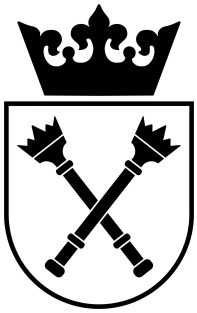
Uniwerystet Jagielloński

Projekt końcowy

z przedmiotu Nauczanie Maszynowe

Cadenza - generator kolejnych nut początkowej próbki melodii

Wojciech Szlosek



Rok akademicki 2022/2023

1 Opis wstępny projektu

Projekt polegał na przewidywaniu (predykcji) kolejnych nut muzycznych względem danej próbki początkowej. Próbka jest każdorazowo wprowadzana przez użytkownika, powinna być w formacie MIDI. Użytkownik może wgrać plik, wyszukując go na swoim komputerze. Możliwym jest predykcja teoretycznie dowolnej liczby nut, domyślnie ustawiona jest stała wartość równa 30.

2 Technikalia

Głównie: Python, TensorFlow, music21, pretty_midi, numpy

3 Ograniczenia i założenia

Model został wytrenowany na fragmencie danych ze zbioru Maestro[1] - tamtejsze dane to pliki w formacie MIDI, każdy z utworów został zagrany na fortepianie klasycznym. Stąd najlepiej jako dane wejściowe użyć klasycznej próbki melodii.

Przyjęto, że odległość czasowa między poszczególnymi nutami/akordami jest stała. W takiej konwencji generowany jest rezultat.

Format wejściowy plików to format MIDI - to popularny format cyfrowy dźwięku. Wymóg ten bierze się przede wszystkim z ograniczeń jakościowych innych formatów muzycznych. Konwersja np. z pliku .mp3 zaburza możliwości dokładnego odczytywania nut.

4 Zbiór danych treningowych

Do treningu modelu użyto 500 plików muzycznych ze wspomnianego wcześniej zbioru Maestro. Uznano, że jest to reprezentatywny przykład spośród całego zbioru, wystarczający do generowania oryginalnych rezultatów. W wyniku analizy danych wykryto, że minimalna ilość nut i akordów w plikach to 474. W konsekwencji, znormalizowano każdy z plików do pierwszych 400 nut (wystarczająco długa próbka każdego z utworów).

Odczytanie nut i akordów z każdego pliku MIDI zostało wsparte biblioteką music21[2]. Biblioteka ta rozróżnia nuty od akordów (jako osobne struktury danych), stąd ja również korzystam z takiego podziału. Akordy zostały przedstawione słownie jako "suma" nut (np. string "x+y+z"). Nuty zostały z kolei zinterpretowane po prostu po swojej nazwie. Finalnie uzyskano wektor wektorów M, gdzie M[i] oznacza i-ty plik MIDI i M jest postaci [[(str) nuta/akord, (str) nuta/akord, ...], ...].

Każdy z elementów należało zamienić na liczbę. Jako że ustalone, że ich unikalnych wartości jest 1335, to każdej z nich dopasowano unikalną wartość nuty/akordu. Tym sposobem uzyskano jednoznaczną referencję nuty do liczby (i odwrotnie). Finalnie zmapowano wszystkie wartości M do unikalnej liczby, otrzymując wektor wektorów zawierających liczby. Następnym etapem była zamiana każdej z liczb na wektor wartości [0, 1]. Indeks jedynej wartości jeden symbolizował liczbę (nutę). Pozostałe wartości w wektorze to zera. Można więc uznać, że taki wektor symbolizował prawdopodobieństwo wystąpienia każdej z nut, gdzie P(idx) oznaczało prawdopodobieństwo wystąpienia nuty idx.

Końcowym krokiem było określenie danych wejściowych (X) oraz rezultatu (Y) z wektora M, będącego wygenerowaną nutą/akordem (a w czasie treningu przyjętym jako prawidłowy output). Domyślnie przyjąłem, że 20 kolejnych nut wygeneruje kolejną, nową, dwudziestą pierwszą nutę. Zatem: X[0:20] → Y[0]; X[1:21] → Y[1] - i tak dalej do końca utworu i dla każdego pliku.

[1] https://magenta.tensorflow.org/datasets/maestro
[2] http://web.mit.edu/music21

5 Architektura modelu

```
model = Sequential()
model.add(LSTM(128, return_sequences=True))
model.add(LSTM(80))
model.add(Dense(unique_notes, activation="softmax"))
model.compile(loss='categorical_crossentropy', optimizer=tf.keras.optimizers.Adam(), metrics=['accuracy'])
```

Architektura modelu przedstawiona jest na screenie powyżej. Obejmuje warstwy LSTM i końcową warstwę Dense (zauważmy, że o wielkości unique_notes, która obejmuje u mnie 1335 wartości) z aktywacją softmax, wybraną pod wyznaczenie prawdopodobieństwa na wystąpienie danej nuty w następnej kolejności. Metryką jest wskaźnik accuracy, a funkcją loss categorical crossentropy. Uczenie obejmowało ok. 200 epok, finalna miara accuracy wyniosła ok. 80%. Loss na koniec obejmował wartość ok. 0.17.

Architektura była wielkokrotnie modyfikowana, obecnie sprawuje się dobrze dla testowych przykładów - tonacja rezultatów jest ciągła, rytm często jest zachowany. Sieć LSTM była bardzo dobra pod predykcję muzyczną. Long Short-Term Memory pozwala na zachowanie długoterminowych zależności między danymi - tendencje i rytm utworów muzycznych mogły być przechowane i utrwalone, co przekłada się z pewnością na rezultaty.

6 Wyniki

Z technicznego punktu widzenia, model generuje kolejne nuty generując wektorowe wartości prawdopodobieństw (gdzie indeks oznacza reprezentację liczbową nuty/akordu). Stworzono funkcje do wyszukania indeksu z największym prawdopodobieństwem, indeks ten zamieniono na liczbę, a liczbę zmapowano na nazwę nuty/akordu. Tą wartość z kolei opakowano w strukturę z biblioteki music21, co pozwoliło na wygenerowanie końcowego pliku MIDI dostępnego do odtworzenia lub pobrania.

Przykładowe rezultaty znajdują się w folderze "outputs/". Uważam, że otrzymane wyniki charakteryzują się dobrym dopasowaniem do próbki, tonacja jest zachowana, rytm nie odbiega od całości. Moją opinię podzielają również osoby, które zajmują się muzyką zawodowo.

Zachęcam do testowania i własnych wniosków.

7 Sposób testowania projektu[3]

Z GitHuba projektu[4] wystarczy pobrać notebooka "MusicGenerator.ipynb", przetrenowany przeze mnie model "music_model.hdf5" oraz folder "data/" z zawartością (wystarczą pliki pickle int_to_note oraz note_to_int. Uruchomiony notebook pobierze wymagane zależności, zaimportuje biblioteki i załaduje dane, w tym model, a następnie poprosi o wybór próbki startowej w formacie MIDI z pamięci komputera. Najlepsze rezultaty obserwuje się dla danych muzycznych fortepianu i pianina - jako że na takich danych wytrenowano model, ale nic nie stoi na przeszkodzie, by próbować też na innych próbkach. Następnie zostaną wygenerowane kolejne nuty (domyślnie 30), zapisując rezultat w pliku w formacie MIDI, który można pobrać (polecana opcja) lub odsłuchać bezpośrednio z poziomu notebook (niższa jakość dźwięku).

Domyślnie, po dwudziestej nucie od oryginału, następuje całkowicie odmienny dźwięk (sekwencja nut A0) i chwila ciszy. Jest to zamierzony zabieg, by oddzielić oryginalną próbkę od wygenerowanej muzyki.

[3] Testowano na Google CoLab
[4] https://github.com/wszlosek/Cadenza