Checkpointing Efficace pour les Réseaux de Neurones Profonds

Bryan Chen [1] Wei Tsang Ooi [2] Yannis Montreuil [3]
Lai Xing Ng [4] Axel Carlier [1]

[1] Toulouse INP - ENSEEIHT, France
[2] National University of Singapore, Singapour
[3] Université Pierre et Marie Curie, France
[4] A*STAR, Agency for Science, Technology and Research, Singapour

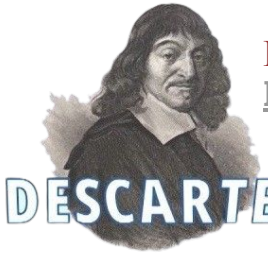
Soutenance PFE à l'ENSEEIHT, Septembre 2024

1. Contexte du sujet
   - Programme Descartes
   - Working package 4
   - Empoisonnement des données
   - Sujet de stage
2. Problématique
3. Démarche adoptée
   - LC-checkpoint (Lossy compression checkpoint)
   - Notre optimisation
   - Delta-LoRA
   - Approche proposée
4. Résultats
5. Conclusion

Contexte du sujet

Plusieurs projets au sein de CNRS@Create :

- Programme Descartes (5 ans)
- Programme Space (2 ans)
- Programme Calypso (2 ans)
- Programme EcoCTs (2 ans)
- Programme ScanCells (2 ans)



INTELLIGENT MODELLING FOR
DEciSion making in CriticAl urban sysTEmS

ville intelligente (smart city)
centrée sur les personnes

Contexte du sujet

Plusieurs projets au sein de CNRS@Create :

- Programme Descartes (5 ans)
- Programme Space (2 ans)
- Programme Calypso (2 ans)
- Programme EcoCTs (2 ans)
- Programme ScanCells (2 ans)

cnrs → cnrs @CREATE Singapore — CREATE Campus for Research Excellence And Technological Enterprise

DESCARTES

INTELLIGENT MODELLING FOR DEciSion making in CriticAl urban sysTEmS

ville intelligente (smart city) **centrée sur les personnes**

mettre en œuvre de nouvelles méthodes **hybrides d'intelligence artificielle** (hybrid AI ou HAI) pour:

- améliorer la prise de décision
- répondre aux limites de l'IA actuelle (disponibilité, responsabilité, interaction homme-IA, confiance)

Programme Descartes

Programme Descartes comprend ainsi 3 piliers :

- Calcul intelligent
- Le pouvoir aux mains des citoyens dans les villes
- Ingénierie et builder

Working package 4 : Collaboration Humain et IA

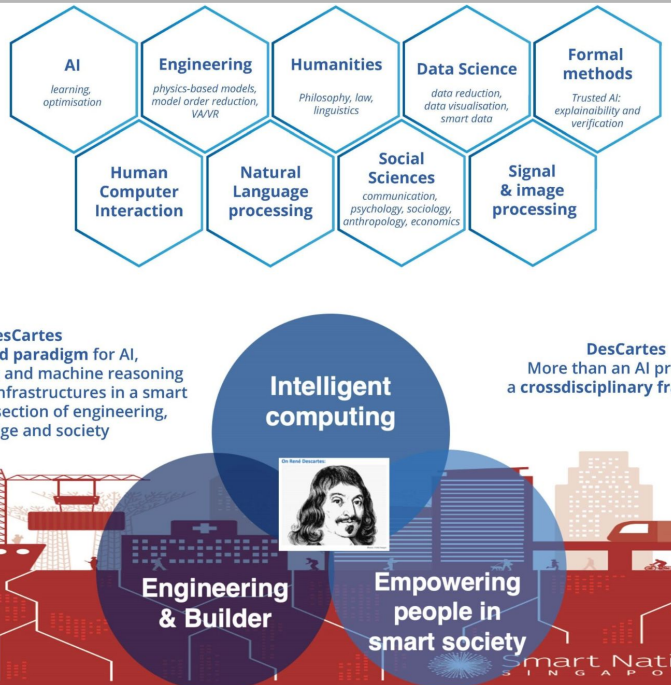


Fig.1: Les trois piliers du programme Descartes

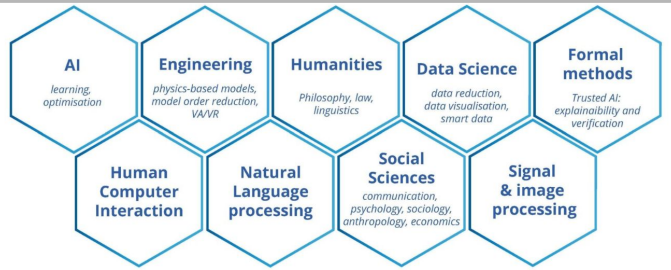
Programme Descartes

Programme Descartes comprend ainsi 3 piliers :

- Calcul intelligent
- Le pouvoir aux mains des citoyens dans les villes
- Ingénierie et builder

Working package 4 : Collaboration Humain et IA

- apporter des aspects humains (non modélisable informatiquement) dans les systèmes d'IA
- augmenter la perception et la cognition humaines (aide à la prise de décision)



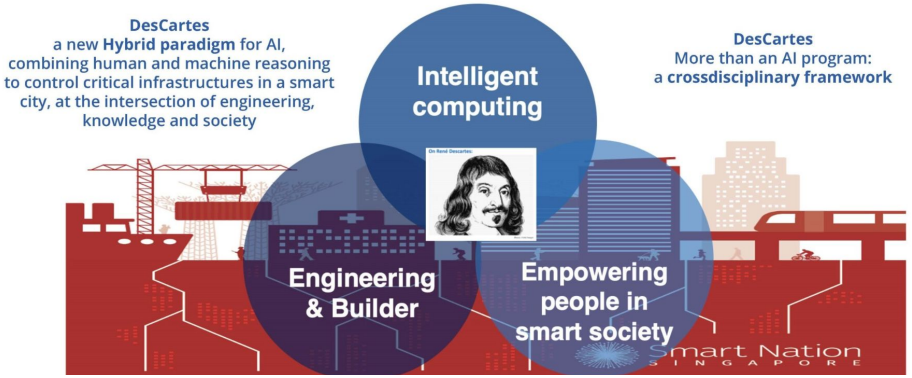


Fig.1: Les trois piliers du programme Descartes

Working package 4 (WP4)

Le WP4 intègre un / plusieurs expert(s) (opérateur(s) / contrôleur(s)) dans la boucle d'apprentissage (prise de décision jointe) des modèles d'IA

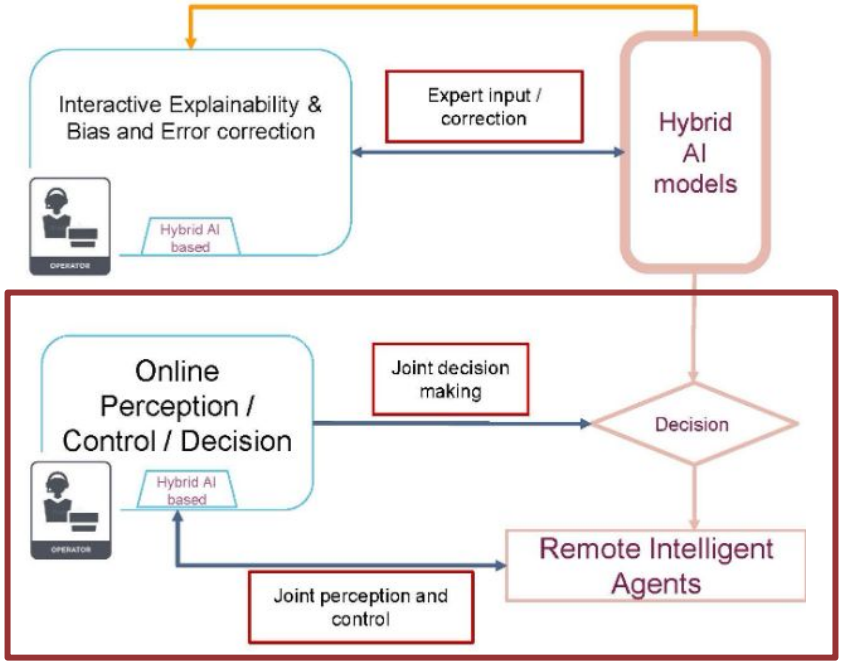


Fig.2: Présentation du WP4

Working package 4 (WP4)

Le WP4 intègre un / plusieurs expert(s) (opérateur(s) / contrôleur(s)) dans la boucle d'apprentissage (prise de décision jointe) des modèles d'IA

Lorsque le modèle d'IA n'admet pas une bonne précision, alors le modèle diffère (learning-to-defer) aux experts qui apporteront leur expertise au problème

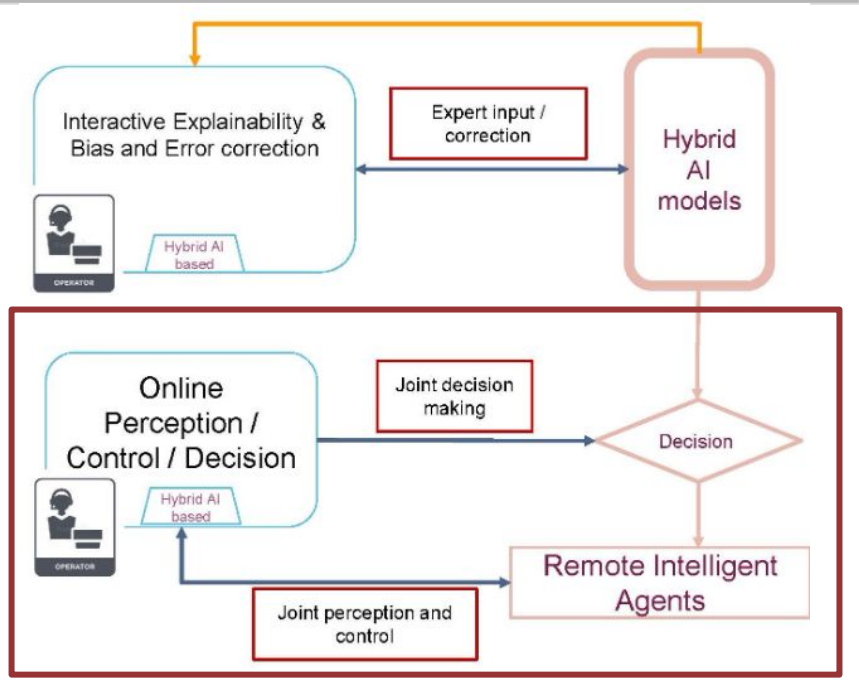


Fig.2: Présentation du WP4

Working package 4 (WP4)

Le WP4 intègre un / plusieurs expert(s) (opérateur(s) / contrôleur(s)) dans la boucle d'apprentissage (prise de décision jointe) des modèles d'IA

Lorsque le modèle d'IA n'admet pas une bonne précision, alors le modèle diffère (learning-to-defer) aux experts qui apporteront leur expertise au problème

Le modèle apprend la nouvelle connaissance apportée par l'expert

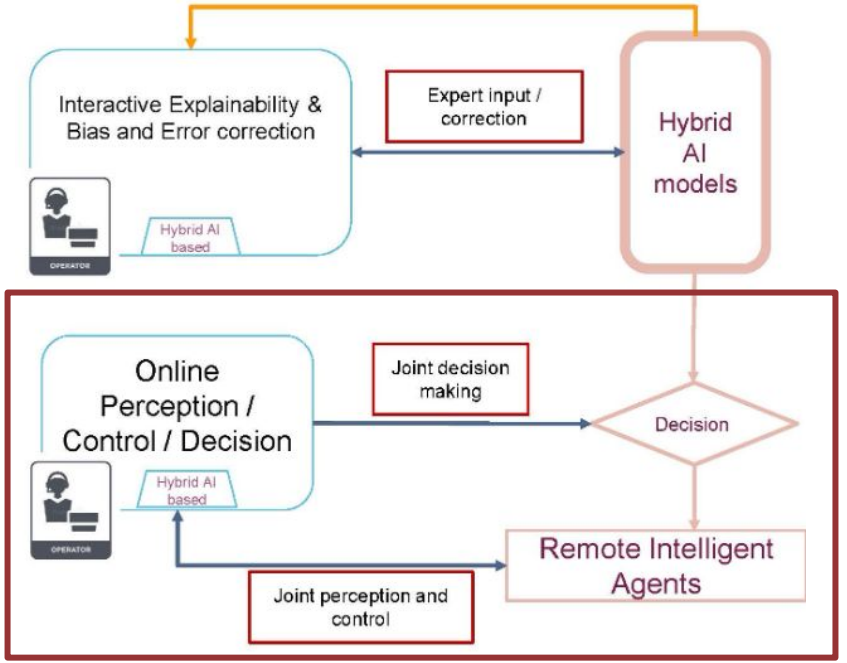


Fig.2: Présentation du WP4

Working package 4 (WP4)

Le WP4 intègre un / plusieurs expert(s) (opérateur(s) / contrôleur(s)) dans la boucle d'apprentissage (prise de décision jointe) des modèles d'IA

Lorsque le modèle d'IA n'admet pas une bonne précision, alors le modèle diffère (learning-to-defer) aux experts qui apporteront leur expertise au problème

Le modèle apprend la nouvelle connaissance apportée par l'expert

Eviter les hallucinations dans les modèles d'IA

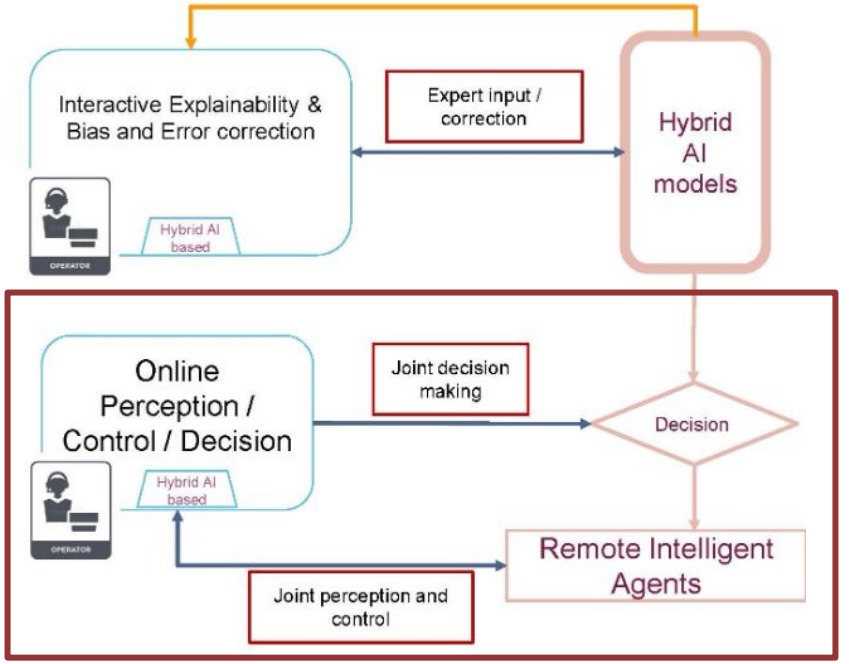


Fig.2: Présentation du WP4

Sujet de stage

Que faire si l'expert est malicieux ou corrompu, on dit qu'il y a **empoisonnement des données**.

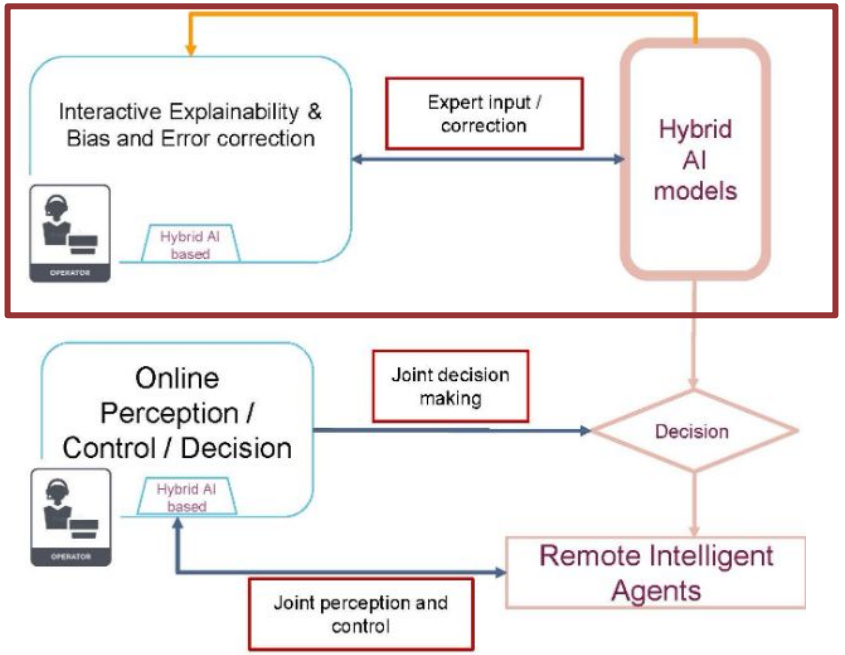


Fig.2: Présentation du WP4

Empoisonnement des données

Que faire si l'expert est malicieux ou corrompu, on dit qu'il y a **empoisonnement des données**.

Hypothèse : On est capable de **détecter l'empoisonnement**

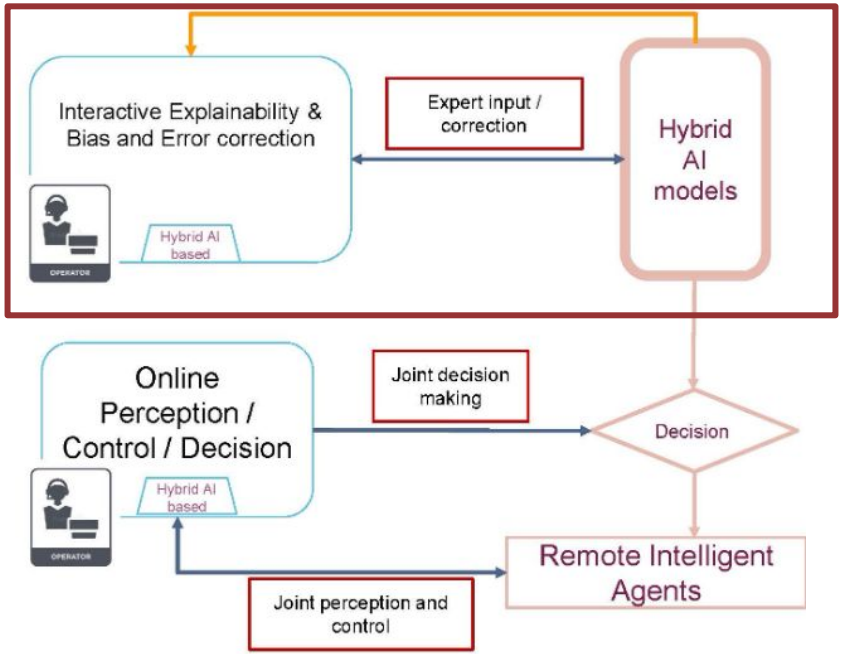


Fig.2: Présentation du WP4

Empoisonnement des données

Que faire si l'expert est malicieux ou corrompu, on dit qu'il y a **empoisonnement des données**.

Hypothèse : On est capable de **détecter l'empoisonnement**

Solution 1 :

On entraîne le modèle à partir du début sur l'ensemble de donnée nettoyé

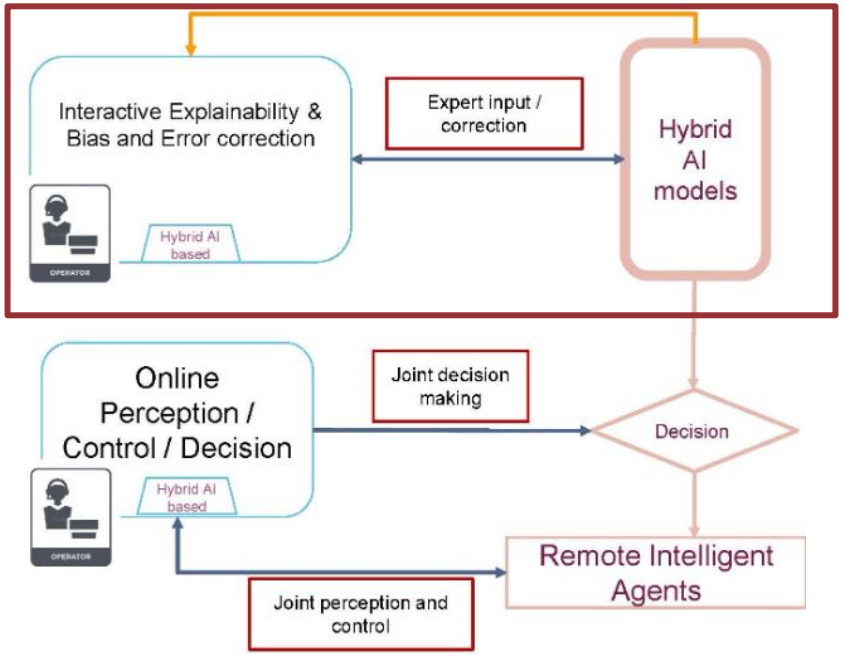


Fig.2: Présentation du WP4

Empoisonnement des données

Que faire si l'expert est malicieux ou corrompu, on dit qu'il y a **empoisonnement des données**.

Hypothèse : On est capable de **détecter l'empoisonnement**

Solution 1 :

On entraîne le modèle à partir du début sur l'ensemble de donnée nettoyé

Solution naïve

- Coûteux en temps
- Coûteux en calcul

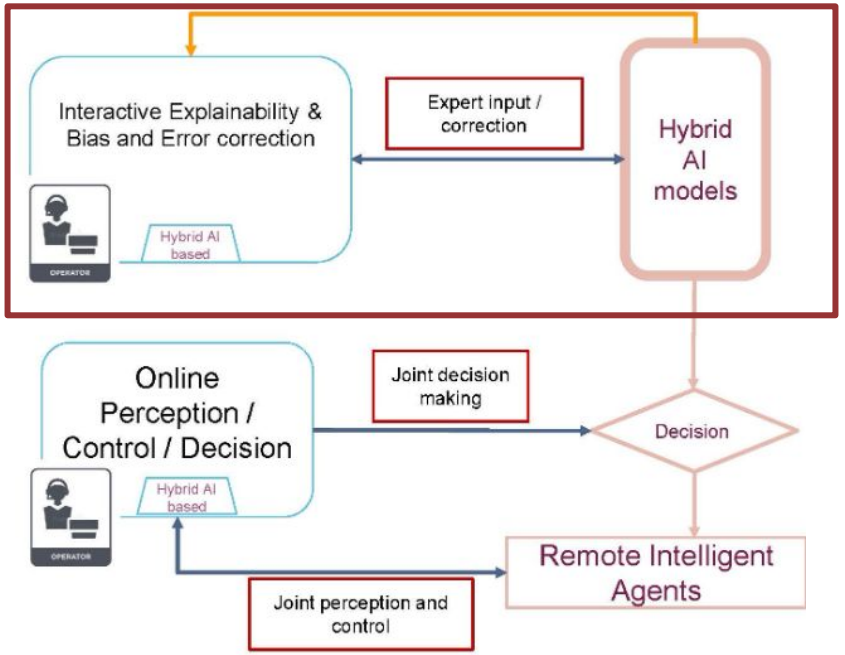


Fig.2: Présentation du WP4

Empoisonnement des données

Que faire si l'expert est malicieux ou corrompu, on dit qu'il y a **empoisonnement des données**.

Hypothèse : On est capable de **détecter l'empoisonnement**

Solution 1 :

On entraîne le modèle à partir du début sur l'ensemble de donnée nettoyé

Solution naïve

- Coûteux en temps
- Coûteux en calcul

Notre solution :

On implémente un système efficace de points de branchement (branch point) lors de l'entraînement

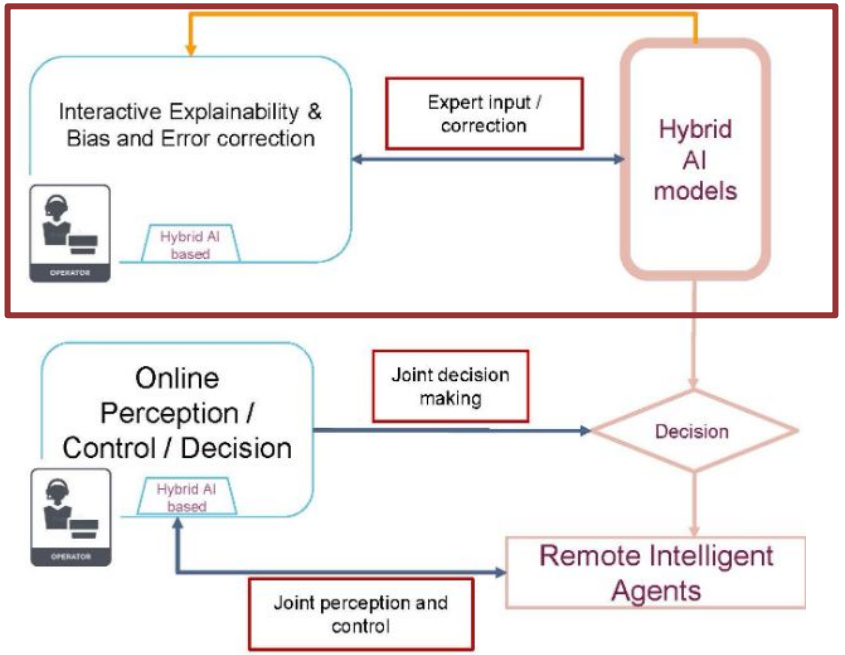


Fig.2: Présentation du WP4

Empoisonnement des données

Que faire si l'expert est malicieux ou corrompu, on dit qu'il y a **empoisonnement des données**.

Hypothèse : On est capable de **détecter l'empoisonnement**

Solution 1 :

On entraîne le modèle à partir du début sur l'ensemble de donnée nettoyé

Solution naïve

- Coûteux en temps
- Coûteux en calcul

Notre solution :

On implémente un système efficace de points de branchement (branch point) lors de l'entraînement

S'il y a détection

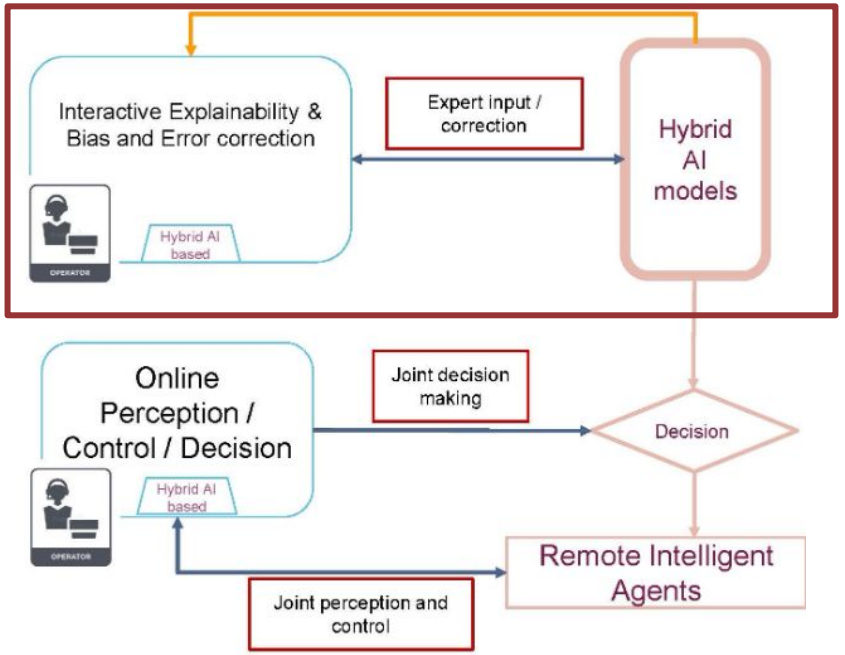


Fig.2: Présentation du WP4

Empoisonnement des données

Que faire si l'expert est malicieux ou corrompu, on dit qu'il y a **empoisonnement des données**.

Hypothèse : On est capable de **détecter l'empoisonnement**

Solution 1 :

On entraîne le modèle à partir du début sur l'ensemble de donnée nettoyé

Solution naïve

- Coûteux en temps
- Coûteux en calcul

Notre solution :

On implémente un système efficace de points de branchement (branch point) lors de l'entraînement

S'il y a détection

Entraînement à partir du dernier point de branchement précédant la détection

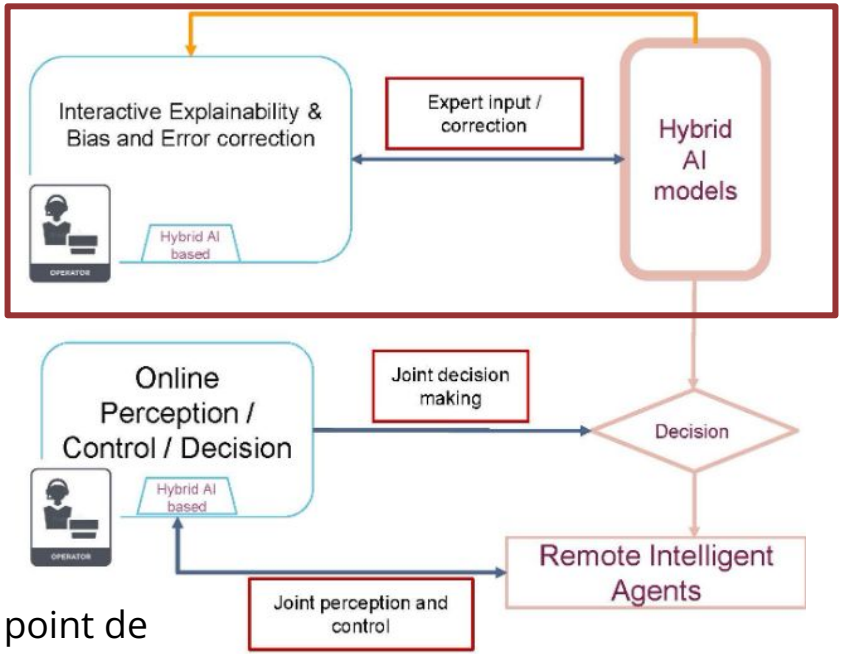


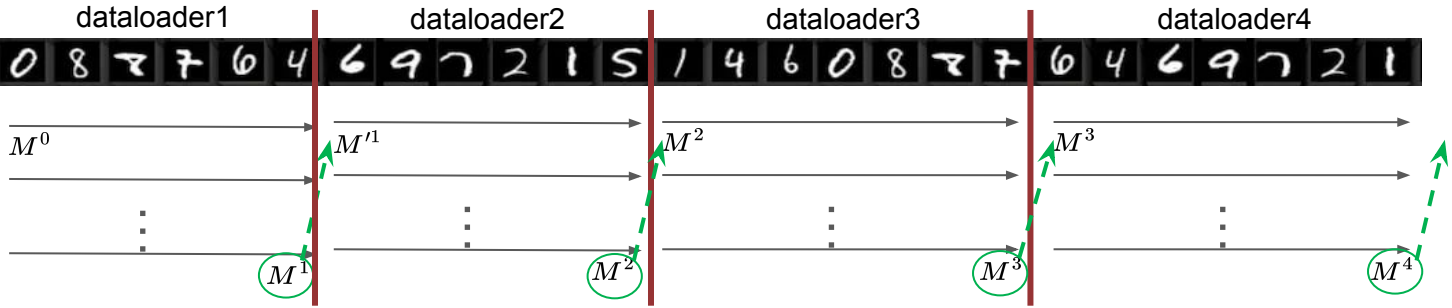
Fig.2: Présentation du WP4

Empoisonnement des données

Schéma du système de points de branchement :

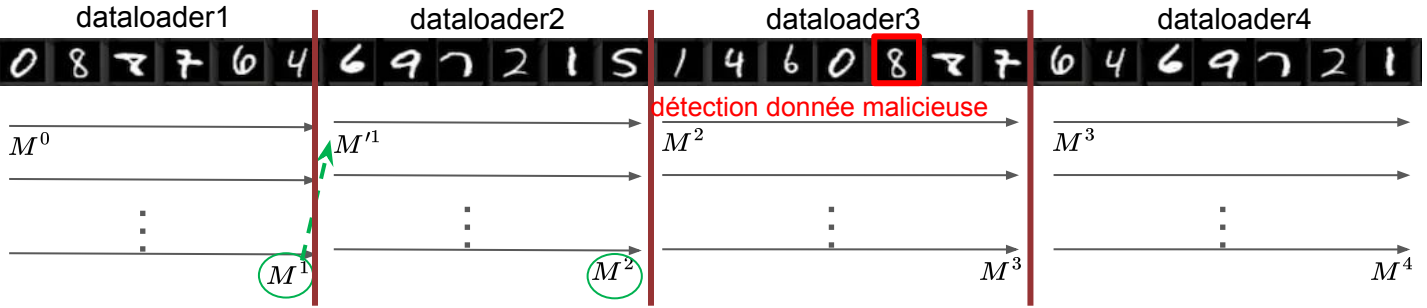
Empoisonnement des données

Schéma du système de points de branchement :



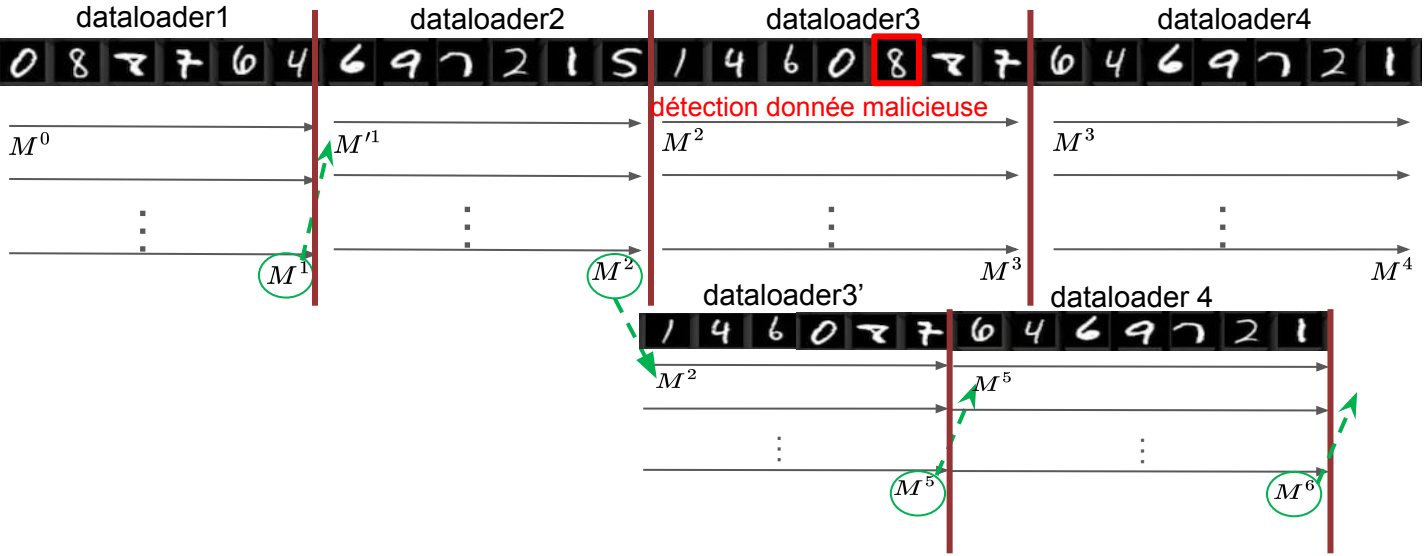
Empoisonnement des données

Schéma du système de points de branchement :



Empoisonnement des données

Schéma du système de points de branchement :



Sujet de stage

Checkpointing Efficace pour les réseaux de neurones profonds :

Être robuste à la panne (explosion de gradient, division par zéro, machine en panne, etc.)

Sujet de stage

Checkpointing Efficace pour les réseaux de neurones profonds :

Être robuste à la panne (explosion de gradient, division par zéro, machine en panne, etc.)

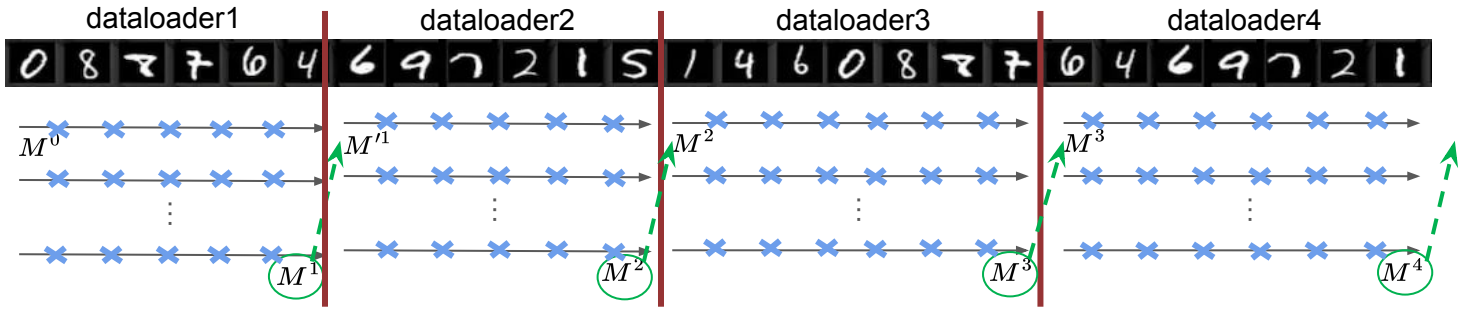
Utilisation de points de reprise (= checkpoint) (≠ point de branchement) pendant l'entraînement au sein d'un ensemble d'entraînement

Sujet de stage

Checkpointing Efficace pour les réseaux de neurones profonds :

Être robuste à la panne (explosion de gradient, division par zéro, machine en panne, etc.)

Utilisation de points de reprise (= checkpoint) (≠ point de branchement) pendant l'entraînement au sein d'un ensemble d'entraînement

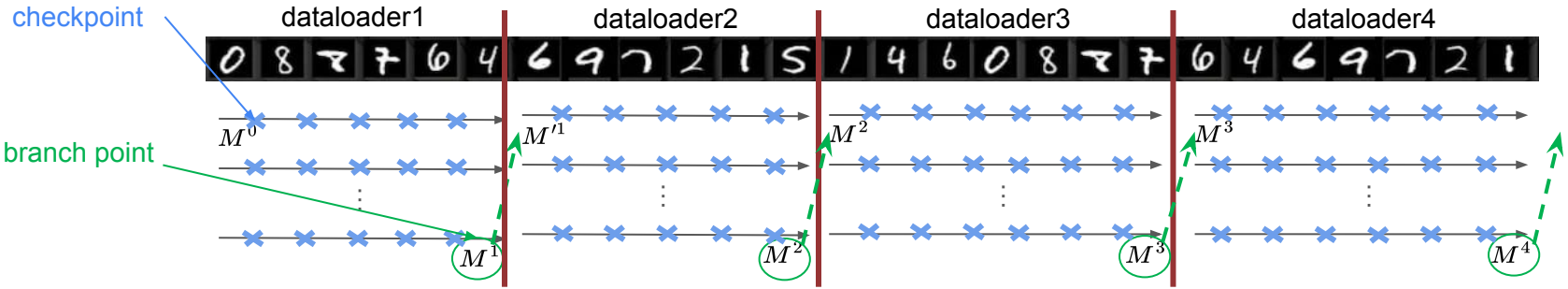


Sujet de stage

Checkpointing Efficace pour les réseaux de neurones profonds :

Être robuste à la panne (explosion de gradient, division par zéro, machine en panne, etc.)

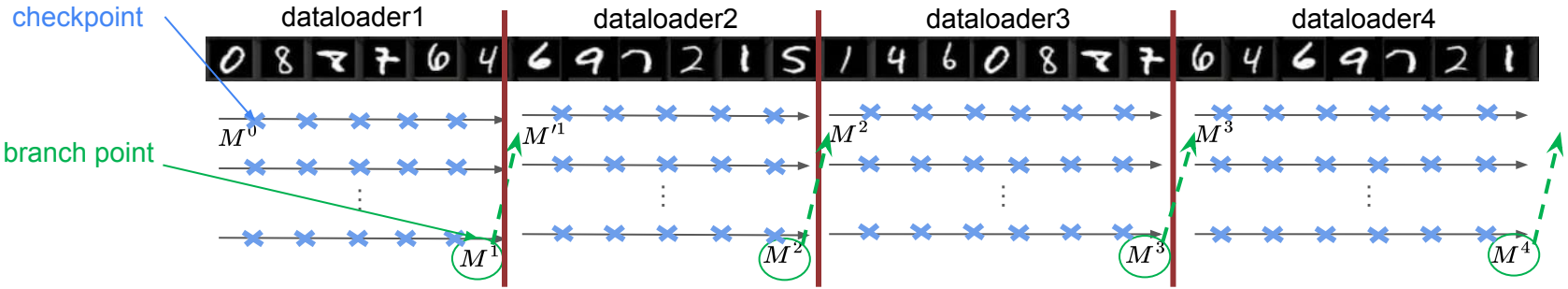
Utilisation de points de reprise (= checkpoint) (≠ point de branchement) pendant l'entraînement au sein d'un ensemble d'entraînement



Sujet de stage

Checkpointing Efficace pour les réseaux de neurones profonds :

Si on save le modèle entier à chaque fois, cela coûte en espace de stockage



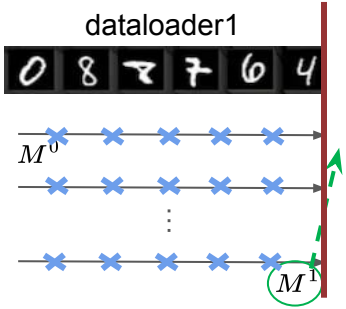
Problématique

Problématique

Comment créer un système d'entraînement robuste à la panne et à l'empoisonnement de donnée tout en minimisant l'espace de stockage nécessaire ?

Démarche adoptée

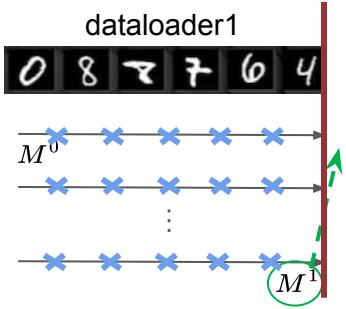
Pour chaque ensemble de donnée (dataloader)



Démarche adoptée

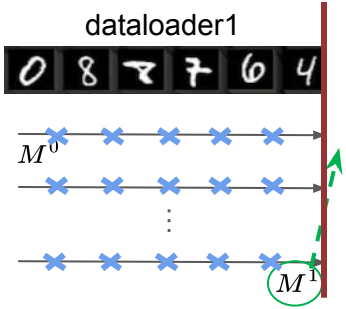
Pour chaque ensemble de donnée (dataloader)

Optimisation en combinant deux techniques de compression



Démarche adoptée

Pour chaque ensemble de donnée (dataloader)

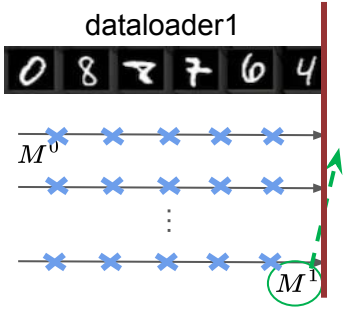


Optimisation en combinant deux techniques de compression

- LC-checkpoint (On Efficient Constructions of Checkpoints) [1]

Démarche adoptée

Pour chaque ensemble de donnée (dataloader)

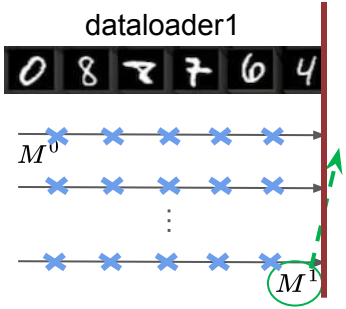


Optimisation en combinant deux techniques de compression

- LC-checkpoint (On Efficient Constructions of Checkpoints) [1]
- Delta-LoRA (Delta-LoRA: Fine-Tuning High-Rank Parameters with the Delta of Low-Rank Matrices) [2]

Démarche adoptée

Pour chaque ensemble de donnée (dataloader)



Optimisation en combinant deux techniques de compression

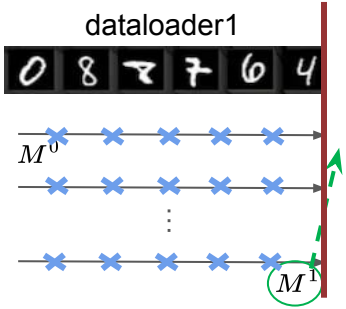
- LC-checkpoint (On Efficient Constructions of Checkpoints) [1]
- Delta-LoRA (Delta-LoRA: Fine-Tuning High-Rank Parameters with the Delta of Low-Rank Matrices) [2]

Au sein des techniques utilisées pour LC-checkpoint

- Codage de Huffman [3] ↔ Algorithme GZip

Démarche adoptée

Pour chaque ensemble de donnée (dataloader)



Optimisation en combinant deux techniques de compression

- LC-checkpoint (On Efficient Constructions of Checkpoints) [1]
- Delta-LoRA (Delta-LoRA: Fine-Tuning High-Rank Parameters with the Delta of Low-Rank Matrices) [2]

Au sein des techniques utilisées pour LC-checkpoint

- Codage de Huffman [3] ↔ Algorithme GZip

LC-checkpoint (Lossy compression checkpoint)

Schéma de codage différentiel / d'encodage delta :

LC-checkpoint (Lossy compression checkpoint)

Schéma de codage différentiel / d'encodage delta : données sous forme de différences (deltas) entre des données séquentielles

LC-checkpoint (Lossy compression checkpoint)

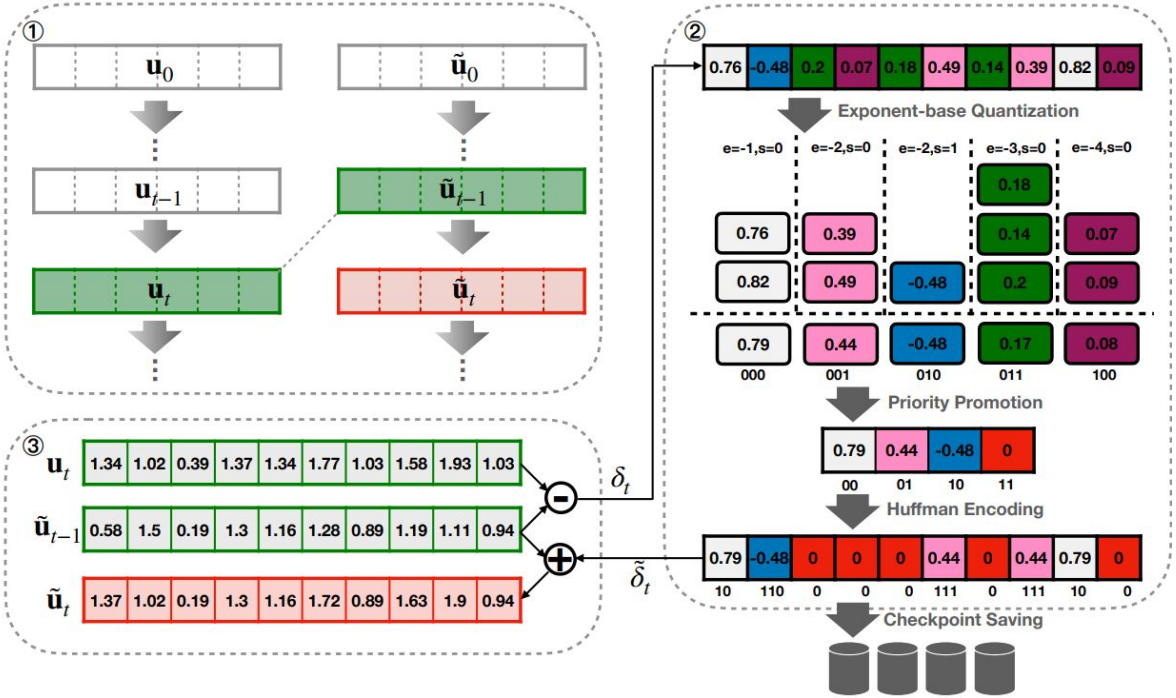
Schéma de codage différentiel / d'encodage delta : données sous forme de différences (deltas) entre des données séquentielles

$$\tilde{\mathbf{u}}_t = \mathbf{u}_0 + \sum_{i<t} \tilde{\delta}_i$$

- $\tilde{\mathbf{u}}_t$ approximation de l'état vérité terrain du modèle
- $\mathbf{u}_0$ état initial du modèle
- $\tilde{\delta}_i$ checkpoint sauvegardé par le système à l'itération i

LC-checkpoint (Lossy compression checkpoint)

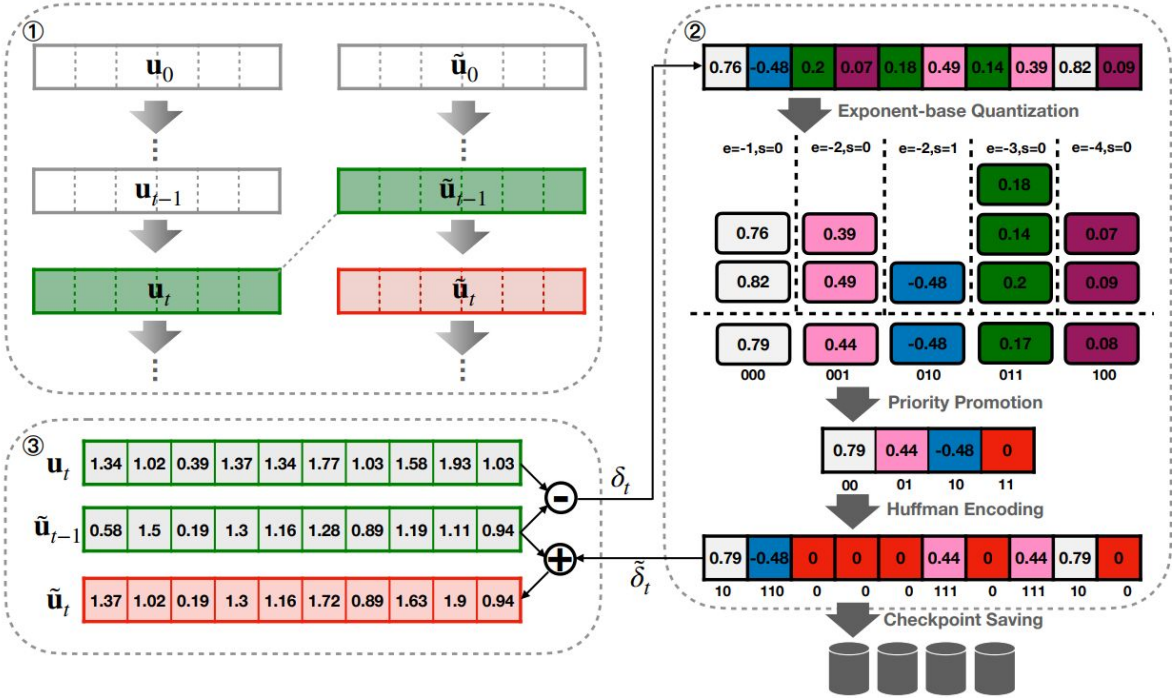
Combinaison de plusieurs techniques :



LC-checkpoint (Lossy compression checkpoint)

Combinaison de plusieurs techniques :

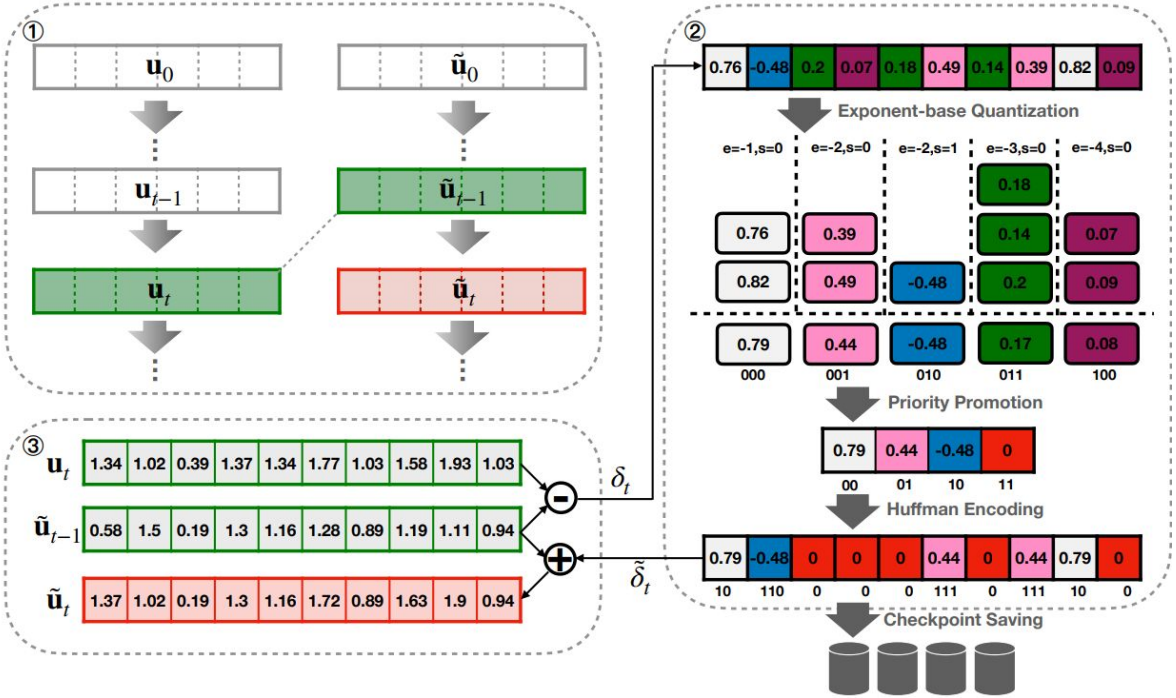
- Quantification à base exponentielle



LC-checkpoint (Lossy compression checkpoint)

Combinaison de plusieurs techniques :

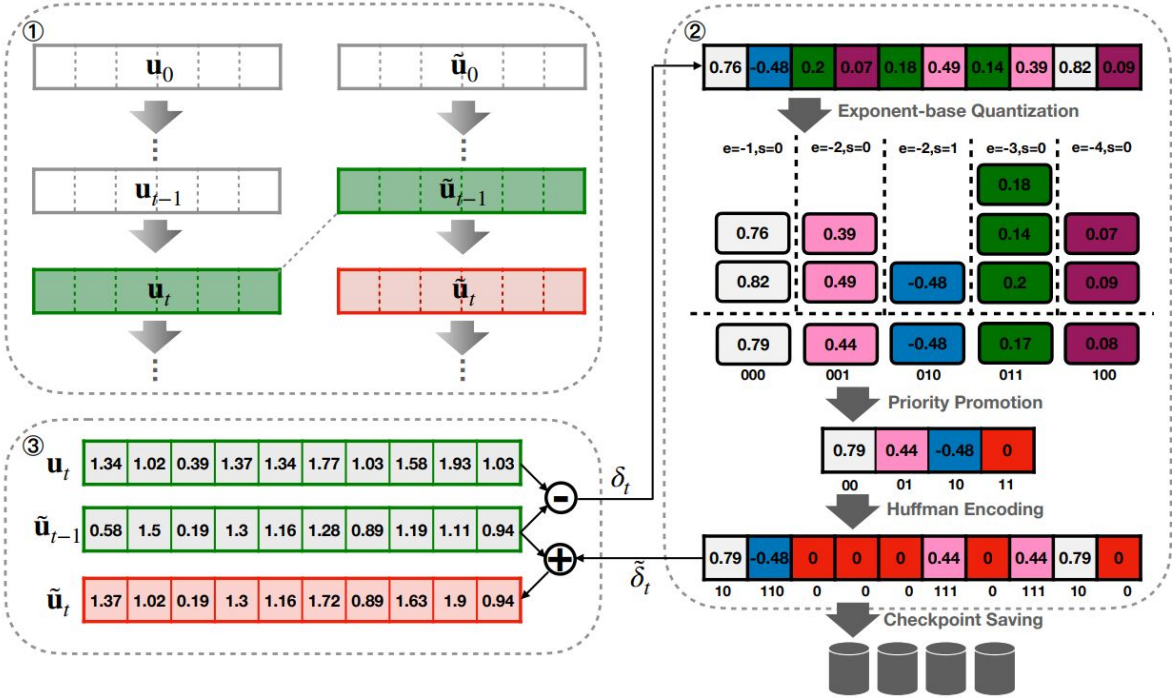
- Quantification à base exponentielle
- Promotion de priorité



LC-checkpoint (Lossy compression checkpoint)

Combinaison de plusieurs techniques :

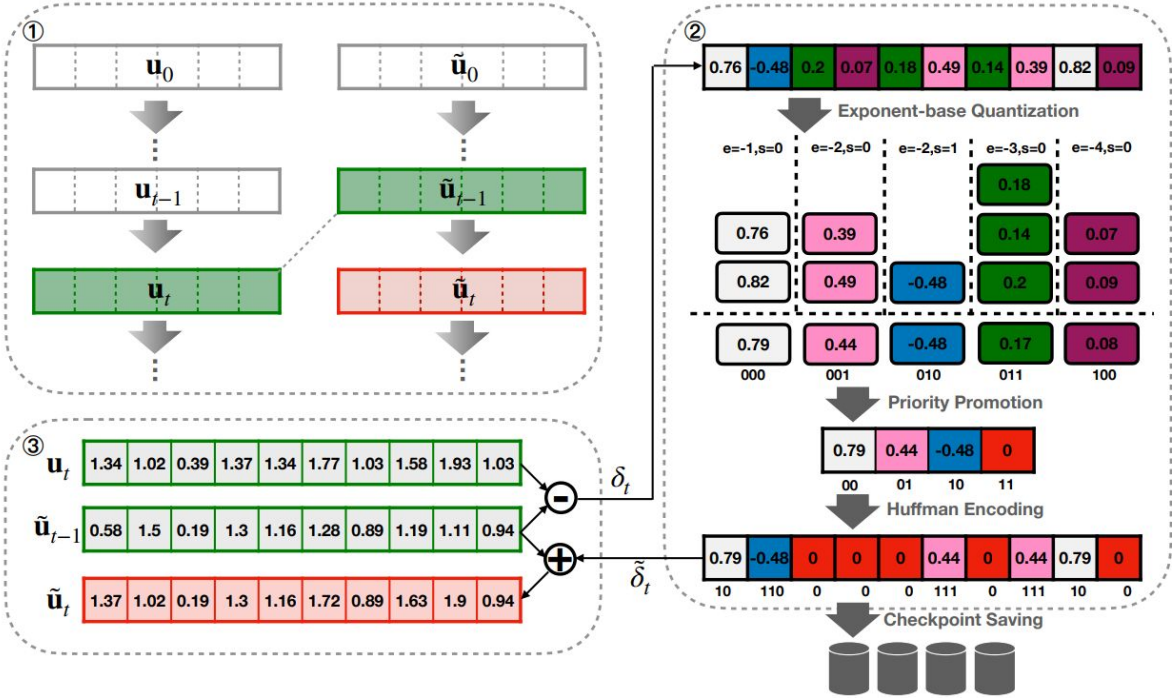
- Quantification à base exponentielle
- Promotion de priorité
- Encodage de Huffman



LC-checkpoint (Lossy compression checkpoint)

Combinaison de plusieurs techniques :

- Quantification à base exponentielle
- Promotion de priorité
- Encodage de Huffman



LC-checkpoint (Lossy compression checkpoint)

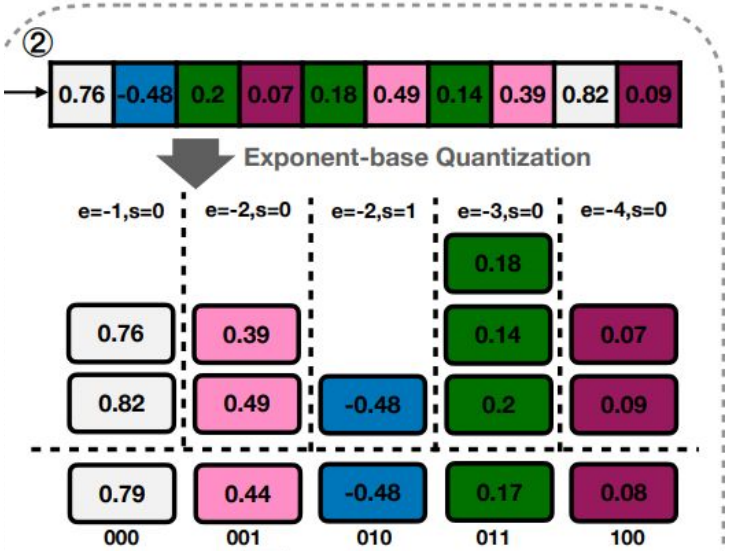
Quantification à base exponentielle :

→ **réduit la précision en conservant une bonne approximation**

utilise la base exponentielle : $v = (-1)^s \times m \times 2^e$

- v : valeur flottante
- s : signe
- m : mantisse
- e : exposant

ajuster la base exponentielle permet de contrôler le niveau de compression et la perte d'information



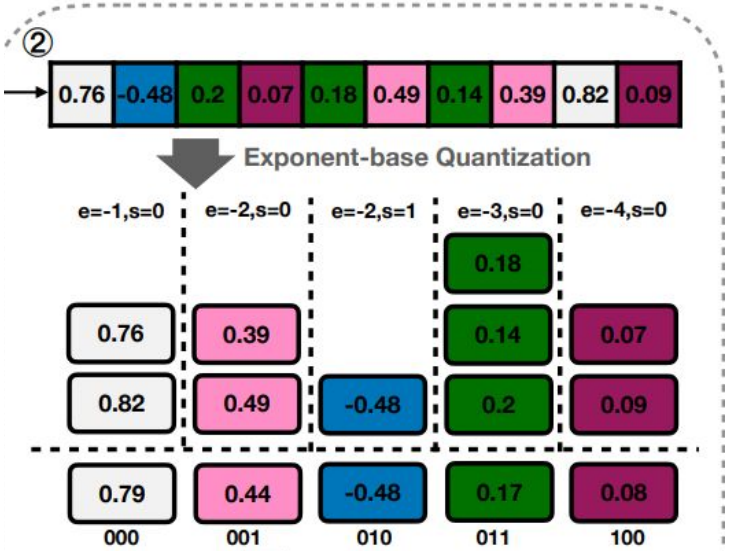
LC-checkpoint (Lossy compression checkpoint)

Quantification à base exponentielle :

on va encoder par $\delta_t = \mathbf{u}_t - \tilde{\mathbf{u}}_{t-1} \in \mathbf{R}^n$

1/ Répartit les entrées δ_t en plusieurs groupes en fonction de **l'exposant et des signes identiques**

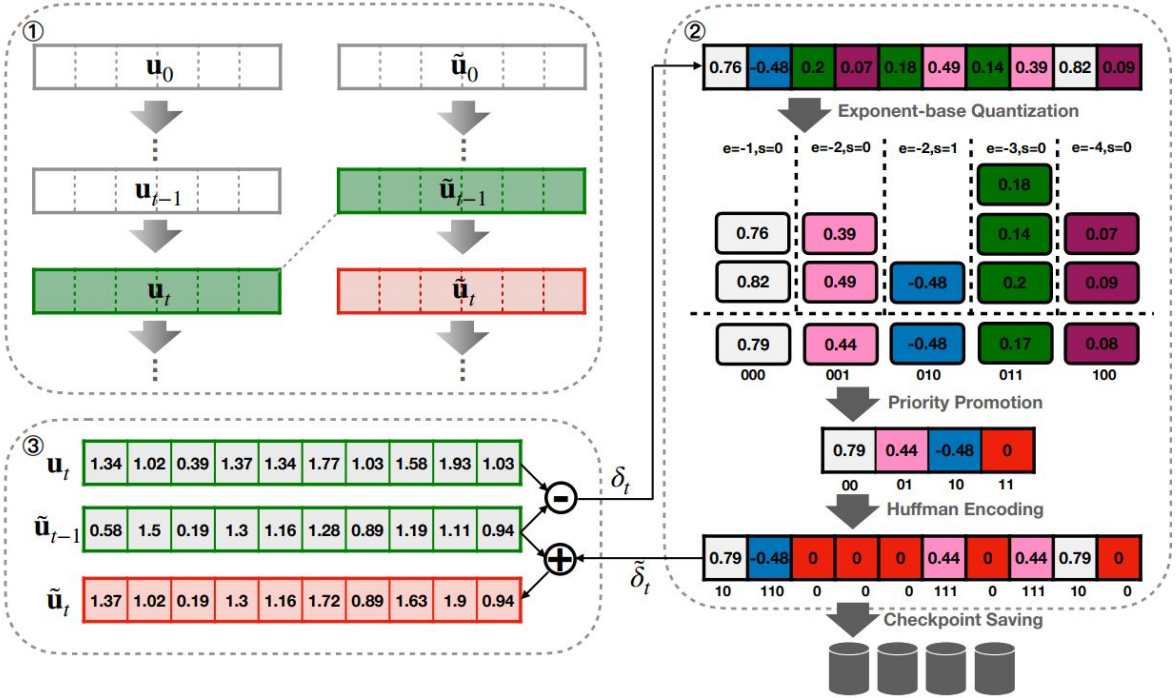
2/ Représente chaque bucket par la **moyenne des valeurs maximales et minimales**



LC-checkpoint (Lossy compression checkpoint)

Combinaison de plusieurs techniques :

- Quantification à base exponentielle
- Promotion de priorité
- Encodage de Huffman

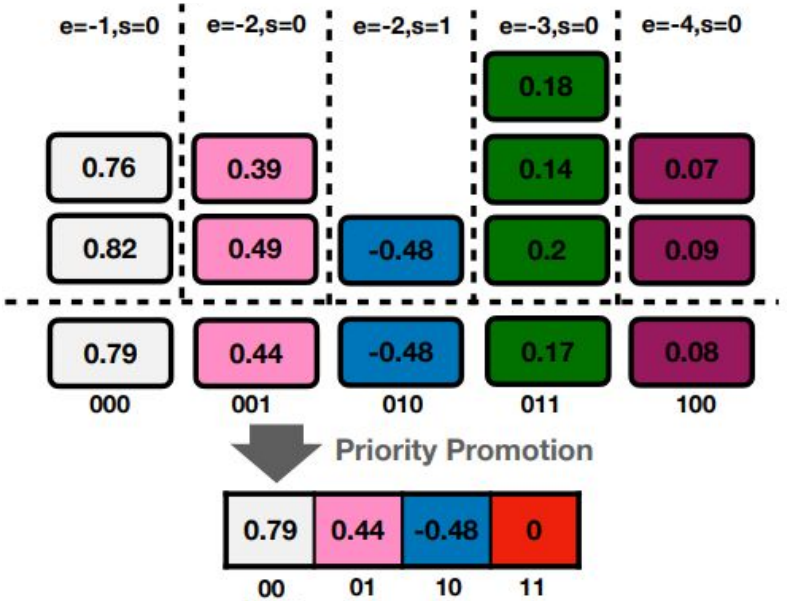


LC-checkpoint (Lossy compression checkpoint)

Promotion de priorité

Conserve $2^x - 1$ buckets avec le **plus grand e** et fusionne le reste **en un seul avec la valeur 0**, avec x la largeur de bit

Remarque : Une largeur de bits de 3 a été jugée suffisante pour la plupart des modèles [3]

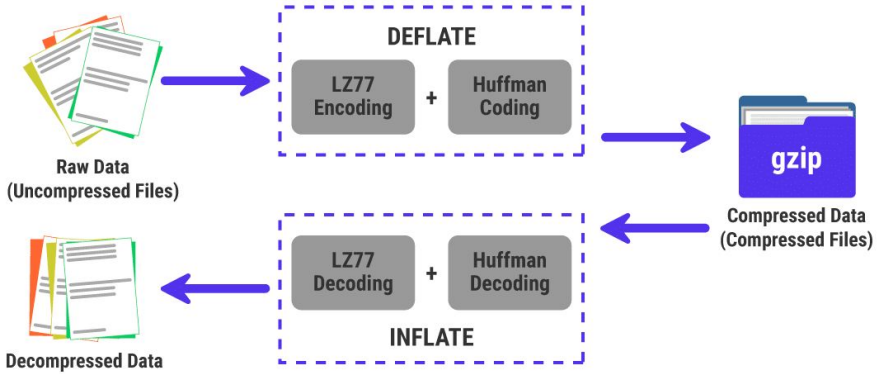


Notre optimisation

Algorithme GZip ↔ Huffman — **Technique de compression sans perte d'information**

Convertit chaque moyenne de compartiment (bucket) en une chaîne de bits

Étapes de l'algorithme



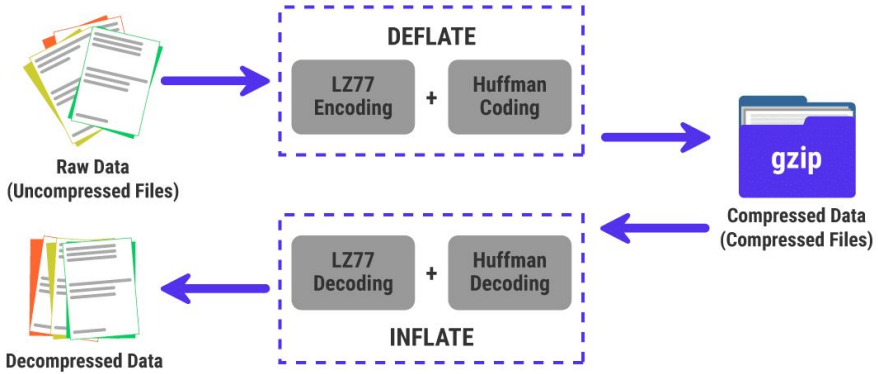
Notre optimisation

Algorithme GZip ↔ Huffman **Technique de compression sans perte d'information**

Convertit chaque moyenne de compartiment (bucket) en une chaîne de bits

Étapes de l'algorithme

- Algorithme LZ77 [4] : réduit redondance
  - Remplace les séquences répétées de données par une seule occurrence de cette séquence



Notre optimisation

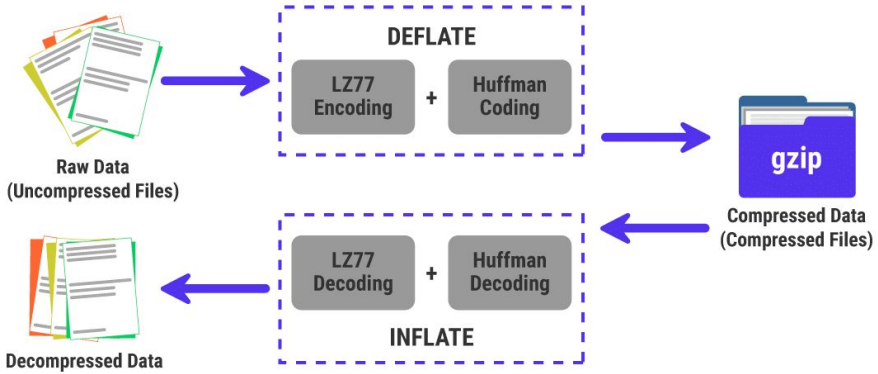
Algorithme GZip ↔ Huffman

Technique de compression sans perte d'information

Convertit chaque moyenne de compartiment (bucket) en une chaîne de bits

Étapes de l'algorithme

- Algorithme LZ77 [4] : réduit redondance
  - Remplace les séquences répétées de données par une seule occurrence de cette séquence
- Encodage de Huffman : Compression des données



Notre optimisation

Algorithme GZip ↔ Huffman

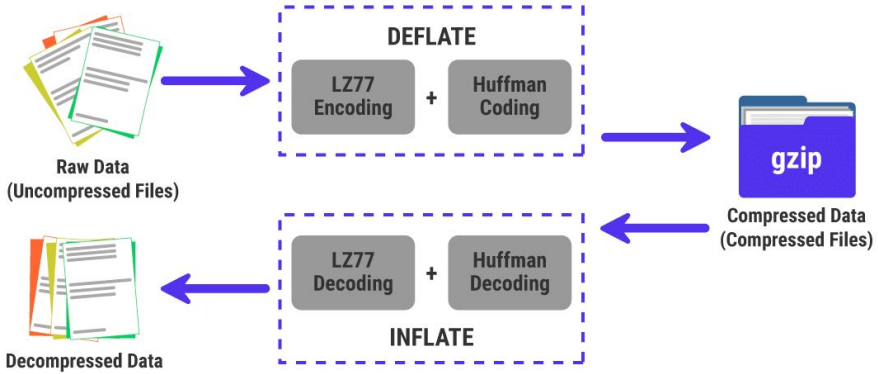
Technique de compression sans perte d'information

Convertit chaque moyenne de compartiment (bucket) en une chaîne de bits

Étapes de l'algorithme

- Algorithme LZ77 [4] : réduit redondance
  - Remplace les séquences répétées de données par une seule occurrence de cette séquence
- Encodage de Huffman : Compression des données

GZip fournit une solution plus performante que l'encodage de Huffman et plus facile à implémenter



Notre optimisation

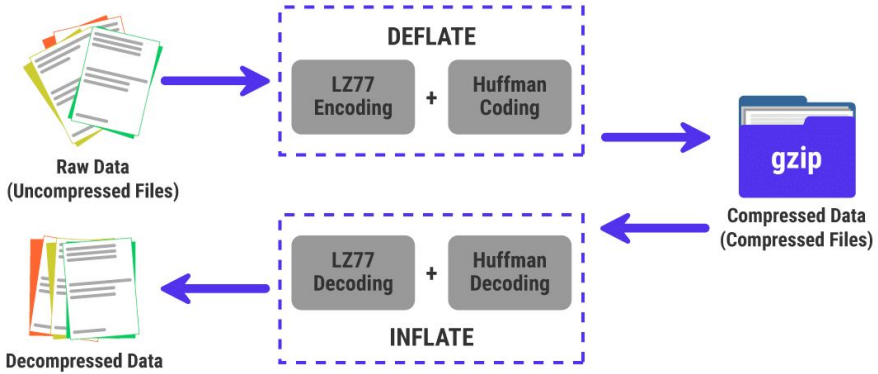
Algorithme GZip ↔ Huffman

Technique de compression sans perte d'information

Convertit chaque moyenne de compartiment (bucket) en une chaîne de bits

Étapes de l'algorithme

- Algorithme LZ77 [4] : réduit redondance
  - Remplace les séquences répétées de données par une seule occurrence de cette séquence
- Encodage de Huffman : Compression des données



GZip fournit une solution plus performante que l'encodage de Huffman et plus facile à implémenter

Dispose aussi d'une bibliothèque sur Python

Notre optimisation

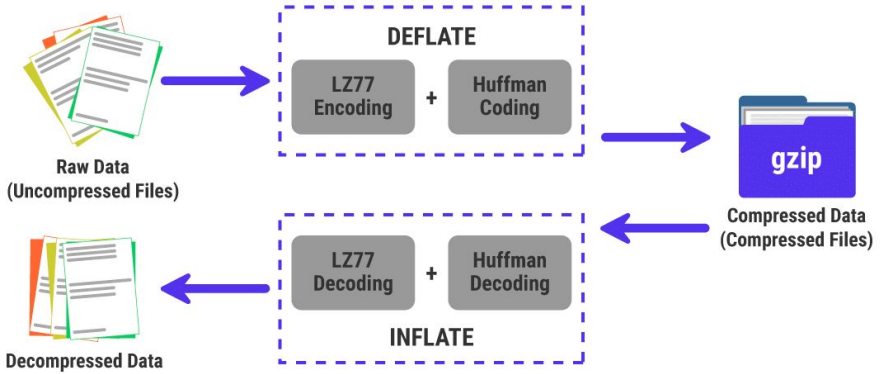
Algorithme GZip ↔ Huffman

Technique de compression sans perte d'information

Convertit chaque moyenne de compartiment (bucket) en une chaîne de bits

Étapes de l'algorithme

- Algorithme LZ77 [4] : réduit redondance
  - Remplace les séquences répétées de données par une seule occurrence de cette séquence
- Encodage de Huffman : Compression des données



GZip fournit une solution plus performante que l'encodage de Huffman et plus facile à implémenter

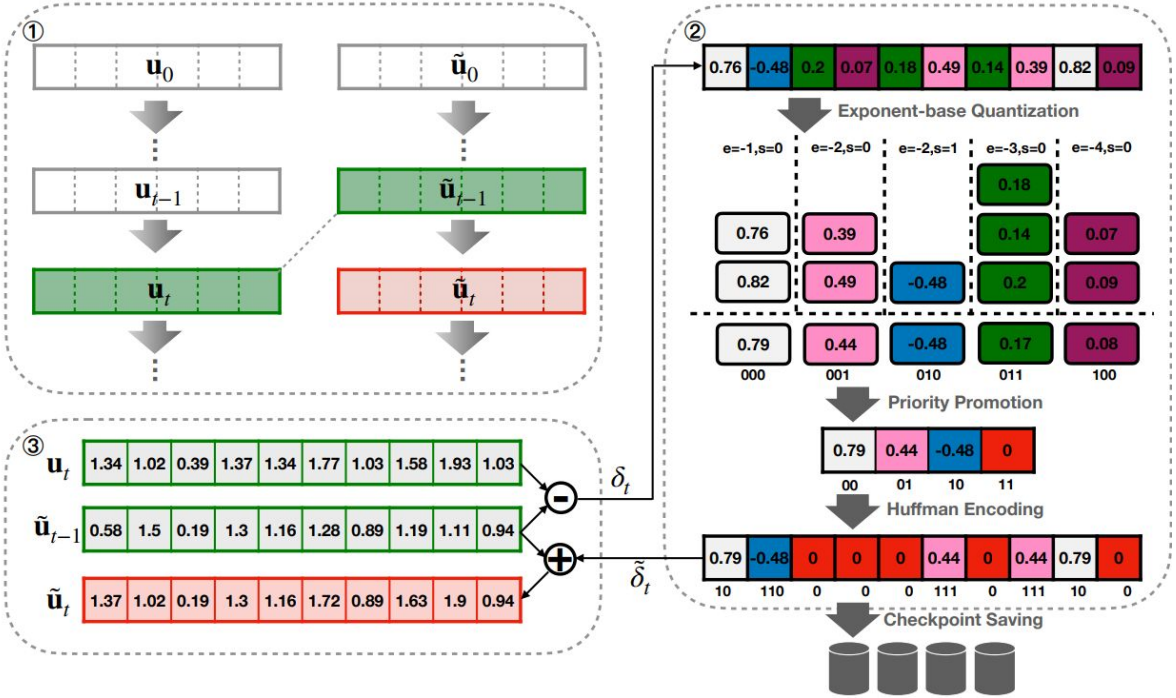
Dispose aussi d'une bibliothèque sur Python

Abus de langage : on va l'appeler LC-checkpoint

Notre optimisation

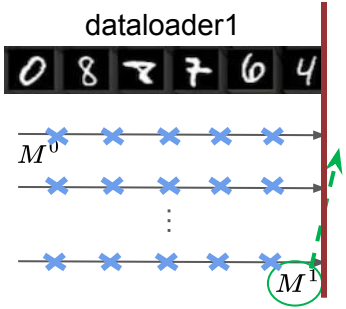
En sortie de GZip

Approximation des deltas qu'on ajoute à l'approximation de l'état du modèle, devenant la nouvelle approximation de l'état du modèle.



Démarche adoptée

Pour chaque ensemble de donnée (dataloader)



Optimisation en combinant deux techniques de compression

- LC-checkpoint (On Efficient Constructions of Checkpoints) [1]
- Delta-LoRA (Delta-LoRA: Fine-Tuning High-Rank Parameters with the Delta of Low-Rank Matrices) [2]

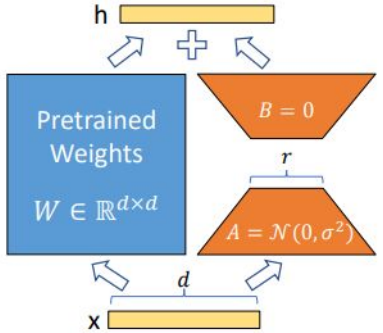
Au sein des techniques utilisées pour LC-checkpoint

- Codage de Huffman [3] ↔ Algorithme GZip

Delta-LoRA

LoRA - Low Rank Adaptation [5]

Méthode populaire d'adaptation (fine-tuning : lorsqu'on entraîne un modèle pré-entraîné sur un autre ensemble de donnée) ajoutant un nombre limité de paramètres en conservant la performance

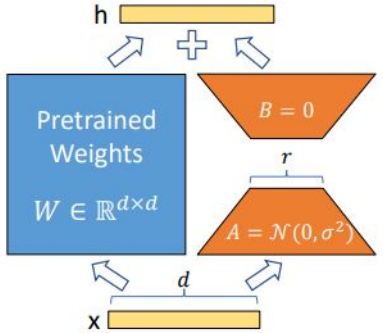


Delta-LoRA

LoRA - Low Rank Adaptation [5]

Méthode populaire d'adaptation (fine-tuning : lorsqu'on entraîne un modèle pré-entraîné sur un autre ensemble de donnée) ajoutant un nombre limité de paramètres en conservant la performance

Fonctionnement de LoRA



Delta-LoRA

LoRA - Low Rank Adaptation [5]

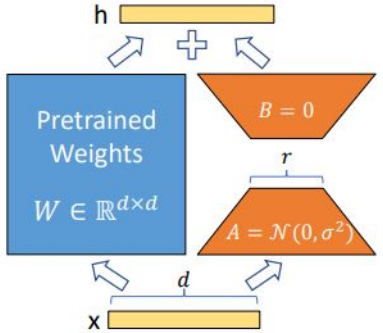
Méthode populaire d'adaptation (fine-tuning : lorsqu'on entraîne un modèle pré-entraîné sur un autre ensemble de donnée) ajoutant un nombre limité de paramètres en conservant la performance

Fonctionnement de LoRA

Suppose que le changement de poids du modèle ΔW a une faible dimension intrinsèque pour mettre à jour la matrice de poids du modèle pré-entraîné figé W_0 de taille $d \times k$

$$W = W_0 + \Delta W \quad \text{avec} \quad \Delta W = A \times B$$

$$A \in R^{d \times r}$$
$$B \in R^{r \times k}$$
$$r \ll \min(d, k)$$



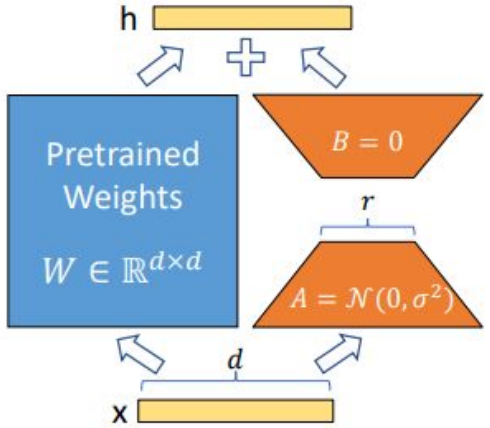
Fonctionnement de LoRA

L'adaptation se fait uniquement sur les matrices A et B, qui contiennent beaucoup moins de paramètres que W_0

W_0 est figé

$d \times k$	$A \in R^{d \times r}$ $B \in R^{r \times k}$ $r \ll \min(d, k)$

$$W = W_0 + \Delta W \quad \text{avec} \quad \Delta W = A \times B$$



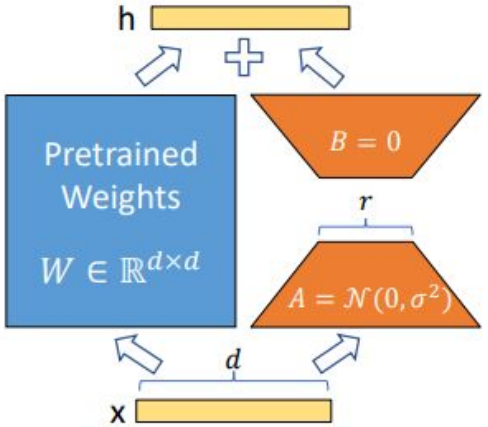
Fonctionnement de LoRA

L'adaptation se fait uniquement sur les matrices A et B, qui contiennent beaucoup moins de paramètres que W_0

W_0 est figé

$d \times k$	$A \in R^{d \times r}$ $B \in R^{r \times k}$ $r \ll \min(d, k)$

$$W = W_0 + \Delta W \quad \text{avec} \quad \Delta W = A \times B$$



Remarque :

- Application sur **toutes couches denses**
- Montré expérimentalement qu'un **rang de 8** est un bon équilibre entre compression et performance pour des grands modèles comme LLaMA.

Delta-LoRA : Version modifiée de LoRA

Optimise les **paramètres de haute dimension**

Delta-LoRA : Version modifiée de LoRA

Optimise les **paramètres de haute dimension**

Fonctionnement de Delta-LoRA

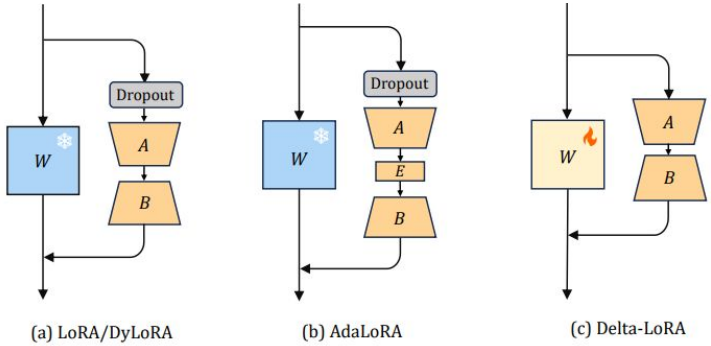
Delta-LoRA : Version modifiée de LoRA

Optimise les **paramètres de haute dimension**

Fonctionnement de Delta-LoRA

$$W^{(t+1)} = W^{(t)} + \Delta AB \quad \text{avec} \quad \Delta AB = A^{(t+1)}B^{(t+1)} - A^{(t)}B^{(t)}$$

Mettre à jour les matrices de faible rang A et B et également propager l'adaptation aux poids pré-entraînés W via des mises à jour utilisant la différence du produit de deux matrices consécutives de faible rang $A^{(t+1)}B^{(t+1)} - A^{(t)}B^{(t)}$



Approche proposée

Checkpointing Efficace pour les réseaux de neurones profonds

Intégrer les forces respectives de LC-checkpoint et Delta-LoRA dans une seule méthode de compression optimisée

Approche proposée

Checkpointing Efficace pour les réseaux de neurones profonds

Intégrer les forces respectives de LC-checkpoint et Delta-LoRA dans une seule méthode de compression optimisée

Fonctionnement

Approche proposée

Checkpointing Efficace pour les réseaux de neurones profonds

Intégrer les forces respectives de LC-checkpoint et Delta-LoRA dans une seule méthode de compression optimisée

Fonctionnement

Applique au modèle pré-entraîné auquel on y ajoute des couches de Delta-LoRA, les différentes techniques utilisées dans LC-checkpoint :

Approche proposée

Checkpointing Efficace pour les réseaux de neurones profonds

Intégrer les forces respectives de LC-checkpoint et Delta-LoRA dans une seule méthode de compression optimisée

Fonctionnement

Applique au modèle pré-entraîné auquel on y ajoute des couches de Delta-LoRA, les différentes techniques utilisées dans LC-checkpoint :

- calcul des deltas via l'**encodage delta**

Approche proposée

Checkpointing Efficace pour les réseaux de neurones profonds

Intégrer les forces respectives de LC-checkpoint et Delta-LoRA dans une seule méthode de compression optimisée

Fonctionnement

Applique au modèle pré-entraîné auquel on y ajoute des couches de Delta-LoRA, les différentes techniques utilisées dans LC-checkpoint :

- calcul des deltas via l'**encodage delta**
- application à chaque delta la **quantification à base exponentielle** (Exponent-based quantization)

Approche proposée

Checkpointing Efficace pour les réseaux de neurones profonds

Intégrer les forces respectives de LC-checkpoint et Delta-LoRA dans une seule méthode de compression optimisée

Fonctionnement

Applique au modèle pré-entraîné auquel on y ajoute des couches de Delta-LoRA, les différentes techniques utilisées dans LC-checkpoint :

- calcul des deltas via l'**encodage delta**
- application à chaque delta la **quantification à base exponentielle** (Exponent-based quantization)
- **promotion de priorité** (Priority Promotion) pour donner plus d'importance aux informations plus significatifs

Approche proposée

Checkpointing Efficace pour les réseaux de neurones profonds

Intégrer les forces respectives de LC-checkpoint et Delta-LoRA dans une seule méthode de compression optimisée

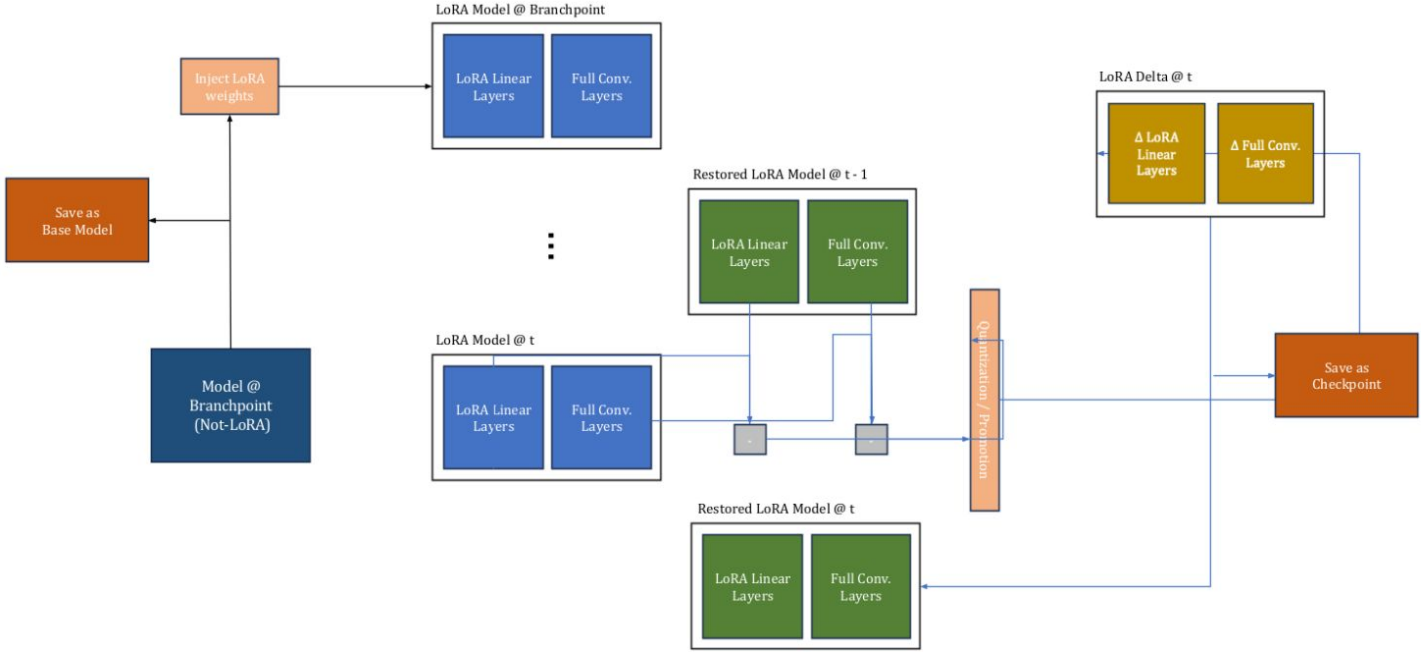
Fonctionnement

Applique au modèle pré-entraîné auquel on y ajoute des couches de Delta-LoRA, les différentes techniques utilisées dans LC-checkpoint :

- calcul des deltas via l'**encodage delta**
- application à chaque delta la **quantification à base exponentielle** (Exponent-based quantization)
- **promotion de priorité** (Priority Promotion) pour donner plus d'importance aux informations plus significatifs
- **algorithme GZip**

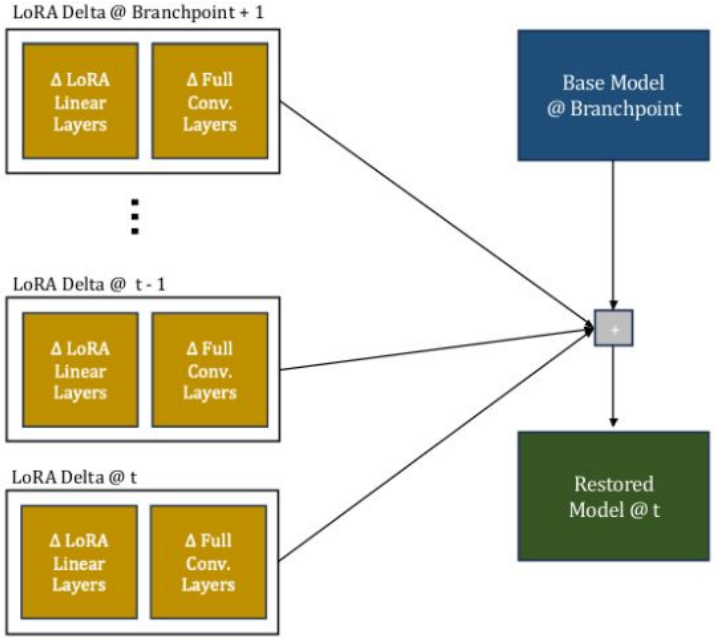
Approche proposée

Schéma du Checkpointing Efficace pour les réseaux de neurones profonds



Approche proposée

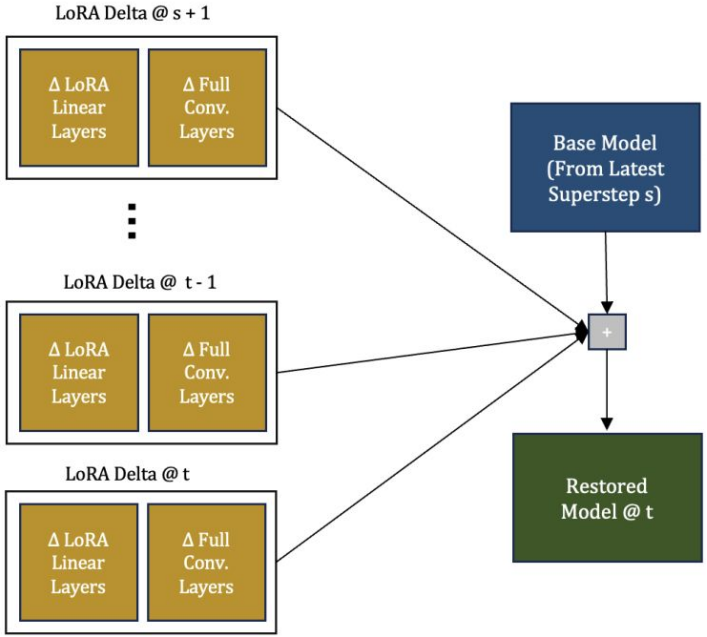
Restauration en cas de panne



Récupération du modèle au dernier branchpoint puis on décompresse tous les checkpoints jusqu'au dernier précédant la panne

Approche proposée

Optimisation en temps pour la restauration



Mettre à jour le modèle à des itérations de manière périodique, nommé **superstep (s)**

A tous les supersteps, le modèle est sauvegardé (nommé **full snapshot**)

<u>En cas de panne :</u>
Modèle au dernier superstep et s'il y a des checkpoints (les deltas) après ce superstep, on les ajoute jusqu'au dernier checkpoint précédant la panne

Résultats

Objectif du stage

Objectif du stage

Étendre les travaux menés par l'ancien stagiaire :

Résultats

Objectif du stage

Étendre les travaux menés par l'ancien stagiaire :

- Reproduire ses résultats

Objectif du stage

Étendre les travaux menés par l'ancien stagiaire :

- Reproduire ses résultats
- Utiliser des modèles avec plus de paramètres

Objectif du stage

Étendre les travaux menés par l'ancien stagiaire :

- Reproduire ses résultats
- Utiliser des modèles avec plus de paramètres
- Utiliser des ensembles d'entraînement plus complexes

Objectif du stage

Étendre les travaux menés par l'ancien stagiaire :

- Reproduire ses résultats
- Utiliser des modèles avec plus de paramètres
- Utiliser des ensembles d'entraînement plus complexes
- Considérer l'empoisonnement des données

Objectif du stage

Étendre les travaux menés par l'ancien stagiaire :

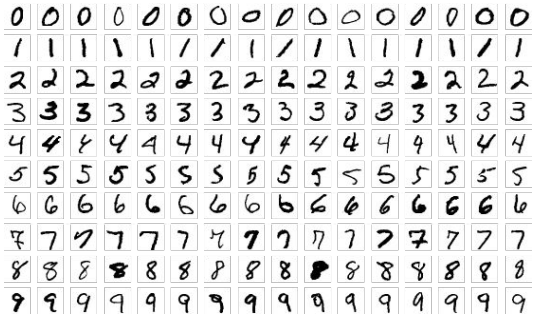
- Reproduire ses résultats
- Utiliser des modèles avec plus de paramètres
- Utiliser des ensembles d'entraînement plus complexes
- Considérer l'empoisonnement des données
- Faire une étude du rang pour Delta-LoRA

Résultats

Résultats préliminaires

Résultats préliminaires

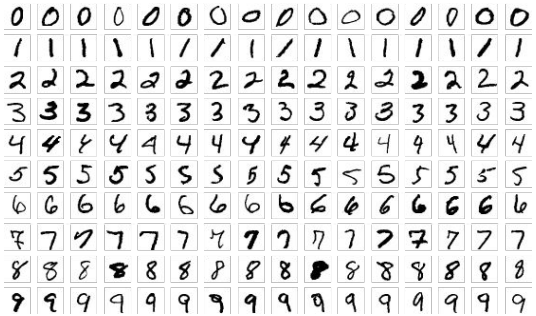
Intérêt montré en entraînant sur les 1000 premières images de l'ensemble d'entraînement de MNIST [6], validant sur les 1000 secondes images de l'ensemble d'entraînement de MNIST, obtenant les résultats suivants pour LeNet-5 [8], AlexNet[9] et VGG-16 [10]



Résultats préliminaires

Intérêt montré en entraînant sur les 1000 premières images de l'ensemble d'entraînement de MNIST [6], validant sur les 1000 secondes images de l'ensemble d'entraînement de MNIST, obtenant les résultats suivants pour LeNet-5 [8], AlexNet[9] et VGG-16 [10]

Model	Mechanism	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
AlexNet	LC	808.35%	87.629%	98.4% / 98.7% (-0.3%)
	LC + dLoRA	25995.409%	99.615%	95.3% / 98.7% (-3.4%)
VGG-16	LC	813.74%	87.711%	99.1% / 99.4% (-0.3%)
	LC + dLoRA	4188.412%	97.612%	98.4% / 99.4% (-1.0%)
LeNet	LC	537.584%	81.39%	95.9% / 95.9% (-0.0%)
	LC + dLoRA	1889.2869%	94.707%	93.8% / 95.9% (-2.1%)



Résultats

Reproduction des résultats

Résultats

Reproduction des résultats

Il manquait des fichiers essentiels à l'exécution du code lorsque cet étudiant m'a transféré son code

↓

Réimplémenter les parties manquantes

Résultats

Reproduction des résultats

Il manquait des fichiers essentiels à l'exécution du code lorsque cet étudiant m'a transféré son code

↓

Réimplémenter les parties manquantes

Model	Mechanism	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
AlexNet	LC	813.32%	87.705%	98.2% / 99.0% (-0.8%)
	LC + dLoRA	28865.393%	99.654%	96.2% / 99.0% (-3.8%)
VGG-16	LC	813.705%	87.711%	99.1% / 99.4% (-0.3%)
	LC + dLoRA	4185.214%	97.611%	98.1% / 99.4% (-1.3%)
LeNet	LC	562.256%	82.215%	96.1% / 96.1% (-0.0%)
	LC + dLoRA	1885.7610%	94.697%	91.2% / 96.1% (-4.9%)

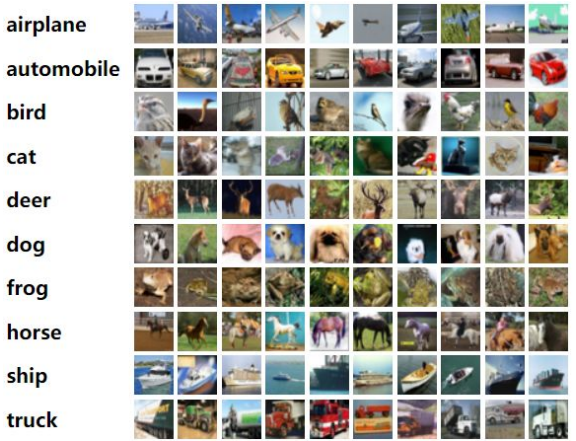
Résultats de la reproduction

Model	Mechanism	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
AlexNet	LC	808.35%	87.629%	98.4% / 98.7% (-0.3%)
	LC + dLoRA	25995.409%	99.615%	95.3% / 98.7% (-3.4%)
VGG-16	LC	813.74%	87.711%	99.1% / 99.4% (-0.3%)
	LC + dLoRA	4188.412%	97.612%	98.4% / 99.4% (-1.0%)
LeNet	LC	537.584%	81.39%	95.9% / 95.9% (-0.0%)
	LC + dLoRA	1889.2869%	94.707%	93.8% / 95.9% (-2.1%)

Résultats de l'ancien stagiaire

Extension des résultats sur CIFAR-10

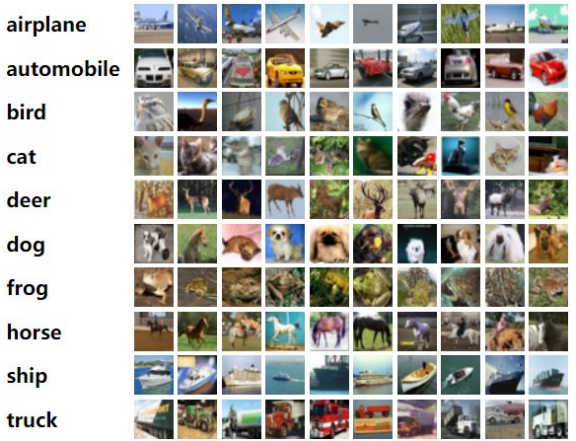
Étendre les résultats précédents en entraînant sur les 1000 premières images de l'ensemble d'entraînement de CIFAR-10 [7], validant sur les 1000 secondes images de l'ensemble d'entraînement de CIFAR-10, obtenant les résultats suivants



Extension des résultats sur CIFAR-10

Étendre les résultats précédents en entraînant sur les 1000 premières images de l'ensemble d'entraînement de CIFAR-10 [7], validant sur les 1000 secondes images de l'ensemble d'entraînement de CIFAR-10, obtenant les résultats suivants

Model	Mechanism	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
AlexNet	LC	634.431%	84.238%	99.6% / 99.6% (-0.0%)
	LC + dLoRA	23491.97%	99.574%	94.5% / 99.6% (-5.1%)
VGG-16	LC	814.574%	87.724%	100.0% / 100.0% (-0.0%)‡
	LC + dLoRA	4756.479%	97.898%	98.2% / 100.0% (-1.8%)‡
LeNet	LC	415.728%	81.061%	98.4% / 98.4% (-0.0%)
	LC + dLoRA	1698.463%	94.112%	86.0% / 98.4% (-12.4%)



Résultats

Extension avec VGG-16 Full, ResNet-50 [11] et Vision Transformer [12]

Pour des raisons de temps, maintient de la même configuration sur MNIST tant pour l'entraînement que pour l'évaluation des modèles garantissant que les résultats soient comparables.

Extension avec VGG-16 Full, ResNet-50 [11] et Vision Transformer [12]

Pour des raisons de temps, maintient de la même configuration sur MNIST tant pour l'entraînement que pour l'évaluation des modèles garantissant que les résultats soient comparables.

Model	Mechanism	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
VGG-16†	LC	771.56%	87.04%	99.5% / 99.5% (-0.0%)
	LC + dLoRA	6160.776%	98.378%	98.9% / 99.5% (-0.6%)
ResNet-50	LC	760.59%	86.85%	100.0% / 100.0% (-0.0%)‡
	LC + dLoRA	761.939%	86.876%	99.3% / 100.0% (-0.7%)‡
ViT-Tiny	LC	244.15%	59.04%	76.0% / 76.0% (-0.0%)
	LC + dLoRA	223.928%	55.343%	75.6% / 76.0% (-0.4%)
AlexNet	LC	813.32%	87.71%	98.2% / 99.0% (-0.8%)
	LC + dLoRA	28865.393%	99.654%	96.2% / 99.0% (-2.8%)
VGG-16⋆	LC	811.67%	87.68%	99.9% / 99.9% (-0.0%)
	LC + dLoRA	4562.891%	97.808%	99.1% / 99.9% (-0.8%)
LeNet	LC	562.26%	82.22%	96.1% / 96.1% (-0.0%)
	LC + dLoRA	1885.761%	94.697%	91.2% / 96.1% (-4.9%)

† VGG-16 Full Version
⋆ VGG-16 Lite Version

Résultats

Extension avec VGG-16 Full, ResNet-50 [11] et Vision Transformer [12]

Pour des raisons de temps, maintient de la même configuration sur MNIST tant pour l'entraînement que pour l'évaluation des modèles garantissant que les résultats soient comparables.

Model	Mechanism	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
VGG-16†	LC	771.56%	87.04%	99.5% / 99.5% (-0.0%)
	LC + dLoRA	6160.776%	98.378%	98.9% / 99.5% (-0.6%)
ResNet-50	LC	760.59%	86.85%	100.0% / 100.0% (-0.0%)‡
	LC + dLoRA	761.939%	86.876%	99.3% / 100.0% (-0.7%)‡
ViT-Tiny	LC	244.15%	59.04%	76.0% / 76.0% (-0.0%)
	LC + dLoRA	223.928%	55.343%	75.6% / 76.0% (-0.4%)
AlexNet	LC	813.32%	87.71%	98.2% / 99.0% (-0.8%)
	LC + dLoRA	28865.393%	99.654%	96.2% / 99.0% (-2.8%)
VGG-16⋆	LC	811.67%	87.68%	99.9% / 99.9% (-0.0%)
	LC + dLoRA	4562.891%	97.808%	99.1% / 99.9% (-0.8%)
LeNet	LC	562.26%	82.22%	96.1% / 96.1% (-0.0%)
	LC + dLoRA	1885.761%	94.697%	91.2% / 96.1% (-4.9%)

† VGG-16 Full Version
⋆ VGG-16 Lite Version

La version Lite est la version sur un seul canal de couleur

La version Full est la version du papier original conçu sur ImageNet

Résultats

Extension avec VGG-16 Full, ResNet-50 [11] et Vision Transformer [12]

Pour des raisons de temps, maintient de la même configuration sur MNIST tant pour l'entraînement que pour l'évaluation des modèles garantissant que les résultats soient comparables.

Model	Mechanism	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
VGG-16†	LC	771.56%	87.04%	99.5% / 99.5% (-0.0%)
	LC + dLoRA	6160.776%	98.378%	98.9% / 99.5% (-0.6%)
ResNet-50	LC	760.59%	86.85%	100.0% / 100.0% (-0.0%)‡
	LC + dLoRA	761.939%	86.876%	99.3% / 100.0% (-0.7%)‡
ViT-Tiny	LC	244.15%	59.04%	76.0% / 76.0% (-0.0%)
	LC + dLoRA	223.928%	55.343%	75.6% / 76.0% (-0.4%)
AlexNet	LC	813.32%	87.71%	98.2% / 99.0% (-0.8%)
	LC + dLoRA	28865.393%	99.654%	96.2% / 99.0% (-2.8%)
VGG-16⋆	LC	811.67%	87.68%	99.9% / 99.9% (-0.0%)
	LC + dLoRA	4562.891%	97.808%	99.1% / 99.9% (-0.8%)
LeNet	LC	562.26%	82.22%	96.1% / 96.1% (-0.0%)
	LC + dLoRA	1885.761%	94.697%	91.2% / 96.1% (-4.9%)

† VGG-16 Full Version
⋆ VGG-16 Lite Version

La version Lite est la version sur un seul canal de couleur

La version Full est la version du papier original conçu sur ImageNet

Résultats

Extension avec VGG-16 Full, ResNet-50 [11] et Vision Transformer [12]

Pour des raisons de temps, maintient de la même configuration sur MNIST tant pour l'entraînement que pour l'évaluation des modèles garantissant que les résultats soient comparables.

Model	Mechanism	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
VGG-16†	LC	771.56%	87.04%	99.5% / 99.5% (-0.0%)
	LC + dLoRA	6160.776%	98.378%	98.9% / 99.5% (-0.6%)
ResNet-50	LC	760.59%	86.85%	100.0% / 100.0% (-0.0%)‡
	LC + dLoRA	761.939%	86.876%	99.3% / 100.0% (-0.7%)‡
ViT-Tiny	LC	244.15%	59.04%	76.0% / 76.0% (-0.0%)
	LC + dLoRA	223.928%	55.343%	75.6% / 76.0% (-0.4%)
AlexNet	LC	813.32%	87.71%	98.2% / 99.0% (-0.8%)
	LC + dLoRA	28865.393%	99.654%	96.2% / 99.0% (-2.8%)
VGG-16⋆	LC	811.67%	87.68%	99.9% / 99.9% (-0.0%)
	LC + dLoRA	4562.891%	97.808%	99.1% / 99.9% (-0.8%)
LeNet	LC	562.26%	82.22%	96.1% / 96.1% (-0.0%)
	LC + dLoRA	1885.761%	94.697%	91.2% / 96.1% (-4.9%)

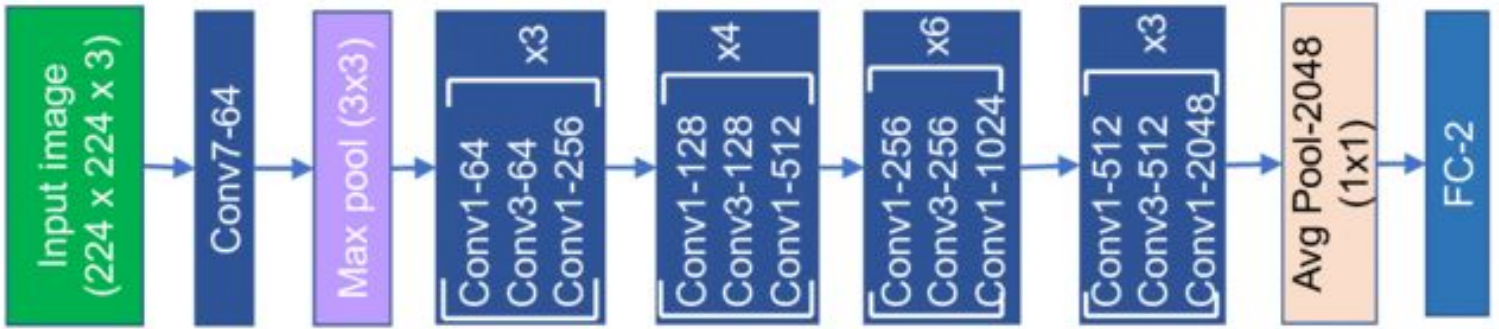
† VGG-16 Full Version
⋆ VGG-16 Lite Version

La version Lite est la version sur un seul canal de couleur

La version Full est la version du papier original conçu sur ImageNet

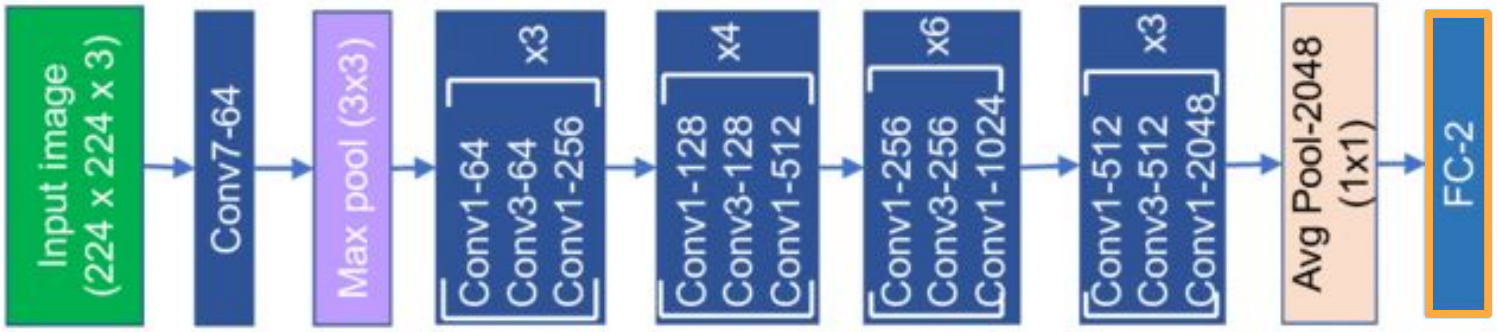
Résultats

Architecture d'un ResNet-50



Résultats

Architecture d'un ResNet-50



Etudes des couches d'application de Delta-LoRA sur Vision Transformer (ViT)

En raison de la compression insuffisante obtenue avec le ViT-Tiny sur MNIST, nous avons décidé de développer un modèle ViT-Small (ViT-S), qui est une configuration plus avancée et proche du plus petit modèle de ViT du papier original.

Résultats

Etudes des couches d'application de Delta-LoRA sur Vision Transformer (ViT)

En raison de la compression insuffisante obtenue avec le ViT-Tiny sur MNIST, nous avons décidé de développer un modèle ViT-Small (ViT-S), qui est une configuration plus avancée et proche du plus petit modèle de ViT du papier original.

Configuration de ViT-S :

- n_heads = 8
- n_blocks = 8
- hidden_dim = 512
- mlp_size = 2048

Résultats

Etudes des couches d'application de Delta-LoRA sur Vision Transformer (ViT)

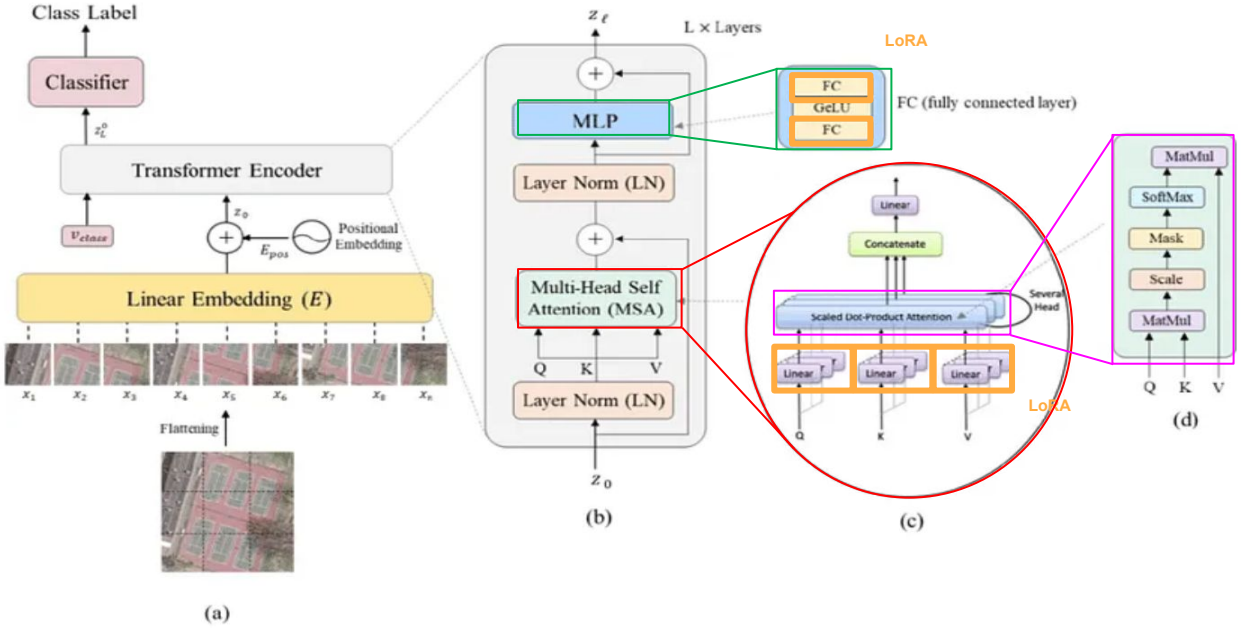
En raison de la compression insuffisante obtenue avec le ViT-Tiny sur MNIST, nous avons décidé de développer un modèle ViT-Small (ViT-S), qui est une configuration plus avancée et proche du plus petit modèle de ViT du papier original.

Configuration de ViT-S :
- n_heads = 8
- n_blocks = 8
- hidden_dim = 512
- mlp_size = 2048

Configuration de ViT-T :
- n_heads = 2
- n_blocks = 2
- hidden_dim = 8
- mlp_size = 32

Résultats

Architecture du Vision Transformer



Résultats

Etudes des couches d'application de Delta-LoRA sur Vision Transformer (ViT)

En raison de la compression insuffisante obtenue avec le ViT-Tiny sur MNIST, nous avons décidé de développer un modèle ViT-Small (ViT-S), qui est une configuration plus avancée et proche du plus petit modèle de ViT du papier original.

Configuration de ViT-S :
- n_heads = 8
- n_blocks = 8
- hidden_dim = 512
- mlp_size = 2048

Déterminer quelles couches bénéficient le plus de la compression Delta-LoRA

Résultats

Etudes des couches d'application de Delta-LoRA sur Vision Transformer (ViT)

En raison de la compression insuffisante obtenue avec le ViT-Tiny sur MNIST, nous avons décidé de développer un modèle ViT-Small (ViT-S), qui est une configuration plus avancée et proche du plus petit modèle de ViT du papier original.

Configuration de ViT-S :
- n_heads = 8
- n_blocks = 8
- hidden_dim = 512
- mlp_size = 2048

Déterminer quelles couches bénéficient le plus de la compression Delta-LoRA

Model	Mechanism	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
ViT-S (MSAxMLP)	LC	764.484%	86.919%	97.3% / 97.3% (-0.0%)
	LC + dLoRA	12644.316%	99.209%	91.4% / 97.3% (-5.9%)
ViT-S (MSA)	LC	767.055%	86.963%	96.7% / 96.7% (-0.0%)
	LC + dLoRA	790.168%	87.344%	91.4% / 96.7% (-5.3%)
ViT-S (MLP)	LC	761.956%	86.876%	97.0% / 97.2% (-0.2%)
	LC + dLoRA	8215.592%	98.783%	96.7% / 97.2% (-0.5%)

Résultats

Etudes des couches d'application de Delta-LoRA sur Vision Transformer (ViT)

En raison de la compression insuffisante obtenue avec le ViT-Tiny sur MNIST, nous avons décidé de développer un modèle ViT-Small (ViT-S), qui est une configuration plus avancée et proche du plus petit modèle de ViT du papier original.

Configuration de ViT-S :
- n_heads = 8
- n_blocks = 8
- hidden_dim = 512
- mlp_size = 2048

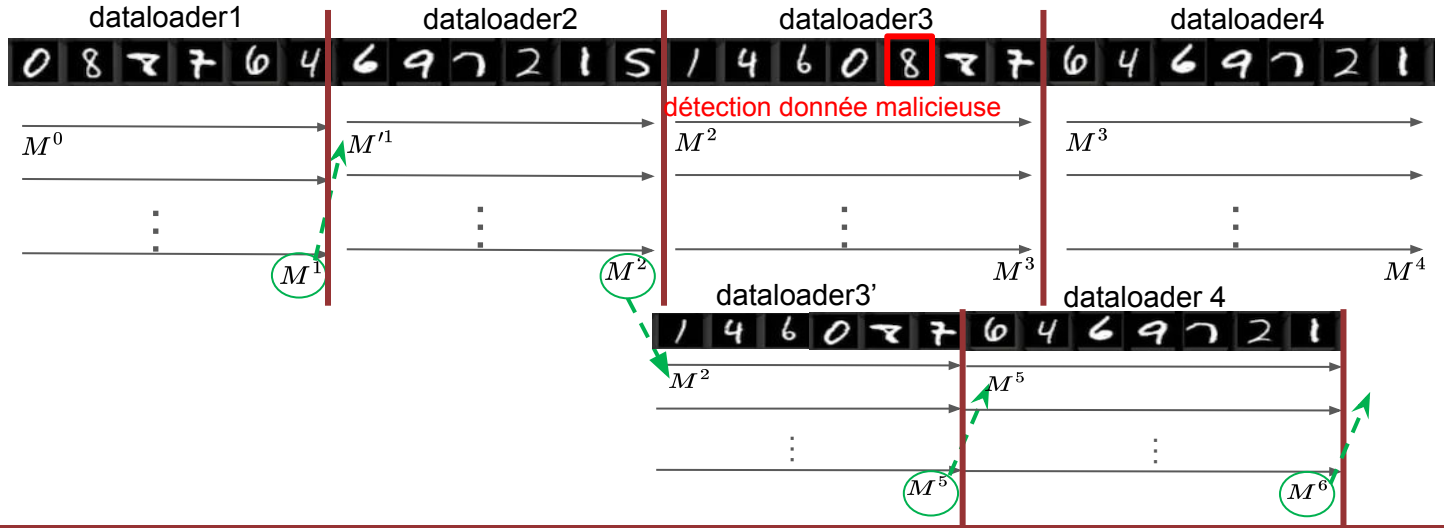
Déterminer quelles couches bénéficient le plus de la compression Delta-LoRA

Model	Mechanism	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
ViT-S (MSAxMLP)	LC	764.484%	86.919%	97.3% / 97.3% (-0.0%)
	LC + dLoRA	12644.316%	99.209%	91.4% / 97.3% (-5.9%)
ViT-S (MSA)	LC	767.055%	86.963%	96.7% / 96.7% (-0.0%)
	LC + dLoRA	790.168%	87.344%	91.4% / 96.7% (-5.3%)
ViT-S (MLP)	LC	761.956%	86.876%	97.0% / 97.2% (-0.2%)
	LC + dLoRA	8215.592%	98.783%	96.7% / 97.2% (-0.5%)

Taux de compression nettement plus élevé lorsque Delta LoRA appliquée sur MSAxMLP par rapport à son application sur MSA et sur MLP

Résultats

Apprentissage incrémental



Est-ce que notre schéma peut se développer à grande échelle considérant l'apprentissage incrémental ?

Résultats

Apprentissage incrémental

Première étude : toutes les classes de l'ensemble d'intérêt sont représentées dans chaque sous-ensemble, en considérant 3 configurations :

Apprentissage incrémental

Première étude : toutes les classes de l'ensemble d'intérêt sont représentées dans chaque sous-ensemble, en considérant 3 configurations :

- 4 dataloaders avec chacun 25% de l'ensemble d'entraînement d'intérêt

Apprentissage incrémental

Première étude : toutes les classes de l'ensemble d'intérêt sont représentées dans chaque sous-ensemble, en considérant 3 configurations :

- 4 dataloaders avec chacun 25% de l'ensemble d'entraînement d'intérêt
- 20 dataloaders avec chacun 5% de l'ensemble d'entraînement d'intérêt

Résultats

Apprentissage incrémental

Première étude : toutes les classes de l'ensemble d'intérêt sont représentées dans chaque sous-ensemble, en considérant 3 configurations :

- 4 dataloaders avec chacun 25% de l'ensemble d'entraînement d'intérêt
- 20 dataloaders avec chacun 5% de l'ensemble d'entraînement d'intérêt
- 80 dataloaders avec chacun 1.25% de l'ensemble d'entraînement d'intérêt

Apprentissage incrémental

Première étude : toutes les classes de l'ensemble d'intérêt sont représentées dans chaque sous-ensemble, en considérant 3 configurations :

- 4 dataloaders avec chacun 25% de l'ensemble d'entraînement d'intérêt
- 20 dataloaders avec chacun 5% de l'ensemble d'entraînement d'intérêt
- 80 dataloaders avec chacun 1.25% de l'ensemble d'entraînement d'intérêt

Plus la granularité est importante, plus cela se rapproche du scénario que l'on recherche

Apprentissage incrémental : Résultats pour LeNet-5 sur MNIST

Split	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
4*25% (15k-15k-15k-15k)	LC	888.872%	88.75%	98.87%/98.85% (+0.02%)
	LC + dLoRA	2481.857%	99.209%	92.09%/98.85% (-6.76%)
20*5% (20* 3k)	LC	723.688%	86.182%	98.39%/98.37% (+0.02%)
	LC + dLoRA	2270.412%	95.596%	92.63%/98.37% (-5.74%)
80* 1.25% (80* 750)	LC	885.524%	88.71%	97.65%/97.75% (-0.1%)
	LC + dLoRA	2478.693%	95.97%	94.08%/97.75% (-3,67%)

Résultats

Apprentissage incrémental : Résultats pour LeNet-5 sur MNIST

Split	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
4*25% (15k-15k-15k-15k)	LC	888.872%	88.75%	98.87%/98.85% (+0.02%)
	LC + dLoRA	2481.857%	99.209%	92.09%/98.85% (-6.76%)
20*5% (20* 3k)	LC	723.688%	86.182%	98.39%/98.37% (+0.02%)
	LC + dLoRA	2270.412%	95.596%	92.63%/98.37% (-5.74%)
80* 1.25% (80* 750)	LC	885.524%	88.71%	97.65%/97.75% (-0.1%)
	LC + dLoRA	2478.693%	95.97%	94.08%/97.75% (-3,67%)

Le taux de compression ne diffère pas tant que cela en fonction des configurations

Résultats

Apprentissage incrémental : Résultats pour LeNet-5 sur MNIST

Split	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
4*25% (15k-15k-15k-15k)	LC	888.872%	88.75%	98.87%/98.85% (+0.02%)
	LC + dLoRA	2481.857%	99.209%	92.09%/98.85% (-6.76%)
20*5% (20* 3k)	LC	723.688%	86.182%	98.39%/98.37% (+0.02%)
	LC + dLoRA	2270.412%	95.596%	92.63%/98.37% (-5.74%)
80* 1.25% (80* 750)	LC	885.524%	88.71%	97.65%/97.75% (-0.1%)
	LC + dLoRA	2478.693%	95.97%	94.08%/97.75% (-3.67%)

Le taux de compression ne diffère pas tant que cela en fonction des configurations

La perte de la performance est moins importante pour 80*1.25% que pour les autres.

Résultats

Apprentissage incrémental : Résultats pour LeNet-5 sur MNIST

Split	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
4*25% (15k-15k-15k-15k)	LC	888.872%	88.75%	98.87%/98.85% (+0.02%)
	LC + dLoRA	2481.857%	99.209%	92.09%/98.85% (-6.76%)
20*5% (20* 3k)	LC	723.688%	86.182%	98.39%/98.37% (+0.02%)
	LC + dLoRA	2270.412%	95.596%	92.63%/98.37% (-5.74%)
80* 1.25% (80* 750)	LC	885.524%	88.71%	97.65%/97.75% (-0.1%)
	LC + dLoRA	2478.693%	95.97%	94.08%/97.75% (-3.67%)

Le taux de compression ne diffère pas tant que cela en fonction des configurations

La perte de la performance est moins importante pour 80*1.25% que pour les autres

Le modèle apprend mieux lorsque la taille du slot des nouvelles données arrivant est faible

Apprentissage incrémental : Résultats pour ViT-B/16 sur CIFAR-10

Split	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
4*25% (4* 12.5k)	LC	591.368%	83.083%	98.70%/98.74% (-0.04%)
	LC + dLoRA	8173.368%	98.777%	97.79%/98.74% (-0.95%)

Obtention d'une faible perte de la performance

Résultats

Apprentissage incrémental : Résultats pour ViT-B/16 sur CIFAR-10

Split	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
4*25% (4* 12.5k)	LC	591.368%	83.083%	98.70%/98.74% (-0.04%)
	LC + dLoRA	8173.368%	98.777%	97.79%/98.74% (-0.95%)

Obtention d'une faible perte de la performance, tout en gardant un taux de compression important, même dans le cas de quatre dataloaders de 25%

Apprentissage incrémental : Résultats sur le rang

Apprentissage incrémental : Résultats sur le rang

Le rang optimal pouvant évoluer d'un dataloader à l'autre, une analyse du rang a été considérée.

Apprentissage incrémental : Résultats sur le rang

Le rang optimal pouvant évoluer d'un dataloader à l'autre, une analyse du rang a été considérée

Une étude d'un rang constant, optimal pour tous les dataloaders
a été menée sur ViT-B/16 pré-entrainé sur ImageNet-1k et adapté à CIFAR-10

Résultats

Apprentissage incrémental : Résultats sur le rang

Rank	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
4	LC	554.215%	81.956%	98.47%/98.5% (-0.03%)
	LC + dLoRA	8675.55%	98.847%	97.53%/98.5% (-0.91%)
3	LC	959.664%	89.58%	98.51%/98.51% (-0.00%)
	LC + dLoRA	8810.807%	98.865%	97.53%/98.51% (-0.92%)
2	LC	553.623%	81.937%	98.49%/98.49% (-0.00%)
	LC + dLoRA	8950.415%	98.883%	97.48%/98.49% (-1.01%)
1	LC	959.664%	89.58%	98.33%/98.51% (-0.18%)
	LC + dLoRA	9094.627%	98.9%	97.56%/98.51% (-0.95%)

Performance similaire pour les différents rangs

Résultats

Apprentissage incrémental : Résultats sur le rang

Rank	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
4	LC	554.215%	81.956%	98.47%/98.5% (-0.03%)
	LC + dLoRA	8675.55%	98.847%	97.53%/98.5% (-0.91%)
3	LC	959.664%	89.58%	98.51%/98.51% (-0.00%)
	LC + dLoRA	8810.807%	98.865%	97.53%/98.51% (-0.92%)
2	LC	553.623%	81.937%	98.49%/98.49% (-0.00%)
	LC + dLoRA	8950.415%	98.883%	97.48%/98.49% (-1.01%)
1	LC	959.664%	89.58%	98.33%/98.51% (-0.18%)
	LC + dLoRA	9094.627%	98.9%	97.56%/98.51% (-0.95%)

Performance similaire pour les différents rangs

Différence au niveau du taux de compression, où un rang de 1 obtient le meilleur taux de compression

Résultats

Apprentissage incrémental : Résultats sur le rang

Rank	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
4	LC	554.215%	81.956%	98.47%/98.5% (-0.03%)
	LC + dLoRA	8675.55%	98.847%	97.53%/98.5% (-0.91%)
3	LC	959.664%	89.58%	98.51%/98.51% (-0.00%)
	LC + dLoRA	8810.807%	98.865%	97.53%/98.51% (-0.92%)
2	LC	553.623%	81.937%	98.49%/98.49% (-0.00%)
	LC + dLoRA	8950.415%	98.883%	97.48%/98.49% (-1.01%)
1	LC	959.664%	89.58%	98.33%/98.51% (-0.18%)
	LC + dLoRA	9094.627%	98.9%	97.56%/98.51% (-0.95%)

Performance similaire pour les différents rangs

Différence au niveau du taux de compression, où un rang de 1 obtient le meilleur taux de compression

rank = 1 suffit pour maintenir une performance aussi bonne que les autres sur CIFAR-10, peut-être étant donné la simplicité de la tâche de classification de l'ensemble de donnée

Apprentissage incrémental : Résultats sur le rang

Complexifier la tâche en utilisant un ensemble de donnée plus compliqué

Stanford Cars Dataset

- 8144 images pour le training
- 8041 images pour le test
- 16185 images en tout
- 196 classes

Apprentissage incrémental : Résultats sur le rang

Complexifier la tâche en utilisant un ensemble de donnée plus compliqué

Stanford Cars Dataset

- 8144 images pour le training
- 8041 images pour le test
- 16185 images en tout
- 196 classes

Modèle

ViT-L/16 pré-entraîné sur ImageNet-21k

Résultats

Apprentissage incrémental : Résultats sur le rang

Rank	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
8	LC	766.253%	86.949%	76.81%/77.19% (-0.38%)
	LC + dLoRA	7036.127%	98.579%	10.53%/77.19% (-66.66%)

- Atteinte d'une accuracy de 77.19% en full

Apprentissage incrémental : Résultats sur le rang

Rank	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
8	LC	766.253%	86.949%	76.81%/77.19% (-0.38%)
	LC + dLoRA	7036.127%	98.579%	10.53%/77.19% (-66.66%)

- Atteinte d'une accuracy de 77.19% en full
- Accuracy de 10.53% avec notre schéma

Apprentissage incrémental : Résultats sur le rang

Rank	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
8	LC	766.253%	86.949%	76.81%/77.19% (-0.38%)
	LC + dLoRA	7036.127%	98.579%	10.53%/77.19% (-66.66%)

- Atteinte d'une accuracy de 77.19% en full
- Accuracy de 10.53% avec notre schéma

Raisons potentielles

Apprentissage incrémental : Résultats sur le rang

Rank	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
8	LC	766.253%	86.949%	76.81%/77.19% (-0.38%)
	LC + dLoRA	7036.127%	98.579%	10.53%/77.19% (-66.66%)

- Atteinte d'une accuracy de 77.19% en full
- Accuracy de 10.53% avec notre schéma

Raisons potentielles

- Par manque de temps, on a dû se contenter d'un nombre d'epoch faible pour le fine-tuning (ici 3)

Apprentissage incrémental : Résultats sur le rang

Rank	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
8	LC	766.253%	86.949%	76.81%/77.19% (-0.38%)
	LC + dLoRA	7036.127%	98.579%	10.53%/77.19% (-66.66%)

- Atteinte d'une accuracy de 77.19% en full
- Accuracy de 10.53% avec notre schéma

Raisons potentielles

- Par manque de temps, on a dû se contenter d'un nombre d'epoch faible pour le fine-tuning (ici 3)
- A-t-on peut-être trop compressé en appliquant Delta-LoRA sur MSA x MLP ? et en choisissant un rang de 8 ?

Conclusion

Synthèse

Conclusion

Synthèse

- Schéma **exploite les potentialités** de LC-checkpoint et de Delta-LoRA

Conclusion

Synthèse

- Schéma **exploite les potentialités** de LC-checkpoint et de Delta-LoRA
- **Intérêt** sur **différentes architectures** et sur **différents ensembles**
  - LeNet-5, AlexNet, VGG-16 et ViT
  - MNIST, CIFAR-10, Stanford Cars (en cours)

Conclusion

Synthèse

- Schéma **exploite les potentialités** de LC-checkpoint et de Delta-LoRA
- **Intérêt** sur **différentes architectures** et sur **différents ensembles**
  - LeNet-5, AlexNet, VGG-16 et ViT
  - MNIST, CIFAR-10, Stanford Cars (en cours)
- Pour les **couches d'application de Delta-LoRA** sur ViTs pour CIFAR-10
  - Si COMPRESSION, alors MSA + MLP de l'encodeur
  - Si PERFORMANCE, alors MLP de l'encodeur

Conclusion

Synthèse

- Schéma **exploite les potentialités** de LC-checkpoint et de Delta-LoRA
- **Intérêt** sur **différentes architectures** et sur **différents ensembles**
  - LeNet-5, AlexNet, VGG-16 et ViT
  - MNIST, CIFAR-10, Stanford Cars (en cours)
- Pour les **couches d'application de Delta-LoRA** sur ViTs pour CIFAR-10
  - Si COMPRESSION, alors MSA + MLP de l'encodeur
  - Si PERFORMANCE, alors MLP de l'encodeur
- Pour l'**apprentissage incrémental**
  - Pour LeNet-5 sur MNIST, la perte de performance est plus basse lorsque dataloaders de taille faible
  - Pour ViT-B/16 sur CIFAR-10, performance proche du full model et taux de compression importante

Conclusion

Synthèse

- Schéma **exploite les potentialités** de LC-checkpoint et de Delta-LoRA
- **Intérêt** sur **différentes architectures** et sur **différents ensembles**
  - LeNet-5, AlexNet, VGG-16 et ViT
  - MNIST, CIFAR-10, Stanford Cars (en cours)
- Pour les **couches d'application de Delta-LoRA** sur ViTs pour CIFAR-10
  - Si COMPRESSION, alors MSA + MLP de l'encodeur
  - Si PERFORMANCE, alors MLP de l'encodeur
- Pour l'**apprentissage incrémental**
  - Pour LeNet-5 sur MNIST, la perte de performance est plus basse lorsque dataloaders de taille faible
  - Pour ViT-B/16 sur CIFAR-10, performance proche du full model et taux de compression importante
- Pour le **rang** sur ViT-B/16 sur CIFAR-10
  - rang de 1 performance aussi bonne qu'avec les autres tout en ayant un taux de compression important

Conclusion

Critiques

- **Choix des configurations** de découpage de l'ensemble
  - 4*25%
  - 20*5%
  - 80*1.25%

Conclusion

Critiques

- **Choix des configurations** de découpage de l'ensemble
  - 4*25%
  - 20*5%
  - 80*1.25%
- **Rang constant pour tous les dataloader** à la place d'un rang dynamique

Conclusion

Critiques

- **Choix des configurations** de découpage de l'ensemble
  - 4*25%
  - 20*5%
  - 80*1.25%
- **Rang constant pour tous les dataloader** à la place d'un rang dynamique
- Si **LC-checkpoint** était considéré comme SoTA en 2020, nouveaux algorithmes ont émergés depuis :
  - QD-Compressor [2023]
  - DynaQuant [2023]
  - ExCP [2024]
- Pareil pour **Delta-LoRA**, SoTA en 2023

Conclusion

Critiques

- **Choix des configurations** de découpage de l'ensemble
  - 4*25%
  - 20*5%
  - 80*1.25%
- **Rang constant pour tous les dataloader** à la place d'un rang dynamique
- Si **LC-checkpoint** était considéré comme SoTA en 2020, nouveaux algorithmes ont émergés depuis :
  - QD-Compressor [2023]
  - DynaQuant [2023]
  - ExCP [2024]
- Pareil pour **Delta-LoRA**, SoTA en 2023
- Choix des valeurs des **superstep**

Conclusion

Perspectives

- **Généralisation** sur le découpage de l'ensemble d'intérêt
- **Extension à d'autres domaines** comme l'analyse de sentiment
  - DistilBERT (66M paramètres) sur IMDb (bases de données de revues de film)
  - RoBERTa sur IMDb
  - gpt-2 sur IMDb
- **Rang croissant** avec l'accumulation de nouvelles données permettant de pallier la réduction de l'accuracy face à une complexité croissante des connaissances à intégrer
- **Optimisation** du code (quantification à base exponentielle, la promotion de priorité)
- **Développement d'API** pour les chercheurs qui sont intéressés par la prévention d'empoisonnement de donnée et d'un crash éventuel pendant l'entraînement

Conclusion

Merci pour votre attention

Conclusion

Questions

Bibliographie

[1] Chen Y., Liu Z., Ren B., Jin X., *On Efficient Constructions of Checkpoints*. arXiv:2009.13003, 2020. [cs.LG]
[2] Zi B., Qi X., Wang L., Wang J., Wong K-F., Zhang L., *Delta-LoRA: Fine-Tuning High-Rank Parameters with the Delta of Low-Rank Matrices*. arXiv:2309.02411, 2023. [cs.LG]
[3] Huffman DA., *A Method for the Construction of Minimum-Redundancy Codes*. Proceedings of the IRE, Vol. 40, No.9, pp. 1098-1101, Sept. 1952.
[4] Ziv J., Lempel A., *A Universal Algorithm for Sequential Data Compression*. IEEE Transactions on information theory, Vol. it-23, No.3, May 1977.
[5] Hu E. et al., *LoRA: Low-Rank Adaptation of Large Language Models*. arXiv:2106.09685, 2021.
[6] LeCun Y., Cortes C., J.C. Burges C., *Modified National Institute of Standards and Technology database*. 1994.
[7] Krizhevsky A., Nair V., Hinton G., *Canadian Institute For Advanced Research - 10*. 2009.
[8] LeCun Y., Bottou L., Bengio Y., Haffner P., *Gradient Based Learning Applied to Document Recognition*. Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324,1998.
[9] Krizhevsky A., Sutskever I., Hinton G.E., *ImageNet Classification with Deep Convolutional Neural Networks*. NeurIPS 2012.
[10] Simonyan K. et al. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. arXiv:1409.1556, Oxford University, 2014.
[11] He K., Zhang X., Ren S.,Sun J., *Deep Residual Learning for Image Recognition*. arXiv:1512.03385, Microsoft, 2015.
[12] Dosovitskiy A. et al., *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. arXiv:2010.11929, 2020.

Bibliographie

[13] Niederfahrenhorst A., Hakhamaneshi K., Ahmad R., *Fine-Tuning LLMs: LoRA or Full-Parameter? An in-depth Analysis with Llama 2*. anyscale, 2023.
[14] Robbins H., Monro S., *A Stochastic Approximation Method*. University of North Carolina, 1951.
[15] McCloskey M., J. Cohen N., *Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem*. Academic Press, Psychology of Learning and Motivation, vol. 24, pp. 109-165, 1989.
[16] Jin H., Wu D., Zhang S., Zou X., Jin S., Tao D., Liao Q., Xia W., *Design of a Quantization-based DNN Delta Compression Framework for Model Snapshots and Federated Learning*. Washington State University, EECS, 2023.
[17] Agrawal A., Reddy S., Bhattamishra S., Prabhakara Sarath Nookala V., Vashishth V., Rong K., Tumanov A., *DynaQuant: Compressing Deep Learning Training Checkpoints via Dynamic Quantization*. arXiv:2306.11800, Georgia Institute Of Technology, University of Oxford, 2023.
[18] Li W., Chen X., Shu H., Tang Y., Wang Y., *ExCP: Extreme LLM Checkpoint Compression via Weight-Momentum Joint Shrinking*. arXiv:2406.11257, Huawei, 2024.
[19] Krause J., Jin H., Yang J., Fei-Fei L., *Fine-Grained Recognition without Part Annotations*. arXiv:1702.01721, Adobe Research, Stanford, 2013.
[20] Loshchilov I., Hutter F., *Decoupled Weight Decay Regularization*. arXiv:1711.05101, ICLR, 2019.
[21] Pragati Baheti, *A Comprehensive Guide to Convolutional Neural Networks*. V7Labs, Microsoft, 2021.
[22] Li C., Farkhoor H., Liu R., Yosinski J., *Measuring the Intrinsic Dimension of Objective Landscapes*. arXiv:1804.08838, 2018.
[23] Alex Krizhevsky, Vinod Nair, Geoffrey Hinton, *Canadian Institute For Advanced Research - 100*. 2009.

Annexes

Annexes

Résultats

Reproduction des résultats

Il manquait des fichiers essentiels à l'exécution du code lorsque cet étudiant m'a transféré son code

↓

Réimplémenter les parties manquantes

Obtenus dans les conditions suivantes

Model	AlexNet	VGG-16	LeNet-5
Branching Point	80.5%	72.85%	76.8%
Dataset	MNIST	MNIST	MNIST
Bit-width	3	3	3
LoRA Scaling	0.5	0.5	0.5
Batch Size	32	32	32
Learning Rate	0.01	0.01	0.01
Epochs	20	20	20
Super-Step	Every 10 iteration	Every 10 iteration	Every 10 iteration

Conditions de la reproduction

Model	AlexNet	VGG-16	LeNet-5
Branching Point	80.72%	72.85%	77.75%
Dataset	MNIST	MNIST	MNIST
Bit-width	3	3	3
LoRA Scaling	0.5	0.5	0.5
Batch Size	32	32	32
Learning Rate	0.01	0.01	0.01
Epochs	20	20	20
Super-Step	Every 10 iteration	Every 10 iteration	Every 10 iteration

Conditions de l'ancien stagiaire

Résultats préliminaires

Obtenus dans les conditions suivantes

Model	AlexNet	VGG-16	LeNet-5
Branching Point	80.72%	72.85%	77.75%
Dataset	MNIST	MNIST	MNIST
Bit-width	3	3	3
LoRA Scaling	0.5	0.5	0.5
Batch Size	32	32	32
Learning Rate	0.01	0.01	0.01
Epochs	20	20	20
Super-Step	Every 10 iteration	Every 10 iteration	Every 10 iteration

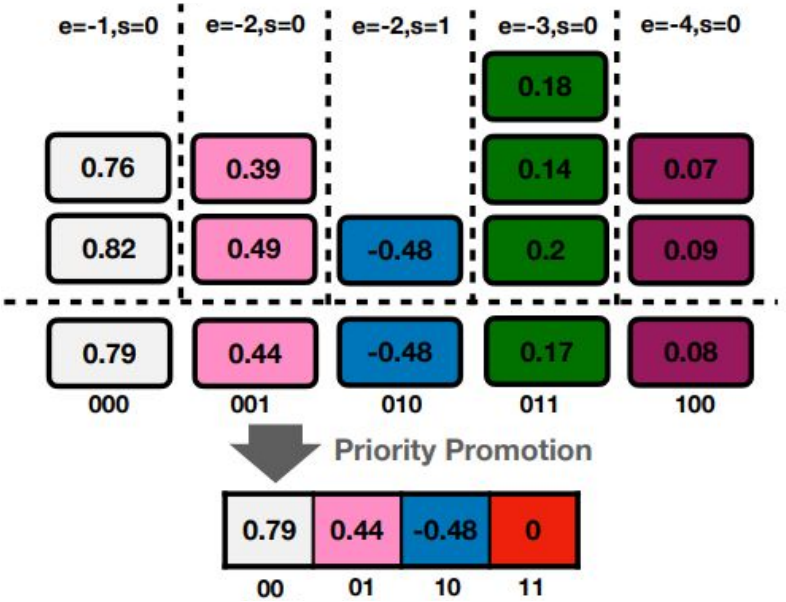
LC-checkpoint (Lossy compression checkpoint)

Promotion de priorité

Lorsque $\delta_{t,i} \approx 0 (= u_{i,\tilde{t}-1} \approx u_{i,t})$, il est plus efficace de regrouper les mises à jour

Conserve $2^x - 1$ buckets avec le **plus grand e** et fusionne le reste **en un seul avec la valeur 0**, avec x la largeur de bit

Remarque : Une largeur de bits de 3 a été jugée suffisante pour la plupart des modèles [3]



LC-checkpoint (Lossy compression checkpoint)

Encodage de Huffman

LC-checkpoint (Lossy compression checkpoint)

Encodage de Huffman **Technique de compression sans perte d'information**

LC-checkpoint (Lossy compression checkpoint)

Encodage de Huffman **Technique de compression sans perte d'information**

Convertit chaque moyenne de compartiment (bucket) en une chaîne de bits

LC-checkpoint (Lossy compression checkpoint)

Encodage de Huffman **Technique de compression sans perte d'information**

Convertit chaque moyenne de compartiment (bucket) en une chaîne de bits

Étapes de l'encodage

LC-checkpoint (Lossy compression checkpoint)

Encodage de Huffman **Technique de compression sans perte d'information**

Convertit chaque moyenne de compartiment (bucket) en une chaîne de bits

Étapes de l'encodage

- Fréquence des caractères de chaque caractère

LC-checkpoint (Lossy compression checkpoint)

Encodage de Huffman

Technique de compression sans perte d'information

Convertit chaque moyenne de compartiment (bucket) en une chaîne de bits

Étapes de l'encodage

- Fréquence des caractères de chaque caractère
- Placés dans un arbre binaire afin que les caractères les plus fréquents aient les chemins les plus courts depuis la racine
  - Associe à chaque noeud la fréquence du caractère
  - Fusionne les noeuds ayant les fréquences les plus faibles pour former un noeud dont la fréquence est la somme

LC-checkpoint (Lossy compression checkpoint)

Encodage de Huffman

Technique de compression sans perte d'information

Convertit chaque moyenne de compartiment (bucket) en une chaîne de bits

Étapes de l'encodage

- Fréquence des caractères de chaque caractère
- Placés dans un arbre binaire afin que les caractères les plus fréquents aient les chemins les plus courts depuis la racine
  - Associe à chaque noeud la fréquence du caractère
  - Fusionne les noeuds ayant les fréquences les plus faibles pour former un noeud dont la fréquence est la somme
- Le chemin pour atteindre chaque caractère depuis la racine de l'arbre définit son code : aller à gauche peut représenter un '0' et aller à droite un '1'

Delta-LoRA

LoRA - Low Rank Adaptation [5]

Méthode populaire d'adaptation (fine-tuning : lorsqu'on entraîne un modèle pré-entraîné sur un autre ensemble de donnée) ajoutant un nombre limité de paramètres en conservant la performance

Motivation

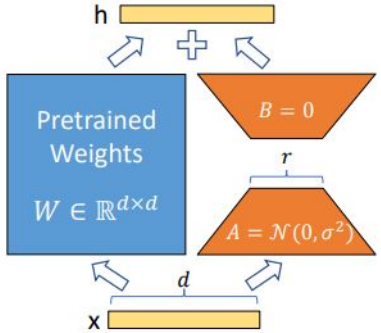
par un article publié en 2018 qui traite de la dimensionnalité intrinsèque des grands modèles [5], affirmant qu'il existe une paramétrisation de faible dimension

Fonctionnement de LoRA

Suppose que le changement de poids du modèle ΔW a une faible dimension intrinsèque pour mettre à jour la matrice de poids du modèle pré-entraîné figé W_0 de taille $d \times k$

$$W = W_0 + \Delta W \quad \text{avec} \quad \Delta W = A \times B$$

$$A \in R^{d \times r}$$
$$B \in R^{r \times k}$$
$$r \ll \min(d, k)$$



Annexes

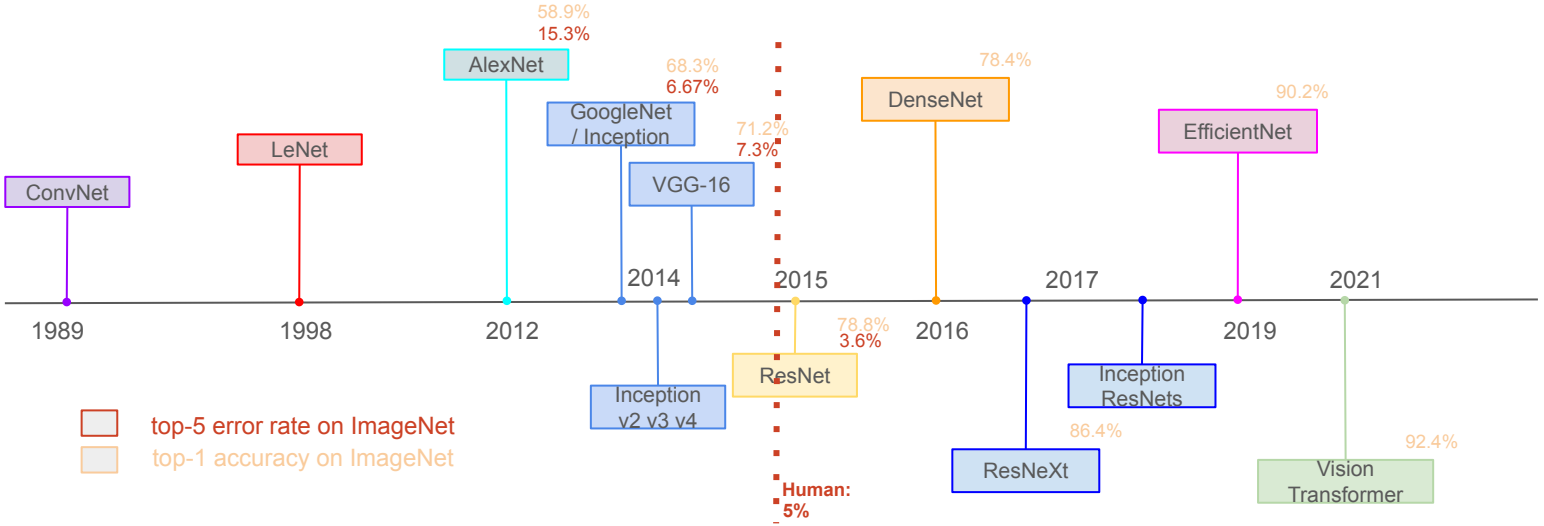
Dataset Splitting	4*15k (=25%)	20*3k (=5%)	80*750 (=1.25%)	4*12.5k (=25%)
Model	LeNet-5			ViT-B/16
Branching Point	74.6%			HF‡ pretrained ImageNet1k
Dataset	MNIST			CIFAR-10
Bit-width	3			3
LoRA Scaling	0.5			0.5
Batch Size	128			128
Learning Rate	0.08			4e-5
Epochs	100			5
Super-Step	Every 10 iterations			Every 10 iterations
Stopping Criterion	Early stopping			Early stopping
Optimizer	SGD			Adam¶

TABLE 11 – Comparaison des configurations de LeNet-5 et ViT-B/16 sur les différents datasets

Annexes

Choix Des Modèles

source : https://paperswithcode.com/sota/image-classification-on-imagenet



Apprentissage incrémental : Résultats sur le rang

Rank	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
16	LC	534.542%	81.292%	98.27%/98.23% (+0.04%)
	LC + dLoRA	7325.409%	98.63%	96.56%/98.23% (-1.67%)
4	LC	534.414%	81.29%	98.27%/98.21% (+0.06%)
	LC + dLoRA	8675.518%	98.847%	96.71%/98.21% (-1.5%)

ViT-B/16 obtient des performances similaires pour des rangs différents

Apprentissage incrémental : Résultats sur le rang

Rank	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
16	LC	534.542%	81.292%	98.27%/98.23% (+0.04%)
	LC + dLoRA	7325.409%	98.63%	96.56%/98.23% (-1.67%)
4	LC	534.414%	81.29%	98.27%/98.21% (+0.06%)
	LC + dLoRA	8675.518%	98.847%	96.71%/98.21% (-1.5%)

ViT-B/16 obtient des performances similaires pour des rangs différents, mais des taux de compression plus importants pour un rang faible.

Apprentissage incrémental : Résultats sur le rang

Rank	Methods	Compression Ratio	Space Savings	Final Accuracy (Restored / Full)
16	LC	534.542%	81.292%	98.27%/98.23% (+0.04%)
	LC + dLoRA	7325.409%	98.63%	96.56%/98.23% (-1.67%)
4	LC	534.414%	81.29%	98.27%/98.21% (+0.06%)
	LC + dLoRA	8675.518%	98.847%	96.71%/98.21% (-1.5%)

ViT-B/16 obtient des performances similaires pour des rangs différents, mais des taux de compression plus importants pour un rang faible.

Seconde étude a été menée pour approfondir ce rang faible dans le même contexte

Calcul du taux de compression et du gain en espace de stockage

Formules

```
compression_ratio = round((uncompressed_size / compressed_size), 5) * 100
space_savings = round(1 - (compressed_size / uncompressed_size), 5) * 100
```