

# 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



# 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>1</sup> (annexe 3)
- ▶ Taille : entre 1 et 405 Milliards de neurones

<sup>1</sup>Vaswani et al, Attention is all you need,2017

## Auto-attention

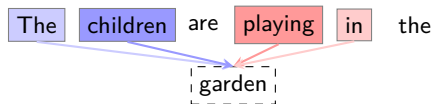


Figure: Illustration du mécanisme d'auto-attention

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

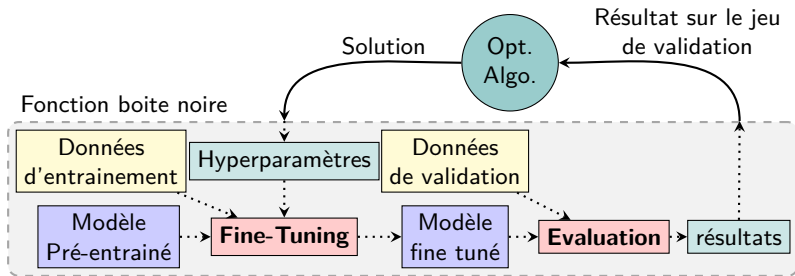


Figure: Fonctionnement général de l'optimisation des hyperparamètres



# Sommaire

1. Introduction

2. Design et Implémentation

3. Résultats et Analyses

4. Conclusion

















## Stratégie de Recherche : Bayesian Multi-Scale Optimistic Optimization (BaMSOO)

Décomposition suivant SOO, mais utilisant des Process Gaussien pour éviter les évaluations non prometteuses.

## Evaluation par BaMSOO

- ▶ If  $UCB(x) > f^+$  :
  - $g(x) = f(x)$  real evaluation
- ▶ Else :
  - $g(x) = LCB(x)$  use LCB to replace  $f(x)$

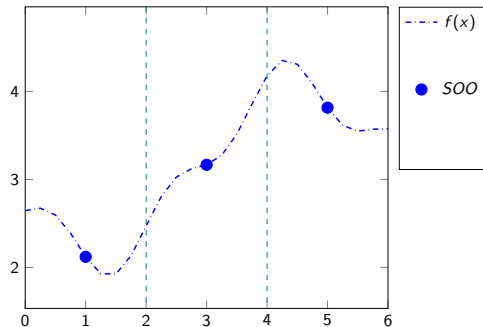


Figure: Illustration de l'Algorithme BaMSOO





## Stratégie de Recherche : Bayesian Multi-Scale Optimistic Optimization (BaMSOO)

Décomposition suivant SOO, mais utilisant des Process Gaussien pour éviter les évaluations non prometteuses.

## Evaluation par BaMSOO

- ▶ If  $UCB(x) > f^+$  :
  - $g(x) = f(x)$  real evaluation
- ▶ Else :
  - $g(x) = LCB(x)$  use LCB to replace  $f(x)$

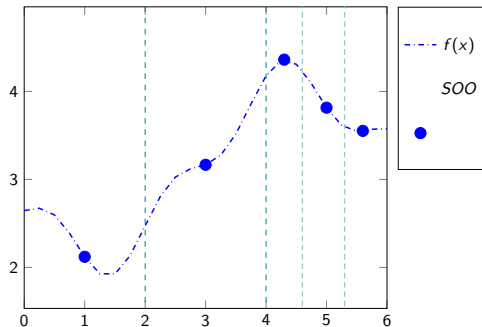


Figure: Illustration de l'Algorithme BaMSOO







# Sommaire

### 3. Résultats et Analyses



## Résultats des 3 algorithmes

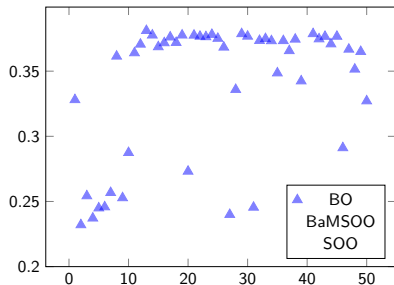


Figure: Résultats sur MMLU (test)

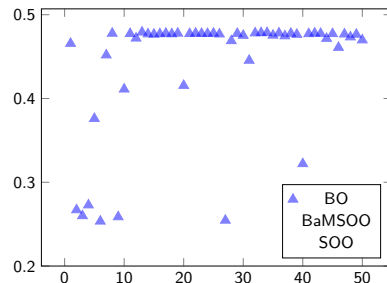


Figure: Résultats sur Hellaswag (Validation)

## Résultats des 3 algorithms

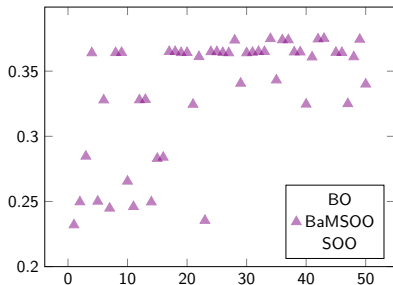


Figure: Résultats sur MMLU (test)

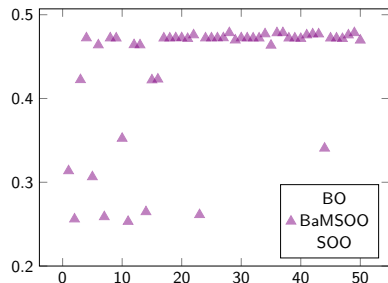


Figure: Résultats sur Hellaswag (Validation)



## Résultats des 3 algorithms

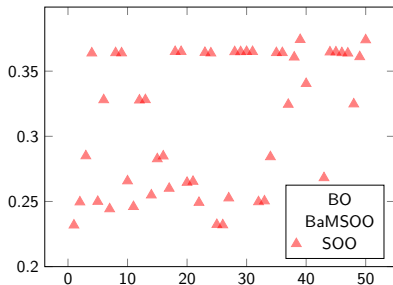


Figure: Résultats sur MMLU (test)

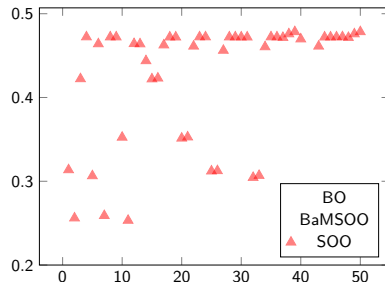


Figure: Résultats sur Hellaswag (Validation)

## Résultats des 3 algorithmes

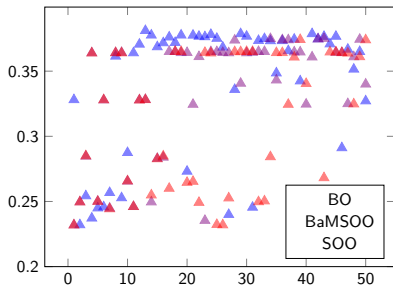


Figure: Résultats sur MMLU (test)

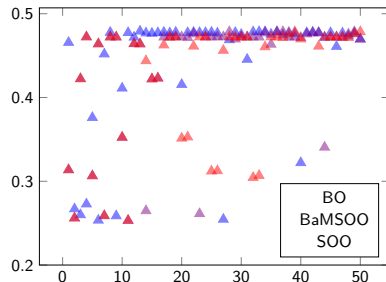


Figure: Résultats sur Hellaswag (Validation)

## Analyse

Jeu de données	Borne Inf. <sup>1</sup>	Borne Sup. <sup>2</sup>	BO-GP	SOO	BaMSOO
Hellaswag (validation)	47.90	41.5	<b>47.91</b>	47.84	47.84
MMLU (testing)	37.61	49.3	<b>38.11</b>	37.42	37.50

**Table:** Bornes et meilleurs résultats sur les 2 jeu de données

1 : expérience avec LHS; 2 : Fine tuning par Meta

## Points clés

- ▶ Borne Sup. sur Hellaswag non pertinente
- ▶ Seul BO arrive au dessus de LHS
- ▶ BaMSOO n'améliore que peu SOO
- ▶ principe de BaMSOO fonctionnel (visible annexe 8)
- ▶ Espace de solution n'évolue que peu, le retravailler pour mesurer pleinement la performance des algorithmes



# Sommaire

1. Introduction

2. Design et Implémentation

3. Résultats et Analyses

4. Conclusion

## Résultats du stage

Une conclusion

## Apprentissage

Une conclusion

## Poursuite du projet professionnel

Une conclusion



*Merci.*

## Annexe 1 : Travaux connexes

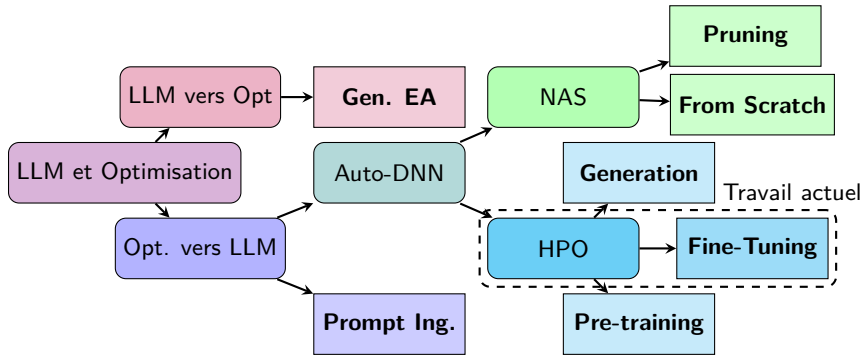


Figure: Classification des travaux similaires

## Annexe 2 : Stratégie d'Evaluation de Solutions

### Implémentation

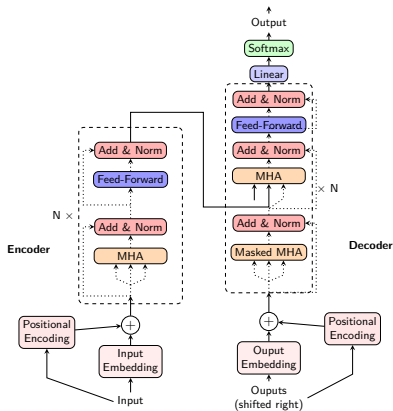
#### ► Fine Tuning

- Modèle : LLaMa-3.2-1B
- Jeu de données d'entraînement : Alpaca
- LitGPT framework : basé sur Pytorch, facilite le Fine Tuning de LLM

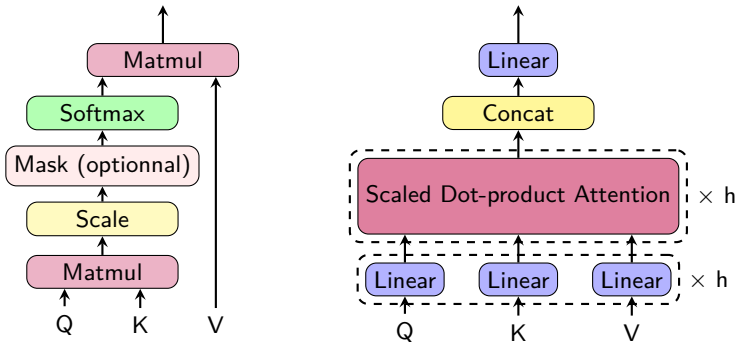
#### ► Evaluation

- librairie lm\_eval : standard pour l'évaluation de LLM
- Evaluation par la précision sur des jeu de données Benchmark : Hellaswag et MMLU

## Annexe 3 : Transformer architecture

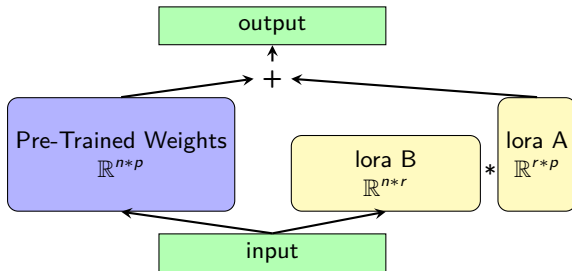


## Annexe 4 : Multi-Head Attention



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

## Annexe 5 : Low Rank Adaptation (LoRA)



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

## Annexe 6 : Résultats pour B0

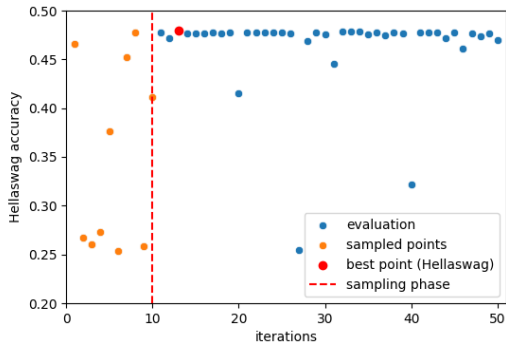


Figure: Evolution des score lors de l'expérience BO

## Annexe 7 : Résultats pour S00

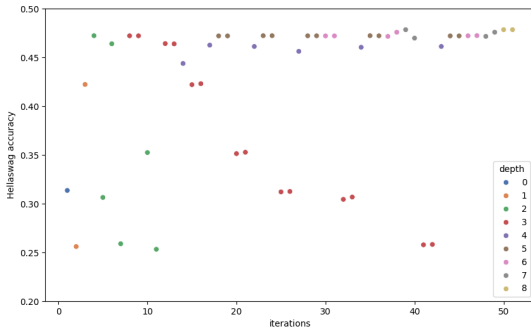


Figure: Evolution des score lors de l'expérience S00





