

Grands modèles de langage

Nicolas Wicker

Laboratoire Paul Painlevé, Université Lille, 59655, Villeneuve d'Ascq, France.



Le langage de la musique en informatique :

Le format MIDI (1982)

- MIDI : Musical Instrument Digital Interface
- standard de communication entre ordinateurs (Atari St) et instruments musicaux
- se lit avec beaucoup de logiciels, par exemple : timidity sur Ubuntu, windows media player sur Windows)

A quoi cela ressemble-t-il ?

Fichier binaire lisible par les machines mais pas par l'humain
(lu avec hexdump ici)

```
00000000 544d 6468 0000 0600 0100 0200 e001 544d
00000010 6b72 0000 1300 ff00 0351 a107 0020 58ff
00000020 0404 1802 0108 2fff 4d00 7254 006b 0100
00000030 003c 03ff 5005 6169 6f6e c000 0200 2d90
00000040 8250 8000 502d 4e83 3090 8150 3270 0450
00000050 3080 8150 906c 5034 8010 5032 346e 8250
00000060 9062 502d 3e85 2d80 1250 2b90 8350 804a
00000070 502b 9016 502d 0c82 2d80 8550 9034 502d
00000080 1881 2d80 8450 9038 5030 1881 3080 5850
00000090 3290 8150 3470 0250 3280 8150 343e 8250
000000a0 9020 502d 5085 5030 8008 502d 4283 5030
000000b0 9016 502d 3281 2d80 8650 900e 502d 2481
000000c0 2d80 4c50 2d90 8150 8018 502d 4882 2b90
000000d0 8150 804c 502b 9024 502d 1881 2d80 5850
```



Le langage de la musique en informatique :

Exemple : unité le “tick”

POSITION_217

NOTE_ON_74

DURATION_107

POSITION_291

NOTE_ON_72

DURATION_78

POSITION_366

NOTE_ON_71

DURATION_46

POSITION_366

NOTE_ON_69

DURATION_54

Le langage de la musique en informatique :

Quelques info supplémentaires

- on peut indiquer les TPQ (Tick Per Quarters) : c'est le nombre de battements par note noire
- TEMPO : nombre de noires (ou battements) par minute (à diminuer si on veut ralentir la musique)

LLM pour la musique

Dans la littérature :

- ChatMusician (2024) à l'université HKUST (Hong-Kong)
- Midi-LLM: Adapting Large Language Models for Text-to-MIDI Music Generation (Neurips 2025)

Deux programmes pour commencer (1/4)

Un convertisseur midi vers tokens

```
import miditoolkit

nomFichier="FichiersMidi/GrandMidiPiano/Bach, Johann Sebastian, Partita in \
G major, BWV 829, aecenXn3obw.mid"
midi = miditoolkit.MidiFile(nomFichier)

tokens = []

for inst in midi.instruments:
    for note in inst.notes:
        tokens.append(f"POSITION_{note.start}")
        tokens.append(f"NOTE_ON_{note.pitch}")
        tokens.append(f"DURATION_{note.end - note.start}")

for token in tokens:
    print(token)
```

Deux programmes pour commencer (2/4)

Un convertisseur tokens vers midi

```
import sys
import miditoolkit

# --- paramètres globaux ---
TPQ = 480 # ticks per quarter
TEMPO = 120

nomFichier=sys.argv[1]
print(nomFichier)
file=open(nomFichier,"r")
contenuFichier=file.read()
contenuFichierSplt=contenuFichier.split("\n")

tokens=[]
for token in contenuFichierSplt:
    tokens.append(token)
print(tokens)
# créer le MIDI
midi = miditoolkit.MidiFile(ticks_per_beat=TPQ)

midi.tempo_changes.append(
    miditoolkit.TempoChange(TEMPO, time=0)
)
# instrument (piano)
instrument = miditoolkit.Instrument(
    program=0,
    is_drum=False,
    name="Piano"
)
```

Deux programmes pour commencer (3/4)

```
current_time = 0
current_velocity = 80

i = 0
while i < len(tokens):
    token = tokens[i]

    if token == "BAR":
        # optionnel : on pourrait recalculer current_time
        i += 1
        continue

    elif token.startswith("POSITION_"):
        current_time = int(token.split("_")[1])
        i += 1

    elif token.startswith("VELOCITY_"):
        current_velocity = int(token.split("_")[1])
        i += 1

    elif token.startswith("NOTE_ON_"):
        pitch = int(token.split("_")[2])

        # on s'attend à un DURATION juste après
        next_token = tokens[i + 1]
        assert next_token.startswith("DURATION_")

        duration = int(next_token.split("_")[1])
```

Deux programmes pour commencer (4/4)

```
note = miditoolkit.Note(  
    velocity=current_velocity,  
    pitch=pitch,  
    start=current_time,  
    end=current_time + duration  
)  
  
instrument.notes.append(note)  
i += 2 # NOTE_ON + DURATION  
  
else:  
    i += 1  
  
# ajouter l'instrument  
midi.instruments.append(instrument)  
  
# sauvegarde  
midi.dump("generated.mid")
```

Difficulté : prise en compte du temps

En particulier

- il est possible d'avoir plusieurs notes en même temps
- les notes n'ont pas toutes la même durée

Cependant

- on s'autorise à n'avoir qu'un instrument de musique !
- on peut créer des tokens qui compose les token “note” et “durée”

Pratique pour lire tous les fichiers musicaux

```
import os

for f in os.scandir("."):
    print(f.name)
```

Objectif du projet

A faire

- créer un LLM qui apprend à prédire de la musique
- en entrée, quelques notes et on observe ce qui est généré sur une séquence longue
- par groupe de deux ou trois avec les rendus suivants :
 - le rapport (5 pages maximum)
 - quelques midis générés (5 au maximum)
 - les codes par mail