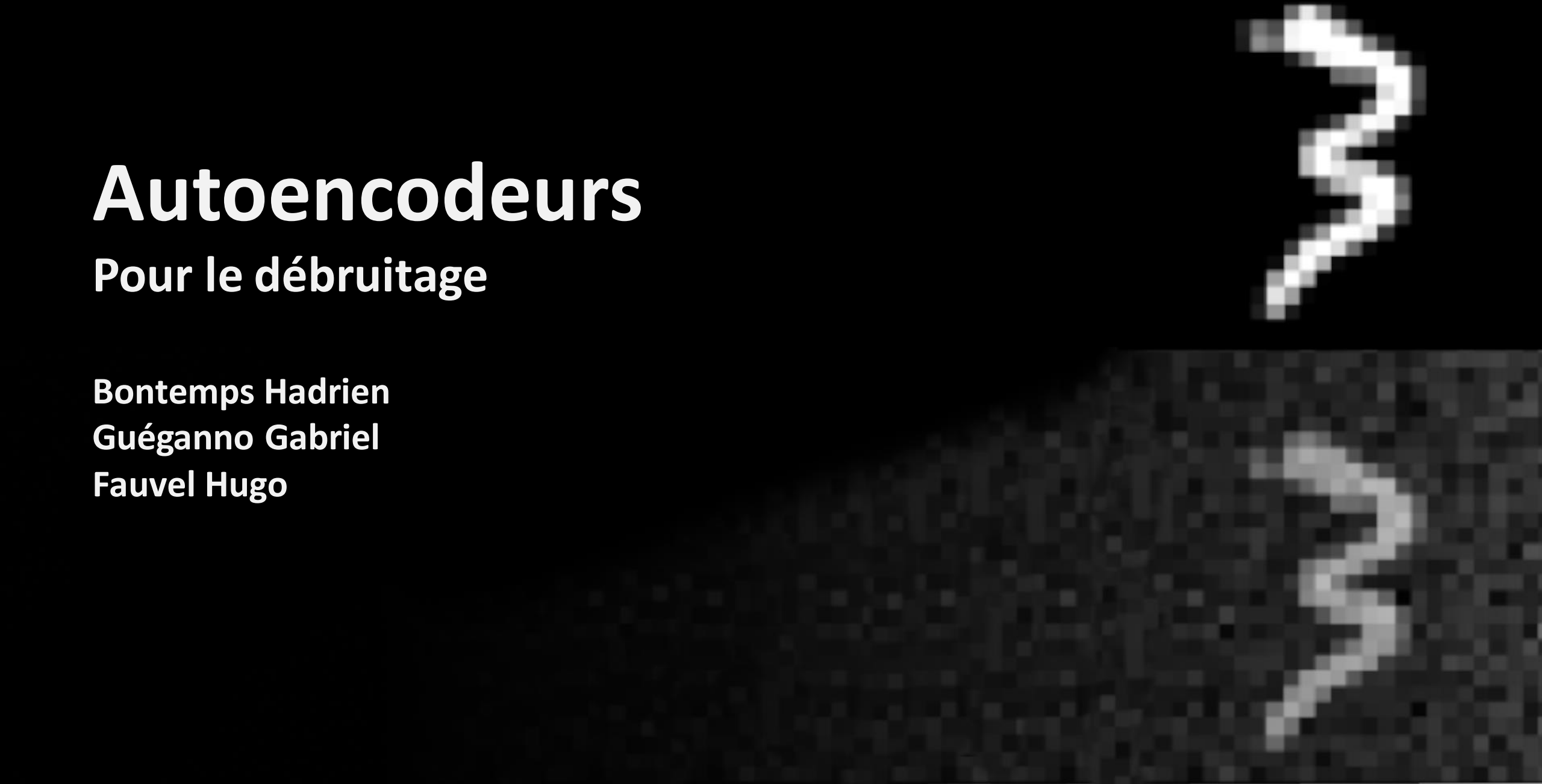
Autoencodeurs

Pour le débruitage

Bontemps Hadrien
Guéganno Gabriel
Fauvel Hugo



Plan

I. Principes des autoencodeurs
    I. Philosophie des autoencodeurs
    II. Architecture et fonction objectif
    III. Réduction du bruit par denoising autoencodeur (DAE)

II. Méthode
    I. Base de données
    II. Pré-traitement des données
    III. Architectures du réseau de neurone

III. Résultats
    I. Résultats en fonction du bruit
    II. Interprétation

IV. Conclusion

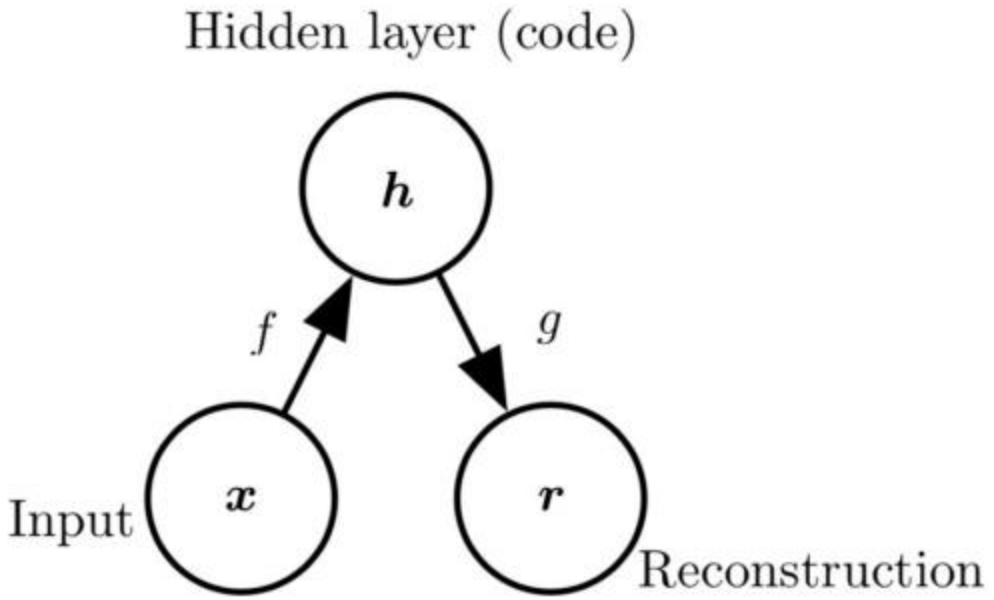
I - Principes des autoencodeurs

Autoencodeur

Objectif: Reconstruire la donnée initiale

Contrainte: Modifier l'espace de représentation en passant par au moins une couche cachée, souvent de taille inférieure aux données initiales

Non supervisé, il n'y a pas de label

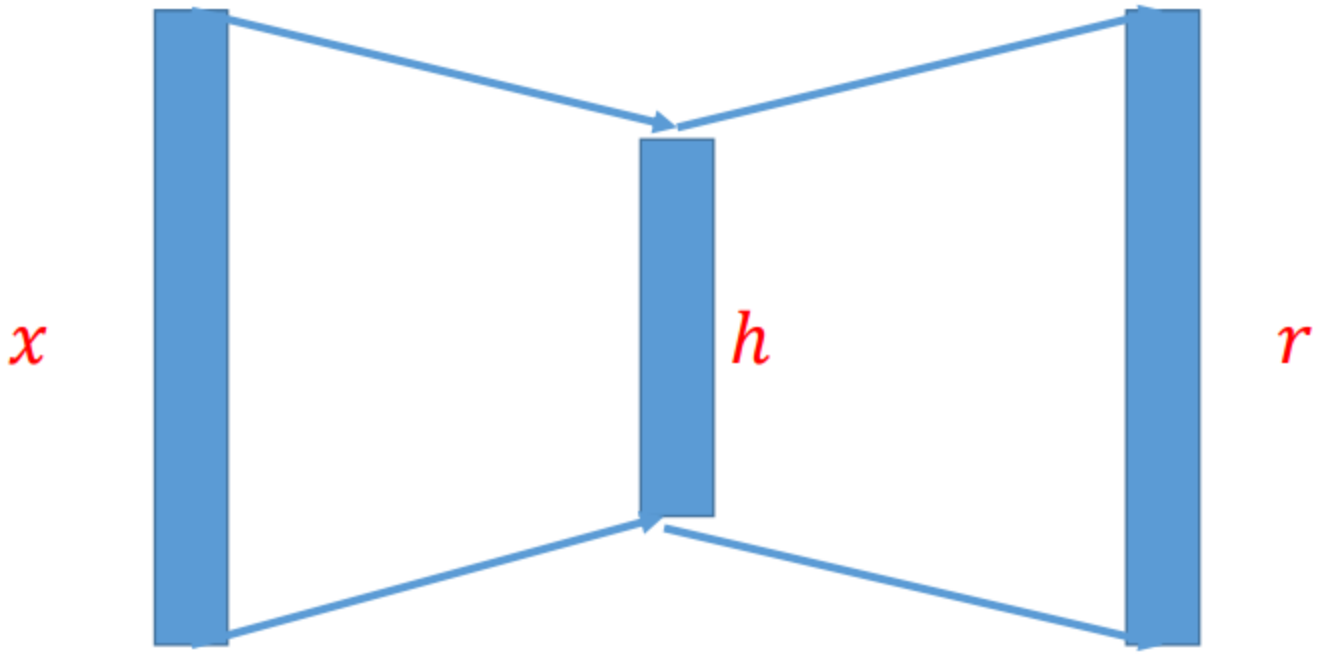


I - Principes des autoencodeurs

$$L(x, r) = L(x, g(f(x)))$$

La fonction objectif à minimiser, représentée par la fonction de perte, est la différence entre l'entré et la sortie

$$L(\mathbf{x}, g(f(\mathbf{x})))$$



I - Principes des autoencodeurs

Familles d'autoencodeurs

- Sparse
- Denoising
- Variational
- Concrete
- ...

Applications

- Extraction de caractéristiques
- Réduction de la dimension
- Débruitage
- Segmentation
- …

1986-1991

Mark A. Kramer, "Nonlinear Principal Component Analysis Using Autoassociative Neural Networks", MIT, Laboratory for Intelligent Systems in Process Engineering, 1991

2008

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 1096–1103, New York, NY, USA, 2008

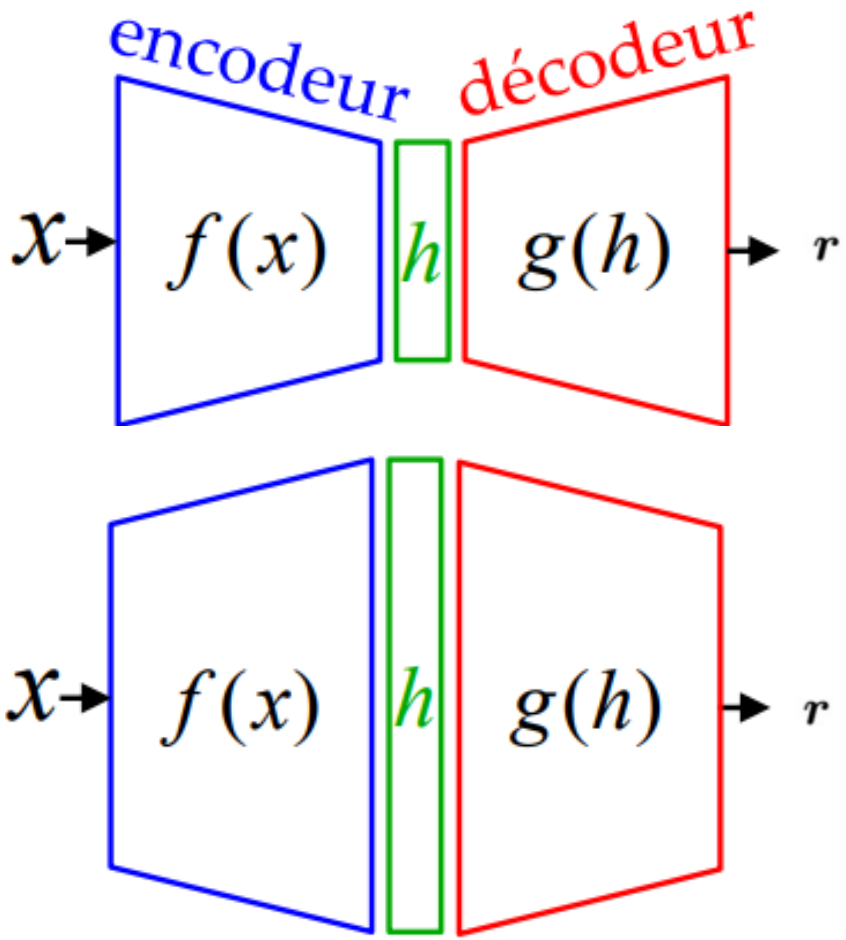
2015

Olaf Ronneberger, Philipp Fischer et Thomas Brox, « U-Net: Convolutional Networks for Biomedical Image Segmentation », *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Springer International Publishing, 2015, p. 234–241

I - Principes des autoencodeurs

Sous-complet:
- Dimension de h inférieur à l'entrée
- Projection vers un espace de dimension plus petit
- Si f et g sont linéaires, comportement proche voir équivalent à une PCA
- Si f et g sont non-linéaires, la projection peut être plus "puissante"

Sur-complet:
- Dimension de h supérieure à l'entrée
- Recopie d'information
- Doit être régularisé

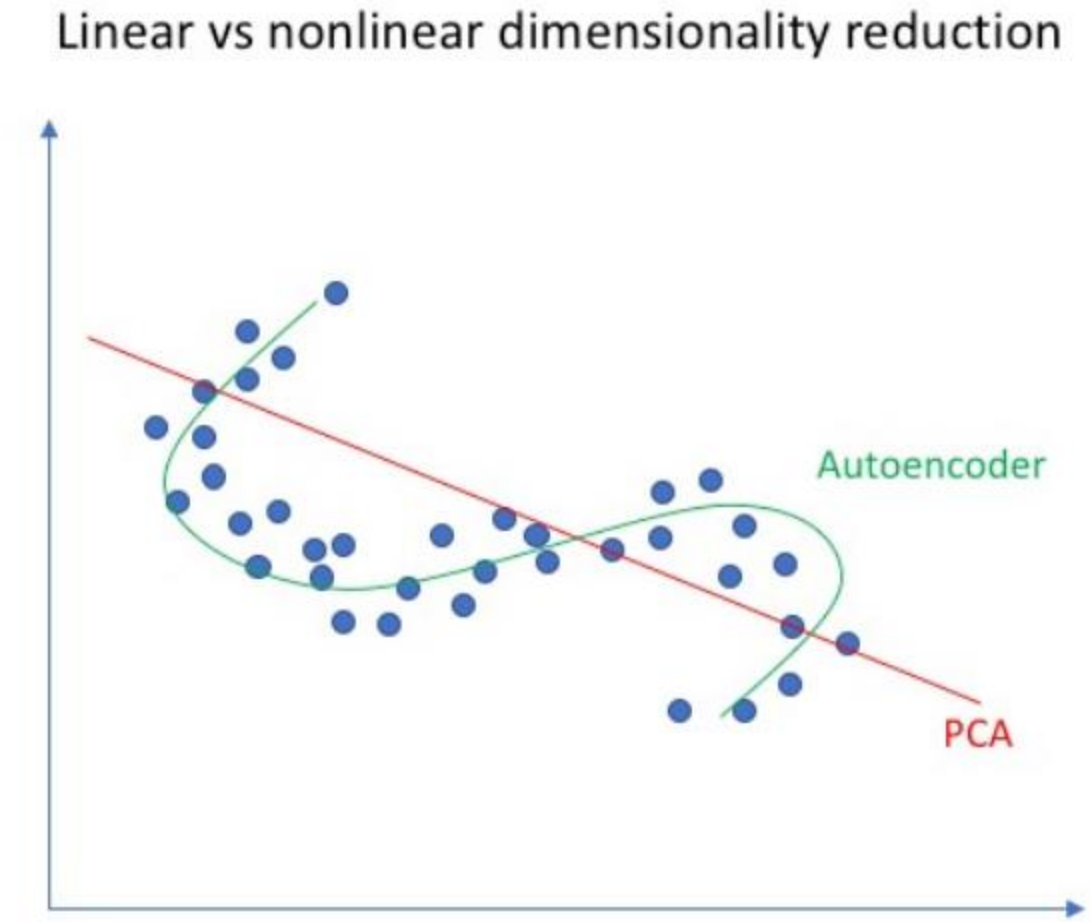


I - Principes des autoencodeurs

Propriété

Un autoencodeur est équivalent à une PCA si

- L'encodeur est linéaire
- Le décodeur est linéaire
- La loss est la mean square error
- Les données d'entrées sont normalisées



I - Principes des autoencodeurs

Quelques risques

Si l'encodeur non-linéaire à un pouvoir séparateur très grand, une entrée I peut corresponde à un indice I dans la couche latent.

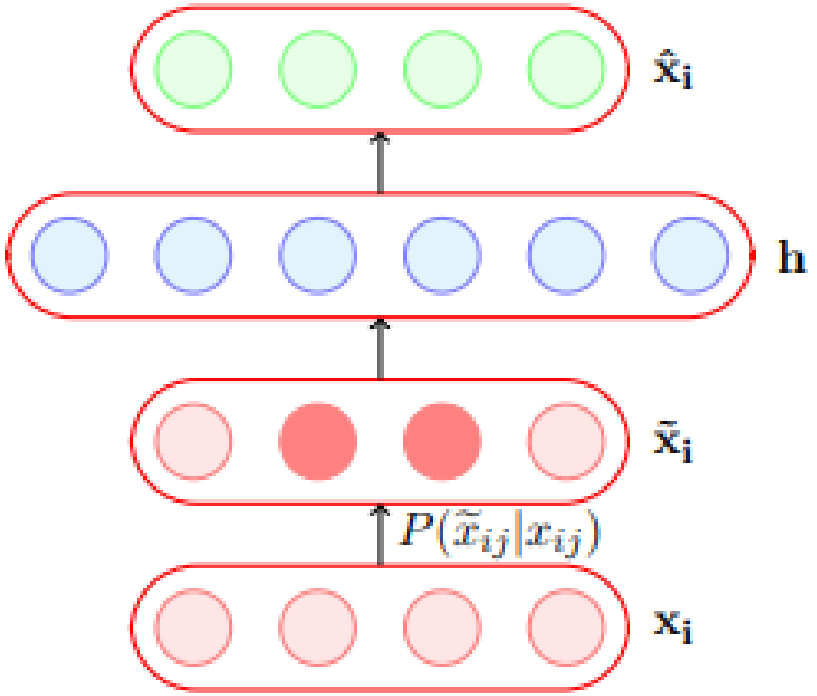
Un pré-entrainement *Greedy Layer-Wise* est conseillé afin d'éviter la disparition du gradient

Bengio, Y., Lamblin, P., Popovici, D., & Larochelle, H. (2007), "Greedy layer-wise training of deep networks. ", Advances in Neural Information Processing Systems 19 (pp. 153–160). MIT Press.

I - Principes des autoencodeurs

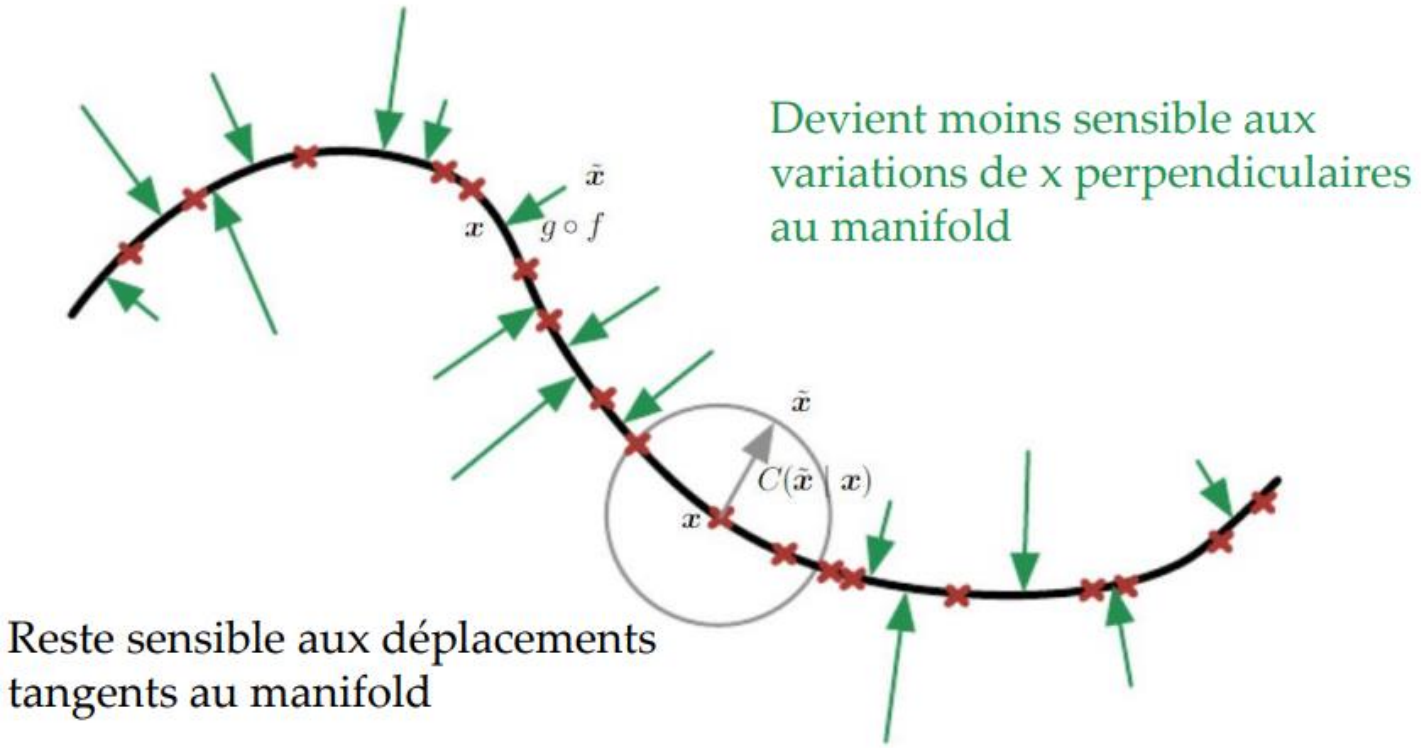
Dans un DAE, la fonction objective compare le signal reconstruit avec le signal non bruité.

$$L(\mathbf{x}, g(f(\tilde{\mathbf{x}}))$$



Q. You, Y. J. Zhang, "A new training principle for stacked denoising autoencoders", in Seventh International COnference on Image and Graphics, Qingdao, 2013, pp 384-389 (sur des chiffres)
K. Wu, Z. Gao, C. Pen, X. Wen, "Text window denoising autoencoder: building deep architecture for Chinese word segmentation", Commun Comput Inf. Sci., 2013 (sur des sinogrammes)

I - Principes des autoencodeurs



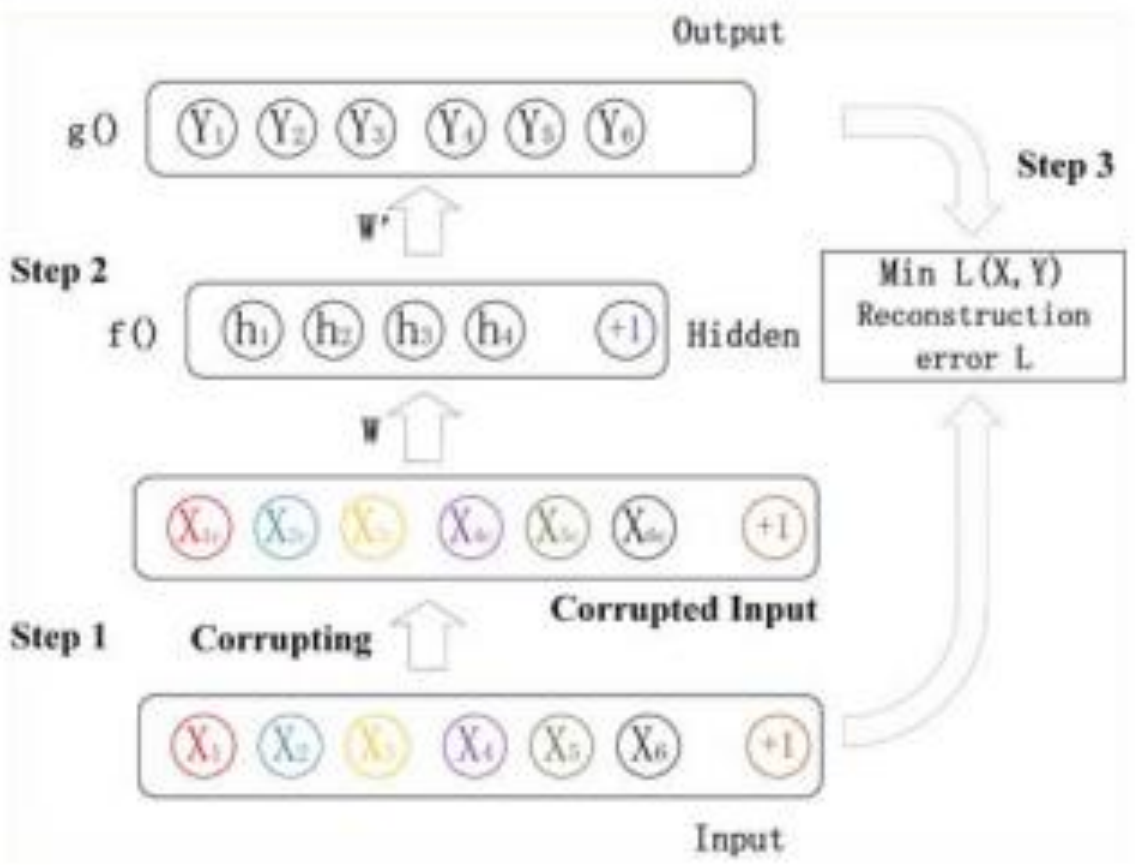
Extraire caractéristiques par la corrélation entre les signaux d'entrés

II - Méthode

Bases de données

Nom	Signaux	MNIST	Sinogramme
Remarque	Signaux à une, deux ou trois fréquences	Chiffre de 0 à 9	15 sinogrammes
Taille	4096x1	28x28	32x32
Cardinal	Base d'apprentissage: 900 Base de test: 100	Base d'apprentissage: 60000 Base de test: 10000	Base d'apprentissage: 14000 Base de test: 1000

II - Méthode



De M. Lingheng, D. Shifei, X. Yu, "Research on denoising sparse autoencoder", in International Journal of Machine Learning and Cybernetics, 2017.

$$SNR = 10\ log_{10} \frac{\sum_{x=1}^{M} \sum_{y=1}^{N} I(x,y)^2}{\sum_{x=1}^{M} \sum_{y=1}^{N} \left[I(x,y) - \hat{I}\ (x,y) \right]^2}$$

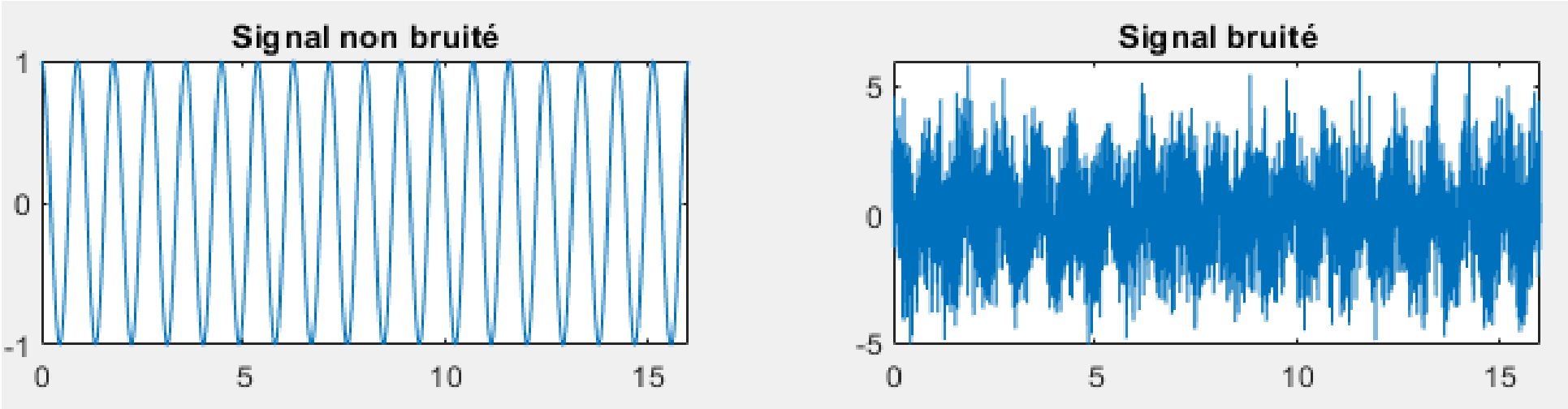
$$PSNR = 10\ log_{10} \frac{255^2}{MSE}$$

II - Méthode

Débruitage de signaux

Objectif : Supprimer le bruit gaussien blanc d'un signal (1D)

Dataset : Signaux à une fréquence bruités et non bruités



II - Méthode

Débruitage de signaux

Deux types de réseaux testés :

Input : signal (1x4096)

Output : signal (1x4096)

Couches denses

Layer (type)	Output Shape	Param #
Linear-1	[-1, 1, 1024]	4,195,328
ReLU-2	[-1, 1, 1024]	0
Linear-3	[-1, 1, 512]	524,800
ReLU-4	[-1, 1, 512]	0
Linear-5	[-1, 1, 256]	131,328
ReLU-6	[-1, 1, 256]	0
Linear-7	[-1, 1, 128]	32,896
ReLU-8	[-1, 1, 128]	0
Linear-9	[-1, 1, 256]	33,024
ReLU-10	[-1, 1, 256]	0
Linear-11	[-1, 1, 512]	131,584
ReLU-12	[-1, 1, 512]	0
Linear-13	[-1, 1, 1024]	525,312
ReLU-14	[-1, 1, 1024]	0
Linear-15	[-1, 1, 4096]	4,198,400
Sigmoid-16	[-1, 1, 4096]	0

Total params: 9,772,672
Trainable params: 9,772,672
Non-trainable params: 0

Couches de convolution 1D
+ MaxPooling

Layer (type)	Output Shape	Param #
Conv1d-1	[-1, 128, 4096]	512
ReLU-2	[-1, 128, 4096]	0
MaxPool1d-3	[-1, 128, 2048]	0
Conv1d-4	[-1, 64, 2048]	24,640
ReLU-5	[-1, 64, 2048]	0
MaxPool1d-6	[-1, 64, 1024]	0
Conv1d-7	[-1, 32, 1024]	6,176
ReLU-8	[-1, 32, 1024]	0
MaxPool1d-9	[-1, 32, 512]	0
Conv1d-10	[-1, 16, 512]	1,552
ReLU-11	[-1, 16, 512]	0
ConvTranspose1d-12	[-1, 32, 1024]	1,568
ReLU-13	[-1, 32, 1024]	0
ConvTranspose1d-14	[-1, 64, 2048]	6,208
ReLU-15	[-1, 64, 2048]	0
ConvTranspose1d-16	[-1, 128, 4096]	24,704
ReLU-17	[-1, 128, 4096]	0
ConvTranspose1d-18	[-1, 1, 4096]	385
Sigmoid-19	[-1, 1, 4096]	0

Total params: 65,745
Trainable params: 65,745
Non-trainable params: 0

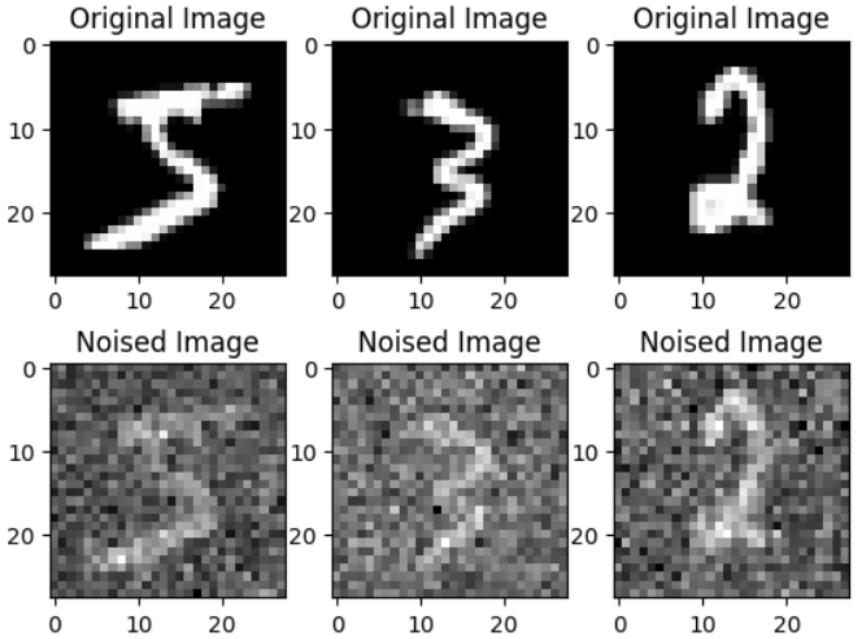
II - Méthode

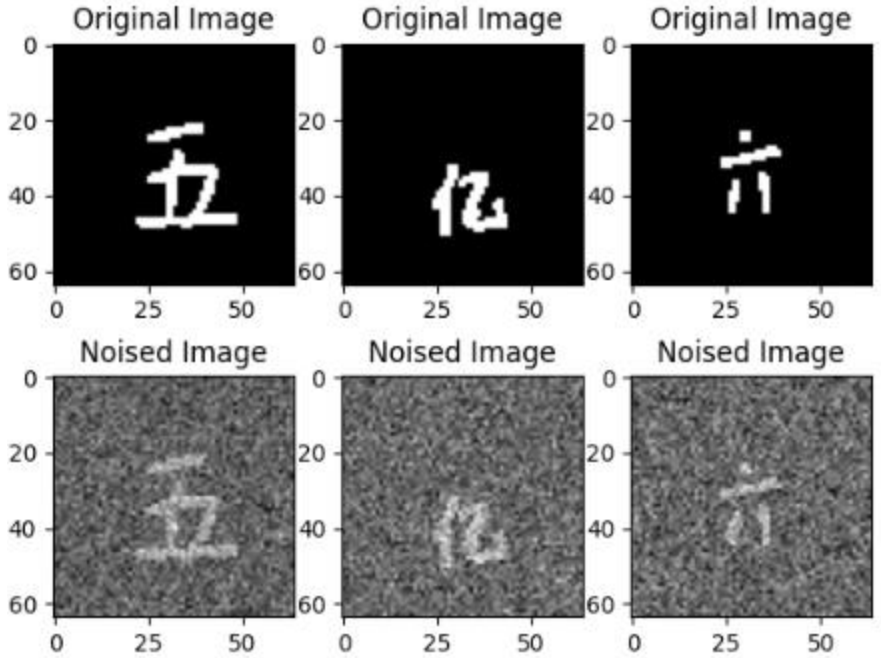
Débruitage d'images

Objectif : Supprimer le bruit gaussien blanc d'une image (2D)

Dataset :

- base de données MNIST
- Base de données Sinogrammes





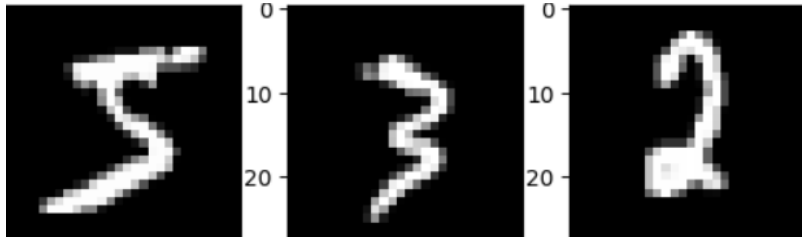
II - Méthode

Débruitage de la base de données MNIST

Un réseau à base de couches denses

Input : image aplatie (1x784)

Output : image aplatie (1x784)



Layer (type)	Output Shape	Param #
Linear-1	[-1, 1, 256]	200,960
ReLU-2	[-1, 1, 256]	0
Linear-3	[-1, 1, 128]	32,896
ReLU-4	[-1, 1, 128]	0
Linear-5	[-1, 1, 64]	8,256
ReLU-6	[-1, 1, 64]	0
Linear-7	[-1, 1, 128]	8,320
ReLU-8	[-1, 1, 128]	0
Linear-9	[-1, 1, 256]	33,024
ReLU-10	[-1, 1, 256]	0
Linear-11	[-1, 1, 784]	201,488
Sigmoid-12	[-1, 1, 784]	0

Total params: 484,944
Trainable params: 484,944
Non-trainable params: 0

II - Méthode

Input : image aplatie (1x1024)

Output : image aplatie (1x1024)

Débruitage de la base de données de sinogrammes

Un réseau à base de couches denses

Layer (type)	Output Shape	Param #
Linear-1	[-1, 1, 256]	262,400
ReLU-2	[-1, 1, 256]	0
Linear-3	[-1, 1, 128]	32,896
ReLU-4	[-1, 1, 128]	0
Linear-5	[-1, 1, 64]	8,256
ReLU-6	[-1, 1, 64]	0
Linear-7	[-1, 1, 128]	8,320
ReLU-8	[-1, 1, 128]	0
Linear-9	[-1, 1, 256]	33,024
ReLU-10	[-1, 1, 256]	0
Linear-11	[-1, 1, 1024]	263,168
Sigmoid-12	[-1, 1, 1024]	0

Total params: 608,064
Trainable params: 608,064
Non-trainable params: 0

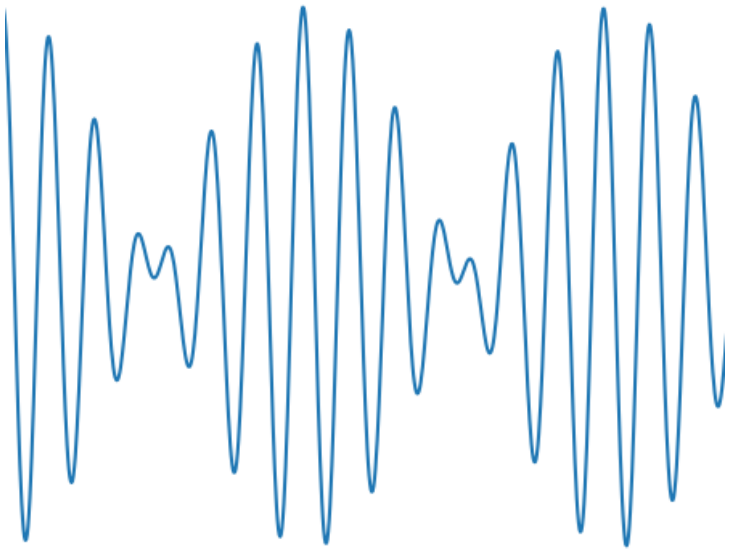
Un réseau à base de convolutions 1D

Layer (type)	Output Shape	Param #
Conv1d-1	[-1, 128, 1024]	512
ReLU-2	[-1, 128, 1024]	0
MaxPool1d-3	[-1, 128, 512]	0
Conv1d-4	[-1, 64, 512]	24,640
ReLU-5	[-1, 64, 512]	0
MaxPool1d-6	[-1, 64, 256]	0
Conv1d-7	[-1, 32, 256]	6,176
ReLU-8	[-1, 32, 256]	0
MaxPool1d-9	[-1, 32, 128]	0
Conv1d-10	[-1, 16, 128]	1,552
ReLU-11	[-1, 16, 128]	0
ConvTranspose1d-12	[-1, 32, 256]	1,568
ReLU-13	[-1, 32, 256]	0
ConvTranspose1d-14	[-1, 64, 512]	6,208
ReLU-15	[-1, 64, 512]	0
ConvTranspose1d-16	[-1, 128, 1024]	24,704
ReLU-17	[-1, 128, 1024]	0
ConvTranspose1d-18	[-1, 1, 1024]	385
Sigmoid-19	[-1, 1, 1024]	0

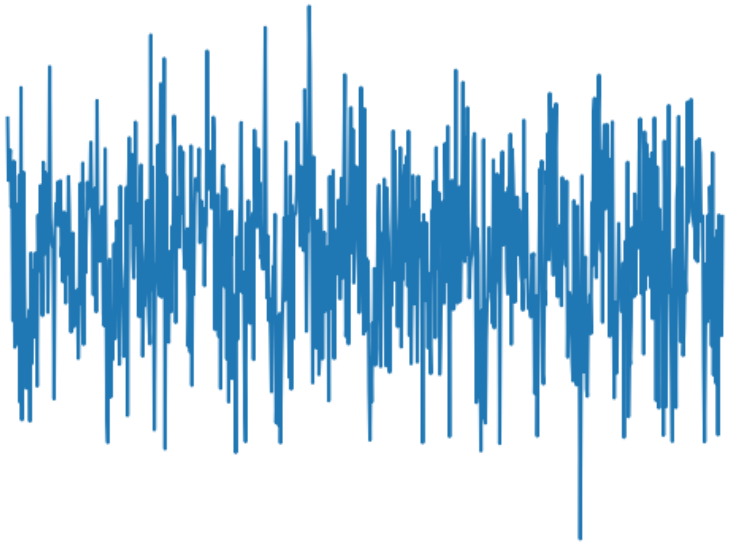
Total params: 65,745
Trainable params: 65,745
Non-trainable params: 0

III - Résultats

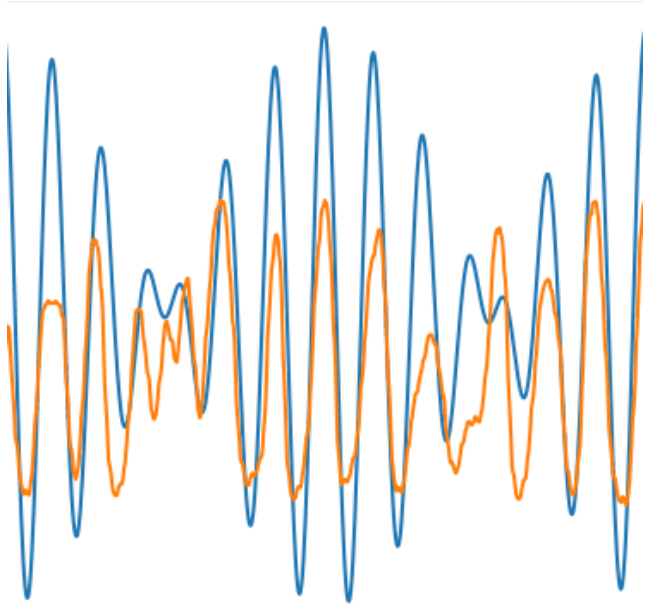
Signaux 1D



Signal initial (f0, f1)



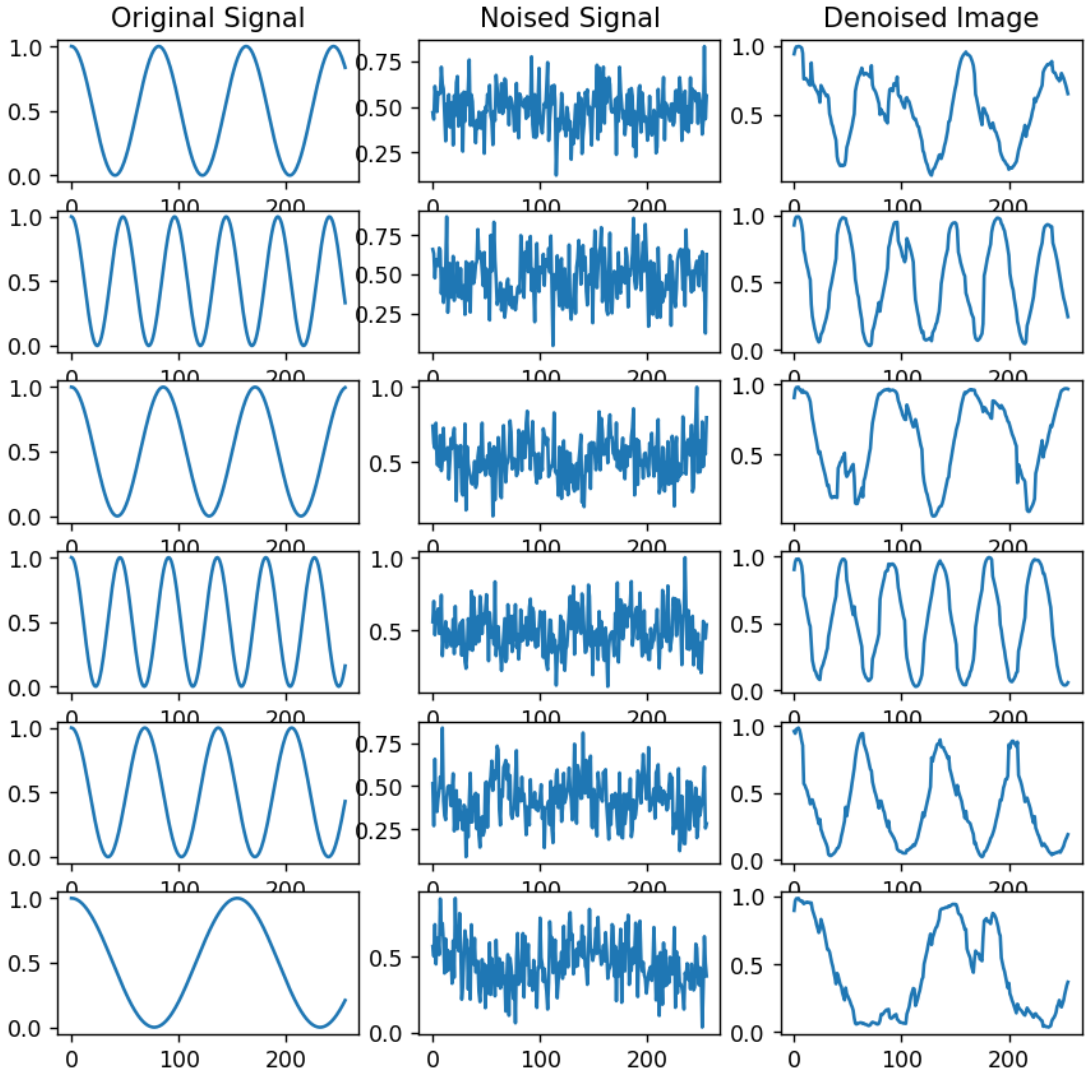
Signal bruité (awgn)



Signal débruité à différentes EPOCH de l'entraînement (EPOCH 1 à 50)

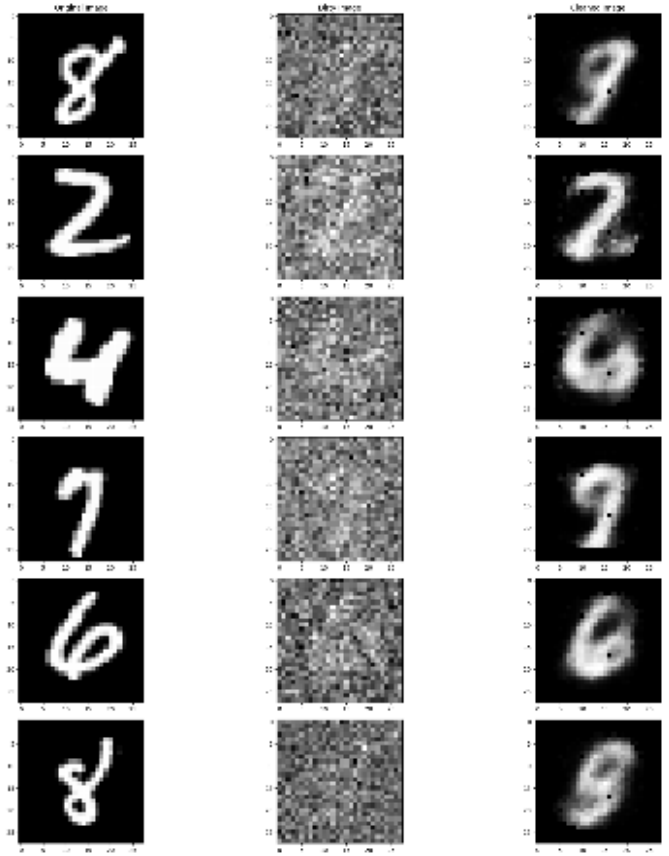
III - Résultats

Signaux 1D

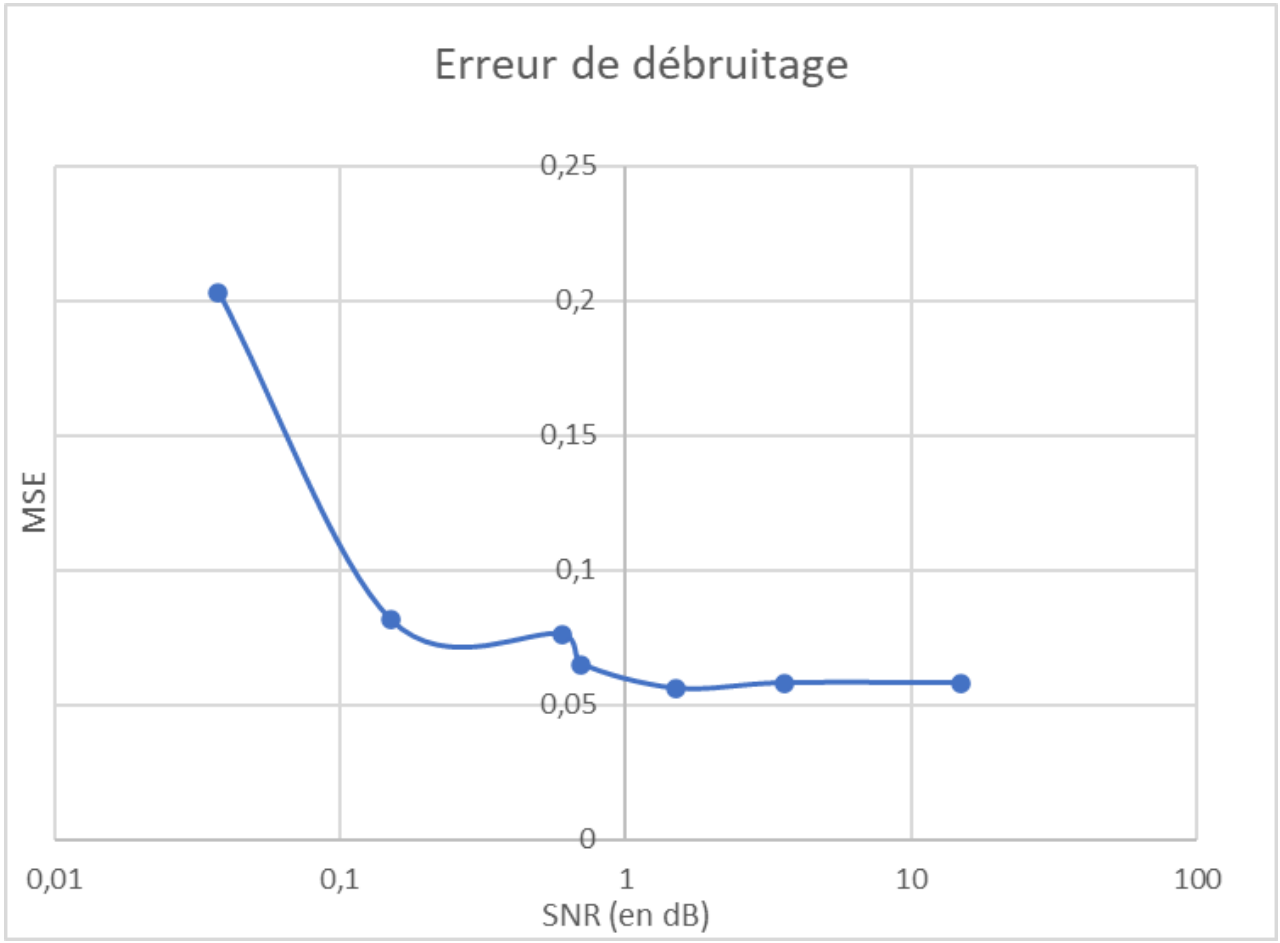


III - Résultats

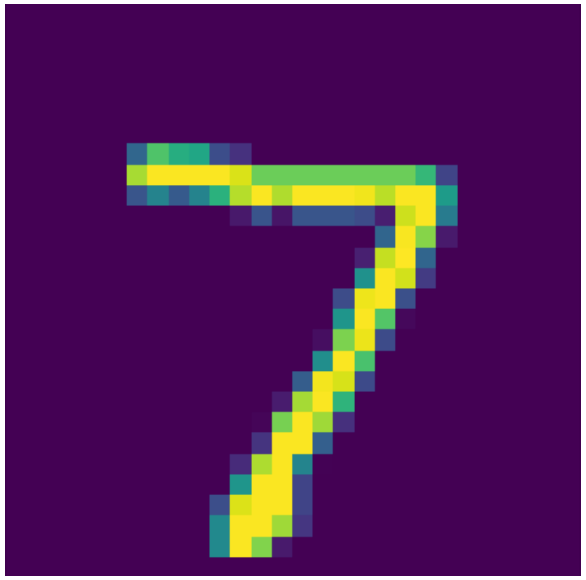
Chiffres manuscrits (MNIST)



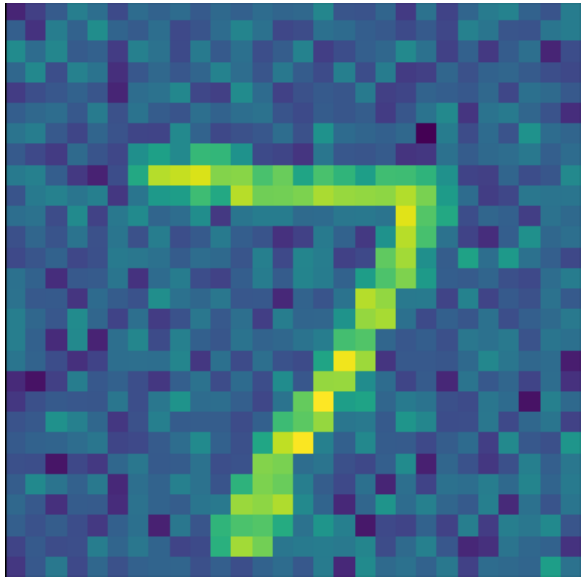
Reconstruction d'images
test pour un SNR de 0.15



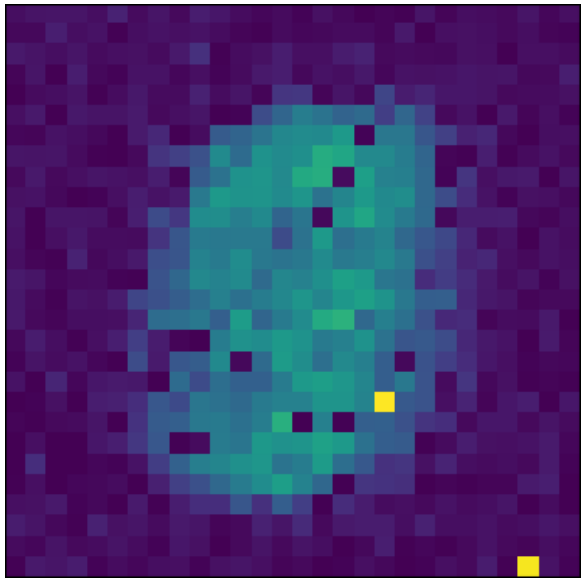
III - Résultats



Chiffre initial



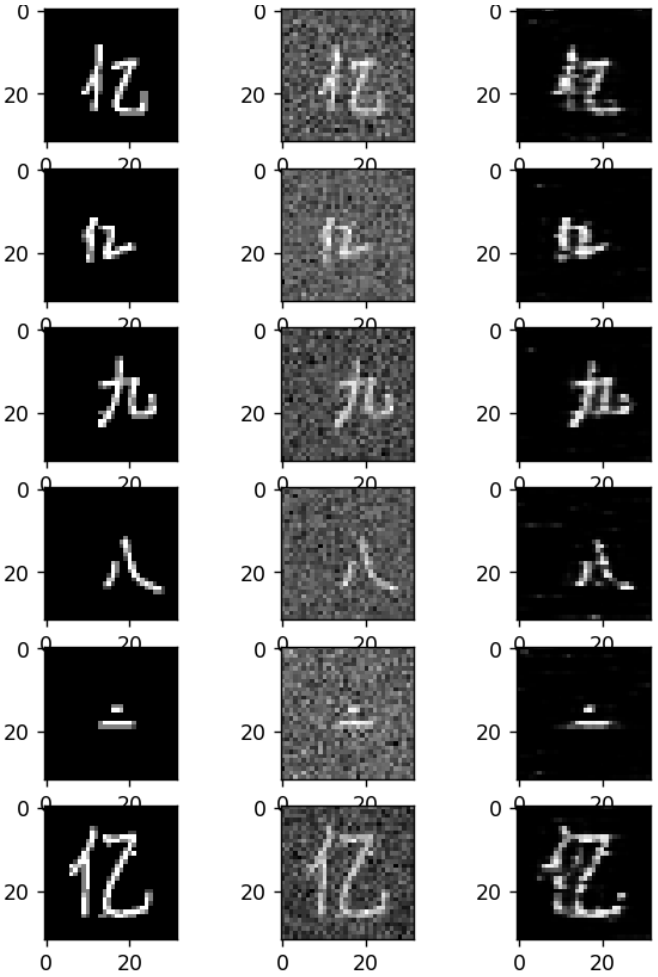
Chiffre bruité (awgn)



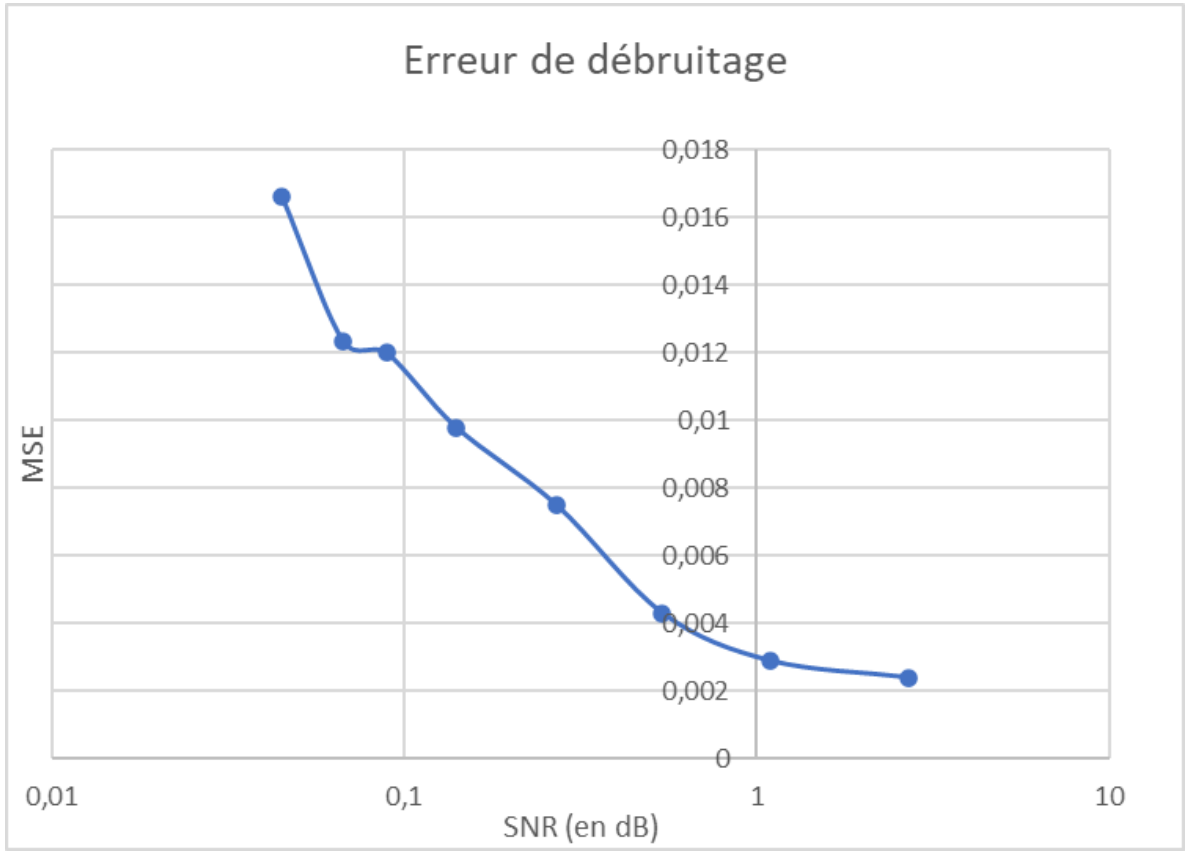
Chiffre débruité à différentes EPOCH de l'entraînement (EPOCH 1 à 50)

III - Résultats

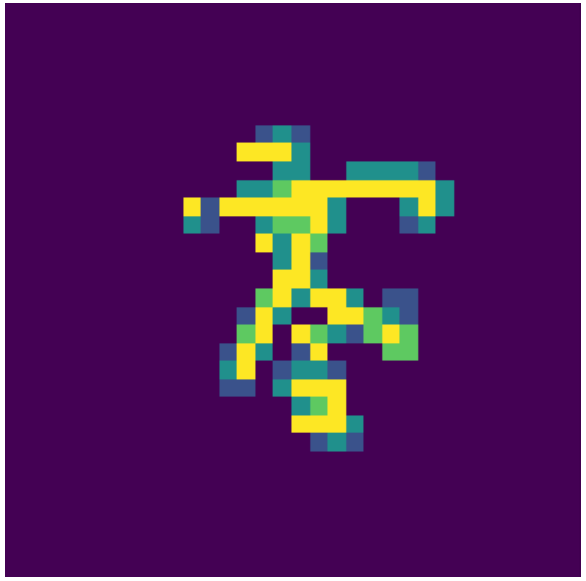
Sinogrammes



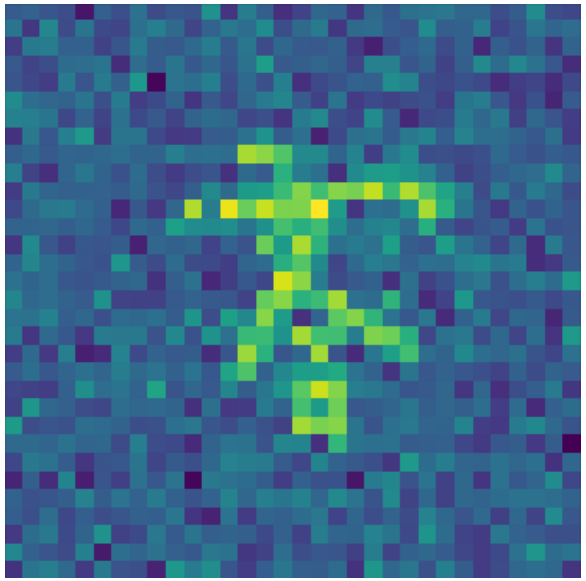
Reconstruction d'images test pour un SNR de 0.25



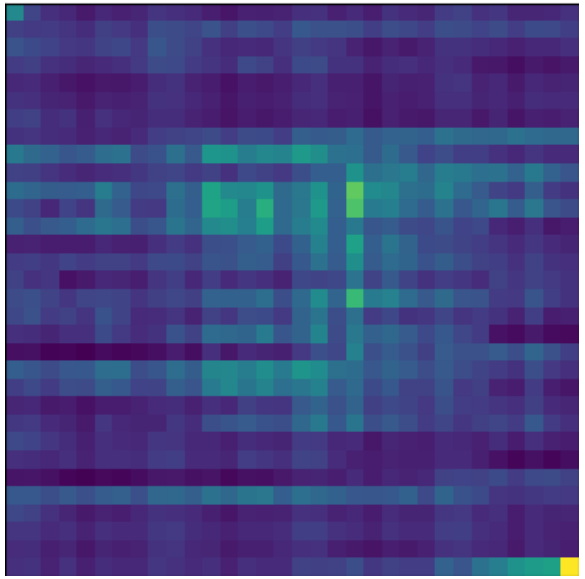
III - Résultats



Sinogramme initial



Sinogramme bruité (awgn)

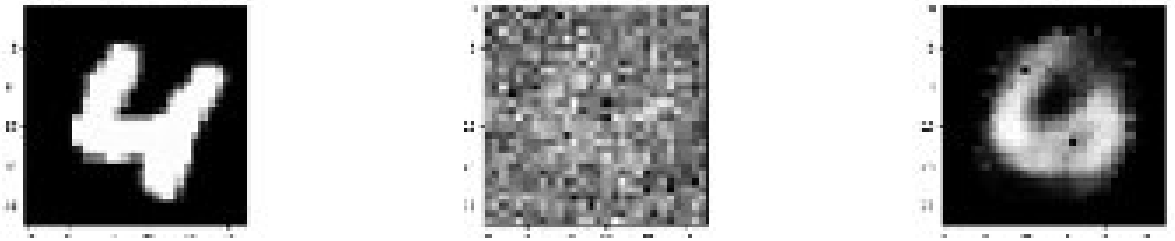


Sinogramme débruité à différentes EPOCH de l'entrainement (EPOCH 1 à 50)

IV - Conclusion

Bilan:

- Autoencodeurs efficaces pour le débruitage de signaux simples et modérément bruités



(SNR = 0.15)

Limites:

- Temps de calcul
- La MSE ne permet pas de pénaliser un signal reconstruit différent du signal initial lorsque les deux signaux sont proches au sens des moindres carrés

Perspectives:

- Ajouter une branche de classification pour pénaliser un signal reconstruit proche d'un autre élément de la base ?