Prédiction de “hits” musicaux en utilisant différents types de réseaux de neurones

Guillaume CORNET
guillaume.cornet@student.ecp.fr

Thibaut SEYS
thibaut.seys@student.ecp.fr

CentraleSupélec - Avril 2018

Abstract

Le but de notre projet est de prédire le succès commercial d’une musique, au moyen de deux approches, la première consistant en l’utilisation d’un MLP prenant en entrée les métadonnées (genres, artiste, tonalité, tempo, etc...) d’une musique, et la seconde consistant en un RNN prenant en entrée la représentation de la mélodie d’une musique stockée dans un fichier midi.

Introduction

La notion de “Hit Song Science” est apparue dès 2003 pour désigner l’ensemble des techniques de prédiction de popularité d’une chanson. L’industrie musicale représente 15.7 milliards de dollars à l’échelle mondiale en 2016 [1]. Pouvoir déterminer à l’avance quelles musiques vont marcher est un rêve de beaucoup de maisons de disque. Mais existe-t-il des critères objectifs permettant de prédire quelle musique sera le prochain “hit” ? C’est ce que nous avons cherché à faire dans ce projet. Nous avons voulu comparer une première technique qui ne s’intéresse qu’aux métadonnées sans s’intéresser à la mélodie, et une autre technique fondée uniquement sur la mélodie.

Etat de l’art

De nombreuses techniques ont été utilisées dans le domaine. Une équipe de chercheurs Taïwanais [2] a utilisé pour cela des réseaux de convolution en prenant en entrée un spectrogramme des fréquences obtenu à partir de l’analyse de fichiers chansons audio. Des étudiants de l’université de Stanford [3] ont utilisé différentes approches Machine Learning sur les métadonnées pour déterminer les features caractéristiques de ce qui rend une musique populaire. Ils ont ensuite effectué des classifications et régressions à partir de ces features.

I. Jeu de données

A. Million Song Dataset

Nous avons travaillé sur un extrait du Million Song Dataset.

Celui-ci contient un ensemble de métadonnées rassemblées sur 1 million de chansons. Ces métadonnées sont un agrégat de données, comportant à la fois des données brutes comme le tempo, la tonalité et l’année de sortie, et des indicateurs sur le niveau d’énergie ou de ”danceabilité” calculés sur plusieurs sites de musique comme musicbrainz.org, 7digital et Echo Nest. Ce dernier nous intéresse particulièrement car il contient

un indicateur de popularité, la “song hotttnesss”. Cet indicateur calculé par un algorithme donne une idée de la popularité relative d’une chanson sur une échelle de 0 (peu populaire) à 1 (très populaire). “Seven Nation Army”des White Stripes est un exemple de chanson avec un indicateur de 1.

Dans le cadre de notre approche, nous avions aussi besoin de la partition pour comparer la prédiction basée sur la mélodie et celle basée sur les métadonnées.

B. Lakh midi dataset

Un moyen d’avoir la partition musicale est de travailler sur des fichiers midi. Ce format contient l’ensemble des informations sur les instruments utilisés, les notes jouées ainsi que leur durée, et constitue ainsi une réelle partition virtuelle. C’est la raison pour laquelle nous avons travaillé avec le Lakh midi dataset.

Ce jeu de données comporte un ensemble d’environ 45000 fichiers midi alignés de manière automatique avec les métadonnées du Million Song Dataset. Pour chaque métadonnée d’une musique, quelques fichiers midi ont été matchés avec une certaine probabilité de correspondance. Pour simplifier, nous avons à chaque fois pris le fichier midi avec le plus haut degré de correspondance.

C. Traitement des données

Les métadonnées du Million Song Dataset sont très complètes mais également assez lourdes et stockées dans un format particulier, le hdf5 ou h5. Ce format est assez difficilement exploitable directement en python, nous avons donc dû faire un pré-traitement en sélectionnant à l’avance les champs qui nous semblaient utiles, et générer à partir de ceux-ci un csv qui constitue notre réel jeu de données.

II. Expérimentations sur les métadonnées

Nous avons d’abord commencé par utiliser un MLP pour effectuer une régression sur la “song hotttnesss”.

Les champs que nous avons présélectionnés sont les suivants : id de l’artiste, score de popularité de l’artiste, latitude & longitude (du lieu de naissance de l’artiste), année de sortie, tonalité, mode (majeur ou mineur), tempo, durée, énergie du morceau, “danceability” du morceau (chiffre entre 0 et 1 indiquant la possibilité de danser sur le morceau, évalué par base d’utilisateurs), liste de genres associés à l’artiste.

Nous nous sommes rapidement aperçus que beaucoup de champs n’étaient pas calculés ou remplis directement, notamment le champ qui nous intéresse, la “hotttnesss”.Sur les 31034 musiques de notre jeu de données, il reste environ 14000 musiques avec une “hotttnesss” effectivement calculée. Certains champs comme l’énergie ou la danceability n’étaient pas calculés et nous avons finalement dû les écarter.

Nous avons donc en entrée environ 14000 vecteurs comportant chacun un ensemble de caractéristiques et une valeur de “hotttnesss” que nous cherchons à évaluer.

Sur ce jeu de données, nous prenons 70% pour l’entraînement du modèle, 20% pour le test et 10% pour la validation.

Notre approche consiste en un MLP à 2 couches cachées comportant respectivement 2048 et 1024 neurones. Nous réalisons une régression avec descente de gradient. Il existe différentes méthodes pour cela, la descente de gradient stochastique ajustant le modèle à chaque exemple d'apprentissage, et la descente de gradient classique qui passe en revue tous les exemples d'apprentissage et corrige le modèle à chaque itération. Nous avons utilisé un mode intermédiaire, une descente de gradient par minibatchs. L'avantage de cette méthode est qu'elle évite une convergence prématurée du modèle. Notre fonction de coût était l'erreur quadratique moyenne ou Mean Square Error (MSE).

Nous avons effectué une première estimation en prenant en compte seulement un paramètre, celui qui nous semblait le plus déterminant, qui est le degré de notoriété de l'artiste sur une échelle de 0 à 1. Puis, nous avons affiné les résultats en ajoutant la tonalité, le mode et le tempo. Ensuite, nous avons ajouté le volume sonore et extrait les genres principaux auxquels l'artiste appartient.Les résultats se trouvent dans la partie IV.

Le problème de la régression est que cela est assez éloigné de ce que l'on cherche dans la réalité. On cherche les musiques qui seront potentiellement populaires.

Nous avons donc également réalisé une classification sur la même architecture. La médiane de la valeur de "hotttnesss" se situant aux alentours de 0.5, on a essayé de séparer les chansons peu populaires des chansons populaires.

III. Expérimentations sur les midis

L'expérimentation sur les séquences de midis commencent par la génération des données d'entrée. Pour cela une étude du format midi est nécessaire.

Ce format contient un ensemble de données liées aux rythmes et à la résolution de la musique, ainsi que la séquence à laquelle les notes sont pressées et relâchées. Ce vaste ensemble de données est stockée dans le format midi sous la forme d'événement [4]. Ces événements ont lieu à un "tick" donné. Les "ticks" peuvent être reliés au temps grâce au tempo et à la résolution du fichier. D'autres part le format midi est composée de seize canaux sur lesquelles sont jouées les notes. C'est-à-dire qu'il est possible de ne presser que seulement seize notes à la fois.

Le format midi peut donc être vu comme une séquence de notes pressées sur les 16 canaux différents. Nous allons donc représenter chaque fichier midi comme une séquence des notes jouées sur les 16 canaux à un tick donnée et utiliser un réseaux récurrent pour relier notre séquence de notes à la "song hotttnesss".

Pour entrer un peu plus dans le détail de notre séquence d'entrée, nous allons expliquer quelles sont exactement les données contenues dans notre séquence d'entrée. A chaque "tick" de la piste midi, nous avons trois grandeurs représentant la note qui est jouée à ce instant : sa fréquence, sa vélocité (force à laquelle elle est jouée) et le type d'instrument qui la joue. A cela nous ajoutons les données communes aux seize canaux : le tempo et quatre grandeurs correspondant à la signature musicale du fichier. Pour

chaque “tick” de la musique nous avons donc une dimension d’entrée de 3 x 16 + 1 + 4 = 53.

Le premier problème que nous avons rencontré est le fait que la plupart des fichiers midi que nous avons utilisés sont composés d’un nombre de “ticks” de l’ordre de 90000. Cela représente une séquence de taille trop importante à faire tenir sur la mémoire de notre machine. Nous avons donc réduit la taille de la séquence d’entrée à 2000 “ticks” et nous avons créé des batchs de 100 séquences sur lesquels nous allons apprendre nos réseaux de neurones. Malgré cela, notre temps d’apprentissage reste extrêmement élevé, ce qui a limité nos expérimentations pour cette partie (seulement 5 epochs pour l’apprentissage).

Pour les expérimentations, nous avons choisi de nous concentrer un problème de régression sur la prédiction de la “song hotttnesss”. Pour cela nous avons choisi d’utiliser un réseau récurrent avec en sortie une couche “fully-connected”. La fonction d’activation de cette dernière couche est l’activation linéaire. La grandeur que nous avons choisi d’optimiser est la MSE. Nous avons ensuite utiliser la méthode “grid-search” sur différents hyperparamètres : le nombre de neurones de la partie récurrente (10, 50, 100), le type de la partie récurrente (LSTM ou GRU) et enfin le coefficient de dropout (0.0, 0.2 ou 0.5). Les résultats des expérimentations sont présentés dans la partie IV.

Note : Nous avons également essayé une autre approche de représentation de la séquence du fichier midi utilisée par différents projets de génération musicale [5, 6], mais cela n’aboutit pas à une baisse de la mémoire utilisée par les séquences d’entrées tout en limitant le nombre d’information utilisée du fichier midi.

IV. Résultats

A. Utilisation des métadonnées

La quantité que nous avons regardée pour la régression est la MSE. à titre de comparaison, la MSE entre notre ensemble de validation et une répartition aléatoire vaut 0.124. La variance de l’ensemble de validation vaut 0.0371.

On obtient pour la régression avec l’ensemble des paramètres une MSE de 0,0293.

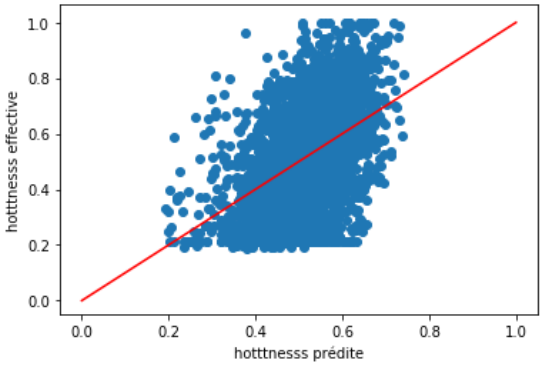

FIGURE 1 - *Répartition des valeurs sur l’espace (valeur prédite, valeur réelle)*

On peut voir que notre modèle n’est pas nécessairement très précis mais que la répartition est quand même centrée sur la droite d’équation y = x. On obtient des résultats meilleurs que l’aléatoire ou qu’en centrant l’estimation sur la moyenne de l’ensemble de validation.

En effectuant plusieurs tests avec différents paramètres, nous nous sommes aperçu que le paramètre artist_familiarity était déterminant dans la prédiction de la valeur de hotttnesss, ce qui n’est pas étonnant puisque la notoriété d’un artiste influe beaucoup sur le succès de ces

musiques, mais ce qui est également problématique dans notre but qui est de juger une musique à sa valeur seule.

Nous avons donc tenté de faire une autre régression en supprimant ce paramètre.

Nous avons obtenu une MSE de 0,0347 et la répartition suivante :

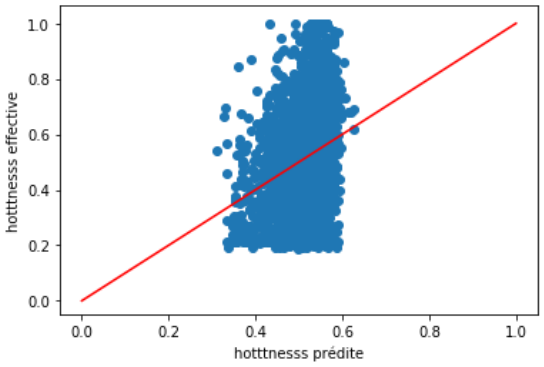

FIGURE 2 - *Répartition des valeurs sur l'espace (valeur prédite, valeur réelle)*

On voit que la performance diminue de manière notable, cependant on reste meilleur qu'une répartition aléatoire ou constante en la moyenne de l'ensemble de validation puisque la variance de l'ensemble vaut 0.0371.

Pour la classification, on obtient pour l'ensemble des paramètres une précision de 66,5%, ce qui est correct.

B. Utilisation de fichiers midis

Nous avons utilisé pour cette partie un ensemble d'entraînement de 11100 données, un ensemble de tests de 3000 données et un ensemble de validation de 1500 données, ce qui correspond à une répartition d'environ 70% - 20% - 10%. Voici un tableau récapitulatif de la méthode grid search utilisée sur la MSE obtenue sur l'évaluation de l'ensemble de test :

		Type de couches	
Dropout	**Neurones**	LSTM	GRU
0.0	10	0.072	0.069
0.2		0.068	0.069
0.5		0.072	0.073
0.0	50	0.068	0.069
0.2		0.068	0.070
0.5		0.068	0.069
0.0	100	0.068	0.068
0.2		0.068	0.068
0.5		0.068	0.070

Le faible écart entre les différentes MSE nous apprend que nous sommes en réalité en situation de sous-apprentissage. Nous pouvons le vérifier en regardant la différence entre les prédictions et les valeurs réelles de la "song hotttness" sur l'ensemble de validation sur le réseaux LSTM avec un nombre de neurones de 100 et un dropout de 0.

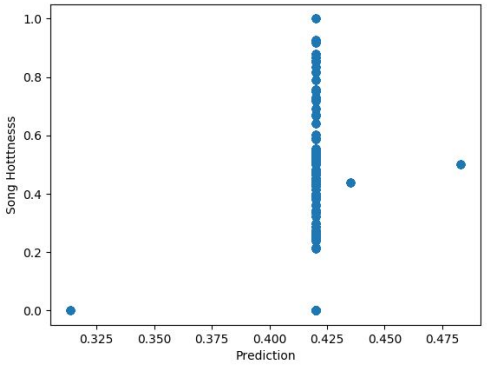

FIGURE 3 - *Répartition des valeurs sur l'espace (valeur prédite, valeur réelle)*

Dans notre approche, la prédiction de la “song hotttnesss” à partir du fichier midi est un échec. Nous pensons que cela peut être dû aux raisons suivantes :

- La taille de séquence de 2000 “ticks” pour un fichier midi est beaucoup trop petite par rapport à la taille réelle du fichier.

- Notre grid search n’a pas assez balayé l’espace des hyperparamètres, notamment au niveau du nombre de neurones de la partie récurrente du réseau.

- Notre nombre d’epochs utilisé pour l’apprentissage est trop faible.

- Nous n’avons pas choisi la bonne représentation du fichier midi en entrée de notre réseau.

Conclusion

Nous savions d’ores et déjà avant ce projet que l’évaluation d’une musique n’était pas une science exacte et que tenter de prédire le succès d’une musique sur des critères “objectifs” était une opération délicate, nous en avons eu la confirmation. Bien que nous ayons eu des résultats meilleurs que le hasard ou la “chance”, ils ne sont pas assez significatifs pour prédire de manière précise si une chanson sera le prochain “hit” de l’été. La musique reste une caractéristique de ce qui nous rend humain.

D’autre part, il semble beaucoup plus facile, et moins gourmand en ressource, de prédire la “hotttnesss” d’une chanson à partir de ses metadonnées plutôt qu’à partir de sa séquence de notes. Il semblerait donc à première vue que le succès d’une musique soit plus lié à des facteurs externes qu’aux séquences de notes qui la composent.

Références

[1] http://www.snepmusique.com/wp-content/uploads/2017/04/Global-Music-Report-2017.pdf

[2] L. Yang, S. Chou, J. Liu, Y. Yang, Y. Chen. Revisiting The Problem of Audio-Based Hit Song Prediction Using Convutional Neural Network. 2017.

[3] J. Pham, E. Kyauk, E. Park, Predicting Song Popularity, 2015.

[4] http://www.somascape.org/midi/tech/mfile.html

[5] https://github.com/hexahedria/biaxial-rnn-music-composition

[6] https://github.com/burliEnterprises/tensorflow-music-generator