Rapport TP2

1. Fiche TP2 : Training an MLP (Keras)

1.1 Load and Preprocess the MNIST Dataset

- On a 60 000 images pour l'entraînement et 10 000 pour le test.
- **Prétraitement des données** :
  - **Normalisation** : On divise les pixels (valeurs entre 0 et 255) par 255 pour obtenir des valeurs entre **0 et 1**.
  - Cela aide le modèle à converger plus rapidement en évite des mises à jour trop grandes des poids.

1.2 Split the Train Set into Train and Validation Sets

- On divise l'ensemble d'entraînement en **80% pour l'entraînement** et **20% pour la validation**.
- La validation permet de contrôler overfitting et d'ajuster les hyperparamètres.

1.3 Design an MLP Architecture

L'architecture demandée est la suivante :

1. **Couche d'entrée** : 28×28=784 neurones (car chaque image est convertie en un vecteur de 784 valeurs).
2. **Deux couches cachées** :
   - 1ère couche cachée : 128 neurones, activation ReLU.
   - 2ème couche cachée : 64 neurones, activation ReLU.
3. **Couche de sortie** :
   - 10 neurones (correspondant aux chiffres de 0 à 9).
   - Activation Softmax (convertit les scores en probabilités).

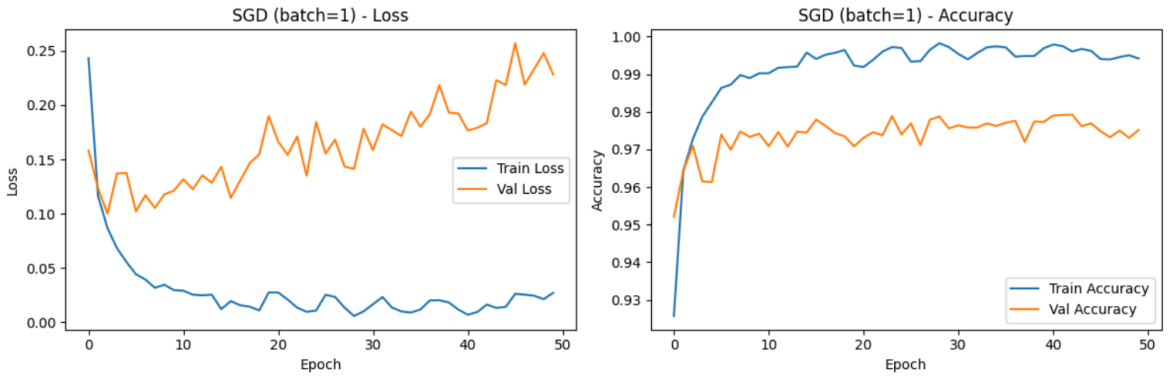
1.4 Train the Model with Different SGD Variants

On entraîne le modèle avec plusieurs variantes de **SGD** :

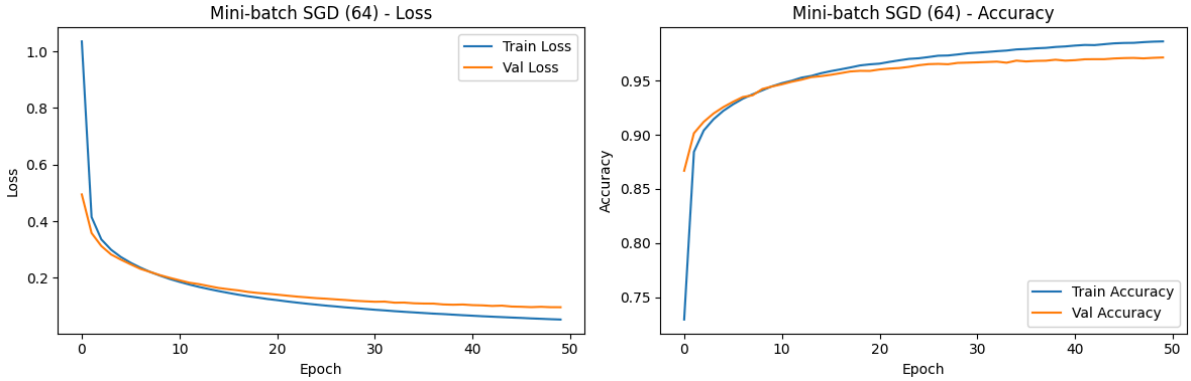
- **SGD standard** : Mise à jour des weights après **chaque sample**.
- **Mini-batch SGD** : Mise à jour après un **lot (batch) de 64 sample**.
- **Batch SGD** : Mise à jour après avoir parcouru **toutes les données**.

— On entraîne pendant **50 époques** avec un **taux d'apprentissage (learning rate) de 0.01**.

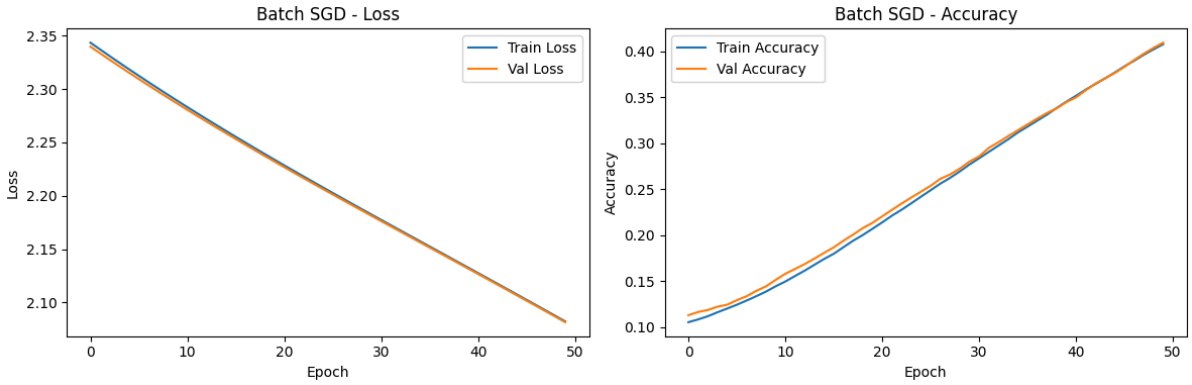
- **Learning Curves** :



SGD



Mini Batch SGD (size =64)



Batch SGD

Strategy	SGD	Mini batch SGD	Batch SGD
Training Time	5276.93s (1h46min)	114.40s	20.84s

- **Observations & Explications:**
  - **Batch SGD** : La perte diminue lentement et de manière linéaire, et la précision augmente progressivement mais à un rythme très faible. Comme la mise à jour des poids ne se fait qu'une seule fois par époque, l'entraînement est stable mais inefficace.
  - **Mini-Batch SGD** : Trouve un compromis entre stabilité et rapidité. La perte diminue plus rapidement que Batch SGD, et la précision atteint plus de 95% rapidement.
  - **SGD** : Très instable à cause des mises à jour des poids après chaque sample. La précision en entraînement progresse vite mais fluctue, tandis que la précision en validation est plus basse, elle est trop lente et trop bruitée pour de grands ensembles de données.

  Le **Mini-Batch SGD** est le **meilleur compromis** entre **vitesse, stabilité et précision**
- **Conclusion:**

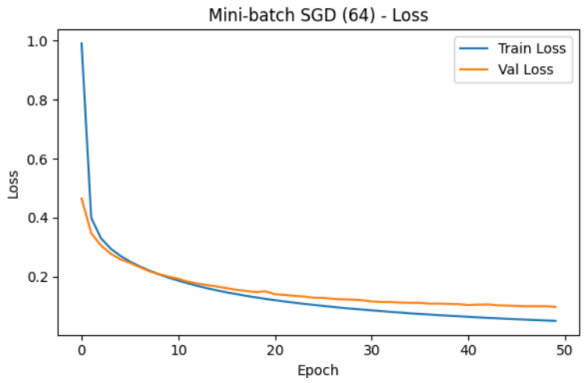
  Le Mini-Batch SGD est le meilleur compromis entre vitesse, stabilité et précision.

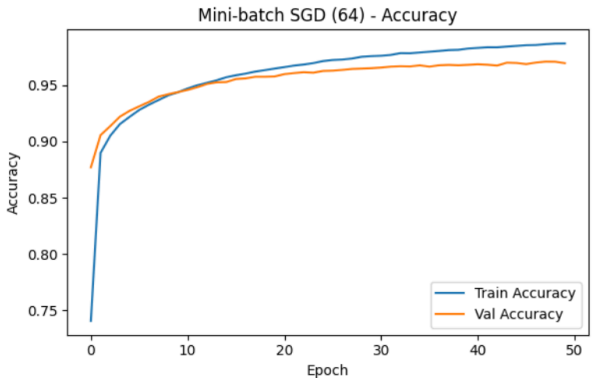
1.5 Compare SGD Variants

- On a:

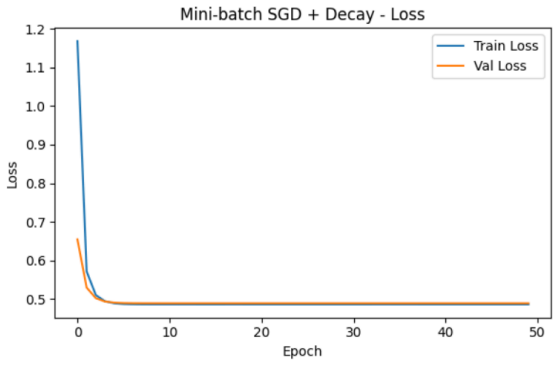
Strategy	Time (s)	Val Loss	Val Accuracy	Precision	Recall	F1 Score
Mini-batch SGD	114.4012	0.0980	0.9695	0.9694	0.9692	0.9693
Mini-batch SGD + Decay	133.2793	0.4886	0.8702	0.8689	0.8686	0.8683
SGD + Decay + Momentum	5122.7701	0.2031	0.9417	0.8970	0.8969	0.8967

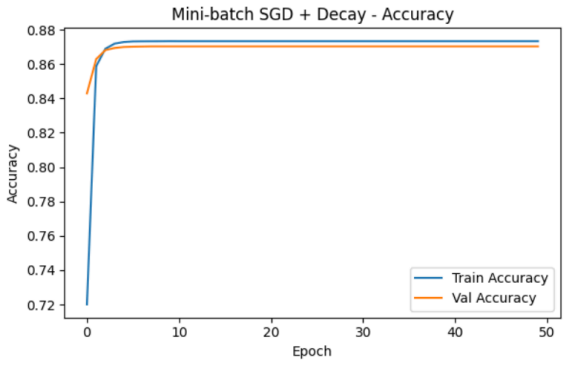
- **Comparaison:**
  - Mini batch :



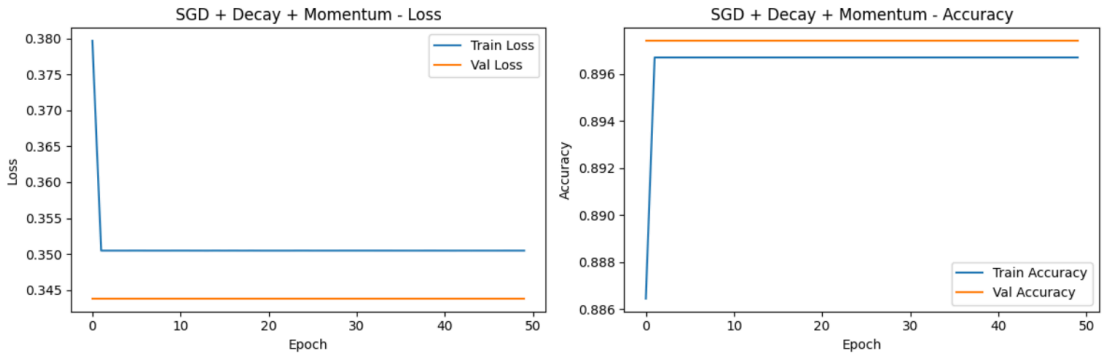


  - Mini batch SGD with decay :





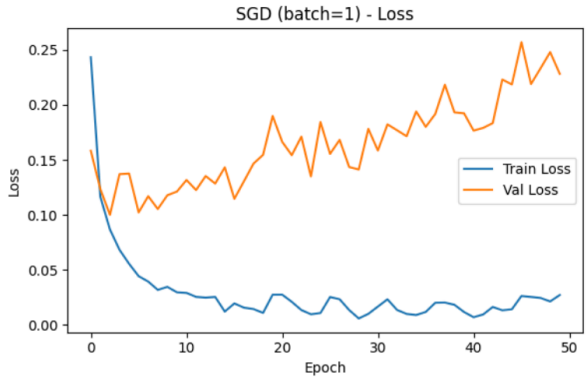
  - SGD with decay and momentum:

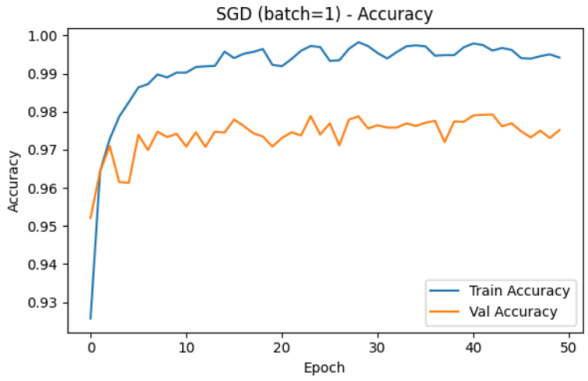


- **Observations :**
  - L'optimisation avec **SGD + Decay + Momentum** présente une convergence rapide, mais la perte semble se stabiliser tôt sans amélioration significative.
  - **Mini-batch SGD + Decay + Momentum** obtient une meilleure convergence avec une réduction plus progressive de la perte et une meilleure précision finale.
  - **Mini-batch SGD (64)** continue d'améliorer la précision de manière plus stable sur plusieurs époques, bien que la perte diminue plus lentement.
- **Explications :**
  - **Momentum** accélère la convergence en conservant une direction stable vers le minimum.
  - **Learning rate decay** ajuste progressivement le taux d'apprentissage pour éviter les oscillations et garantir une meilleure stabilité.
  - **Mini-batch SGD** favorise une mise à jour plus efficace des poids avec un bon équilibre entre vitesse et précision.

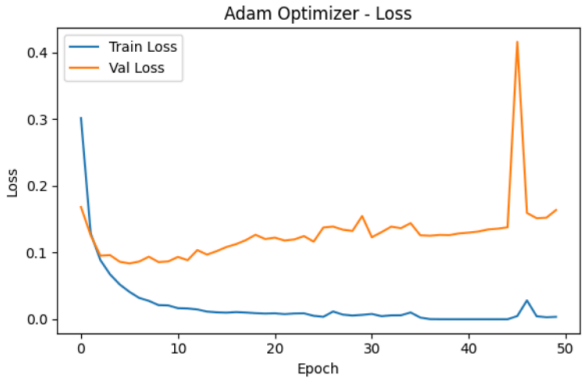
1.6 Compare between SGD optimizers

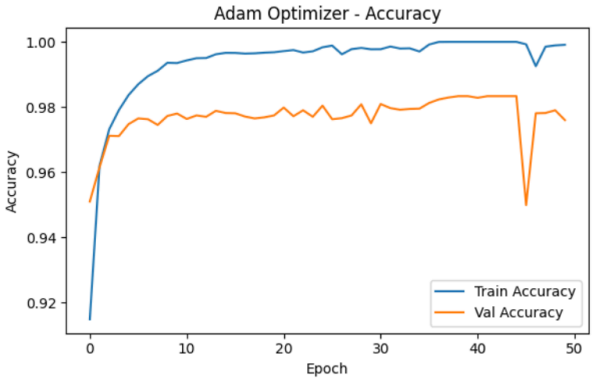
- SGD:



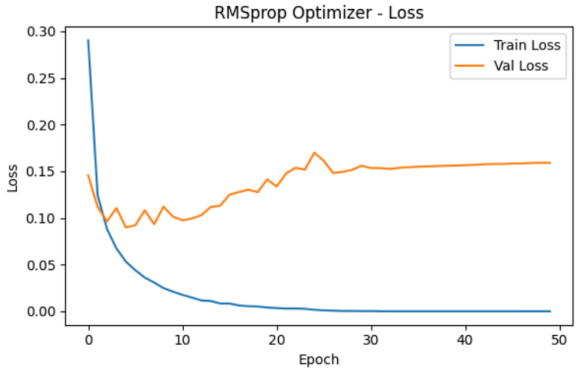


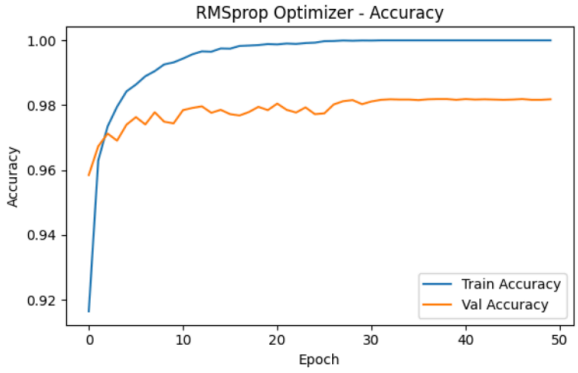
- Adam:



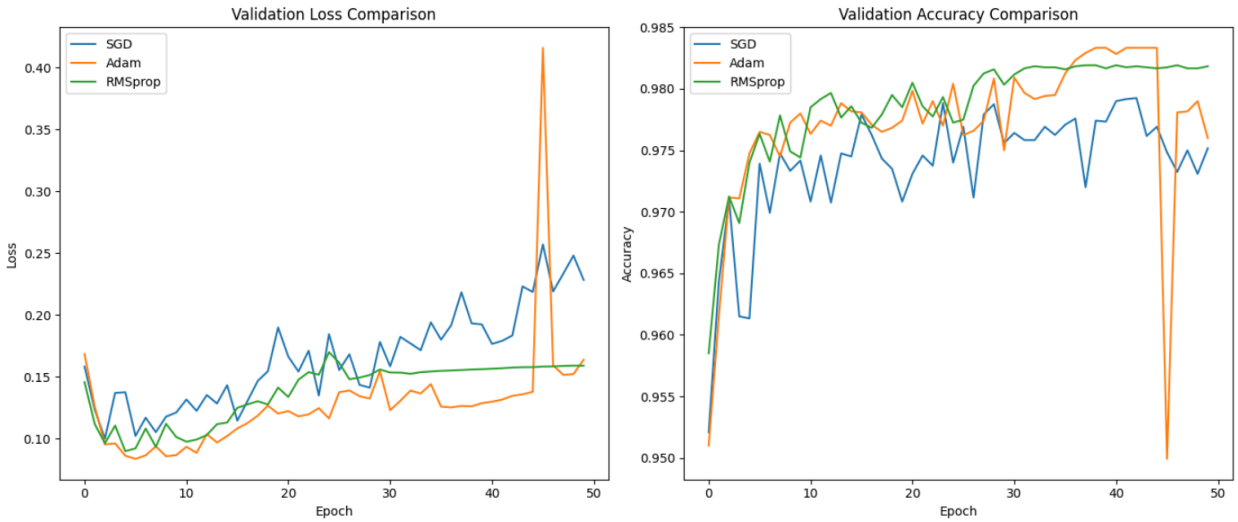


- RmsProp:





Comparaison:



- On a:

Optimizer	Training Time (s)	Val Loss	Val Accuracy	Precision	Recall	F1 Score
SGD	5276.9311	0.2283	0.9752	0.9750	0.9750	0.9750
Adam	142.3271	0.1638	0.9760	0.9762	0.9756	0.9759
RMSprop	154.8925	0.1590	0.9818	0.9817	0.9817	0.9817

Observation :

- **Perte de validation** :
  - **SGD** montre une tendance généralement croissante avec de fortes fluctuations, indiquant une instabilité dans l'entraînement.
  - **Adam** commence avec une perte plus faible mais présente un pic soudain vers la fin, ce qui peut être dû à un problème de taux d'apprentissage ou de surapprentissage.
  - **RMSprop** maintient une courbe de perte plus stable et plus basse, suggérant une meilleure convergence.
- **Précision de validation** :

- **Tous les optimiseurs** montrent une amélioration de la précision au fil des époques.
- **RMSprop et Adam** atteignent la meilleure précision, avec **RMSprop** étant légèrement plus stable.
- **SGD** est moins performant et moins stable.
- **Adam a une chute brutale vers la fin**, ce qui pourrait être dû au surapprentissage, à un taux d'apprentissage élevé ou à des gradients qui disparaissent.

Explication :

- **L'instabilité de SGD** peut être due à l'absence de taux d'apprentissage adaptatifs, le rendant plus sensible au bruit.
- **Adam et RMSprop** utilisent des taux d'apprentissage adaptatifs, leur permettant de converger plus rapidement et de manière plus stable.
- **Le pic final d'Adam** suggère un problème avec la diminution du taux d'apprentissage, nécessitant un meilleur réglage des hyperparamètres.
- **RMSprop semble être le meilleur optimiseur** pour ce modèle, équilibrant à la fois une perte faible et une précision élevée.

1.7 Choosing the best model

- On a:

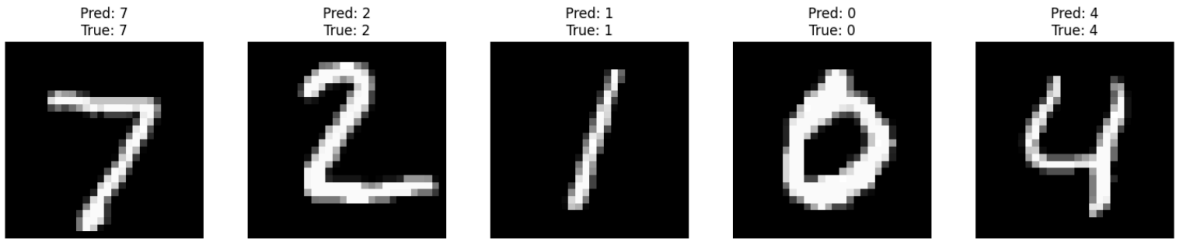
Model	Val Loss	Val Accuracy	Precision	Recall	F1 Score	Train Loss	Train Accuracy
RMSprop	0.159000	0.981800	0.981700	0.981700	0.981700	0.000000	1.000000
Adam	0.163800	0.976000	0.976200	0.975600	0.975900	0.003600	0.999100
SGD (batch=1)	0.228300	0.975200	0.975000	0.975000	0.975000	0.027000	0.994200
Mini-batch SGD (64)	0.098000	0.969500	0.969400	0.969200	0.969300	0.050300	0.986900
Mini-batch SGD+Decay	0.488600	0.870200	0.868900	0.868600	0.868300	0.485600	0.873300
Batch SGD	2.128400	0.337500	0.389500	0.336600	0.293800	2.131300	0.331900
SGD+Decay+Momentum	2.360700	0.065000	0.196000	0.064700	0.043800	0.343600	0.894500

1. **Précision en Validation (Val Accuracy) :**
    - **RMSprop** atteint **98,18 %**, le score le plus élevé parmi tous les modèles évalués.
    - Comparaison : Adam (97,60 %), SGD (batch=1) (97,52 %).
2. **Perte en Validation (Val Loss) :**
    - **RMSprop** présente la perte la plus faible (**0,159**), signe d'une excellente généralisation.
3. **Métriques Complémentaires :**
    - **Précision**, **Rappel** et **F1-Score** sont tous autour de **98,17 %**, confirmant une performance équilibrée et cohérente.

Conclusion :

RMSprop se distingue comme le modèle le plus performant sur les données de validation, avec des métriques robustes et cohérentes. Une validation supplémentaire sur un jeu de test indépendant est recommandée pour confirmer sa généralisation, mais il représente actuellement la meilleure option selon les critères évalués.

- Pour illustrer davantage l'efficacité du modèle, visualisons un exemple de prédiction en utilisant le modèle Rmsprop:



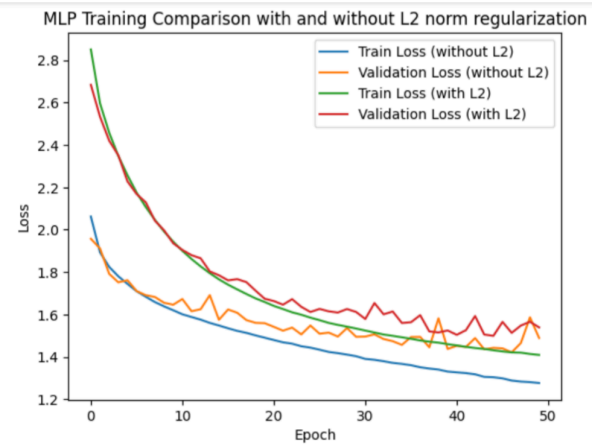
2. Fiche TP2_suite : Optimizing hyperparameters

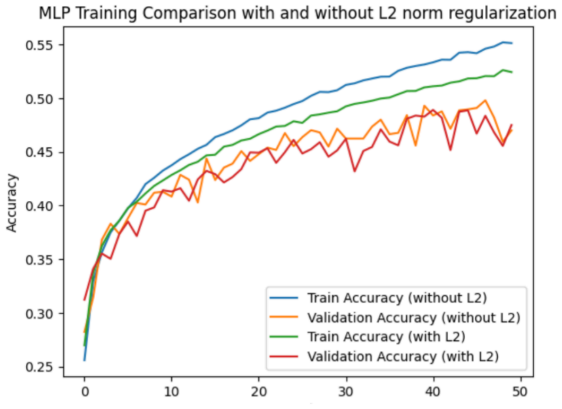
2.1 Train the designed architecture with Mini-batch SGD

Nous avons entraine l'architecture concue en utilisant SGD en mini-batch avec une taille de lot de 128 et un taux d'apprentissage de 0.01 pendant 50 epoques.

2.2 Add L2 norm regularization

Nous avons ajoute une regularisation L2 a la deuxieme couche entierement connectee et avons compare les resultats obtenus entre l'architecture avec et sans L2 norm.





Architecture	Loss on Validation Data	Accuracy on Validation Data
Without L2 norm	1.4847	46,72 %
With L2 norm	1.5449	47,42 %

Observations :

- En comparant les modèles avec et sans régularisation L2, on observe une **légère augmentation de la perte** sur les données de validation, passant de **1.4847 à 1.5449**, ainsi qu'une **légère amélioration de la précision**, qui passe de **46,72 % à 47,42 %**.

Explication :

En examinant l'impact de la régularisation L2 sur les performances du modèle, nous pouvons tirer les conclusions suivantes :

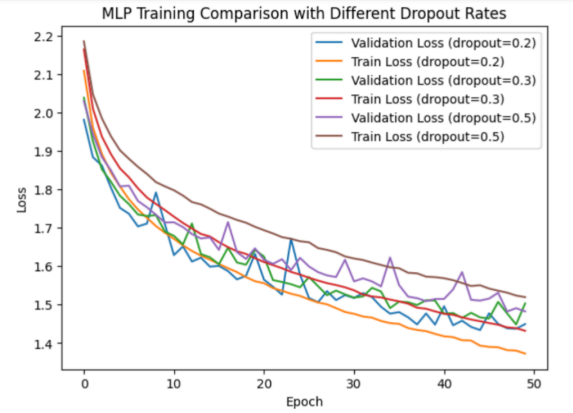
1. Sans régularisation L2 :

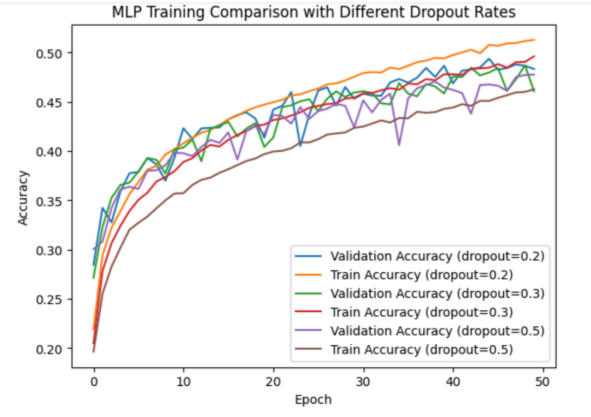
- Le modèle **sans régularisation L2** présente une perte et une précision légèrement **meilleures sur les données de validation** par rapport à la version régularisée.
- Cela suggère que **sans régularisation**, le modèle peut se **concentrer excessivement sur les détails spécifiques** des données d'entraînement, ce qui peut **nuire à sa performance sur de nouvelles données**.
- Il est possible que le modèle **mémorise du bruit ou des caractéristiques non pertinentes** des données d'entraînement, ce qui réduit son efficacité sur de nouvelles données et entraîne des prédictions moins fiables.

2. Avec régularisation L2 :

- L'ajout de la **régularisation L2** entraîne une **légère augmentation de la perte** sur les données de validation, mais aussi une **petite amélioration de la précision**.
- Cela signifie que la régularisation **empêche le modèle de trop se focaliser sur les détails spécifiques des données d'entraînement**, favorisant une **meilleure généralisation** sur des données non vues auparavant.
- L'**augmentation de la perte** indique que les prédictions du modèle sur l'ensemble de validation sont **moins influencées par le bruit ou des caractéristiques non pertinentes** des données d'entraînement. Cela **améliore sa robustesse et ses performances générales**, comme en témoigne la précision légèrement supérieure.

2.3 Effect of Dropout Rates





Observations :

D’après le tableau :

- **Un taux de dropout de 0,2** a donné les **meilleurs résultats**, avec une perte de validation de **1,436** et une précision de **0,486**.
- **Un taux de dropout de 0,3** a entraîné une **légère baisse des performances**, avec une perte de validation de **1,471** et une précision de **0,481**.
- **Un taux de dropout de 0,5** a donné les **moins bons résultats**, avec une perte de validation de **1,512** et une précision de **0,461**.

Explication :

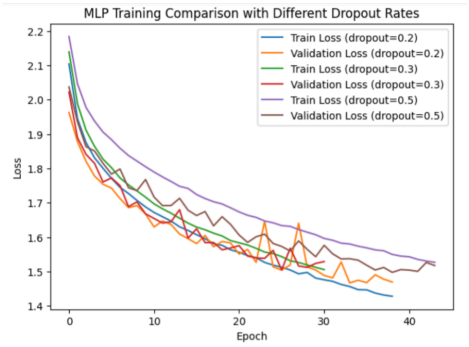
Généralisation et stabilité :

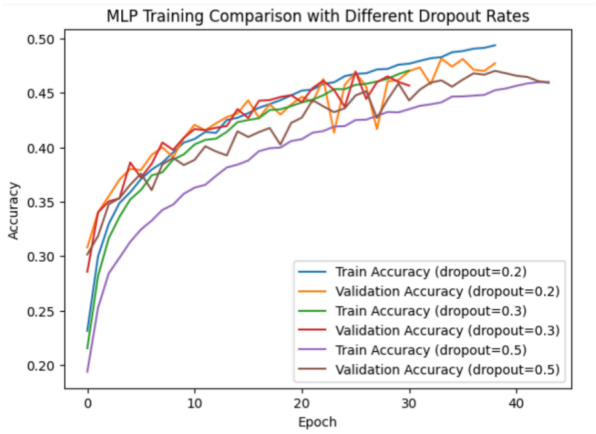
- **Un taux de dropout de 0,2** offre un **équilibre optimal** entre la **prévention du surapprentissage** et la **capacité d'apprentissage du modèle**.
- La **régularisation par dropout** consiste à **désactiver aléatoirement un pourcentage de neurones** pendant l’entraînement, empêchant ainsi le modèle de **trop dépendre d’une seule caractéristique ou d’un seul neurone**.
- Cela **encourage le réseau** à apprendre des **caractéristiques plus robustes et généralisables**, ce qui améliore ses performances sur des **données non vues**.
- Avec un taux de dropout de 0,2, la régularisation est suffisante pour éviter le surapprentissage, sans réduire excessivement la capacité du modèle à apprendre à partir des données.
- En conséquence, le modèle atteint une meilleure généralisation et stabilité, faisant de 0,2 le taux de dropout optimal pour cette tâche.

2.4 Effect of Early Stopping with Different Dropout Rates

Dropout Rate	Training Stopped at Epoch	Validation Loss
0.2	45	1.529
0.3	45	1.525

0.5	1	1.486

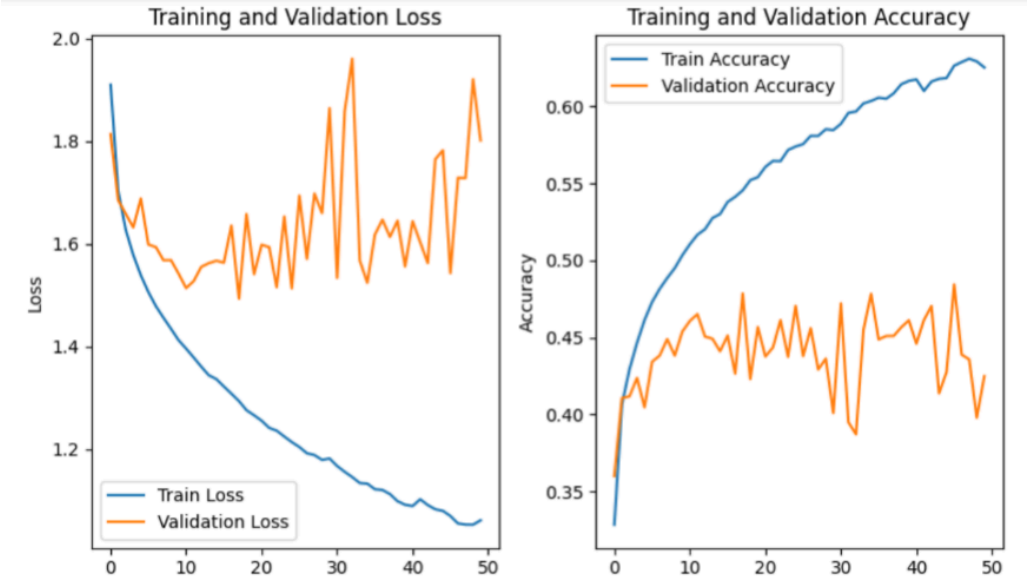




Observations & Explications :

- Les modèles avec **des taux de dropout de 0,2 et 0,3** ont arrêté l'entraînement à **l'époque 45**, indiquant qu'ils ont **convergé vers un point stable**.
- Ces taux de dropout ont fourni **une régularisation suffisante** pour **éviter le surapprentissage**, tout en permettant un **apprentissage efficace**.
- En revanche, le modèle avec un **taux de dropout de 0,5** a **arrêté l'entraînement après seulement une époque**, suggérant que ce taux était **trop élevé**.
- Un **dropout de 0,5** a supprimé **une proportion trop importante de neurones**, **réduisant ainsi considérablement la capacité d'apprentissage** du modèle.
- Par conséquent, le modèle avec **un dropout de 0,5** a montré **une mauvaise convergence**, entraînant un **arrêt prématuré de l'entraînement**.

2.5 Effect of Batch Normalization



Normalization	Validation Loss	Validation Accuracy
Before	1.4847	46.72%
After	1.5004	47.46%

Observations :

- L'introduction de la batch normalization a entraîné une légère augmentation de la perte de validation et de la précision, par rapport au modèle sans batch normalization.

Explication :

- La batch normalization ajuste l'échelle et la moyenne de chaque caractéristique, garantissant une distribution cohérente des entrées à travers les couches du réseau.
- Ce processus de normalisation stabilise l'apprentissage, permettant au modèle d'apprendre plus efficacement.
- En maintenant des gradients stables pendant l'entraînement, la batch normalization réduit le risque d'explosion ou de disparition des gradients, favorisant ainsi une meilleure convergence.

2.6 Random Search

Explication :

- **Le Random Search** est une technique d'optimisation des hyperparamètres qui **sélectionne aléatoirement** des combinaisons d'hyperparamètres à partir d'un **espace de recherche prédéfini** et évalue leur performance.
  Dans ce cas, l'espace de recherche comprenait
  trois valeurs pour le **taux d'apprentissage** (**0.001, 0.01, 0.1**) et **trois valeurs** pour le **taux de dropout** (**0.2, 0.3, 0.5**).
- L'algorithme de **Random Search** a testé différentes **combinaisons** de ces hyperparamètres en utilisant la **validation croisée**, afin de trouver celle qui offre **la meilleure précision en validation**.
- Les **meilleurs hyperparamètres** trouvés par le Random Search sont:
  - **Taux d'apprentissage** : **0.01**
  - **Taux de dropout** : **0.2**
  - **Taille de batch** : **32**
- Le modèle entraîné avec ces hyperparamètres optimaux a atteint une **précision en validation d'environ 51,65%**, ce qui montre que **ces hyperparamètres ont permis d'améliorer les performances** du modèle par rapport aux autres combinaisons testées.