https://doi.org/10.1007/s11265-024-01911-0

Check for updates

Análise de quantização de ponto flutuante de redes neurais artificiais perceptron multicamadas

Hussein Al-Rikabi[1] · Balázs Renczes[1]

Recebido: 7 de setembro de 2023 / Revisado: 30 de janeiro de 2024 / Aceito: 18 de fevereiro de 2024 / Publicado online: 25 de março de 2024

Resumo
O impacto da quantização em redes neurais artificiais (ANNs) Multi-Layer Perceptron (MLP) é apresentado neste artigo. Nesta arquitetura, o aumento constante do tamanho e a exigência de diminuir a precisão dos bits são dois fatores que contribuem para o aumento significativo dos erros de quantização. Apresentamos uma ferramenta analítica que modela a propagação da potência do ruído de quantização (QNP) em ANNs MLP de ponto flutuante. Ao contrário da abordagem de última geração, que compara os dados exatos e quantizados experimentalmente, o algoritmo proposto pode prever o QNP teoricamente quando o efeito da quantização da operação e o erro de quantização do coeficiente (CQE) são considerados. Isso auxilia nas decisões para determinar a precisão necessária durante o projeto de hardware. O algoritmo é flexível no tratamento de ANNs MLP de parâmetros definidos pelo utilizador, tais como o tamanho e o tipo de função de ativação. Além disso, é construído um ambiente de simulação que pode realizar cada operação com uma precisão de bits ajustável. A precisão do cálculo da QNP é verificada com dois conjuntos de dados de referência disponíveis publicamente, utilizando o ambiente de simulação de precisão padrão como referência.

Palavras-chave Precisão numérica · Redes Neurais Artificiais · Ruído de quantização · Quantização de coeficientes · Aritmética de ponto flutuante

1 Introdução

Nas últimas duas décadas, as redes neurais artificiais (ANNs) têm desempenhado um papel cada vez mais importante na resolução de tarefas de processamento de sinais digitais. São modelos de dados estatísticos não lineares que replicam o papel das redes neurais biológicas [1]. Como as ANN apresentam um desempenho consideravelmente elevado, além dos métodos convencionais baseados em modelos, tornaram-se candidatas óbvias para resolver problemas, entre outras aplicações, na área do processamento de imagens [1, 2], regressão [3–5], classificação [6] e previsão de séries temporais [7–9].

Durante a etapa de projeto de hardware das ANNs, os principais critérios de otimização aplicados são precisão, tamanho e velocidade. O número de bits que representam cada sinal e cada operação

A quantização nas ANNs afeta criticamente essas propriedades. Desde a introdução da representação numérica de precisão dupla em ponto flutuante usando 64 bits (FP64) [10], até recentemente, o efeito do erro de quantização era negligenciado na maioria dos casos. No entanto, a redução da precisão dos bits tornou-se cada vez mais importante no design de hardware para obter soluções baratas e/ou rápidas [11]. As plataformas de hardware recentes têm como objetivo usar aritmética de precisão reduzida, simplificando o caminho dos dados e reduzindo os requisitos de armazenamento de memória, largura de banda e energia [9]. Paralelamente a essa tendência, o tamanho das ANNs está a crescer continuamente para atingir altas taxas de aprendizagem. As arquiteturas de rede consistem em centenas ou mesmo milhares de neurónios dispostos em várias camadas [12].

Tanto a redução da precisão quanto o aumento do tamanho contribuem para o fenômeno de que o efeito da quantização

seja mais significativo. Isso torna a análise
do ruído de quantização, uma questão de ponta, o que motivou este artigo. Em particular, focamos na avaliação de ponto flutuante (FP), considerando o seguinte. Em comparação com a representação numérica de ponto fixo, o FP é frequentemente vantajoso devido à ampla gama que pode cobrir. Além disso, à medida que os cálculos FP se tornam mais rápidos e menos dispendiosos de implementar, a tendência para o uso de números FP

✉ Hussein Al-Rikabi rhussein@mit.bme.hu

Balázs Renczes
renczes@mit.bme.hu

[1] Departamento de Sistemas de Medição e Informação, Universidade de Tecnologia e Economia de Budapeste, Mu˝egyetem rkp. 3, Budapeste H-1111, Hungria

está a acelerar [3]. No entanto, a análise do ruído de quantização não é simples, como será demonstrado nas secções seguintes.

Existem poucos estudos na literatura que mostram o efeito do erro de quantização. Ele foi investigado extensivamente para sistemas como a transformada rápida de Fourier em [13], os algoritmos de ajuste de seno para testes de conversores analógico-digitais em [14] e os transmissores multicarrier com banco de filtros em [15]. No entanto, existe uma lacuna óbvia na área das redes neurais artificiais de perceptron multicamadas (MLP). Neste estudo, as redes neurais artificiais MLP serão investigadas em detalhe a partir deste ponto de vista.

Este artigo trata de dois tipos de erros: o ruído de potência de quantização (QNP) e o erro de quantização do coeficiente (CQE). O QNP é gerado devido à quantização de cada operação. Por outro lado, o CQE ocorre ao quantizar as constantes. Ambos os tipos serão descritos e analisados detalhadamente nas Secções 3, 4 e 5. As principais contribuições deste estudo podem ser resumidas da seguinte forma:

1. É apresentada uma análise completa da quantização FP das ANNs MLP.
2. É codificada uma ferramenta genérica capaz de estimar o erro de quantização antes da fase de implementação do hardware com parâmetros definidos pelo utilizador.
3. Calcular e compensar com precisão o CQE para obter uma estimativa precisa do QNP.
4. O procedimento apresentado para avaliar o QNP e o CQE pode ser usado para outros sistemas não lineares.

A fim de fornecer uma visão geral concisa da novidade deste trabalho, oferecemos uma análise teórica aprofundada que investiga cada operação dentro da rede, com foco específico na geração e propagação do erro de quantização. A literatura existente concentra-se principalmente na avaliação experimental de diferentes esquemas de quantização para várias arquiteturas de redes neurais. Alguns desses esquemas empregam o uso mínimo de bits, como formatos binários ou ternários [16, 17], enquanto outros utilizam formatos inteiros [18]. No entanto, é importante observar que, até onde sabemos, a análise e os métodos propostos ainda não foram explorados. Em outras palavras, nenhum dos métodos anteriores investigou teoricamente o fenómeno da quantização em uma ampla gama de resoluções FP.

É importante destacar que o algoritmo proposto produz um limite teórico para os erros de quantização, ao contrário da abordagem experimental das soluções de última geração, conforme destacado a seguir.

Recentemente, tem havido um interesse crescente no desempenho das ANNs em relação à resolução de bits aplicada. Shah et al. [12] projetaram uma ANN MLP para aplicações de reconhecimento de fala usando representação de ponto fixo de cinco e seis bits. Após treinar os modelos usando representação FP

Para a implementação, eles utilizaram um esquema de quantização pós-treinamento (PTQ). Eles observaram experimentalmente a sensibilidade do peso e a faixa de saída da operação para decidir a resolução de cada peso e operação. Kristian et al. [19] implementaram uma ANN MLP em Field Programmable Gate Array (FPGA) utilizando a representação de ponto flutuante de precisão simples de 32 bits (FP32). O modelo implementado foi comparado a uma implementação de ponto fixo de 16 bits em termos de erro de quantização. Concluiu-se que a implementação de ponto fixo era treze vezes menor em tamanho lógico do que a implementação FP32. Peric et al. [20] apresentaram uma análise CQE comparando as representações numéricas FP32 e de ponto fixo de 32 bits para os pesos das implementações da MLP ANN. Foi demonstrado que ambas as representações numéricas produzem o mesmo erro de quantização. Huang et al. [5] utilizaram a representação de ponto fixo de 32 bits para implementar uma MLP ANN em FPGA para uma aplicação de tomografia. Eles observaram a partir de suas experiências que a implementação de ponto fixo afeta ligeiramente o desempenho do seu sistema.

Existem também estudos que melhoram as ANNs quantizadas, realizando a quantização antes e durante o treino [21]. No nosso estudo, realizamos a análise após treinar a rede com resolução total; ou seja, enfrentamos o PTQ.

Uma análise de erro de quantização para uma pequena ANN MLP contendo três neurónios com Funções de Ativação (AFs) TanSig em uma única camada foi apresentada em [22]. Apenas a representação FP32 foi considerada nesse estudo. Concluiu-se que a representação FP32 é válida para a aplicação apresentada, mas não é uma conclusão geral. A ideia de generalizar esta abordagem e criar uma ferramenta genérica para tamanhos e precisões arbitrárias de ANNs MLP com diferentes AFs motivou-nos para este estudo.

É essencial mencionar que estamos a analisar redes neurais MLP neste artigo. Embora as ideias apresentadas possam ser aplicadas a outras estruturas de rede, como redes neurais convolucionais profundas e redes neurais recorrentes, essa extensão excede os limites deste artigo e é um tópico para investigação futura.

O restante deste artigo está organizado da seguinte forma. A Secção 2 apresenta uma breve introdução teórica sobre as redes neurais artificiais MLP e o ruído de quantização. A Secção 3 apresenta um ambiente de simulação para que o erro de quantização seja simulado passo a passo ao longo da avaliação dos cálculos da rede com precisão FP limitada. A Secção 4 explica o método proposto para calcular teoricamente o QNP aproximado. A análise CQE é apresentada na Secção 5. A Secção 6 apresenta a verificação do método através da comparação do erro de quantização simulado e do ruído de quantização previsto teoricamente. Isto é realizado em dois estudos de caso: um modelo de previsão de regressão de uma estação meteorológica e um modelo de classificação no diagnóstico de cancro. Finalmente, a Secção 7 conclui o artigo.

2 Fundamentos teóricos

Esta secção apresenta uma breve visão geral das ANNs MLP e do teorema de quantização necessário para caracterizar o QNP das redes investigadas.

2.1 Redes Neurais Artificiais Perceptron Multicamadas (MLP)

Uma arquitetura ANN MLP consiste em uma camada de entrada, uma camada de saída e camadas ocultas, que estão totalmente conectadas. Cada camada contém neurônios inspirados na estrutura do cérebro humano [12]. Um neurônio é o objeto central da arquitetura que pondera as entradas recebidas de um processo determinístico anterior
e soma os produtos ponderados resultantes. A soma de

Os produtos são polarizados e passados para um AF como um argumento para fornecer a saída do neurónio [23]. O AF é uma parte crítica da ANN devido ao seu enorme impacto na operação de treino [24]. O neurónio pode ser melhor descrito pela seguinte equação:

$$y = F\left(b + \sum_{j=1}^{m} w_j \cdot x_j\right), \quad (1)$$

onde F denota a AF, b é o viés, x_j é a entrada do neurónio, w_j é o peso correspondente e m é o número de entradas [25].

2.2 Erro de quantização

A quantização é o processo de aproximar um sinal contínuo por um conjunto de símbolos discretos ou valores inteiros. As grandezas físicas têm precisão infinita, mas quando digitalizadas e inseridas num computador ou num sistema digital, são arredondadas para o valor representável mais próximo. Este processo introduz um erro de quantização. O valor representável mais próximo é definido pelo tipo de representação numérica. A representação FP tem sido comumente usada há décadas. Ela foi emitida pela norma IEEE-754 [10]. Uma representação FP define um valor com um vetor de bits contendo três partes: sinal, expoente e mantissa. A resolução do número FP é definida pela precisão, que é o número de bits da mantissa, enquanto o expoente define o intervalo [26].

O arredondamento da aritmética FP é considerado uma operação que segue a operação infinitamente precisa. O erro de quantização pode ser descrito com a seguinte diferença [13]:

$$v = Q(y) - y, \quad (2)$$

onde v denota o erro de quantização e $Q(.)$ é a função de quantização, portanto, $Q(y)$ é o valor quantizado da saída da operação. Como mostra a equação, a quantização é uma determinístico anterior. No entanto, após a operação, o valor original não pode mais ser recuperado. Consequentemente, o erro de quantização exato também permanece desconhecido.

Para caracterizar o processo, [13] sugeriu um modelo que descreve o erro de quantização como uma fonte de ruído aditivo. Esta fonte adiciona um ruído de quantização uniformemente distribuído ao valor exato. A largura da distribuição uniforme é determinada pelo quantizador. Para a representação FP, o nível de ruído depende do QNP correspondente, que pode ser aproximado da seguinte forma [13]:

$$E\{v_{FP}^2\} = 0{,}18 \cdot 2^{-2p} \cdot E\{y^2\}, \quad (3)$$

onde $E\{y^2\}$ é o valor esperado da camada quanti-

$_{FP}$ ^{2}

erro de quantização, ou seja, o QNP do quantizador FP, $E\{y\}$ é a potência do sinal na entrada do quantizador e p é a precisão, que são os bits da mantissa. Por exemplo, p 24 para o formato de precisão simples. Segue-se que, tendo um bit de mantissa oculto na representação, o significando é de 23 bits
bit [10].

Esta aproximação torna-se útil, uma vez que é uma função contínua do nível de entrada, ao contrário do valor exato, que é uma função escalonada. Durante a análise, nós,
portanto, aplique esta aproximação. Uma comparação entre as características aproximadas e exatas é mostrada na Fig. 1. Como o ruído de quantização é modelado como uma variável aleatória, realizaremos uma análise estatística para determinar o QNP. Com base neste resultado, serão apresentados limites de erro para caracterizar a imprecisão da saída.

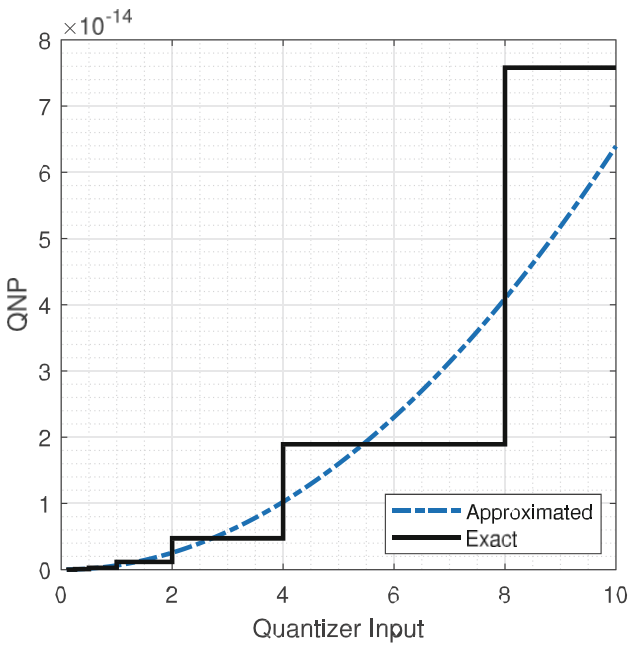

Figura 1 Comparação entre as características aproximadas e exatas do QNP na representação FP com $p = 24$.

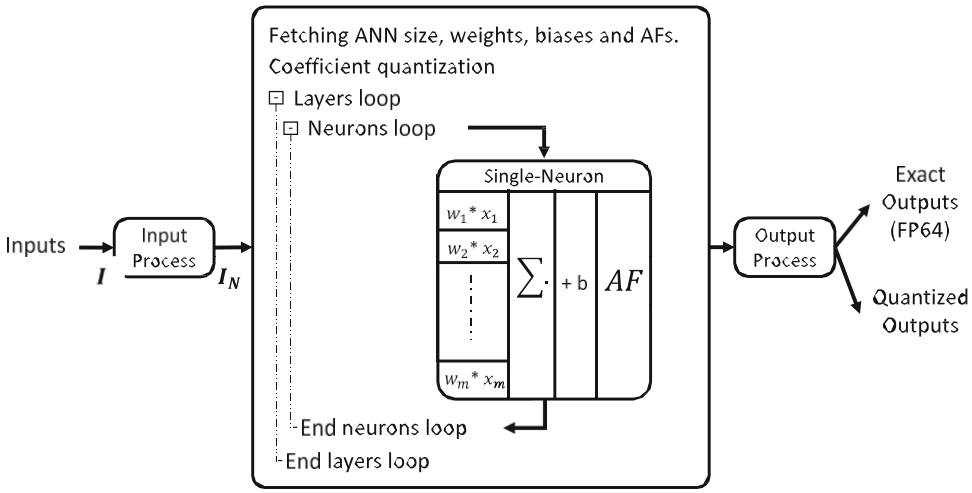

Figura 2 Diagrama de blocos da simulação do método proposto.

É importante mencionar aqui que não é possível alterar a precisão de uma representação numérica FP usando a caixa de ferramentas integrada do MATLAB, por isso usamos a caixa de ferramentas fornecida em [13] para a quantização FP com precisão variável. A resolução do quantizador é basicamente definida pela precisão, enquanto o expoente não afeta o erro de quantização no caso de não haver estouro; portanto, o expoente define apenas o intervalo. A direção de arredondamento utilizada neste trabalho é o método padrão de arredondamento para o mais próximo. Além do método proposto ser a configuração padrão para a maioria das aplicações práticas, a análise de outras técnicas de arredondamento, como arredondamento para o infinito ou para o zero, também exigiria novas aproximações de quantização, além da (3). Isso excede os limites deste estudo.

3 Cálculo do ruído de quantização em ANNs MLP

Após uma breve visão geral teórica, o QNP em MLP ANNs é simulado nesta secção. O nosso objetivo é fornecer uma solução flexível que possa lidar com modelos MLP com diferentes tamanhos, precisões e AFs. Conforme descrito na Seção 2, esse valor exato não está disponível em sistemas quantizados devido à natureza irreversível da quantização. No entanto, durante as simulações, a aritmética FP64 pode ser aplicada como referência, e podemos acompanhar a propagação do erro no sistema comparando os resultados FP64 (exatos) e de precisão limitada.

O diagrama de blocos do algoritmo proposto é apresentado na Fig. 2. É possível observar que três etapas principais devem ser realizadas, que são o processo de entrada, o processamento das camadas e o processo de saída. O processo de entrada é uma etapa de normalização que mapeia o conjunto de dados de entrada I para valores normalizados I_N no intervalo $[-1; +1]$ da seguinte forma:

$$I_N = (I - I_{\min}) \cdot \frac{2}{I_{\max} - I_{\min}} - 1, \quad (4)$$

onde $I_{\max}$ e $I_{\min}$ representam o intervalo do conjunto de dados de treino de entrada I.

O processo de saída tem as mesmas operações na ordem inversa, que desnormaliza os dados de volta ao intervalo original. Desta forma, os parâmetros e operações entre os blocos do processo de entrada/saída não estão restritos ao intervalo de normalização. No entanto, após a normalização, os valores podem aumentar dependendo dos parâmetros, mas não aumentarão massivamente, pois os valores de entrada são normalizados.

Entre esses processos, os valores das saídas exatas e quantizadas dos neurónios são calculados e salvos na camada seguinte. Dessa forma, obtemos as saídas quantizadas e exatas finais da rede, e a diferença entre elas é o erro de quantização. O quadrado desse erro é o QNP simulado a ser comparado com a aproximação teórica apresentada na Seção 4.

A Figura 3 mostra como cada operação no bloco de neurónio único é quantizada, onde os blocos denotados por Q representam a função de quantização descrita em (2). Da mesma forma, todas as outras operações no modelo são quantizadas. Desta forma, o algoritmo pode ser usado genericamente para simular e analisar redes MLP com um tamanho arbitrário. O erro de quantização simulado de um neurónio é calculado realizando as seguintes ações.

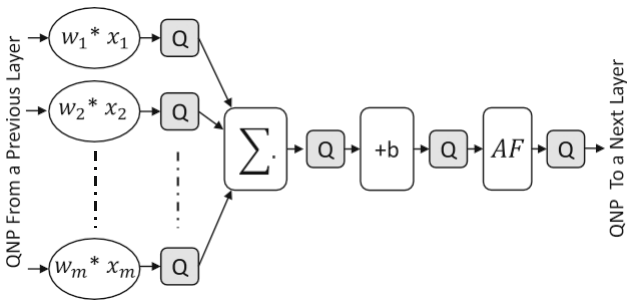

Figura 3 Diagrama de bloco de neurónio único mostrando o processo de quantização após cada operação.

Tabela 1 Lista de AFs incluídas neste estudo e o número correspondente de quantizadores.

Função de ativação	Explicação da abreviatura	Número de quantizadores
TanSig	Tangente hiperbólica sigmoidal	4
LogSig	Sigmoide logarítmica	3
RadBas	Base radial	2
TriBas	Base triangular	1
PureLin	Linear	0
SatLin	Linear saturado	0
SatLins	Linear saturado simétrico	0
PosLin	Linear positivo (ReLU)	0
HardLim	Limite rígido	0
HardLims	Limite rígido simétrico	0

Primeiro, recuperamos os dados característicos da rede e quantizamos os coeficientes da rede com a precisão investigada. O CQE é um tipo de erro bastante diferente que será investigado na Secção 5. Em seguida, em dois ciclos incorporados, os neurónios são processados camada por camada. De acordo com (1), as entradas são ponderadas e a soma ponderada é calculada e adicionada ao viés. Por fim, o resultado é conduzido através de um AF. Após cada operação, o resultado é quantizado com a precisão desejada. Paralelamente, as operações também são realizadas em FP64 como referência. Dessa forma, ganhamos controle sobre o erro de quantização, de modo que poderemos verificar a eficácia da aproximação teórica que será fornecida na Seção 4.

Durante a simulação, o algoritmo pode lidar com dez AFs comumente usados, delineados na Tabela 1. A tabela também inclui o número de operações de quantização necessárias para simular o comportamento da rede em uma precisão limitada. É importante mencionar que a AF Unidade Linear Retificada (ReLU) é, por definição, uma AF PosLin. Os diagramas de blocos de três AFs diferentes são representados na Fig. 4. Como os outros AFs têm diagramas muito mais simples, que são principalmente operações de saturação, não os apresentamos nesta figura. Observe que o número de blocos pode ser maior do que o número correspondente de quantizadores. Por exemplo, a função TanSig consiste em seis blocos. Entre eles, a multiplicação por 2 não injeta um QNP extra, pois essas operações ± afetam apenas o expoente e o sinal do número FP. Isso diminui o número de quantizadores necessários para quatro. No entanto, esses dois blocos também contribuem para a propagação de erros de quantização. Como parte da análise teórica do QNP, isso será discutido em detalhes na Seção 4.

4 Análise teórica do QNP

Nesta secção, analisamos o QNP de uma rede MLP do ponto de vista teórico. Para isso, o QNP é calculado após cada operação como uma soma de duas fontes de ruído.

A primeira fonte de ruído tem origem no QNP das operações anteriores. Esta fonte é moldada pela operação real. Por outras palavras, é propagada. Será discutida com mais detalhe mais adiante nesta secção.

A segunda fonte de ruído modela o comportamento de quantização da operação atual como uma variável aleatória uniformemente distribuída, conforme discutido na Seção 2. Durante a análise, como as variáveis de ruído de quantização são independentes umas das outras, a variância após cada operação pode ser determinada como a soma das variâncias dessas duas fontes.

Como o ruído de quantização tem média zero, após cada operação, o valor esperado será o valor real. Devido a essa propriedade, a variância do ruído de quantização e o QNP, que é o valor esperado ao quadrado, também coincidem. Se propagarmos todas as fontes de ruído de quantização pela rede, na saída, obteremos o QNP cumulativo. Isso é benéfico, pois poderemos definir um limite de erro para

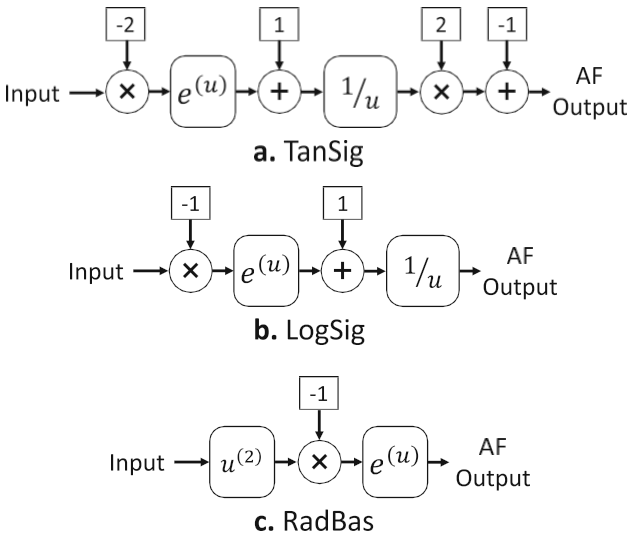

Figura 4 Diagramas de blocos de **a**. TanSig, **b**. LogSig e **c**. RadBas AFs, onde *u* representa uma entrada da operação correspondente.

a saída quantizada da rede que foi calculada na Secção 3, considerando o seguinte. Como temos várias operações na rede, podemos assumir que o QNP resultante seguirá a distribuição gaussiana de acordo com o Teorema do Limite Central. O desvio padrão é igual à raiz quadrada do QNP. Com esta suposição, podemos afirmar com 99,7% que a saída exata está na faixa $3\sqrt{\text{QNP}}$ de $\pm$ da saída quantizada.

Nos cálculos, o QNP injetado pelo quantizador atual é aproximado por (3). Esta é a primeira fonte de ruído. Conforme descrito acima, o QNP das operações anteriores é moldado pela operação atual, que é a segunda fonte. Para calcular o ruído de quantização propagado pela rede, seja uma função geral G uma função da entrada u. Com a linearização do ponto de operação [27, 28], temos que

$$\Delta G(u) \approx \frac{\partial G(u)}{\partial u} \cdot \Delta u. \tag{5}$$

No nosso caso, u é uma variável aleatória com um valor esperado do valor real e um ruído de quantização aditivo das etapas anteriores. Vamos denotar o desvio padrão desse ruído por σ_u . Como a suposição linear se aplica, o desvio padrão de $G(u)$ também estará em uma relação linear com o desvio padrão da entrada e, consequentemente, a variância é

$$\sigma^2_{G(u)} \approx \left(\frac{\partial G(u)}{\partial u}\right)^2 \cdot \sigma_u^2. \tag{6}$$

Utilizando esta abordagem, as fórmulas de propagação da variância são derivadas para as operações utilizadas neste estudo e estão listadas na Tabela 2.

Para demonstrar o cálculo do QNP na saída da rede, vamos investigar as etapas de cálculo para um neurónio único. Para as operações necessárias, consultamos (1) e Fig. 2. As etapas de avaliação são as seguintes:

1. Variação após ponderação. Uma vez que o neurónio recebe entradas de uma camada anterior ou da normalização fase de quantização, obtêm-se as variâncias de erro propagadas e propagamos o erro de acordo com a fórmula de propagação da multiplicação. A variância de saída é calculada como a soma da variância devido à propagação do erro e da variância devido à quantização após a multiplicação. Esta variância é propagada para o passo 2.
2. Variação após a soma. Para cada operando a ser somado, obtemos a variação da etapa 1 e propagamos o erro de acordo com a fórmula de propagação da adição. Além disso, após cada adição, a variação devido à quantização é adicionada. A variação acumulada resultante é propagada para a etapa 3.
3. Variança após adicionar o viés. Obtemos a variação da etapa 2 e propagamos o erro de acordo com a fórmula de propagação da adição. A variação devido à quantização é adicionada e a variação acumulada resultante é propagada para a etapa 4.
4. Variação após o AF. Obtemos a variação da etapa 3 e a levamos para a entrada do primeiro bloco do AF. Propagamos o erro através deste bloco com a fórmula de propagação de erro correspondente na Tabela 2. A variância devido à quantização é adicionada e a variância acumulada resultante é propagada para o próximo bloco no AF. Isto é repetido até que o último bloco seja avaliado. A variância acumulada é propagada para a próxima camada e será uma entrada para o passo 1.

Tabela 2 Lista de operações juntamente com a fórmula de propagação correspondente.

Operação	Fórmula de propagação da variância
$G(u) = \text{constante}$	$\sigma^2_{G(u)} = 0$
$G(u) = u + \text{constante}$	$\sigma^2_{G(u)} = \sigma_u^2$
$G(u) = u \cdot \text{constante}$	$\sigma^2_{G(u)} = (\text{constante})^2 \cdot \sigma_u^2$
$G(u) = u^2$	$\sigma^2_{G(u)} = 4G(u) \cdot \sigma_u^2$
$G(u) = e^u$	$\sigma^2_{G(u)} = G^2(u) \cdot \sigma_u^2$
$G(u) = 1/u$	$\sigma^2_{G(u)} = G^4(u) \cdot \sigma_u^2$

Seguindo estes passos, os cálculos também podem ser demonstrados usando equações matemáticas. Referindo-se a (3), Fig. 3 e Tabela 2, a estimativa teórica do QNP pode ser calculado para o neurónio. Sejam as saídas das três primeiras fases: $out_{1,i}$, out_2 e out_3 respetivamente, então o QNP é propagado da seguinte forma:

$$\text{QNP}_{1,i} = w_i^2 \cdot \text{QNP}_{x,i} + 0{,}18 \cdot 2^{-2p} \cdot E\{out_{1,i}^2\} \tag{7}$$

$$\text{QNP}_2 = \sum_{i=1}^{m} \text{QNP}_{1,i} + 0{,}18 \cdot 2^{-2p} \cdot E\{out_2^2\} \tag{8}$$

$$\text{QNP}_3 = \text{QNP}_2 + 0{,}18 \cdot 2^{-2p} \cdot E\{out_3^2\} \tag{9}$$

onde m é o número de entradas para este neurónio, portanto, o número de multiplicações.

Pode-se observar em (8) que a operação de somatório pro-

produz um erro de quantização apenas uma vez no resultado final da soma, mas isso depende do tipo de acumulador. Alguns acumuladores em diferentes microcontroladores acumulam a soma passo a passo. Como as ANNs são implementadas principalmente em plataformas de hardware paralelas, como FPGAs, assumimos que se trata de um acumulador paralelo. No entanto, a análise e a derivação do QNP podem ser realizadas usando o acumulador sequencial.

Para a quarta fase no neurónio, o AF é um bloco genérico. Vamos supor que seja um AF LogSig, referindo-se à Fig. 4b, deixe as saídas das operações AF serem AF_{op1} , AF_{op2} , AF_{op3} e AF_{op4} , então o QNP é propagado através do AF respectivamente da seguinte forma:

$$\mathrm{QNP}_{\mathrm{LogSig},1} = \mathrm{QNP}_3 \tag{10}$$

$$\mathrm{QNPLogSig},2 = \mathrm{QNPLogSig},1 \cdot AF_{op2}^{2} + 0,18 \cdot 2^{-2p} \cdot E\{AF_{op2}^{2}\}. \tag{11}$$

$$\mathrm{QNPLogSig},3 = \mathrm{QNPLogSig},2 + 0,18 \cdot 2^{-2p} \cdot E\{AF_{op3}^{2}\} \tag{12}$$

$$\mathrm{QNPLogSig},4 = \mathrm{QNPLogSig},3 \cdot AF_{op4}^{4} + 0,18 \cdot 2^{-2p} \cdot E\{AF_{op4}^{2}\} \tag{13}$$

Deve-se mencionar que, devido à propagação de erros, o QNP não cresce necessariamente após cada operação. Por um lado, existem etapas que mantêm o QNP acumulado constante. Estas são a propagação de erros devido à multiplicação por 1 e quando o AF é linear. Por outro lado, o QNP devido à propagação de erros também pode diminuir, dependendo do operando de entrada. Por exemplo, a multiplicação por uma constante com valor absoluto inferior a um diminuirá o QNP acumulado. Além disso, o QNP na saída de um AF pode ser zero quando o AF tem saída constante ou, em outras palavras, saturada. Ou seja, a derivada de uma constante é zero.

5 Análise do erro de quantização do coeficiente

O QNP foi analisado e aproximado nas secções anteriores, mas o impacto dos erros de coeficiente nunca foi devidamente abordado. Esta secção aprofunda o CQE; portanto, será apresentado um método para calcular e neutralizar esse erro.

Este tipo de erro contribui para o erro total na saída, deslocando ou enviesando cada amostra. Este comportamento de enviesamento ocorre devido à natureza do CQE; ou seja, o erro entre os valores exatos e quantizados do coeficiente é fixo quando diferentes amostras de entrada são introduzidas. Estes pequenos erros de enviesamento são acumulados e propagados para a saída de cada coeficiente no modelo. Desta forma, cada amostra na saída do modelo é enviesada. É crucial preservar o sinal desse enviesamento para que este fenómeno possa ser compensado. Conforme descrito na literatura, este tipo de erro não pode ser aproximado teoricamente para sistemas não lineares e só pode ser calculado usando simulação [13].

Para calcular o CQE, foi configurado um modelo de modo a que apenas os coeficientes fossem quantizados, enquanto as operações eram realizadas na resolução padrão (FP64). Desta forma, os

A saída deste modelo pode ser comparada com o modelo exato, e a diferença é simplesmente o CQE. Infelizmente, este método não pode ser utilizado quando se considera a quantização da operação, mas ainda assim é útil para verificar o método seguinte.

Um método mais sistemático e geral é apresentado neste artigo. Cada coeficiente gera um CQE que se propaga através de diferentes operações para afetar o resultado final. É necessário calcular o CQE propagado após cada operação; portanto, fórmulas de propagação de erros são derivadas para todas as operações utilizadas neste estudo, e elas são apresentadas na Tabela 3. As três primeiras fórmulas da tabela são fórmulas de propagação especiais que se aplicam apenas à propagação do erro de quantização [13], enquanto as restantes fórmulas são derivadas utilizando (5) [27, 28].

Para verificar as fórmulas derivadas na Tabela 3, os resultados dos dois métodos são comparados e o erro entre eles é insignificante, como pode ser visto na Fig. 5. O CQE para os dois métodos nesta experiência estão na faixa de 10^{-4} , enquanto a diferença entre eles para todas as amostras está na faixa de 10^{-7} . Portanto, a precisão é de 99,99%, o que foi alcançado em todas as experiências realizadas com diferentes parâmetros de simulação. Esta observação confirma a aplicabilidade do método geral com alta precisão.

O método geral apresentado foi utilizado quando as ANNs MLP totalmente quantizadas foram analisadas. Desta forma, o CQE na saída é conhecido; portanto, pode ser compensado na saída do modelo.

Após discutir o QNP e o CQE, parte do código para o RadBas AF é mostrado na Fig. 6 para demonstrar como o algoritmo funciona. Existem três operações neste AF apresentadas na Fig. 4c. Para cada operação, o quantizado out-

Tabela 3 Fórmulas de propagação de erro utilizadas no cálculo do

CQE. Operação	Propagação de erro
$G(u) = \pm u \pm \text{constante}(c)$	$\sigma_{G(u)} = \pm\sigma_u \pm \sigma_c$
$G(u) = \pm u_1 \pm u_2 \pm \ldots$	$\sigma_{G(u)} = \pm\sigma_{u1} \pm \sigma_{u2} \pm \ldots$
$G(u) = u \cdot \text{constante}(c)$	$\sigma_{G(u)} = (c \cdot \sigma_u) + (u \cdot \sigma_c)$
$G(u) = e^{u}$	$\sigma_{G(u)} = G(u) \cdot \sigma_u$
$G(u) = u^2$	$\sigma_{G(u)} = 2 \cdot G(u) \cdot (\sigma_u / u)$
$G(u) = 1/u$	$\sigma_{G(u)} = -1 \cdot (\sigma_u / u^2)$
$G(u) = \lvert u \rvert$	$\sigma_{G(u)} = \sigma_u$, quando $(u > 0)$ $\sigma_{G(u)} = -\sigma_u$, quando $(u < 0)$
$G(u) = Pos\ Lin(u)$	$\sigma_{G(u)} = \sigma_u$, quando $(u > 0)$ $\sigma_{G(u)} = 0$, quando $(u < 0)$
$G(u) = Sat\ Lin(u)$	$\sigma_{G(u)} = \sigma_u$, quando $(0 < u < 1)$ $\sigma_{G(u)} = 0$, Caso contrário
$G(u) = Sat\ Lins(u)$	$\sigma_{G(u)} = \sigma_u$, quando$(\lvert u \rvert < 1)$ $\sigma_{G(u)} = 0$, Caso contrário
$G(u) = Hard\ Lim(u)$	$\sigma_{G(u)} = 0$
$G(u) = Limites\ Rígidos(u)$	$\sigma_{G(u)} = 0$

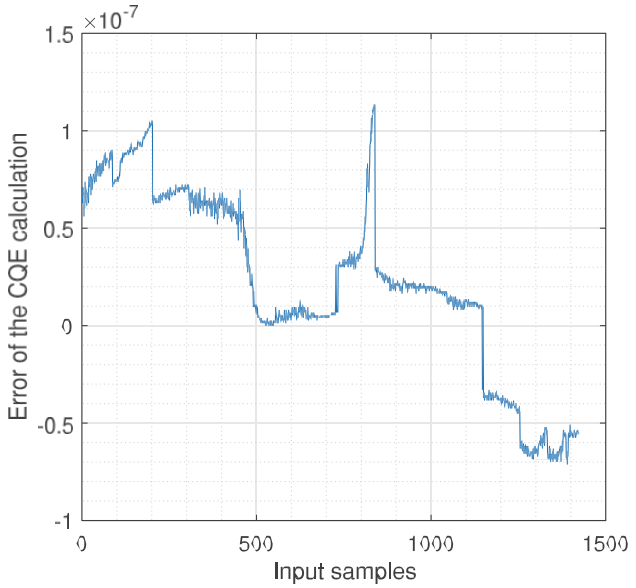

Figura 5 A diferença entre o CQE calculado com base nas operações FP64 e a propagação de erro proposta.

put é calculado, então o QNP é estimado juntamente com o cálculo do CQE. Desta forma, os valores são guardados para a camada seguinte.

6 Estudos de caso

Nesta secção, a análise proposta será realizada em dois conjuntos de dados disponíveis publicamente. Para verificar a aplicabilidade do método, os resultados da análise teórica, descritos na Secção 4, serão comparados com os erros de quantização simulados, conforme demonstrado na Secção 3. Para esta comparação, o CQE, apresentado na Secção 5, também é considerado.

6.1 Estudo de caso 1: Estação meteorológica

O primeiro conjunto de dados foi obtido do canal 12397 do ThingSpeak™ de uma estação meteorológica em Natick, Massachusetts, EUA. Como esse canal armazena novas leituras todos os dias, usamos os dados armazenados durante uma semana, de 1º de julho de 2023 a 8 de julho de 2023. Os dados podem ser acessados usando o comando "thingS-peakRead" do MATLAB, especificando o ID do canal, a duração e os atributos. Neste conjunto de dados, a humidade, a temperatura, a pressão atmosférica e a velocidade do vento são os atributos usados para calcular o ponto de orvalho [29].

Diferentes experiências de configuração de treino de MLP ANN foram utilizadas para esta análise, incluindo o número de neurónios, o número de camadas e o tipo de AF em cada camada. O objetivo é monitorizar o comportamento do ruído de quantização para diferentes estruturas MLP. Assim, inicialmente, foi treinada uma pequena rede com dez neurónios em uma camada oculta com um AF LogSig, além da camada de saída padrão de um único neurónio com AF PureLin.

O conjunto de dados consiste em 8000 amostras. Ele foi dividido em duas partes iguais: a primeira metade foi usada para fins de treinamento, validação e teste com porcentagens de (80,10,10), respectivamente. A segunda metade foi mantida para a análise de erro de quantização. A quantização FP foi realizada com precisão simples, ou seja, com precisão de 24 bits, expoente de 8 bits e arredondamento padrão para o método mais próximo. Os valores médios teóricos e simulados do QNP na saída final do modelo são $2{,}8\ 10^{-11}$ e $2{,}95\ 10^{-11}$, respetivamente.

Mais ilustrações do comportamento da quantização podem ser vistas nos histogramas mostrados na Fig. 7. A figura traça dois histogramas para o número de ocorrências da média quadrática do erro de quantização em função do desvio padrão teórico. Para interpretar os resultados, vamos relembrar a suposição de que o ruído de quantização na saída é de distribuição gaussiana com média zero. Para cada instância de entrada, o
O erro de quantização na saída *(v)* é normalizado pelo desvio padrão da variável aleatória gaussiana $(\sqrt{QNP})$. Em outras palavras, as variáveis aleatórias são padronizadas. O primeiro histograma, com o marcador circular, é plotado quando o CQE é neutralizado. A figura mostra que os erros reais estão principalmente
na faixa $\pm 3\ v/\sqrt{QNP}$, e a forma da curva é
Gaussiana com uma média em torno de zero. Por outro lado, o segundo histograma é apresentado para enfatizar o impacto do CQE, onde pode-se observar que o histograma é deslocado devido ao CQE em cada amostra de saída. Por esse motivo, é crucial compensar esse fenómeno.

Para observar o efeito da precisão no QNP, a mesma ANN treinada foi simulada com diferentes precisões de 3 a 52 bits, conforme mostrado na Fig. 8. Lembre-se de que FP64, que tem precisão de 53 bits, é usado como referência. Assim, para $p = 53$, o erro seria exatamente 0. Em cada precisão, são mostradas as médias dos QNPs reais e aproximados. Como observação adicional, podemos observar que, como esperado, o QNP diminui para um quarto do valor original se a precisão for aumentada em um. Esta é uma consequência natural das propriedades da quantização. Como a escala vertical é logarítmica, o QNP, como uma função de p, pode ser caracterizado por uma linha reta.

Para abordar o efeito da quantização em uma grande ANN MLP, um modelo com nove camadas foi treinado usando o mesmo conjunto de dados. Por definição, este é um modelo profundo no qual cada camada contém neurônios organizados da seguinte forma: (5, 10, 20, 40, 30, 20, 10, 5, 3) respectivamente, e todas essas camadas têm AFs TanSig. Este modelo foi simulado com a representação FP de meia precisão que tem precisão de 11 bits, incluindo o bit oculto, e os valores médios do QNP simulado e estimado são ambos iguais a $1{,}27\ 10^{-2}$.

A relação entre o número de camadas e o erro de quantização não pode ser calculada com exatidão, pois muitos outros parâmetros influenciam os fenômenos de quantização, como o

Figura 6 Parte do código do algoritmo mostrando o cálculo dos dois ciclos, o QNP aproximado e o cálculo do CQE, para o RadBas AF.

```
% Definições:
% A letra (q) denota o ciclo quantizado.
% Entrada(q) / Saída(q) são terminais AF.
% QNPin e CQEin são de operações anteriores.
% roundq é a função quantizadora.
% j é o índice do neurónio no loop da camada.
% Q é o quantizador predefinido de:
% p é a precisão reduzida pretendida.

% Ciclo 1: Resolução de bits padrão
(FP64) op1   = Entrada.^2;
op2    = (-1)*op1;
op3    = exp(op2);
Saída(j,:) = op3; % Guardar a saída do neurónio na camada
seguinte.

op1q   = Inputq.^2;      % Primeira operação RadBas.
QNP1 = (4.*op1q.*QNPin) + 0,18*(2^(-2*p))*((op1q.^2));
CQE1 = 2*op1q.*(CQEin./Inputq);
op1q   = roundq(op1q,Q);
op2q   = (-1)*op1q;       % Segunda operação RadBas.
CQE2 = CQE1.*(-1);
op3q   = exp(op2q);      % Terceira operação RadBas.
QNP3 = QNP1.*((op3q.^2)) + 0,18*(2^(-2*p))*((op3q.^2));
CQE3 = op3q.*CQE2;
op3q   = roundq(op3q,Q);
Saídaq(j,:) = op3q;        % Guardando a saída do
neurónio, QNP QNPout(j,:) = QNP3;     % e CQE para a
camada seguinte.
CQEout(j,:) = CQE3;
```

número de neurónios, os valores dos coeficientes e o tipo de AF. A utilização do método proposto pode ser bastante vantajosa, pois permite caracterizar com precisão o QNP da estrutura da rede investigada.

6.2 Estudo de caso 2: Diagnóstico de cancro

O segundo conjunto de dados foi recolhido para realizar a classificação no diagnóstico do cancro. Nove parâmetros de entrada (tamanho, forma, etc.) e duas classes incorporadas numa saída, que decide se o caso é um tipo de cancro benigno ou maligno, foram considerados neste conjunto de dados. Os dados medidos estão disponíveis publicamente em [30]. Contém 698 casos medidos divididos em duas partes iguais, de forma semelhante à Secção 6.1.

Uma ANN de dez neurónios numa camada oculta com um AF Tan-Sig foi treinada para realizar a classificação. Como este estudo de caso trata de um problema de classificação, é essencial verificar quando a ANN classifica incorretamente os dados e fornece um diagnóstico errado. A Figura 9 mostra o número de amostras com diagnóstico errado entre as 349 amostras. Com precisão de 5 bits ou menos, a rede diagnosticou incorretamente pelo menos um caso. Tendo em mente que o número de bits exponenciais não afeta o erro de quantização, conforme descrito na Seção 2, por esse motivo, focamos apenas na precisão. Nesta experiência, empregamos 4 bits para o expoente, o que é suficiente para evitar um estouro ao alterar a precisão. Por exemplo, com três bits de precisão, há duas amostras classificadas incorretamente. Para explicar como essas duas amostras foram classificadas incorretamente, apresentamos a Fig. 10, em

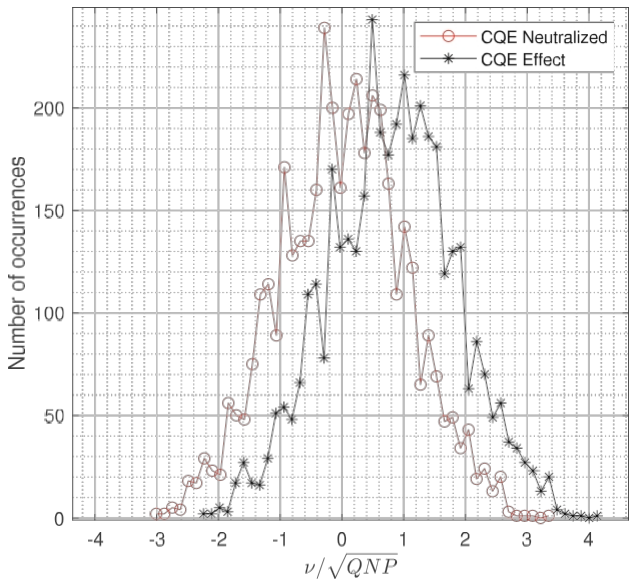

Figura 7 Estudo de caso da estação meteorológica: Histogramas da média quadrática do QNP simulado exato e do desvio padrão teórico para dois casos com e sem a compensação para o CQE.

onde são observadas as quatro amostras mais próximas do limiar. Pode-se observar que, para a segunda amostra, o valor exato é 0,4, mas devido à quantização, tornou-se 0,5, que está no limiar, e na operação de classificação, é arredondado para um. Uma situação semelhante pode ser observada na quarta amostra, enquanto as outras amostras são classificadas corretamente. Esta demonstração mostra claramente a importância da análise: com os limites de erro, o utilizador pode ser notificado de que o resultado está sujeito a erros de classificação apenas devido à quantização.

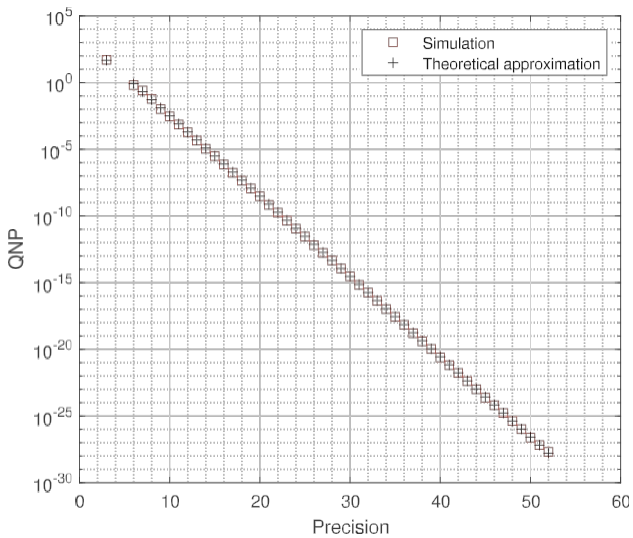

Figura 8 Estudo de caso da estação meteorológica: O efeito da diminuição da precisão no QNP.

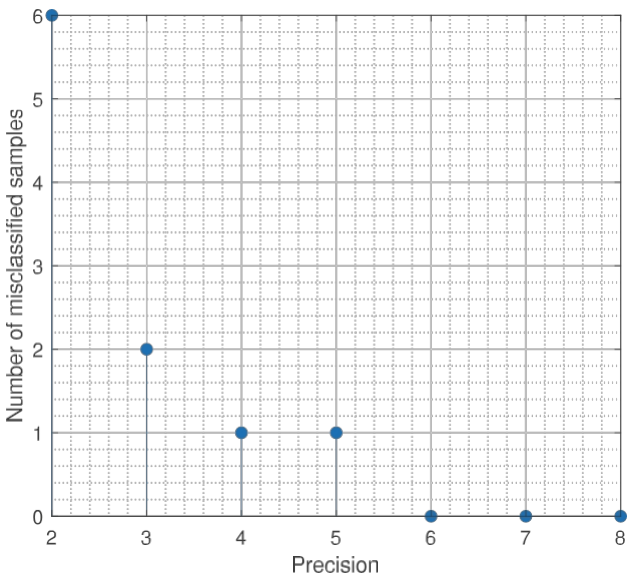

Figura 9 Estudo de caso sobre diagnóstico de cancro: O efeito da diminuição da precisão na classificação.

Nestes dois estudos de caso, foram utilizados e analisados os AFs TanSig, LogSig, RadBas e PureLin. Os outros AFs listados na Tabela 1 estão menos envolvidos. O seu comportamento pode ser descrito da seguinte forma. O TriBas tem apenas uma operação de adição que produz QNP e propaga erros de etapas anteriores, mas fica saturado quando a entrada excede o intervalo ±1.

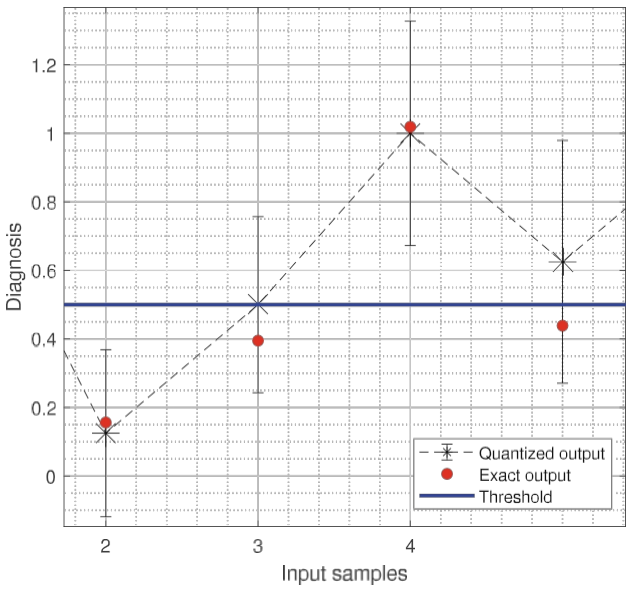

Figura 10 Estudo de caso de diagnóstico de cancro: resposta de saída para quatro amostras para ilustrar o efeito do erro de quantização na decisão de classificação ao usar três bits de precisão. A barra de erro do desvio padrão também é mostrada em cada ponto.

Portanto, na saturação, o QNP na saída do AF é zero, conforme mencionado anteriormente. Os AFs Satlin, SatLins e PosLin (ReLU) são lineares com saturação. Na fase de saturação, eles produzem erro de quantização zero. Na fase linear, eles não produzem nenhum erro de quantização adicional, e apenas o erro de entrada é propagado. Por fim, os HardLim e HardLims estão sempre em saturação, o que torna o QNP na sua saída zero. Esses AFs de saturação também contribuem para a propagação do CQE, conforme listado na Tabela 3.

7 Conclusões

Neste estudo, as redes neurais artificiais MLP foram analisadas em termos de quantização FP. A análise inclui uma aproximação estatística do QNP e um cálculo preciso do CQE. Assim, o efeito do CQE pode ser observado e tratado para uma melhor aproximação do QNP.

Foi apresentado um algoritmo genérico para calcular o QNP e o CQE de uma rede de precisão reduzida com representação numérica FP. O algoritmo proposto abrange dez AFs que são comumente usados em aplicações recentes. Em cálculos operacionais, o QNP e o CQE foram propagados através do modelo, empregando a fórmula necessária mencionada em cada etapa.

Dois estudos de caso foram considerados para modelos de regressão e classificação. O algoritmo codificado conseguiu prever com precisão o QNP em ambos os casos numa ampla gama de bits de precisão aplicados. Em geral, as redes MLP podem apresentar um comportamento altamente não linear, o que dificulta a determinação do erro de quantização na saída da rede. O algoritmo descrito neste artigo visa facilitar o projeto de hardware, no sentido de que, em uma determinada precisão, ele produz os limites de erro da saída. Isso pode ser feito antes da implementação real. Dessa forma, o projetista de hardware pode selecionar a resolução suficiente adequada para a aplicação na fase de projeto.

Contribuições dos autores Os autores confirmaram as suas contribuições para o artigo da seguinte forma: redação do manuscrito, codificação e design: Al-Rikabi, Hussein; Concepção do estudo, análise e interpretação dos resultados, preparação do rascunho do manuscrito e todo o projeto foi supervisionado pelo Dr. Renczes, Balázs. Todos os autores revisaram os resultados e aprovaram a versão final do manuscrito.

Financiamento Financiamento de acesso aberto fornecido pela Universidade de Tecnologia e Economia de Budapeste. O projeto n.º TKP2021-EGA-02 foi implementado com o apoio do Ministério da Cultura e Inovação da Hungria, proveniente do Fundo Nacional de Investigação, Desenvolvimento e Inovação, financiado ao abrigo do regime de financiamento TKP2021-EGA.

Disponibilidade de dados e código O código-fonte e os conjuntos de dados gerados e/ou analisados durante o presente estudo estão disponíveis mediante solicitação razoável aos autores correspondentes.

Declarações

Aprovação ética Declaro o seguinte: o manuscrito não foi publicado nem está a ser considerado para publicação em outro lugar; não faz parte de outro artigo publicado; não há plágio; todos os dados e recursos utilizados foram aprovados; e o software utilizado é licenciado.

Conflitos de interesses Os autores declaram que não há conflitos de interesses financeiros ou não financeiros relevantes a relatar.

Referências

1. Chao, Z., & Kim, H. J. (2020). Segmentação de imagens cerebrais com base na combinação de rede neural de retropropagação e sistema AdaBoost. *Journal of Signal Processing Systems, 92*, 289–298.
2. Sahoo, M., Dey, S., Sahoo, S., Das, A., Ray, A., Nayak, S., & Subudhi, E. (2023). Abordagem de rede neural MLP (perceptrão multicamadas) e RBF (função de base radial) para estimar e otimizar o teor de 6-gingerol em Zingiber officinale Rosc. em diferentes condições agroclimáticas. *Culturas e produtos industriais, 198*, 116658.
3. Yin, P., Wang, C., Liu, W., Swartzlander, E. E., & Lombardi, F. (2018). Projetos de multiplicadores de ponto flutuante aproximado com precisão variável para aplicações tolerantes a erros. *Journal of Signal Processing Systems, 90*, 641–654.
4. Barrachina, J. A., Ren, C., Morisseau, C., Vieillard, G., & Ovarlez, J. P. (2023). Comparação entre arquiteturas equivalentes de redes neurais de valores complexos e valores reais - aplicação na segmentação de imagens SAR polarimétricas. *Journal of Signal Processing Systems, 95*(1), 57–66.
5. Huang, A., Cao, Z., Wang, C., Wen, J., Lu, F., & Xu, L. (2021). Uma Rede neural em chip baseada em FPGA para tomografia TDLAS em chamas dinâmicas. *Transações IEEE sobre Instrumentação e Medição, 70*, 1–11.
6. Garg, M., Arora, A., & Gupta, S. (2021). Uma identificação humana eficiente através do sistema de reconhecimento da íris. *Revista de Sistemas de Processamento de Sinais, 93*, 701–708.
7. Yu, H., Shou, G., Zhang, X., Li, H., Liu, Y., & Hu, Y. (2023). Aplicação de redes neurais para prever a escala de tempo local UTC com conjunto de relógios. *Transações IEEE sobre Instrumentação e Medição, 72*, 1–9.
8. Chinatamby, P., & Jewaratnam, J. (2023). Um estudo comparativo de desempenho sobre a previsão de PM2,5 em áreas industriais utilizando diferentes algoritmos de treino de redes neurais feedforward-backpropagation (FBNN). *Chemosphere, 317*, 137788.

9. Hassan, O., Paul, T., Shuvo, M. H., Parvin, D., Thakker, R., Chen, M., Mosa, A. S. M., & Islam, S. K. (2022). Inferência de aprendizagem profunda energeticamente eficiente incorporada em FPGA para deteção de apneia do sono. *Journal of Signal Processing Systems*, *94*(6), 609–619. https://doi.org/10.1007/s11265-021-01722-7 , https://link.springer.com/article/10.1007/s11265-021-01722-7
10. IEEE. (2019). *Norma para aritmética binária de ponto flutuante*. Std 754-2019, IEEE.
11. Liang, S., Yin, S., Liu, L., Luk, W., & Wei, S. (2018). FP-BNN: Rede neural binarizada em FPGA. *Neurocomputing, 275*, 1072–1086.
12. Shah, M., Arunachalam, S., Wang, J., Blaauw, D., Sylvester, D., Kim, H. S., Seo, J. S., & Chakrabarti, C. (2018). Uma arquitetura de rede neural de ponto fixo para aplicações de fala em hardware com recursos limitados. *Journal of Signal Processing Systems, 90*, 727–741.
13. Widrow, B., & Kollár, I. (2008). *Ruído de quantização: erro de arredondamento em computação digital, processamento de sinais, controlo e comunicações*. Cambridge, Reino Unido: Cambridge University Press.
14. Renczes, B. (2017). Cálculo preciso de argumentos de ponto flutuante para algoritmos de ajuste de seno. *IEEE Transactions on Instrumentation and Measurement, 66*(11), 2988–2996.
15. Alrwashdeh, M., & Kollár, Z. (2022). Análise do ruído de quantização em transmissores FBMC. *Digital Signal Processing, 131*, 103760.
16. Huang, K., Ni, B., & Yang, X. (2019). Quantização eficiente para redes neurais com pesos binários e ativações de baixa largura de bits. *Proceedings of the AAAI Conference on Artificial Intelligence, 33*(1), 3854–3861.
17. Zhu, C., Han, S., Mao, H., & Dally, W. J. (2016). Quantização ternária treinada. Pré-impressão recuperada de http://arxiv.org/abs/1612.01064. Conferência Internacional sobre Representações de Aprendizagem (ICLR) (2017).
18. Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., & Kalenichenko, D. (2018). *Quantização e treino de redes neurais para inferência eficiente apenas com aritmética inteira* (pp. 2704–2713). Salt Lake City, UT, EUA: 2018 IEEE/CVF Con- https://doi.org/10.1109/CVPR.2018.00286, https://arxiv.org/abs/1712.05877
19. Nichols, K. R., Moussa, M. A., & Areibi, S. M. (2002). *Viabilidade da aritmética de ponto flutuante em redes neurais artificiais baseadas em FPGA* (pp. 8–13). San Diego, Califórnia: Proc. CAINE, 15.ª Conferência Internacional sobre Aplicações Informáticas na Indústria e Engenharia. URL: Scopus.
20. Peric´, Z. H., Savic, M. S., Dincic, M. R., Vuc˘ic´, N. J., Djosic, D., & Milosavljevic, S. (2021). *Quantizadores de ponto flutuante e ponto fixo de 32 bits para quantização de pesos de redes neurais* (pp. 1–4). Bucareste, Roménia: 12.º Simpósio Internacional sobre Tópicos Avançados em Engenharia Elétrica (ATEE). https://doi.org/10.1109/ATEE52255.2021.9425265
21. He, L., Zheng, S., Chen, W., Ma, Z. M., & Liu, T. Y. (2019). OptQuant: Treinamento distribuído de redes neurais com mecanismos de quantização otimizados. *Neurocomputing, 340*, 233–244.
22. Al-Rikabi, H., & Renczes, B. (2022). *Análise de erros de arredondamento de ponto flutuante em redes neurais artificiais* (pp. 79–83). Brescia: 25.º Simpósio Internacional IMEKO TC-4 sobre Medição de Quantidades Elétricas, IMEKO TC-4 2022 e 23.º Workshop Internacional sobre Modelagem e Teste de ADC e DAC, IWADC 2022. https://doi.org/10.21014/tc4-2022.15
23. Alsadi, N., Gadsden, S. A., & Yawney, J. (2023). Estimativa inteligente: uma revisão da teoria, aplicações e avanços recentes. *Processamento de Sinais Digitais, 135*, 103966.
24. Al-Rikabi, H., Al-Ja'afari, M. A., Ali, A. H., & Abdulwahed, S. H. (2020). Implementação de modelo genérico de funções de ativação de redes neurais profundas usando modelo SCPWL otimizado por GWO em FPGA. *Microprocessadores e Microssistemas, 77*, 103141.
25. Shanmuganathan, S. (2016). *Modelagem de redes neurais artificiais: uma introdução* (pp. 1–14). Cham: Springer International Publishing.
26. Russinoff, D. M. (2019). *Verificação formal do projeto de hardware de ponto flutuante* (1.ª ed.). Cham: Springer.
27. BIPM, IEC, IFCC, ILAC, ISO, IUPAC, IUPAP, OIML. Avaliação de dados de medição — Guia para a expressão da incerteza na medição. Comité Conjunto para Guias em Metrologia, JCGM 100:2008.
28. Dieck, R. H. (2007). *Incerteza de medição: métodos e aplicações* (4.ª ed.). ISA-The Instrumentation, Systems, and Automation Society.
29. Wetjen, E. *Canal Thingspeak 12397; Estação meteorológica MathWorks, Natick, EUA*. https://thingspeak.com/channels/12397. Acedido em julho de 2023.
30. Garza-Ulloa, J. (2022). Engenharia biomédica aplicada usando inteligência artificial e modelos cognitivos - capítulo 5 - conjunto de dados - princípios de modelos de aprendizagem profunda aplicados à engenharia biomédica. https://doi.org/10.17632/nc9m5zm8st.1. Acedido em julho de 2023.

Hussein Al-Rikabi obteve o seu mestrado em engenharia eletrónica pela Universidade de Tecnologia de Bagdade, Iraque, em 2017. Posteriormente, em 2021, iniciou os seus estudos de doutoramento em engenharia elétrica na Universidade de Tecnologia e Economia de Budapeste, Hungria. As suas áreas de investigação atuais incluem a investigação de erros de quantização em modelos de inteligência artificial, formação sensível à quantização e otimização da formação.

Balázs Renczes obteve o seu mestrado e doutoramento em Engenharia Elétrica pela Universidade de Tecnologia e Economia de Budapeste, Hungria, em 2013 e 2017, respetivamente. Desde 2023, é professor associado no Departamento de Sistemas de Medição e Informação da Universidade de Tecnologia e Economia de Budapeste. Atualmente, investiga problemas de processamento de sinais em sistemas quantizados de forma aproximada e a otimização de técnicas de identificação de sistemas não lineares.