AYI NEDJIMI Consultants

Optimisation cluster GPU pour l'inférer

16 mai 2026 • Mis à jour le 17 mai 2026 • 17 min de lecture • 3130 mots •

Optimisez votre cluster GPU pour l'inférence LLM : tensor parallelism, pipe
batching strategies, autoscaling Kubernetes. Guide technique pour les équ

À RETENIR

A retenir -- Optimisation cluster GPU LLM

L'**optimisation d'un cluster GPU pour l'inference LLM** peut reduire les couts
deploiement naif. Les leviers principaux sont : la selection de la strategie de
pipeline), l'optimisation du KV-cache management (PagedAttention), le conti
et l'autoscaling intelligent pour adapter la capacite a la charge reelle. Pour le
critique pour rendre economiquement viable un deploiement LLM on-premis

L'**optimisation des clusters GPU pour l'inference LLM** est devenue une competen
DSI qui deploient des LLM en production. La difficult modernes (
caracteristiques d'inference tres differentes des mo els : ils son

bound, leur usage de la VRAM GPU varie dramatiquement selon la longueur des s
concurrente requiert des strategies de batching sophistiquees pour etre efficaces
30 a 50% de sa capacite theorique, doublant les couts d'infrastructure par rappor
detaille les techniques d'optimisation applicables avec les frameworks open sourc
(A100, H100, RTX 4090) pour maximiser le throughput et minimiser la latence tout

Tensor parallelism vs Pipeline parallelism -- quelle strategie choisir

Pour les modeles qui ne tiennent pas dans la VRAM d'un seul GPU, deux strategie

Le ***Tensor Parallelism*** (TP) divise chaque layer du modele entre plusieurs GPU : le
horizontalement, chaque GPU traite une fraction de chaque layer et les resultats s
Avantage : latence reduite car tous les GPU travaillent en parallele sur chaque toke
(necessite NVLink ou un interconnect haute bande passante comme InfiniBand), e
communication est inferieur au temps de calcul.

Le ***Pipeline Parallelism*** (PP) divise le modele en stages consecutifs, chaque GPU t
differents batches en parallele (pipeline pipelining). Avantage : moins de communi
stages adjacents). Inconvenient : latence plus elevee due aux "pipeline bubbles" (

2 GPU : Tensor Parallelism TP=2 est presque toujours superieur si les GPU sont

4-8 GPU, meme noeud : TP=4 ou TP=8 avec NVLink est optimal pour les latenc

Multi-noeuds (>8 GPU) : combinaison TP dans le noeud + PP entre noeuds pou
latence

```
# vLLM avec tensor parallelism sur 4 GPU A1
python -m vllm.entrypoints.openai.api_serve                    llama/Lla
```

Réponse sous 24h

Devis gratuit →