Travail d'initiative personnelle encadré

Descente de gradient stochastique

Benjamin LOISON (MPSI 1)

2018-2019

1 Introduction intuitive

diapositive

L'**objectif** de mon TIPE est d'obtenir un algorithme capable lorsqu'on lui fournit une image comportant des poissons, de les dénombrer et d'identifier leurs espèces. Cette intelligence artificielle serait basé sur un **réseau de neurones**. Un outil intermédiaire est la **descente de gradient**.
L'algorithme du gradient stochastique est un **entraînement de paramètres** sur une base de données. Après entraînement on peut utiliser ces paramètres afin de faire une **prédiction**. Je vais décrire dans un premier temps comment on procède à l'entraînement.

diapositive

Commençons par nous faire une **intuition** de ce qu'est la descente de gradient. De façon imagé, en considérant les axes (Ox) et (Oy) et une courbe représentative d'une fonction, la méthode de descente de gradient consiste alors à **calculer le minimum de cette fonction**.
La technique de la descente de gradient consiste par ses conditions initiales fixés par nous-même, à placer une **bille subissant d'une certaine manière la gravité sur la courbe**, par exemple en M_0. On travaille ici par exemple avec la fonction polynomiale: $x \longrightarrow x^4 - 3x^3$. On définit aussi dans les conditions initiales, la norme de la **vitesse** de la balle ce que l'on appellera par la suite un **taux d'apprentissage** α. L'algorithme du gradient consiste alors à l'**itération** du processessus de chute, ainsi on obtient les points M_1 et M_2.
On peut facilement prédire et constater que le point M_3 ne va pas se trouver plus proche que M_2 par rapport au minimum recherché.

diapositive

Le point M_4 se situe à nouveau sur la branche de droite de la courbe car la descente de gradient consiste "à descendre sur la courbe", c'est-à-dire à se diriger dans la direction où le vecteur gradient est le plus petit. Sachant que le vecteur gradient est de manière infinitésimale le vecteur le plus petit en norme permettant de maximiser la croissance de la courbe.
La suite des points à partir de M_3 ne semble pas intéressante car elle ne semble pas a priori converger vers le minimum.

Ce **problème** vient du taux d'apprentissage que l'on voudrait donc **faire diminuer petit à petit en s'approchant du minimum**, le problème c'est que l'**on ne connait pas où est alors le minimum**.

diapositive

Une technique est alors de choisir un taux d'apprentissage qui pour les mauvaises prédictions va tendre à diminuer **en prenant sucessivement les valeurs d'une suite strictement décroissante**, tel que: $\left(\frac{1}{n+1}\right)_{n\in\mathbb{N}}$

diapositive

2 Principe de l'algorithme

J'ai cherché tout d'abord à m'entraîner sur la base de données standard **MNIST** avant de considérer un cas réel avec des photos de poissons même si je vais montrer que comparer des photos de chiffres ou de poissons en apprentissage automatique, c'est presque la même chose. Effectivement cette base de données est constituée d'**images** de 28 sur 28 pixels **de chiffres écrits à la main** en noir sur blanc avec un domaine de couleurs de blanc et noir allant de 0 à 1 par pas de 0.01 sachant que 0 désigne le blanc et 1 le noir.
J'ai essayé donc dans un premier temps de faire un algorithme de **classification binaire**, c'est-à-dire un algorithme capable de dire si oui ou non une image donnée correspond par exemple à un 6. La **sortie de l'algorithme étant en pratique un réel** entre 0 et 1. **Plus le réel est proche de 1** et plus on est sûr que l'image considéré correspond à dans mon exemple à un 6.
Après avoir codé cette classification binaire, il est facile de **faire 10 fois l'algorithme précédent** de classification binaire afin de successivement savoir si l'image donnée en entrée est un 0 ou pas, puis si c'est un 1 ou pas et ainsi de suite. On peut alors prédire par rapport à une image quel est le chiffre écrit en prenant le chiffre associée à une sortie la plus proche de 1.

diapositive

On peut interpréter la descente de gradient comme la **convergence de la droite passant au plus des points de l'espace**, où l'espace est ici de dimension 28^2 et les coordonnées des points représentent successivement les valeurs entre 0 et 1 de chaque pixel.
On travaille sur un **chiffre donné**, noté n.
- On rappelle que α est le taux d'apprentissage et il **peut dépendre de l'itération considérée**.
- On pose m le nombre de photos étudiées sachant que la **base de données d'entraînement de MNIST comporte 60 000 photos munis du chiffre associé**.
Soit $j \in [\![1; 28^2]\!]$,θ_j **est le j-ème paramètre** définissant la droite passant au plus près des points.
$h_\theta(x^{(i)})$ est la **prédiction** de notre algorithme pour la i-ème photo $x^{(i)}$.
$y^{(i)}$ vaut **0 ou 1**, 1 si la photo correspond à un n et 0 sinon.
On a $h_\theta(x^{(i)}) = \sigma(\theta_1 + \theta_2 x_2 + ... + \theta_{28^2} x_{28^2})$, avec σ la fonction **bijective sigmoïde** de $\mathbb{R}$ dans $[0; 1]$, définie par: $\sigma(x) = \frac{1}{1+e^{-x}}$
- On a clairement $\sigma(0) = 0.5$ et $\sigma(x) \geq 0.5$ si $x \in \mathbb{R}^+$ et $\sigma(x) \leq 0.5$ si $x \in \mathbb{R}^-$
L'algorithme du gradient consiste à **minimiser la fonction de coût J utilisant la méthode des moindres carrés** c'est-à-dire l'erreur de notre algorithme, avec $J(\theta_1, \theta_2, ..., \theta_{28^2}) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$.

diapositive

Lire dérivation...

diapositive

3 Algorithme, résultats et critique

Expliquer brièvement le code.

diapositive

18.3 % d'erreur peut sembler être important mais il faut se rappeller qu'il ne faut pas comparer avec une **espérance** intuitive de 50 % en répondant de manière aléatoire. Effectivement la véritable espérance en répondant de manière aléatoire est de 10 %.
Autrement dit mon algorithme est presque 8 fois plus efficace qu'un algorithme qui répondrai de façon aléatoire.
On peut donc parler d'intelligence artificielle et plus précisément d'algorithme de machine learning supervisé.
D'ailleurs l'information importante vis-à-vis de ce 18.3 % d'erreur est que ce taux d'erreur est atteint pour une **base de données de test**, c'est-à-dire sur des données que l'algorithme n'a jamais rencontré auparavant, chaque entrée peut alors être considéré comme une question que l'on poserai à l'algorithme et il répond environ 4 fois sur 5 le bon chiffre.
Considéré une droite dans un espace de dimension 28^2 peut sembler premièrement être une bonne idée pour s'approcher d'une relation permettant de déterminer quel chiffre est écrit sur la photo mais le modèle affine peut ne pas passer par tous les points même avec les meilleures paramètres θ. Un **modèle polynomiale** peut par exemple être plus approprié. On peut d'ailleurs remarquer que je n'ai pas utilisé de **taux d'apprentissage adapatatif** ce qui peut améliorer l'efficacité de mon algorithme.
Dans l'algorithme on procède à un balayage sur les θ puis pour chaque θ on procède à un balayge sur les images. On peut se demander si en **inversant l'ordre des boucles** pour cela n'améliorerait pas ou si cela est équivalent.

4 Optimisation temporelle

diapositive

Avec la bibliothèque **numpy** en Python, on aurait pu procéder à un calcul de h_θ avec un produit de vecteurs.

diapositive

En **C++ avec des threads**, ce langage de plus bas niveau et équipé de calcul parallèle pour les 10 chiffres séparément on passe d'un temps d'éxécution sur la **base de données d'entraînement de 2 heures en Python sans threads à 60 secondes en C++ avec des threads** ! On aurait pu envisagé d'utiliser la puissance d'une **carte graphique** afin d'effectuer tous ces petits calculs avec un langage tel que OpenCL.

5 Conclusion

Maintenant que la descente de gradient est familière on peut s'intéresser au **problème initial des poissons**. On peut le réduire à une descente de gradient à condition de **dénombrer et isoler graphiquement** avec par exemple la bibliothèque OpenCV les zones contenant des poissons et appliquer la descente de gradient sur ces images. Effectivement la descente de gradient ne fait en soit **pas de distinction entre des photos de poissons ou des photos de chiffres** puisque l'on peut attribuer un numéro à chaque espèce de poissons. Il faudrait alors considérer des **tableaux de dimension trois fois plus grandes** afin de considérés les couleurs rouge, verte et bleue. Il faudrait aussi **redimmensionner les images** avec un algorithme d'interpolation adéquat les photos de différentes tailles de poissons.