Autoencodeurs

Julie Alleau, Juliette Lidoine et Chloé Serre-Combe

19 Octobre 2021

1 Introduction

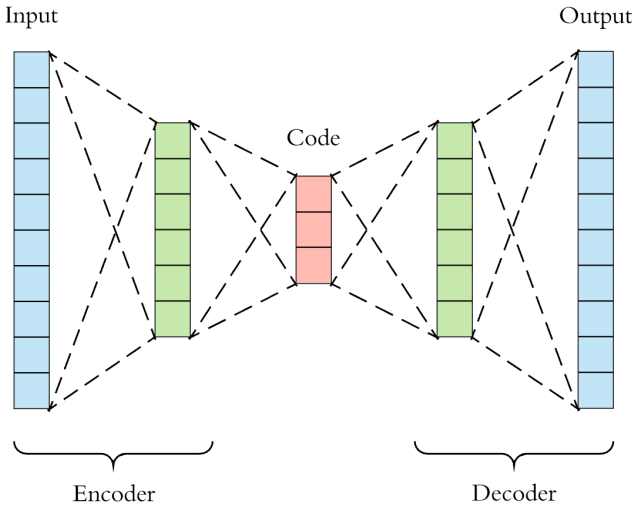
Le but de ce projet est de résumer l'article sur les autoencodeurs que vous pouvez retrouver ici. Dans une première partie nous allons vous présenter les autoencodeurs d'un point de vue général. Ensuite nous allons nous concentrer sur les autotencodeurs débruiteurs en vous présentant ses fonctionnalités. Nous allons également vous proposer un exemple pour illustrer ce type d'autoencodeur dont le code se trouve ici. Pour finir, on conclura en donnant les forces et les faiblesses des autoencodeurs.

2 Définition

Un autoencodeur est un réseau de neurones utilisé pour la réduction de dimension afin de déterminer des caractérisques ou propriétés des données. Il peut être vu comme un cas particulier des réseaux de neurones par propagation avant, et il est entrainé de manière non supervisée, c'est-à-dire qu'il apprend de manière automatique là où les données ne sont pas labellisées/classées.

La structure d'un autoencodeur se compose d'une couche cachée h correspondant au code, ayant une dimension moins importante que l'entrée, et de deux parties principales : une fonction encodeur $h = f(x)$ (qui prend des données en grande dimension et les compresse en plus petite dimension), une fonction décodeur g produisant la reconstruction $r = g(h)$, pour une entrée x (qui prend des données en petite dimension et les rétroprojette vers la plus grande dimension). Le nombre de neurones sera le même pour l'entrée et la sortie comme on peut le voir dans l'image ci-dessous. Ainsi, l'encodeur doit être surjectif et le décodeur injectif.

Figure 1: Structure d'un autoencodeur.



Certains réseaux de neurones se basent sur l'hypothèse selon laquelle les données que nous modélisons se trouvent sur ou autour d'une surface de plus petite dimension caractérisée par des goupes de plans tangents. Cette surface s'appelle une variété (ou manifold). Si cette hypothèse est vraie, connaître la variété permettrait de faciliter la compréhension des données. Un autoencodeur peut servir à déterminer une telle structure. Si l'autoencodeur apprend une fonction de reconstruction qui est invariante pour de petites perturbations alors il capturera la structure variété des données.

Le but est alors de copier l'entrée x vers la sortie avec une certaine subtilité étant donné qu'il ne copie jamais l'entrée parfaitement, il fait une approximation. En effet, si l'autoencodeur réussit à trouver la fonction identité $g(f(x)) = x$ pour chaque point de x ce serait inutile puisque l'approximation trouvée permet de garder seulement les propriétés et caractéristiques utiles des données d'entrée en supprimant, par exemple, du bruit. Pour arriver à ses fins, l'autoencodeur fait une réduction de la dimension au sein même de la couche cachée. Il y aura alors en quelque sorte une perte d'information sur l'entrée qui sera, par la suite, compensée par une bonne architecture du réseau. Il peut arriver que l'encodeur et le décodeur aient une trop grande capacité. Dans ce cas, l'autoencodeur ne pourra pas extraire d'information utile car il va réussir à trouver la fonction identité.

Le processus d'apprentissage de l'autoencodeur va minimiser la fonction de perte $L(x, g(f(x)))$, qui peut être pénalisée, afin d'avoir le meilleur résultat approximé possible. Il y aura alors un compromis à trouver entre l'approximation de l'entrée en sortie, qui ne doit pas être exacte, et cette minimisation de la perte ainsi que les contraintes ou pénalités de régularisation à ajouter.

Il existe plusieurs sortes d'autoencodeurs par exemple les autoencodeurs sous-completés, clairsemés ou bien encore les autoencodeurs contractifs. Néanmoins nous allons nous intéresser à la famille des autoencodeurs régularisés et plus précisement aux autoencodeurs débruiteurs.

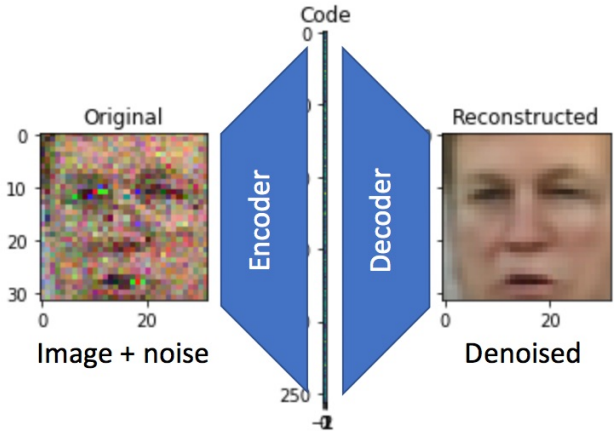
3 Autoencodeurs débruiteurs

Les autoencodeurs dont la dimension des données d'entrée est plus grande que la dimension de la couche cachée, nécessitent d'être régularisés afin de ne pas avoir des fonctions encodeur et décodeur linéaires. En effet, une fonction encodeur linéaire et une fonction décodeur linéaire également recopieraient bien l'entrée vers la sortie mais nous sans apporter d'informations sur nos données.

Les autoencodeurs débruiteurs font partie des autoencodeurs régularisés. Ces autoencodeurs régularisés utilisent une fonction de perte encourageant le modèle à avoir d'autres propriétés en plus de sa capacité à recopier l'entrée vers la sortie. Ces propriétés peuvent être la représentation parcimonieuse, mais encore la robustesse au bruit et aux entrées manquantes.

Les autoencodeurs débruiteurs ont pour objectif comme leur nom l'indique d'enlever le bruit ou la corruption des données d'entrées. Par exemple, pour une image d'entrée $\tilde{x}$ bruitée on pourra récupérer après reconstruction, l'image x non bruitée c'est-à-dire une approximation de l'image originale. Le but est que le bruit aléatoire disparaisse lors de la reconstruction.

Figure 2: Autoencodeur débruiteur.



L'autoencodeur débruiteur minimise une fonction de perte légèrement différente de celle des autres autoencodeurs, notée $L(x, g(f(\tilde{x})))$ où $\tilde{x}$ est une copie bruitée de x. Ceci revient à minimiser la même fonction de perte que pour les autres autoencodeurs mais, ici, la perte est calculée entre les données originales x et les données reconstruites $g(f(\tilde{x}))$ en sortie.

L'autoencodeur débruiteur apprend une distribution de reconstruction $p_{reconstructeur}(x|\tilde{x})$ estimée à partir des données d'entraînement $(x, \tilde{x})$ où x est un échantillon "entraînement" des données originales, et $\tilde{x}$ un échantillon "entraînement" des données bruitées.

4 Application des autoencodeurs débruiteurs

L'autoencodeur débruiteur peut être appliqué à des images ainsi qu'à des signaux. Nous avons appliqué un autoencodeur sur une partie des données `MNIST` à l'aide de la librairie `scikit_learn`.

L'autoencodeur est structuré de la manière suivante :

- Un encodeur constitué de deux couches : une de 500 neurones et une de 300.
- Un décodeur constitué de deux couches : une de 300 neurones et une de 500.
- Une couche cachée constituée de 50 neurones pour débruiter ou de 4 neurones pour déterminer seulement la classe. En effet, si on diminue trop le nombre de neurones de la couche cachée, l'autoencodeur ne sera plus capable de générer correctement l'image débruitée mais il donnera une image possédant les mêmes caractéristiques, c'est à dire de la même classe.

Nous entraînons l'autoencodeur sur un échantillon d'apprentissage contenant les images bruitées en entrée et les images originales, non bruitées, en sortie. Et nous obtenons les résultats fournis en annexes page 4.

Sur la Figure 3, nous pouvons voir que l'image avec ajout d'un bruit gaussien raisonnable est bien débruitée par l'autoencodeur avec 50 neurones en couche cachée et la classe de l'image d'entrée est bien reconnu comme un 5 par l'autoencodeur à 4 neurones en couche cachée.

Sur la Figure 4, nous pouvons voir que lorsque que le bruit est trop important, l'autoencodeur à 50 neurones, en couche cachée, n'arrive plus à reconstruire correctement l'image alors que celui à 4 neurones, en couche cachée, va reconnaître les caractéristiques du chiffre 6. Ce dernier semble donc plus intéréssant sur des données très bruitées. Néanmoins, après plusieurs tests effectués, on remarque que cet autoencodeur ne trouve pas à tous les coups la bonne classe pour l'image comme on peut le voir sur la Figure 5.

Sur la Figure 6, nous sommes dans le cas où l'image en entrée n'a pas été corrompue. Nous pouvons voir que l'autoencodeur va quand même débruiter cette image. Celle-ci va donc être modifiée et l'image de sortie sera différente de l'image originale. Pour l'autoencodeur à 50 neurones en couche cachée, la forme du chiffre 5 restera visible et l'autoencodeur à 4 neurones va ici avoir du mal à reconnaître les caractéristiques de ce chiffre 5 qui est particulier, il va hésiter entre un 1 et un 5.

5 Conclusion : Avantages et limites

Les autoencodeurs peuvent être plus intéressants que d'autres réseaux de neurones, dans le sens où ils peuvent être entraînés sur des données non-labellisées, c'est à dire qu'ils utilisent un processus d'apprentissage non supervisé. De plus, en réduisant la dimension des données, les autoencodeurs permettent d'écarter le bruit. Les techniques d'autoencodeurs peuvent effectuer des transformations non linéaires avec leur fonction d'activation non linéaire et leurs couches multiples, contrairement à l'analyse sur composantes principales. Lorsque les données sont de nature complexe et non linéaires, les autoencodeurs sont alors très utiles.

Les autoencodeurs ont aussi des limites. En effet, ils peuvent détruire l'interprétabilité des données ou du modèle et si les données d'entraînement ne sont pas représentatives des données test, alors l'information peut être assombrie voire même perdue. De plus, pour générer des résultats pertinants, ils ont besoin d'un nombre très important de données. Enfin, le choix du nombre de neurones dans la couche cachée est déterminant. En effet, si celui-ci est trop faible il risque d'y avoir une perte d'information trop importante entre l'encodeur et le décodeur, et si il est trop important (sur-complété), le modèle n'apprendra plus les caractéristiques des données et se contentera de retourner l'entrée sans modification (cf. Figure 7).

Annexes

Figure 3: Sortie des autoencodeurs débruiteurs avec ajout d'un bruit gaussien raisonnable en entrée.

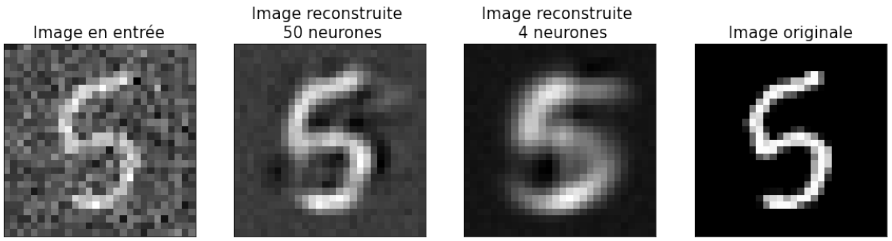


Figure 4: Sortie des autoencodeurs débruiteurs avec ajout d'un important bruit gaussien en entrée.

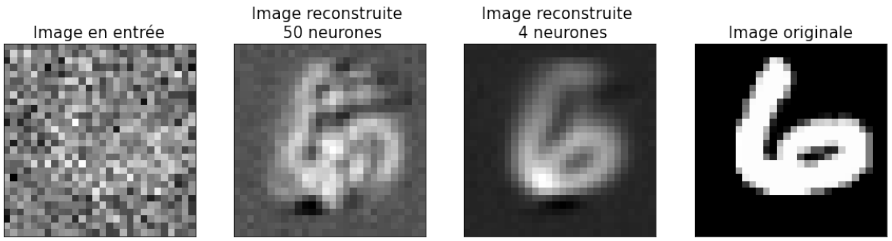


Figure 5: Sortie de l'autoencodeur débruiteur avec 4 neurones en couche cachée avec ajout d'un important bruit gaussien en entrée.

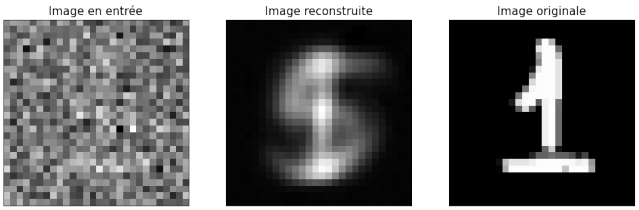


Figure 6: Sortie des autoencodeurs débruiteurs avec une image non bruitée en entrée.

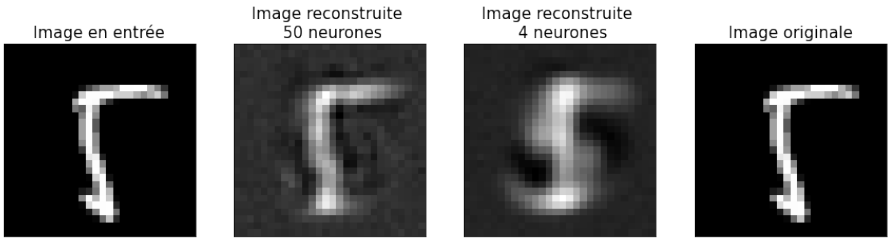
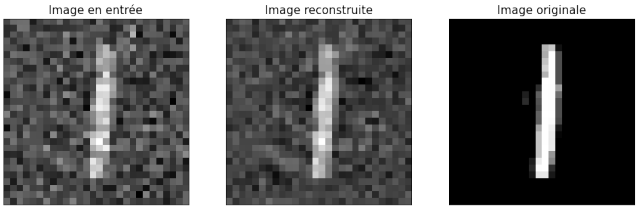


Figure 7: Sortie d'un autoencodeurs débruiteur avec un nombre trop important de neurones en couche cachée.



Lien utile :

Lien github : https://github.com/chloesrcb/Autoencoders

Bibliographie

[1] Article, Chapitre 14 sur les autoencodeurs.
[https://www.deeplearningbook.org/contents/autoencoders.html].

[2] Alfredo Canziani. Introduction aux auto-encodeurs, 2020.
[https://atcold.github.io/pytorch-Deep-Learning/fr/week07/07-3/].

[3] Seungchul Lee. Introduction aux auto-encodeurs.
[https://i-systems.github.io/teaching/ML/iNotes/15_Autoencoder.html].

[4] Source de l'image de la Figure 1.
[https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798].

[5] Source de l'image de la Figure 2.
[https://zhangruochi.com/Autoencoders-Project/2019/07/11/denoising.jpg]