ADAM: A method for Stochastic Optimization
Résumé d'article

Guillaume BERNARD-REYMOND, Guillaume BOULAND,
Camille MOTTIER, Abel SILLY

26 septembre 2024

Table des matières

1 Introduction

L'article étudié ici, intitulé « ADAM : A method for Stochastic Optimization », a été écrit en 2015 par Diederik P. Kingma (Université d'Amsterdam, OpenAI) et Jimmy Lei Ba (Université de Toronto), dans le cadre d'une conférence à l'International Conference on Learning Representations (ICLR).

Cet article présente l'algorithme Adam, algorithme d'optimisation stochastique, basée sur une descente de gradient, dans le cadre d'un espace de paramètres à grande dimension. Outre le fait que cet algorithme est simple à implémenter, efficace computationnellement et nécessite peu de mémoire, il offre une méthode qui marche bien dans un large panel de cas, y compris dans les cas problématiques de gradients éparses ou de fonctions-objectifs non stationnaires. En cela, il combine les qualités d'algorithmes existants au préalable, tels que AdaGrad et RMSProp.

Cet article présente une description précise de l'algorithme Adam, fournit un résultat de convergence de la méthode et détaille l'apport de l'algorithme Adam vis-à-vis d'autres algorithmes.

2 Algorithme

On considère une fonction-objectif stochastique $f(\theta)$ de paramètres θ, qu'on suppose différentiable. On souhaite optimiser les paramètres θ afin de minimiser l'espérance $\mathbb{E}[f(\theta)]$.
Outre la fonction $f(\theta)$, l'algorithme nécessite la donnée d'un pas α, de taux $\beta_1, \beta_2 \in [0, 1[$, d'une constante de stabilisation numérique $\varepsilon > 0$ et de paramètres initiaux θ_0. Il exécute alors le schéma suivant :

Algorithme 1 : Adam

Entrées : $f(\theta)$, α, β_1, β_2, ε, θ_0
$m_0 \longleftarrow 0$
$v_0 \longleftarrow 0$
$t \longleftarrow 0$
tant que θ_t *ne converge pas* **faire**
  $t \longleftarrow t + 1$
  $g_t \longleftarrow \nabla_\theta f_t(\theta_{t-1})$
  $m_t \longleftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$
  $v_t \longleftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$
  $\widehat{m}_t \longleftarrow m_t / (1 - \beta_1^t)$
  $\widehat{v}_t \longleftarrow v_t / (1 - \beta_2^t)$
  $\theta_t \longleftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \varepsilon)$
fin
Sorties : θ_t

Pour comprendre cet algorithme et identifier les apports de la méthode Adam, nous pouvons le mettre en relation avec d'autres algorithmes de descente de gradient stochastique, présentés dans l'annexe. Nous pouvons maintenant interpréter les étapes de l'algorithme Adam et identifier les nouveautés.

- $m_t \longleftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$

  Fournit une estimation de $\mathbb{E}[g_t]$ par moyenne mobile à décroissance exponentielle : $m_t = (1-\beta_1) \sum_{i=1}^{t} \beta_1^{t-i} \cdot g_i$.

  Permet de garder mémoire des gradients précédents pour atténuer les variations.
  On trouve une idée similaire dans l'algorithme SGD avec moment.

- $v_t \longleftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$
  Permettra, lors de la mise à jour des paramètres, une mise à l'échelle du gradient, c'est-à-dire qu'on ne va pas utiliser le même pas pour tous les paramètres. On ralentit le pas pour un paramètre qui a beaucoup changé précédemment, et on l'accélère pour un paramètre qui a peu changé. est-ce bien ça ? c'est le paramètre qui change ou la fonction qu'il fait changer ? !.

On utilise ici encore une estimation de $\mathbb{E}[g_t^2]$ obtenue par moyenne mobile à décroissance exponentielle.
On trouve une idée similaire dans les algorithmes AdaGrad (sans moyenne mobile) et RMSProp.

- $\widehat{m}_t \longleftarrow m_t/(1-\beta_1^t)$ et $\widehat{v}_t \longleftarrow v_t/(1-\beta_2^t)$
  Permet de corriger le biais vers 0 venant de l'initialisation de m_0 et v_0 à 0.
  C'est une innovation fournie par cet algorithme.

En conclusion, on peut voir que cet algorithme combine les idées fournies par les précédents algorithmes de descente de gradient, tels que la personnalisation des pas à chaque paramètre et la mémorisation du passé par le biais des moyennes mobiles, tout en apportant un vrai élément : la correction des biais des moments d'ordre 1 et 2.

3 Efficacité et points forts de la méthode

4 Amélioration des méthodes AdaGrad et RMSProp

+ visu
+ biblio ? ! !

Sources

[1] Diederik P Kingma and Jimmy Ba. Adam : A method for stochastic optimization. *ICLR 2015*, 2014.

[2] Josh Starmer. Optimization for deep learning (momentum, rmsprop, adagrad, adam). https://www.youtube.com/watch?v=NE88eqLngkg, 2023.

5 Annexes

Algorithme 2 : SGD

Entrées : $f(\theta)$, α, θ_0
$t \longleftarrow 0$
tant que θ_t *ne converge pas* **faire**
  $t \longleftarrow t + 1$
  $g_t \longleftarrow \nabla_\theta f_t(\theta_{t-1})$
  $\theta_t \longleftarrow \theta_{t-1} - \alpha \cdot g_t(\theta_{t-1})$
fin
Sorties : θ_t

Algorithme 3 : SGD avec moment (1964)

Entrées : $f(\theta)$, α, ρ, θ_0
$t \longleftarrow 0$
tant que θ_t *ne converge pas* **faire**
  $t \longleftarrow t + 1$
  $g_t \longleftarrow \nabla_\theta f_t(\theta_{t-1})$
  $m_t \longleftarrow \rho \cdot m_{t-1} - \alpha \cdot g_t$
  $\theta_t \longleftarrow \theta_{t-1} + m_t$
fin
Sorties : θ_t

Algorithme 4 : AdaGrad (2011)

Entrées : $f(\theta)$, α, ε, θ_0
$v_0 \longleftarrow 0$
$t \longleftarrow 0$
tant que θ_t *ne converge pas* **faire**
  $t \longleftarrow t + 1$
  $g_t \longleftarrow \nabla_\theta f_t(\theta_{t-1})$
  $v_t \longleftarrow v_{t-1} + g_t^2$
  $\theta_t \longleftarrow \theta_{t-1} - \alpha \cdot g_t / (\sqrt{v_t} + \varepsilon)$
fin
Sorties : θ_t

Algorithme 5 : RMSProp (2012)

Entrées : $f(\theta)$, α, β_2, ε, θ_0

$v_0 \longleftarrow 0$

$t \longleftarrow 0$

tant que θ_t *ne converge pas* **faire**

$t \longleftarrow t + 1$

$g_t \longleftarrow \nabla_\theta f_t(\theta_{t-1})$

$v_t \longleftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$

$\theta_t \longleftarrow \theta_{t-1} - \alpha \cdot g_t / (\sqrt{v_t} + \varepsilon)$

fin

Sorties : θ_t