

Regressão Linear com Computação Aproximada

Pedro e Silva¹, Hytalo Bassi¹, Jailton Nunes Filho¹, Eduardo Paes¹,
Jhullya Martins¹, Daniela Catelan¹

¹ Faculdade de Computação – Universidade Federal de Mato Grosso do Sul (FACOM - UFMS)
Av. Costa e Silva, s/n – Campo Grande – MS – Brasil

{pedro.candia, hytalo.bassi, jailton.nunes-filho}@ufms.br

{eduardo-paes, jhullya.e.s.martins, daniela.catelan}@ufms.br

Abstract. *Training Artificial Intelligence (AI) demands high energy and computational costs. To mitigate this problem, this paper investigates the application of Approximate Computing (AC) in the Mini-Batch Stochastic Gradient Descent (MGBD) algorithm. Using the PACA framework, an automated Design Space Exploration (DSE) was conducted to inject approximate arithmetic instructions at the hardware level (RISV-V ISA). Results demonstrate the feasibility of achieving up to 62.68% energy savings and 62.39% to execution cycles while keeping the model degradation (NRMSE) below 30%. This highlights the effectiveness of AC and DSE tools in noise-resilient algorithms.*

Resumo. *O treinamento de modelos de Inteligência Artificial (IA) demanda alto custo energético e computacional. Para mitigar esse gargalo, este artigo investiga a aplicação de Computação Aproximada (CA) no algoritmo Mini-batch Stochastic Gradient Descent (MBGD). Utilizando o framework PACA, automatizou-se a Exploração do Espaço de Projeto (DSE) para injetar instruções aritméticas aproximadas em nível de hardware (ISA RISC-V). Os resultados demonstram a viabilidade de obter reduções de até 62,68% no consumo de energia e 62,39% nos ciclos de execução, mantendo a degradação do modelo (NRMSE) inferior a 30%. Isso evidencia a eficácia da CA e das ferramentas de DSE em algoritmos resilientes a ruídos.*

1. Introdução

Com a popularização da IA em aplicações de larga escala, o desenvolvimento e o treinamento de grandes modelos têm demandado custos computacionais e energéticos cada vez maiores. Para mitigar esse gargalo, a CA [Xu et al. 2016] propõe uma tolerância controlada a erros em troca de reduções no tempo de execução e no consumo de energia, além de ganhos gerais de desempenho.

Este trabalho investiga o uso de técnicas de CA no treinamento de modelos, focando especificamente no algoritmo MBGD, escolhido por sua resiliência intrínseca a ruídos devido às suas atualizações iterativas. A estratégia metodológica consiste em aplicar aproximações matemáticas diretamente no nível de hardware, por meio da inserção de instruções aproximadas [Catelan 2025] e utilizando o do simulador SPIKE[SPIKE 2019].

Tendo em vista que a seleção de quais operações matemáticas devem ser aproximadas gera um vasto número de configurações, a DSE manual torna-se exaustiva e

inviável. Para contornar esse desafio, o processo foi automatizado utilizando o framework PACA (Project for Approximate Computing Analysis) [K. Neto 2025], que aplica abordagens heurísticas de busca. Assim, o principal objetivo deste estudo é demonstrar a viabilidade da aplicação de técnicas de CA em algoritmos de aprendizado de máquina, assegurando que a precisão final dos resultados se mantenha em níveis aceitáveis.

2. Metodologia

Optou-se pelo algoritmo de MBGD aplicado à Regressão Linear Simples [Géron 2025]. Para estabilizar a magnitude dos gradientes sob aritmética aproximada e acelerar a convergência ao longo de 100.000 épocas, integrou-se o mecanismo de taxa adaptativa RMS-Prop (*Root Mean Square Propagation*) [Tieleman and Hinton 2012], com taxa de aprendizado inicial fixada em $\alpha = 0,03$. O ponto crítico (hotspot) para a CA está no laço interno do algoritmo, contendo 8 operações de ponto flutuante (multiplicações de parâmetros, subtrações de erro, acumulação e divisões). A substituição dessas operações exatas pelas aproximadas na ISA gera um espaço de busca combinatório de $2^8 = 256$ do algoritmo original. Além disso, para a análise da Regressão Linear utilizamos a reta original: $y = 0.79x + 2.85$.

Utilizou-se o framework PACA para automatizar a DSE. A ferramenta substitui as operações por instruções aproximadas em diferentes configurações na ISA RISC-V, compila as variantes via toolchain GNU e as executa no simulador funcional SPIKEX [Catelan 2025]. A partir dos logs de execução, extraem-se o total de instruções executadas, os ciclos de execução e o consumo energético estimado. Como métricas de erro para avaliar as variantes do código foram escolhidas: o Erro Quadrático Médio (RMSE), a sua versão normalizada (NRMSE) e a Taxa de Economia Energética, avaliada por $\Delta E = \left(1 - \frac{E_{\text{aprox}}}{E_{\text{exato}}}\right) \times 100\%$. O conjunto de pontos utilizado no treinamento e a reta exata de regressão (sem CA) que servem de base para a avaliação estão ilustrados na Figura 1. As métricas utilizadas para o cálculo do consumo energético foram baseadas no modelo de energia utilizado no trabalho de [Catelan 2025].

3. Resultados e Discussões

Foram geradas 64 variantes do algoritmo original. A Tabela 1 apresenta as 13 variantes aprovadas pelo filtro de viabilidade $\text{NRMSE} < 30\%$. A análise dos dados revela que inserir mais operações imprecisas não garante, necessariamente, maior eficiência no hardware. O principal destaque é a Variante 13, que substituiu apenas uma instrução exata por uma aproximada (FMULX) e obteve os melhores resultados globais: menor erro e redução de 62,68% no consumo energético.

A Figura 1 ilustra visualmente esse comportamento, mostrando que a reta preditiva da Variante 13 acompanha quase perfeitamente o baseline, o que valida a resiliência do algoritmo à aproximação inserida.

4. Conclusão

Este trabalho comprovou a viabilidade da CA em nível de hardware no treinamento do algoritmo MBGD. Os resultados apresentaram reduções energéticas de até 62,68%, mantendo a degradação do modelo perfeitamente contida ($< 30\%$). Como trabalhos futuros, pretende-se investigar estratégias de controle de erro das variantes e utilizar o MBGD aproximado no treinamento de modelos reais para avaliar seu impacto prático. O pre-

Tabela 1. Análise de variantes, aproximações e métricas de erro.

Variantes	Aproximação	Potência (mW)	Total Instruções	Ciclos	RMSE	NRMSE
Original	-	45245074	14081817	13449244	-	-
1	2 FSUBX	20671899	6257915	6257915	0.271756	0.271756
2	1 FSUBX	20758078	6093056	6256630	0.182756	0.035112
3	1 FMULX, 2 FSUBX	30653403	6893096	9056678	0.181447	0.034860
4	1 FSUBX	21044119	6095682	6258157	0.104367	0.020051
5	1 FMULX, 2 FSUBX	23856807	6395093	7057751	1.387959	0.266659
6	2 FSUBX	20953756	6095103	6257753	1.387959	0.266659
7	3 FSUBX	26305090	6495963	7858261	1.431021	0.274932
8	1 FMULX, 1 FSUBX	23590580	6393046	7056628	0.182756	0.035112
9	1 FMULX, 3 FSUBX	30568029	6895965	9058267	1.431021	0.274932
10	1 FMULX, 2 FSUBX	23504401	6395250	7057913	1.414488	0.271756
11	1 FMULX, 1 FSUBX	23947170	6395672	7058155	0.104367	0.020051
12	2 FSUBX	26390464	6493094	7856672	0.181447	0.034860
13	1 FMULX	16884749	5895456	5058256	0.005000	0.000961

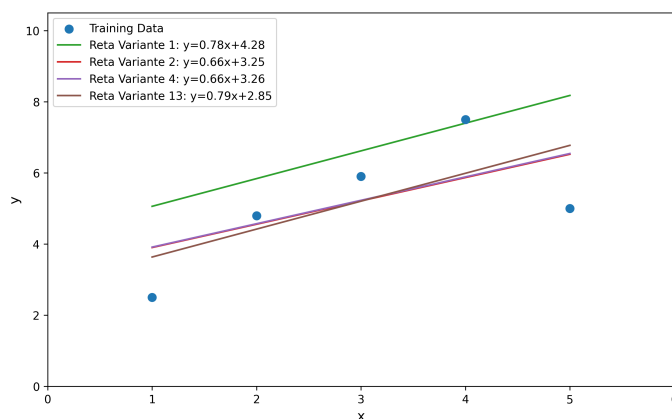


Figura 1. Variantes com maior economia de energia.

sente trabalho foi realizado com apoio da Universidade Federal de Mato Grosso do Sul – UFMS/MEC – Brasil.

Referências

- Catelan, D. L. (2025). Approximate computing: Contributions to the design of arithmetic circuits and instruction-set architectures. *Repositório Institucional da UFMS*.
- Géron, A. (2025). *Hands-On Machine Learning with Scikit-Learn and PyTorch*. O'Reilly Media.
- K. Neto, G. e. a. (2025). Heuristics for energy-efficient instruction-level approximate computing. *SBAC-PAD*.
- SPIKE (2019). Spike risc-v isa simulator. Available at: <https://github.com/riscv-software-src/riscv-isa-sim>. Accessed: June/2025.
- Tieleman, T. and Hinton, G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31.
- Xu, Q., Mytkowicz, T., and Kim, N. S. (2016). Approximate computing: A survey. *IEEE Design Test*, 33(1):8–22.