Data Science Final Project Report

Music Generator

309505018 郭俊廷
0751231 曾揚

1. Motivation:

在這次的final project中，目標是用深度學習實作鋼琴音樂的生成器。輸入一段隨機或指定的旋律(seqence)，使其接續產生後續的音符，進而形成一段新的旋律。本文中使用預訓練好的embedding層將音調轉換成8維之vector，並用包含雙向GRU層及self-attention層的model做監督式學習(supervised learning)。期望生成的旋律能夠接近現實中人所創作出來的風格，困難點主要是在於如何將音樂邏輯與樂理包含在內，包括樂曲的結構性以及和弦的彼此搭配等。

2. Related Work:

我們Final Project主要是實作鋼琴音樂的生成器, 輸入一個MIDI(Musical Instrument Digital Interface)格式的鋼琴音樂檔,針對音樂的每一段和弦(chord)採用chord2vec 作 chord embedding, 然後輸入一個音調(note)或一小段和弦(2~4個音調), 在train model架構採用兩層 self-attention layer 及三層雙向 GRU layer (Gated Recurrent Unit, a variant of Recurrent Neural Network), 再加上dropout, chord embedding目的是為了使音調跟音調之間有相似度, 音調攜帶了和弦訊息，還可以計算一段和弦出現的機率, 最後生成出一段30秒左右的和弦(chord), 再用pretty_midi (a python package)把和弦(chord)轉換成MIDI格式音樂檔

3. Approach:

整體的目標是希望生成出來的鋼琴音, 不僅有旋律, 還可以很自然很有情感, 跟人類演奏出來的越接近越好

3-1 Data :

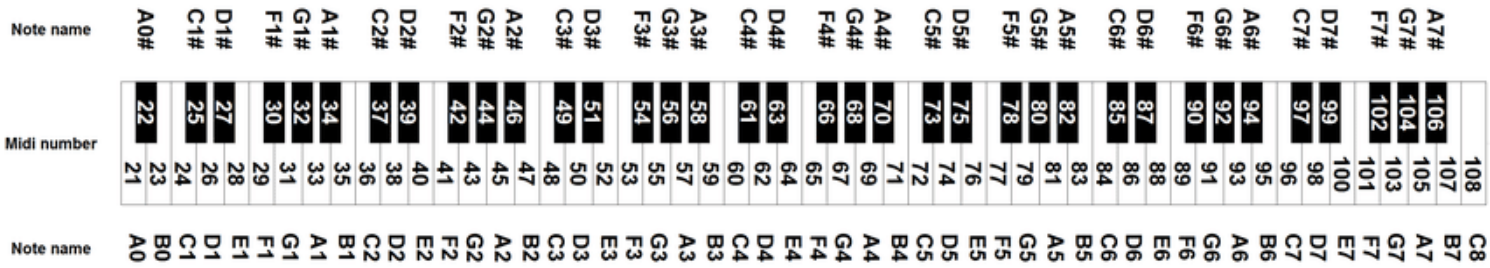
我們的dataset是用The MAESTRO Dataset (https://magenta.tensorflow.org/datasets/maestro), 我們是下載裡面的maestro-v1.0.0-midi.zip, 如下圖。

Split	Performances	Compositions (approx.)	Duration (hours)	Size (GB)	Notes (millions)
Train	954	295	140.1	83.6	5.06
Validation	105	60	15.3	9.1	0.54
Test	125	75	16.9	10.1	0.57
Total	**1184**	**430**	**172.3**	**102.8**	**6.18**

3-2 Data pre-processing:

我們使用 pretty_midi (a python package)讀取MIDI鋼琴音樂檔

在 preprocessing階段, 因為每個和弦(chord)都由1~6個音調所組成, 因此我們實作了 NoteTokenizer, NoteTokenizer的功能是記錄每個和弦分別由那些音調所組成, 每個音調分別用一個數字來表示, 如下圖



音調A0對應21、音調A0#對應22、音調B0對應23, 以此類推, 並且建立音調轉數字note_to_num及數字轉音調num_to_note的這兩個矩陣, 把training data所有出現過的和弦記錄在這兩個矩陣裡。

並且計算出在training data裡, 不同的音調有哪些, 每個音調出現的頻率高低, 這三個值。

3-3 Note2vec

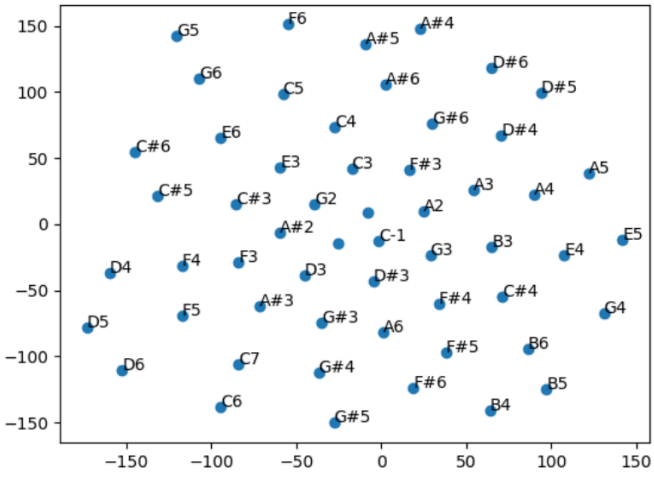
Note2vec是指將每個音調(note)用一個vector來表示, 以電腦來說, 兩個獨立的音調之間是沒有任何關聯性的, 訓練vector時是把容易同時出現的音符關聯性增強 所以我們把每一個音調(note)用一個vector來表示。

我們的note2vec是採用這裡的pretrain model [1]
https://github.com/philhchen/note2vec

3-4 Chord2vec

Chord2vec 最早的來源是 "Chord2Vec: Learning Musical Chord Embeddings, NIPS 2016" [2],這篇paper。Chord2vec的概念就是將每個和弦(chord)用一個vector來表示, 那表示方式就是把每個音調相加。

在前面的Note2vec裡提到每個音調(note)用一個vector來表示, 而每個和弦(chord)是由很多個音調(note)所組成, 這篇paper提到可以把每個音調的vector相加, 變成和弦的vector, 用這個vector去表示這個和弦, 並且作者也實驗出, 這樣做有很好的結果。



3-5 Proposed Model

Input 一個音調或一個和弦, 並用Chord2vec作Chord Embedding

Model架構是 bidirectional GRU-> self-attention-> dropout-> bidirectional GRU-> self-attention-> dropout-> bidirectional GRU-> dropout-> dense-> Leaky ReLU-> dense-> output。

其中 self-attention機制是來自於Google的Transformer, "Attention Is All You Need, NIPS 2017" [6], 他的計算方式有三個步驟, 1. encoder self attention，存在於encoder間. 2. decoder self attention，存在於decoder間，3. encoder-decoder attention, 這種attention算法和過去的attention model相似。self-attention跟以前的Sequence-to-sequence和Attention model相比, 多了encoder-decoder attention, Transformer 也證明了word-pair 之間的關係比 word-chain 更為重要, 而且針對非常長的文本序列, 提出了一種只關注 r 個鄰居的 self-attention機制, 使運算效率增加許多。

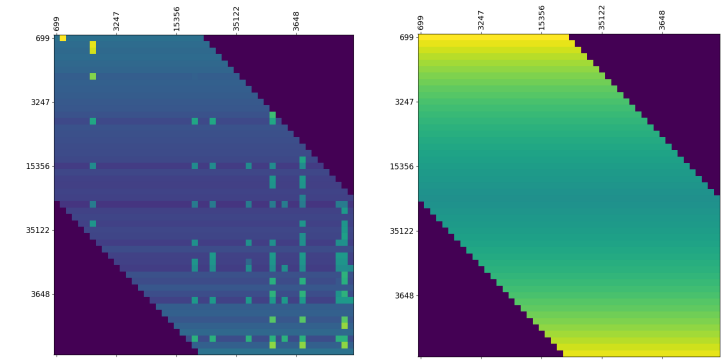
```
Model: "generate_scores_rnn"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 50)]              0
_________________________________________________________________
embedding (Embedding)        (None, 50, 8)             504088
_________________________________________________________________
bidirectional (Bidirectional (None, 50, 256)           105984
_________________________________________________________________
seq_self_attention (SeqSelfA [(None, 50, 256), (None,  65537
_________________________________________________________________
dropout (Dropout)            (None, 50, 256)           0
_________________________________________________________________
bidirectional_1 (Bidirection (None, 50, 256)           296448
_________________________________________________________________
seq_self_attention_1 (SeqSel [(None, 50, 256), (None,  65537
_________________________________________________________________
dropout_1 (Dropout)          (None, 50, 256)           0
_________________________________________________________________
bidirectional_2 (Bidirection (None, 256)               296448
_________________________________________________________________
dropout_2 (Dropout)          (None, 256)               0
_________________________________________________________________
dense (Dense)                (None, 64)                16448
_________________________________________________________________
leaky_re_lu (LeakyReLU)      (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 63011)             4095715
=================================================================
Total params: 5,446,205
Trainable params: 5,446,205
Non-trainable params: 0
```

3-6 Training detail

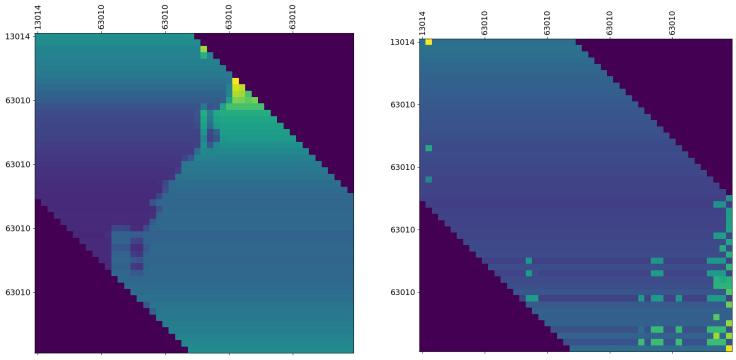
我們使用seq_len = 50, epochs=4, batch size = 96, frame_per_second = 5, 總共63010種和弦去作training, seq_len = 50是指我們input給GRU layer(Gated Recurrent Unit)的長度是50。在訓練的一開始，為了使input的長度為50, 我們在旋律的開頭填補了49個空音符做為初始的input,接續產生後續的旋律。

在Loss function是用sparse_categorical_crossentropy, optimizer是用adam Train完後把model存起來, 就可以開始generate了。

4. Experimental Results:



輸入A3 音調 的生成
Visualize Self-Attention



輸入B3 音調 的生成
Visualize Self-Attention

一些demo結果：

輸入一個音調G3, 所生成的鋼琴音樂 one_note_G3.wav
輸入一個音調B3, 所生成的鋼琴音樂 one_note_B3.wav
輸入一個音調A3, 所生成的鋼琴音樂 one_note_A3.wav

5. Conculsion:

在一開始我們沒有使用Chord2vec作Chord embedding時, 所生成出來的鋼琴音樂很像在亂彈, 很難聽又沒有節奏感, 後來找到NIPS 2016那篇paper後[2], 加入了Chord2vec作Chord embedding, 弦律才聽起來比較正常、比較美妙。在train model時採用Transformer的self-attention機制, 相比以前的Seq2seq和Attention model, 他加入了encoder-decoder之間的attention, 並且計算每個音調跟音調之間的關聯, 在Feedforward時考慮上下和弦之間的關聯性, 可以讓生成出來的音樂更有連貫性, 聽起來更像是人類彈的, 最後也證明這個作法可以得到不錯的結果。

6. Future:

這次在找資料意外發現OpenAI的Jukebox (https://openai.com/blog/jukebox/) Jukebox是用VAE (Variational AutoEncoder)實作的, 希望未來可以用VAE (Variational AutoEncoder)或 GAN (Generative Adversarial Network) 搭配 Chord2vec去生成看看, 因為生成的方法很多種, 說不定用VAE或GAN 會有更不同風格的弦律產生, 說不定會更好聽。

7. Teamwork Assignment:

曾揚:

data preprocessing(note2vec, chord2vec), Paper survey

郭俊廷:

collecting data, model architecture, training model, spotlight video, PPT, report.

8. References:

(1) https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1194/reports/custom/15846208.pdf?fbclid=IwAR3wtcBfWO39O7YmQeuhRxlsTTwL9z7S6RcVzrg7KN-mDjOHHKs_VaaB0Ko

(2) http://www.cs.nott.ac.uk/~psztg/cml/2016/papers/CML2016_paper_5.pdf

(3) https://magenta.tensorflow.org/datasets/maestro

(4) https://github.com/philhchen/note2vec

(5) https://github.com/CyberZHG/keras-self-attention

(6) https://arxiv.org/pdf/1706.03762.pdf