

# Optimization des Hyperparamètres appliquée au Fine Tuning de LLM

Basé sur l'article : *Bayesian and Partition-Based Optimization for Hyperparameter Optimization of LLM Fine-Tuning*

Nathan Davouse

# Sommaire

1. Introduction

2. Design et Implémentation

3. Résultats et Analysis

4. Conclusion

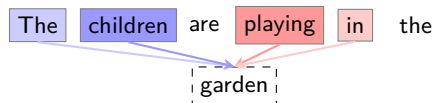
## Large Language Models

## Point clés

- ▶ Etat de l'art pour le traitement de langage naturel.
- ▶ Réseaux de Neurones avec une architecture basé sur le transformer<sup>a</sup> (annexe 1)
- ▶ Taille : entre 1 et 405 Milliards de neurones

<sup>a</sup>Vaswani et al, Attention is all you need, 2017

## Auto-attention



**Figure:** Illustration du mécanisme d'auto-attention

L'auto-attention est la clé du LLM, en permettant de comprendre le contexte



# Optimisation des Hyperparamètres (OHP)

## Hyperparamètres

Paramètres qui ne sont pas entraînés par le modèle  
(learning rate, dropout ...)

## Objectifs

- Meilleure performance qu'en manuel
- Retirer le besoin d'expertise

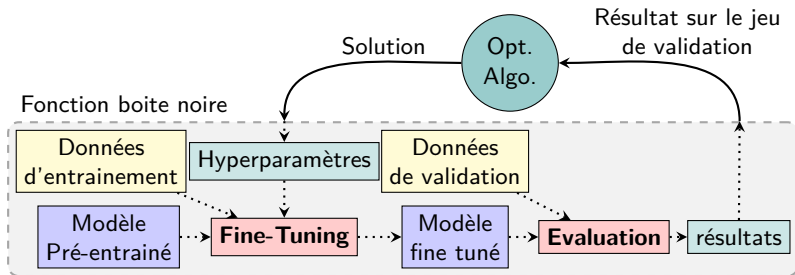


Figure: Fonctionnement général de l'optimisation des hyperparamètres



## Travaux connexes

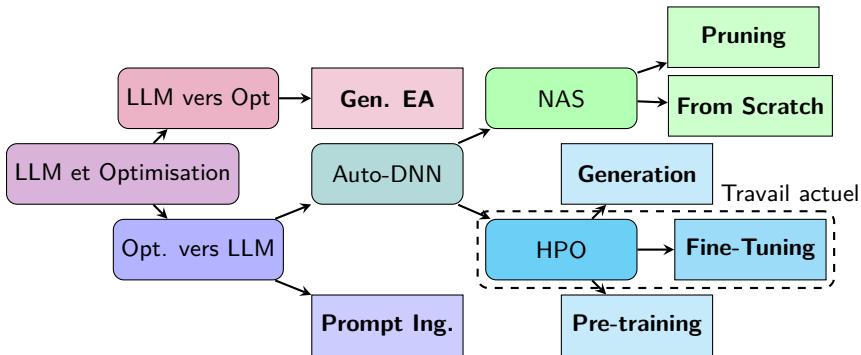


Figure: Classification des travaux similaires

# Sommaire

1. Introduction

2. Design et Implémentation

3. Résultats et Analysis

4. Conclusion





## Strategie de Recherche : Optimisation Bayésienne par Process Gaussien

## Algorithm

- ▶ Echantillon de  $n$  Points (LHS)
- ▶ Evaluer ces  $n$  points
- ▶ Jusqu'à fin du budget
  - Entraîner le Process Gaussien (GP)
  - Optimiser ce GP pour obtenir un nouveau Point
  - Evaluer ce nouveaux point

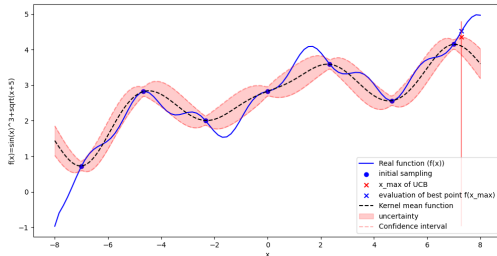


Figure: Exemple d'un surrogate sur une fonction en 1D

## Strategie de Recherche : Simultaneous Optimistic Optimization (SOO)

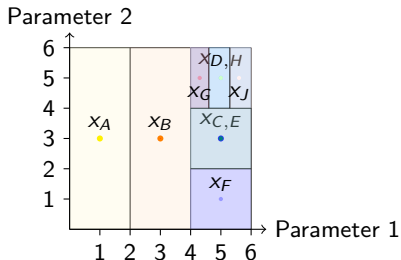


Figure: Partition de l'espace de recherche par SOO

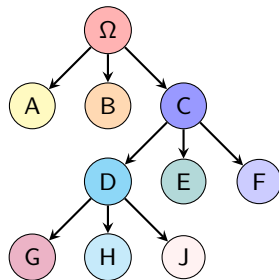


Figure: Arbre correspondant à S00

## Search Strategy : BaMSOO (TO DO)

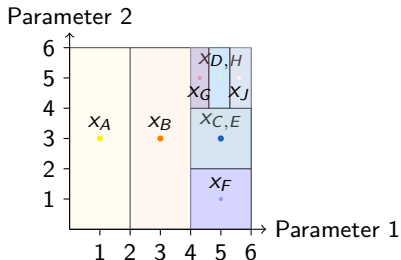


Figure: Partition de l'espace de recherche par SOO

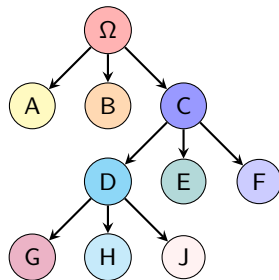


Figure: Arbre correspondant à SOO

# Stratégie d'Evaluation de Solutions

## Implémentation

### ► Fine Tuning

- modèle : LLaMa-3.2-1B
- 

### ► Evaluation

- librairie lm\_eval
- Evaluation par la précision sur des jeu de données Benchmark : Hellaswag et MMLU

## Implémentation

- Programmation Orienté Object en Python
- Travail de documentation : *readme*, indication de type...
- Objectif : permettre le réusage
- Utilisable en ligne de commande pour Grid5000

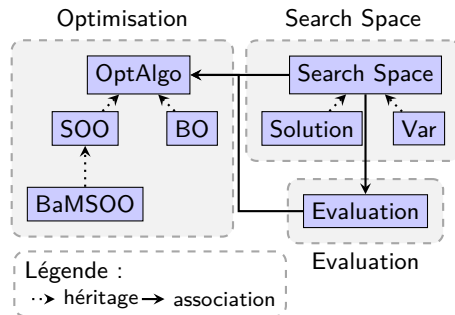


Figure: Diagramme de l'implémentation

# Sommaire

1. Introduction

2. Design et Implémentation

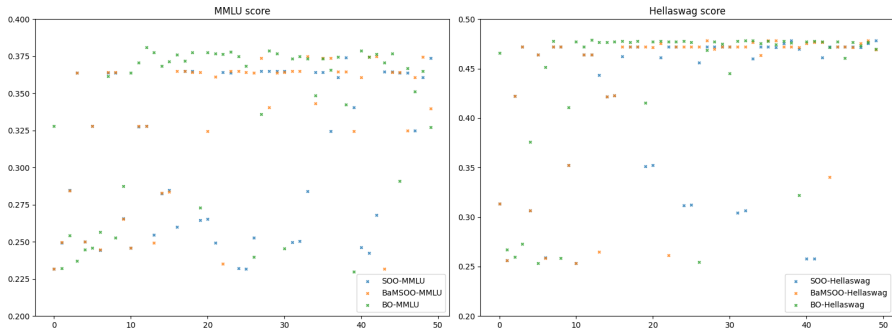
3. Résultats et Analysis

4. Conclusion





## Résultats des 3 algorithmes



**Figure:** Résumé des résultats par sampling. Détails par algorithme dans les annexes 4, 5 et 6



## Perspectives

### Poursuite du travail

- Retour sur l'article et présentation en conférence
- Elargissement de l'espace de recherche
- Diversification sur les modèles/données

### Ouverture

- Parallélisation d'algorithme d'optimisation de fonction coûteuse pour Exascale (project Exa-MA)
- Intégration de substitut dans les algorithmes parallèles à Partition<sup>a</sup>

---

<sup>a</sup>Partition-based Parallel Bayesian Optimisation

# Sommaire

1. Introduction

2. Design et Implémentation

3. Résultats et Analysis

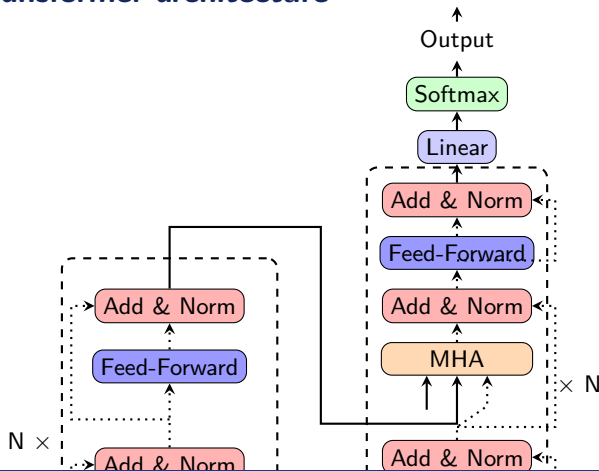
4. Conclusion

## Conclusion

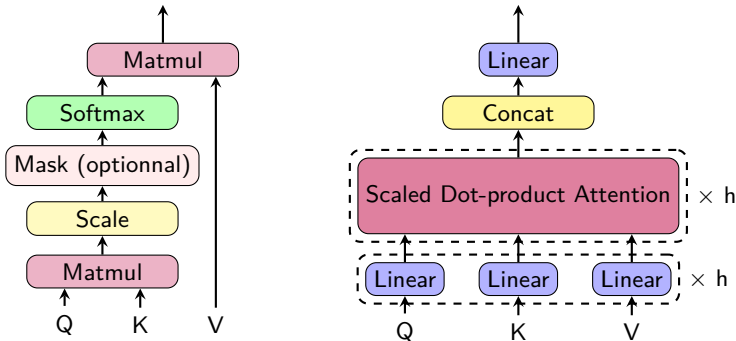
Une conclusion

*Merci.*

## Annexe 1 : Transformer architecture



## Annexe 2 : Multi-Head Attention



**Figure:** Illustration du mécanisme d'auto-attention : A droite le mécanisme complet, a gauche le *Scaled Dot-product Attention*



## Annexe 3 : Low Rank Adaptation (LoRA)

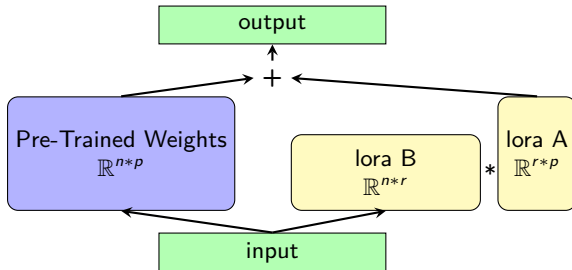


Figure: Illustration de l'application du Low Rank Adaptation (LoRA)

## Annexe 4 : Résultats pour BO

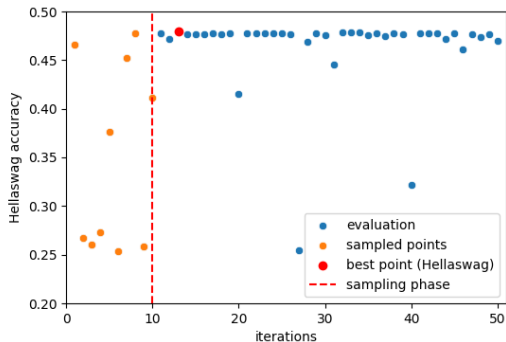


Figure: Evolution des score lors de l'expérience BO

## Annexe 5 : Résultats pour SOO

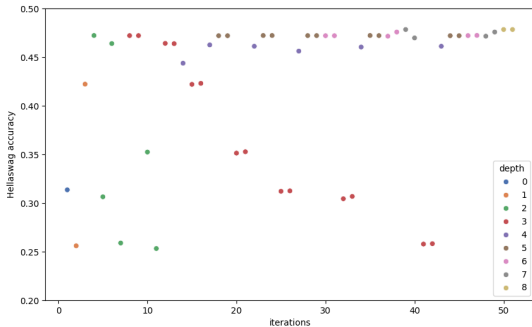


Figure: Evolution des score lors de l'expérience SOO

