Génération de mélodies avec un RNN-LSTM - Partie 2 : training

ressource: tuto youtube *Melody generation with RNN-LSTM* de *Valerio Velardo*

In [14]:

```
import os
import music21 as m21
import json
import numpy as np
import tensorflow.keras as keras
```

charger fichier de données et fichier de mapping

In [15]:

```
def load(file_path):
    with open(file_path, "r") as fp:
        song = fp.read()
    return song

songs = load("data/han/file_dataset")

def load_json(file_path):
    with open(file_path, "r") as fp:
        mappings = json.load(fp)
    return mappings

mappings = load_json("data/han/mapping.json")
```

générer séquences d'entrainements

In [16]:

```
def convert_songs_to_int(songs, mappings):
    int_songs = []

    # transform songs string to list
    songs = songs.split()

    # map songs to int
    for symbol in songs:
        int_songs.append(mappings[symbol])

    return int_songs
```

In [17]:

```
def generate_training_sequences(sequence_length):
    """Create input and output data samples for training. Each sample is a sequence.

    :param sequence_length (int): Length of each sequence. With a quantisation at 16th no
tes, 64 notes equates to 4 bars

    :return inputs (ndarray): Training inputs
    :return targets (ndarray): Training targets
    """

    # map songs to int
    int_songs = convert_songs_to_int(songs, mappings)

    inputs = []
    targets = []
```

```
    # generate the training sequences
    num_sequences = len(int_songs) - sequence_length
    for i in range(num_sequences):
        inputs.append(int_songs[i:i+sequence_length])
        targets.append(int_songs[i+sequence_length])

    # one-hot encode the sequences
    vocabulary_size = len(set(int_songs))
    # inputs size: (# of sequences, sequence length, vocabulary size)
    inputs = keras.utils.to_categorical(inputs, num_classes=vocabulary_size)
    targets = np.array(targets)

    return inputs, targets
```

In [19]:

```
inputs, targets = generate_training_sequences(64)
```