Les mathématiques des réseaux de neurones
Partie 2 : les réseaux génératifs

Gabriel Peyré
CNRS & DMA
École Normale Supérieure
gabriel.peyre@ens.fr

Résumé

Dans l'article précédent, nous avons vu comment l'on peut entrainer de façon supervisée des réseaux de neurones. Ceci permet de résoudre efficacement des problèmes de classification, par exemple de reconnaissance d'images. Ce qui est peut être encore plus surprenant, c'est que ces réseaux de neurones sont maintenant également utilisés de façon non-supervisée afin de générer automatiquement des textes ou des images « virtuels », ce que l'on appelle souvent des « deep fakes ». Dans ce second article, je tisserai un lien entre l'apprentissage de réseaux de neurones génératifs et la théorie du transport optimal. Ce problème a été posé par Gaspard Monge au 18e siècle, puis il a été reformulé par Leonid Kantorovitch au milieu du 20e siècle. Il est maintenant devenu un outil de choix pour aborder l'explosion récente de la science des données.

1 Réseaux de neurones génératifs

Au lieu d'utiliser des réseaux de neurones pour analyser des images, des recherches récentes [3] ont montré qu'on pouvait les utiliser « à l'envers » afin de générer des images. Ces réseaux de neurones génératifs trouvent par exemple des applications pour faire des effets spéciaux et pour les jeux vidéos. Ils questionnent également les processus créatifs et artistiques. On retrouve des questions similaires dans l'apprentissage des voitures autonomes et la résolution de jeux de stratégie. La figure 1 montre la structure d'un tel réseau g. Les couches jouent en quelque sorte des rôles miroirs par rapport à l'architecture des réseaux de neurones discriminatifs exposés dans l'article précédent. A partir d'une entrée y composée d'un petit nombre de valeurs, qui sont typiquement tirées aléatoirement, on génère une image $x = g(y)$.

Le problème d'apprentissage de tels réseaux est non-supervisé : on dispose uniquement d'un grand nombre d'images d'apprentissage (que l'on peut par exemple prendre sur internet) et on n'a pas besoin de savoir ce que contiennent ces images. Le but est alors de sélectionner les poids w des neurones du réseau g de sorte que les images aléatoires générées (les images fausses, « fakes » en anglais) ressemblent le plus possible aux images d'apprentissage. La section 3 détaille la formulation mathématique de ce problème. La collecte de données est plus simple que pour l'entrainement de réseaux discriminatifs. Il n'y a plus besoin d'interversion humaine pour indiquer au réseau le contenu des images qu'il doit reconnaitre. De plus, ce principe d'apprentissage non supervisé est proche de la façon dont les enfants apprennent, principalement en observant et manipulant le monde qui les entoure.

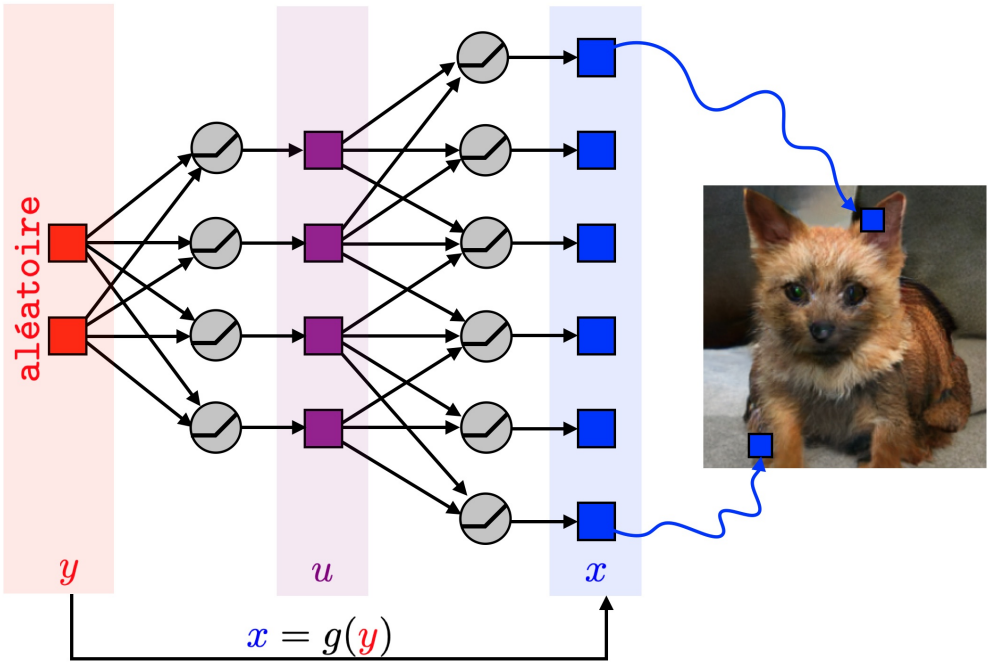

FIGURE 1 – Exemple d'un réseau de neurones génératif simplifié (un réseau permettant de générer des images aussi complexes possède plus de couches).

2 L'apprentissage non supervisé de réseaux génératifs

Le but des réseaux de neurones génératifs n'est pas de résoudre un tâche telles que la reconnaissance d'objets dans les images. Afin d'entrainer les poids w du réseau, il faut formaliser mathématiquement le problème. Il s'agit de générer un ensemble d'images « virtuelles » (les fausses) qui ressemblent aux images réelles d'une base de données. Il ne s'agit pas simplement qu'une image générée ressemble à une image réelle, il faut mettre en correspondance deux ensembles d'images. Par exemple, si dans la base de donnée il y a la moitié de chiens et la moitié de chats, il faut que le réseaux génère également pour moitié des chiens et pour moitié des chats. La question des biais, qui est essentielle dans l'intelligence artificielle, apparait donc clairement.

On va noter $\{x_1, x_2, \ldots, x_n\}$ l'ensemble des n images de la base de données. Le nombre n d'images est très grand, de l'ordre de plusieurs milliers ou millions. Etant donné un réseau de neurone génératif g, qui est paramètré par ses poids w, on note $\{z_1, z_2, \ldots, z_n\}$ un ensemble de n images « fausses » générées aléatoirement par le réseau. Pour générer la première image fausse z_1, ceci signifie que l'on a tiré aléatoirement les valeurs d'entrée y_1 et qu'on a appliqué le réseau avec ces entrées, de sorte à obtenir l'image virtuelle $z_1 = g(y_1)$. On a ensuite fait la même chose avec $z_2 = g(y_2)$ et ainsi de suite.

Le but de l'apprentissage non supervisé est donc de trouver des poids w de sorte que l'ensemble des fausses images $\{z_1, \ldots, z_n\}$ soit les plus proches de l'ensemble des images $\{x_1, \ldots, x_n\}$ de la base de donnée. Le problème d'optimisation s'écrit ainsi

$$\min_w \text{distance}(\{x_1, \ldots, x_n\}, \{z_1, \ldots, z_n\}).$$

Il faut ici se rappeler que les images générées $\{z_1, \ldots, z_n\}$ dépendent du réseau g et donc des poids w. La question mathématique qui se pose est donc de définir une notion de distance entre deux ensembles de points. Il existe de nombreuses façons de le faire, et nous allons en expliquer une qui est bien adaptée à ce problème d'apprentissage. Elle exploite la théorie du transport optimal.

3 Le transport optimal de Monge

Le problème de transport optimal a été formulé par Gaspard Monge [6] en 1781, pour des applications militaires. La question posée est de déterminer la façon la plus économe de transférer des objets depuis un ensemble de sources $\{x_1, \ldots, x_n\}$ vers un ensemble de destinations $\{z_1, \ldots, z_n\}$. Pour Monge, il s'agit de transférer de la terre depuis des remblais pour créer des remblais. Mais c'est une question universelle, qui trouve une multitude d'applications. Pour le problème d'entrainement de réseaux génératifs, les points sources et les destinations sont les images de la base de données et les images fausses générées par le réseau.

Il faut ainsi relier chaque source, par exemple x_1 vers un unique point de destination, que l'on va noter $z_{\sigma(1)}$, où $\sigma(1)$ est un entier entre 1 et n. De manière similaire, x_2 est relié à $z_{\sigma(2)}$ et ainsi de suite. Il faut également que chacune des n destinations soit approvisionnée par une source. Ceci signifie par exemple que x_1 et x_2 ne peuvent pas être reliés à la même destination, il faut relier toutes les sources à des destinations différentes. Ceci signifie que $\{\sigma(1), \ldots, \sigma(n)\}$ doit être une permutation des n premiers nombres entiers. Par exemple, sur la figure 2, sur un exemple simple avec $n = 6$ éléments, on a choisit sur la gauche la permutation

$$(\sigma(1) = 1, \sigma(2) = 5, \sigma(3) = 4, \sigma(4) = 6, \sigma(5) = 3, \sigma(6) = 2).$$

Le problème de Monge consiste alors à trouver la permutation σ qui minimise la somme des coûts de transport. Monge a décidé que le coût de transport entre une source x et une destination z est égal à la distance Euclidienne $\|x - z\|$ entre les deux points, mais on peut choisir d'autre coût. Par exemple un temps de trajet ou bien le prix nécessaire en essence si on utilise des camions, etc. Si on conserve pour simplifier l'idée que le coût est égal à la distance, on doit ainsi résoudre le problème

$$\min_{\text{permutation } \sigma} \|x_1 - z_{\sigma(1)}\| + \|x_2 - z_{\sigma(2)}\| + \ldots + \|x_2 - z_{\sigma(n)}\|.$$

La difficulté est que le nombre total de permutations a tester est très grand. En effet, pour le choix de $\sigma(1)$ il y a n possibilités, pour celui de $\sigma(2)$ il en y a $n - 1$ (puisque la valeur de $\sigma(1)$ est prise), pour $\sigma(2)$ il y en a $n - 2$, etc. Donc le nombre total de permutations égal $n!$, la factorielle du nombre n, qui est définie comme

$$n! = n \times (n - 1) \times (n - 2) \times \ldots \times 2 \times 1.$$

Pour $n = 6$ comme à la figure 2, il y a donc

$$6! = 6 \times 5 \times 4 \times 3 \times 2 \times 1 = 720 \text{ possibilités.}$$

Dans ce cas simple, on peut donc toutes les tester et choisir la meilleure, qui est, comme montré à la droite de la figure 2,

$$(\sigma(1) = 4, \sigma(2) = 3, \sigma(3) = 5, \sigma(4) = 1, \sigma(5) = 6, \sigma(6) = 2).$$

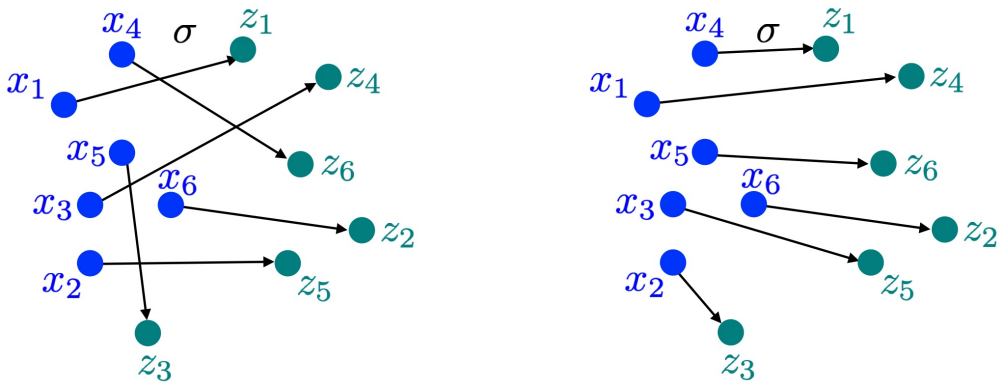

FIGURE 2 – Exemple à gauche d'une permutation σ non-optimale et à droite de la permutation optimale, dans le cas de 6 points en dimension 2.

Le problème, c'est que pour $n = 70$, il y a plus de 10^{100} possibilités (c'est-à-dire un 1 avec 100 zeros après), ce qui est à comparer aux 10^{79} atomes dans l'univers ... Et pour entrainer des réseaux de neurones, n est encore beaucoup plus grand. Il a donc fallu attendre plusieurs révolutions mathématiques et numériques pour pouvoir obtenir une méthode permettant de résoudre ce problème.

4 Le transport optimal de Kantorovitch

Monge avait déjà remarqué que les solutions de son problème avait des structures très particulières. Par exemple, on peut observer sur la figure 2, droite, que les trajets optimaux ne se croisent pas, et Monge l'avait prouvé dans son article. Mais ceci n'est pas suffisant pour résoudre le problème, car il existe encore énormément de trajectoires sans croisement. Il a fallu plus de 200 ans pour comprendre comment obtenir plus d'information sur les solutions afin de les calculer efficacement. C'est Leonid Kantorovitch qui a trouvé en 1942 [4] une nouvelle formulation du problème de transport optimal qui est calculable rapidement. Pour se faire, il a autorisé chaque source à se diviser en plusieurs parties, par exemple deux parties égale avec une pondération de 1/2 chacune. Cette division de la production est intéressante car elle simplifie le problème d'optimisation. Elle est également naturelle pour les préoccupations de Kantorovitch qui était de modéliser et planifier la production en économie. Il a d'ailleurs obtenu le prix Nobel d'économie pour cette idée. Conjointement à ces travaux pionniers de Kantorovitch, George Dantzig a trouvé en 1947 l'algorithme du simplexe [2], qui permet de résoudre efficacement des problème de transport de grande taille. Sa complexité numérique pour résoudre un problème de transport optimal entre n points est de l'ordre de $n^3 = n \times n \times n$, ce qui est beaucoup plus faible que $n!$. Il est au coeur d'un très grand nombre de systèmes industriels qui doivent optimiser l'adéquation entre des moyens de production et de consommation. Et on peut du coup également l'utiliser pour entrainer des réseaux de neurones génératifs !

5 Les réseaux antagonistes

Une difficulté pour appliquer le transport optimal pour entrainer des réseaux génératifs est qu'il faut choisir le coût de transport entre deux images. On pourrait calculer la distance Euclidienne

FIGURE 3 – Deux exemples de « deep fakes » qui sont des images virtuelles interpolant entre chats et chiens.

entre les pixels des images, mais ceci ne marche pas bien, car cela ne prend pas en compte la géométrie des objets présents dans les images. Une idée très fructueuse a été introduite en 2014 par Ian Goodfellow et ses collaborateurs [3]. Elle consiste à utiliser un second réseau de neurones pour déterminer ce coût de transport [5]. Ce second réseaux f est nommé réseaux adversaire et il joue un rôle de discriminateur. Alors que le but du générateur g est de générer des images fausses très ressemblantes, le but de f est au contraire de faire de son mieux pour reconnaitre les vraies et les fausses images. Ces deux réseaux sont entrainés conjointement, c'est pour cela que l'on parle de réseaux antagonistes. L'entrainement de ces deux réseaux corresponds à ce que l'on appelle un jeu à somme nulle, introduit par John Von Neumann [7] et généralisé ensuite par John Nash [8] qui a obtenu tout comme Kantorovitch le prix Nobel d'économie.

Ces avancées récentes [3] ont ainsi permis d'obtenir des résultats excellents pour la génération d'images. La figure 3 montre des résultats obtenus avec la méthode [1] et son utilisation pour calculer des « chemins » d'images entre chiens et chats.

Références

[1] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *Proc. ICLR 2019*, 2019.

[2] George B Dantzig. Origins of the simplex method. In *A history of scientific computing*, pages 141–151. 1990.

[3] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[4] Leonid Kantorovich. On the transfer of masses (in russian). *Doklady Akademii Nauk*, 37(2) :227–229, 1942.

[5] SC Martin Arjovsky and Leon Bottou. Wasserstein generative adversarial networks. In *Proceedings of the 34 th International Conference on Machine Learning, Sydney, Australia*, 2017.

[6] Gaspard Monge. Mémoire sur la théorie des déblais et des remblais. *Histoire de l'Académie Royale des Sciences*, pages 666–704, 1781.

[7] Oskar Morgenstern and John Von Neumann. *Theory of games and economic behavior*. Princeton university press, 1953.

[8] John F Nash. Equilibrium points in n-person games. *Proceedings of the national academy of sciences*, 36(1) :48–49, 1950.