Análise de Desempenho do Software NNCodec Utilizando Redes Neurais para Processamento de Imagens

Sara V. Henssler[1], Jiovana S. Gomes[2], Sergio Bampi[2], Fábio L. Livi Ramos[1]

[1]Universidade Federal do Pampa
(UNIPAMPA)
RS - Brasil

{sarahenssler.aluno,fabioramos}@unipampa.edu.br

[2]PGMICRO - Instituto de Informática, Universidade federal do Rio Grande do Sul
(UFRGS).
RS - Brasil

{jsgomes, bampi} @inf.ufrgs.br

Resumo. *Os algoritmos de redes neurais profundas têm sido amplamente utilizados no contexto do processamento visual. Dada a complexidade e a demanda por armazenamento dos modelos que lidam com esse tipo de dado, em contextos práticos de aplicação com dispositivos e sistemas com limitações de hardware, a compressão torna-se essencial. Este trabalho avalia o desempenho do NNCodec, implementação de referência do padrão de codificação de redes neurais, Neural Network Compression and Representation, tanto em termos de eficiência de compressão quanto do impacto da compressão no desempenho das redes. Os resultados indicam que o NNCodec reduziu significativamente o tamanho das redes, alcançando uma taxa de compressão média de, aproximadamente, 86,78%, e preservou o desempenho dos modelos avaliados.*

1. Introdução

Com o crescimento exponencial das tecnologias de Inteligência Artificial (IA), os algoritmos de Redes Neurais Profundas (DNN – *Deep Neural Networks*) [Islam et al. 2019] têm se tornado indispensáveis para otimizações, em diversas áreas da tecnologia, devido à sua capacidade de reconhecer padrões e aprender a desempenhar tarefas sem a necessidade de programação explícita. Uma das áreas em questão é a de processamento visual, onde a aplicação de DNNs impõe desafios significativos devido à complexidade dos dados. Para extrair características relevantes de imagens, como bordas, texturas e formas, essas redes geralmente adotam arquiteturas mais complexas, resultando em modelos com milhões ou até bilhões de parâmetros, o que implica em elevadas demandas de armazenamento e capacidade computacional para a execução dessas DNNs.

Perante isso, o *Moving Pictures Experts Group* (MPEG) desenvolveu o padrão de compressão e representação de redes neurais, *Neural Network Compression and Representation* (NNC) [Kirchhoffer et al. 2021], publicado como a norma ISO/IEC 15938-17. Este padrão define métodos eficientes para a codificação e decodificação de DNNs, com proposta de não afetar significativamente o desempenho das mesmas. O *software* de referência do padrão, NNCodec [Becking et al. 2023], foi desenvolvido pelo Instituto Fraunhofer Heinrich Hertz (HHI), lançado em 2023 de forma aberta e implementa os métodos previstos no padrão NNC para o processamento de redes neurais.

Portanto, diante do crescente tamanho e complexidade das DNNs, e da crescente demanda por recursos computacionais, principalmente na área de processamento visual, este trabalho propõe analisar o desempenho do software NNCodec tanto em termos de capacidade de compressão quanto no impacto causado pela compressão, nos modelos de DNNs, utilizando DNNs aplicadas a tarefas de processamento de imagens.

2. Metodologia Experimental

Para a avaliação do software, foram selecionados sete modelos pré-treinados de DNNs aplicados a diferentes tarefas de processamento de imagens, testados com seus respectivos *datasets*: (i) AlexNet [Krizhevsky 2014] e (ii) Inception [Szegedy et al. 2016], aplicados à classificação de imagens com o *dataset* ImageNet; (iii) YOLOv11 [Jocher and Qiu 2024], utilizado para detecção de objetos com o *dataset* COCO 2017; (iv) DeeplabV3_Resnet50 [Chen et al. 2018] e (v) U-Net [Ronneberger et al. 2015], empregados em segmentação semântica com os *datasets* COCO 2017 e CARVANA, respectivamente; e, por fim, os modelos da biblioteca CompressAI (C.AI) para compressão de imagens, (vi) mshj2018factorized [Chen et al. 2018] e (vii) cheng2020anchor [Cheng et al. 2020], avaliados com o *dataset* CLIC 2021 Professional Test.

Para aferir a capacidade de compressão do software, os modelos selecionados foram comprimidos por meio de *scripts* em *Python*. Foram utilizados os parâmetros *default* do software para garantir uma avaliação imparcial. A compressão dos modelos foi realizada em um *desktop* equipado com um processador *Intel Core i7-3770*, 16 *gigabytes* de memória RAM e sistema operacional Kubuntu 24. As métricas coletadas no processo de compressão, para cada um dos modelos, foram o tempo de codificação, o tempo de decodificação e a taxa de compressão, que indica a razão entre o tamanho do modelo original e o tamanho do modelo comprimido.

Para verificar o impacto causado pela compressão nos modelos de DNNs, tanto os modelos originais quanto os modelos comprimidos foram submetidos a testes de inferência. A partir da inferência, foram obtidas métricas de desempenho que possibilitaram comparar os modelos comprimidos com os modelos originais e determinar a degradação de desempenho. A inferência dos modelos foi realizada por meio de *scripts* em *Python*, seguindo as recomendações na documentação de cada modelo. Foram utilizadas bibliotecas ou APIs específicas de cada modelo ou *dataset* ou, quando disponível, a própria função *evaluate* do modelo. A máquina utilizada para a inferência foi a mesma usada no processo de compressão, com exceção aplicada para os modelos de segmentação semântica, que são redes convolucionais e apresentariam um tempo de teste muito elevado. Para contornar esse problema, a aceleração por hardware seria o ideal; portanto, optou-se pelo uso de um *desktop* com as seguintes configurações: processador *AMD Ryzen 9 5950X*, placa de vídeo *RTX 4070 Ti*, 32 *gigabytes* de memória RAM e sistema operacional Ubuntu 22.04.5 LTS.

3. Resultados

Os resultados referentes à capacidade de compressão são apresentados na Tabela 1, onde é possível observar que o NNCodec reduziu significativamente o tamanho dos modelos, alcançando uma taxa de compressão média de, aproximadamente, 86,78%. Os tempos de codificação foram superiores aos de decodificação, uma vez que a decodificação apenas aplica os processos inversos já determinados e aplicados na codificação.

Tabela 1. Resultados para a etapa de compressão

Modelo	Tempo de codificação (s)	Tempo de decodificação (s)	Taxa de compressão (%)
AlexNet	36,68	5,87	87,77
Inception	30,38	5,71	85,11
C.AI 2018[1]	2,56	0,52	84,26
C.AI 2020[2]	12,59	2,94	82,03
YOLOv11	13,25	1,99	89,22
DeeplabV3_Resnet50	26,42	4,84	90,46
U-Net	20,90	2,96	88,64

[1] *mshj2018factorized*; [2] *cheng2020anchor*;

Tabela 2. Resultados para a etapa de validação

Modelo	Acurácia (%)	IoU	DSC	BPP	PSNR
AlexNet O[1]	56,51	–	–	–	–
AlexNet R[2]	56,51	–	–	–	–
Inception O[1]	68,98	–	–	–	–
Inception R[2]	69,49	–	–	–	–
C.AI 2018[3] O[1]	–	–	–	0,108	29,37
C.AI 2018[3] R[2]	–	–	–	0,159	30,67
C.AI 2020[4] O[1]	–	–	–	0,083	30,85
C.AI 2020[4] R[2]	–	–	–	0,122	NaN
YOLOv11 O[1]	70,40	–	–	–	–
YOLOv11 R[2]	70,30	–	–	–	–
DeeplabV3_Resnet50 O[1]	72,12[5]	0,107	–	–	–
DeeplabV3_Resnet50 R[2]	72,14[5]	0,107	–	–	–
U-Net O[1]	–	–	0,923	–	–
U-Net R[2]	–	–	0,869	–	–

[1] O: Modelo original; [2] R: Modelo reconstruído; [3] *mshj2018factorized*; [4] *cheng2020anchor*; [5] APA: *Average Pixel Accuracy*.

Na Tabela 2 estão apresentados os resultados da análise de impacto de desempenho. A mesma foi feita utilizando diversas métricas para que fosse possível cobrir a grande variedade de modelos escolhidos, que desempenham tarefas muito diferentes entre si. É possível observar que as redes de classificação e detecção mantiveram desempenho estável após a compressão pelo NNCodec, sem perdas significativas de acurácia. Além disso, a rede Inception apresentou um leve aumento, devido ao efeito de regularização induzido pela compressão, que contribui na diminuição do *overfitting*. Esse comportamento também foi observado na rede DeeplabV3_Resnet50. O modelo U-Net apresentou uma perda de aproximadamente 5,85% na qualidade de suas predições, indicando uma piora na capacidade de segmentação. No modelo C.AI 2018, observou-se aumento do PSNR e do BPP, indicando maior qualidade da imagem reconstruída, porém com pior capacidade de compressão, resultando em imagens com *bitrate* mais elevado. No modelo C.AI 2020, o PSNR retornou como *NaN* devido à presença de valores inválidos na reconstrução das imagens, impossibilitando a determinação de métricas como o *Mean Square Error* (MSE) e o *Peak Value* (valor máximo do pixel), que são necessárias para o cálculo do PSNR.

4. Conclusão

Este trabalho apresentou uma análise de desempenho do software de referência NNCodec, avaliando sua capacidade de compressão e o impacto do processo no desempenho de diferentes modelos de DNNs, aplicados à tarefas de processamento de imagens.

Os resultados obtidos demonstraram que o NNCodec é capaz de reduzir significativamente o tamanho dos modelos, alcançando uma taxa de compressão média de, aproximadamente, 86,78%. De forma geral, a compressão não gerou grandes alterações no desempenho dos modelos, sendo o modelo C.AI 2020 e o U-Net os que apresentaram as maiores perdas. Além disso, foi possível obeservado que a compressão pode ajudar a reduzir *overfitting* das redes.

Referências

Becking, D. et al. (2023). Nncodec: An open source software implementation of the neural network coding iso/iec standard. In *Proceedings of the 40th International Conference on Machine Learning*.

Chen, L. et al. (2018). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848.

Cheng, Z. et al. (2020). Learned image compression with discretized gaussian mixture likelihoods and attention modules. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA. IEEE.

Islam, M., Chen, G., and Jin, S. (2019). An overview of neural network. *American Journal of Neural Networks and Applications*, 5(1):7.

Jocher, G. and Qiu, J. (2024). Ultralytics yolo11. `https://github.com/ultralytics/ultralytics`. Accessed: 2025-09-05.

Kirchhoffer, H. et al. (2021). Overview of the neural network compression and representation (nnr) standard. *IEEE Transactions on Circuits and Systems for Video Technology*, pages 1–1.

Krizhevsky, A. (2014). One weird trick for parallelizing convolutional neural networks. *arXiv preprint*.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *Lecture Notes in Computer Science*, volume 9351, pages 234–241. Springer.

Szegedy, C. et al. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA. IEEE.