration of note
7	악센트	decibel
8	옥타브	note
9	꾸밈음, 반복 기호	note

공식화 가능한 기준

II. Refences

- 악상기호(Tistory)
  - https://geumse.tistory.com/14

THANK YOU

컴퓨터공학과 3학년 고영민

2023.12.06