15º Congresso de Inovação, Ciência e Tecnologia do IFSP - 2024

IMPLEMENTAÇÃO DE UMA FUNÇÃO SIGMOIDE APROXIMADA PARA UTILIZAÇÃO EM REDES NEURAIS ARTIFICIAIS EMBARCADAS EM CIRCUITO FPGA PARA COMPUTAÇÃO DE BORDA

VINICIUS DE AZEVEDO BOSSO[1], MIGUEL ANGELO DE ABREU DE SOUSA[2], RICARDO PIRES[3]

[1] Graduando em Engenharia de Controle e Automação, Bolsista CNPq PIBITI, IFSP, Campus São Paulo, vinicius.bosso@aluno.ifsp.edu.br.

[2] Prof. Dr. em Sistemas Eletrônicos, IFSP, Campus São Paulo, angelo@ifsp.edu.br.

[3] Prof. Dr. em Sistemas Automáticos e Microeletrônicos, IFSP, Campus São Paulo, ricardo_pires@ifsp.edu.br.

Área de conhecimento (Tabela CNPq): 3.04.03.03-0 Circuitos Eletrônicos

RESUMO: As Redes Neurais têm se destacado dentro da área da Inteligência Artificial por permitir aplicações em diferentes campos do conhecimento. Tipicamente, os modelos neurais são executados em computadores com *software* específicos. Entretanto, para atender aos requisitos de algumas aplicações tecnológicas recentes, é necessário executar a rede neural em "computação de borda" ("*edge computing*"), ou seja, de forma portátil em circuitos embarcados. Tal implementação, evita problemas de violação de dados e instabilidade da conexão de acesso, recorrentes na execução de redes neurais processadas remotamente, ou seja, em nuvem. A escolha do "*Field Programmable Gate Array*" (FPGA) como *hardware* para a computação de borda possibilita atender aos quesitos estabelecidos e, além disso, permite a construção de um circuito elétrico massivamente paralelo, de modo a reproduzir a arquitetura fundamental de modelos neurais. Por outro lado, tal dispositivo não possui uma unidade aritmética pronta, o que impede a implementação direta em *hardware* dos cálculos aplicados nos neurônios artificiais, devido às operações envolvidas, como na função de ativação sigmoide, largamente utilizada em redes neurais. Atualmente, estudos com aproximações feitas para a função sigmoide possuem atrasos combinacionais e ocupam armazenamentos do FPGA que podem ser reduzidos, fatores pertinentes para aplicações em áreas como a Internet das Coisas (IoT), por exemplo. Este projeto propõe, portanto, a configuração de um circuito FPGA que execute a aproximação de uma função sigmoide para aplicações portáteis e embarcadas com o menor armazenamento e atraso combinacional registrados incluindo a relação com informações referentes ao erro médio e erro máximo da aplicação, quando comparado a outros trabalhos recentes do campo de estudo.

PALAVRAS-CHAVE: função sigmoide; aproximação; FPGA; circuitos embarcados; "edge computing"; inteligência artificial

IMPLEMENTATION OF AN APPROXIMATE SIGMOID FUNCTION FOR USE IN EMBEDDED ARTIFICIAL NEURAL NETWORKS IN FPGA CIRCUIT FOR EDGE COMPUTING

ABSTRACT: Neural Networks have stood out in the field of Artificial Intelligence because they allow applications in different fields of knowledge. Typically, neural models are executed on

computers with specific software. However, to meet the requirements of some recent technological applications, it is necessary to execute the neural network in “edge computing”, that is, in a portable way on embedded circuits. This implementation avoids problems of data breach and instability of the access connection, which are common when executing neural networks processed remotely, that is, in the cloud. The choice of “Field Programmable Gate Array” (FPGA) as hardware for edge computing makes it possible to meet the established requirements and, in addition, allows the construction of a massively parallel electrical circuit, in order to reproduce the fundamental architecture of neural models. On the other hand, such a device does not have a ready-made arithmetic unit, which prevents the direct implementation in hardware of the calculations applied to artificial neurons, due to the operations involved, such as the sigmoid activation function, widely used in neural networks. Currently, studies with approximations made for the sigmoid function have combinational delays and occupy FPGA storage that can be reduced, factors relevant for applications in areas such as the Internet of Things (IoT), for example. This project proposes, therefore, the configuration of an FPGA circuit that performs the approximation of a sigmoid function for portable and embedded applications with the lowest storage and combinational delay recorded, including the relationship with information regarding the average error and maximum error of the application, when compared to other recent works in the field of study.

KEYWORDS: sigmoid function; approximation; FPGA; embedded circuits; edge computing; artificial intelligence

INTRODUÇÃO

As Redes Neurais Artificiais (RNAs) são modelos computacionais constituídos por um conjunto de processadores simples (neurônios artificiais), todos interligados paralelamente por meio de canais de comunicação ponderados (conexões), com capacidade de aprendizado a partir de um conjunto de dados (base de dados) com exemplos coletados previamente (fase de aprendizado) e utilizá-los para processar informações ainda não apresentadas, mas que apresentam o mesmo padrão (fase de operação) (Haykin, 2001). Dentro da fase de operação dos neurônios que constituem a rede, é realizado o cálculo de um potencial de ativação (x), utilizado na função de ativação (g), a qual gera o valor de saída do neurônio (y), função (1). A função de ativação sigmoide é amplamente utilizada para problemas não-lineares.

O FPGA é um tipo de circuito integrado que possibilita a configuração de elementos lógicos e de roteamento livremente para a aplicação que será empregada por meio de operações lógicas internas em campo (Hauck, 1998), além disso, garante baixo custo e consumo de energia comparado a outros circuitos digitais para aplicações embarcadas (Sousa, 2018) e uma construção massivamente paralela (Costa, 2013), útil para processar os cálculos dos neurônios das RNAs simultaneamente, garantindo maior eficiência. Trabalhos atuais como Qin et al. (2020), Pan et al. (2022) e Hajduk (2018), abordam diferentes técnicas pertinentes para aproximar a sigmoide em FPGA. No entanto, áreas como a IoT exigem redes com o menor atraso possível para manipular dados de sensores (Qian et al., 2020) e redes extensas precisam de uma função de ativação que ocupe o menor espaço praticável do dispositivo.

O objetivo deste trabalho, portanto, é mostrar como a função sigmoide foi segmentada e aproximada para equações simplificadas e as conclusões obtidas em questão dos resultados de erro máximo e médio da função abordada em um intervalo de valores comparado com a função ideal, levando em consideração as limitações estabelecidas pela utilização dos valores em binário com ponto fixo na configuração do FPGA e o desempenho da aplicação contribuindo com um novo método de aproximação que ocupa menos área do *chip* e tem maior velocidade de processamento.

MATERIAL E MÉTODOS

O presente trabalho opta pela escolha do dispositivo FPGA com base na comparação com outras tecnologias digitais estudadas como plataforma para execução direta de RNAs, nomeadamente, “Graphics Processing Unit” (GPU) (Dally et al., 2021) e “Central Processing Unit” (CPU) (Hennessy & Patterson, 2011). O *hardware* escolhido se destaca, principalmente, em flexibilidade de

modificações e custo do componente, além de permitir alto paralelismo de computação e baixo consumo de energia (Sousa, 2018).

Para embarcar a função sigmoide, foram utilizadas propostas de Sousa e Torres (2013) que possuem o intuito de aproximar o cálculo da função (1), com o uso de funções de primeiro e segundo grau, escritas apenas com multiplicações e adições, o que garante a aplicabilidade na linguagem de configuração do dispositivo FPGA. Todos os valores utilizados no código de configuração da placa que aproximam o formato da sigmoide foram definidos em linguagem de programação Python e ajustados para serem inseridos na execução do *hardware* de forma aproximada.

$$y = g(x) = \frac{1}{1 + e^{-x}} \quad (1)$$

Em Sousa e Torres (2013), a curva da sigmoide é dividida em partes. Assim, é necessário definir parâmetros fixos para o potencial de ativação (dividido em Xmin, Xmed, Xmax) e seguir uma regra condicional para que um valor de potencial de ativação "X" inserido dentro de um intervalo entre os parâmetros seja usado como entrada para um cálculo que define a saída da função (y), visto na Figura 1. Notou-se a necessidade de acrescentar mais dois parâmetros (Xinf, Xsup) para o potencial de ativação, pois, caso um valor x inserido seja menor que Xinf, recebe o valor 0 e, quando x inserido é maior que Xsup recebe, imediatamente, o valor 1.

As equações que abrangem y são inseridas na configuração do FPGA condicionadas de acordo com os parâmetros anteriores, equação (2). Para realizá-las, é necessário o valor de x inserido e constantes definidas por equações específicas para cada (Sousa & Torres, 2013).

$$y = \begin{cases} 0\ (zero), & x_{inf} \geq x \\ c_0 + b_0 * x, & x_{inf} < x < x_{min} \\ c_1 + x * (b_1 + a_1 * x), & x_{min} \leq x < x_{med} \\ c_2 + x * (b_2 + a_2 * x), & x_{med} \leq x < x_{max} \\ c_3 + b_3 * x, & x_{max} \leq x < x_{sup} \\ 1\ (um), & x_{sup} \leq x \end{cases} \quad (2)$$

A linguagem de programação utilizada para configuração da placa, modelo "Basys" da marca Digilent, foi a "*Very high speed integrated circuit Hardware Description Language*" (VHDL) no software "ISE" versão 14.7, tendo em vista, que foi utilizado o *chip* FPGA Xilinx família Virtex 7. As análises dos resultados das Figuras 1 e 2 foram realizadas em Python e os trabalhos de comparação da Tabela 1, obtidos de artigos sobre métodos de aplicação da função Sigmoide em FPGA.

RESULTADOS E DISCUSSÃO

Após definir os parâmetros para o potencial de ativação e calculadas as constantes, foi possível plotar os resultados de y para cada cálculo da saída da função, utilizando x no intervalo de [-10; 10].

As extremidades da sigmoide foram aproximadas por funções de primeiro grau, enquanto, a curva característica da função é dividida em duas partes e aproximada por funções de segundo grau, todas presentes na equação (2). Os valores dos parâmetros de x são ajustados até obter uma boa aproximação à função ideal, definida pela função (1), visível nas plotagens das Figura 1.

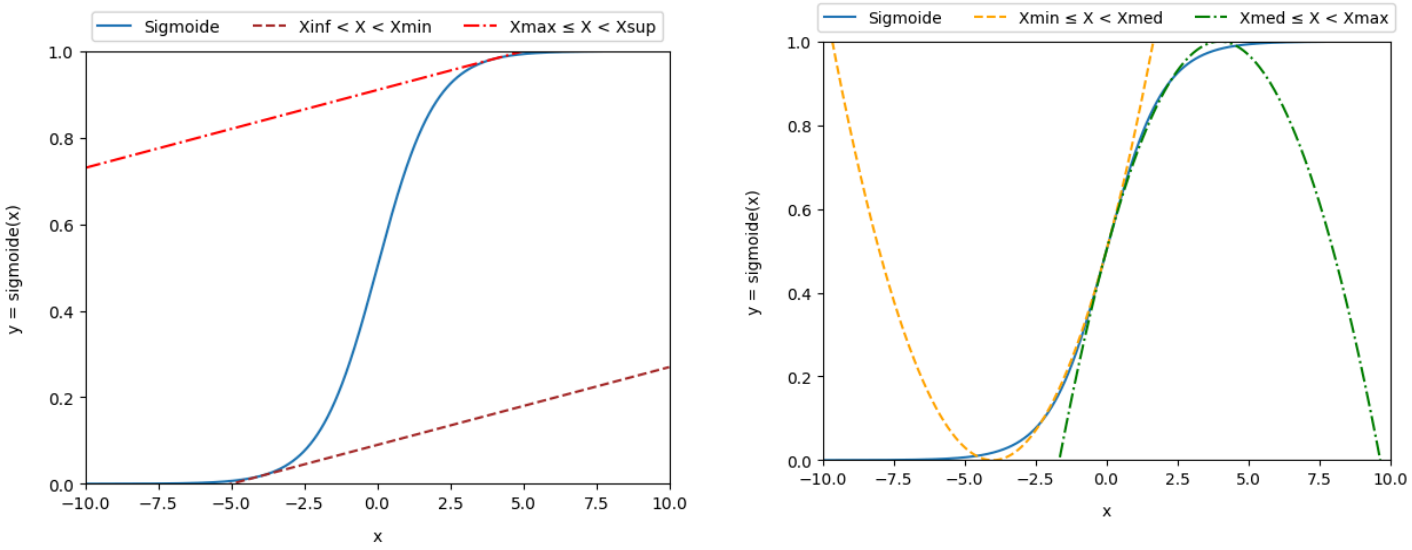

FIGURA 1. Na esquerda, em marrom e verm,elho estão plotadas as aproximações das extremidades da sigmoide em relação à função ideal em azul. Na direita, em amarelo e verde estão plotadas as aproximações da curva da sigmoide em relação à função ideal em azul.

Definidos os valores necessários para os cálculos, é preciso definir a quantidade de *bits* para representá-los em ponto fixo e na lógica de complemento de dois para configuração do FPGA. A escolha de 13 *bits* (com o *bit* mais significativo utilizado como sinal, outros 3 para a parte inteira dos valores e 9 para parte fracionária) mostrou-se apropriada, enquanto a saída y necessita de 11 *bits*, tendo em vista, que a resposta da função varia de 0 até 1, ou seja, necessita de apenas um *bit* para parte inteira do resultado. Com isso, a configuração é feita com base no fluxograma da Figura 2.

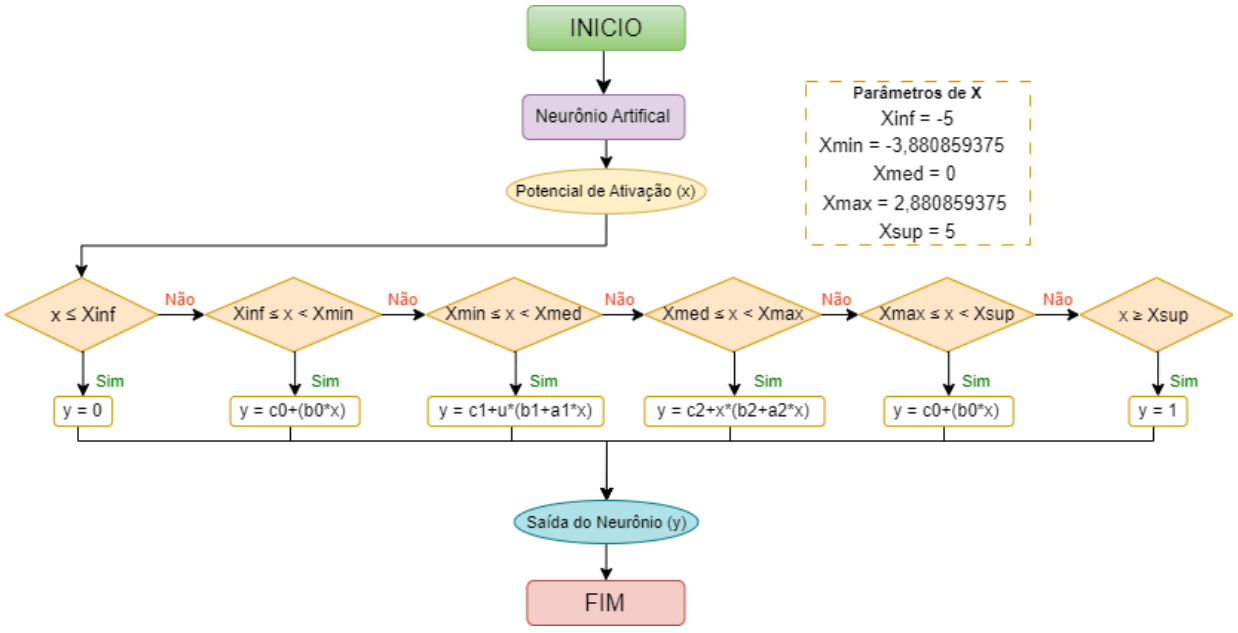

FIGURA 2. Fluxograma da configuração da função sigmoide aproximada proposta em FPGA

O erro da aproximação (máximo e médio) foi quantificado em Python, tendo em vista o intervalo de valores possíveis inseridos para x de [-8; 8[em passo da menor resolução binária descrita anteriormente. Com isso, obtiveram-se os resultados da Figura 3.

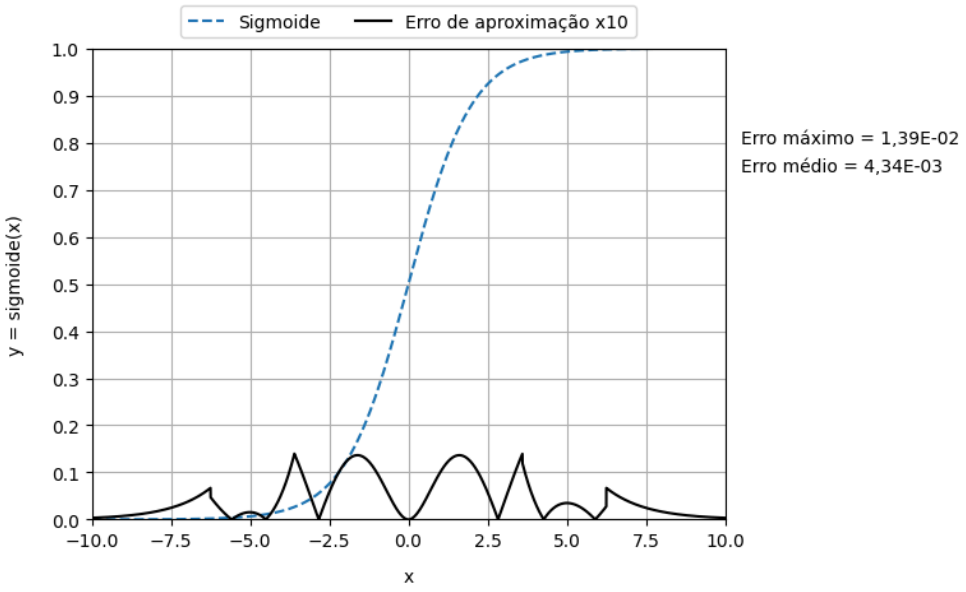

FIGURA 3. Plotagem do erro de aproximação (x10), na cor preta, da sigmoide embarcada em relação à função ideal, na cor azul com linha tracejada, com valores do erro máximo (Emáx) e erro médio (Eméd) obtidos

É possível obter informações no *software "ISE*" de configuração quanto ao desempenho obtido em relação à área do FPGA utilizada, quantificado em blocos lógicos *"Look-Up Tables"* (LUTs) e *"Flip Flops"* (FFs) além de informações relacionadas a velocidade das operações (máxima frequência de *Clock* e de *Delay* combinacional), dados apresentados na Tabela 1.

TABELA 1. Relação dos resultados obtidos neste trabalho com publicações anteriores em relação aos erros máximo e médio, número de LUTs e FFs, frequência de operação e *delay*, em que N/A, demonstra dados não citados pelo autor

Referência	Erro máximo	Erro médio	LUTs	FFs	Freq.(MHz)	Atraso(ns)
Este trabalho	1,39E-02	4,34E-03	119	159	119,22	2,80
Z. Qin et al., 2020	7,80E-03	N/A	158	46	N/A	3,00
Z. Pan et al., 2022 nra in/out16b	5,72E-04	8.60E-05	351	325	200	30,00
Z. Hajduk, 2018	1,19E-07	1.45E-08	1916	792	89,3	940,80

Na Tabela 1, é possível notar que a área ocupada do dispositivo e o atraso da aplicação são menores em comparação com outros estudos, o que torna relevante a abordagem proposta para aplicações de grandes redes neurais, que ocupem grandes áreas do *chip*. Ambas características também são percebidas pelos baixos valores de utilização de *Flip-Flops* (FF) e *Look-Up Tables* (LUTs) presentes no *chip*. Nota-se também, na Tabela 1, que a proposta possui erros máximo e médio maiores que outras propostas. No entanto, o valor de frequência de operação apresenta um resultado intermediário em relação aos trabalhos comparados.

CONCLUSÕES

Tendo em vista as relações da Tabela 1, conclui-se que a função implementada em FPGA, quando comparada a outros trabalhos sobre aplicações da função sigmoide nesse dispositivo portátil, possui erros máximos e médios maiores. No entanto, utiliza a menor área do FPGA e possui o menor atraso combinacional. Dessa forma, pode-se considerar os resultados satisfatórios, principalmente, para aplicação em redes neurais mais extensas que necessitam de grande quantidade de neurônios processados paralelamente.

Diante disso, abordagens futuras necessárias são introduzir a aproximação desenvolvida em aplicações para analisar o quão eficiente e viável se torna na prática a sua implementação, tendo em vista, os dados de desempenho do FPGA atrelado ao estudo de outras funções de ativação que podem ser embarcadas e capazes de produzir resultados com maior precisão e/ou ocuparem menor área do *chip*. além da possibilidade de aprimorar a função desenvolvida no presente trabalho, com um estudo mais aprofundado na precisão e na representação binária dos valores obtidos em *software* externo ao FPGA.

CONTRIBUIÇÕES DOS AUTORES

Todos os autores contribuíram com a revisão do trabalho e aprovaram a versão submetida.

AGRADECIMENTOS

Ao IFSP e ao CNPq pela bolsa de pesquisa na modalidade PIBITI edição 2023/2024.

REFERÊNCIAS

COSTA, C. D. Projetos de circuitos digitais com FPGA. **Érica, São Paulo**, 2013.

DALLY, W. J.; KECKLER, STEPHEN W.; KIRK, D. B. Evolution of the graphics processing unit (GPU). **IEEE Micro**, v. 41, n. 6, p. 42-51, 2021. DOI: 10.1109/MM.2021.3113475.

HAJDUK, Z. Hardware implementation of hyperbolic tangent and sigmoid activation functions. **Bulletin of the Polish Academy of Sciences. Technical Sciences**, v. 66, n. 5, p. 563-577, 2018. DOI: 10.24425/bpas.2018.124272.

HAUCK, S. The roles of FPGAs in reprogrammable systems. **Proceedings of the IEEE**, v. 86, n. 4, 1998. 615-638 p. DOI: 10.1109/5.663540.

HAYKIN, S. **Redes neurais: princípios e prática**. Bookman Editora, 2001.

HENNESSY, J. L.; PATTERSON, D. A. **Computer architecture: a quantitative approach**. Morgan kaufmann, 2017.

PAN, Z; et al. A modular approximation methodology for efficient fixed-point hardware implementation of the sigmoid function. **IEEE Transactions on Industrial Electronics**, v. 69, n. 10, 2022. p. 10694-10703. DOI: 10.1109/TIE.2022.3146573.

QIN, Z. et al. A novel approximation methodology and its efficient vlsi implementation for the sigmoid function. **IEEE Transactions on Circuits and Systems II: Express Briefs**, v. 67, n. 12, 2020. p. 3422-3426. DOI: 10.1109/TCSII.2020.2999458.

QIAN, B. et al., "Orchestrating the development lifecycle of machine learning-based IoT applications: A taxonomy and survey," **ACM Computing Surveys (CSUR)**, v. 53, n. 4, 2020. p. 1–47. DOI: 10.1145/3398020.

SOUSA, M. A. A.; DE JESUS TORRES, T. F. Modeling of pain on a FPGA-based Neural Network. **Parameters**, v. 1, n. 1, 2013. p. 4.8133. DOI: 10.2316/P.2013.793-034.

SOUSA, M. A. A. **Metodologias para desenvolvimento de mapas auto-organizáveis de Kohonen executados em FPGA**. Tese (Doutorado em Sistemas Eletrônicos) - Escola Politécnica, Universidade de São Paulo, 2018. DOI: 10.11606/T.3.2018.tde-06092018-091449.