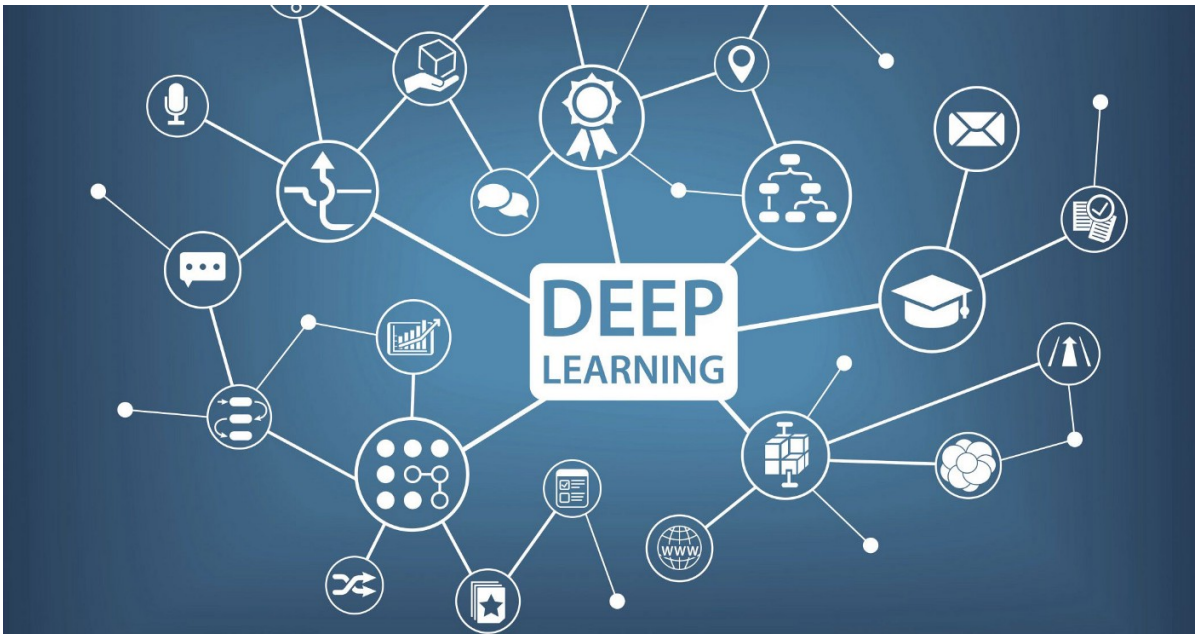
Année académique 25/26
Rapport de Travaux Pratique 1

Conception et le Déploiement de modèles de Deep Learning



Option	Génie Informatique, Niveau 5
Etudiant	NGUIFFO NGAKOU RICK VARNEL
Matricule	21P373
Enseignant responsable	Pr. Louis Fippo Fitime
Date	25/09/2025

i. Résumé

Ce TP est conçu pour les étudiants en informatique souhaitant acquérir des compétences pratiques en Deep Learning (DL) Engineering. Au-delà de la théorie, ce TP se concentre sur les aspects pratiques et d'ingénierie du cycle de vie des modèles DL, incluant le développement, le suivi, l'empaquetage, et le déploiement en environnement de production. Ce document fournit les réponses détaillées aux questions posées dans le cadre des Travaux Pratiques sur la conception et le déploiement de modèles de Deep Learning.

ii. Table des matières

Table des matières

Partie 1 : Fondations du Deep Learning (3h)

1.1. Concepts Théoriques

Rappel des modèles linéaires et de l'optimisation stochastique.

- **Différence entre la descente de gradient classique et la descente de gradient stochastique (SGD) :**
  - **Descente de gradient classique (Batch Gradient Descent) :** Calcule le gradient de la fonction de coût en utilisant *toutes* les données d'entraînement. Cela assure une descente stable mais peut être très lent et coûteux en calcul pour de grands ensembles de données.
  - **Descente de gradient stochastique (SGD) :** Calcule le gradient et met à jour les poids en utilisant *une seule instance* d'entraînement (ou un petit lot, appelé mini-batch). C'est beaucoup plus rapide et permet de traiter de très grands ensembles de données, mais les mises à jour sont plus "bruyantes".
- **Pourquoi la SGD est préférée dans le contexte du deep learning :**
  - **Efficacité Computationnelle :** Essentielle pour les modèles de deep learning avec d'énormes ensembles de données. La SGD permet des mises à jour fréquentes des poids sans traiter l'intégralité du dataset.
  - **Éviter les Minima Locaux :** La nature "bruyante" de la SGD peut aider le modèle à s'échapper des minima locaux et des points selles, fréquents dans les paysages de perte non convexes.
  - **Généralisation :** Le bruit introduit peut agir comme une forme de régularisation, améliorant la généralisation du modèle.

Compréhension des réseaux de neurones modernes.

- **Rôles des couches d'entrée, cachées et de sortie :**
  - **Couche d'entrée :** Reçoit les données brutes. Chaque neurone correspond généralement à une caractéristique de l'entrée, transmettant ces informations aux couches suivantes.
  - **Couches cachées :** Effectuent des transformations non linéaires, apprenant des représentations de plus en plus abstraites et complexes des données.
  - **Couche de sortie :** Produit la prédiction finale du modèle. Le nombre de neurones et la fonction d'activation dépendent de la tâche (classification, régression).
- **Processus de rétropropagation du gradient (backpropagation) en termes simples :** La rétropropagation est l'algorithme d'apprentissage des réseaux de neurones. Pour chaque itération :
  1. **Passe avant (Forward Pass) :** Les données sont propagées pour générer une prédiction.
  2. **Passe arrière (Backward Pass / Rétropropagation) :**

— L'erreur entre prédiction et valeur réelle est calculée.
— Cette erreur est propagée en arrière pour calculer la contribution (gradient) de chaque poids et biais à l'erreur.
— Un optimiseur utilise ces gradients pour ajuster les poids et les biais afin de minimiser l'erreur.

1.2. Exercice 1 : Construction d'un réseau de neurones avec Keras

Question 1 : Expliquez l'utilité des couches Dense et Dropout. Pourquoi la fonction d'activation softmax est-elle utilisée dans la couche de sortie pour ce problème de classification ?

- **Utilité des couches Dense (Fully Connected) :** Chaque neurone reçoit une entrée de tous les neurones de la couche précédente. Elles sont utilisées pour apprendre des motifs complexes et des relations non linéaires, combinant les informations pour les tâches de classification ou régression.
- **Utilité des couches Dropout :** Technique de régularisation qui désactive aléatoirement un pourcentage de neurones pendant l'entraînement (par exemple, 20% dans le code). Cela prévient le surapprentissage en forçant le réseau à apprendre des représentations plus robustes, car aucun neurone ne dépend trop d'un autre.
- **Pourquoi Softmax dans la couche de sortie pour la classification :** La fonction '`softmax`' est idéale pour la classification multi-classes (comme MNIST avec 10 classes). Elle transforme un vecteur de logits en une distribution de probabilité, où la somme des sorties est 1. Chaque sortie représente la probabilité que l'entrée appartienne à la classe correspondante, fournissant des scores de confiance.

Question 2 : L'optimiseur adam est utilisé. Faites une recherche et expliquez brièvement en quoi il s'agit d'une amélioration par rapport à la SGD simple.

- **Adam (Adaptive Moment Estimation)** est un optimiseur qui combine les avantages d'Adagrad et RMSprop.
- **Améliorations par rapport à la SGD simple :**
  1. **Taux d'apprentissage adaptatifs :** Adam calcule des taux d'apprentissage dynamiques pour *chaque paramètre* en utilisant des moyennes mobiles des gradients passés (premier et second moments). Cela permet une convergence plus rapide dans les vallées plates et plus stable avec de grands gradients.
  2. **Correction du biais :** Adam inclut des termes de correction du biais pour les estimations des moments au début de l'entraînement, les rendant plus fiables.
  3. **Convergence plus rapide :** Généralement plus rapide et stable, en particulier pour les architectures profondes.
  4. **Moins de réglage manuel :** Nécessite moins d'ajustements manuels du taux d'apprentissage que la SGD.

Question 3 : Comment les concepts de "vectorisation" et de "calculs par lots" sont-ils appliqués dans le code ci-dessus ?

- **Vectorisation :** Opérations effectuées sur des tableaux entiers (vecteurs ou matrices) plutôt que sur des éléments individuels.
  - **Exemples dans le code :**
    - ‘`x_train.astype("float32") / 255.0`‘ : Normalisation des pixels appliquée à toutes les valeurs simultanément.
    - ‘`x_train.reshape(60000, 784)`‘ : Redimensionnement des images effectué sur l’ensemble du tableau.
    - Les opérations internes des couches ‘Dense‘ et des fonctions d’activation sont intrinsèquement vectorisées (multiplications matricielles).
- **Calculs par lots (Batch Processing) :** Division de l’ensemble de données en petits sous-ensembles (lots) pour les mises à jour des poids.
  - **Application dans le code :**
    - ‘`model.fit(..., batch_size=128, ...)`‘ : Entraîne le modèle en utilisant des lots de 128 images. Les gradients sont calculés et les poids mis à jour une fois par lot.
  - **Avantages :** Stabilité des mises à jour des gradients, rapidité par rapport au Batch Gradient Descent, et efficacité sur les architectures matérielles modernes (GPU/CPU) grâce à la parallélisation.

Partie 2 : Ingénierie du Deep Learning (5h)

1.1. Exercice 5 : Déploiement et CI/CD

Question 1 : Expliquez comment un pipeline de CI/CD (par exemple, avec GitHub Actions) pourrait automatiser la construction et le déploiement de votre image Docker sur un service comme Google Cloud Run ou Amazon Elastic Container Service.

Un pipeline de CI/CD automatise le développement, le test et le déploiement. Avec GitHub Actions :

1. **Déclenchement :** Pousser du code sur une branche spécifique ou ouvrir une Pull Request.
2. **Intégration Continue (CI) :**
   - **Vérification & Tests :** Extrait le code, exécute les tests unitaires/intégration. Échec = arrêt du pipeline.
   - **Construction Docker :** Utilise le '`Dockerfile`' pour construire l'image Docker (installe les dépendances, copie le code).
   - **Tagging & Push :** Tagge l'image avec une version unique et la pousse vers un registre d'images (GCR/Artifact Registry pour Google Cloud, ECR pour AWS).
3. **Déploiement Continu (CD) :**
   - **Authentification :** S'authentifie auprès du service cloud cible (Google Cloud ou AWS) via des secrets GitHub Actions.
   - **Déploiement (Cloud Run/ECS) :** Utilise l'interface CLI ('`gcloud`' ou '`aws cli`') pour déployer la nouvelle image Docker. Spécifie la configuration (port, ressources). Cloud Run crée une nouvelle révision, ECS met à jour la définition de tâche.
   - **Tests post-déploiement (optionnel) :** Exécute des tests pour valider le fonctionnement de l'application déployée.
   - **Notification :** Informe du succès ou de l'échec du déploiement.

Le pipeline CI/CD transforme donc un push de code en un processus automatisé de test, construction et déploiement, garantissant rapidité, fiabilité et cohérence.

Question 2 : Une fois le modèle déployé, quels sont les indicateurs clés que vous mettriez en place pour le monitoring et le débogage en production ? Citez au moins trois types d'indicateurs.

Le monitoring est essentiel pour assurer la performance et détecter les problèmes.

1. **Indicateurs de Performance Technique de l'Application (API/Infrastructure) :** Mesurent la santé et la réactivité de l'application web et de son infrastructure.
   - **Latence des requêtes (Response Time) :** Temps moyen/médian/$95^{\text{ème}}$ percentile pour les requêtes API.

- **Taux d'erreurs (Error Rate) :** Pourcentage de requêtes API échouant (HTTP 5xx, 4xx).
- **Utilisation des ressources :** Utilisation CPU, mémoire, réseau de l'instance.

2. **Indicateurs de Performance du Modèle (Qualité des Prédictions) :** Mesurent l'exactitude et la fiabilité des prédictions du modèle en production (nécessite les vraies étiquettes).
   - **Précision (Accuracy) / F1-Score / Recall / Précision :** Métriques de performance calculées sur les prédictions en production (ex : MNIST accuracy).
   - **Confiance moyenne des prédictions :** Valeur moyenne des probabilités de Softmax. Une baisse peut indiquer que le modèle est moins sûr.
   - **Distribution des prédictions :** Fréquence de chaque classe prédite. Un changement peut signaler un problème.
3. **Indicateurs de Dérive des Données (Data Drift) et Dérive du Concept (Concept Drift) :** Mesurent si les données d'entrée ou la relation entrée-sortie ont changé.
   - **Distribution des caractéristiques d'entrée :** Statistiques (moyenne, écart-type, histogramme) des données entrantes comparées aux données d'entraînement.
   - **Valeurs manquantes ou aberrantes :** Fréquence de valeurs inattendues ou extrêmes.
   - **Distribution des labels réels vs. prédits :** Comparaison si les labels réels sont disponibles.

Références

[1] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[2] F. Chollet, *Deep Learning with Python*. Manning Publications, 2017.

[3] D. Kingma and J. Ba, "Adam : A Method for Stochastic Optimization," *International Conference on Learning Representations (ICLR)*, 2015.

[4] M. Fowler, *Continuous Integration*. Addison-Wesley, 2018.

[5] G. Kim, et al., *The DevOps Handbook*. IT Revolution Press, 2020.

[6] M. Sculley, et al., "Machine Learning : The High-Interest Credit Card of Technical Debt," in *Proceedings of the 28th International Conference on Neural Information Processing Systems (NIPS)*, 2015.