Rapport de projet de Deep Learning : The Constrative Learning Framework Method (SimCLR)

Samuel Metin, Tremblay Marion et Flavie Bertrand

11 janvier 2026

1 Introduction et Contexte

Le but global du projet est maximiser la performance d'un modèle avec uniquement 100 labellisées.
L'objectif de ce projet est d'implémenter et d'optimiser une méthode s'inspirant de l'article référence *"A Simple Framework for Contrastive Learning of Visual Representations"*. Nous utilisons dans cette étude la base de données MNIST (chiffres manuscrits). On sépare le jeu de données de la manière suivante :

- **Jeu d'entraînement :**
  - 49 900 images non-étiquetées.
  - 100 images étiquetées.
- **Jeu de validation (10 000 images non-étiquetées)**.
- **Jeu de test (10 000 images étiquetées)**.

Le but global est de maximiser la précision (accuracy) du modèle final sur le test, en n'utilisant que ces 100 étiquettes.

Figure 1: Aperçu des images

2 Description des méthodes

2.1 La méthode baseline

Le modèle baseline a une architecture personnalisée, mais assez classique. Nous sommes partis de celui l'encodeur utilisé lors du TP1 contenant 2 couches de convolutions, une couche de max pooling et une couche de dropout. Puis, nous avons rajouté des couches de batch normalisation après chaque couche de convolution pour obtenir le premier bloc de l'architecture. Pour améliorer les performances de notre modèle, nous avons répété le bloc une deuxième fois. Le choix des paramètres et des couches a été fait notamment sur le critère de l'accuracy obtenu avec les 100 exemples annotés (car rapide à évaluer).

```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
Net                                      [1, 10]                   --
├─Conv2d: 1-1                            [1, 32, 28, 28]           320
├─BatchNorm2d: 1-2                       [1, 32, 28, 28]           64
├─Conv2d: 1-3                            [1, 32, 28, 28]           9,248
├─BatchNorm2d: 1-4                       [1, 32, 28, 28]           64
├─MaxPool2d: 1-5                         [1, 32, 14, 14]           --
├─Dropout: 1-6                           [1, 32, 14, 14]           --
├─Conv2d: 1-7                            [1, 64, 14, 14]           18,496
├─BatchNorm2d: 1-8                       [1, 64, 14, 14]           128
├─Conv2d: 1-9                            [1, 64, 14, 14]           36,928
├─BatchNorm2d: 1-10                      [1, 64, 14, 14]           128
├─MaxPool2d: 1-11                        [1, 64, 7, 7]             --
├─Dropout: 1-12                          [1, 64, 7, 7]             --
├─Linear: 1-13                           [1, 512]                  1,606,144
├─BatchNorm1d: 1-14                      [1, 512]                  1,024
├─Dropout: 1-15                          [1, 512]                  --
├─Linear: 1-16                           [1, 10]                   5,130
==========================================================================================
Total params: 1,677,674
Trainable params: 1,677,674
Non-trainable params: 0
Total mult-adds (Units.MEGABYTES): 19.98
==========================================================================================
Input size (MB): 0.00
Forward/backward pass size (MB): 1.21
Params size (MB): 6.71
Estimated Total Size (MB): 7.93
```

Figure 2: Architecture de la baseline

2.2 La méthode SimCLR

La méthode de l'article est SimCLR, son but est de regrouper les représentations d'images similaires et d'éloigner celles d'images différentes.
Le SimCLR se décompose en 4 étapes :

- **L'augmentation des données :** Pour chaque image, on applique deux combinaisons de transformations aléatoires pour générer deux vues corrélées x_i et x_j. Dans notre cas on a utilisé la combinaison d'une rotation (jusqu'à 45°), d'un *crop and resize* et d'un *gaussian blur*. En effet, dans l'article la distorsion des couleurs est dite très impactante sur l'expérience des auteurs mais notre dataset MNIST est en noir et blanc, donc apprendre une invariance aux couleurs n'est pas adapté ici. De même le flip horizontal ou vertical peut être dangereux, notamment avec les chiffres 9 et 6, d'où notre choix de ne pas les ajouter aux transformations et de garder des rotations légères.

Figure 3: Aperçu des images obtenues après transformations

- **L'encodage :** Par un réseau de neurones (ResNet dans l'article et l'architecture décrite précédemment dans notre cas) qui extrait les vecteurs de caractéristiques de ces vues.
- **La projection (Projection head) :** Via un petit réseau de neurones composé d'un Perceptron Multicouche (MLP) avec une couche cachée et une activation ReLU : $z_i = W^{(2)}\sigma(W^{(1)}h_i)$.
- **L'évaluation d'une contrastive loss function :** c'est une fonction définie pour une tâche de prédiction contrastive. Elle calcule la similarité cosinus entre les vecteurs et pénalise le modèle s'il ne parvient pas à reconnaître les deux vues d'une même image. Pour un batch de N images elle s'applique donc sur 2N images augmentées par les différentes augmentations. Ici la taille du batch est donc un paramètre important car en augmentant il disperse les paires positives parmi davantage d'autres images.

$$\ell_{i,j} = -\log \frac{\exp\left(\text{sim}(z_i, z_j)/\tau\right)}{\sum_{k=1}^{2N} \mathbf{1}_{[k\neq i]} \exp\left(\text{sim}(z_i, z_k)/\tau\right)} \quad \text{avec } \tau \text{ un paramètre de température.}$$

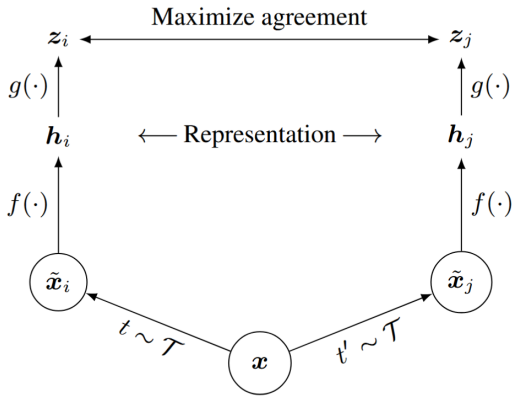


Figure 4: Explication du SimCLR

2.3 Entraînement et évaluation

Nous évaluons l'apport du pré-entraînement SimCLR sur la capacité d'un modèle à se généraliser avec peu de données étiquetées. Nous comparons alors deux pipeline d'entraînement.

Baseline

- Initialisation E_1 : on part de l'architecture personalisée intitialisée avec des poids aléatoires.
- Entraînement : Le modèle est entraîné sur un jeu de données de 100 images labellisées.
- Résultat E_1^* : On obtient un modèle de base dont les performances serviront de références.

Méthode SimCLR

- Pré-entraînement E_2 : On entraîne seulement la partie encodeur du modèle précédent avec les poids aléatoires. Apprentissage non supervisé avec 49 900 données non-labelisées pour le train et 10000 donnée non labelisées pour la validation.
- Fine-Tuning $E_2 \rightarrow E_2^*$: On reprend l'encodeur obtenu après le pré-entraînement et on entraîne un classifieur linéaire sur le jeu de 100 données labellisées encodées par le modèle précédent. Dans un cas on fine-tune seulement le classifieur, dans l'autre on fine-tune aussi les poids de l'encodeur.
- Résultat E_2^* : On obtient un modèle final.

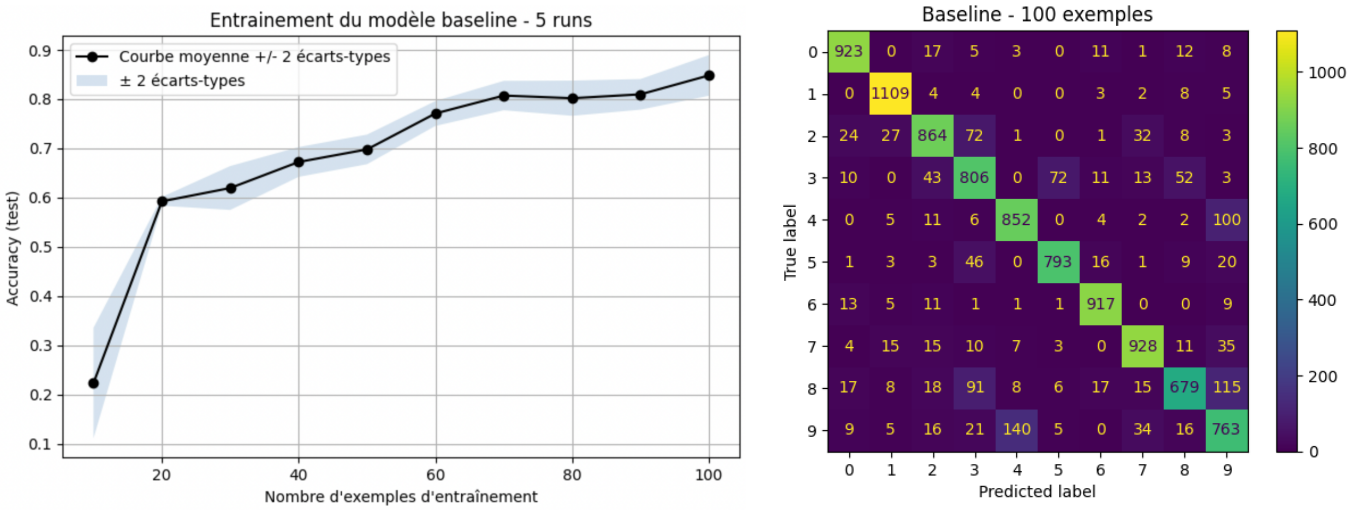
Évaluation

Nous avons évalué les performances des modèles selon plusieurs métriques :

- Accuracy sur le test set : Métrique principale permettant de mesurer la capacité de généralisation du modèle sur les 10 000 images de test.
- Courbes d'apprentissage : Suivi de l'évolution de la loss pendant l'entraînement pour valider la convergence.
- Matrices de confusion : Analyse des erreurs de classification pour identifier les classes problématiques.

3 Résultats

3.1 Résultats Baseline



(a) Accuracy test

(b) Matrice de confusion

Figure 5: Résultats de la baseline

La Figure 5 présente les résultats du modèle baseline entraîné directement sur 100 exemples labellisés sans pré-entraînement. D'après la Figure 5a, l'accuracy test atteint un plateau autour de 80-85% après environ 60 exemples d'entraînement. On observe une convergence relativement rapide dès les 20 premiers exemples, où l'accuracy passe de 20% à environ 60%. La zone bleue représentant ±2 écarts-types montre une variance importante, témoignant d'une instabilité dans l'apprentissage liée au nombre limité de données. D'après la Figure 5b, l'analyse révèle plusieurs patterns intéressants. Les chiffres 0, 1, 6 et 7 sont très bien reconnus avec respectivement 923, 1109, 917 et 928 prédictions correctes. En revanche, certaines confusions systématiques apparaissent : le chiffre 4 est souvent confondu avec le 9 (100 erreurs), le chiffre 8 présente des confusions avec le 3 (91 erreurs) et le 9 (115 erreurs), le chiffre 9 est parfois prédit comme 4 (140 erreurs).

3.2 Résultats SimCLR

Avant d'aborder les résultats du fine-tuning, il est important d'analyser la phase de pré-entraînement SimCLR.

La Figure 6 présente l'évolution de la fonction de perte au cours du pré-entraînement. Elle permet d'indiquer la qualité de l'apprentissage des représentations par l'encodeur.

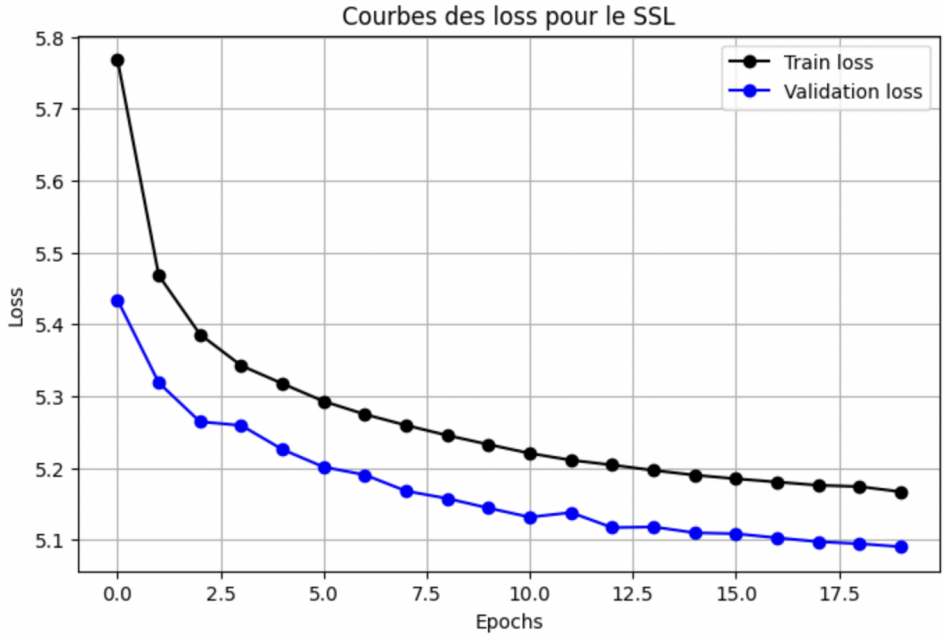
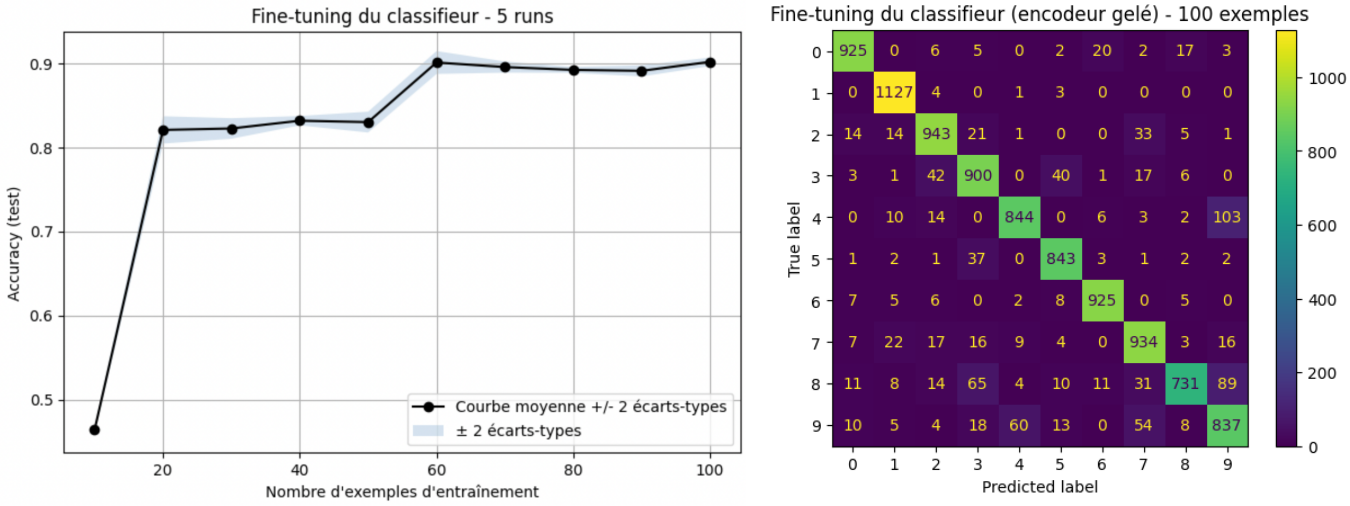


Figure 6: Contrôle de l'apprentissage non-supervisé

Les deux courbes (train loss en noir et validation loss en bleu) montrent une décroissance rapide et régulière de la loss, partant d'environ 5.78 à l'epoch 0 pour atteindre un plateau autour de 5.17-5.18 après 20 epochs. Cette convergence rapide indique que le modèle apprend efficacement à rapprocher les représentations des vues augmentées d'une même image (paires positives) tout en éloignant celles d'images différentes (paires négatives). Un point notable est le faible écart entre la loss d'entraînement et la loss de validation tout au long du pré-entraînement. Les deux courbes restent très proches et parallèles, avec un écart maximal de seulement 0.05-0.07. Cela montre l'absence de sur-apprentissage.

3.2.1 Linear probing (encodeur gelé)

Cette approche consiste à geler l'encodeur pré-entraîné et à n'entraîner que la couche de classification finale.



(a) Accuracy test

(b) Matrice de confusion

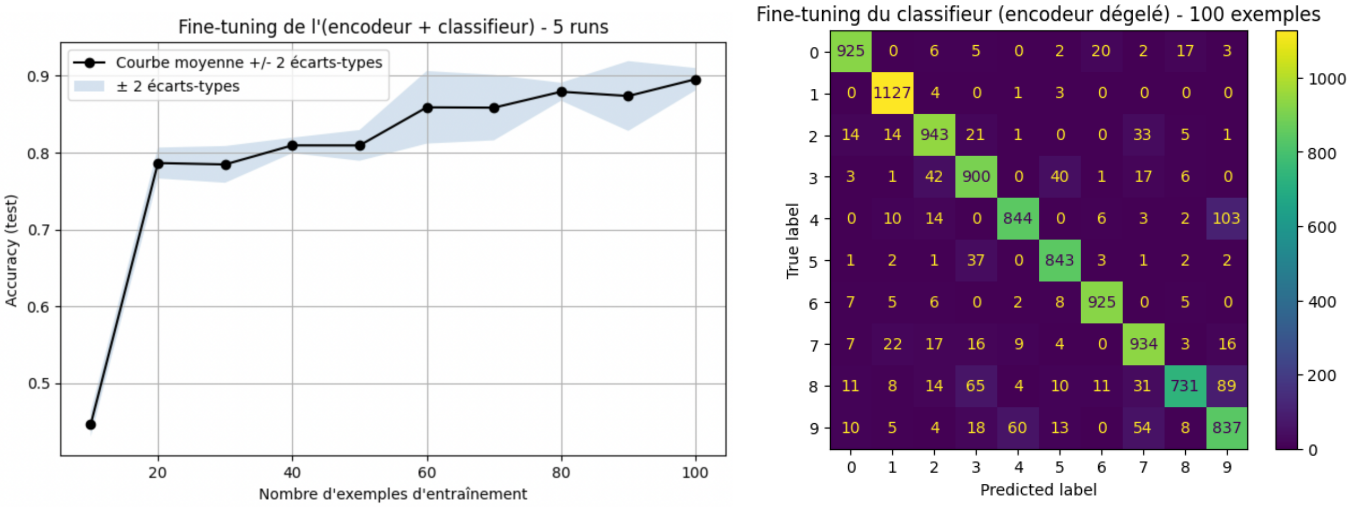
Figure 7: Résultats après fine-tuning du classifieur

La courbe d'apprentissage (Figure 7a) montre une amélioration spectaculaire par rapport à la baseline, avec une accuracy finale atteignant environ 91% sur le test set. La convergence est beaucoup plus rapide

: dès 20 exemples, le modèle atteint déjà 82% d'accuracy. La variance (zone bleue) est nettement réduite par rapport à la baseline, indiquant une plus grande stabilité de l'apprentissage. La matrice de confusion (Figure 7b) révèle des améliorations significatives sur les classes problématiques de la baseline. Le chiffre 5, qui était parfois confondu dans la baseline, montre maintenant 843 prédictions correctes contre 793 dans la baseline. Les confusions 8-3, 8-5, 8-9 diminuent également drastiquement à 65, 14, 89 cas respectivement. Le chiffre 9 atteint 837 prédictions correctes avec des confusions 9-4 réduites à 60 cas.

3.2.2 Fine-tuning complet (encodeur non gelé)

Cette approche consiste à entraîner à la fois l'encodeur et le classifieur en même temps.



(a) Accuracy test

(b) Matrice de confusion

Figure 8: Résultats après fine-tuning de l'encodeur et classifieur

La Figure 8a montre une accuracy finale d'environ 90%, légèrement inférieure aux 91% obtenus avec l'encodeur gelé (Figure 7a). Cette légère baisse de performance suggère que sur MNIST avec seulement 100 exemples, le fine-tuning complet peut introduire un léger surapprentissage par rapport au gel de l'encodeur. La matrice de confusion (Figure 8b) montre des performances comparables à la Figure 7b.

4 Conclusion

Ce projet a permis de valider l'efficacité de la méthode SimCLR dans un contexte d'apprentissage avec peu de données labellisées. Les résultats obtenus montrent qu'un pré-entraînement contrastif sur 49 900 images non-étiquetées permet d'améliorer significativement les performances d'un modèle n'ayant accès qu'à 100 exemples labellisés pour la classification. Le pré-entraînement SimCLR améliore l'accuracy de 85% à 91%, soit un gain de 6 points de pourcentage par rapport à la baseline. Cette amélioration s'accompagne d'une meilleure stabilité de l'entraînement. Bien que les résultats soient encourageants, plusieurs pistes d'amélioration peuvent être explorées, tels que l'utilisation d'une architecture de l'encodeur plus profonde, l'étude de l'influence du nombre d'images labellisées, ou encore d'autres augmentations des images.