Réinterprétation des modèles génératifs de diffusion

Denis DUVAL[1,2] Agnès DESOLNEUX[2]

[1]GE HealthCare, 283 Rue de la Minière, 78530 Buc, France

[2]Centre Borelli, CNRS and ENS paris-Saclay, 4 avenue des sciences, 91190 Gif-sur-Yvette, France

Résumé – Les méthodes de l'état de l'art pour la génération d'images se basent pratiquement toutes sur l'apprentissage profond, c'est le cas en particulier des modèles de diffusion. Ces modèles sont capables de générer des images de haute qualité, fidèles à la loi des données d'entrainement. Cependant, ces modèles sont des boîtes noires du point de vue de l'humain : il est difficile d'expliquer quelles décisions le modèle prend et pourquoi. Dans ce travail, nous proposons une analyse théorique de la créativité des modèles de diffusion DDPM en les réinterprétant comme une version modifiée du célèbre algorithme de synthèse de texture de Kwatra et al. Nous montrons que sous certaines hypothèses, inverser un processus de diffusion est aussi simple que de faire des "copier-coller" de patchs des images d'entrainement.

Abstract – State-of-the-art methods for image generation are almost all based on deep learning, as for example diffusion models. These models are capable of generating high-quality images that are faithful to the law of the training data. However, these models act as a black box from a human perspective : it is difficult to explain what decisions the model makes and why. In this work, we propose a theoretical analysis of the creativity of DDPM diffusion models by reinterpreting them as a modified version of the famous texture synthesis algorithm by Kwatra et al. We show that under certain assumptions, reversing a diffusion process is as simple as "copy-pasting" patches from the training images.

1 Introduction

1.1 Modèles génératifs et explicabilité

Les modèles génératifs pour l'image sont devenus très performants. Grâce à l'apprentissage profond, les modèles de l'état de l'art peuvent apprendre des distributions de probabilités complexes, en grande dimension, et en générer de nouveaux échantillons. Cependant il y a deux aspects dont les modèles d'apprentissage profond souffrent et que nous considérons dans ce travail :

- leur **explicabilité** : il est souvent difficile de comprendre ce qu'un modèle d'apprentissage profond apprend des données d'entrainement et son inférence sur des données nouvelles.
- la **dépendance au nombre de données** : il est connu que les modèles d'apprentissage profond ont une performance qui croît avec le nombre de données, et que dans beaucoup de situations pratiques le modèle souffre d'un nombre de données très réduit, lui rendant la tâche de généralisation difficile.

Dans ce travail, on se propose de donner une réponse à l'explicabilité des modèles génératifs, en particulier les modèles de diffusion, dans une configuration où le nombre de données d'entraînement est très faible par rapport à leur dimension. Plus précisément nous nous basons sur l'analyse théorique de créativité faite dans [2], qui calcule l'approximation optimale du score de la probabilité, lorsque le réseau possède certains biais inductifs, qui sont exactement ceux à l'origine des opérations de convolution dans les réseaux de neurones [4].

En partant de la procédure d'échantillonnage qui résulte d'un tel score, nous proposons de montrer, que sous ces mêmes hypothèses, un modèle de diffusion est en réalité très similaire à une version modifiée, éventuellement non stationnaire, de l'algorithme de synthèse de texture de Kwatra [3]. Il est alors intéressant de comprendre qu'un modèle de diffusion est dans ce cas réduit à une méthode n'impliquant ni apprentissage profond ni versions bruitées des données d'entraînement.

1.2 Notations

- d représente le nombre de pixels d'une image. Une image est ici un élément x de $\mathbb{R}^d$.
- $[d]$ est l'ensemble $\{1, ..., d\}$.
- $(\Omega_i)_{i\in[d]}$ est une famille de voisinages de pixels i.e. vérifiant $\forall i,\ \Omega_i \subset [d]$ et $i \in \Omega_i$.
- Pour $x \in \mathbb{R}^d$ et $i \in [d]$, x^i et x^{Ω_i} désignent respectivement la i-ème coordonnée de x et la sous-variable de x sur l'ensemble de coordonnées Ω_i (patch de x restreint à Ω_i).
- $\mathcal{D}$ est un sous-ensemble fini de $\mathbb{R}^d$, contenant les données d'entraînement.
- $\mathcal{N}(\mu, \Sigma)$ représente une loi Gaussienne multivariée de moyenne μ et de matrice de covariance Σ tandis que $\varphi(x; \mu, \Sigma)$ est sa densité de Lebesgue en x.
- I désigne la matrice identité, dont la dimension dépend du contexte.
- $\mathbb{E}_{x\sim p}[\cdot]$ désigne l'espérance sur la variable x sous la loi de probabilité de densité de Lebesgue p.
- $\nabla \log p$ est appelé le score de la distribution de probabilité dont la densité de Lebesgue est p.

2 Modèles de diffusion et algorithme de Kwatra

2.1 Modèles de diffusion

Dans ce travail, nous considérons les modèles DDPM (Denoising Diffusion Probabilistic Models) introduits par [1]. Plus précisément, à partir d'une distribution initiale $q(x_0)$ on considère la chaîne de Markov $(x_t)_{t\in\{0,\dots,T\}}$ définie par le noyau de transition suivant :

$$q(x_t|x_{t-1}) = \varphi(x_t; \sqrt{\alpha_t}x_{t-1}, (1-\alpha_t)I).$$

Il est connu que l'on peut en déduire le noyau de transition de x_0 à x_t :

$$q(x_t|x_0) = \varphi(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I) \tag{1}$$

où $\bar{\alpha}_t = \prod_{s=1}^{T} \alpha_s$. Les paramètres $(\alpha_t)_{t\in[T]}$ sont choisis de telle sorte que $\bar{\alpha}_T \simeq 0$ et donc $q(x_T)$ peut être approchée par une loi $\mathcal{N}(0, I)$ quelle que soit la distribution initiale.

L'idée générale des modèles de diffusion est d'approcher les transitions inverses $q(x_{t-1}|x_t)$ par des noyaux gaussiens :

$$p(x_{t-1}|x_t) = \varphi(x_{t-1}|\mu(x_t, t), \sigma^2(t)I_d) \tag{2}$$

où $\mu(\cdot,\cdot)$ est en pratique un réseau de neurones, afin de simuler le processus de diffusion en temps inverse de $t = T$ à $t = 0$. Avec la paramétrisation suivante :

$$\mu(x_t, t) = \frac{x_t + (1-\alpha_t)s(x_t, t)}{\sqrt{\alpha_t}}, \tag{3}$$

la minimisation de la fonction de perte dans [1] est équivalente à la minimisation d'une somme pondérée sur t du score matching à l'instant t :

$$\min_{s(\cdot,t)\in\mathcal{S}} \mathcal{L}_t(s(\cdot,t)) = \mathbb{E}_{x_t\sim q_t}[\|s(x_t,t) - \nabla_{x_t}\log q_t(x_t)\|_2^2] \tag{4}$$

où $\mathcal{S}$ est un sous-ensemble des fonctions de $\mathbb{R}^d$ dans lui-même, qui contient les fonctions avec des biais inductifs choisis. En pratique, x_t est échantillonné selon la distribution empirique :

$$\hat{q}_t(x_t) = \frac{1}{|\mathcal{D}|}\sum_{x\in\mathcal{D}} \varphi(x_t; \sqrt{\bar{\alpha}_t}x, (1-\bar{\alpha}_t)I).$$

Cette distribution empirique est ensuite utilisée dans les calculs de [2] pour garder la dépendance du score aux données d'entraînement. On peut calculer le score de la distribution empirique :

$$\nabla_{x_t}\log\hat{q}_t(x_t) = -\frac{1}{1-\bar{\alpha}_t}\sum_{x\in\mathcal{D}}(x_t - \sqrt{\bar{\alpha}_t}x)w_t(x|x_t) \tag{5}$$

avec

$$w_t(x|x_t) = \frac{\varphi(x_t; \sqrt{\bar{\alpha}_t}x, (1-\bar{\alpha}_t)I)}{\sum_{x'\in\mathcal{D}}\varphi(x_t; \sqrt{\bar{\alpha}_t}x', (1-\bar{\alpha}_t)I)}. \tag{6}$$

Le score de (5) est le score optimal en pratique si $s(x_t, t)$ n'est pas contraint *i.e.* si $\mathcal{S}$ est égal à l'ensemble des fonctions de $\mathbb{R}^d$ dans lui-même.

2.2 Score optimal sous contrainte de localité

Pour la génération d'images, les réseaux utilisés pour approcher le score dans (4) sont principalement composés d'opérations de convolution, dont les filtres sont de tailles très réduites par rapport à la taille de l'image. Une telle opération de convolution possède en particulier un biais de localité : la sortie d'une opération de convolution en un pixel ne dépend que de l'entrée réduite à un voisinage autour de ce pixel (le voisinage induit par le filtre de convolution).

On considère alors $\mathcal{S}_{\text{local}}$ le sous-ensemble des fonctions s vérifiant :

$$\forall i \in [d], \exists s'_i \text{ fonction t.q.} \forall x \in \mathbb{R}^d, s(x)^i = s'_i(x^{\Omega_i}). \tag{7}$$

Les auteurs de [2] résolvent (4) sur $\mathcal{S}_{\text{local}}$:

$$\forall i \in [d], \quad s_t^*(x_t)^i = -\frac{1}{1-\bar{\alpha}_t}\sum_{x\in\mathcal{D}}(x_t^i - \sqrt{\bar{\alpha}_t}x^i)w_t^i(x^{\Omega_i}|x_t^{\Omega_i}) \tag{8}$$

avec :

$$w_t^i(x^{\Omega_i}|x_t^{\Omega_i}) = \frac{\varphi(x_t^{\Omega_i}; \sqrt{\bar{\alpha}_t}x^{\Omega_i}, (1-\bar{\alpha}_t)I)}{\sum_{x'\in\mathcal{D}}\varphi(x_t^{\Omega_i}; \sqrt{\bar{\alpha}_t}x'^{\Omega_i}, (1-\bar{\alpha}_t)I)}. \tag{9}$$

Comparé au score de la distribution empirique (5) i.e. le score optimal sans contrainte, le poids a posteriori w_t^i dépend désormais de la distance en norme euclidienne uniquement réduite à un voisinage autour du pixel en i.

2.3 Diffusion inverse résultante

On s'intéresse desormais à écrire les étapes résultantes de la diffusion inverse avec l'approximation du score trouvée en (8). En injectant (8) dans (3) puis (3) dans (2), on réécrit :

$$x_{t-1} \sim p(\cdot|x_t) \iff$$

$$\forall i \in [d], y_{t-1}^i = (1-\lambda_t)y_t^i + \lambda_t\sum_{x\in\mathcal{D}} x^i w_t'^i(x^{\Omega_i}|y_t^{\Omega_i}) + \sigma'(t)\epsilon^i \tag{10}$$

avec $y_t = \frac{x_t}{\sqrt{\bar{\alpha}_t}}$, $\lambda_t = \frac{1-\alpha_t}{1-\bar{\alpha}_t}$, $\sigma'(t) = \frac{\sigma(t)}{\sqrt{\bar{\alpha}_{t-1}}}$, $\epsilon^i \sim \mathcal{N}(0,1)$ et

$$w_t'^i(x^{\Omega_i}|y_t^{\Omega_i}) = \frac{\varphi(y_t^{\Omega_i}; x^{\Omega_i}, \frac{(1-\bar{\alpha}_t)}{\bar{\alpha}_t}I)}{\sum_{x'\in\mathcal{D}}\varphi(y_t^{\Omega_i}; x'^{\Omega_i}, \frac{(1-\bar{\alpha}_t)}{\bar{\alpha}_t}I)}.$$

Nous résumons la diffusion inverse résultante dans l'algorithme 1 ci-dessous.

Algorithme 1 : Diffusion inverse avec score localement contraint

$y_T \sim \mathcal{N}(0, \frac{1}{\bar{\alpha}_t}I)$
pour $t = T, \dots, 1$ **faire**
  pour $i \in [d]$ **faire**
    $z_t^i = (\text{soft argmin}_{\tau_t, z\in\mathcal{Z}_i}||y_t^{\Omega_i} - z||_2^2)^i$
    $y_{t-1}^i = \text{conv}_{\lambda_t}(y_t^i, z^i) + \sigma'(t)\epsilon_i$
  fin
fin
retourner y_0

Dans l'algorithme 1, on note :

- $\text{conv}_{\lambda_t}(a,b) = (1-\lambda_t)a + \lambda_t b$
- $\text{soft argmin}_{\tau_t, z\in\mathcal{Z}} ||y-z||_2^2 = \sum_{z\in\mathcal{Z}} z \frac{\exp(-\frac{||y-z||_2^2}{2\tau_t})}{\sum_{z'\in\mathcal{Z}} \exp(-\frac{\|y-z'\|_2^2}{2\tau_t})}$

avec $\tau_t = \frac{1-\bar{\alpha}_t}{\bar{\alpha}_t}$.

2.4 L'algorithme de Kwatra

Dans la limite de niveaux de bruit faibles dans la diffusion (autrement dit pour les petits t), conv_{λ_t} retourne son deuxième argument et soft $\text{argmin}_{\tau_t, z\in\mathcal{Z}}$ devient l'opération $\text{argmin}_{\tau_t, z\in\mathcal{Z}}$. L'étape itérative dans l'algorithme 1 revient dans cette limite à remplacer y_t^i par le pixel en i de la donnée d'entraînement x dont le patch x^{Ω_i} est le plus proche de $y_t^{\Omega_i}$ parmi les données d'entraînement. Cette opération de mise à jour est très similaire à l'opération de mise à jour de l'algorithme de synthèse de texture de Kwatra [2]. En effet cet algorithme cherche à minimiser l'énergie suivante :

$$E(x,z) = \sum_{i\in[d]} \|x^{\Omega_i} - z(i)\|_2^2$$

au moyen d'un algorithme itératif alternant une étape de minimisation d'énergie et une étape de calcul des patchs les plus proches, voir l'algorithme 2. Ici, la famille $(z(i))_{i\in[d]}$ est une famille de patches à valeurs dans $\mathcal{Z}$, un dictionnaire fixe qui est l'ensemble des patches d'une image exemple. On note $z(i)^0$ la coordonnée centrale du patch.

Algorithme 2 : Algorithme de Kwatra original

$z_0(i) =$ voisinage aléatoire dans $\mathcal{Z} \quad \forall i \in [d]$
pour $n = 0$ *à* N **faire**
  pour $i \in [d]$ **faire**
    $x_{n+1}^i = (\text{argmin}_x \sum_{i\in[d]} ||x^{\Omega_i} - z_n(i)||_2^2)^i$
    $z_{n+1}(i) = \text{argmin}_{z\in\mathcal{Z}} ||x_{n+1}^{\Omega_i} - z||_2^2$
  fin
  si $z_{n+1}(i) = z_n(i)\ \forall i \in [d]$ **alors**
    $x = x_{n+1}$
    arrêt
  fin
fin
retourner x sinon x_{N+1}

La principale différence avec l'algorithme 1 est que la mise à jour dans la diffusion n'affecte que le pixel central du patch alors que dans la version originale les pixels sont moyennés parmi tous les patchs auxquels ils appartiennent dans la famille $(\Omega_i)_{i\in[d]}$. Une autre différence est que le dictionnaire de patches $\mathcal{Z}$ ne dépend pas de la position du pixel alors qu'il peut a priori en dépendre dans la diffusion.

On se propose alors de modifier l'algorithme de Kwatra, afin de répliquer les opérations réalisées lors de la diffusion inverse, en supprimant tout ajout de bruit gaussien au cours de la procédure, résultant en l'algorithme 3. Désormais, le dictionnaire de patches dépend de la position i : on note $\mathcal{Z}_i = \{x^{\Omega_i}, x \in \mathcal{D}\}$ l'ensemble de tous les patchs d'entraînement à une position i.

Algorithme 3 : Algorithme de Kwatra modifié

$z_0(i) =$ voisinage aléatoire dans $\mathcal{Z}_i \quad \forall i \in [d]$
pour $n = 0$ *à* N **faire**
  pour $i \in [d]$ **faire**
    $x_{n+1}^i = z_n(i)^0$
    $z_{n+1}(i) = \text{argmin}_{z\in\mathcal{Z}_i} ||x_{n+1}^{\Omega_i} - z||_2^2$
  fin
  si $z_{n+1}^i = z_n^i\ \forall i \in [d]$ **alors**
    $x = x_{n+1}$
    arrêt
  fin
fin
retourner x sinon x_{N+1}

2.5 Lien entre les algorithmes 1 et 3

Il est faux que les algorithmes 1 et 3 sont égaux. Néanmoins, la diffusion inverse simulée avec l'EDO et l'algorithme de Kwatra modifié renvoient les mêmes types d'images, au sens de la définition suivante.

Définition 1. *Un point x_0 est localement cohérent si chaque pixel x_0^i est égal au pixel central du patch x^{Ω_i} le plus proche de $x_0^{\Omega_i}$ en norme euclidienne, parmi les patchs d'entraînements $\mathcal{Z}_i = \{x^{\Omega_i}, x \in \mathcal{D}\}$.*

Pour la diffusion avec le score local optimal, [2] montre le résultat suivant :

Théorème 1 (Voir Theorem 4.1. dans [2])**.** *La diffusion inverse avec le score en* (8)*, simulée avec l'EDO correspondante, ne peut se terminer que sur des points localement cohérents.*

Quant à l'algorithme de Kwatra modifié, la condition d'arrêt de l'algorithme garantit que si l'algorithme termine alors l'image renvoyée est nécessairement localement cohérente. De plus, si le premier point calculé x_1 par l'aglorithme est localement cohérent alors l'algorithme termine et renvoie x_1. Pour n'importe point localement cohérent, la probabilité qu'il soit le premier point calculé par l'algorithme est strictement positive, puisque les voisinages sont en nombre fini et l'échantillonnage de départ parmi ces voisinages est uniforme. On peut alors affirmer :

Proposition 1. *L'ensemble des points de convergence de l'algorithme de Kwatra modifié 3 est l'ensemble des points localement cohérents pour les patchs dans $(\mathcal{Z}_i)_{i\in[d]}$.*

3 Expériences

Nous avons testé la diffusion inverse avec le score local optimal (Algo 1) et la version modifiée de Kwatra (Algo 3) sur le jeu de données public MNIST. Afin de se placer dans une configuration avec peu de données d'entraînement, nous avons consideré des sous-ensembles réduits de MNIST (de tailles variant de 1 à 200), contenant uniquement des images de "8". Ci-dessous nous montrons, pour différentes tailles de patch, des images générées par les deux algorithmes. L'expérience de diffusion a été realisée avec un "cosine noise schedule", discrétisé sur $T = 1000$ étapes. Pour la version modifiée de l'algorithme de Kwatra, nous montrons de plus les images les

plus proches dans l'ensemble d'entraînement, au sens de la norme L^2.

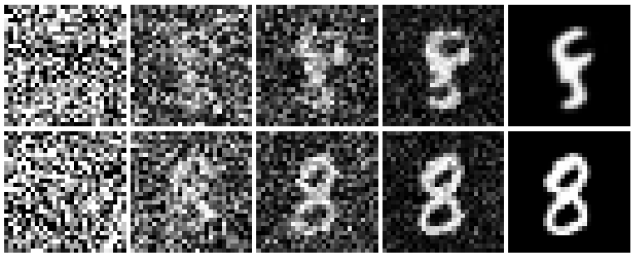


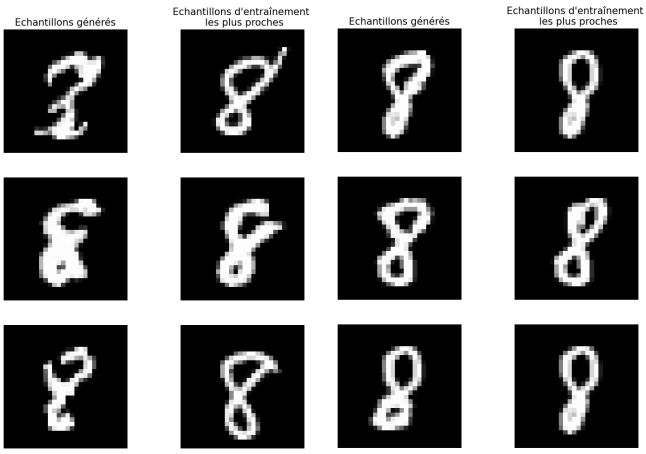
FIGURE 1 : Diffusion résultante avec score local optimal (algo 1). La ligne du haut et la ligne du bas correspondent respectivement à des tailles de patch de 3×3 et 9×9.

Outre la qualité des échantillons générés, il est intéressant d'étudier la marge d'innovation qu'il reste à un algorithme contraint à la cohérence locale, définie plus haut. Pour cela, on quantifie l'innovation d'un échantillon comme la distance en norme L^2 à l'échantillon le plus proche dans l'ensemble d'entraînement. On observe ensuite l'évolution de l'innovation moyenne avec la taille de l'ensemble d'entraînement, pour différentes tailles de patches. Afin que cela soit significatif, on considère une suite de sous-ensembles croissante, et on calcule l'innovation moyenne sur 1000 échantillons générés pour chaque taille de l'ensemble d'entraînement. Les résultats sont rassemblés dans la figure 3.

Sur cette figure, on peut observer un comportement de l'algorithme qui était attendu : l'innovation moyenne est plus grande pour des tailles de patch petites. En effet, une grande taille de patch force l'algorithme à recopier des plus grandes parties de certaines images d'entraînement (lorsque le patch englobe toute l'image, l'algorithme renvoie exactement une des images d'entraînement) lui laissant moins de marge d'innovation. Inversement, une petite taille de patch permet à l'algorithme de plus innover, au prix de structures grande échelle plus difficilement respectées (voir figure 2).

4 Conclusion et perspectives

Dans ce travail nous avons proposé une analyse de créativité des modèles de diffusion DDPM, sous certaines hypothèses du réseau convolutionnel approximant le score. Nous avons montré que dans une configuration avec peu d'échantillons d'entraînements et où le réseau effectue des opérations locales, le processus inverse de diffusion n'est qu'une version modifiée d'un célèbre algorithme de synthèse de texture. Dans un travail futur, nous étudierons l'optimalité de la fonction de perte des modèles de diffusion sur des ensembles de fonctions plus représentatifs des architectures de réseau utilisés en pratique. En particulier, il serait intéressant d'étudier des ensembles de fonctions qui rendent compte de l'attention des réseaux à plusieurs résolutions de l'image, comme c'est le cas des réseaux avec une architecture de type U-Net.



Patch 3×3 Patch 9×9

FIGURE 2 : Echantillons générés par l'algorithme modifié de Kwatra et les échantillons d'entraînement les plus proches, pour 2 tailles de patch différentes.

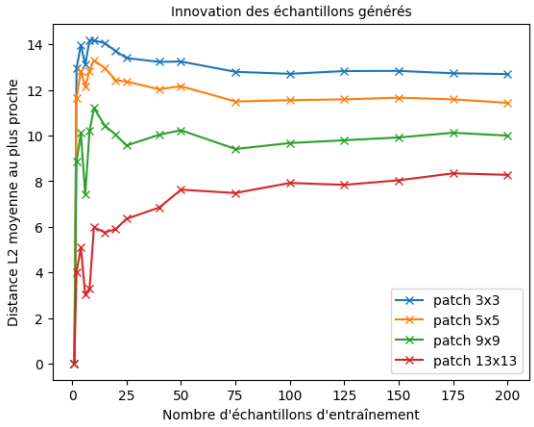


FIGURE 3 : Evolution de l'innovation moyenne de l'algorithme modifié de Kwatra sur la classe "8" de MNIST avec le nombre d'échantillons d'entraînements.

Références

[1] Jonathan HO, Ajay JAIN et Pieter ABBEEL : Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[2] Mason KAMB et Surya GANGULI : An analytic theory of creativity in convolutional diffusion models. *arXiv preprint arXiv :2412.20292*, 2024.

[3] Vivek KWATRA, Irfan ESSA, Aaron BOBICK et Nipun KWATRA : Texture optimization for example-based synthesis. *In ACM Siggraph 2005 Papers*, pages 795–802. 2005.

[4] Yann LECUN, Bernhard BOSER, John DENKER, Donnie HENDERSON, Richard HOWARD, Wayne HUBBARD et Lawrence JACKEL : Handwritten digit recognition with a back-propagation network. *Advances in neural information processing systems*, 2, 1989.