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

Large Language Models

Point clés

- Etat de l'art pour le traitement de language naturel.
- Réseaux de Neurones avec une architecture basé sur le transformer[1] (annexe 1)
- Taille : entre 1 et 405 Milliards de neurones

[1] Vaswani et al, Attention is all you need,2017

Auto-attention

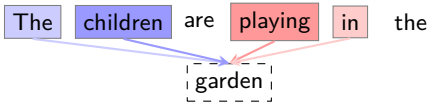


Figure: Illustration du mécanisme d'auto-attention

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

Fine Tuning

Aspect	Pre-entrainement	Fine Tuning
Objectif	Apprentissage general	Adaptation à un domaine
Données	Larges et diverses	Restreintes et Spécifiques
Ressources	Centaines de GPU	au moins 1 GPU
Durée	Semaine/Mois	Heures/Jours

Table: Comparaison entre le Pre-entrainement et le Fine Tuning de LLM

Parameter-Efficient Fine-Tuning (PEFT)

- Ensemble de méthodes pour réduire le nombre de paramètres à entrainer
- Utilisation de la méthode LoRA (annexe 3)
- Amène des nouveaux hyperparamètres

Optimisation des Hyperparamètres (OHP)

Hyperparamètres

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

Objectifs

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

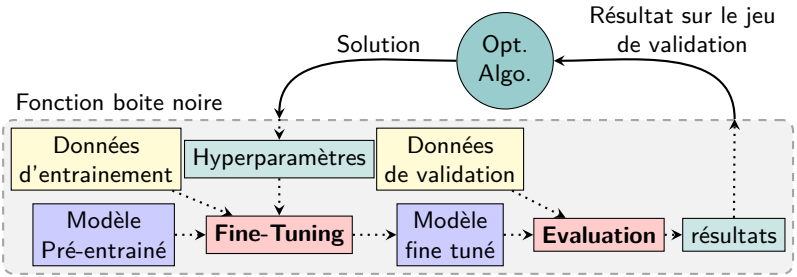


Figure: Fonctionnement général de l'optimisation des hyperparamètres

Formulation du problème

Equation

$$\eta^* \in \arg\max_{\eta \in \mathcal{A}} f(\eta), \quad f : \mathbb{R}^d \to \mathbb{R} \tag{1}$$

Avec η une solution de dimension d et f la fonction représentant l'entrainement et l'évaluation d'un modèle.

Charactéristiques de la fonction f

- Boite-noire : non dérivable
- Couteux : une évaluation se compte en dizaine de minutes
- Bruité : évaluer 2 fois la même solution peut donner un résultat différent
- Variables mixes : les variables sont de plusieurs type (entier, continu...)

Travaux connexes

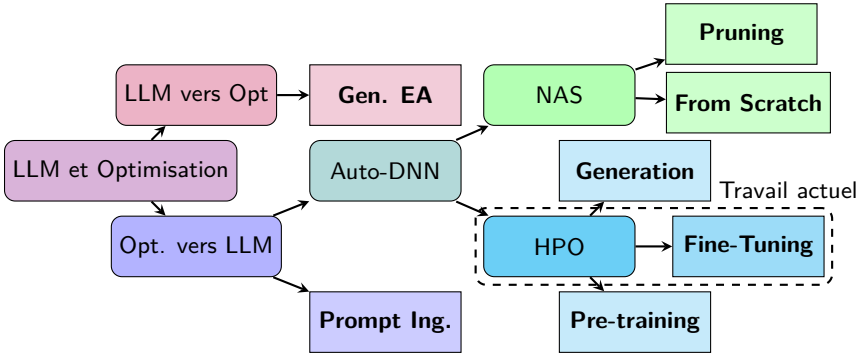


Figure: Classification des travaux similaires

Sommaire

Espace de Recherche

Hyperparamètres	Plage d'Optimisation		Type	Conversion
	Borne Inf.	Borne Sup.		
Learning Rate	-10	-1	log.	$f(x) = 10^x$
LoRA rank	1	64	ent.	$f(x) = \text{round}(x)$
LoRA scale	1	64	ent.	$f(x) = \text{round}(x)$
Dropout	0	0.5	cont.	$f(x) = x$
Weight Decay	-3	-1	log.	$f(x) = 10^x$

Table: Résumé de l'espace de recherche

- Variables mixes : étape de conversion nécessaire

Algorithm

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

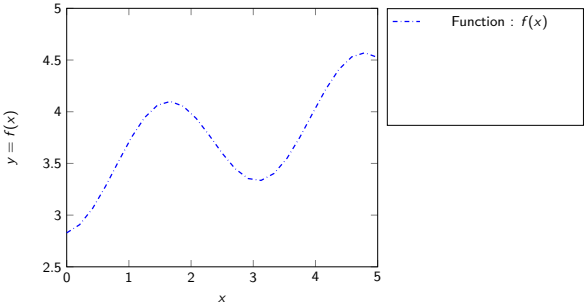
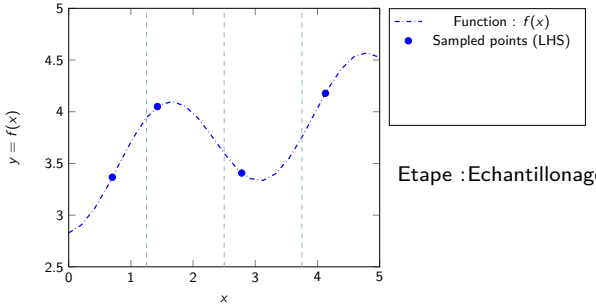


Figure: Example d'un surrogate sur une fonction en 1D

Strategie de Recherche : Optimisation Bayésienne par Process Gaussien

Algorithm

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



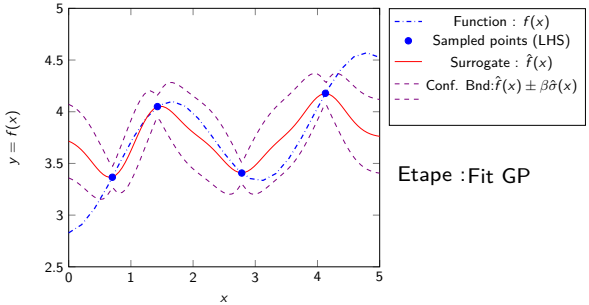
Etape :Echantillonage

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

Strategie de Recherche : Optimisation Bayésienne par Process Gaussien

Algorithm

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



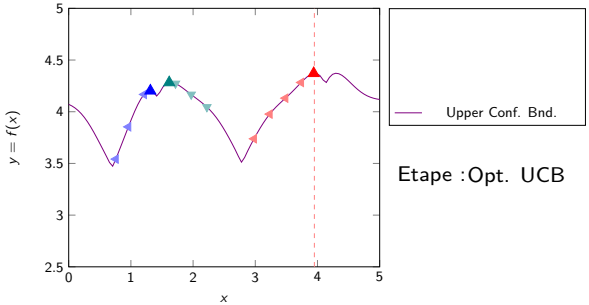
Etape : Fit GP

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

Strategie de Recherche : Optimisation Bayésienne par Process Gaussien

Algorithm

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



Etape :Opt. UCB

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

Strategie de Recherche : Simultaneous Optimistic Optimization (SOO)

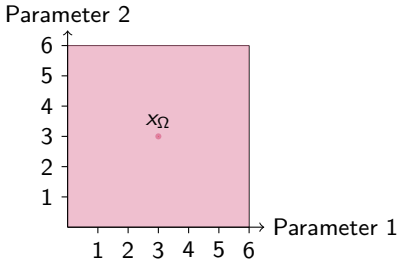


Figure: Partition de l'espace de recherche par SOO

Figure: Arbre correspondant à SOO

Strategie de Recherche : Simultaneous Optimistic Optimization (SOO)

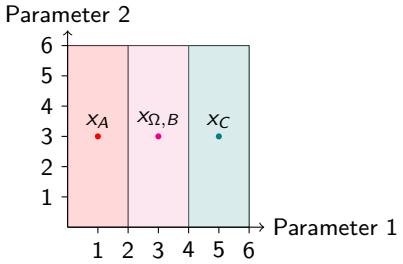


Figure: Partition de l'espace de recherche par SOO

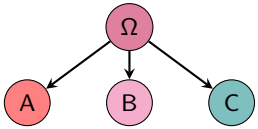


Figure: Arbre correspondant à SOO

Strategie de Recherche : Simultaneous Optimistic Optimization (SOO)

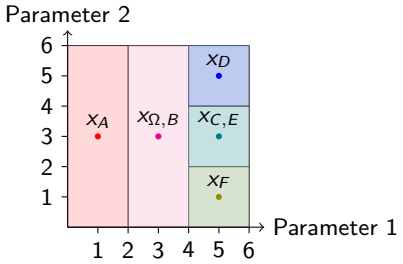


Figure: Partition de l'espace de recherche par SOO

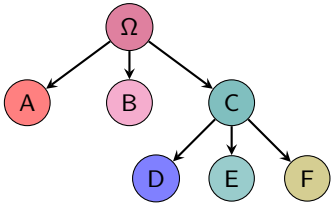


Figure: Arbre correspondant à SOO

Strategie de Recherche : Simultaneous Optimistic Optimization (SOO)

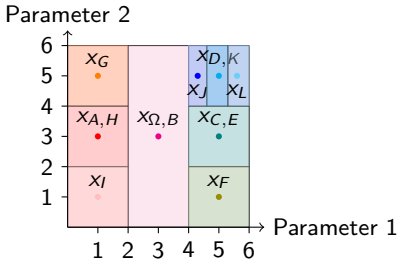


Figure: Partition de l'espace de recherche par SOO

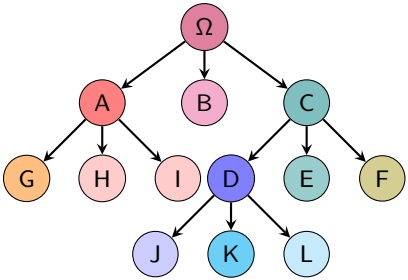


Figure: Arbre correspondant à SOO

Search Strategy : BaMSOO (TO DO)

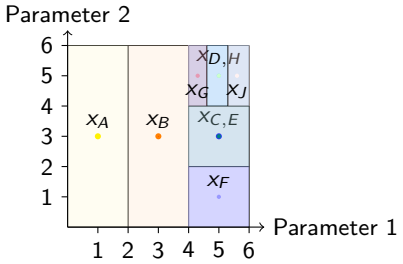


Figure: Partition de l'espace de recherche par SOO

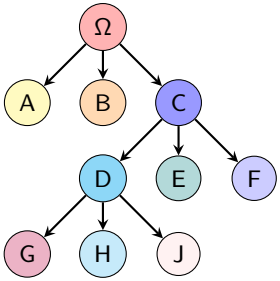


Figure: Arbre correspondant à SOO

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
- Intégralement open-source[2]

[2] `https://github.com/Kiwy3/BO_PBO_HPO_LLM`

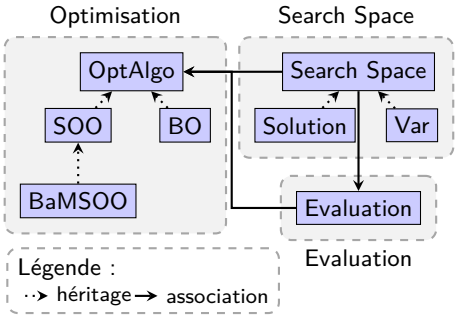


Figure: Diagramme de l'implémentation

Sommaire

Echantillonnage avec LHS

Principe

Explorer l'espace et proposer une borne inférieure

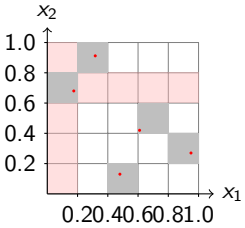


Figure: Illustration du Latin Hypercube Sampling avec $g = 5$

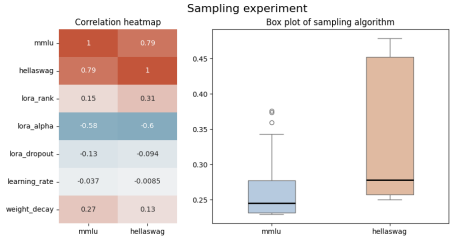


Figure: Résumé des résultats par sampling

Résultats des 3 algorithms

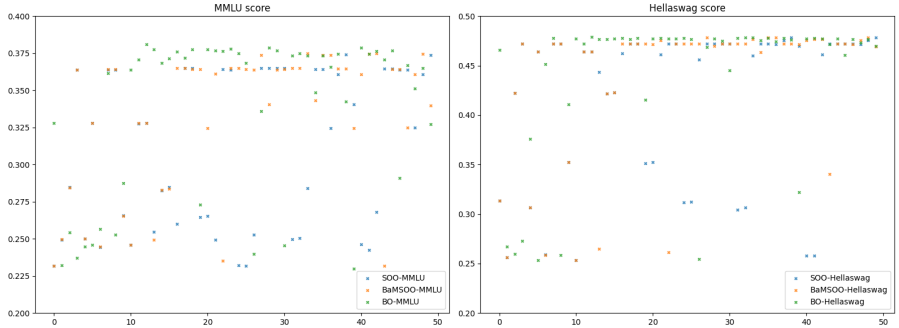


Figure: Résumé des résultats par sampling. *D*étails par algorithme dans les annexes 4, 5 et 6

.

Analyse

Jeu de données	Borne Inf.[1]	Borne Sup.[2]	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

- Parallelisation d'algorithme d'optimisation de fonction couteuse pour Exascale (project Exa-MA)
- Intégration de substitut dans les algorithme parallèle à Partition [3]

[3] *P*artition-based Parallel Bayesian Optimisation

Sommaire

Résultats du stage

Une conclusion

Apprentissage

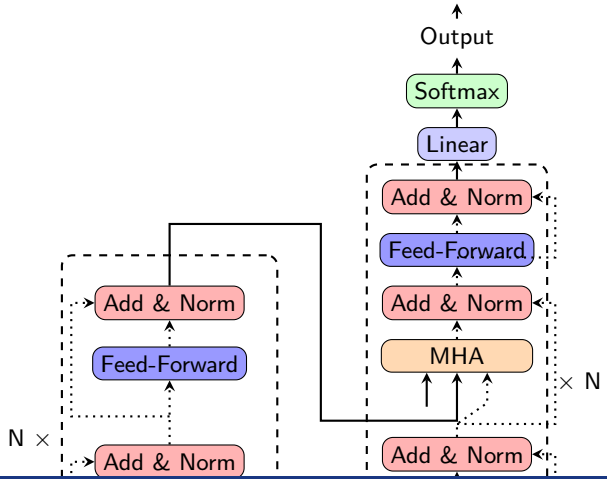
Une conclusion

Poursuite du projet profressionel

Une conclusion

Merci.

Annexe 1 : Transformer architecture



Annexe 2 : Multi-Head Attention

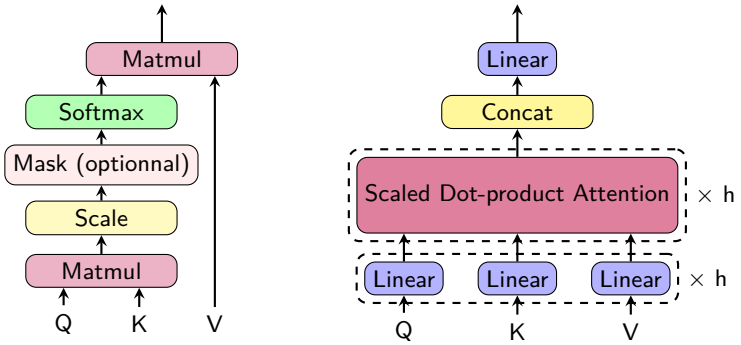


Figure: Illustration du mécanisme d'auto-attention : A droite le mécanisme complet, a gauche le *Scaled Dot-product Attention*

Annexe 3 : Low Rank Adaptation (LoRA)

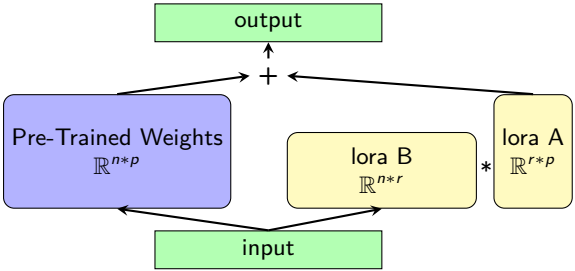


Figure: Illustration de l'application du Low Rank Adaptation (LoRA)

Annexe 4 : Résultats pour BO

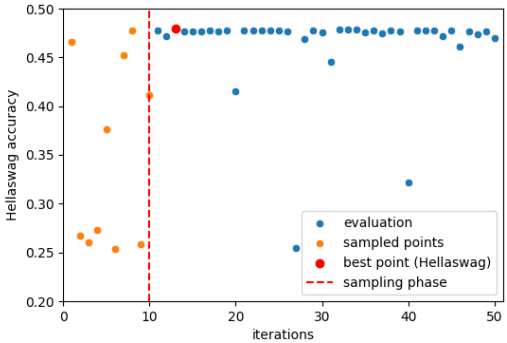


Figure: Evolution des score lors de l'expérience BO

Annexe 5 : Résultats pour SOO

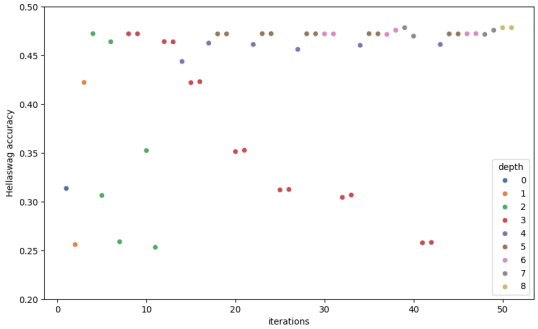


Figure: Evolution des score lors de l'expérience SOO

Annexe 6 : Résultats pour BaMSOO

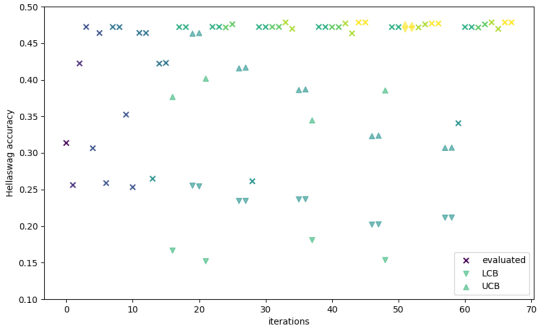


Figure: Evolution des score lors de l'expérience BaMSOO

Annexe 7 : Modèle Beamer UTT

Figure: Presentation du template beamer sur github : lien