Ecole Nationale d'Ingénieurs de Tunis

Mastère Traitement de l'Information et Complexité du Vivant

Algorithmes d'Apprentissage et Application Audio :

Algorithme du gradient stochastique avec momentum

Elaboré par :
Elatrach Mayara
Fezai Lobna
Belaid Mouna

Table des matières

Résumé

L'algorithme de descente de gradient est considéré parmi les algorithmes les plus performants en optimisation. Cet algorithme a évolué d'une façon particulière. Il s'agit notamment de la technique fondatamentale de la machine learning. Ces algorithmes connaissent toujours un très grand intérêt et ils sont largement utilisés en apprentissage profond de l'intelligence artificielle. Il existe 3 variantes de cet algorithme selon la taille de la structure des données disponibles. Il s'agit bien de la descente de gradient par lots, la descente de gradient stochastique et la descente de gradient par petits lots. L'objectif est toujours de minimiser une fonction tout en controlant la variance avec une mise à jour de nos paramètre jusqu'au atteindre la convergence. Dans le contexte de ce projet, on s'intéresse au algorithme de gradient stochastique faisant intervenir le terme momemtum qui sert principalement à améliorer la vitesse de convergence de l'algorithme.

Mots clés : Momentum, Descente de gradient, Taux d'apprentissage, Vitesse de convergence.

1 Introduction

On cherche toujours à minimiser ou à maximiser une fonction dans des divers domaines. Cependant, la résolution d'une telle problématique n'est pas toujours évidente étant donné quand on se trouve parfois avec un grand nombre de paramètres. A cet égard, il y a un recours à une approximation avec une approche itérative qui Pensons tout d'abord, pour résoudre ce problème définir la structure et le critère d'optimisation.
L'algorithme du gradient désigne un algorithme d'optimisation différentiable.
C'est un algorithme d'optimisation utilisé pour minimiser une fonction en se déplaçant de manière itérative dans la direction de la descente la plus profonde, définie par le négatif du gradient. Il existe trois variantes de cet algorithme : la descente de gradient par lots, la descente de gradient stochastique et la descente de gradient par petits lots.

Dans le cadre de notre travail, nous mettons le point sur l'algorithme de gradient stochastique faisant intervenir le terme momemtum qui sert principalement à améliorer la vitesse de convergence de l'algorithme.

Dans une première partie, nous introduisons l'algorithme de descente de gradient en donnant une idée générale sur son fonctionnement. Dans une deuxième partie, nous focalisons notre intérêt sur l'algorithme du gradient stochastique avec le terme momentum. Nous indiquons la raison pour laquelle nous nous sommes intéressées à cet algorithme. Nous évoquons aussi ses performances et ses limites. Finalement, nous présentons une simulation de cet algorithme.

2 Présentation de l'algorithme de descente de gradient

Les modèles de réseaux de neurones ont été appliqués avec succès à un large éventail de problèmes. Bien qu'il y ait plusieurs types d'algorithmes d'apprentissage disponible, la majorité d'entre eux, y compris le populaire algorithme d'apprentissage par rétro-propagation (back-propagation), sont du type à descente de gradient. [2]

L'algorithme de descente de gradient a évolué au cours de temps et des multiples variantes sont maintenant offertes par des outils informatiques et de plateformes en ligne notamment TensorFlow. Mais, le choix entre ces variantes n'est pas évident et dépend de l'objectif et la taille de données. Le principe consiste à procéder selon une approche itérative et adaptative qui nous mène à la solution. Le défit à relever est comment définir la "meilleure" trajectoire ou la trajectoire optimale pour atteindre le minimum. La mise à jour des paramètres d'une itération à une autre détermine la finesse et la vitesse de la convergence.

On définit le critère d'optimisation, la surface d'erreur $J(\Theta)$. C'est une fonction d'erreur qui est paramétrée par les poids qui représentent les forces de connexion entre les unités entrantes. Dans un algortihme adaptative, le gradient de cette surface d'erreur par rapport à chaque poids est ensuite calculé et les poids sont modifiés selon la direction descendante de la pente afin de réduire l'erreur.

L'équation de l'algorithme en sa forme classique est la suivante :

$$\Theta = \Theta - \mu \bigtriangledown_{\Theta} J(\Theta) \tag{1}$$

Où μ est le pas d'adaptation et $\mu \bigtriangledown_{\Theta} J(\Theta)$ le terme incrémental suivant le sens du gradient décroissant. $\bigtriangledown_{\Theta}$ représente l'opérateur de gradient par rapport aux poids et μ est un petit nombre positif appelé taux d'apprentissage déduit des valeurs propres de la matrice composée par les données.

Cette approche est lente vu qu'il s'agit d'un seule mise à jour de notre paramètre. D'ou l'idée de l'aspect stochastique SGD qui met à jour le paramètre pour chaque entrée d'apprentissage x^i et de classe y^i. L'algorithme itératif est donc :

$$\Theta = \Theta - \mu \bigtriangledown_{\Theta} J(\Theta; x^i; y^i) \tag{2}$$

Quoique cet algorithme est amelioré, il risque de ne pas converger et de rester en état d'oscillation autour de notre solution.
La solution proposée est de passer à un algorithme SGD en mini-lots qu'on fixe :

$$\Theta = \Theta - \mu \bigtriangledown_{\Theta} J(\Theta; x^{i:i+n}; y^{i:i+n}) \tag{3}$$

3 Algorithme du gradient stochastique avec momentum

3.1 Intérêt

Il existe plusieurs autres variantes de l'algorithme du gradient stochastique à savoir Adagrad, Adadelta, RMSprop, Adam, etc.
On s'est intéressé à l'algorithme du gratdient stochastique avec momentum car il présente une première amélioration du gradient de la desente en terme de convergence.

3.2 Performances l'algorithme

Dans cette partie, on s'interesse à la convergence et au pas d'adaptation μ. L'inclusion d'un terme momentum augmente considérablement la vitesse de convergence. On fait intervenir le vecteur déviation composé d'un terme qui utilise le résultat de l'itération antérieure pondérée par le terme momentum. Ainsi, l'équation (3) devient sous cette forme :

$$\begin{aligned} \upsilon_t &= \gamma \upsilon_{t-1} + \mu \bigtriangledown_\Theta J(\Theta) \\ \Theta &= \Theta - \upsilon_t \end{aligned} \tag{4}$$

Où γ est terme momentum. Cela montre que la modification du vecteur de poids au temps donné de l'étape actuelle dépend à la fois du gradient actuel et du changement de poids dans l'étape précédente. Intuitivement, il s'agit d'une accumulation de momentum. Le système oscille donc dans une direction et ne se déplace que très lentement le long de l'axe vers le minimum. Le terme momentum aide à moyenner l'oscillation tout en additionnant des contributions le long de la trajet vers le minimum comme l'illustre cette figure.

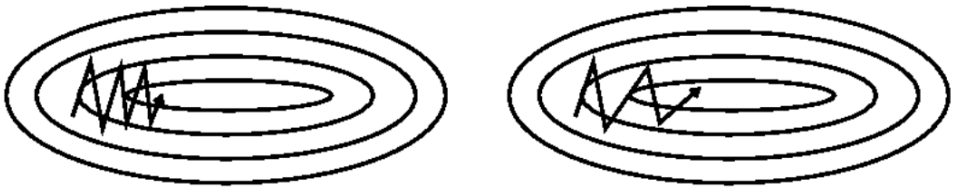


FIGURE 1 – A gauche—SGD sans momentum, right—SGD avec momentum [1]

On note que l'algorithme converge plus vite quand les données d'entrée sont non correlées [4].

3.3 Limites de l'algorithme

Dans la plupart de cas, la desente de gradient ne se dirige pas directement vers le minimum. Effectivement, SGD avec Momentum réduit les oscillations et accélère la convergence. Cependant, en s'approcant du minimum global, la vitesse du Momentum est tellement forte qu'il peut

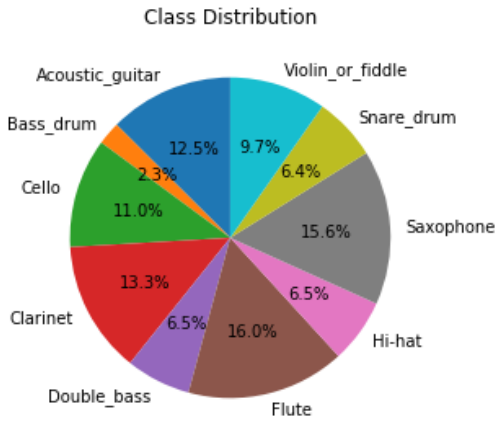
rater le minimum objectif. Ce problème est à la base de l'algorithme de correction de Nestrov qui sert à ralentir le Momentum. On cherche dans la suite des algorithme plus adaptatifs.

4 Simulation[3]

Dans ce projet, on a pris une partie d'une base do données publiée dans Kaggle dans le cadre d'une compétition. La partie qu'on a pris concerne des enregistrements de 10 instruments différents. On a focalisé notre travail qui suit sur les instruments musicaux dans cette base afin de réduire sa taille de dizaines de Go à 128 Mo.

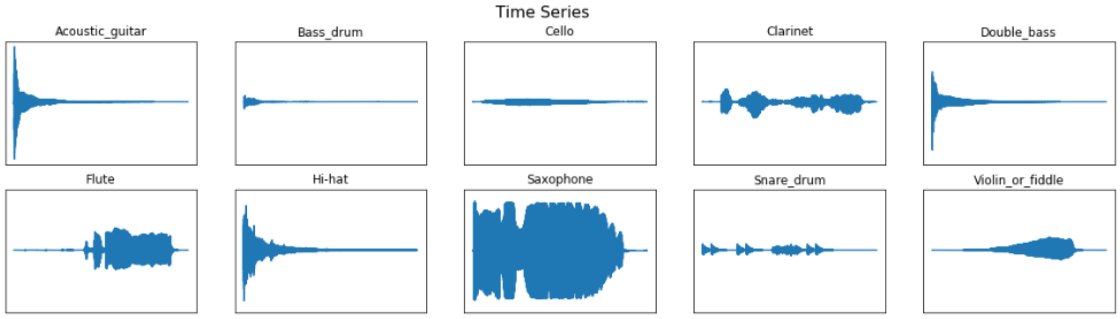
Out[9]:

fname	label	length
5388d14d.wav	Saxophone	4.14
c685f05f.wav	Saxophone	1.54
36d20ab5.wav	Saxophone	4.90
d6665734.wav	Saxophone	10.50
7352e28f.wav	Saxophone	6.24

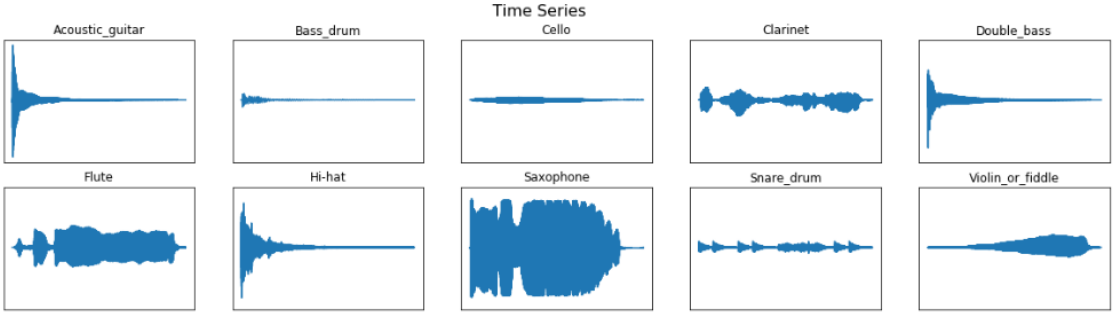


La figure représente quelques enregistrements et leurs longueurs et la figure à gauche représente la distribution associée à chaque instrument

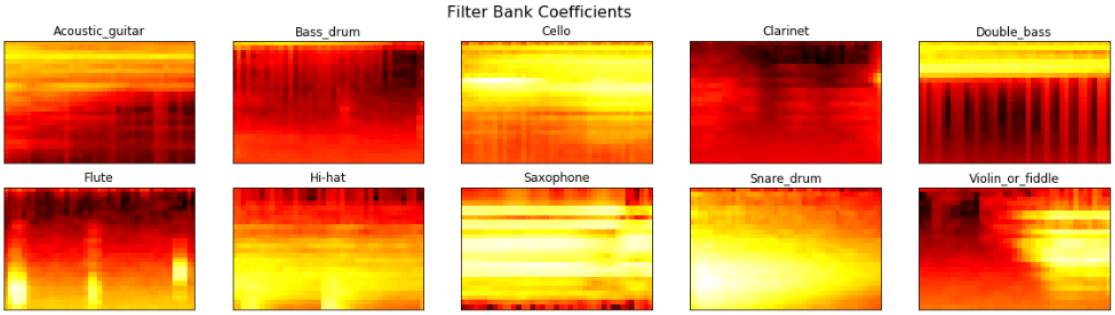
Avant de faire la classification, on a opté pour une étape de nettoyage de la base de données. On a utilisé pour cela une enveloppe de masquage et on a supprimé les fréquences sous le seuil du masquage. Ainsi, on a obtenu une nouvelle base avec moins de bruit.



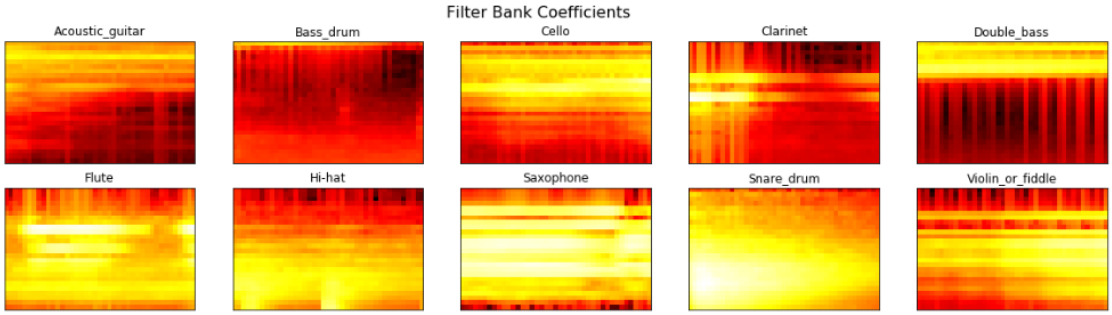
Cette figure représente l'amplitude en moyenne de chaque instrument en fonction du temps des données initiales



Cette figure représente l'amplitude en moyenne de chaque instrument en fonction du temps après l'étape dea nettoyage



Cette figure représente le banc de filtrse de chaque instrument des donneés initiales



Cette figure représente le banc de filtrse de chaque instrument après le nettoyage

Par la suite, on a défini une fonction de répartition de ces nouvelles données. Chaque probabilité d'avoir un instrument est définie par la fréquence d'avoir cet instrument divisé par le nombre total des données.
Cette probabilité est utilisée afin de sélectionner les entrées au réseau de neurones.
On propose dans notre cas d'utiliser un réseau de neurones convolutifs (CNN) avec 4 couches de convolution et 3 couches entièrement connectées. L'optimiseur qu'on propose est le gradient stochastique de type Momentum. Dans l'annexe se trouve le code qu'on a utilisé.

5 Conclusion

Dans ce projet on a focalisé notre intérêt sur l'algorithme du grandient descendant de type Momentum. Il est plus efficient que le gradient descendant classique en terme de convergence. On a essayé d'appliquer cet algorithme comme optimiseur de réseau de neuronnes covolutif. Pour cela on a pris une base de données de Kaggle. On a nettoyé cette base puis on l'a insérée dans le modèle. Ce projet nous a permi de découvrir plusieurs types d'algorithmes d'optimisation et de mettre l'accent sur l'algorithme du grandient descendant de type Momentum.

Références

[1] "Momentum and Learning Rate Adaptation", En ligne, consulté le 06/02/2019. https://www.willamette.edu/~gorr/classes/cs449/momrate.html.

[2] Sebastian Ruder, An overview of gradient descent optimization algorithms, 2017, Insight Centre for Data Analytics, NUI Galway Aylien Ltd., Dublin.

[3] Seth Adams, Loading Data - Deep Learning for Audio Classification", 2018,Chaine youtube.

[4] Jaidane Meriem, Algorithmes d'apprentissage, Applications Audio, 2018.