Emulação de um acelerador em hardware para redes neurais utilizando a memória ReRAM

Luís F. Camponogara, Emanuel S. Maziero, Candice Müller, Fernando C. C. de Castro, Samuel T. Valduga, Natanael R. Gomes

***Resumo*— A implementação dos novos sistemas de comunicação móveis traz desafios para a estruturação da rede, buscando altas taxas de dados, baixa latência, eficiência espectral e densidade de dispositivos. As redes neurais podem otimizar os circuitos, porém esbarram na massiva quantidade de dados a ser processado em tempo real. Nesse trabalho, um acelerador de hardware foi emulado para acelerar a rede neural utilizando a memória ReRAM. Esse acelerador foi testado em uma rede neural de identificação de imagens, resultando em um aumento de 1,01% no número de imagens corretas com uma redução de 8*x* na complexidade computacional da rede.**

***Palavras-Chave*— Computação na memória, Memória ReRAM, Redes neurais, Identificação de imagens.**

***Abstract*— The implementation of new mobile communication systems brings challenges in network structuring, aiming for high data rates, low latency, spectral efficiency, and device density. Neural networks can optimize the circuits, but they face the massive amount of real-time data to be processed. In this study, a hardware accelerator was emulated to speed up the neural network using ReRAM memory. This accelerator was tested on an image identification neural network, resulting in a 1.01% increase in correctly identified images, with an 8*x* reduction in network computational complexity.**

***Keywords*— In-memory computing, ReRAM memory, Neural Network, Image identification.**

I. INTRODUÇÃO

A crescente demanda por elevadas taxas de transferência de dados, comunicação com baixa latência e comunicação massiva vem impulsionando o desenvolvimento de novas tecnologias, como o 5G e o 6G. As redes neurais se baseiam no funcionamento do cérebro humano, sendo compostas por um conjunto de neurônios artificiais conectados com o objetivo de realizar tarefas complexas. Elas podem ser divididas em dois grandes grupos [1], baseados no modelo de aprendizado da rede: o aprendizado supervisionado, entrada e saída da rede neural após uma fase de treinamento com dados conhecidos, enquanto que no aprendizado não supervisionado, não há conhecimento das saídas para as entradas impostas, ou seja, a função da rede é encontrar um padrão nos dados a fim de encontrar alguma informação oculta.

A aplicabilidade das redes neurais em telecomunicações é muito ampla, uma vez que, através delas, é possível encontrar padrões e correlacioná-los. Por exemplo, em [2], utilizando uma rede neural convolucional (CNN), foi proposto realizar o *beamforming* a fim de aumentar a capacidade do canal de comunicação em um sistema *Multiple Input Multiple Output* (MIMO) para o 5G. Outro exemplo de sua aplicabilidade está em [3], no qual foi utilizada uma rede neural recorrente (RNN) para prever o tráfego de aviões em um aeroporto. Além disso, em [4], foi proposta a utilização de uma rede neural profunda (DNN) para prever a qualidade de um canal de comunicação. Por fim, outro exemplo em que foram utilizadas técnicas de aprendizado de máquina em sistemas de telecomunicações está em [5], no qual essa técnica foi utilizada para realizar a estimação de canal.

Devido às grandes dimensões das redes neurais utilizadas nas tecnologias atuais e futuras, como o 5G e o 6G [5], a massiva quantidade de operações matemáticas requerida pelas redes neurais [2], [3], [4] e [5], em sistemas massivos MIMO, redes ultradensas e comunicação massiva entre máquinas, a eficiência computacional da rede e a capacidade de implementar o sistema proposto em tempo real ficam comprometidas. Desta forma, muitas soluções esbarram na elevada complexidade computacional da rede neural, inviabilizando sua aplicação em tempo real, especialmente na etapa de treinamento.

Uma abordagem que vem sendo estudada para tratar este problema é através do uso de aceleradores em hardware, onde as operações computacionais são realizadas diretamente na memória, chamada de *in-memory computing* (IMC). Esta abordagem permite reduzir o tempo de acesso à memória e o consumo de energia [6], [7] e [8].

Para que os conceitos de IMC sejam utilizados no acelerador em hardware da rede, é necessário a utilização de memórias não voláteis. Dentre os diversos modelos existentes, destaca-se a utilização das ReRAMs (*Resistive Random Access Memory*), nas quais os valores armazenados dentro dela são codificados em estados de resistência entre o valor mínimo LRS (*Low Resistive State*) e o valor máximo HRS (*High Resistive State*).

Neste artigo, foi proposto um acelerador em hardware utilizando a memória ReRAM em uma rede neural para classificação de imagens no software MATLAB e, posteriormente, sua emulação em FPGA através de seu equacionamento. Este artigo está organizado da seguinte forma: a Seção II fornece uma visão geral das redes neurais, do IMC e da ReRAM, sendo a integração desses conceitos expressa na Seção III. Na Seção IV, será demonstrada a implementação e emulação na *Field-Programmable Gate Array* (FPGA). Por fim, a Seção V exibe as conclusões obtidas.

L. F. Camponogara, E. S. Maziero, C. Müller, F. C. C. de Castro, S. T. Valduga e N. R. Gomes estão com o Grupo de Pesquisa em Processamento de Sinais e Comunicações da Universidade Federal de Santa Maria, Santa Maria - RS, e-mail: {luis.camponogara, emanuel.maziero }@acad.ufsm.br, {candice.muller, samuel.valduga, natanael-rodrigues.gomes}@ufsm.br, fccde-castro@outlook.com. Este trabalho foi parcialmente financiado pela Coordenação de Pessoal de Nível Superior (CAPES) (Projeto BRAFITEC 249/2019).

II. Aspectos teóricos

Para compreender a implementação do acelerador em hardware para redes neurais através da memória ReRAM, deve-se ter conhecimento dos conceitos separadamente. Na subseção A, será abordado as redes neurais rasas, IMC na subseção B, e a memória ReRAM na subseção C.

A. Redes neurais rasas

As redes neurais rasas, *Shallow neural networks*, são uma variação das redes neurais profundas. Estas redes tem por princípio possuir menos camadas ocultas entre a entrada e a saída da rede [9].

Uma rede neural profunda (DNN) é caracterizada por possuir múltiplos layers entre a entrada e a saída da rede . O conceito de redes rasas deriva do conceito das DNNs, de forma que é possível treinar uma rede que não possua ou possua poucos layers ocultos, no máximo dois. A ideia básica dessa rede está expressa na Figura 1.

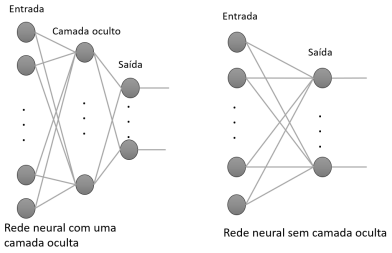

Fig. 1. Ideia base de uma rede neural.

Devido a quantidade de layers ocultos existentes em uma rede neural rasa, sua complexidade e tempo de treinamento da rede são pequenos, em relação a uma DNN. Além disso, outra vantagens deste tipo de rede é a possibilidade de utilizar menos dados de treino. Porém, essa rede tem não consegue convergir para a solução de problemas complexos, já que não consegue correlacionar os dados em poucas camadas ocultas.

B. Computing in-memory (IMC)

Computing in-memory ou como também é chamada de *in-memory computing* é uma arquitetura que busca aumentar o desempenho e a eficiência energética dos sistemas. Nessa arquitetura, é possível realizar o processamento dos dados diretamente na memória, sem a necessidade de transferir os dados para a CPU [10]. Esta área tem atraído diversos pesquisadores por conta de ter um grande impacto em aplicações que envolvem *machine learning* [6], [7] e processamento massivo de dados [10].

Para que a arquitetura IMC funcione, é necessário a utilização de memórias não voláteis que permitam realizar operações algébricas. Dentre as operações que são mais aplicadas são as multiplicações de matrizes por vetores (MVM) e multiplicações e acumulações (MAC).

Dentre as vantagens existentes em utilizar a arquitetura IMC, destaca-se a melhoria no desempenho do sistema e a economia de energia, já que não é necessária a transferência dos dados armazenados na memória. Além disto, não é necessário manter uma alimentação na memória para o dado continuar armazenado, já que é utilizado uma memória não volátil. Ademais, é importante a flexibilidade existente nesta arquitetura, já que pode-se utilizar ela para performar diferentes modelos de redes neurais e configurações [6], [7].

A principal desvantagem existente atualmente é o custo de implementação deste tipo de memória fisicamente, já que a fabricação destes dispositivos é reduzida a pequenas quantidades de teste, e não há massiva comercialização. Além disto, outra grande desvantagem é o difícil controle no processo de escrita do dado na memória. Por fim, cita-se a dificuldade da massiva integração deste sistema de processamento com as CPUs e GPUs existentes, e circuitos combinacionais.

C. Resistive Random Access Memory (ReRAM)

A memória ReRAM é um dos tipos de memórias não-voláteis. Nesse tipo de memória, após um dado ser armazenado internamente, ele estará disponível para uso, sem a necessidade de ser alimentado durante o intervalo em que foi gravado e sua utilização. Esta característica permite a redução do consumo de energia e do tempo de acesso a determinado dado na memória, bem como aumenta a eficiência computacional e suporta um grande paralelismo computacional, já que as operações são realizadas diretamente na memória [6], [7].

Para constituir um chip da memória ReRAM é necessário a utilização de Memristores, dispositivo de característica não volátil. Ele permite armazenar um valor resistivo, entre o LRS (*Low Resistive State*) e o HRS (*High Resistive State*), baseado no fluxo de corrente elétrica que circula pelo dispositivo. Através da aplicação de uma tensão $V(t)$, é possível gerar uma corrente $i(t)$, através da multiplicação da entrada pela condutância do dispositivo, Equação (1).

$$i(t) = V(t) \cdot W \tag{1}$$

Entretanto, operar com esta memória tem algumas dificuldades. A primeira delas é a necessidade da utilização de conversores analógicos-digitais (ADC) e digitais-analógicos (DAC), já que a memória opera em analógico. Os ADCs e DACs representam uma grande parte do circuito. Ademais, o controle do exato valor armazenado nos memristores pode ser um desafio: através de uma variação inesperada no fluxo de corrente elétrica, o dispositivo pode sofrer eventuais distúrbios na operação de leitura ou codificação incorreta no processo de escrita, gerando um erro no resultado.

O modelo utilizado para a demonstração da ReRAM é o *Voltage ThrEshold Adaptive Memristor* (VTEAM) [11], no qual existem duas tensões de *threshold* para sua operação. Quando a tensão aplicada é igual ou superior a tensão V_{off}, o valor armazenado no memristor se torna o HRS, se a tensão aplicada for menor ou igual a tensão V_{on}, o valor armazenado no memristor se torna o LRS, e se a tensão for entre os *thresholds*, não ocorre alteração no valor armazenado.

Para que os pesos e os bias sejam armazenados e performados nas redes neurais, foi utilizado a arquitetura da Figura 2. Nesse caso, os parâmetros da rede neural são armazenados e operados em complemento de 2. Por conta disso, considerando *n* como o número de bits que vai ser utilizado para representar

um valor x, seja positivo ou negativo, é igual ao número de memristores. Apesar desta arquitetura não ser a mais otimizada em relação ao número de memristores, a robustez do circuito quando é aplicado múltiplos pulsos de leitura na entrada é maior, bem como um controle facilitado no processo de escrita.

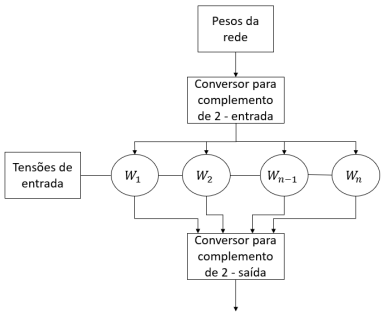

Fig. 2. Pesos armazenados e operados em complemento de 2 utilizando n bits.

Dentre os aceleradores de redes neurais existentes na bibliografia, destaca-se: [6] e [12] . Em [6], foi proposto uma arquitetura que armazenava e operava os pesos em complemento de 2, porém é utilizado múltiplos ciclos para a identificação de uma imagem. Enquanto que, em [12], os pesos são armazenados e operados em analógico, ou seja, é realizado uma conversão direta do valor a ser armazenado para um estado de resistência, sendo a imagem processada em um único ciclo, entretanto quando esse método é realizado, a robustez do circuito fica comprometida devido aos efeitos *IR-drop* que a memória esta sujeita de acordo com [13].

III. Operação da rede neural proposta

O acelerador de redes neurais proposto, Figura 3, é baseado no conceito de IMC nas memórias ReRAM. Esse acelerador utiliza quantizadores para converter os parâmetros da rede para 4 bits em complemento de 2, buscando obter um balanço entre o número de memristores e a robustez do circuito. Além disso, outro quantizador para substituir os ADCs após as matrizes de memristores. Por fim, a imagem será processada em um único ciclo.

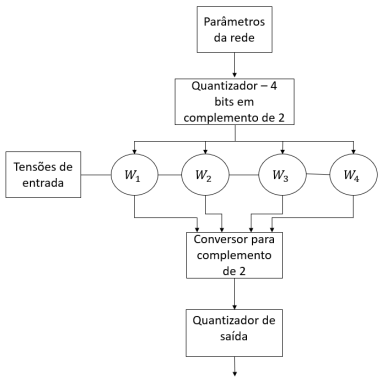

Fig. 3. Acelerador para redes neurais implementado.

Como forma de validar o acelerador proposto, foi implementado uma rede neural rasa para a classificação de imagens, em especial para datasets pequenos. Isso se deve ao fato de essas redes possuírem um baixo número de sinapses e fácil treinamento. Um exemplo de dataset pequeno é o *Digit Dataset* [14], imagens de dígitos escritos a mão com imagens de 8x8 pixels, Figura 4. Entretanto, o acelerador pode ser utilizado para problemas mais complexos, a exemplo de sistemas operando em tempo real e 5G, conforme abordado em [2], [3], [4], [5], [6] e [7].

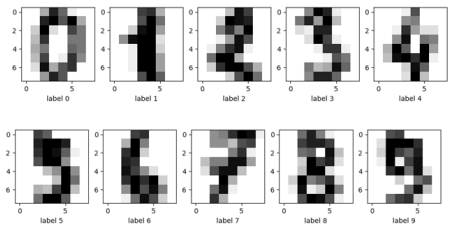

Fig. 4. Seleção de algumas imagens do Digit dataset.

O *Digit Dataset* possui 1797 imagens, sendo definido 1500 imagens para o treino e 297 para a validação. Para a rede neural, não foi utilizado layer oculto, para que a FPGA Zybo 7010 consiga operar a rede neural.

IV. Resultados de simulação e discussões

A Seção de resultados de simulação e discussões foi dividida em duas subseções. A primeira subseção é relativa ao treino e validação inicial da rede neural rasa. Em sequência disto, é abordado a emulação da memória ReRAM.

A. Treino e validação da rede neural rasa

Utilizando as bibliotecas *tensorflow* e *keras*, foi treinada a rede neural rasa para a identificação de imagens, sem a utilização de camadas ocultas. Para o treinamento da rede, foram utilizados os parâmetros da Tabela I.

TABELA I
PARÂMETROS UTILIZADOS NO TREINAMENTO DA REDE.

Batch size	16
Número de épocas de treinamento	600
Função de ativação	Softmax
Otimizador	sgd
Métrica	*accuracy*

Na Tabela I, o *batch size* é o tamanho máximo do lote que é executado a cada vez na rede. A função de ativação *softmax* retorna probabilidades, usualmente utilizada em classificação multi-classe, e o otimizador, *sgd*, é o gradiente descendente estocástico, [15] . Por fim, métrica por *accuracy* é a maneira de avaliar o desempenho da rede de classificação de imagens, porcentagem de imagens identificadas corretamente, de forma a buscar o maior valor de previsões corretas.

Durante a etapa de treinamento da rede, foi alcançada uma precisão de 98,13%, identificando corretamente 1472 das 1500 imagens testadas. Já durante a etapa de teste, houve uma queda na precisão da rede para 90,91%, com 270 imagens corretamente identificadas em relação às 297 imagens testadas.

Analisando a perda do treino e de validação ao decorrer das épocas de treinamento da rede neural, Figura 5, observa-se que tanto a perda de treino, *train loss*, quando a perda de validação, *validation loss*, estão de acordo com o que é esperado. Essas perdas devem diminuir conforme o número de épocas de treinamento da rede aumenta.

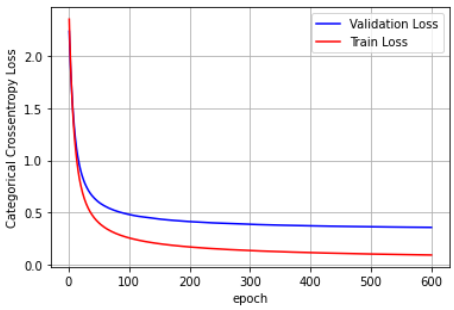

Fig. 5. Perda de treino e validação da rede neural.

B. Emulação da memória ReRAM

Para emular a operação das memórias ReRAM no software MATLAB e, posteriormente, na FPGA, deve-se definir quais serão os valores extremos de resistência, HRS e LRS, que o memristor irá utilizar. Neste projeto, os valores utilizados são $100K\Omega$ e $1K\Omega$ respectivamente, sendo definidos através de um balanço entre: a influência do HRS no cálculo, o consumo de energia e o tempo necessário para a leitura e escrita dos dados na memória.

Após o treino da rede neural, os pesos e bias da rede neural são números reais com o padrão *float*, o qual utiliza 32 bits para representar o número. Para que eles possam ser representados por 4 bits, é necessário realizar o processo de quantização destes valores. Uma das diversas maneiras para realizar a quantização é utilizar a Equação 2, já que ela permite eliminar os valores extremamente negativos e positivos, baseados em uma precisão de quantização b e o fator de escala sf [16].

$$Q = min\left(max\left(round\left(\frac{input}{sf}\right), -2^{b-1}\right), 2^{b-1} - 1\right), \tag{2}$$

onde, a função *round* deve realizar o arredondamento simples do dado de entrada pelo sf; a função *max* é utilizada para eliminar valores extremamente negativos, baseado na comparação do maior valor entre o resultado da operação *round* e o menor valor em complemento de 2 que pode ser representado com b; por fim, a função *min* é utilizada para eliminar valores extremamente positivos, baseado na comparação do menor valor entre o resultado da operação anterior e o maior valor em complemento de 2 que pode ser representado com b.

Na tentativa de realizar o processo de quantização nos pesos e no bias no software MATLAB da maneira com que fosse reduzido o mínimo possível o número de imagens identificadas corretamente pela rede, foi utilizado um processo estocástico na definição do valor de sf. Esse processo é baseado na variação do valor de sf, que inicia em 0 com passo de adaptação de 0,0001. Durante a etapa de processamento dos dados, foi plotado um gráfico que relacionava o valor do sf com a *accuracy* da rede neural, Figura 6. Após o término do processamento, foi definido o sf como 0,3425, e a rede conseguiu identificar 273 imagens corretas das 297 imagens testadas, 91,92%, demonstrando um aumento de *accuracy* da rede em 1,01%. Utilizando estes novos valores para os pesos, foi implementado na FPGA o equacionamento da memória ReRAM e verificado com um hardware externo o processo de emulação.

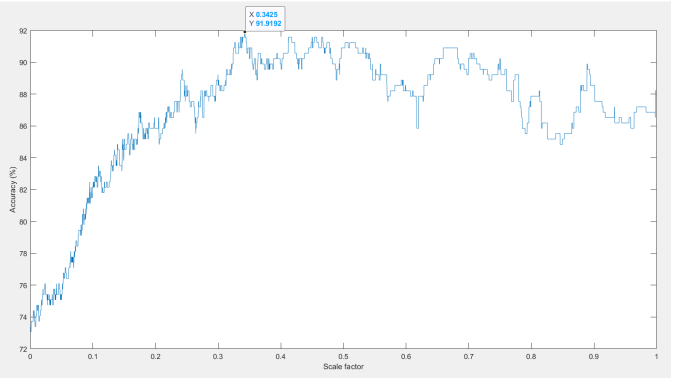

Fig. 6. Gráfico do *sf X accuracy*.

Para embarcar a rede neural na FPGA Zybo 7010, foi utilizado a linguagem de descrição de hardware VHDL (*VHSIC Hardware Description Language*). Dentro da memória interna da FPGA foi inserido os pesos da rede, os bias e as imagens a serem performadas. Na configuração da placa, definiu-se que a operação iniciaria após o switch 1, *SW0*, ser ativado e o reset no sistema após o botão 0, *BTN0*, ser pressionado. Como forma de controlar a saída da rede na FPGA, foi implementado um contador binário de 9 bits, o qual permite representar números entre 0 e 511, para exibir o número de imagens identificadas corretamente.

Quando o *SW0* é ativado, a placa executa internamente as operações relativas a execução da rede neural com a emulação da ReRAM e o resultado obtido está apresentado na Figura 7.

O jumper roxo utilizado no led mais da direita, da Figura 7 representa o bit mais significativo e o led mais da esquerda representa o bit menos significativo. Neste caso, o número binário que a FPGA exibe no contador binário desenvolvido através de leds é 100010001_2, sendo o número decimal correspondente é 273 .

Analisando os resultados obtidos da emulação da memória ReRAM, observa-se que para um dataset pequeno como o *Digit dataset*, a rede demonstrou uma melhora quando os pesos passaram pelo processo de quantização. Apesar das CNNs obterem resultados mais precisos que as MLPs em identificação de imagens por conta da correlação espacial entre os pixels adjacentes em uma imagem, a rede treinada demonstrou resultados satisfatórios em termos de *accuracy* em simulação e emulação da rede.

Apesar de na Figura 5, a rede não demonstrar ter sofrido o processo de *overfitting*, que ocorre quando a perda de validação aumenta apesar da perda de treino estar diminuindo

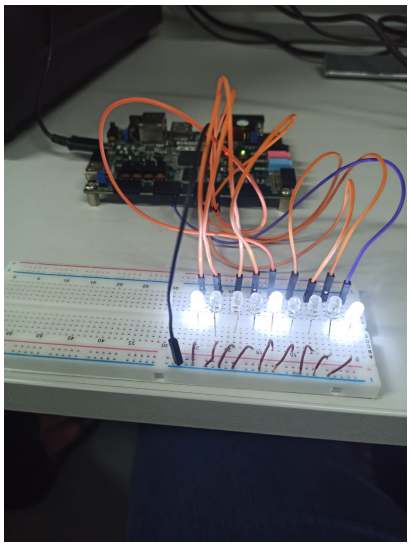

Fig. 7. Emulação da memória ReRAM na FPGA.

com o passar das épocas, foi verificado que através do processo de quantização dos pesos e bias a precisão aumentou. Além disto, esse processo também permitiu e emulação memória ReRAM utilizando a codificação em complemento de 2 com 4 bits, reduzindo o poder computacional do hardware necessário para executar a rede, aumento a eficiência computacional e energética.

V. Conclusão

Para problemas simples, como a classificação de imagens de dígitos escritos a mão, a rede neural rasa demonstrou bons resultados, de forma que conseguiu identificar corretamente 90,91% das imagens testadas durante a validação. Esse resultado foi melhorado, ou seja, mais imagens foram identificadas corretamente, quando o acelerador emulado foi inserido na operação da rede, aumentando em 1,01% o número de imagens identificadas corretamente. Além disto, através do processo de quantização aplicado, que converte os dados da saída do treinamento da rede neural de 32 bits para a representação utilizando 4 bits em complemento de 2, observa-se uma redução em 8x da complexidade computacional da rede.

A utilização de um acelerador em hardware, utilizando a memória ReRAM, apresenta vantagens significativas na implementação de redes neurais. Os memristores, com seus dois estados de resistência (LRS e HRS), permitem uma operação confiável em sistemas onde a confiabilidade é crucial. Isso permite sua aplicação em dados sensíveis ou críticos, garantindo a integridade e a segurança dos resultados obtidos.

Além disso, o acelerador emulado proporciona um aumento notável na eficiência computacional. Ao processar os dados diretamente na memória, há uma redução significativa na latência do sistema. Isso é particularmente vantajoso em redes neurais que operam em tempo real, como em sistemas 5G, onde o processamento rápido e eficiente dos dados é essencial.

Em resumo, a utilização de um acelerador em hardware, juntamente com a memória ReRAM, oferece benefícios notáveis. A combinação desses elementos permite um processamento mais eficiente, com menor latência e maior confiabilidade, impulsionando o desempenho das redes neurais e abrindo caminho para aplicações avançadas em diversos setores.

Referências

[1] L. Kovács and G. Z. Terstyánszky, "Diagnosising faults by supervised and unsupervised learning,"1999 European Control Conference (ECC), Karlsruhe, Germany, 1999, pp. 4238-4242, doi: 10.23919/ECC.1999.7099999.

[2] K. Aljohani, I. Elshafiey and A. Al-Sanie, "Implementation of Deep Learning in Beamforming for 5G MIMO Systems,"2022 39th National Radio Science Conference (NRSC), Cairo, Egypt, 2022, pp. 188-195, doi: 10.1109/NRSC57219.2022.9971327.

[3] S. Ji, H. Yang, L. Gong, Z. Li, M. Kadoch and M. Cheriet, "Airport network traffic prediction in 5G scenarios: a deep learning approach,"2020 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), Paris, France, 2020, pp. 1-5, doi: 10.1109/BMSB49480.2020.9379722.

[4] N. Diouf, M. Ndong, D. Diop, K. Talla, M. Sarr and A. C. Beye, "Channel Quality Prediction in 5G LTE Small Cell Mobile Network Using Deep Learning,"2022 9th International Conference on Soft Computing & Machine Intelligence (ISCMI), Toronto, ON, Canada, 2022, pp. 15-20, doi: 10.1109/ISCMI56532.2022.10068487.

[5] Y. Zhang, M. Alrabeiah and A. Alkhateeb, "Deep Learning for Massive MIMO With 1-Bit ADCs: When More Antennas Need Fewer Pilots,"in IEEE Wireless Communications Letters, vol. 9, no. 8, pp. 1273-1277, Aug. 2020, doi: 10.1109/LWC.2020.2987893.

[6] Xue, CX., Chiu, YC., Liu, TW. et al. A CMOS-integrated compute-in-memory macro based on resistive random-access memory for AI edge devices. Nat Electron 4, 81–90 (2021). https://doi.org/10.1038/s41928-020-00505-5

[7] Wan, W., Kubendran, R., Schaefer, C. et al. A compute-in-memory chip based on resistive random-access memory. Nature 608, 504–512 (2022). https://doi.org/10.1038/s41586-022-04992-8

[8] Q. Qin et al., "Hybrid Precoding with a Fully-Parallel Large-Scale Analog RRAM Array for 5G/6G MIMO Communication System,"2022 International Electron Devices Meeting (IEDM), San Francisco, CA, USA, 2022, pp. 33.2.1-33.2.4, doi: 10.1109/IEDM45625.2022.10019426.

[9] O. Gorokhovatskyi and O. Peredrii, "Shallow Convolutional Neural Networks for Pattern Recognition Problems,"2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP), Lviv, Ukraine, 2018, pp. 459-463, doi: 10.1109/DSMP.2018.8478540.

[10] Huang, Dan, et al. "Identifying challenges and opportunities of in-memory computing on large HPC systems."Journal of Parallel and Distributed Computing 164 (2022): 106-122.

[11] S. Kvatinsky, M. Ramadan, E. G. Friedman, and A. Kolodny, “Vteam: A general model for voltage-controlled memristors,” IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 62, no. 8, pp. 786–790, 2015.

[12] A. Azamat, F. Asim and J. Lee, "Quarry: Quantization-based ADC Reduction for ReRAM-based Deep Neural Network Accelerators,"2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD), Munich, Germany, 2021, pp. 1-7, doi: 10.1109/ICCAD51958.2021.9643502.

[13] Z. He, J. Lin, R. Ewetz, J. -S. Yuan and D. Fan, "Noise Injection Adaption: End-to-End ReRAM Crossbar Non-ideal Effect Adaption for Neural Network Mapping,"2019 56th ACM/IEEE Design Automation Conference (DAC), Las Vegas, NV, USA, 2019, pp. 1-6.

[14] S. Learn, “The digit dataset.” https://scikit-learn.org/stable/auto_ examples/datasets/plot_digits_last_image.html. Acesso em: 24-04-2023.

[15] Z. Chen and J. Cheng, "A Parallel Softmax Classification Algorithm Based on MapReduce,"2018 13th International Conference on Computer Science & Education (ICCSE), Colombo, Sri Lanka, 2018, pp. 1-5, doi: 10.1109/ICCSE.2018.8468863.

[16] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha, “Learned step size quantization,” in International Conference on Learning Representations, 2020.