

# 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

# 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 1)
- ▶ Taille : entre 1 et 405 Milliards de neurones

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

## Auto-attention

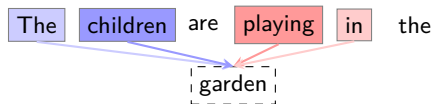


Figure: Illustration du mécanisme d'auto-attention

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







## Travaux connexes

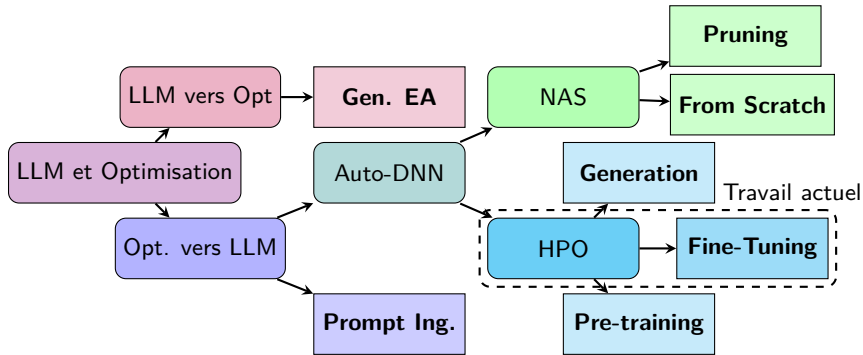


Figure: Classification des travaux similaires

# 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

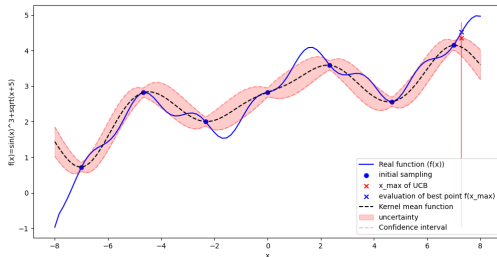


Figure: Exemple d'un surrogate sur une fonction en 1D





# 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
- ▶ Intégralement open-source<sup>2</sup>

<sup>2</sup>[https://github.com/Kiwy3/B0\\_PBO\\_HPO\\_LLM](https://github.com/Kiwy3/B0_PBO_HPO_LLM)

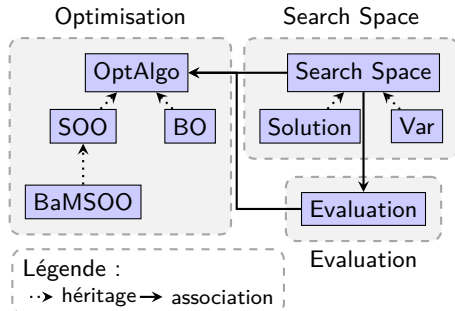


Figure: Diagramme de l'implémentation

# Sommaire

1. Introduction

2. Design et Implémentation

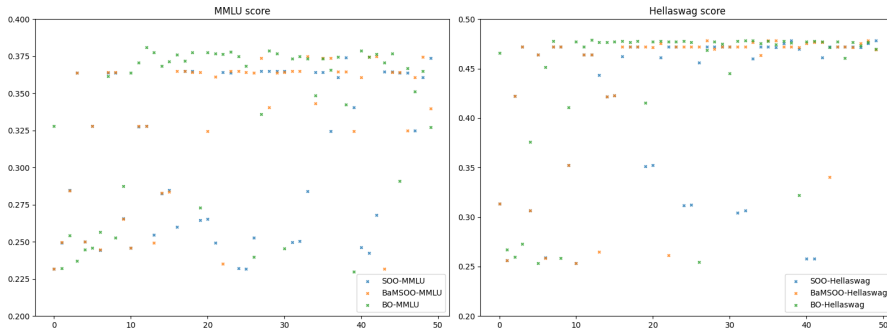
**3. Résultats et Analysis**

4. Conclusion





## Résultats des 3 algorithmes



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

## Analyse

Jeu de données	Borne Inf. <sup>1</sup>	Borne Sup. <sup>2</sup>	BO-GP	SOO	BaMSOO
Hellaswag (validation)	47.90	41.5	47.91	47.84	47.84
MMLU (testing)	37.61	49.3	38.11	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 6)
- ▶ Espace de solution n'évolue que peu, le retravailler pour mesurer pleinement la performance des algorithmes

## Perspectives

### Poursuite du travail

- ▶ Retour sur l'article et présentation en conférence (si validation)
- ▶ Elargissement de l'espace de recherche
- ▶ Diversification sur les modèles/données

### Généralisation hors LLM

- ▶ Parallélisation d'algorithme d'optimisation de fonction coûteuse pour Exascale (project Exa-MA)
- ▶ Intégration de substitut dans les algorithme parallèle à Partition <sup>3</sup>

---

<sup>3</sup>Partition-based Parallel Bayesian Optimisation

# Sommaire

## 4. Conclusion

## Résultats du stage

Une conclusion

## Apprentissage

Une conclusion

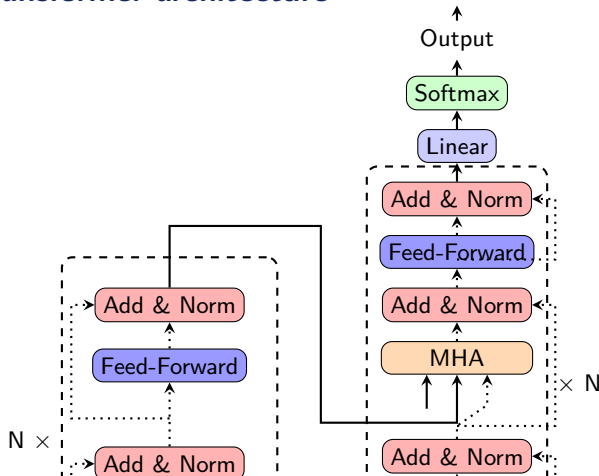
## Poursuite du projet professionnel

Une conclusion

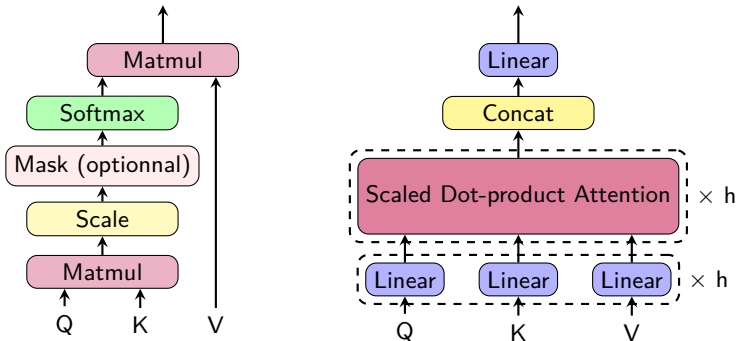
*Merci.*



## Annexe 1 : Transformer architecture



## Annexe 2 : Multi-Head Attention



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





## Annexe 5 : Résultats pour S00

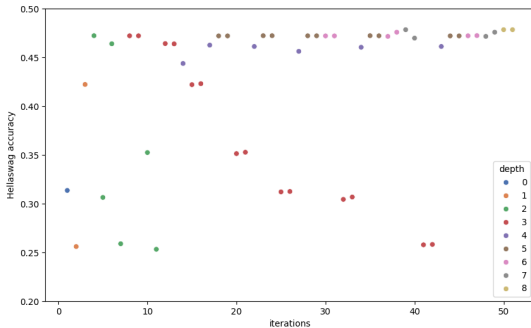


Figure: Evolution des score lors de l'expérience S00

