Etude de la performance de LLaMA versus autres LLMs

Auteur de l'étude : Oleiwan Joe

Cadre : Projet sur la détection de la subjectivité dans les FakeNews.

Base :

- Article Scientifique du groupe Meta AI : LLaMA: Open and Efficient Foundation Language Models

  Lien : https://arxiv.org/pdf/2302.13971.pdf

Conclusion immédiate : LLaMA est plus performant que les autres LLMs (à l'exception de Mixtral) dans différents aspect mais pas dans la totalité.

I- Data utilisé pour l'entrainement :

Toutes les données d'entraînement sont **open source** :

Dataset	Sampling prop.	Epochs	Disk size
CommonCrawl	67.0%	1.10	3.3 TB
C4	15.0%	1.06	783 GB
Github	4.5%	0.64	328 GB
Wikipedia	4.5%	2.45	83 GB
Books	4.5%	2.23	85 GB
ArXiv	2.5%	1.06	92 GB
StackExchange	2.0%	1.03	78 GB

II- Points techniques :

Tokenizer : byte-pair encoding (BPE) algorithm basé sur le Sentence-pièce de Kudo and Richardson, 2018.

Modèles : 7B, 13B, 33B, 65B :

params	dimension	n heads	n layers	learning rate	batch size	n tokens
6.7B	4096	32	32	$3.0e^{-4}$	4M	1.0T
13.0B	5120	40	40	$3.0e^{-4}$	4M	1.0T
32.5B	6656	52	60	$1.5e^{-4}$	4M	1.4T
65.2B	8192	64	80	$1.5e^{-4}$	4M	1.4T

Table 2: **Model sizes, architectures, and optimization hyper-parameters.**

Architecture par rapport à l'architecture transformer originale introduite par Vaswani et al. en 2017 :

1. **Pre-normalization (Inspirée par GPT-3)** : LLaMA utilise une technique de pré-normalisation où la normalisation est appliquée à l'entrée de chaque sous-couche du transformateur au lieu de sa sortie. Ceci est fait pour améliorer la stabilité durant l'entraînement. La fonction de normalisation utilisée est RMSNorm, qui a été introduite par Zhang et Sennrich en 2019.
2. **Fonction d'activation SwiGLU (Inspirée par PaLM)** : Au lieu de l'activation ReLU habituellement utilisée dans les réseaux de neurones, LLaMA utilise la fonction d'activation SwiGLU, proposée par Shazeer en 2020. Cette fonction est connue pour améliorer les performances du modèle. LLaMA utilise une dimension de 2/3×4d pour cette fonction d'activation, contrairement à PaLM qui utilise une dimension de 4d.
3. **Rotary Embeddings (Inspirés par GPTNeo)** : LLaMA élimine les embeddings positionnels absolus et utilise à la place des embeddings positionnels rotatifs (RoPE), qui ont été introduits par Su et al. en 2021. Ces embeddings sont ajoutés à chaque couche du réseau et permettent au modèle de mieux capturer la relation entre les positions dans les séquences de données.

Les techniques d'optimisation utilisées pour rendre l'entraînement du modèle LLaMA plus efficace :

1. **Implémentation efficace de l'attention multi-têtes causale** : Pour réduire l'utilisation de la mémoire et le temps d'exécution, LLaMA utilise une version optimisée de l'attention multi-têtes causale qui ne stocke pas les poids d'attention et ne calcule pas les scores clé/requête masqués par la causalité. Cette implémentation est disponible dans la bibliothèque xformers et s'inspire des travaux de Rabe et Staats (2021) ainsi que de l'approche de rétropropagation de Dao et al. (2022).
2. **Checkpointing pour économiser les activations** : Afin d'améliorer encore l'efficacité de l'entraînement, LLaMA sauvegarde les activations coûteuses à calculer, comme les sorties des couches linéaires. Cela est réalisé en implémentant manuellement la fonction de rétropropagation pour les couches de transformateurs, plutôt qu'en se fiant à la fonction d'autograd de PyTorch. Cette technique de checkpointing réduit la quantité de calculs nécessaires lors de la passe arrière.
3. **Parallélisme de modèle et de séquence** : Pour minimiser l'usage de la mémoire, LLaMA emploie le parallélisme de modèle et de séquence, une technique qui permet de distribuer les calculs sur plusieurs unités de traitement comme les GPU. Ce concept a été décrit par Korthikanti et al. (2022).

4. **Chevauchement des calculs et communications** : LLaMA maximise l'efficience en chevauchant le calcul des activations avec la communication entre les GPU sur le réseau, ce qui est crucial lors des opérations de réduction globale (all_reduce).

Grâce à ces optimisations, l'entraînement d'un modèle LLaMA avec 65 milliards de paramètres peut traiter environ 380 tokens par seconde par GPU sur 2048 GPU A100 avec 80 Go de RAM, permettant de compléter l'entraînement sur un ensemble de données contenant 1,4 billion de tokens en environ 21 jours.

III- Principaux résultats :

		BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA
GPT-3	175B	60.5	81.0	-	78.9	70.2	68.8	51.4	57.6
Gopher	280B	79.3	81.8	50.6	79.2	70.1	-	-	-
Chinchilla	70B	83.7	81.8	51.3	80.8	74.9	-	-	-
PaLM	62B	84.8	80.5	-	79.7	77.0	75.2	52.5	50.4
PaLM-cont	62B	83.9	81.4	-	80.6	77.0	-	-	-
PaLM	540B	**88.0**	82.3	-	83.4	**81.1**	76.6	53.0	53.4
LLaMA	7B	76.5	79.8	48.9	76.1	70.1	72.8	47.6	57.2
	13B	78.1	80.1	50.4	79.2	73.0	74.8	52.7	56.4
	33B	83.1	82.3	50.4	82.8	76.0	**80.0**	**57.8**	58.6
	65B	85.3	**82.8**	**52.3**	**84.2**	77.0	78.9	56.0	**60.2**

Le modèle que nous pourrions implémenter à défaut de performances.

Table 3: **Zero-shot performance on Common Sense Reasoning tasks.**

Tâches en Zero-shot et Few-shot :

- **Zero-shot** : Description textuelle de la tâche et exemple de test donnés, le modèle génère une réponse ou classe des réponses proposées.
- **Few-shot** : Quelques exemples de la tâche (entre 1 et 64) et un exemple de test sont donnés, le modèle génère la réponse ou classe différentes options.

Explication des benchmarks pour compréhension des résultats :

1. **BoolQ (Boolean Questions)** : Un benchmark de compréhension de lecture où les modèles répondent à des questions fermées par oui ou non en se basant sur un passage de texte court.
2. **PIQA (Physical Interaction QA)** : Teste la compréhension physique d'un modèle par des questions pratiques, demandant de prédire l'issue d'interactions physiques dans des scénarios donnés.

3. **SIQA (Social Interaction QA)** : Un benchmark de raisonnement social qui pose des questions basées sur des scénarios de la vie quotidienne pour tester la compréhension des normes sociales et des conséquences des actions.
4. **HellaSwag** : Un benchmark pour l'évaluation de la compréhension du langage modèle et la prédiction de fin d'histoires. Les modèles doivent choisir la continuation la plus plausible d'un scénario parmi plusieurs options.
5. **WinoGrande** : Un test de raisonnement de bon sens à grande échelle qui améliore le WSC (Winograd Schema Challenge) original. Il évalue la capacité d'un modèle à résoudre des problèmes de désambiguïsation pronominale.
6. **ARC (AI2 Reasoning Challenge) Easy (ARC-e) et Challenge (ARC-c)** : Un ensemble de questions de quiz scientifique de niveau scolaire. ARC-e contient des questions plus faciles, tandis que ARC-c comprend des questions jugées plus difficiles pour les modèles d'IA.
7. **OBQA (Open Book Question Answering)** : Un benchmark de compréhension de lecture qui teste la capacité du modèle à répondre à des questions de connaissance générale sans accéder à une source externe d'information (comme si l'examen était "à livre ouvert").

Performances sur les benchmarks :

- LLaMA montre des performances compétitives sur une variété de benchmarks de raisonnement de bon sens, y compris BoolQ, PIQA, SIQA, HellaSwag, WinoGrande, ARC (facile et challenge) et OpenBookQA.
- Sur des benchmarks de question-réponse en livre fermé comme Natural Questions et TriviaQA, LLaMA-65B atteint des performances de pointe en zero-shot et few-shot.
- LLaMA-13B est compétitif avec GPT-3 et Chinchilla malgré sa taille plus petite, fonctionnant sur un seul GPU V100 pendant l'inférence.

Performances sur des Tâches spécifiques :

- **Raisonnement mathématique** : LLaMA est évalué sur des benchmarks de raisonnement mathématique, tels que MATH et GSM8k, montrant de meilleures performances que certains modèles finement ajustés sur des données mathématiques malgré l'absence de fine-tuning sur de telles données.
- **Génération de code** : LLaMA démontre une capacité à générer du code à partir d'une description en langage naturel sur des benchmarks comme HumanEval et MBPP, surpassant d'autres modèles généraux qui n'ont pas été finement ajustés pour la génération de code.

Compréhension massive en multitâche :

		Humanities	STEM	Social Sciences	Other	Average
GPT-NeoX	20B	29.8	34.9	33.7	37.7	33.6
GPT-3	175B	40.8	36.7	50.4	48.8	43.9
Gopher	280B	56.2	47.4	71.9	66.1	60.0
Chinchilla	70B	63.6	54.9	79.3	**73.9**	67.5
PaLM	8B	25.6	23.8	24.1	27.8	25.4
	62B	59.5	41.9	62.7	55.8	53.7
	540B	**77.0**	**55.6**	**81.0**	69.6	**69.3**
LLaMA	7B	34.0	30.5	38.3	38.1	35.1
	13B	45.0	35.8	53.8	53.3	46.9
	33B	55.8	46.0	66.7	63.4	57.8
	65B	61.8	51.7	72.9	67.4	63.4

Table 9: **Massive Multitask Language Understanding (MMLU).** Five-shot accuracy.

- Sur le benchmark MMLU, LLaMA-65B est légèrement derrière Chinchilla et PaLM, ce qui pourrait être dû à l'utilisation limitée de livres et de documents académiques dans les données d'entraînement de LLaMA.
    - ➔ Nous verrons dans une autre étude entre Mixtral et LLaMa que Mixtral comble cet aspect essentiel pour notre projet sur la détéction de la subjectivité.

Le modèle fine-tuned (Instruction Finetuning pour le MMLU) LLaMA-I :

OPT	30B	26.1
GLM	120B	44.8
PaLM	62B	55.1
PaLM-cont	62B	62.8
Chinchilla	70B	67.5
LLaMA	65B	63.4
OPT-IML-Max	30B	43.2
Flan-T5-XXL	11B	55.1
Flan-PaLM	62B	59.6
Flan-PaLM-cont	62B	66.1
LLaMA-I	65B	**68.9**

Table 10: **Instruction finetuning – MMLU (5-shot).** Comparison of models of moderate size with and without instruction finetuning on MMLU.

LLaMA-I (65B) surpasse d'autres modèles de taille modérée affinés avec des instructions sur MMLU, mais il reste encore loin de l'état de l'art, qui est de 77.4 pour le modèle GPT code-davinci-002 sur MMLU.

Point sur la toxicité, biais et vérité de réponses du modèle :

Les modèles de langage peuvent générer un langage toxique, par exemple : insultes, discours de haine ou menaces.

1. Évaluation de la toxicité avec RealToxicityPrompts : Les scores de toxicité augmentent avec la taille du modèle, en particulier pour les prompts respectueux.
2. Évaluation des biais avec CrowS-Pairs, en mettant l'accent sur le genre : LLaMA présente des biais, en particulier dans les catégories religieuse, d'âge et de genre, mais obtient des résultats légèrement meilleurs que d'autres modèles.
3. Analyse des biais de genre avec WinoGender : LLaMA présente des biais de genre dans les résolutions de co-référence, montrant des préférences pour certaines professions.
4. Évaluation de la vérité des réponses avec TruthfulQA : LLaMA obtient des scores plus élevés que GPT-3 en termes de vérité des réponses, mais la qualité des réponses reste faible.

    ➔ Les résultats sont mitigés, nous attendons de meilleures résultats avec Mixtral.

IV-Conclusion :

Dans cette étude comparative de performance entre LLaMA et d'autres modèles de langage, nous avons exploré en détail les capacités, les techniques d'optimisation, et les résultats de LLaMA dans une variété de tâches et de benchmarks. Voici les conclusions clés de cette étude :

1. **Capacités de LLaMA** : LLaMA a démontré sa polyvalence en générant du texte, en effectuant des tâches de compréhension de lecture, en résolvant des problèmes de raisonnement, en générant du code Python, et en participant à diverses tâches de question-réponse, allant du zero-shot au few-shot.

2. **Architecture Améliorée** : LLaMA se distingue par plusieurs améliorations par rapport à l'architecture de base des transformateurs, notamment la pré-normalisation, l'utilisation de la fonction d'activation SwiGLU, et l'adoption des embeddings positionnels rotatifs (RoPE).

3. **Optimisations Efficaces** : L'entraînement de LLaMA est rendu efficace grâce à diverses techniques d'optimisation, telles que l'attention multi-têtes causale optimisée, le checkpointing pour économiser les activations, le parallélisme de modèle et de séquence, et le chevauchement des calculs et des communications.

4. **Performances Concurrentielles** : LLaMA a affiché des performances concurrentielles sur de nombreux benchmarks, notamment BoolQ, PIQA, SIQA, HellaSwag, WinoGrande, ARC (facile et challenge), et OpenBookQA. Il a également excellé dans les tâches de raisonnement mathématique et de génération de code.

5. **Défis de Toxicité et de Biais** : L'étude a souligné les défis liés à la toxicité, aux biais, et à la vérité des réponses. LLaMA a montré une tendance à augmenter la toxicité avec la taille du modèle et a révélé des biais de genre dans certaines tâches, bien que les performances en termes de vérité des réponses restent limitées.

6. **Conclusion sur MMLU** : LLaMA-65B a obtenu des résultats compétitifs, bien qu'il reste derrière Chinchilla et PaLM sur le benchmark MMLU, en partie en raison de l'accès limité aux livres et aux documents académiques dans ses données d'entraînement.

7. **Potentiel avec Mixtral** : La conclusion a évoqué que Mixtral pourrait compléter certains aspects essentiels pour le projet de détection de la subjectivité, suggérant que des résultats plus prometteurs pourraient être atteints avec l'utilisation de Mixtral en complément de LLaMA.

En résumé, LLaMA est un modèle de langage puissant et polyvalent avec des améliorations architecturales significatives et des optimisations efficaces. Cependant, il fait face à des défis liés à la toxicité, aux biais, et à la qualité des réponses. Nous verrons à travers une autre étude sur le LLM Mixtral comment répondre à ces manques, pour répondre aux besoins spécifiques d'applications comme la détection de la subjectivité dans les FakeNews.