Les auto-encodeurs de débruitage

Mathis Aulagnier – UQAC – 8INF804

Objectif

Débruiter des images
Analyser les caractéristiques extraites

Motivation, application possible

Améliorer les performance de modèles confrontés à des données bruitées.
Data Augmentation

Jeu de donnée

MNIST

Base de données de chiffres écrits à la main.

Sommaire

01
Architecture de l'auto-encodeur

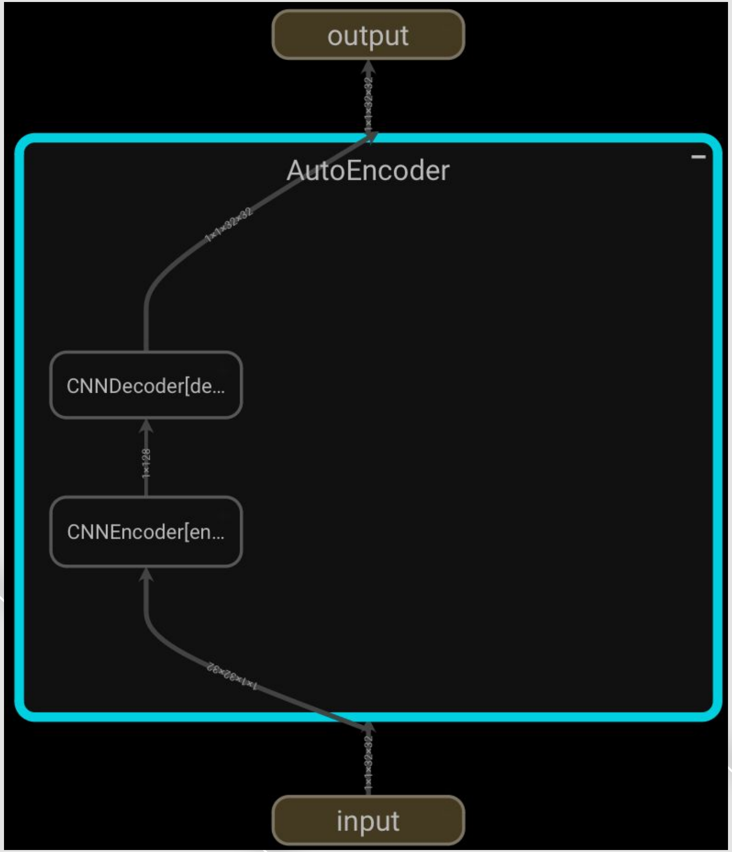
Architecture

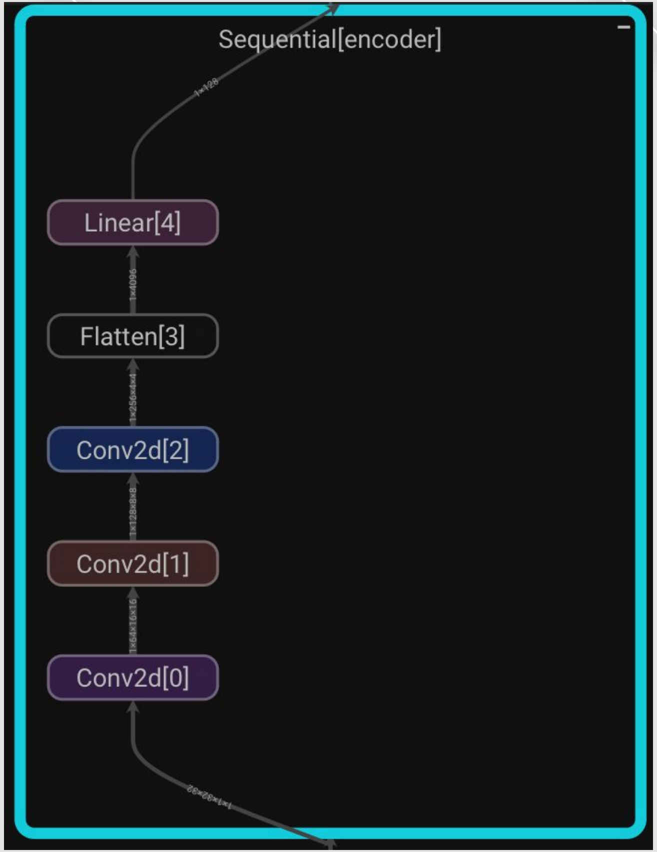
```
class CNNEncoder(nn.Module):
    def __init__(self, latent_dim=128, in_channel=1):
        super(CNNEncoder, self).__init__()
        self.encoder = nn.Sequential(
            nn.Conv2d(in_channel, 64, kernel_size=3, padding=1, stride=2), # 1x32x32 -> 64x14x14
            nn.ReLU(),
            nn.BatchNorm2d(64),

            nn.Conv2d(64, 128, kernel_size=3, padding=1, stride=2),# 64x14x14 -> 128x7x7
            nn.ReLU(),
            nn.BatchNorm2d(128),

            nn.Conv2d(128, 256, kernel_size=3, padding=1, stride=2),# 128x7x7 -> 256x4x4
            nn.ReLU(),
            nn.BatchNorm2d(256),

            nn.Flatten(), # 256x4x4 -> 4096
            nn.Linear(256 * 4 * 4, latent_dim) # 4096 -> latent_dim = 128
        )

    def forward(self, x):
        return self.encoder(x)
```

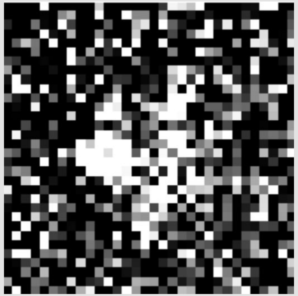
Architecture

```
class CNNDecoder(nn.Module):
    def __init__(self, latent_dim=128, in_channel=1):
        super(CNNDecoder, self).__init__()
        self.decoder = nn.Sequential(
            nn.Linear(latent_dim, 256 * 4 * 4),
            nn.Unflatten(1, (256, 4, 4)),
            nn.ConvTranspose2d(256, 128, kernel_size=3, stride=2, padding=1, output_padding=1),
            nn.ReLU(),
            nn.BatchNorm2d(128),
            nn.ConvTranspose2d(128, 64, kernel_size=3, stride=2, padding=1, output_padding=1),
            nn.ReLU(),
            nn.BatchNorm2d(64),
            nn.ConvTranspose2d(64, in_channel, kernel_size=3, stride=2, padding=1, output_padding=1),
            nn.Tanh()
        )

    def forward(self, x):
        return self.decoder(x)
```

TensorBoard







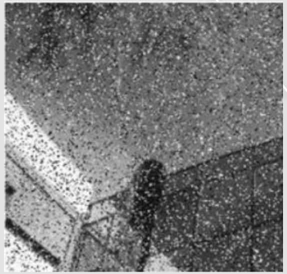
02
Entrainement du débruiteur

Le Bruit ?

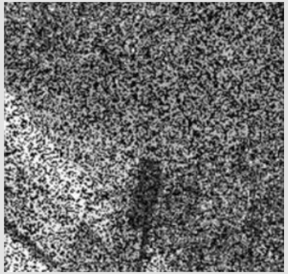
Qu’est-ce que le bruit ?

- **Bruit gaussien**

- **Bruit de sel et poivre**



- **Bruit speckle**

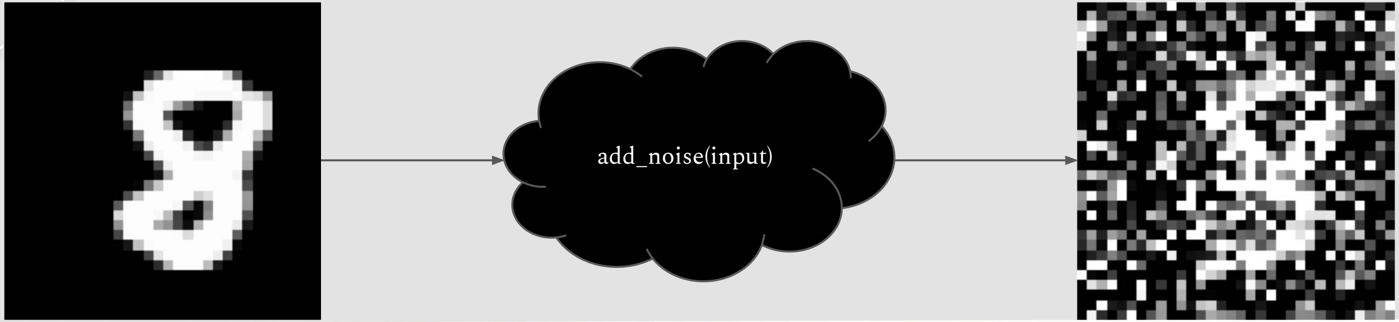


- **Bruit périodique**

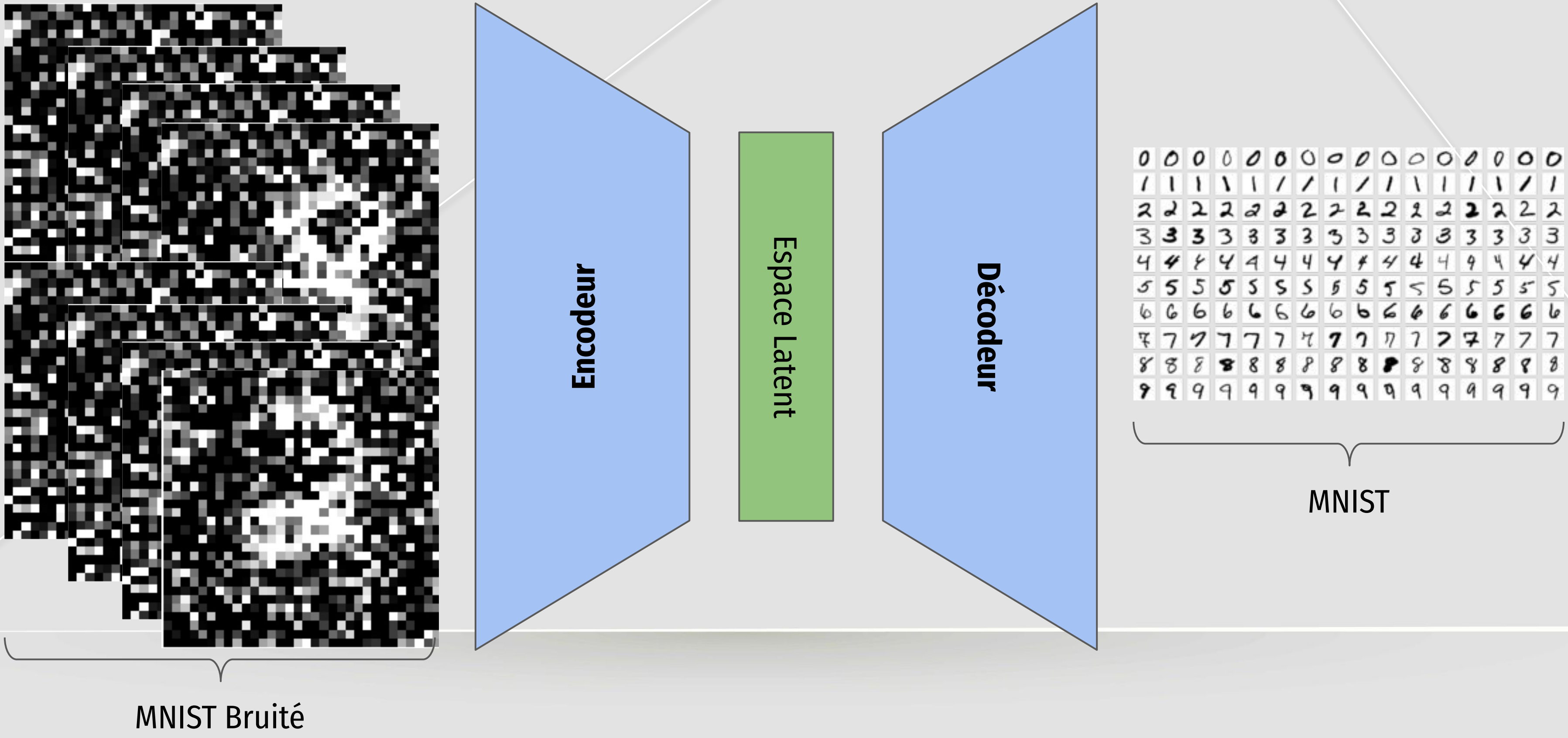
Exemple de bruit

Qu’est-ce que le bruit ?

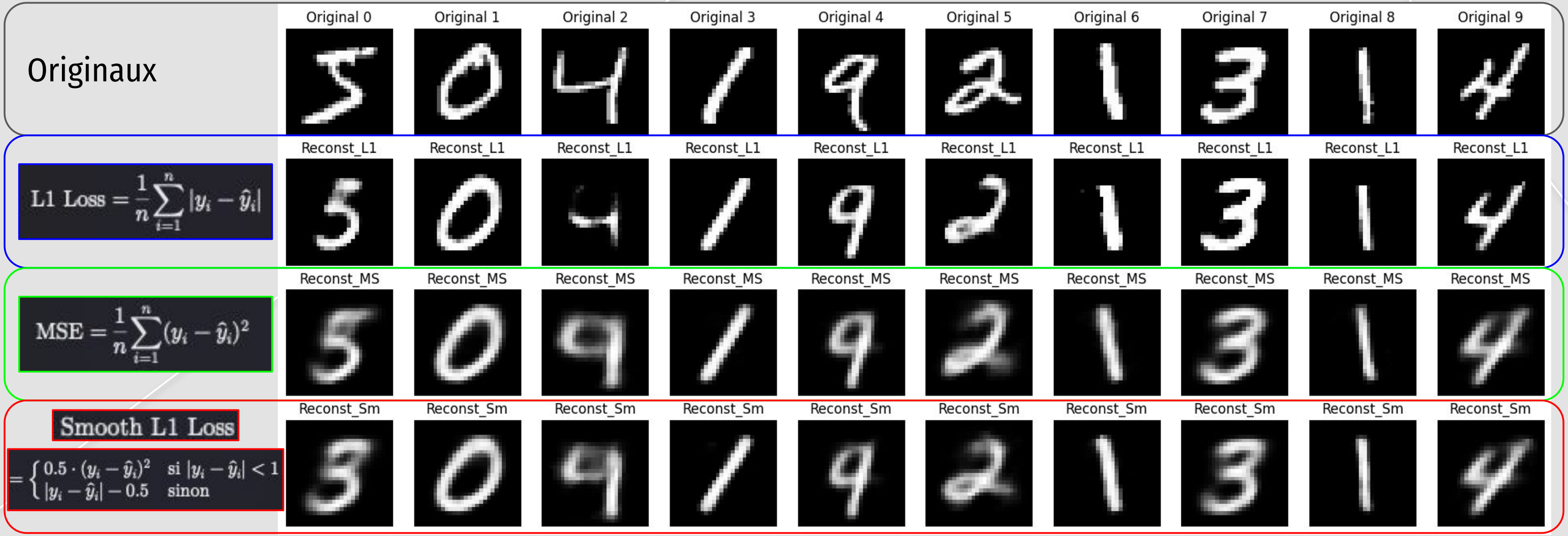
Bruit gaussien :



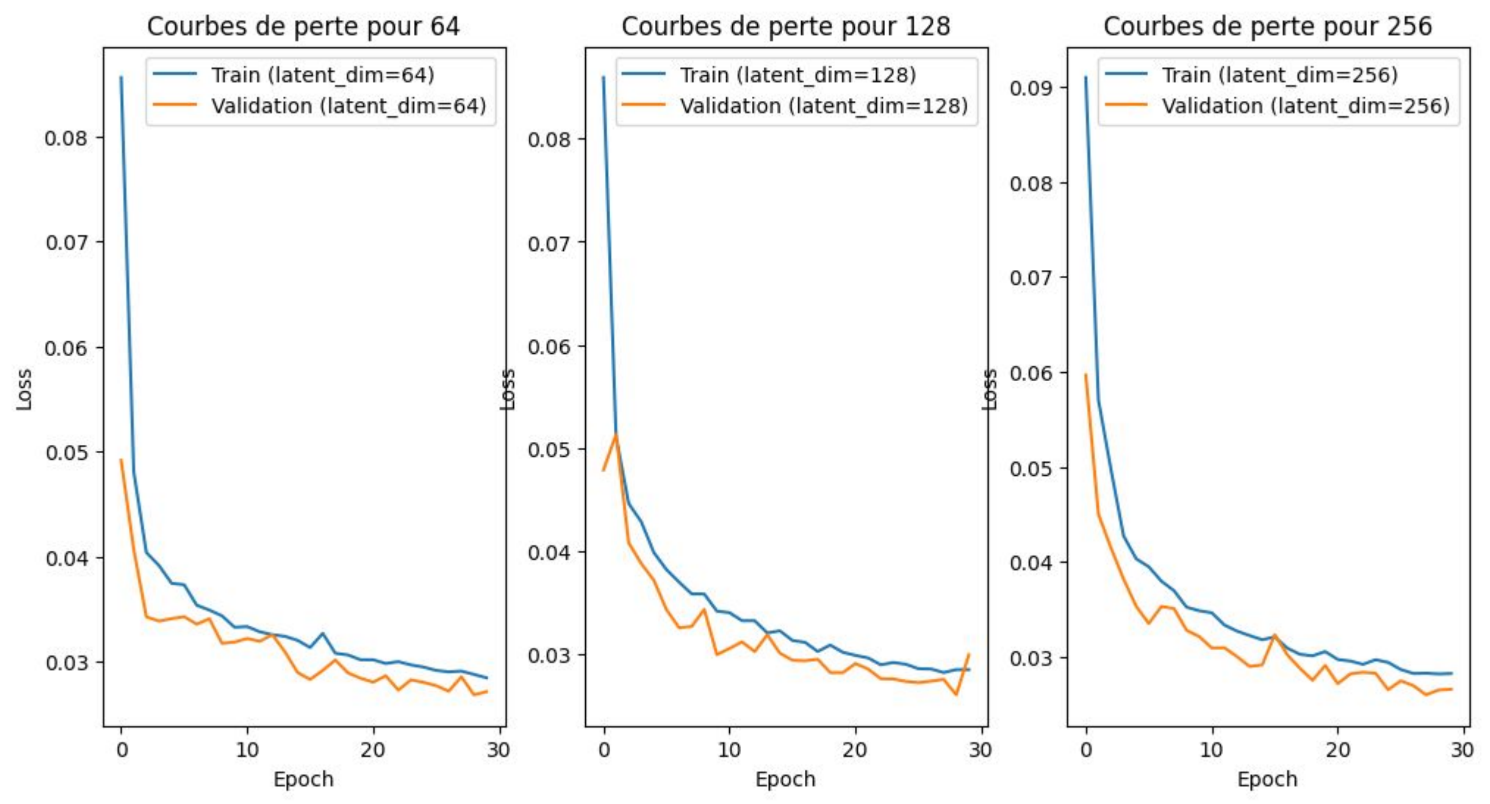
Le but du débruiteur



La fonction loss



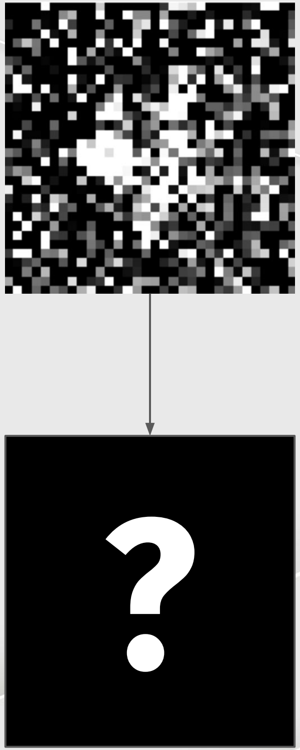
Les courbes de pertes



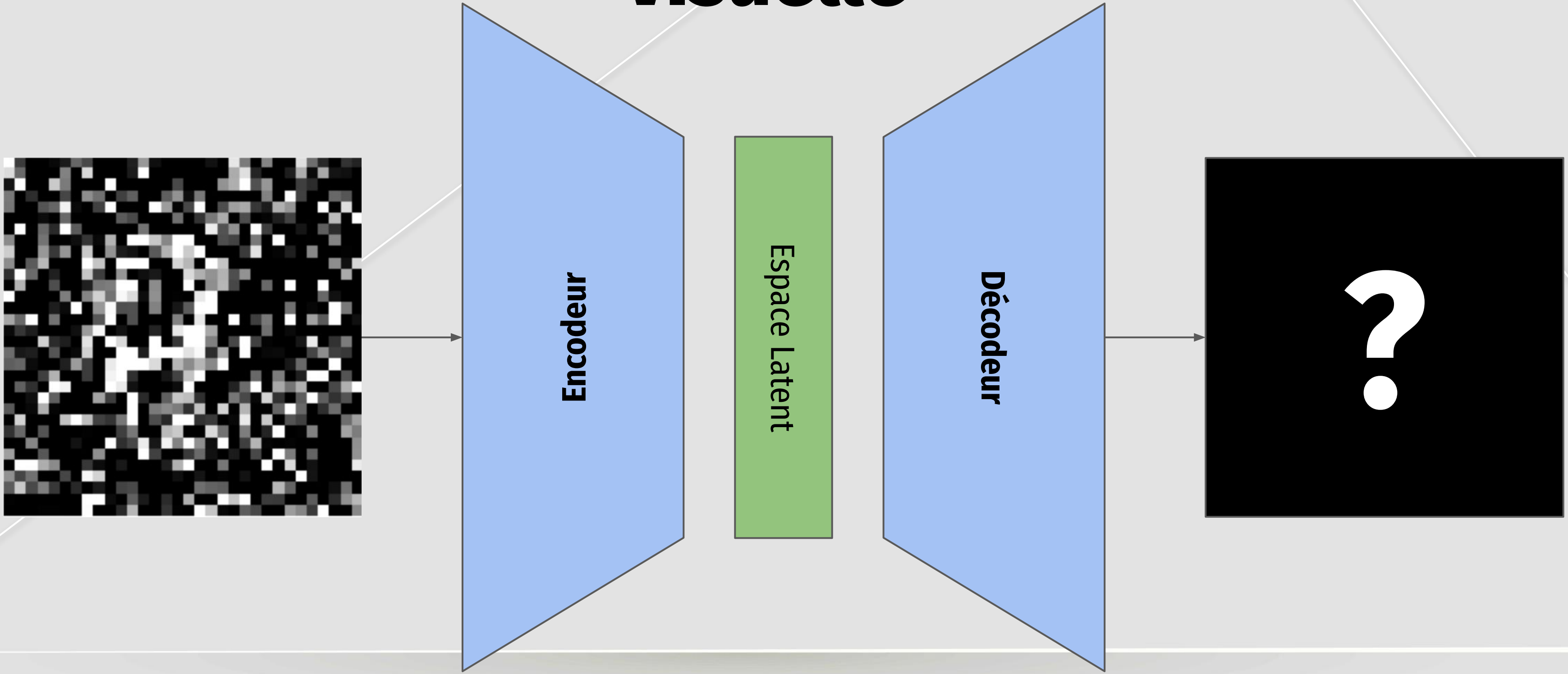
03

Les résultats

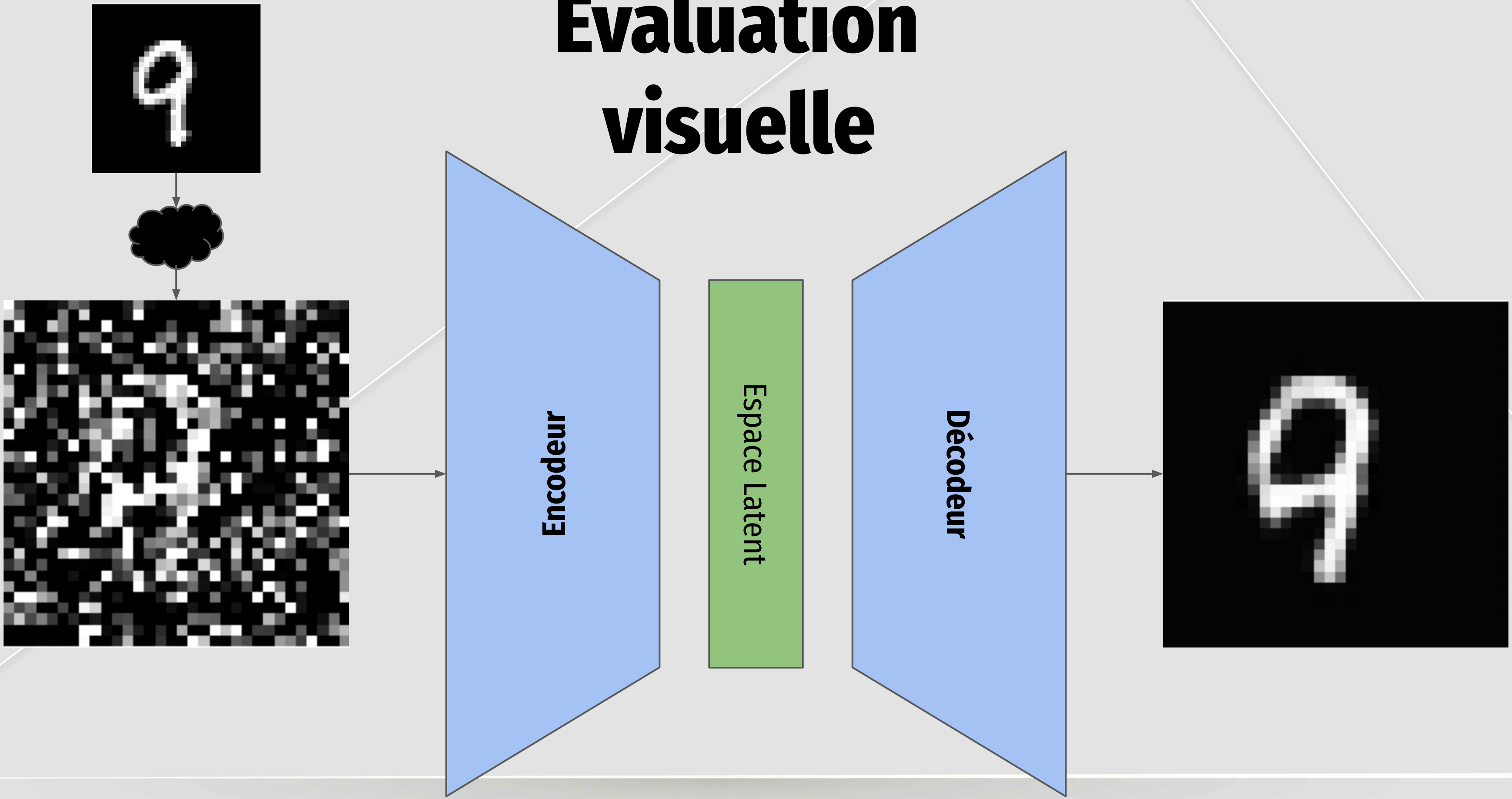
Observation des performances

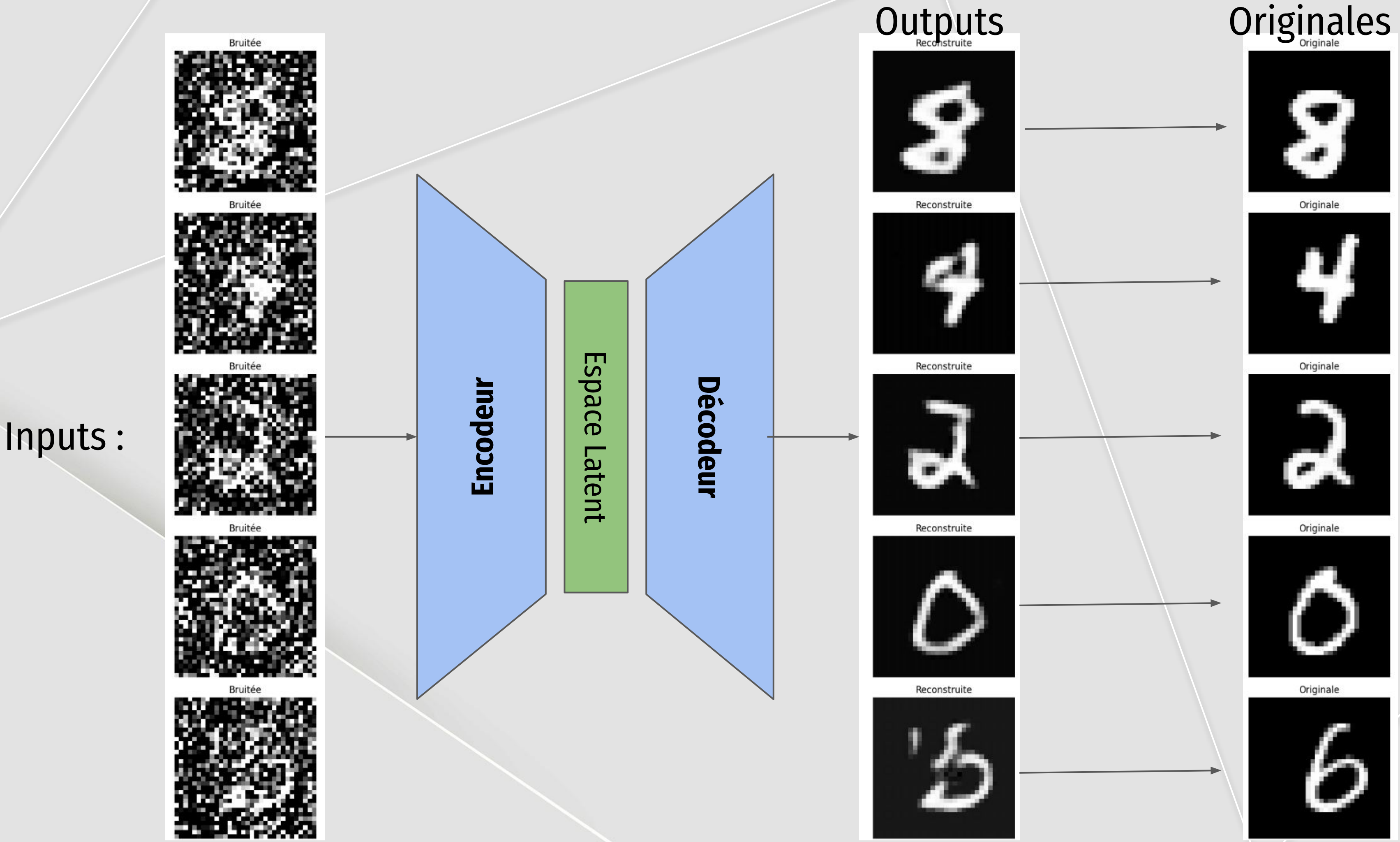


Évaluation visuelle



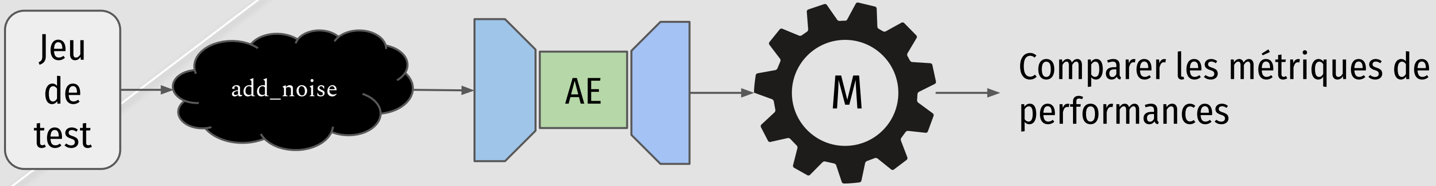
Évaluation visuelle




Outputs
Originales
Inputs :
Bruitée
Bruitée
Bruitée
Bruitée
Bruitée
Encodeur
Espace Latent
Décodeur
Reconstruite
Reconstruite
Reconstruite
Reconstruite
Reconstruite
Originale
Originale
Originale
Originale
Originale

Evaluation du modèle ?

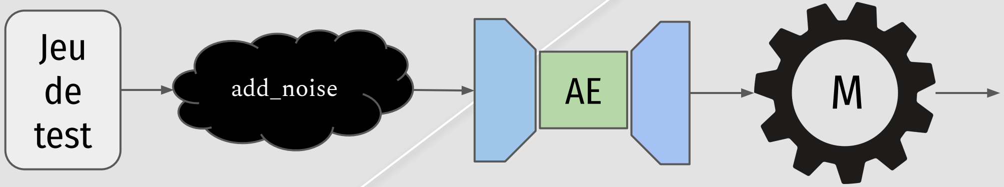
1. On entraine un modèle M sur MNIST
2. On évalue ses performances sur le jeu de test.
3. On entraîne notre auto-encodeur AE sur le jeu d'entraînement bruité.



Bruité le jeu de test

Classifier nos images débruité avec notre modèle M

Evaluation du modèle ?



Comparer les métriques de performances

Bruité le jeu de test

Classifier nos images débruité avec notre modèle M

Performance de M sur le jeu de test.

Rapport de Classification sur le jeu de test:

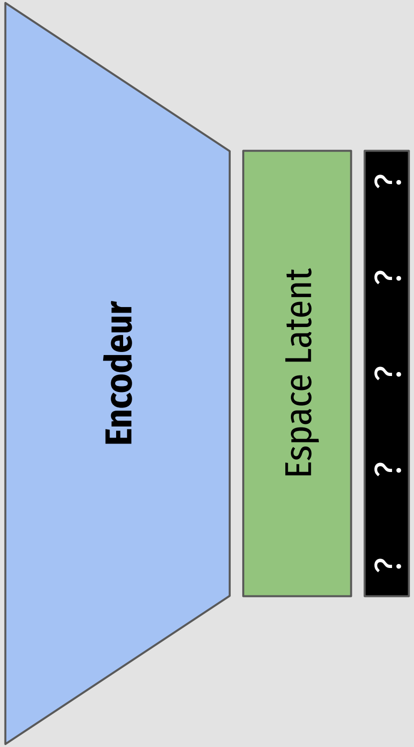
	precision	recall	f1-score	support
0	0.99	0.99	0.99	980
1	0.99	0.99	0.99	1135
2	0.99	0.99	0.99	1032
3	0.98	0.98	0.98	1010
4	0.98	0.99	0.98	982
5	0.98	0.98	0.98	892
6	0.99	0.98	0.99	958
7	0.99	0.99	0.99	1028
8	0.97	0.99	0.98	974
9	0.98	0.97	0.98	1009
accuracy			0.98	10000
macro avg	0.98	0.98	0.98	10000
weighted avg	0.99	0.98	0.99	10000

Performance de M sur le jeu de test bruité puis débruité par AE.

	precision	recall	f1-score	support
0	0.97	0.98	0.97	980
1	0.96	0.98	0.97	1135
2	0.97	0.93	0.95	1032
3	0.94	0.96	0.95	1010
4	0.95	0.93	0.94	982
5	0.95	0.93	0.94	892
6	0.97	0.96	0.96	958
7	0.88	0.96	0.92	1028
8	0.94	0.91	0.92	974
9	0.93	0.88	0.91	1009
accuracy			0.94	10000
macro avg	0.94	0.94	0.94	10000
weighted avg	0.94	0.94	0.94	10000

Notebook edito

Accuracy: 0.9434



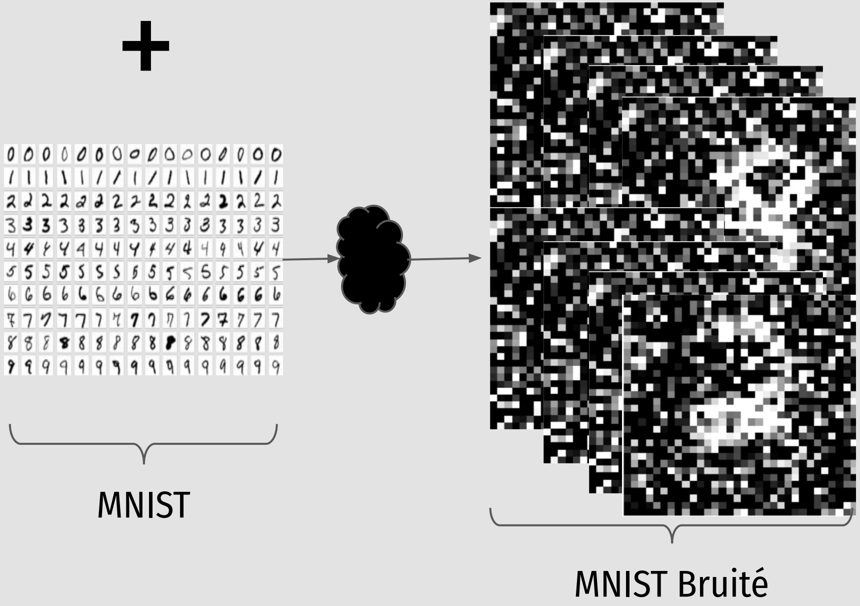
04 Analyse de l'espace latent

LABELS

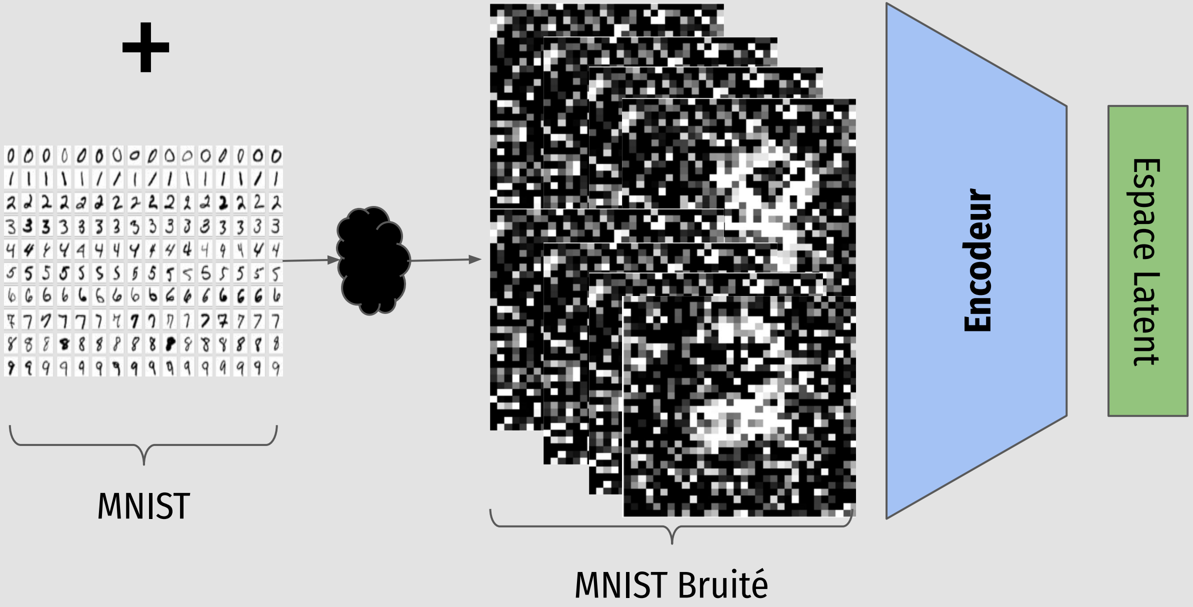
+

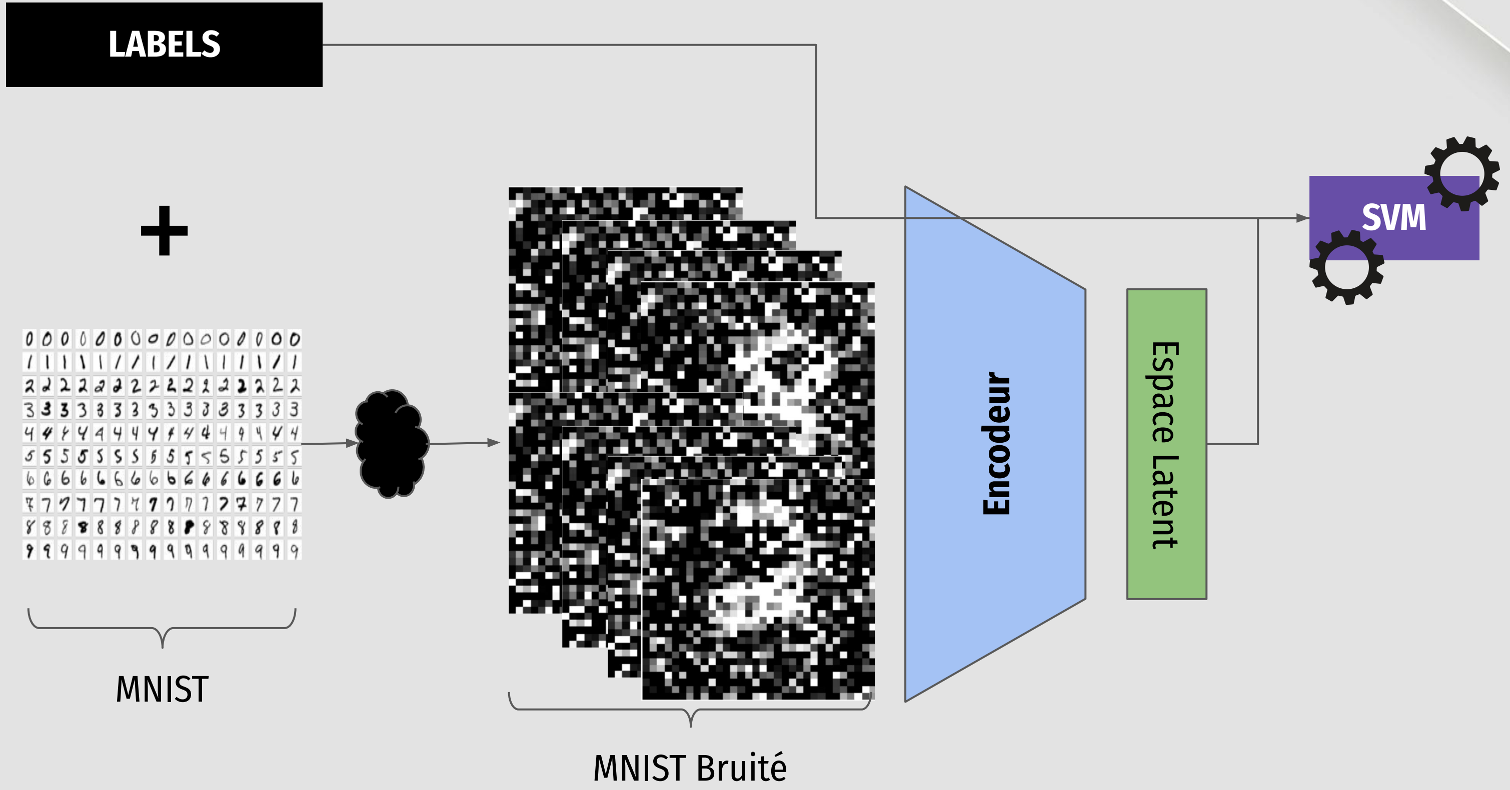
MNIST

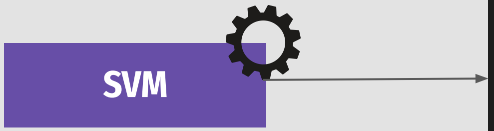
LABELS



LABELS




LABELS
+
MNIST
MNIST Bruité
Encodeur
Espace Latent
SVM



Test Results:

	precision	recall	f1-score	support
0	0.95	0.97	0.96	980
1	0.96	0.99	0.97	1135
2	0.94	0.94	0.94	1032
3	0.94	0.93	0.93	1010
4	0.92	0.93	0.93	982
5	0.94	0.93	0.93	892
6	0.93	0.96	0.95	958
7	0.95	0.93	0.94	1028
8	0.93	0.92	0.92	974
9	0.93	0.91	0.92	1009
accuracy			0.94	10000
macro avg	0.94	0.94	0.94	10000
weighted avg	0.94	0.94	0.94	10000

Test Accuracy: 93.96%

Conclusion

Résumé

1. Création d'un Auto-encodeur
2. Entrainement pour débruiter des images.
3. Evaluation de notre modèle
4. Analyse de l'espace latent

<u>Améliorations</u>

1. Intégrer des mécanismes d'attention.
2. L'utilisation de variational Autoencoders (VAE).
3. Utilisation de jeux de données plus complexes.
4. Explorer d'autre bruits.

Merci !

Avez-vous des questions ?