Relatório Técnico Deepseek-V3

Deepseek-AI

research@deepseek.com

Resumo

Apresentamos Deepseek-V3, um modelo de linguagem de mistura de especialistas (MOE) forte com parâmetros totais de 671b com 37b ativado para cada token. Para obter inferência eficiente e treinamento econômico, o DeepSeek-V3 adota a atenção latente de várias cabeças (MLA) e as arquiteturas Deepseekmoe, que foram minuciosamente validadas em Deepseek-V2. Além disso, o DeepSeek-V3 pioneiro em uma estratégia livre de perdas auxiliares para balanceamento de carga e define um objetivo de treinamento de previsão de vários toques para um desempenho mais forte. Antes do treino, o Deepseek-V3 em 14,8 trilhões de tokens diversos e de alta qualidade, seguido de etapas de aprendizado de ajuste fino e reforço supervisionadas para aproveitar completamente suas capacidades. Avaliações abrangentes revelam que o DeepSeek-V3 supera outros modelos de código aberto e atinge o desempenho comparável aos principais modelos de código fechado. Apesar de seu excelente desempenho, o Deepseek-V3 requer apenas 2,788m de H800 GPU Hours para seu treinamento completo. Além disso, seu processo de treinamento é notavelmente estável. Durante todo o processo de treinamento, não experimentamos picos de perda irrecuperável ou realizamos reversão. Os pontos de verificação do modelo estão disponíveis em https://github.com/deepseek-ai/deepseek-v3.

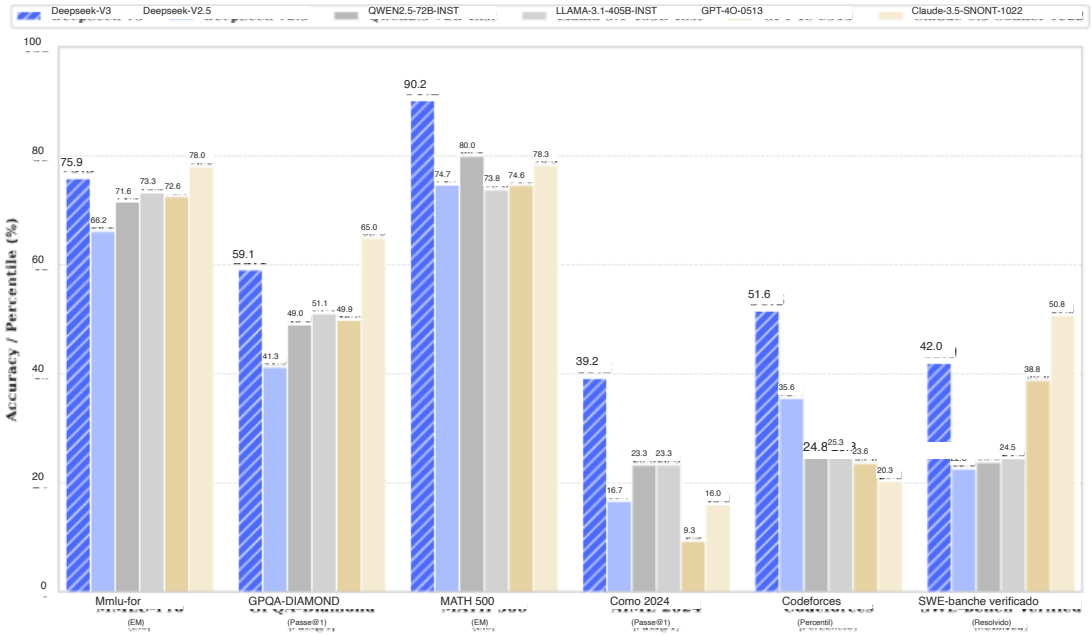


Figura 1 | Desempenho de referência do Deepseek-V3 e seus colegas.

Conteúdo

1. Introdução

Nos últimos anos, os grandes modelos de idiomas (LLMs) estão passando por rápida iteração e evolução (Antrópica, 2024; Google, 2024; OpenAI, 2024a), diminuindo progressivamente a lacuna para a inteligência geral artificial (AGI). Além dos modelos de código fechado, modelos de código aberto, incluindo a série Deepseek (Deepseek-AI, 2024a, B, C; Guo et al., 2024), série LLAMA (AI@meta, 2024a, b; Touvron et al., 2023a , b), a série Qwen (Qwen, 2023, 2024a, b) e a série Mistral (Jiang et al., 2023; Mistral, 2024), também estão fazendo avanços significativos, tentando fechar a lacuna com seus colegas de fonte fechada. Para aumentar ainda mais os limites das capacidades do modelo de código aberto, ampliamos nossos modelos e introduzimos o modelo Deepseek-V3, um grande modelo de mistura de especialistas (MOE) com parâmetros 671b, dos quais 37b são ativados para cada token.

Com uma perspectiva prospectiva, buscamos consistentemente um forte desempenho do modelo e custos econômicos. Portanto, em termos de arquitetura, o Deepseek-V3 ainda adota atenção latente de várias cabeças (MLA) (Deepseek-AI, 2024C) para inferência eficiente e Deepseekmoe (Dai et al., 2024) para treinamento econômico. Essas duas arquiteturas foram validadas em DeepSeek-V2 (Deepseek-AI, 2024C), demonstrando sua capacidade de manter o desempenho robusto do modelo, alcançando treinamento e inferência eficientes. Além da arquitetura básica, implementamos duas estratégias adicionais para aprimorar ainda mais os recursos do modelo. Em primeiro lugar, o DeepSeek-V3 Pi- Oneers uma estratégia livre de perdas auxiliares (Wang et al., 2024a) para balanceamento de carga, com o objetivo de minimizar o impacto adverso no desempenho do modelo que surge do esforço para incentivar o balanceamento de carga. Em segundo lugar, o Deepseek-V3 emprega um objetivo de treinamento de previsão de vários toques, que observamos para melhorar o desempenho geral nos benchmarks de avaliação.

Para obter treinamento eficiente, apoiamos o treinamento de precisão misto do FP8 e implementamos otimizações abrangentes para a estrutura de treinamento. O treinamento de baixa precisão emergiu como uma solução promissora para treinamento eficiente (Dettmers et al., 2022; Kalamkar et al., 2019; Narang et al., 2017; Peng et al., 2023b), sua evolução está intimamente ligada a avanços. em recursos de hardware (Luo et al., 2024; Micikevicius et al., 2022; Rouhani et al., 2023a). Neste trabalho, introduzimos uma estrutura de treinamento de precisão mista do FP8 e, pela primeira vez, valide sua eficácia em um modelo extremamente em larga escala. Através do suporte para computação e armazenamento FP8, alcançamos treinamento acelerado e uso reduzido de memória da GPU. Quanto à estrutura de treinamento, projetamos o algoritmo dualpipe para o paralelismo eficiente da pipeline, que tem menos bolhas de pipeline e oculta a maior parte da comunicação durante o treinamento através da sobreposição de computação-comunicação. Essa sobreposição garante que, à medida que o modelo aumenta ainda mais, desde que mantemos uma taxa de computação / comunicação constante, ainda possamos empregar especialistas em granulação fino nos nós enquanto alcançamos uma sobrecarga de comunicação quase zero. Além disso, também desenvolvemos núcleos de comunicação entre todos os nó eficientes para utilizar completamente as larguras de banda Infiniband (IB) e NVLink. Além disso, otimizamos meticulosamente a pegada de memória, possibilitando treinar Deepseek-V3 sem usar o paralelismo do tensor caro. Combinando esses esforços, alcançamos alta eficiência de treinamento.

Durante o pré-treinamento, treinamos Deepseek-V3 em tokens de alta qualidade e de alta qualidade. O processo de pré-treinamento é notavelmente estável. Durante todo o processo de treinamento, não encontramos picos de perda irrecuperável ou precisamos reverter. Em seguida, realizamos uma extensão de comprimento de contexto de dois estágios para Deepseek-V3. No primeiro estágio, o comprimento máximo do contexto é estendido a 32k e, no segundo estágio, é ainda estendido a 128k. Depois disso, realizamos o pós-treinamento, incluindo o ajuste fino supervisionado (SFT) e o aprendizado de reforço (RL) no modelo básico de Deepseek-V3, para alinhá-lo com preferências humanas e desbloquear ainda mais seu potencial. Durante a fase pós-treinamento, destilamos a capacidade de raciocínio da série de modelos Deepseek-R1 e, enquanto isso, mantemos cuidadosamente o equilíbrio entre a precisão do modelo

Custos de treinamento	Pré-treinamento	Extensão de contexto	Pós-treinamento	Total
Nas horas da GPU H800	2664K	119K	5k	2788K
em USD	US $ 5,328M	US $ 0,238M	$ 0,01M	US $ 5,576M

Tabela 1 l Os custos de treinamento do DeepSeek-V3, assumindo que o preço do aluguel do H800 é de US $ 2 por hora da GPU.

e comprimento da geração.

Avaliamos o Deepseek-V3 em uma variedade abrangente de benchmarks. Apesar de seus custos econômicos de treinamento, as avaliações abrangentes revelam que o Deepseek-V3-BASE emergiu como o modelo básico de código aberto mais forte atualmente disponível, especialmente em código e matemática. Sua versão de bate-papo também supera outros modelos de código aberto e atinge o desempenho comparável aos principais modelos de código fechado, incluindo GPT-4O e Claude-3.5 Sonet, em uma série de benchmarks padrão e de ponta aberta.

Por fim, enfatizamos novamente os custos econômicos de treinamento do DeepSeek-V3, resumidos na Tabela 1, alcançados através do nosso co-design otimizado de algoritmos, estruturas e hardware. Durante a fase de pré-treinamento, o treinamento de Deepseek-V3 em cada trilhão de tokens requer apenas 180 mil horas de GPU H800, ou seja, 3,7 dias em nosso cluster com gpus de 2048 H800. Consequentemente, nossa fase de pré-treinamento é concluída em menos de dois meses e custa 2664 mil horas de GPU. Combinado com 119 mil horas de GPU para a extensão do comprimento do contexto e as horas de 5k GPU para pós-treinamento, o Deepseek-V3 custa apenas 2,788 milhões de horas de GPU para seu treinamento completo. Supondo que o preço do aluguel da GPU H800 seja de US $ 2 por hora da GPU, nosso custo total dos custos é de apenas US $ 5,576 milhões. Observe que os custos acima mencionados incluem apenas o treinamento oficial do DeepSeek-V3, excluindo os custos associados a pesquisas anteriores e experimentos de ablação em arquiteturas, algoritmos ou dados.

Nossa principal contribuição inclui:

Arquitetura: Estratégia inovadora de balanceamento de carga e objetivo de treinamento

- No topo da arquitetura eficiente do Deepseek-V2, pioneiros em uma estratégia livre de perdas auxiliares para balanceamento de carga, o que minimiza a degradação do desempenho que surge do incentivo ao balanceamento de carga.
- Investigamos um objetivo de previsão de vários toques (MTP) e provamos benéfico para modelar o desempenho. Também pode ser usado para decodificação especulativa para aceleração de inferência.

Pré-treinamento: em direção à eficiência de treinamento final

- Projetamos uma estrutura de treinamento de precisão mista do FP8 e, pela primeira vez, valide a viabilidade e a eficácia do treinamento FP8 em um modelo extremamente em larga escala.
- Através do co-design de algoritmos, estruturas e hardware, superamos o gargalo da comunicação no treinamento de MOE entre nós, alcançando a sobreposição de comunicação de computação quase cheia. Isso aumenta significativamente nossa eficiência de treinamento e reduz os custos de treinamento, permitindo -nos ampliar ainda mais o tamanho do modelo sem sobrecarga adicional.
- A um custo econômico de apenas 2,664m H800 GPU Hours, concluímos o pré-treinamento do Deepseek-V3 em tokens 14,8T, produzindo o modelo básico de código aberto mais forte. Os estágios de treinamento subsequentes após o pré-treinamento requerem apenas 0,1 milhão de horas de GPU.

Pós-treinamento: Destilação do Conhecimento de Deepseek-R1

- Introduzimos uma metodologia inovadora para destilar as capacidades de raciocínio do modelo de longa cadeia de pensamento (COT), especificamente de um dos modelos da série R1 Deepseek, em LLMs padrão, particularmente Deepseek-V3. Nosso oleoduto incorpora elegantemente o

Os padrões de verificação e reflexão de R1 no Deepseek-V3 e melhora notavelmente seu desempenho de raciocínio. Enquanto isso, também mantemos o controle sobre o estilo de saída e o comprimento do Deepseek-V3.

Resumo dos resultados da avaliação central

• Conhecimento: (1) em benchmarks educacionais como MMLU, MMLU-Pro e GPQA, Deepseek-V3 supera todos os outros modelos de código aberto, alcançando 88,5 em MMLU, 75,9 em MMLU-Pro e 59.1 em GPQA. Seu desempenho é comparável aos principais modelos de código fechado, como GPT-4O e Claude-Sonnet-3.5, estreitando a lacuna entre os modelos de código aberto e de código fechado nesse domínio. (2) Para benchmarks de factualidade, o Deepseek-V3 demonstra desempenho superior entre os modelos de código aberto no SimpleQA e no Chinese Simpleqa. Enquanto ele segue atrás do GPT-4O e do Claude-Sonnet-3.5 no conhecimento factual em inglês (SimpleQA), supera esses modelos no conhecimento factual chinês (chinês SimpleQA), destacando sua força no conhecimento factual chinês.

• Código, matemática e raciocínio: (1) Deepseek-V3 atinge o desempenho de última geração em benchmarks relacionados à matemática entre todos os modelos de código aberto e de código fechado que não têm uma base aberta. Notavelmente, ele supera até a previsão de O1 em benchmarks específicos, como o Math-500, demonstrando seus recursos robustos de raciocínio matemático. (2) Nas tarefas relacionadas à codificação, o Deepseek-V3 surge como o modelo de melhor desempenho para codificar os benchmarks de concorrência, como o LivecodeBench, solidificando sua posição como modelo principal nesse domínio. Para tarefas relacionadas à engenharia, enquanto o DeepSeek-V3 tem um desempenho ligeiramente abaixo do Claude-Sonnet-3.5, ele ainda supera todos os outros modelos por uma margem significativa, demonstrando sua competitividade em diversos parâmetros técnicos.

No restante deste artigo, primeiro apresentamos uma exposição detalhada da nossa arquitetura do modelo Deepseek-V3 (Seção 2). Posteriormente, apresentamos nossas infraestruturas, abrangendo nossos clusters de computação, a estrutura de treinamento, o suporte ao treinamento FP8, a estratégia de implantação de inferência e nossas sugestões sobre o design futuro de hardware. Em seguida, descrevemos nosso processo de pré-treinamento, incluindo a construção de dados de treinamento, configurações hiper-parâmetro, técnicas de extensão de contexto de longo prazo, as avaliações associadas e algumas discussões (seção 4). Depois disso, discutimos nossos esforços no pós-treinamento, que incluem ajuste fino supervisionado (SFT), aprendizado de reforço (RL), as avaliações correspondentes e discussões (seção 5). Por fim, concluímos este trabalho, discutimos as limitações existentes do Deepseek-V3 e propomos direções potenciais para pesquisas futuras (seção 6).

2. Arquitetura

Introduzimos pela primeira vez a arquitetura básica do Deepseek-V3, apresentada por atenção latente de várias cabeças (MLA) (Deepseek-AI, 2024C) para inferência eficiente e Deepseekmoe (Dai et al., 2024) para treinamento econômico. Em seguida, apresentamos um objetivo de treinamento de previsão com vários toques (MTP), que observamos para melhorar o desempenho geral nos benchmarks de avaliação. Para outros detalhes menores não mencionados explicitamente, o DeepSeek-V3 adere às configurações do DeepSeek-V2 (Deepseek-AI, 2024C).

2.1. Arquitetura básica

A arquitetura básica do Deepseek-V3 ainda está dentro da estrutura do transformador (Vaswani et al., 2017). Para inferência eficiente e treinamento econômico, o Deepseek-V3 também adota o MLA e o Deepseekmoe, que foram minuciosamente validados por Deepseek-V2. Comparado com o Deepseek-V2, uma exceção é que também introduzimos um balanceamento de carga livre de perda de perda auxiliar

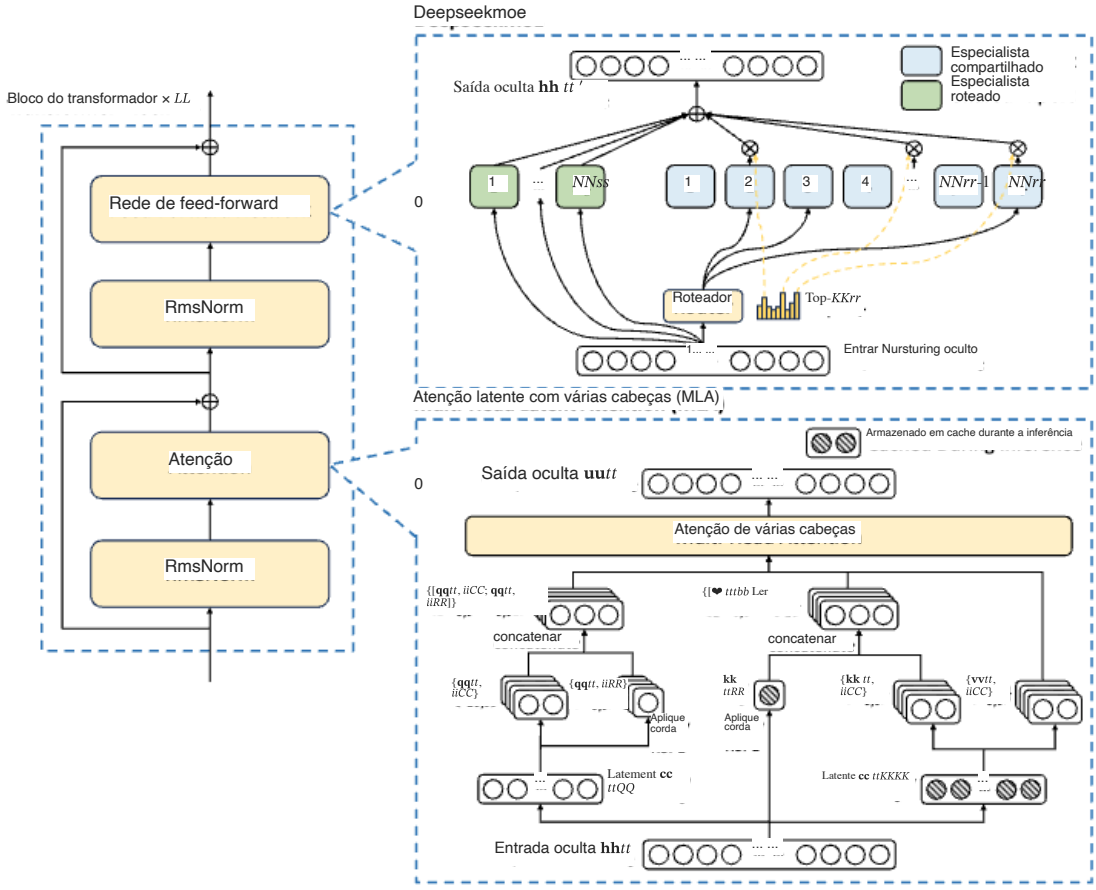


Figura 2 | Ilustração da arquitetura básica de Deepseek-V3. Após o Deepseek-V2, adotamos o MLA e o Deepseekmoe para obter inferência eficiente e treinamento econômico.

Estratégia (Wang et al., 2024a) para que o Deepseekmoe mitigue a degradação do desempenho induzida pelo esforço para garantir o equilíbrio da carga. A Figura 2 ilustra a arquitetura básica do Deepseek-V3, e revisaremos brevemente os detalhes do MLA e Deepseekmoe nesta seção.

2.1.1. Atenção latente de várias cabeças

Por atenção, o Deepseek-V3 adota a arquitetura do MLA. Vamos denotar a dimensão de incorporação, denotar o número de cabeças de atenção, denotar a dimensão por cabeça e h$t \in$ Rd denotar a entrada de atenção para o token t em uma determinada camada de atenção. O núcleo do MLA é a compressão da junta de baixo rank para chaves e valores de atenção para reduzir o cache do valor-chave (KV) durante a inferência:

$$\text{c.Fogo hu } (WW \text{ h}t, \tag{1}$$

$$[\text{K}Ct, 1; \text{k}Ct, 2; ...; \text{k}Ct, nh] = \text{k}Ct = WU\ K\ \text{c}KVt, \tag{2}$$

$$\text{K.= Corda } (W\ KR\text{h}t), \tag{3}$$

$$\text{k}t, i = [\text{k}Ct, i; \text{K}Rt], \tag{4}$$

$$[\text{V}Ct, 1; \text{V}Ct, 2; ...; \text{v}Ct, nh] = \text{v}Ct = WUV\ \text{c}KVt, \tag{5}$$

Σ

Onde c$KV \in$ Rdc é o vetor latente compactado para chaves e valores; dc ($\ll dhnh$) indica o KV dimensão de compressão; $W\,DKV \in$ R$dc \times d$ denota a matriz de projeção abaixo; $WU\,K$, $WUV \in$ R$dh\ nh \times dc$ são as matrizes de projeção para cima para chaves e valores, respectivamente; $W\,KR \in$ R$dRh \times d$ é a matriz usada para produzir a chave desacoplada que carrega incorporação posicional rotativa (corda) (Su et al., 2024); Corda (·) indica a operação que aplica matrizes de corda; e [·; ·] Denota concatenação. Observe que para o MLA, apenas os vetores de caixa azul (ou seja, CKVt e KRt) precisam ser armazenados em cache durante a geração,

o que resulta em cache KV reduzido significativamente, mantendo o desempenho comparável à atenção padrão de várias cabeças (MHA) (Vaswani et al., 2017).

Para as consultas de atenção, também realizamos uma compactação de baixo rank, que pode reduzir a memória de ativação durante o treinamento:

$$\text{c}Qt = W\,DQ\ \text{h}t, \tag{6}$$

$$[\text{Q}Ct, 1; \text{Q}Ct, 2; ...; \text{q}Ct, nh] = \text{q}Ct = WUQ\ \text{c}Qt, \tag{7}$$

$$[\text{Q}Rt, 1; \text{q}Rt, 2; ...; \text{Q}Rt, nh] = \text{q}Rt = \text{roup}\ (WQR\text{c}Qt), \tag{8}$$

$$\text{q}t, i = [\text{q}Ct, i; \text{q}Rt, i], \tag{9}$$

onde C$Qt \in$ R$d'c$ é o vetor latente compactado para consultas; $d'c$ ($\ll dhnh$) indica a dimensão da compressão de consulta; $W\,DQ \in$ R$d'c \times d$, $WUQ \in$ R$dh\ nh \times d'c$ são as matrizes de projeção e projeção de baixo para consultas, respectivamente; e $WQR \in$ R$dRh\ nh \times d'c$ é a matriz para produzir as consultas dissociadas que carregam corda.

Por fim, as consultas de atenção (qt, i), as chaves (kj, i) e os valores (vCj, i) são combinados para produzir a saída final de atenção ut:

$$\text{O}t, i = t \quad \text{Qv}\ _{e,}\ \text{J}, \quad j = 1\ \text{softmax}j^{\text{J}}\ {}_{,}dh + dRh) \quad \text{v}Cj, i, \tag{10}$$

$$\text{U}t = WO\ [\text{o}t, 1; \text{O}t, 2; ...; \text{o}t, nh], \tag{11}$$

onde $WO \in$ R$d \times dh\ nh$ denota a matriz de projeção de saída.

2.1.2. Deepseekmoe com balanceamento de carga livre de perdas auxiliares

Arquitetura básica de Deepseekmoe. Para redes de feed-forward (FFNs), o Deepseek-V3 emprega a arquitetura Deepseekmoe (Dai et al., 2024). Comparado com arquiteturas MOE tradicionais como Gshard (Lepikhin et al., 2021), o Deepseekmoe usa especialistas de granulação mais fina e isola alguns especialistas como compartilhados. Vamos denotar a entrada FFN do token t-TH, calculamos a saída FFN H't da seguinte maneira:

$$\text{h}'t = \text{u}t + Ns\ \ \text{Ffn}\ (s\ s)\ i\ \ (\text{u}t) + Nr\ \ i = 1\ gi,\ t\ \text{ffn}\ \ (\text{u}t), \quad i = 1 \quad (r)\ i \tag{12}$$

$$gi,\ t = \begin{matrix} g'i,\ t\ Nr\ j = \\ 1\ g'j,\ t\ t\ g\ g \\ g\ g\ g\ g\ g\ g\ g\ g \\ g\ g\ g\ g \end{matrix} \tag{13}$$

$$g'i,\ t \quad si,\ t, \quad si,\ t \in \text{Topk}\ (\{sj,\ t \mid 1 \leqslant j \leqslant Nr\},\ Kr),), \quad = (\quad 0, \quad \text{de outra forma}, \tag{14}$$

$$si\ t = \text{sigmoid u}\ i, \tag{15}$$

∑

onde Ns e Nr denotam o número de especialistas compartilhados e especialistas roteados, respectivamente; Ffn (s) i (·)

e FFN (r) i (·) denotam o especialista compartilhado e o i i-TH Roted Expert, respectivamente; Kr denota

o número de especialistas em roteamento ativado; gi, t é o valor de gatagem para o i i i i nt especialista; si, t é a afinidade de token-to-Expert; Ei é o vetor centróide do i i-Th Roted Expert; e Topk (·, K) denota o conjunto compreendendo K pontuações mais altas entre os escores de afinidade calculados para o token t-TH e todos os especialistas roteados. Um pouco diferente de Deepseek-V2, o Deepseek-V3 usa a função sigmóide para calcular os escores de afinidade e aplica uma normalização entre todos os escores de afinidade selecionados para produzir os valores de bloqueio.

Balanceamento de carga livre de perda de perda auxiliar. Para os modelos MOE, uma carga especializada desequilibrada levará ao colapso do roteamento (Shazeer et al., 2017) e diminuirá a eficiência computacional em cenários com paralelismo especializado. As soluções convencionais geralmente dependem da perda auxiliar (Fedus et al., 2021; Lepikhin et al., 2021) para evitar carga desequilibrada. No entanto, uma perda auxiliar muito grande prejudicará o desempenho do modelo (Wang et al., 2024a). Para obter uma melhor troca entre o equilíbrio de carga e o desempenho do modelo, pioneiros em uma estratégia de balanceamento de carga livre de perda de perda de perda (Wang et al., 2024a) para garantir o equilíbrio de carga. Para ser específico, introduzimos um termo de polarização bi para cada especialista e adicionamos-o às pontuações de afinidade correspondentes si, t para determinar o roteamento mais importante:

$$g'i,\ t = \begin{cases} si,\ t, & si,\ t + bi \in \in \text{Topk}\ (\{sj,\ t + bj \mid 1 \leqslant \leqslant j\ Nr\},\ Kr), \\ 0, & \text{de outra forma.} \end{cases} \tag{16}$$

Observe que o termo de polarização é usado apenas para roteamento. O valor de bloqueio, que será multiplicado com a saída FFN, ainda é derivado da pontuação original de afinidade si, t. Durante o treinamento, continuamos monitorando a carga especializada em todo o lote de cada etapa de treinamento. No final de cada etapa, diminuiremos o termo de polarização por γ se o especialista correspondente estiver sobrecarregado e aumentá-lo por γ se o especialista correspondente for submarcado, onde γ é um hiper-parâmetro chamado velocidade de atualização de polarização. Através do ajuste dinâmico, o Deepseek-V3 mantém a carga especializada equilibrada durante o treinamento e alcança melhor desempenho do que os modelos que incentivam o equilíbrio da carga por meio de perdas auxiliares puras.

Perda auxiliar complementar em sequência. Embora Deepseek-V3 depende principalmente do Estratégia Auxiliar-Loss Livre para Equilíbrio de Carga, para evitar desequilíbrio extremo em uma única sequência, também empregamos uma perda de equilíbrio complementar em sequência:

$$\text{Lbal} = \alpha \underset{i = 1}{\overset{Nr}{}} fi \\ Pi, \tag{17}$$

$$fi = \frac{\overline{t} = 1\ 1\ si,\ t \in \text{Topk}\ (\{sj,\ t \mid 1 \leqslant j \leqslant Nr\},\ Kr),)}{NrKrT}, \tag{18}$$

$$s'i,\ t = \underset{t}{\overset{si,\ Nr\ Nr}{= 1\ sj,\ t}}, \tag{19}$$

$$Pi = \underset{1T}{\overset{T\Sigma}{t = 1}} \\ s'i,\ t, \tag{20}$$

onde o fator de equilíbrio α é um hiper-parâmetro, que receberá um valor extremamente pequeno para o Deepseek-V3; 1 (·) indica a função indicadora; e T denota o número de tokens em uma sequência. A perda de equilíbrio em sequência incentiva a carga especializada em cada sequência a ser equilibrada.

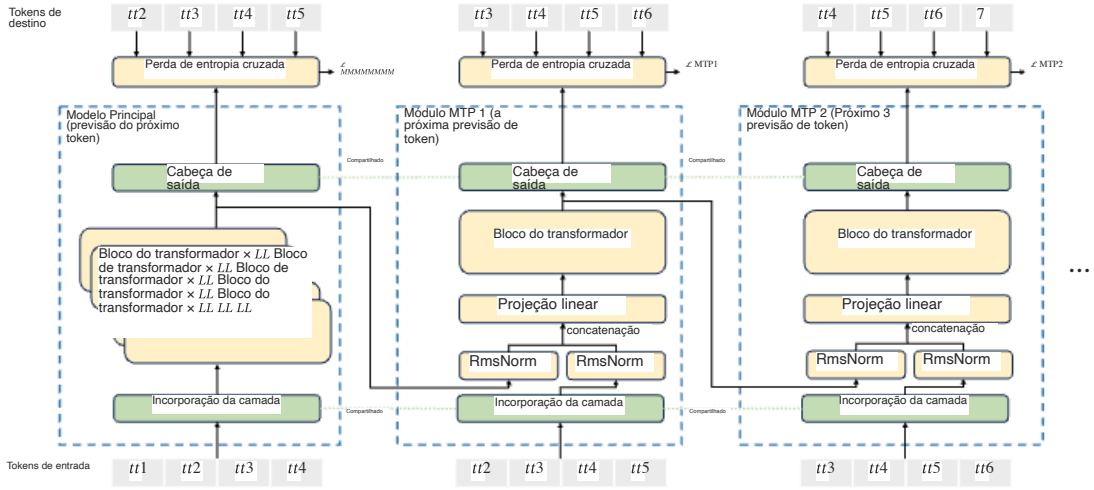


Figura 3 | Ilustração de nossa implementação de previsão de vários toques (MTP). Mantemos a cadeia causal completa para a previsão de cada token a cada profundidade.

Roteamento limitado por nó. Como o roteamento limitado pelo dispositivo usado pelo Deepseek-V2, o Deepseek-V3 também usa um mecanismo de roteamento restrito para limitar os custos de comunicação durante o treinamento. Em suma, garantimos que cada token seja enviado para a maioria dos nós, que são selecionados de acordo com a soma das pontuações de afinidade mais altas dos especialistas distribuídos em cada nó. Sob essa restrição, nossa estrutura de treinamento MOE pode quase obter sobreposição completa de computação-comunicação.

Não cair de token. Devido à estratégia efetiva de balanceamento de carga, o Deepseek-V3 mantém um bom equilíbrio de carga durante seu treinamento completo. Portanto, o DeepSeek-V3 não diminui os tokens durante o treinamento. Além disso, também implementamos estratégias de implantação específicas para garantir o balanço de carga de inferência; portanto, o Deepseek-V3 também não solta tokens durante a inferência.

2.2. Previsão com vários toques

Inspirado em Gloeckle et al. (2024), investigamos e estabelecemos um objetivo de previsão de vários toques (MTP) para Deepseek-V3, que estende o escopo de previsão a vários tokens futuros em cada posição. Por um lado, um objetivo MTP densifica os sinais de treinamento e pode melhorar a eficiência dos dados. Por outro lado, a MTP pode permitir que o modelo seja pré-planejamento de suas representações para obter uma melhor previsão de tokens futuros. A Figura 3 ilustra nossa implementação do MTP. Diferente de Gloeckle et al. (2024), que Paralelly prevê D tokens adicionais usando cabeças de saída independentes, prevemos sequencialmente tokens adicionais e mantemos a cadeia causal completa em cada profundidade de previsão. Introduzimos os detalhes de nossa implementação MTP nesta seção.

Módulos MTP. Para ser específico, nossa implementação MTP usa módulos seqüenciais para prever D tokens adicionais. O módulo k-TH MTP consiste em uma camada de incorporação compartilhada EMB (·), uma cabeça de saída compartilhada (·), um bloco de transformador trm$k$ (·) e uma matriz de projeção $Mk \in$ Rd × 2d. Para o token de entrada ti, na profundidade da previsão, primeiro combinamos a representação do token i i i i-TH no (k-1) -th profundidade hk-1 i rd e a incorporação do (i + k) -th token Emb (ti + k) ∈ Rd

Σ

com a projeção linear:

$$h' = M_k [\text{rmsNorm}(h_k - 1\ i\ ); \text{rmsNorm}(\text{EMB}(\text{midefiente} \quad (21)$$

onde [·; ·] Denota concatenação. Especialmente, quando k = 1, hk -1 i refere -se à representação dada pelo modelo principal. Observe que para cada módulo MTP, sua camada de incorporação é compartilhada com o modelo principal. O H'ki combinado serve como a entrada do bloco de transformador na k k-fth profundidade para

Produza a representação de saída na profundidade atual Hki:

$$h_{k1:T-k} = \text{trm}_k(h'_{k1:T-k}), \quad (22)$$

onde T representa o comprimento da sequência de entrada e i: j denota a operação de fatiamento (inclusive os limites esquerdo e direito). Finalmente, tomando Hki como entrada, a cabeça de saída compartilhada calculará a distribuição de probabilidade para o k k k^{o} token de previsão adicional $Pki+1+k \in \mathbb{R}V$, onde V é o tamanho do vocabulário:

$$P_{ki+k+1} = \text{OUTHEAD}(H_{ki}). \quad (23)$$

A cabeça de saída étora (·) mapeia linearmente a representação em logits e, posteriormente, aplica a função Softmax (·) para calcular as probabilidades de previsão do k k th token adicional. Além disso, para cada módulo MTP, sua cabeça de saída é compartilhada com o modelo principal. Nosso princípio de manter a cadeia causal de previsões é semelhante à de Eagle (Li et al., 2024b), mas seu objetivo principal é a decodificação especulativa (Leviathan et al., 2023; Xia et al., 2023), enquanto utilizamos MTP para melhorar o treinamento.

Objetivo de treinamento MTP. Para cada profundidade de previsão, calculamos uma perda de entropia cruzada l$k$mtp:

$$L_{k\text{mtp}} = \text{crossentrepropy}(P_{k2+k:T+1}, t_{2+k:T+1}) = -\frac{1}{T}\sum_{i=2+k}^{T+1} \log P_{ki}[t_i], \quad (24)$$

onde T denota o comprimento da sequência de entrada, ti denota o token de verdade no solo na posição i-TH, e Pki [ti] denota a probabilidade de previsão correspondente de ti, dada pelo módulo k-TH MTP. Finalmente, calculamos a média das perdas do MTP em todas as profundidades e a multiplicamos por um fator de ponderação λ para obter a perda geral de MTP LMTP, que serve como um objetivo de treinamento adicional para Deepseek-V3:

$$L_{\text{mtp}} = \frac{\lambda}{D}\sum_{k=1}^{D} l_{k\text{mtp}}. \quad (25)$$

MTP em inferência. Nossa estratégia MTP visa melhorar o desempenho do modelo principal; portanto, durante a inferência, podemos descartar diretamente os módulos MTP e o modelo principal pode funcionar de forma independente e normalmente. Além disso, também podemos redirecionar esses módulos MTP para decodificação especulativa para melhorar ainda mais a latência da geração.

3. Infraestruturas

3.1. Clusters de computação

O Deepseek-V3 é treinado em um cluster equipado com GPUs 2048 NVIDIA H800. Cada nó no cluster H800 contém 8 GPUs conectados por NVLink e NVSwitch dentro dos nós. Em diferentes nós, as interconexões Infiniband (IB) são utilizadas para facilitar as comunicações.

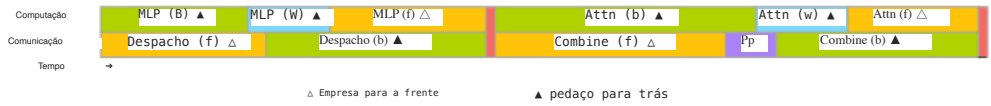


Figura 4 I Estratégia sobreposta para um par de pedaços individuais para frente e para trás (os limites dos blocos de transformadores não estão alinhados). A Orange indica para a frente, o verde denota "para trás para a entrada", o azul denota "para trás para pesos", o roxo denota a comunicação pp e o vermelho denota barreiras. Tanto a comunicação tudo para todos quanto a PP pode ser totalmente oculta.

3.2. Estrutura de treinamento

O treinamento do Deepseek-V3 é apoiado pela estrutura HAI-LLM, uma estrutura de treinamento eficiente e leve criada por nossos engenheiros desde o início. No geral, o Deepseek-V3 aplica o paralelismo de pipeline de 16 vias (PP) (Qi et al., 2023a), paralismo especialista em 64 vias (EP) (Lepikhin et al., 2021) abrange 8 nós e zero- 1 paralelismo de dados (DP) (Rajb- Handeri et al., 2020).

Para facilitar o treinamento eficiente do Deepseek-V3, implementamos engenharia meticulosa otimizações. Em primeiro lugar, projetamos o algoritmo dualpipe para o paralelismo eficiente da pipeline. Comparado com os métodos PP existentes, o Dualpipe possui menos bolhas de pipeline. Mais importante, ele se sobrepõe às fases de computação e comunicação nos processos para a frente e para trás, abordando assim o desafio da sobrecarga de comunicação pesada introduzida pelo paralelismo especialista em nós. Em segundo lugar, desenvolvemos kernels de comunicação entre todos os nó eficientes para utilizar completamente as larguras de banda IB e NVLink e conservar multiprocessadores de streaming (SMS) dedicados à comunicação. Finalmente, otimizamos meticulosamente a pegada de memória durante o treinamento, permitindo-nos treinar Deepseek-V3 sem usar o paralelismo tensorado caro (TP).

3.2.1. Dualpipe e Computação-Comunicação Sobreposição

Para o Deepseek-V3, a sobrecarga de comunicação introduzida pelo paralelismo especialista em nós resulta em uma taxa de computação / comunicação ineficiente de aproximadamente 1: 1. Para enfrentar esse desafio, projetamos um algoritmo inovador de paralelismo de pipeline chamado Dualpipe, que não apenas acelera o treinamento do modelo, sobrepondo efetivamente as fases de computação para frente e para trás, mas também reduz as bolhas de tubulação.

A idéia principal do Dualpipe é sobrepor a computação e a comunicação em um par de pedaços individuais para frente e para trás. Para ser específico, dividimos cada pedaço em quatro componentes: atenção, despacho para todos, MLP e combinação de todos. Especialmente, para um pedaço para trás, a atenção e o MLP são divididos em duas partes, para trás para entrada e para trás para pesos, como em Zerobubble (Qi et al., 2023b). Além disso, temos um componente de comunicação PP. Como ilustrado na Figura 4, para um par de pedaços para frente e para trás, reorganizamos esses componentes e ajustamos manualmente a proporção de SMS de GPU dedicados à comunicação versus computação. Nesta estratégia sobreposta, podemos garantir que a comunicação de todos os todos e PP possa ser totalmente oculta durante a execução. Dada a estratégia de sobreposição eficiente, a programação completa do dualpipe é ilustrada na Figura 5. Emprega um agendamento bidirecional de pipeline, que alimenta micro-lotes de ambas as extremidades do pipeline simultaneamente e uma parcela significativa das comunicações pode ser totalmente sobreposta. Essa sobreposição também garante que, à medida que o modelo aumenta ainda mais, desde que mantemos uma taxa constante de computação / comunicação, ainda podemos empregar especialistas em grãos finos em todos os nós enquanto alcançamos uma sobrecarga de comunicação quase zero, quase zero .

Figura 5 | Exemplo de agendamento de dualpipe para 8 pp ranks e 20 micro lotes em duas direções. Os micro-lotes na direção inversa são simétricos para os que estão na direção para a frente, por isso omitimos o ID do lote para a simplicidade da ilustração. Duas células fechadas por uma borda preta compartilhada se sobrepõem mutuamente em computação e comunicação.

Método	Bolha	Parâmetro	Ativação
1f1b	$(PP$ - 1) $(F + B)$	1×	PP
Zb1p	$(PP$ - 1) $(F + B$ - $2W)$	1×	PP
Dualpipe (nosso)	$(PP2$ - 1) $(F$ & $B + B$ - $3W)$	2×	PP + 1

Tabela 2 | Comparação de bolhas de pipeline e uso de memória em diferentes oleodutos paralelos

Métodos. F denota o tempo de execução de um pedaço para a frente, B denota o tempo de execução de um pedaço total para trás, W denota o tempo de execução de um pedaço de "atraso para pesos" e F & B indica o tempo de execução de dois pedaços para frente e para trás mutuamente.

Além disso, mesmo em cenários mais gerais sem uma carga pesada de comunicação, a dupla ainda exibe vantagens de eficiência. Na Tabela 2, resumimos as bolhas de pipeline e o uso da memória em diferentes métodos PP. Conforme mostrado na tabela, comparado com ZB1p (Qi et al., 2023b) e 1F1b (Harlap et al., 2018), o dualpipe reduz significativamente as bolhas de pipeline enquanto aumentam apenas a memória de ativação de pico em $1PP$ vezes. Embora o Dualpipe exija manter duas cópias dos parâmetros do modelo, isso não aumenta significativamente o consumo de memória, pois usamos um grande tamanho de EP durante o treinamento. Comparado com a quimera (Li e Hoefler, 2021), o dualpipe exige apenas que os estágios e micro-lotes do pipeline sejam divisíveis por 2, sem exigir que os micro-lotes sejam divisíveis por estágios de pipeline. Além disso, para o dualpipe, nem as bolhas nem a memória de ativação aumentarão à medida que o número de micro-lotes cresce.

3.2.2. Implementação eficiente da comunicação entre todos os nó para todos

Para garantir um desempenho computacional suficiente para o Dualpipe, personalizamos os kernels de comunicação entre todos os nó eficientes (incluindo despacho e combinação) para economizar o número de SMs dedicados à comunicação. A implementação dos kernels é co-projetada com o algoritmo MOE Gating e a topologia de rede do nosso cluster. Para ser específico, em nosso cluster, as GPUs cruzadas são totalmente interconectadas com o IB e as comunicações intra-nós são tratadas via NVLink. O NVLink oferece uma largura de banda de 160 GB/s, aproximadamente 3,2 vezes a de Ib (50 GB/s). Para aproveitar efetivamente as diferentes larguras de banda de IB e NVLink, limitamos cada token a ser despachado a no máximo 4 nós, reduzindo assim o tráfego de IB. Para cada token, quando sua decisão de roteamento for tomada, ele será transmitido primeiro via IB para as GPUs com o mesmo índice de nó em seus nós de destino. Quando atingir os nós de destino, nos esforçaremos para garantir que ele seja encaminhado instantaneamente via NVLink para GPUs específicas que hospedam seus especialistas alvo, sem serem bloqueados pela chegada posteriormente. Dessa forma, as comunicações via IB e NVLink são totalmente sobrepostas e cada token pode selecionar com eficiência uma média de 3,2 especialistas por nó sem incorrer em sobrecarga adicional do NVLink. Isso implica que, embora Deepseek-V3

Seleciona apenas 8 especialistas roteados na prática, ele pode aumentar esse número para um máximo de 13 especialistas (4 nós × 3.2 especialistas/nó), preservando o mesmo custo de comunicação. No geral, sob essa estratégia de comunicação, apenas 20 SMs são suficientes para utilizar completamente as larguras de banda do IB e NVLink.

Em detalhes, empregamos a técnica de especialização de Warp (Bauer et al., 2014) e partimos 20 SMS em 10 canais de comunicação. Durante o processo de despacho, (1) o envio de IB, (2) o encaminhamento IB-NVLink e (3) o recebimento do NVLink são tratados pelas respectivas dobras. O número de urdidores alocados a cada tarefa de comunicação é ajustado dinamicamente de acordo com a carga de trabalho real em todos os SMs. Da mesma forma, durante o processo de combinação, (1) envio de nvlink, (2) encaminhamento e acumulação de nvlink-to-IB e (3) recebimento e acumulação de IB também são tratados por urdidores ajustados dinamicamente. Além disso, a despacho e a combinação de kernels se sobrepõem ao fluxo de computação, por isso também consideramos seu impacto em outros núcleos de computação de SM. Especificamente, empregamos instruções personalizadas de PTX (execução de encadeamento paralelo) e ajustamos automaticamente o tamanho do pedaço de comunicação, o que reduz significativamente o uso do cache L2 e a interferência em outros SMs.

3.2.3. EXTREMAM

Para reduzir a pegada de memória durante o treinamento, empregamos as seguintes técnicas.

Recomputação de RMSNorm e MLA Up-Projeção. Recomputamos todas as operações do RMSNorm e as projeções do MLA durante a propagação de volta, eliminando assim a necessidade de armazenar persistentemente suas ativações de saída. Com uma sobrecarga menor, essa estratégia reduz significativamente os requisitos de memória para armazenar ativações.

Média móvel exponencial na CPU. Durante o treinamento, preservamos a média de movimentação exponencial (EMA) dos parâmetros do modelo para a estimativa precoce do desempenho do modelo após a decaimento da taxa de aprendizado. Os parâmetros EMA são armazenados na memória da CPU e são atualizados de forma assíncrona após cada etapa de treinamento. Este método nos permite manter os parâmetros da EMA sem incorrer em memória adicional ou sobrecarga de tempo.

Chefe de incorporação e saída compartilhado para previsão de vários toques. Com a estratégia de dualpipe,

Implantamos as camadas mais rasas (incluindo a camada de incorporação) e camadas mais profundas (incluindo a cabeça de saída) do modelo na mesma classificação de PP. Esse arranjo permite o compartilhamento físico de parâmetros e gradientes, da cabeça de incorporação e saída compartilhada, entre o módulo MTP e o modelo principal. Esse mecanismo de compartilhamento físico aumenta ainda mais nossa eficiência de memória.

3.3. Treinamento FP8

Inspirado por avanços recentes no treinamento de baixa precisão (Dettmers et al., 2022; Noune et al., 2022; Peng et al., 2023b), propomos uma estrutura de precisão mista de granulação fina utilizando o formato de dados FP8 para treinamento profundo- V3. Embora o treinamento de baixa precisão tenha grande promessa, geralmente é limitado pela presença de discrepantes em ativações, pesos e gradientes (Fishman et al., 2024; He et al.; Sun et al., 2024). Embora um progresso significativo tenha sido feito em quantização de influência (Frantar et al., 2022; Xiao et al., 2023), existem relativamente poucos estudos demonstrando aplicação bem-sucedida de técnicas de baixa precisão no modelo de linguagem em larga escala

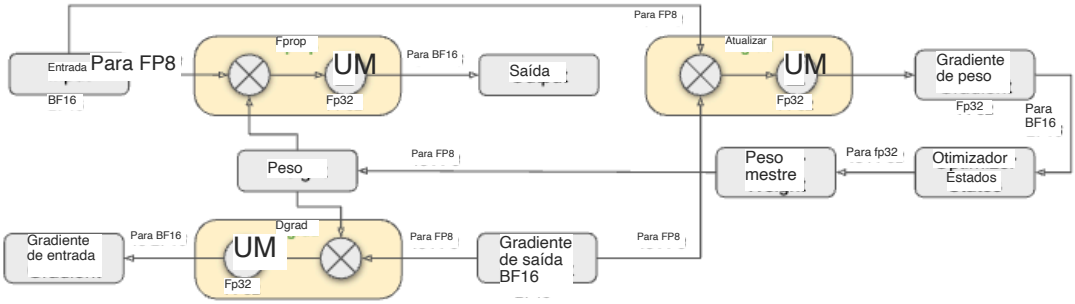


Figura 6 | A estrutura geral de precisão mista com o formato de dados FP8. Para esclarecimentos, apenas o operador linear é ilustrado.

pré-treinamento (Fishman et al., 2024). Para abordar esse desafio e estender efetivamente a faixa dinâmica do formato FP8, introduzimos uma estratégia de quantização de granulação fina: agrupamento em ladrilhos com elementos $1 \times Nc$ ou agrupamento em bloco com elementos $Nc \times Nc$. A sobrecarga de desquantidade associada é amplamente mitigada sob nosso processo de acumulação de precisão aumentada, um aspecto crítico para alcançar a multiplicação de matriz geral FP8 precisa (GEMM). Além disso, para reduzir ainda mais a sobrecarga de memória e comunicação no treinamento de MOE, abrigamos e despacamos atividades no FP8, enquanto armazenamos estados de otimizador de baixa precisão no BF16. Validamos a estrutura de precisão mista FP8 proposta em duas escalas de modelo semelhantes a Deepseek-V2-Lite e Deepseek-V2, treinamento para aproximadamente 1 trilhão de tokens (consulte mais detalhes no Apêndice B.1). Notavelmente, em comparação com a linha de base BF16, o erro de perda relativa do nosso modelo de treinamento FP8 permanece consistentemente abaixo de 0,25%, um nível de nível dentro da faixa aceitável de aleatoriedade do treinamento.

3.3.1. Estrutura de precisão mista

Com base em técnicas amplamente adotadas em treinamento de baixa precisão (Kalamkar et al., 2019; Narang et al., 2017), propomos uma estrutura de precisão mista para o treinamento do FP8. Nesse trabalho, a maioria das operações de densidade de computação é realizada no FP8, enquanto algumas operações importantes são estrategicamente mantidas em seus formatos de dados originais para equilibrar a eficiência do treinamento e a estabilidade numérica. A estrutura geral é ilustrada na Figura 6.

Em primeiro lugar, para acelerar o treinamento do modelo, a maioria dos núcleos de computação, ou seja, operações da GEMM, é implementada na precisão do FP8. Essas operações da GEMM aceitam os tensores FP8 como entradas e produzem saídas em BF16 ou FP32. Conforme representado na Figura 6, todas as três GEMMs associadas ao operador linear, a saber, FPROP (Pass Forward), DGRAD (PASS de ativação para trás) e WGrad (Pass Backward Pass), são executadas no FP8. Esse design teoricamente dobra a velocidade computacional em comparação com o método BF16 original. Além disso, o FP8 WGrad Gemm permite que as ativações sejam armazenadas no FP8 para uso no passe para trás. Isso reduz significativamente o consumo de memória.

Apesar da vantagem de eficiência do formato FP8, certos operadores ainda exigem uma precisão mais alta devido à sua sensibilidade a cálculos de baixa precisão. Além disso, alguns operadores de baixo custo também podem utilizar uma precisão mais alta com uma sobrecarga insignificante para o custo geral de treinamento. Por esse motivo, após investigações cuidadosas, mantemos a precisão original (por exemplo, BF16 ou FP32) para os seguintes componentes: o módulo de incorporação, a cabeça de saída, módulos de portamento MOE, operadores de normalização e operadores de atenção. Essas retenções direcionadas de alta precisão garantem dinâmica de treinamento estável para Deepseek-V3. Para garantir ainda mais a estabilidade numérica, armazenamos os pesos principais, gradientes de peso e estados de otimizadores em maior precisão. Enquanto

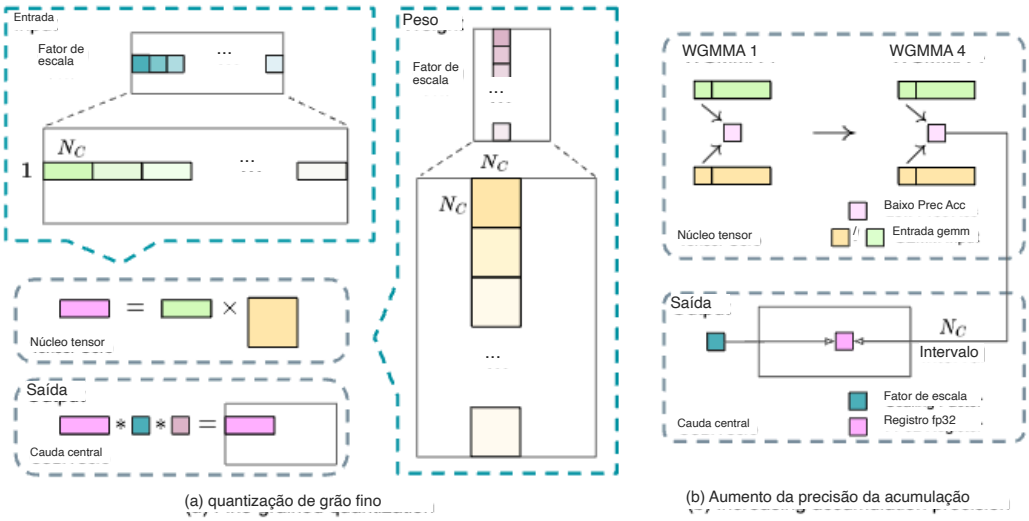


Figura 7 | (a) Propomos um método de quantização de granulação fina para mitigar os erros de quantização causados por discrepantes de recurso; Para a simplicidade da ilustração, apenas o FPROP é ilustrado. (b) Em conjunto com nossa estratégia de quantização, melhoramos a precisão do FP8 GEMM, promovendo os núcleos de CUDA em um intervalo de NC = 128 elementos MMA para o acúmulo de alta precisão.

Esses componentes de alta precisão incorrem em algumas despesas gerais de memória, seu impacto pode ser minimizado através de sharding eficiente em várias fileiras de DP em nosso sistema de treinamento distribuído.

3.3.2. Precisão aprimorada da quantização e multiplicação

Com base em nossa estrutura de FP8 de precisão mista, introduzimos várias estratégias para aprimorar a precisão do treinamento de baixa precisão, concentrando-se no método de quantização e no processo de multiplicação.

Quantização de granulação fina. Nas estruturas de treinamento de baixa precisão, os transbordamentos e os fluxos de baixa precisão são desafios comuns devido à faixa dinâmica limitada do formato FP8, que é restringido por seus bits reduzidos de expoente. Como prática padrão, a distribuição de entrada está alinhada ao intervalo representável do formato FP8, dimensionando o valor absoluto máximo do tensor de entrada para o valor máximo representável de FP8 (Narang et al., 2017). Esse método torna o treinamento de baixa precisão altamente sensível aos valores extremos de ativação, o que pode degradar fortemente a precisão da quantização. Para resolver isso, propomos um método de quantização de granulação fina que aplica a escala em um nível mais granular. Conforme ilustrado na Figura 7 (a), (1) para ativações, agrupamos e escalamos elementos em uma base de 1x128 (ou seja, por token por 128 canais); e (2) para pesos, agrupamos e escalamos elementos com base em bloqueio de 128x128 (ou seja, por 128 canais de entrada por 128 canais de saída). Essa abordagem garante que o processo de quantização possa acomodar melhor os outliers, adaptando a escala de acordo com grupos menores de elementos. No Apêndice B.2, discutimos ainda mais a instabilidade do treinamento quando agruparmos e escalamos ativações em bloqueio da mesma maneira que a quantização de pesos.

Uma modificação-chave em nosso método é a introdução de fatores de escala por grupo ao longo da dimensão interna das operações da GEMM. Essa funcionalidade não é suportada diretamente no FP8 GEMM padrão. No entanto, combinado com nossa estratégia de acumulação precisa de FP32, pode

ser implementado com eficiência.

Notavelmente, nossa estratégia de quantização de granulação fina é altamente consistente com a idéia de formatos de mi-croscaling (Rouhani et al., 2023b), enquanto os núcleos de tensor de NVIDIA de próxima geração (série Blackwell) anunciaram o suporte para formatos de microscaling com com formatos Granularidade de quantização menor (NVIDIA, 2024A). Esperamos que nosso design possa servir como uma referência para trabalhos futuros para acompanhar as mais recentes arquiteturas da GPU.

Aumento da precisão da acumulação. As operações GEMM de baixa precisão geralmente sofrem de problemas de desbaste, e sua precisão depende em grande parte do acúmulo de alta precisão, que é comumente realizado em uma precisão de FP32 (Kalamkar et al., 2019; Narang et al., 2017). No entanto, observamos que a precisão do acúmulo de gemm FP8 no NVIDIA H800 GPUS é limitada a reter cerca de 14 bits, o que é significativamente menor que a precisão do acúmulo de FP32. Esse problema se tornará mais pronunciado quando a dimensão interna K for grande (Wortsman et al., 2023), um cenário típico em treinamento de modelo em larga escala, onde o tamanho do lote e a largura do modelo são aumentados. Tomando operações GEMM de duas matrizes aleatórias com k = 4096, por exemplo, em nosso teste preliminar, a precisão limitada de acumulação nos núcleos tensores resulta em um erro relativo máximo de quase 2%. Apesar desses problemas, a precisão limitada de acumulação ainda é a opção padrão em algumas estruturas FP8 (NVIDIA, 2024b), restringindo severamente a precisão do treinamento.

Para resolver esse problema, adotamos a estratégia de promoção aos núcleos da CUDA para maior precisão (Thakkar et al., 2023). O processo é ilustrado na Figura 7 (b). Para ser específico, durante a execução do MMA (Matrix multiply-acumulado) em núcleos tensores, os resultados intermediários são acumulados usando a largura de bit limitada. Uma vez atingido um intervalo de NC, esses resultados parciais serão copiados para os registros FP32 nos núcleos CUDA, onde a acumulação de FP32 de precisão total é realizada. Como mencionado anteriormente, nossa quantização de granulação fina aplica fatores de escala por grupo ao longo da dimensão interna K. Esses fatores de escala podem ser multiplicados com eficiência nos núcleos CUDA como processo de desquantização com um custo computacional adicional mínimo.

Vale a pena notar que essa modificação reduz a taxa de instrução WGMMA (Matrix de nível de matriz no nível WarpGroup) para um único grupo de WarpGroup. No entanto, na arquitetura H800, é típico para dois WGMMA persistirem simultaneamente: enquanto um grupo de Warproup executa a operação de promoção, o outro é capaz de executar a operação de MMA. Esse design permite a sobreposição das duas operações, mantendo alta utilização de núcleos tensores. Com base em nossos experimentos, a configuração NC = 128 elementos, equivalente a 4 WGMMAs, representa o intervalo de acumulação mínimo que pode melhorar significativamente a precisão sem introduzir uma sobrecarga substancial.

Mantissa sobre expoentes. Em contraste com o formato Hybrid FP8 adotado por trabalhos anteriores (Nvidia, 2024b; Peng et al., 2023b; Sun et al., 2019b), que usa E4M3 (expoente de 4 bits e Mantissa de 3 bits) no FPROP e E5M2 (Expoente de 5 bits e Mantissa de 2 bits) em dgrad e wgrad, adotamos o formato E4M3 em todos os tensores para maior precisão. Atribuímos a viabilidade dessa abordagem à nossa estratégia de quantização fina, ou seja, escala de azulejos e blocos. Ao operar em grupos de elementos menores, nossa metodologia compartilha efetivamente bits de expoente entre esses elementos agrupados, mitigando o impacto da faixa dinâmica limitada.

Quantização online. A quantização tardia é empregada em quadros de quantização tensores (NVIDIA, 2024B; Peng et al., 2023b), que mantém uma história do absoluto máximo absoluto

valores entre iterações anteriores para inferir o valor atual. Para garantir escalas precisas e simplificar a estrutura, calculamos o valor absoluto máximo on-line para cada bloco de peso de 1x128 ou bloco de peso 128x128. Com base nisso, derivamos o fator de escala e, em seguida, quantizamos a ativação ou peso on -line no formato FP8.

3.3.3. Armazenamento e comunicação de baixa precisão

Em conjunto com nossa estrutura de treinamento FP8, reduzimos ainda mais o consumo de memória e a sobrecarga de comunicação, compactando ativações em cache e estados de otimizadores em formatos de menor precisão.

Estados do otimizador de baixa precisão. Adotamos o formato de dados BF16 em vez de FP32 para rastrear o primeiro e o segundo momento no otimizador ADAMW (Loshchilov e Hutter, 2017), sem incorrer em degradação de desempenho observável. No entanto, os pesos mestre (armazenados pelo otimizador) e gradientes (usados para acumulação de tamanho de lotes) ainda são retidos no FP32 para garantir a estabilidade numérica ao longo do treinamento.

Ativação de baixa precisão. Como ilustrado na Figura 6, a operação WGrad é realizada no FP8. Para reduzir o consumo de memória, é uma opção natural para ativações de cache no formato FP8 para o passe para trás do operador linear. No entanto, são tomadas considerações especiais em vários operadores para treinamento de alto custo de alta precisão:

(1) Entradas do linear após o operador de atenção. Essas ativações também são usadas no passe para trás do operador de atenção, o que o torna sensível à precisão. Adotamos um formato de dados E5M6 personalizado exclusivamente para essas ativações. Além disso, essas ativações serão convertidas de um ladrilho de quantização 1x128 para um bloco de 128x1 no passe para trás. Para evitar a introdução de erros de quantização extra, todos os fatores de escala são redondos, ou seja, potência integral de 2.

(2) Entradas do operador Swiglu em MOE. Para reduzir ainda mais o custo da memória, cache as entradas do operador Swiglu e recomputamos sua saída no passe para trás. Essas ativações também são armazenadas no FP8 com nosso método de quantização de grão fino, atingindo um equilíbrio entre a eficiência da memória e a precisão computacional.

Comunicação de baixa precisão. A largura de banda de comunicação é um gargalo crítico no treinamento de modelos MOE. Para aliviar esse desafio, quantizamos a ativação antes da MOE Up-Projeções no FP8 e depois aplicamos componentes de expedição, que é compatível com o FP8 FPROP nas projeções Moe Up. Como as entradas do linear após o operador de atenção, os fatores de escala para essa ativação são poder integrante de 2. Uma estratégia semelhante é aplicada ao gradiente de ativação antes das projeções de MOE. Para os componentes combinados para a frente e para trás, mantemos -os no BF16 para preservar a precisão do treinamento em partes críticas do pipeline de treinamento.

3.4. Inferência e implantação

Implantamos Deepseek-V3 no cluster H800, onde as GPUs dentro de cada nó são interconectadas usando o NVLink, e todas as GPUs no cluster estão totalmente interconectadas via IB. Para garantir simultaneamente o objetivo do nível de serviço (SLO) para serviços on-line e alta taxa de transferência, empregamos a seguinte estratégia de implantação que separa os estágios de preenchimento e decodificação.

3.4.1. Preenchimento

A unidade de implantação mínima do estágio de pré -enchimento consiste em 4 nós com 32 GPUs. A parte da atenção emprega paralelismo de tensor de 4 vias (TP4) com paralelismo de sequência (SP), combinado com o paralelismo de dados de 8 vias (DP8). Seu pequeno tamanho de TP de 4 limita a sobrecarga da comunicação TP. Para a parte MOE, usamos o paralelismo de especialistas em 32 vias (EP32), o que garante que cada especialista processe um tamanho de lote suficientemente grande, aumentando assim a eficiência computacional. Para a comunicação MOE para tudo, usamos o mesmo método do treinamento: primeiro transferência de tokens através dos nós via IB e depois encaminhamos entre as GPUs intra-nó via NVLink. Em particular, usamos o paralelismo de tensor de 1 vias para os MLPs densos em camadas rasas para salvar a comunicação TP.

Para alcançar o balanceamento de carga entre diferentes especialistas na parte MOE, precisamos garantir que cada GPU processe aproximadamente o mesmo número de tokens. Para esse fim, introduzimos uma estratégia de implantação de especialistas redundantes, que duplica especialistas em carga de alta carga e os implanta de forma redundante. Os especialistas em alta carga são detectados com base nas estatísticas coletadas durante a implantação on-line e são ajustadas periodicamente (por exemplo, a cada 10 minutos). Depois de determinar o conjunto de especialistas redundantes, reorganizamos cuidadosamente os especialistas entre as GPUs dentro de um nó com base nas cargas observadas, esforçando-se para equilibrar a carga nas GPUs o máximo possível sem aumentar a sobrecarga de comunicação cruzada em todos os nó. Para a implantação do Deepseek-V3, estabelecemos 32 especialistas redundantes para o estágio de pré-enchimento. Para cada GPU, além dos 8 especialistas originais que ele hospeda, ela também hospedará um especialista redundante adicional.

Além disso, no estágio de pré-enchimento, para melhorar a taxa de transferência e ocultar a sobrecarga da comunicação de todos os todos e TP, processamos simultaneamente dois micro-lotes com cargas de trabalho computacionais semelhantes, sobrepondo a atenção e o MOE de um micro-lote com o despacho e combinar outro.

Finalmente, estamos explorando uma estratégia de redundância dinâmica para especialistas, onde cada GPU hospeda mais especialistas (por exemplo, 16 especialistas), mas apenas 9 serão ativados durante cada etapa de inferência. Antes do início da operação tudo para tudo em cada camada, calculamos o esquema de roteamento globalmente ideal em tempo real. Dada a computação substancial envolvida no estágio de pré - enchimento, a sobrecarga de calcular esse esquema de roteamento é quase insignificante.

3.4.2. Decodificação

Durante a decodificação, tratamos o especialista compartilhado como roteado. Nessa perspectiva, cada token selecionará 9 especialistas durante o roteamento, onde o especialista compartilhado é considerado como uma carga pesada que sempre será selecionada. A unidade de implantação mínima do estágio de decodificação consiste em 40 nós com 320 GPUs. A parte da atenção emprega TP4 com SP, combinada com DP80, enquanto a parte MOE usa EP320. Para a parte MOE, cada GPU hospeda apenas um especialista e 64 GPUs são responsáveis por hospedar especialistas redundantes e especialistas compartilhados. A comunicação para todos as peças de despacho e combinação é realizada por meio de transferências diretas de ponto a ponto sobre o IB para obter baixa latência. Além disso, aproveitamos a tecnologia IBGDA (NVIDIA, 2022) para minimizar ainda mais a latência e aumentar a eficiência da comunicação.

Semelhante ao preenchimento, determinamos periodicamente o conjunto de especialistas redundantes em um determinado intervalo, com base na carga de especialista estatístico do nosso serviço on -line. No entanto, não precisamos reorganizar especialistas, pois cada GPU hospeda apenas um especialista. Também estamos explorando a estratégia de redundância dinâmica para decodificar. No entanto, isso requer uma otimização mais cuidadosa do algoritmo que calcula o esquema de roteamento globalmente ideal e a fusão com o kernel de despacho para reduzir a sobrecarga.

Além disso, para aprimorar a taxa de transferência e ocultar a sobrecarga da comunicação em todos, também estamos explorando o processamento de dois micro-lotes com cargas de trabalho computacionais semelhantes simultaneamente no estágio de decodificação. Ao contrário do preenchimento, a atenção consome uma parte maior do tempo no estágio de decodificação. Portanto, sobrepunhamos a atenção de um micro-lote com o Dispatch+Moe+Combine de outro. No estágio de decodificação, o tamanho do lote por especialista é relativamente pequeno (geralmente dentro de 256 tokens), e o gargalo é o acesso à memória e não a computação. Como a parte MOE precisa apenas carregar os parâmetros de um especialista, a sobrecarga de acesso à memória é mínima; portanto, o uso de menos SMS não afetará significativamente o desempenho geral. Portanto, para evitar impactar a velocidade de computação da parte da atenção, podemos alocar apenas uma pequena porção do SMS para despachar+MOE+combinar.

3.5. Sugestões sobre design de hardware

Com base em nossa implementação do esquema de treinamento de comunicação e FP8, propomos as seguintes sugestões sobre o design de chips aos fornecedores de hardware da IA.

3.5.1. Hardware de comunicação

No Deepseek-V3, implementamos a sobreposição entre computação e comunicação para ocultar a latência de comunicação durante o cálculo. Isso reduz significativamente a dependência da largura de banda de comunicação em comparação com a computação e comunicação em série. No entanto, a implementação atual de comunicação depende de SMS caros (por exemplo, alocamos 20 dos 132 SMs disponíveis na GPU H800 para esse fim), o que limitará a taxa de transferência computacional. Além disso, o uso de SMS para a comunicação resulta em ineficiências significativas, pois os núcleos tensores permanecem inteiramente subutilizados.

Atualmente, o SMS executa principalmente as seguintes tarefas para comunicação para tudo:

- Encaminhando dados entre o domínio IB (Infiniband) e NVLink enquanto agregam o tráfego de IB destinado a várias GPUs no mesmo nó de uma única GPU.
- Transporte de dados entre buffers RDMA (regiões de memória GPU registradas) e buffers de saída/saída.
- Execução de operações de redução para todas as combinações para tudo. • Gerenciando o layout de memória de grão fino durante a transferência de dados em grama para múltiplos

Especialistas em todo o domínio IB e NVLink.

Aspiramos a ver futuros fornecedores desenvolvendo hardware que descarrega essas tarefas de comunicação da valiosa unidade de computação SM, servindo como co-processador de GPU ou co-processador de rede como a Nvidia Sharp Graham et al. (2016). Além disso, para reduzir a complexidade da programação de aplicativos, pretendemos que esse hardware unifique as redes IB (escala) e NVLink (escala) da perspectiva das unidades de computação. Com essa interface unificada, as unidades de computação podem facilmente realizar operações como leitura, gravação, multicast e reduzir em todo o domínio ib-nvlink unificado por meio de envio de solicitações de comunicação com base em primitivas simples.

3.5.2. Calcule o hardware

Maior precisão de acumulação de gemm fp8 em núcleos tensores. Na atual implementação do núcleo do tensor da arquitetura NVIDIA Hopper, o FP8 GEMM (Matriz Geral Multiply) emprega acumulação de ponto fixo, alinhando os produtos Mantissa, mudando a mudança da direita com base no expoente máximo antes da adição. Nossos experimentos revelam que ele usa apenas os 14 mais altos

Bits de cada produto Mantissa após a mudança à direita do preenchimento de sinal e truques bits excedendo esse intervalo. No entanto, por exemplo, para obter resultados precisos de FP32 do acúmulo de multiplicações de 32 FP8 × FP8, é necessária pelo menos uma precisão de 34 bits. Assim, recomendamos que os projetos futuros de chips aumentem a precisão da acumulação em núcleos tensores para suportar a acumulação de precisão total ou selecione uma largura de bits de acumulação apropriada de acordo com os requisitos de precisão dos algoritmos de treinamento e inferência. Essa abordagem garante que os erros permaneçam dentro dos limites aceitáveis, mantendo a eficiência computacional.

Suporte para quantização em ladrilhos e blocos. As GPUs atuais suportam apenas a quantização por tensor, sem o suporte nativo para quantização de granulação fina, como nossa quantização em azulejos e bloqueios. Na implementação atual, quando o intervalo NC for atingido, os resultados parciais serão copiados de núcleos tensoros para núcleos CUDA, multiplicados pelos fatores de escala e adicionados aos registros FP32 nos núcleos CUDA. Embora a sobrecarga de desquantização seja significativamente mitigada combinada com nossa estratégia de acumulação precisa de FP32, os movimentos frequentes de dados entre núcleos tensores e núcleos de CUDA ainda limitam a eficiência computacional. Portanto, recomendamos chips futuros para apoiar a quantização de granulação fina, permitindo que os núcleos tensores recebam fatores de escala e implementem o MMA com a escala de grupo. Dessa maneira, todo o acúmulo e desquantização parcial da soma podem ser concluídas diretamente dentro de núcleos tensores até que o resultado final seja produzido, evitando movimentos frequentes de dados.

Suporte para quantização on -line. As implementações atuais lutam para apoiar efetivamente a quantização on -line, apesar de sua eficácia demonstrada em nossa pesquisa. No processo existente, precisamos ler 128 valores de ativação de BF16 (a saída da computação anterior) a partir de HBM (alta memória de largura de banda) para quantização, e os valores quantizados de FP8 são então escritos de volta ao HBM, apenas para serem lidos novamente para MMA . Para abordar essa ineficiência, recomendamos que os futuros chips integrem acesso FP8 Cast e TMA (acelerador de memória tensoros) em uma única operação fundida, para que a quantização possa ser concluída durante a transferência de ativações da memória global para a memória compartilhada, evitando leituras frequentes de memória e gravação . Também recomendamos apoiar uma instrução de elenco no nível de urdidura para acelerar, o que facilita ainda mais a melhor fusão da normalização da camada e do elenco FP8. Como alternativa, uma abordagem de computação quase memória pode ser adotada, onde a lógica de computação é colocada perto do HBM. Nesse caso, os elementos BF16 podem ser fundidos diretamente para o FP8, à medida que são lidos do HBM na GPU, reduzindo o acesso à memória fora do chip em aproximadamente 50%.

Suporte para operações da GEMM transposta. A arquitetura atual torna complicado fundir a transposição da matriz com operações da GEMM. Em nosso fluxo de trabalho, as ativações durante o passe para a frente são quantizadas em ladrilhos 1x128 FP8 e armazenadas. Durante o passe atrasado, a matriz precisa ser lida, desquantalizada, transposta, requantizada em telhas de 128x1 e armazenada no HBM. Para reduzir as operações de memória, recomendamos chips futuros para permitir leituras diretas transpostas de matrizes da memória compartilhada antes da operação do MMA, para as precisões necessárias no treinamento e na inferência. Combinada com a fusão da conversão de formato FP8 e acesso ao TMA, esse aprimoramento otimizará significativamente o fluxo de trabalho de quantização.

4. Pré-treinamento

4.1. Construção de dados

Comparado com o Deepseek-V2, otimizamos o corpus de pré-treinamento, aumentando a proporção de amostras matemáticas e de programação, enquanto expandiu a cobertura multilíngue além do inglês e do chinês. Além disso, nosso pipeline de processamento de dados é refinado para minimizar a redundância, mantendo a diversidade de corpus. Inspirado por Ding et al. (2024), implementamos o método de embalagem de documentos para integridade dos dados, mas não incorporamos a mascaramento de atenção cruzada durante o treinamento. Finalmente, o corpus de treinamento para Deepseek-V3 consiste em 14,8 t tokens de alta qualidade e diversos em nosso tokenizador.

No processo de treinamento do DeepSeekCoder-V2 (Deepseek-AI, 2024a), observamos que a estratégia de preenchimento (FIM) não compromete a capacidade de previsão do próximo token, ao mesmo tempo em que o modelo prevê com precisão o texto do meio com base no contexto pistas. No alinhamento com o DeepSeekCoder-V2, também incorporamos a estratégia FIM no pré-treinamento do Deepseek-V3. Para ser específico, empregamos a estrutura de prefixo-sufix-middle (PSM) para estruturar dados da seguinte maneira:

<| FIM_BEGIN |> fPRE <| FIM_HOLE |> fSUF <| FIM_END |> fMIDDLE <| eos_token |>.

Essa estrutura é aplicada no nível do documento como parte do processo de pré-embalagem. A estratégia FIM é aplicada a uma taxa de 0,1, consistente com a estrutura do PSM.

O tokenizer da Deepseek-V3 emprega BPE no nível de byte (Shibata et al., 1999) com um vocabulário prolongado de 128k tokens. Os dados do pré -speatizer e de treinamento para o nosso tokenizador são modificados para otimizar a eficiência multilíngue de compressão. Além disso, em comparação com o DeepSeek-V2, o novo pré-speatizador apresenta tokens que combinam pontuações e quebras de linha. No entanto, esse truque pode introduzir o viés de limite do token (Lundberg, 2023) quando o modelo processa avisos de várias linhas sem quebras de linha de terminal, principalmente para avisos de avaliação de poucos anos. Para resolver esse problema, dividimos aleatoriamente uma certa proporção de tais tokens combinados durante o treinamento, o que expõe o modelo a uma variedade mais ampla de casos especiais e atenua esse viés.

4.2. Hiper-parâmetros

Modelo Hiper-parâmetros. Definimos o número de camadas do transformador como 61 e a dimensão oculta para 7168. Todos os parâmetros aprendidos são inicializados aleatoriamente com um desvio padrão de 0,006. No MLA, definimos o número de cabeças de atenção nh para 128 e a dimensão por cabeça dh para 128. A dimensão da compressão KV é definida como 512, e a dimensão da compressão de consulta $d'c$ é definida como 1536. Para as perguntas desacopladas e chave, definimos a dimensão por cabeça dRh para 64. Substituímos todos os FFNs, exceto as três primeiras camadas por camadas MOE. Cada camada MOE consiste em 1 especialista compartilhado e 256 especialistas roteados, onde a dimensão oculta intermediária de cada especialista é 2048. Entre os especialistas roteados, 8 especialistas serão ativados para cada token, e cada token será garantido para ser enviado para o máximo 4 nós. A profundidade de previsão de vários toques D é definida como 1, ou seja, além do próximo token exato, cada token preverá um token adicional. Como Deepseek-V2, o DeepSeek-V3 também emprega camadas adicionais de rmsNorm após os vetores latentes compactados e multiplica fatores de escala adicionais nos gargalos de largura. Sob essa configuração, o Deepseek-V3 compreende 671b parâmetros totais, dos quais 37b são ativados para cada token.

Treinando hiper-parâmetros. Empregamos o otimizador ADAMW (Loshchilov e Hutter, 2017) com hiper-parâmetros definidos como β1 = 0,9, β2 = 0,95 e peso_decay = 0,1. Definimos o comprimento máximo da sequência como 4K durante o pré-treinamento e o pré-treino Deepseek-V3 em tokens 14,8T. Quanto a

A programação da taxa de aprendizado, primeiro aumentamos linearmente de 0 para 2,2 × 10−4 durante as primeiras 2 mil etapas. Em seguida, mantivemos uma taxa de aprendizado constante de 2,2 × 10−4 até que o modelo consome 10T de treinamento. Posteriormente, decairmos gradualmente a taxa de aprendizado para 2,2 × 10−5 em tokens 4,3T, seguindo uma curva de decaimento de cosseno. Durante o treinamento dos tokens finais de 500B, mantemos uma taxa de aprendizado constante de 2,2 × 10 a 5 nos primeiros 333b e mudamos para outra taxa de aprendizado constante de 7,3 × 10 a 6 nos tokens 167b restantes. A norma de recorte do gradiente é definida como 1.0. Empregamos uma estratégia de agendamento de tamanho em lote, onde o tamanho do lote é gradualmente aumentado de 3072 para 15360 no treinamento dos primeiros 469B tokens e depois mantém 15360 no treinamento restante. Aproveitamos o paralelismo do pipeline para implantar diferentes camadas de um modelo em diferentes GPUs e, para cada camada, os especialistas roteados serão implantados uniformemente em 64 GPUs pertencentes a 8 nós. Quanto ao roteamento limitado pelo nó, cada token será enviado para no máximo 4 nós (ou seja, $M = 4$). Para balanceamento de carga livre de perdas auxiliares, definimos a velocidade de atualização de viés γ a 0,001 nos primeiros tokens 14.3T e para 0,0 para os tokens restantes de 500b. Para a perda de saldo, definimos α para 0,0001, apenas para evitar desequilíbrio extremo em qualquer sequência única. O peso da perda de MTP λ é definido como 0,3 para os primeiros tokens 10T e para 0,1 para os tokens 4,8T restantes.

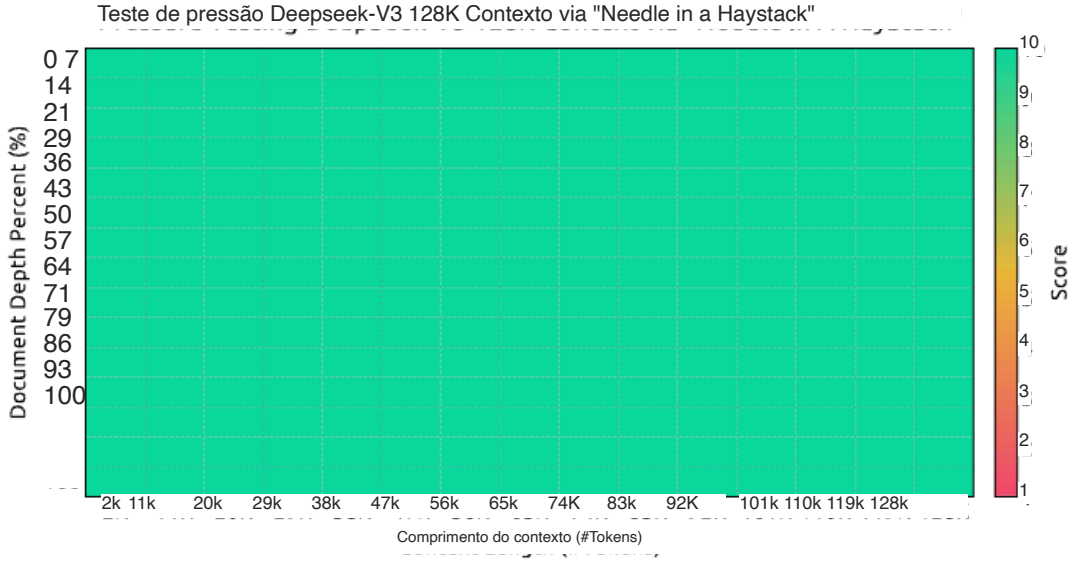


Figura 8 | Resultados da avaliação nos testes de "agulha em um palheiro" (NIAH). Deepseek-V3 tem um bom desempenho em todos os comprimentos das janelas de contexto de até 128k.

4.3. Extensão de contexto longo

Adotamos uma abordagem semelhante ao Deepseek-V2 (Deepseek-AI, 2024C) para permitir recursos de contexto longo no Deepseek-V3. Após o estágio de pré-treinamento, aplicamos fios (Peng et al., 2023a) para extensão de contexto e realizamos duas fases de treinamento adicionais, cada uma compreendendo 1000 etapas, para expandir progressivamente a janela de contexto de 4K para 32k e depois para 128K. A configuração de fios é consistente com a usada no Deepseek-V2, sendo aplicada exclusivamente à chave compartilhada dissociada kRt. Os hiper-parâmetros permanecem idênticos nas duas fases, com a escala $s = 40$, $\alpha = 1$, $\beta = 32$, e o fator de escala $\sqrt{t} = 0{,}1 \ln s + 1$. Na primeira fase, o comprimento da sequência é definido como 32K e o tamanho do lote é 1920. Durante a segunda fase, o comprimento da sequência é aumentado para 128k e o tamanho do lote é reduzido para 480. A taxa de aprendizado para ambas as fases é definida como 7,3 × 10-6, correspondendo à taxa de aprendizado final do estágio de pré-treinamento.

Através deste treinamento em extensão em duas fases, o Deepseek-V3 é capaz de lidar com entradas de até 128k de comprimento, mantendo um forte desempenho. A Figura 8 ilustra que o DeepSeek-V3, após o ajuste fino supervisionado, atinge o desempenho notável no teste "Anexa em um palheiro" (NIAH), demonstrando robustez consistente nos comprimentos das janelas de contexto de até 128k.

4.4. Avaliações

4.4.1. Benchmarks de avaliação

O modelo básico de Deepseek-V3 é pré-treinado em um corpus multilíngue com inglês e chinês, constituindo a maioria, por isso avaliamos seu desempenho em uma série de benchmarks principalmente em inglês e chinês, bem como em uma referência multilíngue. Nossa avaliação é baseada em nossa estrutura de avaliação interna integrada em nossa estrutura HAI-LLM. Os benchmarks considerados são categorizados e listados da seguinte forma, onde os benchmarks sublinhados estão em benchmarks chineses e duplos sub-linados são multilíngues:

Os conjuntos de dados de múltipla escolha múltipla incluem MMLU (Hendrycks et al., 2020), MMLU-Redux (Gema et al., 2024), Mmlu-pro (Wang et al., 2024b), Mmmlu (OpenAI, 2024b), C et - Eval (Huang et al., 2023) e Cmmlu (Li et al., 2023).

Os conjuntos de dados de entendimento e raciocínio de idiomas incluem Hellaswag (Zellers et al., 2019), Piqa (Bisk et al., 2020), ARC (Clark et al., 2018) e Bigbench Hard (BBH) (Suzgun et al., 2022) .

Os conjuntos de dados de respostas de perguntas de livro fechado incluem Triviaqa (Joshi et al., 2017) e Natu-Ralquestions (Kwiatkowski et al., 2019).

A Companhia de Label é incluída incluiu Race The et al. (2017), Drop (All et al., 2019), C3 (desde et al., 2019a) e CMRC (Whoe et al., 2019).

Os conjuntos de dados de desambiguação de referência incluem ClueWSC (Xu et al., 2020) e Winogrande Sakaguchi et al. (2019).

Os conjuntos de dados de modelagem de idiomas incluem pilha (Gao et al., 2020).

Os conjuntos de dados de entendimento e cultura chineses incluem CCPM (Li et al., 2021).

Os conjuntos de dados matemáticos incluem GSM8Ks (Cobbes et al., 2021), Math (Hendrycks et al., 2021), MGSM (Shi et al., 2023), CMATH (Wee et al., 2023).

Os conjuntos de dados de código incluíram Humaneval (Chen et al., 2021), LivecodeBench-Base (0801-1101) (Jain et al., 2024), MBPP (Austin et al., 2021) e CruxEval (Gu et al., 2024) .

Os exames padronizados incluem Agieval (Zhong et al., 2023). Observe que a Agieval inclui subconjuntos em inglês e chinês.

Seguindo nosso trabalho anterior (Deepseek-AI, 2024b, C), adotamos avaliação baseada em perplexidade para conjuntos de dados, incluindo Hellaswag, Piqa, Winogrande, Race-Middle, Race-High, Mmlu, Mmlu-Redux, Mmlu-Pro, Mmmlu, , Arco-easy, arco-desafio, C-eves, cmmlu, c3 e ccpm e avaliação baseada em geração de adoção para triviaqa, naturaisquestões, gota, matemática, gsm8k, mgsm, humaneval, mbpp, livecodebench-base, bruxval, churrasco, churrasco, churrasco, bbh, bbh, bbh, bbh, bbh, bbh, bbh, mbpp, viva , Agieval, Cluewsc, CMRC e CMATH. Além disso, realizamos avaliação baseada em moda de idioma para teste de pilha e usamos bits por byte (BPB) como a métrica para garantir uma comparação justa entre os modelos usando tokenizadores diferentes.

	Benchmark (métrica)	# Tiros	Deepseek-V2 Base	Qwen2.5 72b base	Chamada-3.1 Base 405b	Deepseek-V3 Base
	Arquitetura	-	Moe	Denso	Denso	Moe
	# Params ativados	-	21b	72b	405b	37b
	# Total Params	-	236b	72b	405b	671b
Inglês	Bateria de teste (BPB)	-	0.606	0.638	0.542	0.548
	BF (EM)	3-shot	78.8	79.8	82.9	87.5
	MMLU (EM)	5-shot	78.4	85.0	84.4	87.1
	MMLU-Redux (EM)	5-shot	75.6	83.2	81.3	86.2
	MMLU-Pro (EM)	5-shot	51.4	58.3	52.8	64.4
	Drop (F1)	3-shot	80.4	80.6	86.0	89.0
	Arc-leasy (em)	25 tiros	97.6	98.4	98.4	98.9
	ARC-Challenge (EM)	25 tiros	92.2	94.5	95.3	95.3
	HellaSwag (EM)	10-shot	87.1	84.8	89.2	88.9
	PIQA (EM)	0-shot	83.9	82.6	85.9	84.7
	WinoGrande (EM)	5-shot	86.3	82.3	85.2	84.9
	Race-Middle (EM)	5-shot	73.1	68.1	74.2	67.1
	RACE-High (EM)	5-shot	52.6	50.3	56.8	51.3
	TriviaQA (EM)	5-shot	80.0	71.9	82.7	82.9
	NaturalQuestions (EM)	5-shot	38.6	33.2	41.5	40.0
	AGIEval (EM)	0-shot	57.5	75.8	60.6	79.6
Código	Humaneval (passe@1)	0-shot	43.3	53.0	54.9	65.2
	Mbpp (passe@1)	3-shot	65.0	72.6	68.4	75.4
	LivecodeBench-Base (Pass@1)	3-shot	11.6	12.9	15.5	19.4
	CRUXEval-I (EM)	2-shot	52.5	59.1	58.5	67.3
	CRUXEval-O (EM)	2-shot	49.8	59.9	59.9	69.8
Matemática	GSM8K (EM)	8-shot	81.6	88.3	83.5	89.3
	MATH (EM)	4-shot	43.4	54.4	49.0	61.6
	MGSM (EM)	8-shot	63.6	76.2	69.9	79.8
	CMath (EM)	3-shot	78.7	84.5	77.3	90.7
chinês	CLUEWSC (EM)	5-shot	82.0	82.5	83.0	82.7
	C-Eval (EM)	5-shot	81.4	89.2	72.5	90.1
	CMMLU (EM)	5-shot	84.0	89.5	73.7	88.8
	CMRC (EM)	1 shot	77.4	75.8	76.0	76.3
	C3 (EM)	0-shot	77.4	76.7	79.7	78.6
	CCPM (EM)	0-shot	93.0	88.5	78.6	92.0
Mmmlu-não-inglês multilíngue (EM)		5-shot	64.0	74.8	73.8	79.4

Tabela 3 | Comparação entre os modelos de base de código aberto representativos de Deepseek-V3 e outros modelos de base aberta. Todos os modelos são avaliados em nossa estrutura interna e compartilham a mesma configuração de avaliação. As pontuações com uma lacuna não excedentes de 0,3 são consideradas no mesmo nível. Deepseek-V3-Base alcança o melhor desempenho na maioria dos benchmarks, especialmente em tarefas de matemática e código.

4.4.2. Resultados da avaliação

Na Tabela 3, comparamos o modelo básico de Deepseek-V3 com os modelos básicos de código aberto de última geração, incluindo Deepseek-V2-Base (Deepseek-AI, 2024C) (nosso lançamento anterior), Qwen2.5 72b Base (Qwen, 2024b) e Base LLAMA-3.1 405B (AI@meta, 2024b). Avaliamos todos esses modelos com nossa estrutura de avaliação interna e garantimos que eles compartilhem a mesma configuração de avaliação. Observe que, devido às alterações em nossa estrutura de avaliação nos últimos meses, o desempenho do DeepSeek-V2-BASE exibe uma pequena diferença em relação aos resultados relatados anteriormente. No geral, o DeepSeek-V3-BASE supera de forma abrangente de maneira profunda e baseada na base QWEN2.5 72B e supera a base LLAMA-3.1 405B na maioria dos benchmarks, tornando-se essencialmente o modelo de código aberto mais forte.

De uma perspectiva mais detalhada, comparamos o Deepseek-V3-Base com os outros modelos de base de código aberto individualmente. (1) Comparado com o Deepseek-V2-Base, devido às melhorias em nossa arquitetura de modelo, à expansão do tamanho do modelo e dos tokens de treinamento e ao aprimoramento da qualidade dos dados, Deepseek-V3-Base atinge um desempenho significativamente melhor, conforme esperado . (2) Comparado com a base QWEN2.5 72B, o modelo de código aberto chinês de última geração, com apenas metade dos parâmetros ativados, Deepseek-V3-Base também demonstra vantagens notáveis, especialmente em inglês, multilíngue, código e benchmarks matemáticos. Quanto aos benchmarks chineses, exceto a CMMLU, uma tarefa chinesa de múltiplas sujeitos de múltiplos sujeitos, Deepseek-V3-Base, também mostra melhor desempenho que Qwen2.5 72b. (3) Comparado com a base LLAMA-3.1 405B, o maior modelo de código aberto com 11 vezes os parâmetros ativados, o Deepseek-V3-BASE também exibe desempenho muito melhor em benchmarks multilíngues, de código e matemática. Quanto aos benchmarks de inglês e chinês, o Deepseek-V3-Base mostra desempenho competitivo ou melhor e é especialmente bom na BBH, MMLU-Série, Drop, C-Eval, CMMLU e CCPM.

Devido a nossas arquiteturas eficientes e otimizações abrangentes de engenharia, o DeepSeek-V3 alcança uma eficiência de treinamento extremamente alta. Sob nossa estrutura de treinamento e infraestruturas, o treinamento de Deepseek-V3 em cada trilhão de tokens requer apenas 180k H800 GPU Hours, o que é muito mais barato do que o treinamento de modelos densos 72b ou 405b.

Benchmark (métrica)	# Tiros	Pequeno moe Linha de base	Pequeno moe com mtp	Grande moe Linha de base	Grande moe com mtp
# Params ativados (inferência)	-	2.4b	2.4b	20.9b	20.9b
# Total Params (Inferência)	-	15.7b	15.7b	228.7b	228.7b
# Treinando tokens	-	1.33t	1.33t	540b	540b
Bateria de teste (BPB)	-	0.729	0.729	0.658	0.657
BF (EM)	3-shot	39.0	41.4	70.0	70.7
MMLU (EM)	5-shot	50.0	53.3	67.5	66.6
Drop (F1)	1 shot	39.2	41.3	68.5	70.6
TriviaQA (EM)	5-shot	56.9	57.7	67.0	67.3
NaturalQuestions (EM)	5-shot	22.7	22.3	27.2	28.5
Humaneval (passe@1)	0-shot	20.7	26.8	44.5	53.7
Mbpp (passe@1)	3-shot	35.8	36.8	61.6	62.2
GSM8K (EM)	8-shot	25.4	31.4	72.3	74.0
MATH (EM)	4-shot	10.7	12.6	38.6	39.8

Tabela 4 | Resultados de ablação para a estratégia MTP. A estratégia MTP aprimora consistentemente o desempenho do modelo na maioria dos benchmarks de avaliação.

4.5. Discussão

4.5.1. Estudos de ablação para previsão de vários toques

Na Tabela 4, mostramos os resultados da ablação para a estratégia MTP. Para ser específico, validamos a estratégia MTP no topo de dois modelos de linha de base em diferentes escalas. Na pequena escala, treinamos um modelo BASELING MOE, compreendendo parâmetros totais de 15,7b em tokens de 1,33T. Em larga escala, treinamos um modelo BASELING MOE compreendendo 228,7b parâmetros totais em tokens de 540b. Além deles, mantendo os dados de treinamento e as outras arquiteturas da mesma forma, anexamos um módulo MTP de 1 profundidade neles e treinamos dois modelos com a estratégia MTP para comparação. Observe que, durante a inferência, descartamos diretamente o módulo MTP; portanto, os custos de inferência dos modelos comparados são exatamente os mesmos. A partir da tabela, podemos observar que a estratégia MTP aprimora consistentemente o desempenho do modelo na maioria dos benchmarks de avaliação.

Benchmark (métrica)	# Tiros	Pequeno moe Aux-Loss-Loss-Libra-Livs-Libes-Libra-Libra	Pequeno moe livre de perda	Grande moe	Grande moe
# Params ativados	-	2.4b	2.4b	20.9b	20.9b
# Total Params	-	15.7b	15.7b	228.7b	228.7b
# Treinando tokens	-	1.33t	1.33t	578b	578b
Bateria de teste (BPB)	-	0.727	0.724	0.656	0.652
BF (EM)	3-shot	37.3	39.3	66.7	67.9
MMLU (EM)	5-shot	51.0	51.8	68.3	67.2
Drop (F1)	1 shot	38.1	39.0	67.1	67.1
TriviaQA (EM)	5-shot	58.3	58.5	66.7	67.7
NaturalQuestions (EM)	5-shot	23.2	23.4	27.1	28.1
Humaneval (passe@1)	0-shot	22.0	22.6	40.2	46.3
Mbpp (passe@1)	3-shot	36.6	35.8	59.2	61.2
GSM8K (EM)	8-shot	27.1	29.6	70.7	74.5
MATH (EM)	4-shot	10.9	11.1	37.2	39.6

Tabela 5 | Resultados de ablação para a estratégia de equilíbrio auxiliar-livre de perdas. Comparado com o método puramente auxiliar baseado em perda de perda, a estratégia livre de perdas auxiliares alcança consistentemente o melhor desempenho do modelo na maioria dos benchmarks de avaliação.

4.5.2. Estudos de ablação para a estratégia de equilíbrio Auxiliar-Loss Lives

Na Tabela 5, mostramos os resultados da ablação para a estratégia de equilíbrio auxiliar-sem perdas. Validamos essa estratégia no topo de dois modelos de linha de base em diferentes escalas. Na pequena escala, treinamos um modelo BASELING MOE, compreendendo parâmetros totais de 15,7b em tokens de 1,33T. Em larga escala, treinamos um modelo BASELING MOE compreendendo 228,7b parâmetros totais em tokens de 578b. Ambos os modelos de linha de base usam puramente perdas auxiliares para incentivar o balanço de carga e usar a função sigmóide de bloqueio com a normalização da afinidade de Top-K. Seus hiper-parâmetros para controlar a força das perdas auxiliares são os mesmos que Deepseek-V2-Lite e Deepseek-V2, respectivamente. No topo desses dois modelos de linha de base, mantendo os dados de treinamento e as outras arquiteturas da mesma forma, removemos todas as perdas auxiliares e introduzimos a estratégia de equilíbrio Auxiliar-Loss Livre para comparação. A partir da tabela, podemos observar que a estratégia livre de perdas auxiliares alcança consistentemente um melhor desempenho do modelo na maioria dos benchmarks de avaliação.

4.5.3. Balanço de carga em lote vs. Balanço de carga em termos de sequência

A principal distinção entre o equilíbrio auxiliar-livre e a perda auxiliar em termos de sequência está em seu escopo de equilíbrio: em termos de lotes versus em termos de sequência. Comparado com a perda auxiliar em termos de sequência, o equilíbrio em lote impõe uma restrição mais flexível, pois não aplica o equilíbrio no domínio em cada sequência. Essa flexibilidade permite que os especialistas se especializem melhor em diferentes domínios. Para validar isso, registramos e analisamos a carga especializada de uma linha de base baseada em perda auxiliar de 16B e um modelo livre de perdas auxiliares de 16B em diferentes domínios no conjunto de testes de pilha. Conforme ilustrado na Figura 9, observamos que o modelo livre de perdas auxiliares demonstra maiores padrões de especialização de especialistas conforme o esperado.

Para investigar melhor a correlação entre essa flexibilidade e a vantagem no desempenho do modelo, também projetamos e validamos uma perda auxiliar em lote que incentive o saldo de carga em cada lote de treinamento em vez de em cada sequência. Os resultados experimentais mostram que, ao atingir um nível semelhante de balanço de carga em lote, a perda auxiliar em lote também pode obter um desempenho de modelo semelhante ao método livre de perda de perda auxiliar. Para ser específico, em nossos experimentos com modelos de 1B MoE, as perdas de validação são: 2.258 (usando uma perda auxiliar em termos de sequência), 2.253 (usando o método Auxiliar-Liber-Free) e 2.253 (usando um lote em lotes

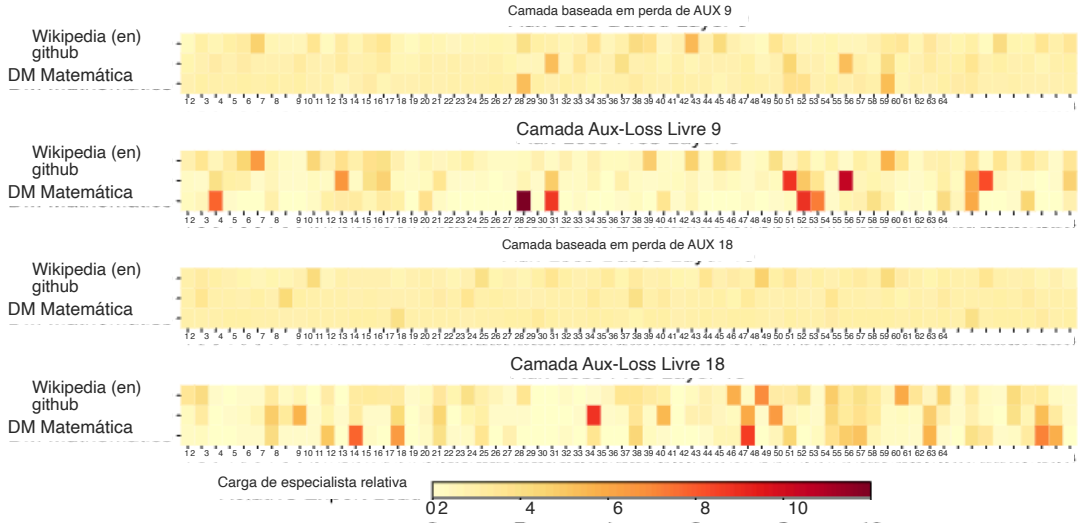


Figura 9 | Modelos especializados em modelos baseados em perda de perda auxiliar e com perda de perda auxiliar em três domínios no conjunto de testes de pilha. O modelo Auxiliar-Loss Livre mostra maiores padrões de especialização de especialistas do que o de base de perda auxiliar. A carga de especialista relativa indica a proporção entre a carga especialista real e a carga especializada teoricamente equilibrada. Devido a restrições de espaço, apresentamos apenas os resultados de duas camadas como exemplo, com os resultados de todas as camadas fornecidas no Apêndice C.

perda auxiliar). Também observamos resultados semelhantes nos modelos MOE 3B: o modelo usando uma perda auxiliar em termos de sequência atinge uma perda de validação de 2,085, e os modelos usando o método livre de perdas auxiliares ou uma perda auxiliar em lote atinge a mesma perda de validação de 2.080.

Além disso, embora os métodos de balanceamento de carga em lote mostrem vantagens de desempenho consistentes, eles também enfrentam dois possíveis desafios na eficiência: (1) o desequilíbrio de carga dentro de certas sequências ou pequenos lotes e (2) desequilíbrio de carga induzida por mudança de domínio durante o inferno - ence. O primeiro desafio é naturalmente abordado por nossa estrutura de treinamento que usa paralelismo e paralelismo de dados especialistas em larga escala, o que garante um tamanho grande de cada micro-lote. Para o segundo desafio, também projetamos e implementamos uma estrutura de inferência eficiente com implantação de especialistas redundantes, conforme descrito na Seção 3.4, para superá -lo.

5. Pós-treinamento

5.1. Tuneamento fino supervisionado

Curatamos nossos conjuntos de dados de ajuste de instrução para incluir 1,5 milhão de instâncias que abrangem vários domínios, com cada domínio empregando métodos distintos de criação de dados adaptados aos seus requisitos específicos.

Dados de raciocínio. Para conjuntos de dados relacionados ao raciocínio, incluindo aqueles focados em matemática, problemas de concorrência de código e quebra-cabeças lógicos, geramos os dados, aproveitando um modelo interno de Deepseek-R1. Especificamente, embora os dados gerados por R1 demonstrem forte precisão, eles sofrem de questões como imenso, baixa formatação e comprimento excessivo. Nosso objetivo é equilibrar a alta precisão dos dados de raciocínio gerados por R1 e a clareza e concisão dos dados de raciocínio regularmente formatados.

Para estabelecer nossa metodologia, começamos desenvolvendo um modelo de especialista adaptado a um domínio específico, como código, matemática ou raciocínio geral, usando um pipeline de treinamento combinado de tênue supervisionado (SFT) e aprendizado de reforço (RL). Este modelo especialista serve como gerador de dados para o modelo final. O processo de treinamento envolve a geração de dois tipos distintos de amostras de SFT para cada instância: os primeiros casais o problema com sua resposta original no formato de<problem, original response> , enquanto o segundo incorpora um prompt de sistema ao lado do problema e da resposta R1 no formato de<system prompt, problem, R1 response> .

O prompt do sistema é meticulosamente projetado para incluir instruções que orientam o modelo para produzir respostas enriquecidas com mecanismos para reflexão e verificação. Durante a fase RL, o modelo aproveita a amostragem de alta temperatura para gerar respostas que integram padrões dos dados gerados por R1 e originais, mesmo na ausência de avisos explícitos do sistema. Após centenas de etapas da RL, o modelo intermediário de RL aprende a incorporar padrões R1, aumentando assim o desempenho geral estrategicamente.

Ao concluir a fase de treinamento da RL, implementamos a amostragem de rejeição para curar dados de SFT de alta qualidade para o modelo final, onde os modelos especializados são usados como fontes de geração de dados. Esse método garante que os dados finais de treinamento mantenham os pontos fortes do Deepseek-R1, produzindo respostas concisas e eficazes.

Dados não raciocinadores. Para dados não-raciocinadores, como escrita criativa, dramatização e resposta de perguntas, utilizamos Deepseek-V2.5 para gerar respostas e recrutar anotadores humanos para verificar a precisão e a correção dos dados.

Configurações SFT. Nós ajustamos o Deepseek-V3-BASE para duas épocas usando o conjunto de dados SFT, usando o agendamento da taxa de aprendizado de decaimento de cosseno que começa em 5 × 10 a 6 e diminui gradualmente para 1 × 10-6. Durante o treinamento, cada sequência única é embalada de várias amostras. No entanto, adotamos uma estratégia de mascaramento de amostra para garantir que esses exemplos permaneçam isolados e mutuamente invisíveis.

5.2. Aprendizagem de reforço

5.2.1. Modelo de recompensa

Empregamos um modelo de recompensa baseado em regras (RM) e um RM baseado em modelo em nosso processo de RL.

RM baseado em regras. Para perguntas que podem ser validadas usando regras específicas, adotamos um sistema de recompensa baseado em regras para determinar o feedback. Por exemplo, certos problemas matemáticos têm resultados determinísticos e exigimos que o modelo forneça a resposta final dentro de um formato designado (por exemplo, em uma caixa), permitindo que aplique regras para verificar a correção. Da mesma forma, para problemas de leetcode, podemos utilizar um compilador para gerar feedback com base em casos de teste. Ao alavancar a validação baseada em regras sempre que possível, garantimos um nível mais alto de confiabilidade, pois essa abordagem é resistente à manipulação ou exploração.

RM baseado em modelo. Para perguntas com respostas de verdade de forma livre, contamos com o modelo de recompensa para determinar se a resposta corresponde à verdadeira verdadeira-verdade. Por outro lado, para perguntas sem uma verdade definitiva, como aquelas que envolvem escrita criativa, o modelo de recompensa tem a tarefa de fornecer feedback com base na pergunta e na resposta correspondente

como entradas. O modelo de recompensa é treinado a partir dos pontos de verificação SFT Deepseek-V3. Para aumentar sua confiabilidade, construímos dados de preferência que não apenas fornecem a recompensa final, mas também incluem a cadeia de pensamento que leva à recompensa. Essa abordagem ajuda a mitigar o risco de hackers de recompensa em tarefas específicas.

5.2.2. Otimização relativa de política do grupo

Semelhante a Deepseek-V2 (Deepseek-AI, 2024C), adotamos otimização relativa de política relativa do grupo (GRPO) (Shao et al., 2024), que abandona o modelo crítico que normalmente é com o mesmo tamanho que o modelo de política, e estima a linha de base das pontuações do grupo. Especificamente, para cada pergunta q, GRPO amostra um grupo de saídas $\{o1, o2, \cdots, oG\}$ do antigo modelo de política $\pi\theta old$ e depois otimiza o modelo de política $\pi\theta$ maximizando o seguinte objetivo:

$$JGRPO\,(\theta) = \mathrm{e}\,[q \sim P\,(Q), \{oi\}\,Gi = 1 \sim \pi\theta old\,(O \mid q)]]$$

$$\frac{1}{G}\sum_{i=1}^{G} \min\left[\frac{\pi\theta\,(oi \mid q)}{\pi\theta old\,(oi \mid q)} Ai,\ \mathrm{clip}\left(\frac{\pi\theta\,(oi \mid q)}{\pi\theta old\,(oi \mid q)}, 1-\varepsilon, 1+\varepsilon\right) Ai\right] - \beta \mathrm{d}KL\pi\theta \,\|. \tag{26}$$

$$\mathrm{D}KL\pi\theta \,\|\, \pi ref = \frac{\pi ref\,(oi \mid q)}{\pi\theta\,(oi \mid q)} - \log\frac{\pi ref\,(oi \mid q)}{\pi\theta\,(oi \mid q)} - 1, \tag{27}$$

onde ε e β são hiper-parâmetros; πref é o modelo de referência; e Ai é a vantagem, derivada das recompensas $\{r1, r2, \ldots, rG\}$ correspondente às saídas dentro de cada grupo:

$$Ai = \frac{ri - \text{média}\,(\{r1, r2, \cdots, rG\})}{\text{std}\,(\{r1, r2, \cdots, rG\})}. \tag{28}$$

Incorporamos instruções de diversos domínios, como codificação, matemática, redação, dramatização e resposta a perguntas, durante o processo de RL. Essa abordagem não apenas alinha o modelo mais de perto com as preferências humanas, mas também aprimora o desempenho dos benchmarks, especialmente em cenários, onde os dados disponíveis da SFT são limitados.

5.3. Avaliações

5.3.1. Configurações de avaliação

Benchmarks de avaliação. Além da referência que usamos para testes de modelos básicos, avaliamos ainda modelos instruídos em Ifeval (Zhou et al., 2023), quadros (Krishna et al., 2024), Longbench V2 (Bai et al., 2024), GPQA (Rein et al., 2023), Simpleqa (Openai, 2024C), C-Simpleqa (He et al., 2024), SWE-banch Verificado (Openai, 2024d), Aider 1, LiveCodebench (Jain et al., 2024) (Perguntas de agosto de 2024 a novembro de 2024), Code Forces 2, Olimpíada de Matemática da Escola Nacional Chinesa (CNMO 2024) 3 e Exame de Matemática Americana de Matemática Invitacional 2024 (Aime 2024) (MAA, 2024).

Comparado linhas de base. Realizamos avaliações abrangentes de nosso modelo de bate-papo contra setores de base fortes, incluindo Deepseek-V2-0506, Deepseek-V2.5-0905, QWEN2.5 72B Instruct, LLAMA-3.1 405B Instruct, Claude-Sonnet-3.5-1022, e GPT-4O-0513. Para a série de modelos Deepseek-V2, selecionamos as variantes mais representativas para comparação. Para modelos de código fechado, as avaliações são realizadas por meio de suas respectivas APIs.

1https: //aider.chat 2https: //codeforces.com 3https: //www.cms.org.cn/home/comp/comp/cid/12.html

Configurações detalhadas de avaliação. Para benchmarks padrão, incluindo MMLU, Drop, GPQA e SimpleQA, adotamos os prompts de avaliação da estrutura simples-EVals4. Utilizamos o formato de prompt zero-eval (LIN, 2024) para MMLU-Redux em uma configuração de tiro zero. Para outros conjuntos de dados, seguimos seus protocolos de avaliação originais com prompts padrão, conforme fornecido pelos criadores do conjunto de dados. Para benchmarks de código e matemática, o conjunto de dados Humaneval-Mul inclui 8 linguagens de programação convencionais (Python, Java, CPP, C#, JavaScript, Typescript, Php e Bash) no total. Utilizamos métodos COT e não-COT para avaliar o desempenho do modelo no LivecodeBench, onde os dados são coletados de agosto de 2024 a novembro de 2024. O conjunto de dados Codeforces é medido usando a porcentagem de concorrentes. O SWE-banch verificado é avaliado usando a estrutura sem agente (Xia et al., 2024). Utilizamos o formato "Dif" para avaliar os benchmarks relacionados à AIDER. Para avaliações matemáticas, o AIME e o CNMO 2024 são avaliados com uma temperatura de 0,7, e os resultados são calculados em média em 16 corridas, enquanto o Math-500 emprega decodificação gananciosa. Permitimos que todos os modelos emitam um máximo de 8192 tokens para cada referência.

	Benchmark (métrica)	Deepseek v2.5-0905	Deepseek 72B-Inst.	Qwen2.5 405B-Inst.	llama-3.1 SONNET-1022	Claude-3.5- 0513	GPT-4O V3	Deepseek V2-0506
	Arquitetura	Moe	Moe	Denso	Denso	-	-	Moe
	# Params ativados	21b	21b	72b	405b	-	-	37b
	# Total Params	236b	236b	72b	405b	-	-	671b
Inglês	MMLU (EM)	78.2	80.6	85.3	88.6	88.3	87.2	88.5
	MMLU-Redux (EM)	77.9	80.3	85.6	86.2	88.9	88.0	89.1
	MMLU-Pro (EM)	58.5	66.2	71.6	73.3	78.0	72.6	75.9
	Drop (3 tiros F1)	83.0	87.8	76.7	88.7	88.3	83.7	91.6
	If-EVAL (rápido rigoroso)	57.7	80.6	84.1	86.0	86.5	84.3	86.1
	GPQA-DIAMOND (PASS@1) 35.3		41.3	49.0	51.1	65.0	49.9	59.1
	SimpleQa (correto)	9.0	10.2	9.1	17.1	28.4	38.2	24.9
	Quadros (acc.)	66.9	65.4	69.8	70.0	72.5	80.5	73.3
	Longbench v2 (acc.)	31.6	35.4	39.4	36.1	41.0	48.1	48.7
Código	Humaneval-Mul (Pass@1) 69.3		77.4	77.3	77.2	81.7	80.5	82.6
	LivecodeBench (Pass@1-Cot) 18.8		29.2	31.1	28.4	36.3	33.4	40.5
	LivecodeBench (passe@1) 20.3		28.4	28.7	30.1	32.8	34.2	37.6
	Codeforces (percentil)	17.5	35.6	24.8	25.3	20.3	23.6	51.6
	SWE Verificado (resolvido) -		22.6	23.8	24.5	50.8	38.8	42.0
	Help-edit (Acc.)	60.3	71.6	65.4	63.9	84.2	72.9	79.7
	Help-Colflot (Acc.) -		18.2	7.6	5.8	45.3	16.0	49.6
A matemática gosta de 2024 (passe@1)		4.6	16.7	23.3	23.3	16.0	9.3	39.2
	MATH-500 (EM)	56.3	74.7	80.0	73.8	78.3	74.6	90.2
	CNMO 2024 (passe@1)	2.8	10.8	15.9	6.8	13.1	10.8	43.2
Chinese CLUEWSC (EM)		89.9	90.4	91.4	84.7	85.4	87.9	90.9
	C-Eval (EM)	78.6	79.5	86.1	61.5	76.7	76.0	86.5
	C-simpleqa (correto)	48.5	54.1	48.4	50.4	51.3	59.3	64.8

Tabela 6 | Comparação entre Deepseek-V3 e outros modelos de bate-papo representativos. Todos os modelos são avaliados em uma configuração que limita o comprimento da saída a 8k. Os benchmarks contendo menos de 1000 amostras são testados várias vezes usando configurações variadas de temperatura para derivar resultados finais robustos. O Deepseek-V3 é o modelo de código aberto com melhor desempenho e também exibe desempenho competitivo contra modelos de código fechado da Frontier.

4https://github.com/openai/simple-evals

5.3.2. Avaliação padrão

A Tabela 6 apresenta os resultados da avaliação, mostrando que o Deepseek-V3 se destaca como o modelo de código aberto com melhor desempenho. Além disso, é competitivo contra modelos de código fechado da Frontier, como GPT-4O e Claude-3.5,.

Benchmarks ingleses. A MMLU é um benchmark amplamente reconhecido, projetado para avaliar o desempenho de grandes modelos de linguagem, em diversos domínios e tarefas de conhecimento. O Deepseek-V3 demonstra desempenho competitivo, em pé de igualdade com modelos de primeira linha, como LLAMA- 3.1-405B, GPT-4O e Claude-Sonet 3.5, enquanto supera significativamente o QWEN2.5 72B. Além disso, o DeepSeek-V3 se destaca no MMLU-Pro, um benchmark de conhecimento educacional mais desafiador, onde ele segue de perto o Claude-Sonnet 3.5. No MMLU-Redux, uma versão refinada do MMLU com rótulos corrigidos, o Deepseek-V3 supera seus pares. Além disso, no GPQA-Diamond, um teste de avaliação no nível de doutorado, o Deepseek-V3 alcança resultados notáveis, classificando logo atrás de Claude 3,5 sonetos e superando todos os outros concorrentes por uma margem substancial.

Em benchmarks de compreensão de longo contexto, como Drop, Longbench V2 e Frames, o Deepseek-V3 continua demonstrando sua posição como um modelo de primeira linha. Ele atinge uma impressionante pontuação de 91,6 F1 na configuração de 3 tiros na queda, superando todos os outros modelos nesta categoria. Nos quadros, uma referência que exige que a resposta a perguntas de mais de 100 mil contextos de token, o DeepSeek-V3 segue de perto o GPT-4O enquanto supera todos os outros modelos por uma margem significativa. Isso demonstra a forte capacidade de DeepSeek-V3 no lidar com tarefas extremamente de longo prazo. A capacidade de longo contexto do Deepseek-V3 é ainda mais validada por seu melhor desempenho na categoria no Longbench V2, um conjunto de dados que foi lançado apenas algumas semanas antes do lançamento do Deepseek V3. Sobre o benchmark de conhecimento factual, o SimpleQA, o Deepseek-V3 fica atrás do GPT-4O e do Claude, principalmente devido ao seu foco e alocação de recursos de design. Deepseek-V3 atribui mais tokens de treinamento para aprender conhecimento chinês, levando a um desempenho excepcional no C-Simpleqa. Na referência que segue a instrução, o DeepSeek-V3 supera significativamente seu antecessor, Deepseek-V2-Series, destacando sua capacidade aprimorada de entender e aderir às restrições de formato definido pelo usuário.

Código e benchmarks matemáticos. A codificação é uma tarefa desafiadora e prática para o LLMS, abrangendo tarefas focadas em engenharia, como verificadas com SWE-banche e aider, além de tarefas algorítmicas como Humaneval e LivecodeBench. Nas tarefas de engenharia, o Deepseek-V3 trata de Claude-Sonnet-3.5-1022, mas supera significativamente os modelos de código aberto. Espera-se que o Deepseek-V3 de código aberto promova os avanços nas tarefas de engenharia relacionadas à codificação. Ao fornecer acesso a seus recursos robustos, a Deepseek-V3 pode impulsionar a inovação e a melhoria em áreas como engenharia de software e desenvolvimento de algoritmos, capacitando desenvolvedores e pesquisadores a ultrapassar os limites do que os modelos de código aberto podem alcançar nas tarefas de codificação. Nas tarefas algorítmicas, o Deepseek-V3 demonstra desempenho superior, superando todas as linhas de base em benchmarks como Humaneval-Mul e LivecodeBench. Esse sucesso pode ser atribuído à sua técnica avançada de destilação de conhecimento, que efetivamente aprimora seus recursos de geração de código e solução de problemas em tarefas focadas no algoritmo.

Nos benchmarks matemáticos, o Deepseek-V3 demonstra desempenho excepcional, superando significativamente as linhas de base e estabelecendo um novo estado da arte para modelos não O1. Especificamente, em Aime, Math-500 e CNMO 2024, Deepseek-V3 supera o segundo melhor modelo, QWEN2.5 72B, em aproximadamente 10% em pontuações absolutas, que é uma margem substancial para referências desafiadoras. Essa capacidade notável destaca a eficácia da técnica de destilação da Deepseek-R1, que se comprovou altamente benéfica para modelos não O1.

Modelo	Arena-dura	ALPACAEVAL 2.0
Deepseek-V2.5-0905	76.2	50.5
QWEN2.5-72B-INSTRUTA	81.2	49.1
Chamada-3.1 405b	69.3	40.5
GPT-4O-0513	80.4	51.1
Claude-Sonnet-3.5-1022	85.2	52.0
Deepseek-V3	85.5	70.0

Tabela 7 | Avaliações de conversação aberta em inglês. Para o Alpacaeval 2.0, usamos a taxa de vitória controlada por comprimento como métrica.

Benchmarks chineses. Qwen e Deepseek são duas séries de modelos representativas com suporte robusto para chinês e inglês. Na referência factual chinesa simpleqa, o DeepSeek-V3 supera o QWEN2.5-72B por 16,4 pontos, apesar de Qwen2.5 ser treinado em um corpus maior que comprometeu 18T tokens, que são 20% mais que os tokens 14,8T que Deepseek-V3 é pré - Torada em.

No C-EVAL, uma referência representativa para avaliação do conhecimento educacional chinês e ClueWSC (Chinese Winograd Schema Challenge), Deepseek-V3 e Qwen2.5-72b exibem níveis de desempenho semelhantes, indicando que ambos os modelos são bem otimizados para desafiar a língua chinesa Raciocínio e tarefas educacionais.

5.3.3. Avaliação aberta

Além dos benchmarks padrão, também avaliamos nossos modelos em geração aberta

Tarefas usando o LLMS como juízes, com os resultados mostrados na Tabela 7. Especificamente, aderimos às configurações originais do Alpacaeeval 2.0 (Dubois et al., 2024) e arena (Li et al., 2024a), que alavancam o GPT- 4-Turbo-1106 como juízes para comparações pareadas. Na arena, o DeepSeek-V3 atinge uma taxa impressionante de vitória de mais de 86% em relação ao GPT-4-0314 da linha de base, realizando a par com modelos de primeira linha como Claude-Sonnet-3.5-1022. Isso ressalta os recursos robustos do Deepseek-V3, especialmente ao lidar com instruções complexas, incluindo tarefas de codificação e depuração. Além disso, o DeepSeek-V3 alcança um marco inovador como o primeiro modelo de código aberto a superar 85% no benchmark arena. Essa conquista preenche significativamente a diferença de desempenho entre os modelos de código aberto e de código fechado, definindo um novo padrão para o que os modelos de código aberto podem realizar em domínios desafiadores.

Da mesma forma, o Deepseek-V3 mostra o desempenho excepcional no Alpacaeeval 2.0, superando os modelos de código fechado e de código aberto. Isso demonstra sua excelente proficiência em redação de tarefas e lidar com cenários diretos de resposta a perguntas. Notavelmente, supera o DeepSeek-V2.5-0905 por uma margem significativa de 20%, destacando melhorias substanciais no combate às tarefas simples e mostrando a eficácia de seus avanços.

5.3.4. Deepseek-V3 como um modelo de recompensa generativa

Comparamos a capacidade de julgamento do DeepSeek-V3 com os modelos de última geração, a saber, GPT-4O e Claude-3.5. A Tabela 8 apresenta o desempenho desses modelos em Recompense (Lambert et al., 2024). O Deepseek-V3 atinge o desempenho em pé de igualdade com as melhores versões do GPT-4O-0806 e Claude-3.5-SONNET-1022, enquanto supera outras versões. Além disso, a capacidade de julgamento do Deepseek-V3 também pode ser aprimorada pela técnica de votação. Portanto, empregamos Deepseek-V3, juntamente com a votação para oferecer um auto-feedback em perguntas abertas, melhorando assim o

Modelo	Bater papo	Chat-Hard	Segurança	Raciocínio	Média
GPT-4O-0513	96.6	70.4	86.7	84.9	84.7
GPT-4O-0806	96.1	76.1	88.1	86.6	86.7
GPT-4O-1120	95.8	71.3	86.2	85.2	84.6
Claude-3.5-SNONT-0620	96.4	74.0	81.6	84.7	84.2
Claude-3.5-SNONT-1022	96.4	79.7	91.1	87.6	88.7
Deepseek-V3	96.9	79.8	87.0	84.3	87.0
Deepseek-V3 (maj@6)	96.9	82.6	89.5	89.2	89.6

Tabela 8 | Performances do GPT-4O, Claude-3.5 e Deepseek-V3 no RecompenseBench.

Modelo	LivecodeBench-Cot		Math-500	
	Passe@1	Comprimento	Passe@1	Comprimento
Deepseek-V2.5 linha de base	31.1	718	74.6	769
Deepseek-V2.5 +R1 Destill	37.4	783	83.2	1510

Tabela 9 | A contribuição da destilação de Deepseek-R1. As configurações de avaliação do CodeBench Live e Math-500 são as mesmas da Tabela 6.

Eficácia e robustez do processo de alinhamento.

5.4. Discussão

5.4.1. Destilação de Deepseek-R1

Ablidamos a contribuição da destilação de Deepseek-R1 com base no Deepseek-V2.5. A linha de base é treinada em dados de berço curto, enquanto seu concorrente usa dados gerados pelos pontos de verificação de especialistas descritos acima.

A Tabela 9 demonstra a eficácia dos dados de destilação, mostrando melhorias significativas nos benchmarks LivecodeBench e Math-500. Nossos experimentos revelam um trade-off de interesse: a destilação leva a um melhor desempenho, mas também aumenta substancialmente o comprimento médio da resposta. Para manter um equilíbrio entre a precisão do modelo e a eficiência computacional, selecionamos cuidadosamente as configurações ideais para Deepseek-V3 na destilação.

Nossa pesquisa sugere que a destilação do conhecimento dos modelos de raciocínio apresenta uma direção promissora para a otimização pós-treinamento. Embora nosso trabalho atual se concentre na destilação de dados de matemática e domínios de codificação, essa abordagem mostra potencial para aplicações mais amplas em vários domínios de tarefas. A eficácia demonstrada nessas áreas específicas indica que a destilação de bico longo pode ser valiosa para melhorar o desempenho do modelo em outras tarefas cognitivas que requerem raciocínio complexo. A exploração adicional dessa abordagem em diferentes domínios continua sendo uma direção importante para pesquisas futuras.

5.4.2. Auto-recompensa

As recompensas desempenham um papel fundamental na RL, direcionando o processo de otimização. Em domínios onde a verificação através de ferramentas externas é direta, como alguns cenários de codificação ou matemática, o RL demonstra eficácia excepcional. No entanto, em cenários mais gerais, construindo um feedback

O mecanismo através da codificação dura é impraticável. Durante o desenvolvimento do Deepseek-V3, para esses contextos mais amplos, empregamos a abordagem constitucional de IA (Bai et al., 2022), alavancando os resultados da avaliação de votação do próprio Deepseek-V3 como fonte de feedback. Esse método produziu efeitos notáveis de alinhamento, aumentando significativamente o desempenho do Deepseek-V3 em avaliações subjetivas. Ao integrar informações constitucionais adicionais, o Deepseek-V3 pode otimizar em direção à direção constitucional. Acreditamos que esse paradigma, que combina informações suplementares com o LLMS como fonte de feedback, é de suma importância. O LLM serve como um processador versátil capaz de transformar informações não estruturadas de diversos cenários em recompensas, facilitando o auto-aperfeiçoamento do LLMS. Além de auto-recompensa, também nos dedicamos a descobrir outros métodos de gratificação geral e escalável para promover consistentemente os recursos do modelo em cenários gerais.

5.4.3. Avaliação de previsão de vários toques

Em vez de prever apenas o próximo single token, o Deepseek-V3 prevê os próximos 2 tokens através da técnica MTP. Combinado com a estrutura de decodificação especulativa (Leviathan et al., 2023; Xia et al., 2023), ele pode acelerar significativamente a velocidade de decodificação do modelo. Surge uma questão natural sobre a taxa de aceitação do token previsto adicionalmente. Com base em nossa avaliação, a taxa de aceitação do segundo token varia entre 85% e 90% em vários tópicos de geração, demonstrando confiabilidade consistente. Essa alta taxa de aceitação permite que o DeepSeek-V3 atinja uma velocidade de decodificação significativamente aprimorada, fornecendo 1,8 vezes TPS (tokens por segundo).

6. Conclusão, limitações e direções futuras

Neste artigo, introduzimos Deepseek-V3, um grande modelo de linguagem MOE com 671B totais totais e parâmetros ativados 37b, treinados em tokens 14,8T. Além das arquiteturas MLA e Deepseekmoe, também pioneira uma estratégia livre de perdas auxiliares para equilibrar a carga e define um objetivo de treinamento de previsão de vários toques para um desempenho mais forte. O treinamento do Deepseek-V3 é econômico devido ao apoio do treinamento FP8 e as meticulosas operações de engenharia. O pós-treinamento também obtém sucesso em destilar a capacidade de raciocínio da série de modelos Deepseek-R1. Avaliações abrangentes demonstram que o DeepSeek-V3 emergiu como o modelo de código aberto mais forte atualmente disponível e atinge o desempenho em parábola para os principais modelos de código fechado como GPT-4O e Claude-3.5. Apesar de seu forte desempenho, ele também mantém os custos econômicos de treinamento. Requer apenas 2,788m de horas de GPU H800 para seu treinamento completo, incluindo pré-treinamento, extensão de comprimento do contexto e pós-treinamento.

Embora reconheçamos seu forte desempenho e custo-efetividade, também reconhecemos que o Deepseek-V3 tem algumas limitações, especialmente na implantação. Em primeiro lugar, para garantir a inferência eficiente, a unidade de implantação recomendada para Deepseek-V3 é relativamente grande, o que pode representar um fardo para equipes de tamanho pequeno. Em segundo lugar, embora nossa estratégia de implantação para a DeepSeek-V3 tenha atingido uma velocidade de geração de ponta a ponta de mais de duas vezes a do Deepseek-V2, ainda há potencial para melhorar. Felizmente, espera -se que essas limitações sejam naturalmente abordadas com o desenvolvimento de hardware mais avançado.

O Deepseek segue constantemente a rota de modelos de código aberto com o longtermismo, com o objetivo de abordar constantemente o objetivo final da AGI (inteligência geral artificial). No futuro, planejamos investir estrategicamente em pesquisas nas seguintes direções.

- Estudaremos e refinamos consistentemente nossas arquiteturas modelo, com o objetivo de melhorar ainda mais

Tanto o treinamento quanto a eficiência de inferência, esforçando -se para abordar suporte eficiente para o comprimento do contexto infinito. Além disso, tentaremos romper as limitações arquitetônicas do transformador, aumentando assim os limites de seus recursos de modelagem.

• Itrearemos continuamente a quantidade e a qualidade de nossos dados de treinamento e exploraremos a incorporação de fontes adicionais de sinal de treinamento, com o objetivo de impulsionar os dados em uma gama mais abrangente de dimensões.

• Exploraremos e iteraremos consistentemente as capacidades de pensamento profundo de nossos modelos, com o objetivo de aprimorar suas habilidades de inteligência e solução de problemas, expandindo sua duração e profundidade de raciocínio.

• Exploraremos os métodos de avaliação de modelos mais abrangentes e multidimensionais para evitar a tendência de otimizar um conjunto fixo de benchmarks durante a pesquisa, o que pode criar uma impressão enganosa dos recursos do modelo e afetar nossa avaliação fundamental.

Referências

Ai@meta. LLAMA 3 MODELA CARD, 2024A. Url https://github.com/meta-llama/llama3/bl
ob/main/model_card.md.

Ai@meta. LLAMA 3.1 Modelo Card, 2024b. Url https://github.com/meta-llama/llama-m
odels/blob/main/modelos/llama3_1/model_card.md.

Antrópico. Claude 3,5 Sonnet, 2024. URL https://www.anthrópica.com/news/claude-3
–5
Sonetos.

J. Austin, A. Odena, M. Nye, M. Bosma, H.M. Programa de síntese com modelos. ARX: 2108.2108.07732,

Y. Bai, S. Davath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon, et al. AI constitucional: inofensidade do feedback da IA. Arxiv pré -impressão Arxiv: 2212.08073,

Y. Bai, S. Tu, J. Zhang, H. Peng, X. Wang, X. LV, S. Cao, J. Xu, L. Hou, Y. Dong, J. Tang e J. Li. Longbench V2: Para uma compreensão e raciocínio mais profundos em multitarefa realistas de longo contexto. Arxiv pré -impressão Arxiv: 2412.15204, 2024.

M. Bauer, S. Treichler e A. Aiken. Singe: Alavancando a especialização do Warp para alto desempenho nas GPUs. Em Anais do 19º Simpósio ACM Sigplan, sobre princípios e prática de programação paralela, PPOPP '14, página 119–130, Nova York, NY, EUA, 2014. Associação de Máquinas de Computação. ISBN 9781450326568. DOI: 10.1145/2555243.2555258. URL https://doi.org/10.1145/2555243.2555258.

Y. Bisk, R. Zellers, RL Bras, J. Gao e Y. Choi. PIQA: Raciocínio sobre o senso comum em linguagem natural. Na Trigésima Quarta Conferência AAAI sobre Inteligência Artificial, AAAI 2020, The Trinta e Segundo Aplicações Inovadoras da Conferência de Inteligência Artificial, IAAI 2020, o décimo Simpósio AAAI sobre Avanços Educacionais em Inteligência Artificial, EAAI 2020, Nova York, NY, EUA, fevereiro 7-12, 2020, páginas 7432-7439. AAAI Press, 2020. doi: 10.1609/aaai.v34i05.6239. URL https://doi.org/10.1609/aaai.v34i05.6239.

M. Chen, J. Tworek, H. Yuan, Q. Yuan, HP de Oliveira Pinto, J. Kaplan, H. Edwards, Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G.

B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, FP como, D. Cummings, M. Plappert, F. Chantzis , E. Barnes, A. Herbert-Voss, Wh Guss, A. Nichol, A. Pago, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A Carr, J. Like, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei , S. McCandlish, I. Sutskever, e W. Zaremba. Avaliando grandes modelos de linguagem treinados no código. CORR, ABS/2107.03374, 2021. URL https://arxiv.org/abs/2107.03374.

P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick e O. Tafjord. Acha que você resolveu a resposta de perguntas? Experimente o ARC, o desafio de raciocínio da AI2. CORR, ABS/1803.05457, 2018. URL http://arxiv.org/abs/1803.05457.

K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, et al. Verificadores de treinamento para resolver problemas de palavras matemáticas. Arxiv pré -impressão Arxiv: 2110.14168, 2021.

Y. Cui, T. Liu, W. Che, L. Xiao, Z. Chen, W. Ma, S. Wang e G. Hu. Um conjunto de dados de extração de span para compreensão de leitura de máquina chinesa. Em K. Inui, J. Jiang, V. Ng e X. Wan, editores, Anais da Conferência de 2019 sobre Métodos Empíricos no Processamento de Linguagem Natural e 9ª Conferência Internacional de Linguagem Natural (EMNLP-IJCNLP), Páginas 5883 –5889, Hong Kong, China, novembro de 2019. Associação de Linguística Computacional. doi: 10.18653/v1/d19-1600. URL https://aclanthology.org/d19-1 600.

D. Dai, C. Deng, C. Zhao, Rx Xu, H. Gao, D. Chen, J. Li, W. Zeng, X. Yu, Y. Wu, Z. Xie, Yk Li, P. Huang, F. Luo, C. Ruan, Z. Sui e W. Liang. Deepseekmoe: Rumo à Ultimate Expert Specialization em modelos de linguagem Misture-of-Experts. CORR, ABS/2401.06066, 2024. URL https://doi.org/10.48550/arxiv.2401.06066.

Deepseek-AI. Deepseek-Coder-V2: quebrando a barreira de modelos de código fechado em Intelligência de Código. CORR, ABS/2406.11931, 2024A. URL https://doi.org/10.48550/arxiv.2406.11 931.

Deepseek-AI. Deepseek LLM: Escalando modelos de idiomas de código aberto com longtermismo. Corrigir ABS/2401.02954, 2024B. URL https://doi.org/10.48550/arxiv.2401.02954.

Deepseek-AI. Deepseek-V2: Um modelo de linguagem forte, econômico e eficiente da mistura de especialistas. CORR, ABS/2405.04434, 2024C. URL https://doi.org/10.48550/arxiv.2405. 04434.

T. Dettmers, M. Lewis, Y. Belkada e L. Zettlemoyer. GPT3. Int8 (): multiplicação de matriz de 8 bits para transformadores em escala. Avanços nos sistemas de processamento de informações neurais, 35: 30318–30332, 2022.

H. Ding, Z. Wang, G. Paolini, V. Kumar, A. Deoras, D. Roth e S. Soatto. Menos truncações melhorar a modelagem de idiomas. Arxiv pré -impressão Arxiv: 2404.10830, 2024.

D. Dua, Y. Wang, P. Dasigi, G. Stanovsky, S. Singh e M. Gardner. Drop: uma referência de compra de leitura que exige raciocínio discreto sobre os parágrafos. Em J. Burstein, C. Doran, e T. Solorio, editores, Proceedings of the 2019 Conference of the North American Capítulo da Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, EUA, 2 de junho -7, 2019, Volume 1 (artigos longos e curtos), páginas 2368-3378. URL https://doi.org/10.18653/v1/n19-1246.

Y. Dubois, B. Galambosi, P. Liang e TB Hashimoto. ALPACAEVAL CONCVRULADO DO LIMPE: Simples Maneira para os avaliadores automáticos de Debias. Arxiv pré -impressão Arxiv: 2404.04475, 2024.

W. Fedus, B. Zoph e N. Parta. Transformadores de comutação: dimensionando modelos de parâmetros de trilhões com esparsidade simples e eficiente. CORR, ABS/2101.03961, 2021. URL https://arxiv.org/ ABS/2101.03961.

M. Fishman, B. Chmiel, R. Banner e D. Soudry. Escala de treinamento FP8 para trilhões de TOKEN LLMS. Arxiv pré -impressão Arxiv: 2409.12517, 2024.

E. Frantar, S. Ashkboos, T. Hoefler e D. Alistarh. GPTQ: quantização precisa pós-treinamento para transformadores pré-treinados generativos. Arxiv pré -impressão Arxiv: 2210.17323, 2022.

L. Gao, S. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite, N. Nabeshima, et al. A pilha: um conjunto de dados de 800 GB de texto diversificado para modelagem de idiomas. Arxiv pré -impressão Arxiv: 2101.00027, 2020.

AP Gema, Joj Leang, G. Hong, A. ACM Sex, R. Saxena, X. X. He, X. Zhao, X. X. Black, Mrg Madani, C. Barale, R. McHardy, J. Harris, J. Kaddour, E. Van Cruis e P. Minervini. Estamos dê com MMLU? CORR, ABS/2406.04127, 2024. URL https: //do.or g/18550/arxiv.2406.04127.

F. Gloeckle, por Idrissi, B. Rozière, D. Lopez-Paz e G. Synnaeve. Modelos de idiomas grandes melhores e mais rápidos por meio de previsão de vários toques. Em quarenta e primeiro conferência internacional sobre aprendizado de máquina, ICML 2024, Viena, Áustria, 21 a 27 de julho de 2024. OpenReview.net, 2024. URL https://openreview.net/forum?id=PewAwACEJIU2.

Google. Nosso modelo de próxima geração: Gemini 1.5, 2024. URL https://blog.google/techno LOGY/AI/Google-Gemini-NEXT-GENERAÇÃO MODEL-FEBROY-2024.

RL Graham, D. Bureddy, P. Lui, H. Rosenstock, G. Shainer, G. Bloch, D. Goldenerg, M. Dubman, S. Kotchubievsky, V. Koushnir, et al. Protocolo de agregação hierárquica escalável (Sharp): uma arquitetura de hardware para redução de dados eficientes. Em 2016, o primeiro workshop internacional sobre otimizações de comunicação no HPC (COMHPC), páginas 1-10. IEEE, 2016.

A. Gu, B. Rozière, H. Leather, A. Solar-Lezama, G. Synnaeve e Si Wang. CruxEval: a Referência para raciocínio, compreensão e execução do código, 2024.

D. Guo, Q. Zhu, D. Yang, Z. Xie, K. Dong, W. Zhang, G. Chen, X. Bi, Y. Wu, Yk Li, F. Luo, Y. Xiong e W. Liang. Deepseek -Coder: Quando o modelo de linguagem grande atende à programação - o aumento da inteligência do código. CORR, ABS/2401.14196, 2024. URL https://doi.org/10.485 50/arxiv.2401.14196.

A. Harlap, D. Narayanan, A. Phanishayee, V. Seshadri, N. Devanur, G. Ganger e P. Gibbons. PipedReam: Treinamento DNN paralelo de pipeline rápido e eficiente, 2018. URL https: //arxiv.or g/abs/1806.03377.

B. He, L. Noci, D. Paliotta, I. Schlag e T. Hofmann. Compreender e minimizar os recursos mais altos no treinamento de transformadores. Na Trigésima Oitava Conferência Anual sobre Sistemas de Processamento de Informações Neurais.

Y. He, S. Li, J. Liu, Y. Tan, W. Wang, H. Huang, X. Bu, H. Guo, C. Hu, B. Zheng, et al. Chinese SimpleQA: uma avaliação de factualidade chinesa para grandes modelos de idiomas. Arxiv pré - impressão Arxiv: 2411.07140, 2024.

D. Hendrycks, C. Burns, Medição

Entendimento maciço de linguagem multitarefa. ARXIV ARXIV ARXIV: 2009.03300, 2020.

D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song e J. Steinhardt. Medição da solução de problemas matemáticos com o conjunto de dados de matemática. Arxiv pré -impressão Arxiv: 2103.03874, 2021.

Y. humano, y C-EVAL: Uma avaliação de mipercisão multi-Experiência de vários níveis Quarenta e primeiro modelos. Presidente do ARXIV ARXIV: 2305.08322, 2023, 2023.

N. Jain, K. Han, A. Gu, W. Li, F. Yan, T. Zhang, S. Wang, A. Solar-Lezama, K. Sen e I. Stoica. LiveCodeBench: Avaliação livre holística e de contaminação de grandes modelos de idiomas para código. CORR, ABS/2403.07974, 2024. URL https://doi.org/10.48550/arxiv.2403.07974.

Aq Jiang, A. Sablayrolles, A. Mensch, C. Bamford, DS Chaplot, D. Dl Casas, F. Bressand,

G. Lengyel, G. Lample, L. Saulnier, et al. Mistral 7b. Arxiv pré -impressão Arxiv: 2310.06825, 2023.

M. Joshi, E. Choi, D. Weld e L. Zettlemoyer. Triviaqa: Um conjunto de dados de critério supervisionado de larga escala para a compreensão de leitura. Em R. Barzilay e M.-Y. Kan, editores, Anais da 55ª Reunião Anual da Associação de Linguística Computacional (Volume 1: Long Papers), páginas 1601–1611, Vancouver, Canadá, julho de 2017. Associação de Linguística Computacional. doi: 10.18653/v1/p17-1147. URL https://aclanthology.org/p17-1147.

D. Kalamkar, D. Mudigre, n. Um estudo do BFLOAT16 para o comércio de aprendizado profundo. Arxiv pré -impressão Arxiv: 1905.12322, 2019.

S. Krishna, K. Krishna, A. Mohananey, S. Schwarcz, A. Stambler, S. Upadhyay e M. Faruqui. Fato, busca e razão: uma avaliação unificada da geração de recuperação upmentada. CORR, ABS/2409.12941, 2024. DOI: 10.48550/arxiv.2409.12941. URL https://doi.org/10.485 50/arxiv.2409.12941.

T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, Ap Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, K. Toutanova, L. Jones, M. Kelcey , M. Chang, Am Dai, J. Uszkoreit, Q. Le e S. Petrov. Perguntas naturais: Uma referência para pesquisas de resposta a perguntas. Trans. Assoc. Computação. Linguistics, 7: 452-466, 2019. doi: 10.1162/tacl \ _a \ _00276. URL https://doi.org/10.1162/tacl_a_00276.

G. Lai, Q. Xie, H. Liu, Y. Yang e Eh Hovy. Raça: conjunto de dados de compreensão de leitura em larga escala dos exames. Em M. Palmer, R. Hwa e S. Riedel, editores, Anais da Conferência de 2017 sobre Métodos Empíricos em Processamento de Linguagem Natural, EMNLP 2017, Copenhague, Dinamarca, 9 a 11 de setembro de 2017, páginas 785-794. Association for Computational Linguistics, 2017. doi: 10.18653/v1/d17-1082. URL https://doi.org/10.18653/v1/d1 7-1082.

N. Lambert, V. Pyatkin, J. Morrison, L. Miranda, por Lin, K. Chandu, N. Dziri, S. Kumar, T. Zick, Y. Choi, et al. Recompensa: Avaliando modelos de recompensa para modelagem de idiomas. Arxiv pré -impressão Arxiv: 2403.13787, 2024.

D. Lepikhin, H. Lee, Y. Xu, D. Chen, O. Firat, Y. Huang, M. Krikun, N. Shazeer e Z. Chen. GSHARD: Modelos gigantes de dimensionamento com computação condicional e sharding automático. Na 9ª Conferência Internacional sobre Representações de Aprendizagem, ICLR 2021. OpenReview.net, 2021. URL https://openreview.net/forum?id=qrwe7xhtmyb.

Y. Leviathan, M. Kalman e Y. Matias. Inferência rápida dos transformadores via especulativo
decodificação. Em Conferência Internacional sobre Aprendizado de Máquinas, ICML 2023, 23-29 de julho de 2023, Honolulu, Havaí, EUA, Volume 202 de Proceedings of Machine Learning Research, páginas 19274-19286. Pmlr, 2023. URL https://proecedings.mlr.press/v202/leviathan23 a.html.

H. Li, Y. Zhang, F. Choto, Y. S. Zhao, Y. CMMLI: Messur- ser uma língua de lanches sob a Chinese, subestima. Presidente do ARXIV ARXIV: 2306.09212, 2023, 2023.

S. Li e T. Hoefler. CHIMERA: Treinando com eficiência redes neurais em larga escala com pipelines bidirecionais. Em Anais da Conferência Internacional para Computação de Alto Desempenho, Networking, Armazenamento e Análise, SC '21, página 1–14. ACM, novembro de 2021. DOI: 10.1145/345881 7.3476145. URL http://dx.doi.org/10.1145/3458817.3476145.

T. Li, W.-L. Chiang, E. Frick, L. Dunlap, T. Wu, B. Zhu, Je Gonzalez e I. Stoica. De dados de crowdsourcing a benchmarks de alta qualidade: oleoduto arena e benchbuilder. Arxiv pré - impressão Arxiv: 2406.11939, 2024a.

W. Li, F. Qi, M. Sun, X. Yi e J. Zhang. CCPM: um conjunto de dados chinês de correspondência de poesia clássica, 2021.

Y. Li, F. Wei, C. Zhang e H. Zhang. Águia: A amostragem especulativa requer a incerteza do recurso de repensar. Em quarenta e primeira conferência internacional sobre aprendizado de máquina, ICML 2024, Viena, Áustria, 21 a 27 de julho de 2024. OpenReview.net, 2024b. Url https://openreview.net/forum? Id = 1ndn7Exyb4.

Por lin. Zeroeeval: uma estrutura unificada para avaliar modelos de idiomas, julho de 2024. URL https://github.com/WildEval/ZeroEval.

I. Loshchilov e F. Hutter. A regularização de decaimento do peso dissociada. Arxiv pré -impressão ARXIV: 1711.05101, 2017.

S. Lundberg. A arte do design imediato: limites imediatos e cura de token, 2023. URL https://towardsdatascience.com/the-art-of-prompt-design-prompt-bound aries-and-token-healing-3b2448b0be38.

Y. Luo, Z. Zhang, R. Wu, H. Liu, Y. Jin, K. Zheng, M. Wang, Z. He, G. Hu, L. Chen, et al. Subir Formato Hifloat8 para aprendizado profundo. Arxiv pré -impressão Arxiv: 2409.16626, 2024.

Maa. Exame de Matemática Americana Invitacional - Aime. No American Invitational Mathematics Examination-Aime 2024, fevereiro de 2024. URL https://maa.org/math -competitions/American-invitational-mathematics-examination-eime.

P. Micikevicius, D. Stosic, N. Burgess, M. Cornea, P. Dubey, R. Grisenthite, S. Ha, A. Heinecke, P. Judd, J. Kamalu, et al. Formatos FP8 para aprendizado de DEP. Arxiv Arxiv: 2209.05433,

Mistral. Mais barato, melhor, mais rápido, mais forte: continuando a empurrar a fronteira da IA e tornando -a Acessível a todos, 2024. URL https://mistral.ai/news/mixtral-8x22b.

S. Narang, G. Diamos, E. Elsen, P. Micikevicius, J. Alben, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh, et al. Treinamento de precisão mista. Em int. Conf. sobre representação de aprendizagem, 2017.

B. Noune, P. Jones, D. Justus, D. Masters e C. Luschi. Formatos numéricos de 8 bits para neural profundo redes. Arxiv pré -impressão Arxiv: 2206.02915, 2022.

Nvidia. Melhorando o desempenho da rede dos sistemas HPC usando o NVIDIA Magnum Io NVSHMEM e GPUDIRECT Async. https://developer.nvidia.com/blog/improving-net work-performance-of-hpc-systems-using-nvidia-magnum-io-nvshmem-e-g Pudirect-async, 2022.

Nvidia. Blackwell Architecture. https://www.nvidia.com/en-us/data-center/tech Nologies/Blackwell-Architecture/, 2024a.

Nvidia. TransformEngine, 2024b. URL https://github.com/nvidia/transformere Ngine. Acessado: 2024-11-19.

Openai. Olá GPT-4O, 2024a. Url https://openai.com/index/hello-gpt-4o/.

Openai. Entendimento multilíngue de linguagem multitarefa múltipla (MMMLU), 2024b. Url https://huggingface.co/datasets/openai/mmmlu.

Openai. Apresentando o SimpleQa, 2024C. URL https://openai.com/index/introducting -Simpleqa/.

Openai. Apresentando o SWE-banch Verificado, estamos lançando um subconjunto de Sweban de validado humano que mais, 2024d. URL https://openai.com/index/introducing-swe bench -verified/.

B. Peng, J. Quesnelle, H. Fan e E. Shippole. YARN: Extensão eficiente da janela de contexto de grande modelos de idiomas. Arxiv pré -impressão arxiv: 2309.00071, 2023a.

H. Peng, K. Wu, Y. Wei, G. Zhao, Y. Yang, Z. Liu, Y. Xiong, Z. Yang, Z. Yang, B. N, B. Hu, et al. FP8-LM: Treinamento FP8 grandes modelos de idiomas. Arxiv pré -impressão Arxiv: 2310.18313, 2023b.

P. Qi, X. Wan, G. Huang e M. Lin. Paralelismo de oleoduto zero bolha. Arxiv pré -impressão ARXIV: 2401.10241, 2023A.

P. Qi, X. Wan, G. Huang e M. Lin. Paralelismo de oleoduto zero bolha, 2023b. URL HTTPS: //arxiv.org/abs/2401.10241.

Qwen. Relatório Técnico de Qwen. Arxiv pré -impressão Arxiv: 2309.16609, 2023.

Qwen. Apresentando Qwen1.5, 2024a. URL https://qwenlm.github.io/blog/qwen1.5.

Qwen. QWEN2.5: Um partido de modelos de fundação, 2024b. URL https://qwenlm.github.io/b LOG/QWEN2.5.

S. Rajbhandari, J. Rasley, O. Ruwase e Y. He. Zero: otimizações de memória para o treinamento de modelos de parâmetros do Trilel. No SC20: Conferência Internacional para Computação de Alto Desempenho, Networking, Armazenamento e Análise, Páginas 1–16. IEEE, 2020.

D. Rein, Bl Hou, AC Stickland, J. Petty, Ry Pang, J. Diranani, J. Michael e Sr Bowman. GPQA: referência de perguntas e respostas à prova de pós-graduação no Google. Arxiv pré -impressão Arxiv: 2311.12022,

BD Rouhani, R. Zhao, A. More, M. Hall, A. Khodamoradi, S. Deng, D. Choudhary, M. Cornea, E. Dellinger, K. Denolf, et al. Formatos de dados de microscaling para aprendizado profundo. Arxiv pré -impressão Arxiv: 2310.10537, 2023a.

BD Rouhani, R. Zhao, A. More, M. Hall, A. Khodamoradi, S. Deng, D. Choudhary, M. Cornea, E. Dellinger, K. Denolf, et al. Formatos de dados de microscaling para aprendizado profundo. ARXIV ARXIV ARXIV: 2310.10537, 2023B.

K. Sakaguchi, RL Bras, C. Bhagavatula e Y. Choi. WinRogrande: um Winograd adversário Desafio de esquema em escala, 2019.

Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, M. Zhang, Y. Li, Y. Wu e D. Guo. Deepseekmath: empurrando os limites do raciocínio matemático em modelos de linguagem aberta. ARXIV ARXIV ARXIV: 2402.03300, 2024.

N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Qv Le, Ge Hinton e J. Dean. Redes neurais escandalosamente grandes: a camada de mistura de especialistas escassamente demitida. Na 5ª Conferência Internacional sobre Representações de Aprendizagem, ICLR 2017. OpenReview.net, 2017. URL https: //openreview.net/forum?id=b1ckmdqlg.

F. Shi, M. Suzgun, M. Freitag, X. Wang, S. Srivats, S. Vosoughi, HW Chung, Y. Tay, S. Ruder, D. Zhou, D. Das e J. Wei. Modelos de idiomas são raciocínio multilíngue da cadeia de pensamentos. Na décima primeira Conferência Internacional sobre Representações de Aprendizagem, ICLR 2023, Kigali, Ruanda, 1-5 de maio de 2023. OpenReview.net, 2023. URL https://openreview.net/forum?id=fr3wgck-ixp.

Y. Shibata, T. Kida, S. Fukamachi, M. Takeda, A. Shinohara, T. Shinohara e S. Arikawa Codificação de pares: um esquema de compressão de texto que acelera a correspondência de padrões. 1999.

J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo e Y. Liu. ROFORMER: transformador aprimorado com rotativo Posição de incorporação. Neurocomputing, 568: 127063, 2024.

K. Sun, D. Yu, D. Yu e C. Cardie. Investigando conhecimento prévio para desafiar chinês Compreensão de leitura de máquinas, 2019a.

M. Sun, X. Chen, Jz Kolter e Z. Liu. Ativações maciças em grandes modelos de linguagem. arxiv pré -impressão arxiv: 2402.17762, 2024.

X. Sun, J. Choi, C.-Y. Chen, N. Wang, S. Venkataramani, Vv Srinivasan, X. Cui, W. Zhang e K. Gopalakrishnan. Treinamento híbrido de ponto flutuante de 8 bits (HFP8) e inferência por redes neurais profundas. Avanços nos sistemas de processamento de informações neurais, 32, 2019b.

M. Suzgun, N. Scales, N. Schärli, S. Gehrmann, Y. Tay, HW Chung, A. Chowdhery, Qv LE, Eh Chi, D. Zhou, et al. Desafiar tarefas de grande parte e se a cadeia de pensamento pode resolvê-las. Arxiv pré -impressão Arxiv: 2210.09261, 2022.

V. Thakkar, P. Ramani, C. Cecka, A. Shivam, H. Lu, E. Yan, J. Komemen, M. Hoemmen, H. Wu, A. Ker, M. Nice, D. Merrill, D. Blasig, F. Qiao, P. Majcher, P. Springer, M. Hohnerbach, J. Wang e M. Gupta. Cutlass, janeiro. 2023. URL https://github.com/nvidia/cutlas s.

H. Tourvon, T. Lavril. Lachaux, T. Law, B. Rosary, N. Goyal, E. Azhar, F. Azhar e Al. LLAMA: Modelos de linguagem de função aberta e de eficiência. O XIV: 2302.13971, 2023A.

H. Tourvron, L. Martin, K. Stone, A. Canton-Ferrer.

R. Hou, H. Inan, M. Kordas, V. Kerkez, M. Khabsa, I. Kloumann, A. Krenev, Ps Koura, M. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y., Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Runta, K. Saladi, A. Schelten, R. Silva, em. Smith, R. Sublawaian, Xe Tan, B. Tang, R. Taylor, A. Williams, JX Kuan, p. Narang, A. Rodriguez, R. Stojnic, S. Edudov e T. Scialom. LLAMA 2: Fundação e fundação aberta e modelos de bate-papo ajustados. CORR, ABS/2307.09288, 2023B. Doi: 10.48550/arxiv.2307.0 URL https://doi.org/10.48550/arxiv. 09288.

A. Vaswani, N. Shazer, N. Parmar, J. Uszkoreit, L. Jones, um Gomez ,. Kaiser e I. Polo-Sukhin. Atenção é tudo o que você precisa. Avanços nos sistemas de processamento de informações neurais, 30,

L. Wang, H. Gao, C. Zhao, X. Sun e D. Dai. Estratégia de balanceamento de carga livre de perda de perda de perda para a mistura de especialistas. CORR, ABS/2408.15664, 2024A. URL https://doi.org/10.48550/arx iv.2408.15664.

Y. Não, X Sim, A. Zhuang, R. Fan, X. Ye e W. Chen. São mais vagas e desafios o idioma multi-skado em referência. CORR, ABS/2406.2406, 2024, 2024B. URL https: /rii.org/10.4850/arxiv.2406.

T. Wei, J. Luan, W. Liu, S. Dong e B. Wang. Cmath: seu modelo de idioma pode passar chinês Teste de matemática da escola primária?, 2023.

M. Wortsman, T. Dettmers, L. Zettlemoyer, A. Morcos, A. Farhadi e L. Schmidt. Treinamento estável e de baixa precisão para modelos de linguagem de visão em larga escala. Avanços nos sistemas de processamento de informações neurais, 36: 10271-10298, 2023.

H. Xi, C. Li, J. Chen e J. Zhu. Transformadores de treinamento com números inteiros de 4 bits. Avanços no neural Sistemas de Processamento de Informações, 36: 49146-49168, 2023.

CS Xia, Y. Deng, S. Dunn e L. Zhang. AGENTELLESS: Software baseado em LLM desmistificante Agentes de engenharia. Arxiv Preprint, 2024.

H. Xia, T. Ge, P. Wang, S. Chen, F. Wei e Z. Sui. Decodificação especulativa: Explorando a execução específica para acelerar a geração SEQ2SEQ. Nos resultados da Associação de Linguística Computacional: EMNLP 2023, Cingapura, 6 a 10 de dezembro de 2023, páginas 3909-3925. Association for Computational Linguistics, 2023. URL https://doi.org/10.18653/v1/2023.findings-emnlp.257.

G. Xiao, J. Lin, M. Seznec, H. Wu, J. Demouth e S. Han. Smoothquant: quantização pós-treinamento precisa e eficiente para modelos de linguagem grandes. Em Conferência Internacional sobre Aprendizado de Máquinas, páginas 38087-38099. PMLR, 2023.

L. Xu, H. S, X Leave, B. Cui, Ju, J Zhao, Q. Zhao, C. Ye, X. Zhang, Z. Yang, K. Richardson e Z. Pista: uma linguagem de fofoca referência em alta. Na 28ª Confissão Internacional da Linista Confessional, dezembro de 2020, Espanha, Espanha, Espanha, Espanha. International Communional Linistration, 2020. DOI: URL https: /do.10.18653/v1/v1/v1/2020.4020.

R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi e Y. Choi. Hellaswag: Uma máquina pode realmente terminar sua frase? Em A. Korhonen, Dr. Traum e L. Màrquez, editores, Anais da 57ª Conferência da Associação de Linguística Computacional, ACL 2019, Florence, Itália, 28 de julho a 2 de agosto de 2019, volume 1: Long Papers, Páginas 4791 –4800. Association for Computational Linguistics, 2019. doi: 10.18653/v1/p19-1472. URL https://doi.org/10.18653/v1/p1 9-1472.

W. Zhong, R. Cui, Y. Guo, S. Luang, S. Lung, Y. AGIEVAL: Um juiz de referência para os modos de avaliação. CRR, ABS/2304, 2023. doi: 10.4850/arxiv.2304.2304. URL https: /rii.org/10.4850/arxiv.2304.

J. Zhou, T. Lu, S. Mishra, S. Brahma, S. Basu, Y. Luan, D. Zhou e L. Hou. Seguidores de instruções Avaliação para grandes modelos de linguagem. Arxiv pré -impressão Arxiv: 2311.07911, 2023.

Apêndice

A. Contribuições e Agradecimentos

Pesquisa e engenharia
Aixin Liu Bing Xue

Bingxuan wang
bochao wu
chengda lu
chenggang
zhao chengqi
deng chenyu
zhang* chong
ruan damai dai
diara guo dejian
yang deli chen
erhang li
fangyun lin
fucong dai fuli
lu* guangbo
hao guanuntingi
rinei li hi h.
Haowei Zhang
Honghui Ding
Huajian Xin*
Huazuo Gao
Hui Qu

LECONG ZHANG
Liang Zhao Litong
Wang Liyue
Zhang Mingchuan
Zhang Minghua
Zhang Minghui
Tang panpan
Huang peiyi wang
qiancheng wang
qiu zhu qinyu
chen qiushi du
Ruiqi ge ge

Ruisong zhang
ruizhe pan runji
wang runxin xu
ruoyu zhang
shanghai lu
shangyan zhou
shanhuang chen
shengfeng ye
shirong ma
shiyu wang
shuiping yu
shunfeng zhou s
Huting Pan Tao
yun tian pei

Jianzhong Guo
Jiashi Li Jiawei
Wang*
Jingchang
Chen Jingyang
Yuan Junjie
Qiu Junlong Li
Junxiao Song
Kai Dong Kai
Hu* Kaige Gao
Kang Guan
Kexin Huang
Kuai Yu Lean
Wang

Wangding Zeng
Wanjia Zhao*
Wen Liu
Wenfeng Liang
Wenjun Gao
Wenqin Yu
Wentao Zhang
Xiao Bi
Xiaodong Liu
Xiaohan Wang
Xiaokang Chen
Xiaokao Zhang
Xiaota Nie Xin
Cheng Xin Liu

Xin xie
xingchao liu
xingkai yu
xinyu yang
xinyuan li
xuecheng su
xuheng lin yk li
yq wang yx wei
yang zhang
yanhong xu
yao li yao zhao
yaofeng sun
yaohui wang yi
yu yiathann
Yiyang Ma *
Yiyuan Liu
Yongqiang Guo
Yu Wu Yuan
Ou Yuduan
Wang Yue
Gong Yuheng
Zou Yujia He
Yunfan Xiong
Yuxiang Luo
Yuxiang You
Yuxuan Liu
Yuyang Zhou
ZF Wu ZZ Ren
Zehui Ren
Zhangli Sha
Zhe Fu Zhenda
Xie Zhengyan
Zhang Zhewen
Hao Zhibin
Gou Zhicheng
Ma

Zhigang yan
zhihong
shao zhiyu
wu zhuoshu
li zihui gu
zijia zhu
zijun liu* zilin
li ziwei xie
ziyang
canção ziyi
gao zizheng
pan

Anotação de dados Beijing

feng hui li jl
cai jiaqi ni lei
xu meng li
ning tian rj
chen rl jin

Ruyi Chen SS
Li Shuang Zhou
Tianyu Sun XQ
Li Xiangyue Jin
Xiaojin Shen
Xiaosha Chen
Xiaowen Sun
Xiaoxiang
Wang Xinnan
Song Xinyi
Zhou YX Zhu
Yanhong Xu
Yan ping Huang
Yaohui Li Yi
Zheng Yuchen
Zhu Yunxian
Ma Zhen
Huang Zhipeng
Xu Zhongyu
Zhang

Negócios e conformidade
Dongjie Ji

Jian Liang Jin
Chen Leyi
Xiao Miaojun
Wang
Mingming Li
Peng Zhang
Shaoqing Wu
Shengfeng Ye
T. Wang

Wl xiao wei e
xianzu wang
xinxia shan
ying tang
yukun zha
yuting yan
zhen zhang

Dentro de cada função, os autores são listados em ordem alfabética pelo primeiro nome. Nomes marcados com * denotam indivíduos que se afastaram de nossa equipe.

B. Estudos de ablação para treinamento de baixa precisão

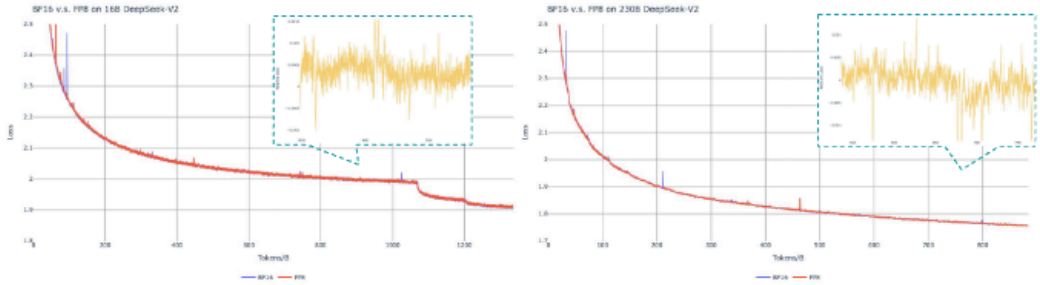


Figura 10 | Comparação de curvas de perda entre o treinamento BF16 e FP8. Os resultados são suavizados pela média móvel exponencial (EMA) com um coeficiente de 0,9.

B.1. Treinamento FP8 vs BF16

Validamos nossa estrutura de precisão mista FP8 com uma comparação com o treinamento BF16 no topo de dois modelos de linha de base em diferentes escalas. Na pequena escala, treinamos um modelo BASELING MOE, compreendendo aproximadamente 16B parâmetros totais em tokens de 1,33T. Em larga escala, treinamos um modelo BASELING MOE, compreendendo aproximadamente 230B parâmetros totais em cerca de 0,9T. Mostramos as curvas de treinamento na Figura 10 e demonstramos que o erro relativo permanece abaixo de 0,25%, com nossa acumulação de alta precisão e estratégias de quantização de granulação fina.

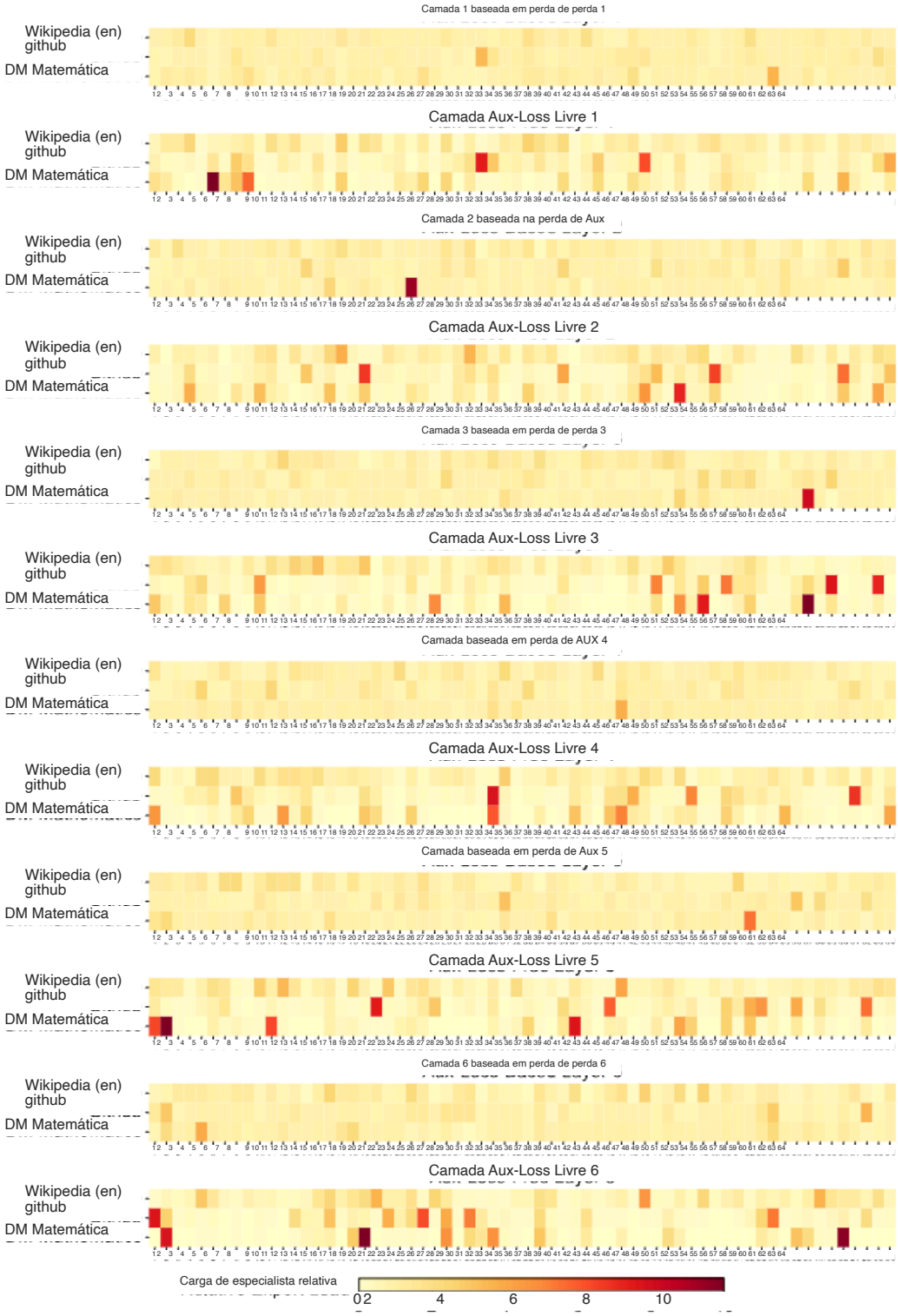
B.2. Discussão sobre quantização em bloco

Embora nossa quantização de grão fina em ladrilhos mitiga efetivamente o erro introduzido pelos outliers de características, ele requer diferentes agrupamentos para quantização da ativação, ou seja, 1x128 no PASS para a frente e 128x1 para passe para trás. Um processo semelhante também é necessário para o gradiente de ativação. Uma estratégia direta é aplicar a quantização em bloco por elementos 128x128, como a maneira como quantizamos os pesos do modelo. Dessa maneira, apenas a transposição é necessária para trás. Portanto, realizamos um experimento em que todos os tensores associados ao DGRAD são quantizados em uma base em bloco. Os resultados revelam que a operação de DGrad, que calcula os gradientes de ativação e propaga de volta para camadas rasas de maneira semelhante à corrente, é altamente sensível à precisão. Especificamente, a quantização em bloco de gradientes de ativação leva a

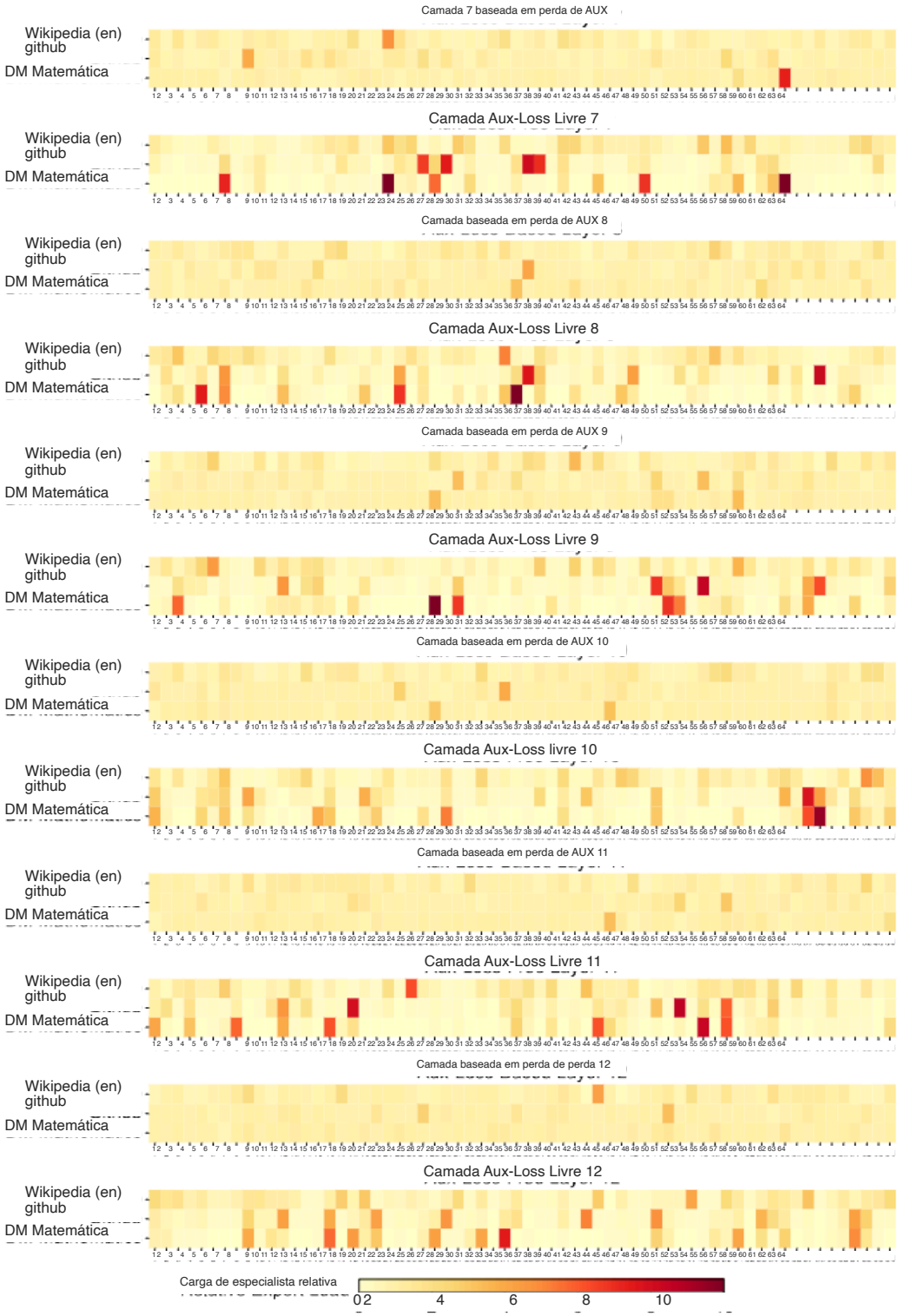
Divergência do modelo em um modelo MOE compreendendo aproximadamente 16b parâmetros totais, treinados para cerca de 300b. Nossa hipótese é de que essa sensibilidade surge porque os gradientes de ativação são altamente desequilibrados entre os tokens, resultando em outliers corlacionados de token (Xi et al., 2023). Esses discrepantes não podem ser efetivamente gerenciados por uma abordagem de quantização em bloco.

C. Padrões de especialização especializados dos modelos de 16b Aux-Loss e Aux-Loss-Free

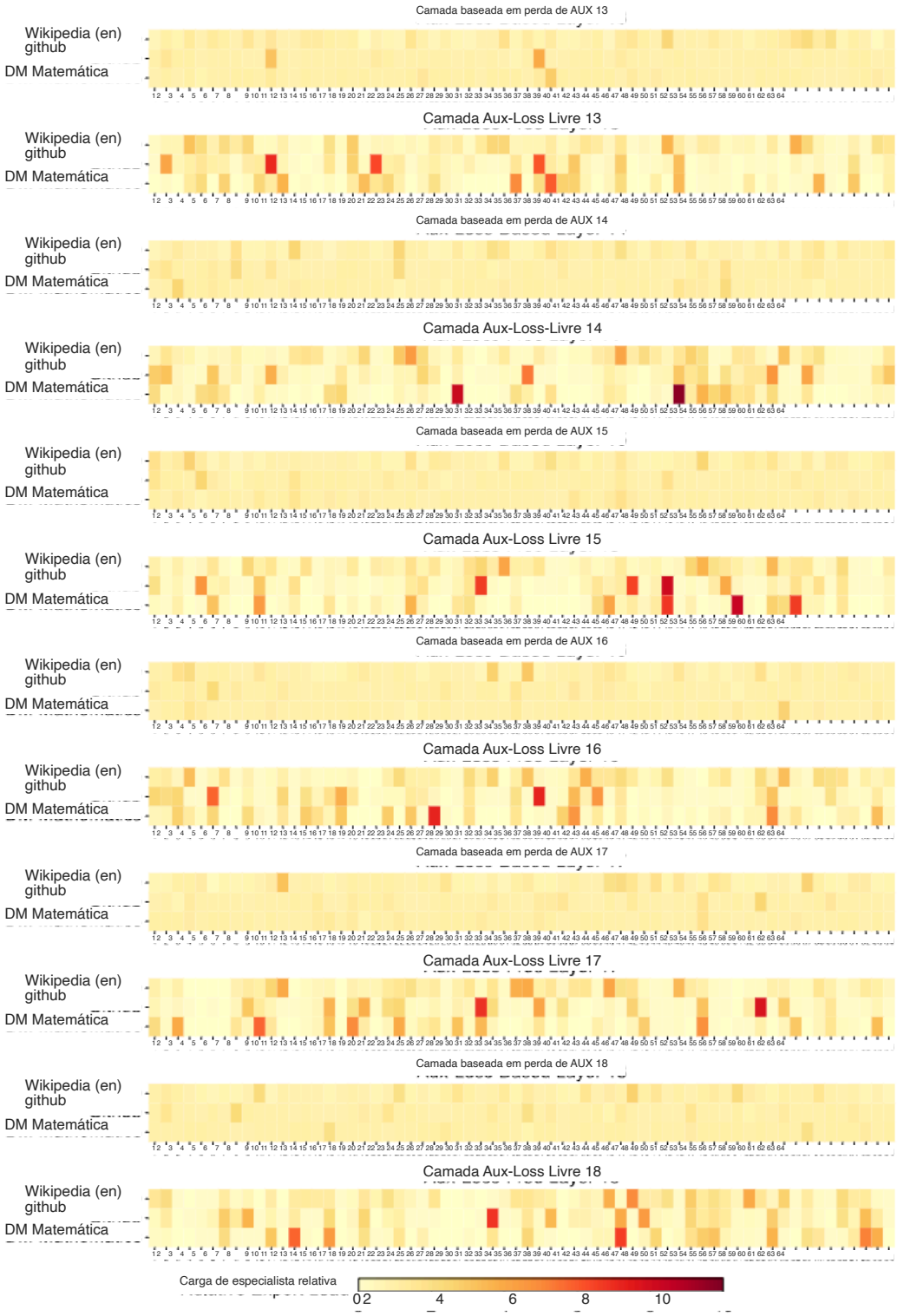
Registramos a carga especializada da linha de base baseada em perda auxiliar de 16B e o modelo livre de perda de perda auxiliar no conjunto de testes de estaca. O modelo livre de perdas auxiliares tende a ter maior especialização especializada em todas as camadas, como demonstrado na Figura 10.


Camada 1 baseada em perda de perda 1
Camada Aux-Loss Livre 1
Camada 2 baseada na perda de Aux
Camada Aux-Loss Livre 2
Camada 3 baseada em perda de perda 3
Camada Aux-Loss Livre 3
Camada baseada em perda de AUX 4
Camada Aux-Loss Livre 4
Camada baseada em perda de Aux 5
Camada Aux-Loss Livre 5
Camada 6 baseada em perda de perda 6
Camada Aux-Loss Livre 6
Wikipedia (en)
github
DM Matemática
Carga de especialista relativa
0
2
4
6
8
10

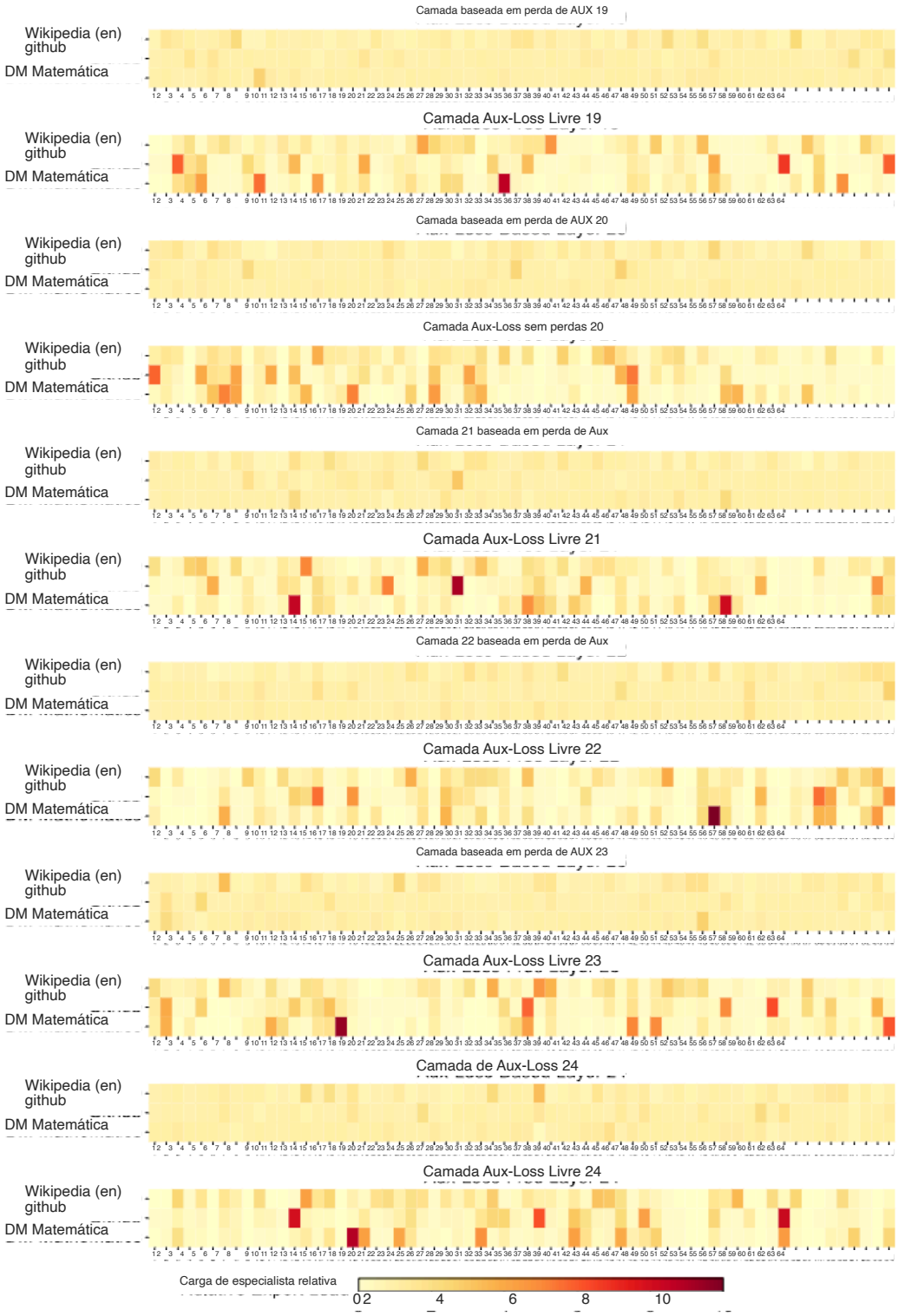
(a) Camadas 1-7



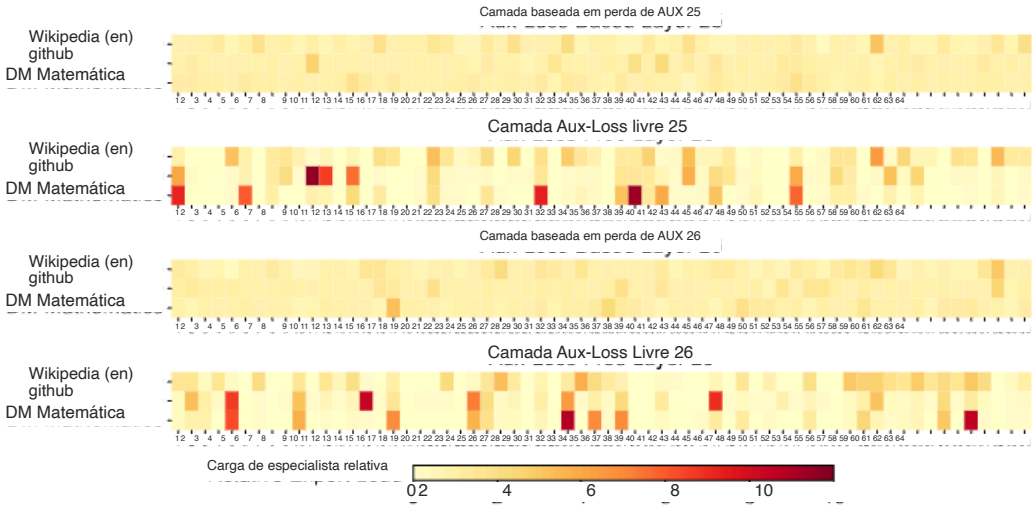
(b) Camadas 7-13



(c) Camadas 13-19



(d) Camadas 19-25



(e) Camadas 25-27

Figura 10 | Modelos especializados em modelos baseados em perda de perda auxiliar e com perda de perda auxiliar em três domínios no conjunto de testes de pilha. O modelo Auxiliar-Loss Livre mostra maiores padrões de especialização de especialistas do que o de base de perda auxiliar. A carga de especialista relativa indica a proporção entre a carga especialista real e a carga especializada teoricamente equilibrada.