Classification de genre musical sur des données multimodales

Saad TAZROUTE

saad.tazroute@centrale-marseille.fr

2 février 2022

Table des figures

Liste des tableaux

Table des matières

1 Introduction

Récemment, l'apprentissage automatique a trouvé ses applications dans un ensemble diversifié de domaines. L'art ne fait pas l'exception. Plusieurs recherches ont été articulés autours de plusieurs types d'art et notamment notamment la musique.
Pour traiter le problèmes de classification dans le domaine de musique, plusieurs approches peuvent être utilisées, à savoir, des approches mono-modales.
Les approches mono-modales concernent l'utilisation d'un seul type de données pour réaliser la tâche. Les approches multi-modales vient pour réaliser une combinaison de l'information et on peut utiliser plusieurs types de données, à savoir : Audio, Texte, Image.
Une autre manière de voir les choses et de faire la disctinction entre l'apprentissage end-to-end où les features sont apprises au fur et à mesure dans une couche du réseau, et l'apprentissage à base de features déjà extraits + un mlp ou un algorithme de ML.
Dans ce projet nous réalisons la classification de genre musicale à base de données sonores(musique), et des images (pochettes d'album).
Nous nous basons sur un apprentissage à base de features déjà extraites à savoir : MFCC features et les features apprises par un réseau profond, et les images de pochettes d'albums.
Nous réalisons une approche mono-modale pour les 2 types de données fournies : Features mfcc et les images.

2 Jeux de données utilisés

Le fichier **msdimapping.csv** est le fichier principal, à partir duquel on peut charger toutes les données, et qui contient les informations relatives à chaque entrée.
Nous nous basons sur la manipulation de ce fichier pour l'extraction des différents informations qui nous seront utiles :

- charger les images
- charger les labels
- charger les features MFCC

Pour les différentes expériences, nous contruisons notre couple (X, y) pour notre tâche d'apprentissage supervisé à base de ce fichier.

3 Cadre Experimentale :

Nous réalisons 3 expériences de classification mono-modales :

3.1 Classification des images de pochettes

Nous utilisons pour ceci un réseau pré-entraîné sur un jeu de données gigantesque, qui lui permettra d'extraires des features représentatives.
Nous utilisons **VGG-19** pré-entraîné sur le jeu de données **ImageNet**.
Nous stackons à la dernière sortie de pooling un pooling,une normalisation, et une couche dense dans cet ordre.
Nous rapportons les résultats

3.2 Classification de la musique à base des features MFCC

Pour la tâche de classification d'audios, nous reposons dans cette expériences sur des features MFCC. Chaque morceau est associé à un matrice de taille $(N \times d)$, où N est le nombre de trames, et d nombre d'extraits sur le nombre de trames.
Le problème de longueur différents de ces features, nous pose problème pour les donners à notre modèle.
Nous avons réalisé une étude sur la longueur, à travers un histogramme présentant le nombre de fichiers par taille.
Nous avons pu constaté qu'on peut tronquer à une longueur maximale autour de 1024 sans perdre beaucoup d'informations, et nous virons les features les moins courts de cette longueur caractéristique.

3.3 Modèles utilisés

Nous réalisons 3 expériences dans ce rapport :

— Classification de pochettes d'albums, à base d'un VGG pré-entraîné sur Imagenet et on stack une couche dense.
— Classification de son à base features MFCC de longueur 1024 (après troncature) : Nous utilisons un réseau convolutif en 1 dimension pour faire la classification.
— Même tâche pour les features de longueur 500.

3.4 Training

Nous réalisons l'entraînement en totalité en local sur la configuration suivante :

— CPU : AMD Ryzen 5600H
— GPU : RTX 3060 Portable 6GB
— RAM : 8 GB DDR4

3.5 Fonctions utiles

Nous avons utilisé des fonctions que j'ai codé pour faciliter la mnipulation de la donnée et surtout pour mener une pipeline de classification.

— **get-X-y** : fonction qui permet de loader les vecteurs **X** des données et **y** des étiquettes pour l'apprentissage, et dépend de chaque notebook, par exemple dans le notebook de classification nous chargeons des images dans le vecteur X et les labels en y.
— **truncate-and-delete-short-samples** Fonction qui permet de tronquer les features **MFCC** et supprimer les features trop court par rapport à une longueur qu'on a fixée.
— **Plot-loss** : réalise le tracé de courbes d'apprentissage.

3.6 Encoding des labels

Nous avons **15 labels en total**, indiquant le genre musicale du morceau, de l'image de l'album ..
Nous avons réalisé un Label encoding sur nos étiquettes à prédire.

4 Résultats

Nous rapportons les résultats obtenus lors des expériences :

Classification de pochettes d'album : VGG + couche de classification

Evaluation	F1-score	Accuracy	Precision	Recall
Score	0.45	0.425	0.435	0.47

TABLE 1 – Métriques d'évaluation sur 30 epochs avec SGD comme optimiseur

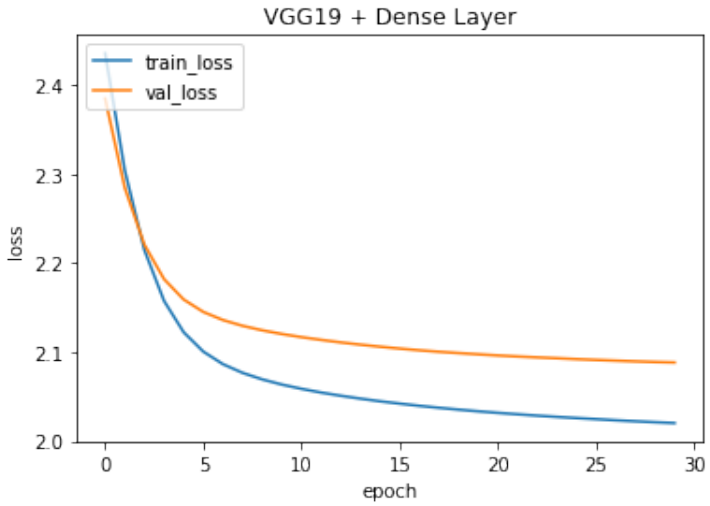

FIGURE 1 – Courbe d'apprentissage pour le modèle utilisé

Nous constatons que l'apprentissage est rapide, et il n'y a pas d'overfitting.
Les résultats pour la classification d'images, reste raisonnable puisque touts les métriques sont autour de 45% ce qui est déjà beaucoup plus mieux qu'un système au hasard 1,5% pour le même système avec des poids aléatoires.

Classification de son à base de features MFCC tronqué à 500

Le jeu de données utilisés dans cette partie :

— 2658 échantillons de test

— 6705 d'échantillons pour l'entraînement

Evaluation	F1-score	Accuracy	Precision	Recall
Score	0.38	0.39	0.43	0.32

TABLE 2 – Métriques d'évaluation sur 60 epochs avec un learning rate de 0.0001

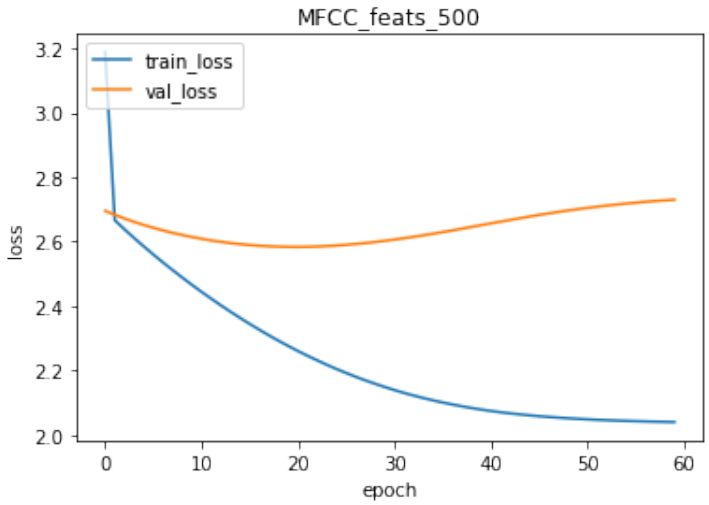

FIGURE 2 – Courbe d'apprentissage pour les features de taille 500

Nous constatons le début d'overfitting du modèle sur les données d'entrainement à partir de l'epoch 30.
Le modèle est le même utilisé pour les autres features de taille 1024, mais avec beaucoup plus d'informations.

Classification de son à base de features MFCC tronqué à 1024

Le jeu de données utilisés dans cette partie :

— 1381 échantillons de test

— 6572 d'échantillons pour l'entraînement

Evaluation	F1-score	Accuracy	Precision	Recall
Score	0.47	0.46	0.46	0.47

TABLE 3 – Métriques d'évaluation sur 60 epochs sur des MFCC tronqués à 1024 avec un learning rate de 0.0001

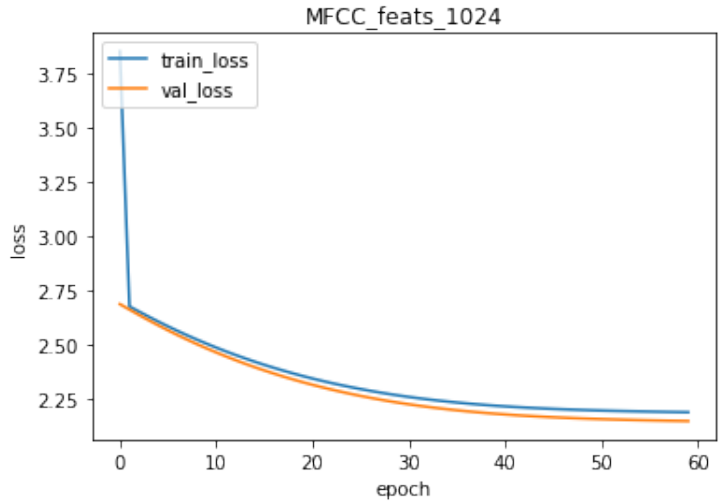

FIGURE 3 – Courbe d'apprentissage pour les features de taille 500

L'apprentissage ne semble pas se saturer à l'epoch 60, nous arrêtons l'apprentissage à cause de limitations matérielles.
Les résultats pour la classification de son reste pas mal également et surtout pour les features MFCC à longueur 1024 qui sont autour de 3x% et qui dépasse le même système mais cette foids ci avec des poids initialisés au hasard (avant la méthode .fit) qui donne des résultats autour de 7%.

5 Discussion

Parmi les difficultés que j'ai rencontré dans ce projet, c'était la gestion de mémoire, avec une RAM de 8GB j'ai pas pu faire grande chose sans faire appel aux techniques de clean-up de la mémoire.

Ce problème est un problème hardware/software, qu'on appelle le "Memory Leak" où la mémoire vive ne se vide pas et il faut faire une gestion très raisonnable quant à ceci.
J'ai utilisé **htop** et **nvtop** pour monitorer ma consommation, et l'outil Garbage Collector pour le nettoyage de la mémoire au cours du calcul.