Uma metodologia para a avaliação de desempenho e custos do treinamento de redes neurais em ambientes de nuvem

Cláudio Márcio de Araújo Moura Filho[1], Érica Teixeira Gomes de Souza[1]

[1]Departamento de Computação – Universidade Federal do Rural de Pernambuco (UFRPE)
Recife – Pernambuco – Brazil

{claudio.marciof,erica.sousa}@ufrpe.br

Abstract. *Deep neural networks are solutions to problems involving pattern recognition and several works try to find ways to optimize the performance of these networks. This optimization requires suitable hardware to be implemented, hardware that can be very expensive for small and medium-sized organizations. The objective of this work is to propose a methodology to evaluate the performance and cost of training convolutional neural networks, considering the factors that most impact training time and evaluate the total financial cost of the environment for this task. In this sense, it was observed that factors such as the size of the input image and the network architecture have a great impact on the training time metric and consequently on the total cost.*

Resumo. *Redes neurais profundas são soluções para problemas que envolvem reconhecimento de padrões e diversos trabalhos tentam encontrar maneiras de otimizar o desempenho dessas redes. Essa otimização necessita de hardware adequado para ser implementada, hardware esse que pode ser muito custoso para pequenas e médias organizações. O objetivo deste trabalho é propor uma metodologia para avaliar o desempenho e custo do treinamento de redes neurais convolucionais, considerando os fatores mais impactantes no tempo de treinamento e avaliar o custo financeiro total do ambiente para essa tarefa. Nesse sentido, observou-se que fatores como o tamanho da imagem de entrada e a arquitetura da rede tem grande impacto na métrica de tempo de treinamento e por consequência no custo total.*

1. Introdução

As Redes neurais profundas são potenciais soluções para uma vasta gama de problemas que envolvem reconhecimento de padrões, classificação e predição. Para se tornar uma solução confiável, estas *DNNs* passam por um processo de treinamento para entender os padrões dos dados e aprender a classificar corretamente a informação que será fornecida em momento posterior [LECUN 2015].

Esse processo de treinamento pode ser muito custoso, visto que para obter resultados satisfatórios faz-se necessário processar um grande volume de dados, isso significa que um dos maiores desafios na criação de modelos de *DNNs* é justamente o tempo necessário para treinar esse modelo para resolver o problema [LECUN 2015].

Uma das alternativas para se acelerar o processo de treinamento de uma *DNN* é utilizar hardware especializado para processamento de dados, como é o caso das *Graphics Processing Units*(*GPUs*). Entretanto, essas unidades de

processamento possuem um custo de aquisição consideravelmente mais alto que os processadores convencionais, as *Central Processing Units* (*CPUs*) [LECUN 2015].

A computação em nuvem surge como uma alternativa para reduzir o alto custo de uma *GPU*. Com a computação em nuvem é possível construir um ambiente computacional para todos os tipos de tarefas sem de fato ter a máquina física e pagando apenas pelo que é utilizado [KANSAL 2014].

Estes ambientes construídos sob-medida podem ser muito promissores para se construir soluções com redes neurais uma vez que podemos ter todo o recurso necessário para uma execução eficiente do treinamento com um custo baseado na utilização de recursos [JUVE 2010][JACKSON 2010].

Devido a necessidade de desenvolver soluções com redes neurais profundas de maneira rápida e da necessidade de um ambiente adequado para desenvolver as mesmas, este trabalho dedica-se a propor uma metodologia para avaliar o treinamento de redes neurais em ambientes de nuvem, avaliando o desempenho e custo do treinamento das redes nesse ambiente.

Os seguintes assuntos serão abordados nas próximas seções: Os trabalhos relacionados serão mostrados na Seção 2 e a metodologia será apresentada na Seção 3. Por fim, as Seções 4 e 5 são destinadas a discutir os resultados dos experimentos realizados e a conclusão deste trabalho, respectivamente.

2. Trabalhos relacionados

O tema redes neurais profundas é de grande interesse para pesquisas principalmente na área de desempenho, porém a maioria das pesquisas está relacionada ao desempenho das *DNNs* em relação a sua precisão em classificar e/ou reconhecer objetos[ZHU 2018], ou seja, quão bem as redes neurais conseguem resolver um problema.

Os ambientes de nuvem por sua vez, já vem sendo abordados em pesquisas na área de computação de alto desempenho como pode ser visto em [JUVE 2010] e [JACKSON 2010], que utilizam as máquinas virtuais da AWS para avaliar o desempenho das mesmas executando tarefas de alta performance, além de [CARNEIRO 2018], que utiliza o ambiente do *Google Colaboratory* para acelerar o treinamento de redes neurais. Nestes trabalhos os autores focam em entender como utilizar a computação em nuvem para executar tarefas que exigem alto poder computacional, como é o caso do treinamento de redes neurais.

Os autores de [ZHU 2018] e [ELSHAWI 2021] focam os seus esforços em entender os mecanismos para avaliar o desempenho das redes neurais considerando diversos fatores como o dataset utilizado, a tarefa que a rede neural exercerá, o tempo de treinamento, o framework no qual a rede é construída e por fim os recursos do ambiente. Esses dois trabalhos propõem modelos de *benchmark* para avaliar o desempenho de redes neurais definindo vários parâmetros como uso de recursos e fazem uma análise desse desempenho levando em consideração os *frameworks* e *datasets* utilizados.

Os trabalhos [SHI 2016], [WU 2019], [LIU 2018], [SHAMS 2017] e [XIE 2022] buscam realizar uma experimentação variando os possíveis fatores que mais impactam o desempenho de uma rede neural. De modo a entender o comportamento das redes perante o treinamento para melhorar a eficácia da solução da rede neural. Observa-se nesses trabalhos o uso recorrente de frameworks como o Tensorflow e o MXNet e de datasets como o MNIST e CIFAR-10, além de realizarem diversos testes em diferentes tipos de ambiente com e sem GPUs.

Nos trabalhos [ADEBAYO; NONSINDISO MANGANYELA;ADIGUN 2020], [EDINAT 2018], [MEETU KANDPAL; GAHLAWAT; PATEL 2017] e [WU; BUYYA; RAMAMOHANARAO 2020], os autores buscam sintetizar as informações sobre como os serviços das nuvens são precificados e buscam responder a questão de como precificar adequadamente os serviços da nuvem mantendo preços justos tanto para os usuários quanto para os provedores. Nota-se um tendência a considerar o modelo dinâmico como o modelo mais justo para precificação, pois permite que o usuário pague apenas pela quantidade de recursos que aloca e utiliza, além de habilitar o provedor a ofertar conjuntos de recursos de maneira a cobrir adequadamente os seus custos.

Diante do exposto até aqui, a proposta deste trabalho, e o que o diferencia de outros trabalhos já existentes, é a proposição de uma metodologia para avaliar o desempenho do treinamento de redes neurais em ambientes de nuvens públicas, considerando o custo para treinar a rede neural nesse tipo de ambiente como uma métrica de desempenho.

3. Metodologia para Avaliação de Desempenho de Redes Neurais em Ambientes de Nuvem

A metodologia proposta tem o objetivo de apresentar atividades que permitam a avaliação de desempenho de redes neurais em ambientes de nuvens, conforme a Figura 1. Essa metodologia é composta de cinco atividades, são elas: Entendimento do Ambiente, Planejamento de Experimentos, Configuração do Ambiente, Medição e Análise de Métricas. Nesta seção o objetivo de cada uma dessas atividades será melhor detalhado.

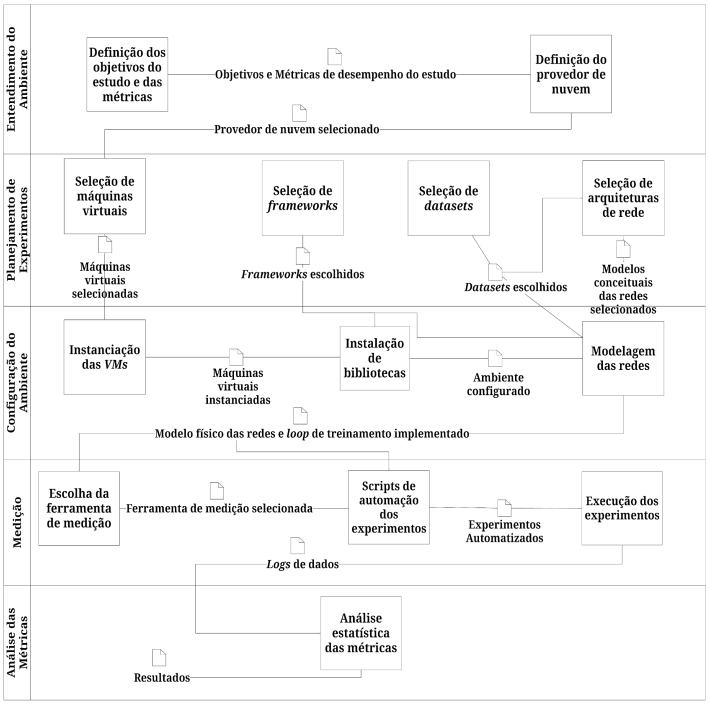

Figura 1 – Metodologia empregada

3.1 Entendimento do Ambiente

Para o entendimento do ambiente, primeiramente é preciso ter em mente qual é o objetivo que queremos atingir. O objetivo em questão é analisar o treinamento de redes neurais em um ambiente de nuvem de uma perspectiva tanto de desempenho quanto de custo financeiro.

Com a definição dos objetivos, as métricas de desempenho e custo podem ser selecionadas. As métricas de desempenho podem ser utilização de recursos, tempo de execução e vazão, já a métrica de custo pode ser os custos totais dos recursos usados.

O provedor de serviços de nuvem pode ser escolhido, ou seja, quem proverá os ambientes onde serão executados os experimentos para coletas de dados. Dessa forma, *Amazon Web Services*(*AWS*), *Microsoft Azure* e a *Google Cloud Platform são provedores que podem ser adotados*.

3.2 Planejamento de experimentos

A atividade de planejamento de experimentos tem o objetivo de definir os fatores e seus níveis para a definição dos experimentos de desempenho e custo.

Para executar os experimentos é importante definir os tipos de máquinas virtuais adotadas nos experimentos. As máquinas virtuais podem ser instanciadas apenas com *CPU* e com *CPU* e *GPU*.

Os *Frameworks* são bibliotecas de código que auxiliam no desenvolvimento acelerado de aplicações, no caso das redes neurais eles fornecem todo o arcabouço necessário para modelar as redes e aplicar as funções de treinamento e otimização nas

mesmas. Há muitos *frameworks*, como por exemplo, *Caffe*, *Chainer*, *CNTK*, *TensorFlow*, *PyTorch* disponíveis para utilização, portanto selecionar os mais interessantes para uma análise detalhada é uma tarefa necessária.

Outro fator importante para o treinamento de redes neurais é o conjunto de dados que será utilizado, pois é ele quem define o problema que a rede neural irá resolver. *Datasets* de boa qualidade idealmente possuem uma grande quantidade de imagens bem distribuídas entre as classes, porém a depender do problema esse balanceamento dos dados não é possível.

Para este trabalho o foco não é resolver um problema em específico utilizando redes neurais, portanto podem ser usados os *datasets* conhecidos como *datasets* de *benchmark*, como MNIST, CIFAR-10 e CIFAR-100, ou seja, conjuntos de imagens utilizadas para testar desempenho de modelos.

Definidos os *datasets* e *frameworks* é necessário agora definir um modelo de rede neural para treinar, esse modelo deve ser complexo o suficiente para ser analisado porém descomplicado a ponto de o treinamento não demandar muito tempo e ser possível coletar dados, sendo o ideal criar um modelo próprio utilizando técnicas utilizadas em redes neurais conhecidas como *LeNet*, *AlexNet* e *VGG*.

3.3 Configuração do ambiente

Para executar o treinamento das redes neurais é importante configurar o ambiente de maneira correta com as bibliotecas necessárias. Esta atividade está relacionada com a criação de máquinas virtuais, instalação das bibliotecas necessárias para executar o treinamento das redes neurais e a criação dos modelos das redes neurais.

3.4 Medição

Nesta atividade o foco é selecionar uma ferramenta adequada para monitorar os experimentos e executar os mesmos. Para selecionar a ferramenta é necessário entender quais métricas serão utilizadas, dentre várias possíveis, métricas como utilização de recursos e tempo de treinamento foram selecionadas para este trabalho.

3.5 Análise de Métricas

Essa atividade tem o objetivo de realizar a análise estatística dos dados coletados nos experimentos. Essa análise dá-se através do resumo estatístico dos dados com a apresentação das médias das métricas de desempenho e custo.

4. Resultados e Discussão

O objetivo dessa seção é avaliar se a metodologia proposta pode ser aplicada na avaliação de desempenho e custo de redes neurais em ambientes de nuvem. Dessa forma um estudo de caso é apresentado.

4.1 Entendimento do Ambiente

Seguindo as atividades propostas na metodologia primeiro foram selecionadas as métricas do estudo. As métricas definidas para serem coletadas foram: tempo de treinamentos, utilização de *CPU*, utilização de memória, utilização de *GPU* e utilização da memória da *GPU*, sendo a primeira métrica, medida em segundos(s), e as 4 conseguintes todas em porcentagem (%).

Logo após a seleção das métricas, um provedor de nuvem precisa ser selecionado. O provedor de nuvem escolhido foi a *Microsoft Azure* devido a oferta de máquinas virtuais com placas gráficas.

4.2 Planejamento de experimentos

Os fatores dos experimentos são o framework, o dataset, a dimensão da imagem, a arquitetura da rede e o ambiente, cada um desses fatores terá diferentes variações que serão chamados de níveis. Esses fatores e seus níveis podem ser vistos na Tabela 1 e serão melhor discutidos adiante.

Tabela 1 - Fatores e Níveis dos experimentos

Fator	Nível 1	Nível 2	Nível 3
Framework	*MXNet*	*PyTorch*	*TensorFlow*
Dataset	*MNIST*	*CIFAR-10*	-
Dimensão da Imagem	32x32	64x64	-
Arquitetura da rede	Rede 1	Rede 2	-
Ambiente	*CPU*	*GPU*	-

4.2.1 Máquinas virtuais

As máquinas virtuais selecionadas são usadas como ambiente para o treinamento das redes neurais. Na Tabela 2 é possível visualizar as configurações das máquinas virtuais adotadas.

Tabela 2 - Configuração das máquinas virtuais

Nome *VM*	*vCPUs*	*RAM*	*GPU*	SO	Custo/hora
F4s_v2	4	8GB	-	Ubuntu 22.04	U$0,169
NC4as_T4_v3	4	28GB	*NVIDIA T4 16GB*	Ubuntu 22.04	U$0,526

4.2.2 Frameworks

Os *frameworks* escolhidos foram o *PyTorch, MXNet e TensorFlow*, por uma questão de usabilidade, uma vez que os mesmos são muito utilizados em projetos acadêmicos e na indústria. Esses frameworks permitem a construção de modelos de maneira rápida e simplificada, além da modelagem das redes, também é possível criar o loop de treino das redes de maneira igualmente simples.

4.2.3 Datasets

Os *datasets* selecionados foram o *MNIST* [LECUN e CORTES 2005], uma base de imagens dos algarismos manuscritos em preto em branco e o *CIFAR-10* [KRIZHEVSKY 2009], uma base de imagens coloridas com 10 classes diferentes que mistura animais e tipos de veículos. Esses *datasets* são muito utilizados por outros trabalhos na área como mostrado na seção de trabalhos relacionados e por esse motivo

foram escolhidos.

4.2.3 Redes Neurais

Com esses fatores em mente, a arquitetura de rede escolhida foi algo semelhante ao que é feito nas redes *VGG* [SIMONYAN e ZISSERMAN 2014], que mescla camadas de convolução, com funções de ativação e camadas de *pooling* e por fim camadas completamente conectadas. A arquitetura das redes podem ser vistas nas Figuras 2 e 3 abaixo.

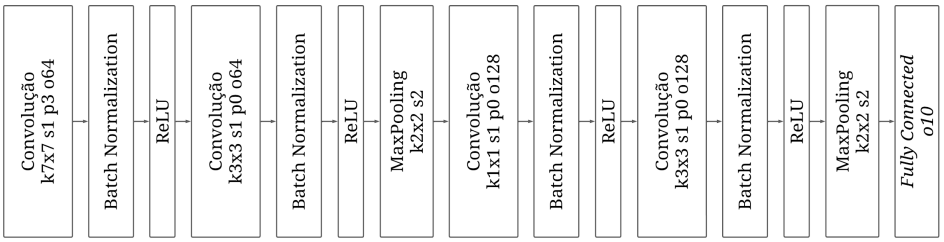

Figura 2 – Arquitetura da Rede 1

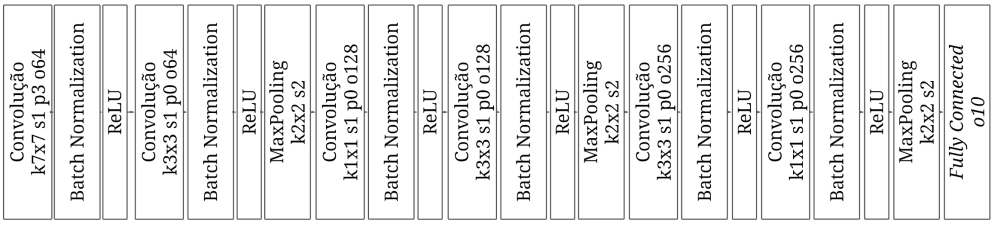

Figura 3 – Arquitetura da Rede 2

Nas figuras podemos ver algumas letras nas camadas, o **k** significa *kernel* e representa o tamanho da janela que a camada vai observar na imagem a cada passo. O **s** representa o *stride*, quantos *pixels* serão andados pela janela do *kernel* a cada interação na imagem. O **p** refere-se ao *padding*, quantos *pixels* serão adicionados nas bordas da imagem. Por fim, o **o** significa a quantidade de canais de saída da camada, ou seja o *output* da camada.

4.3 Configuração do ambiente

Para a configuração do ambiente é necessário a instalação de bibliotecas para executar o código do treinamento da rede neural. Seguem as versões utilizadas das principais bibliotecas. A versão do *TensorFlow* foi a 2.14.0, já para o *PyTorch* a versão utilizada foi a 2.2.0 por último a versão do *MXNet* foi a versão 1.9.1.

Além das versões dos *frameworks*, outra instalação necessária é a das bibliotecas para a utilização da *GPU*, como a placa gráfica em questão é uma placa *NVIDIA*, as bibliotecas necessárias são o *CUDA Toolkit* e o *cuDNN*. A versão do *CUDA* utilizada é a v11.7 e a do cuDNN 8.6.0.

4.4 Medição

Para coletar os dados das métricas foi utilizado um *script* que é executado paralelamente ao treinamento da rede neural e que a cada 10 segundos captura e registra as métricas em um arquivo de *log* com valores separados por vírgula. Além desse *script*, alguns comandos no *loop* de treinamento marcam o tempo de treinamento de cada época.

Para garantir a qualidade dos dados dos experimentos, cada treinamento foi executado 5 vezes, então os 48 cenários iniciais se tornaram 240 execuções, variando os diferentes níveis dos fatores definidos.

Cada experimento consiste em executar o treinamento de uma rede neural por um total de 10 épocas, ou seja, todas as imagens dos conjuntos de dados foram passadas para as redes um total de 10 vezes.

Além do número de épocas, o tamanho de *batch*, ou lote em português, foi fixado em 128 exemplos por iteração, isso quer dizer que a rede é alimentada com 128 imagens por vez, até que todas as imagens tenham sido passadas pela rede.

A Tabela 3 mostra a combinação dos níveis dos fatores formando cenários numerados para melhorar a identificação dos mesmos.

Tabela 3 - Cenários dos experimentos.

CPU						GPU				
Cenário	***Framework***	***Dataset***	**Dim. da Img**	**Arq. da rede**		**Cenário**	***Framework***	***Dataset***	**Dim. da Img**	**Arq. da rede**
C1	*MXNet*	*MNIST*	32x32	Rede 1		C25	*MXNet*	*MNIST*	32x32	Rede 1
C2	*MXNet*	*MNIST*	32x32	Rede 2		C26	*MXNet*	*MNIST*	32x32	Rede 2
C3	*MXNet*	*MNIST*	64x64	Rede 1		C27	*MXNet*	*MNIST*	64x64	Rede 1
C4	*MXNet*	*MNIST*	64x64	Rede 2		C28	*MXNet*	*MNIST*	64x64	Rede 2
C5	*MXNet*	*CIFAR-10*	32x32	Rede 1		C29	*MXNet*	*CIFAR-10*	32x32	Rede 1
C6	*MXNet*	*CIFAR-10*	32x32	Rede 2		C30	*MXNet*	*CIFAR-10*	32x32	Rede 2
C7	*MXNet*	*CIFAR-10*	64x64	Rede 1		C31	*MXNet*	*CIFAR-10*	64x64	Rede 1
C8	*MXNet*	*CIFAR-10*	64x64	Rede 2		C32	*MXNet*	*CIFAR-10*	64x64	Rede 2
C9	*PyTorch*	*MNIST*	32x32	Rede 1		C33	*PyTorch*	*MNIST*	32x32	Rede 1
C10	*PyTorch*	*MNIST*	32x32	Rede 2		C34	*PyTorch*	*MNIST*	32x32	Rede 2
C11	*PyTorch*	*MNIST*	64x64	Rede 1		C35	*PyTorch*	*MNIST*	64x64	Rede 1
C12	*PyTorch*	*MNIST*	64x64	Rede 2		C36	*PyTorch*	*MNIST*	64x64	Rede 2
C13	*PyTorch*	*CIFAR-10*	32x32	Rede 1		C37	*PyTorch*	*CIFAR-10*	32x32	Rede 1
C14	*PyTorch*	*CIFAR-10*	32x32	Rede 2		C38	*PyTorch*	*CIFAR-10*	32x32	Rede 2
C15	*PyTorch*	*CIFAR-10*	64x64	Rede 1		C39	*PyTorch*	*CIFAR-10*	64x64	Rede 1
C16	*PyTorch*	*CIFAR-10*	64x64	Rede 2		C40	*PyTorch*	*CIFAR-10*	64x64	Rede 2
C17	*TensorFlow*	*MNIST*	32x32	Rede 1		C41	*TensorFlow*	*MNIST*	32x32	Rede 1
C18	*TensorFlow*	*MNIST*	32x32	Rede 2		C42	*TensorFlow*	*MNIST*	32x32	Rede 2
C19	*TensorFlow*	*MNIST*	64x64	Rede 1		C43	*TensorFlow*	*MNIST*	64x64	Rede 1
C20	*TensorFlow*	*MNIST*	64x64	Rede 2		C44	*TensorFlow*	*MNIST*	64x64	Rede 2
C21	*TensorFlow*	*CIFAR-10*	32x32	Rede 1		C45	*TensorFlow*	*CIFAR-10*	32x32	Rede 1
C22	*TensorFlow*	*CIFAR-10*	32x32	Rede 2		C46	*TensorFlow*	*CIFAR-10*	32x32	Rede 2
C23	*TensorFlow*	*CIFAR-10*	64x64	Rede 1		C47	*TensorFlow*	*CIFAR-10*	64x64	Rede 1
C24	*TensorFlow*	*CIFAR-10*	64x64	Rede 2		C48	*TensorFlow*	*CIFAR-10*	64x64	Rede 2

A seguir são apresentadas figuras (Figuras 2 a 5) com gráficos das médias das métricas coletadas, através da execução do treinamento da rede neural dos 48 cenários anteriormente mencionados.

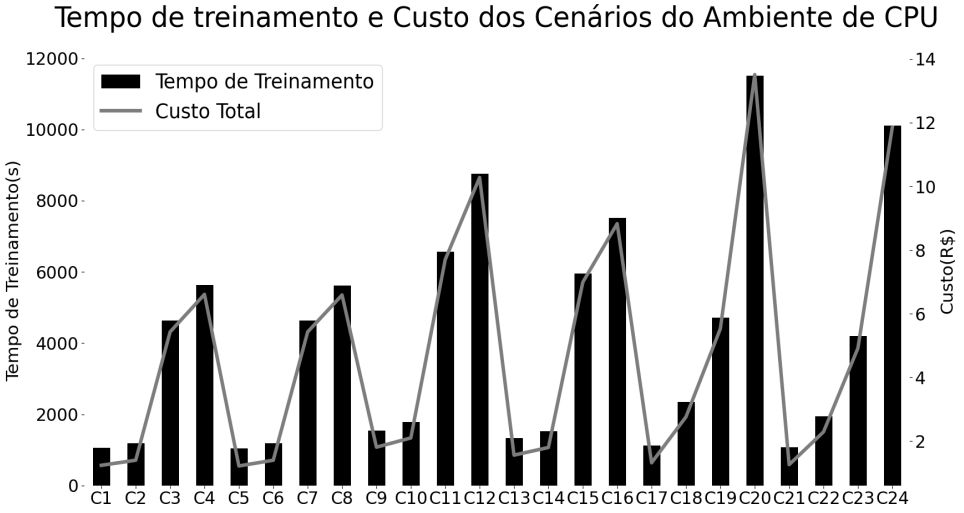

Figura 4 - Gráfico das médias de tempo de treinamento e custo do ambiente *CPU*.

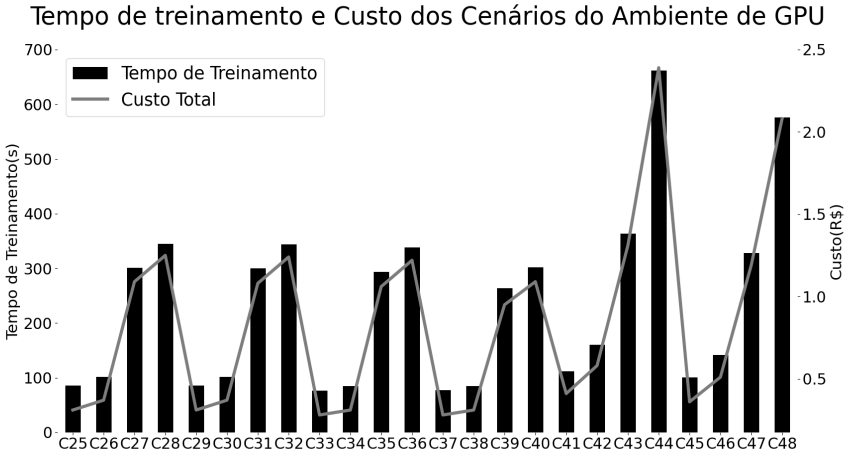

Figura 5 - Gráfico das médias de tempo de treinamento e custo do ambiente *GPU*.

Os resultados apresentados nas Figuras 4 e 5 mostram que quanto maior a imagem maior é o tempo de treinamento da rede neural. É possível observar uma semelhança nos gráficos ao compararmos as execuções nos ambientes com *CPU* x *GPU*, os cenários com imagens 64x64 aumentam em cerca de 200% o tempo de treinamento em relação às imagens 32x32.

A complexidade da rede escolhida também impacta o tempo de treinamento, é possível notar isso nos gráficos onde os cenários utilizam a Rede 2, esses cenários requerem mais tempo para treinar a rede.

O ambiente de *GPU* é pelo menos 10 vezes mais rápido que o ambiente de *CPU*, e mesmo com o valor do custo/hora do ambiente com *GPU* sendo o triplo do cenário com *CPU*. O custo total do ambiente de *CPU* foi cerca de 4 a 6 vezes menor que o do ambiente de *GPU*.

Portanto, é necessário pensar bem nas dimensões da imagem de entrada e na

arquitetura da rede utilizada, pois elas têm grande influência no tempo total de treinamento da rede neural e consequentemente no custo total.

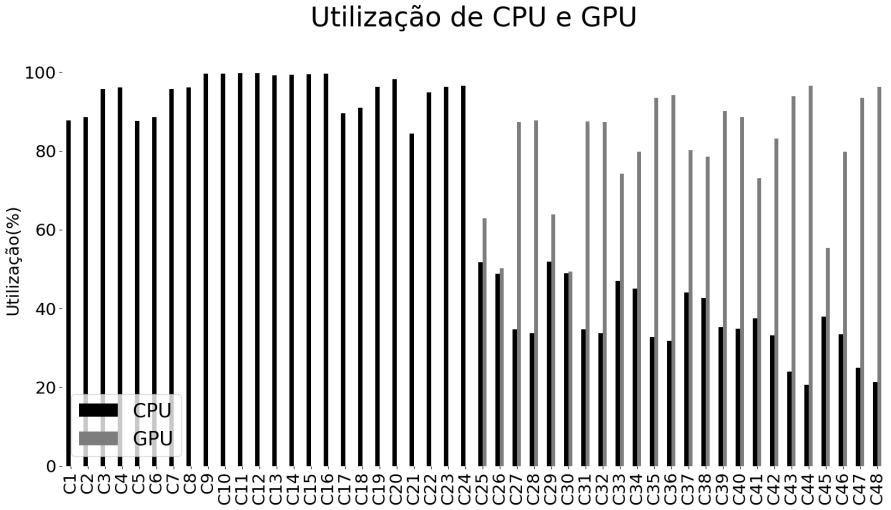

Figura 6- Gráfico das médias de utilização de *CPU* e *GPU*

Considerando a Figura 6, observa-se que todos os frameworks fazem boa utilização dos recursos de processamento, sendo o *PyTorch* o *framework* que utiliza mais recursos da *CPU* com valores sempre próximos a 100%. Quanto aos cenários com *GPU* nota-se que os cenários com imagens 64x64 tendem a consumir 2 vezes mais da *GPU* em relação a *CPU*.

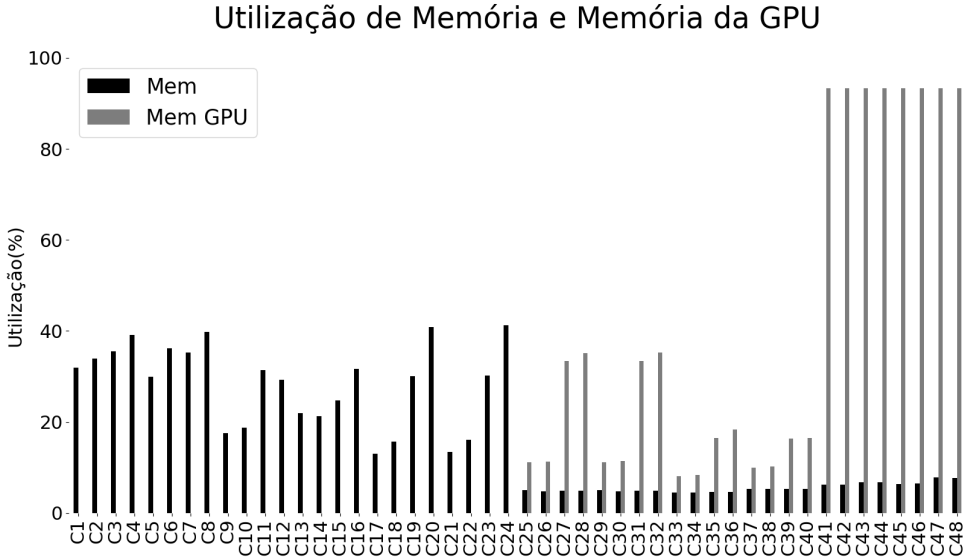

Figura 7 - Gráfico das médias de utilização de memória e memória da *GPU*

A Figura 7 mostra que o *TensorFlow* é o que mais utiliza a memória da *GPU* em todos os cenários, tendendo a 100% de utilização, entretanto o mesmo *framework* apresenta desempenho de tempo de treinamento de 2 a 3 vezes maior que os outros ao qual é comparado.

Sobre o *PyTorch* é possível notar um desempenho que o coloca entre o *MXNet* e o *Tensorflow* no quesito tempo de treinamento. Além disso, consome ao máximo a

CPU, utilizando cerca de 40% de memória e utiliza pouca memória da *GPU*, consumindo cerca de 20% apesar de utilizar mais de 80% da *GPU*.

O *dataset* não afeta as métricas em geral, contudo é possível perceber que os cenários com *CIFAR-10* possuíram tempos de treinamento menores quando comparados aos cenários que utilizaram o *MNIST*.

Por fim, em relação aos custos, o valor total gasto indicado pelos painel de gerenciamento de custos da *Microsoft Azure* foi de aproximadamente de R$158,00, com a soma dos custos.

5. Conclusão

Este trabalho apresenta uma metodologia para avaliação de desempenho de redes neurais em ambientes de nuvem baseada na técnica de planejamento de experimentos. Além disso, o trabalho apresenta um estudo de caso que coletou dados de máquinas virtuais providas pela *Microsoft Azure* ao treinar redes neurais.

Observa-se no estudo, que a nuvem demonstra ter grande potencial em oferecer um ambiente com aceleradores(*GPUs*) a um baixo custo, permitindo o desenvolvimento de redes neurais 10 vezes mais rápido, em comparação a ambientes sem esses aceleradores. Além do custo reduzido, esse ambiente pode ser bem aproveitado pelos diversos *frameworks* existentes como demonstrado nas métricas coletadas no estudo, no entanto a maior utilização de recursos de CPU, GPU e memória parece não afetar o tempo de treinamento de maneira consistente, portanto uma investigação variando a quantidade de recursos disponível se faz necessária.

Outros aspectos, como o estudo deste treinamento das redes com paralelização de máquinas não foram abordados e podem ser encarados como objeto de estudo em trabalhos futuros.

Referências

ADEBAYO, I. O.; NONSINDISO MANGANYELA; ADIGUN, M. O. Cost-Benefit Analysis of Pricing Models in Cloudlets. 25 nov. 2020.

CARNEIRO, T. et al. Performance Analysis of Google Colaboratory as a Tool for Accelerating Deep Learning Applications. IEEE Access, v. 6, p. 61677–61685, 2018.

CORREIA, Fernando. Definição de computação em nuvem segundo o NIST. Plataforma Nuvem, 2011. Disponível em: https://plataformanuvem.wordpress.com/2011/11/21/definicao-de-computacao-em-nuvem-segundo-o-nist/. Acesso em: 02/07/2023.

EDINAT A. Cloud Computing Pricing Models: A Survey. International Journal of Scientific Engineering and Research (IJSER). 2018.

ELSHAWI, R. et al. DLBench: a comprehensive experimental evaluation of deep learning frameworks. Cluster Computing, v. 24, n. 3, p. 2017–2038, 7 fev. 2021.

JACKSON, K. R. et al. Performance Analysis of High Performance Computing Applications on the Amazon Web Services Cloud. 2010 IEEE Second International

Conference on Cloud Computing Technology and Science, nov. 2010.

JUVE, G. et al. Scientific workflow applications on Amazon EC2. 16 maio 2010.

KANSAL, S. et al. Pricing Models in Cloud Computing. Proceedings of the 2014 International Conference on Information and Communication Technology for Competitive Strategies, 27 out. 2014.

KRIZHEVSKY, A. 2009. CIFAR-10 and CIFAR-100 datasets. Disponível em: <https://www.cs.toronto.edu/~kriz/cifar.html>. Acesso em: 20/02/2024.

LECUN, Y., e CORTES, C. (2005). The mnist database of handwritten digits.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep Learning. Nature, v. 521, n. 7553, p. 436–444, maio 2015.

LILJA, D. J. Measuring Computer Performance: a practitioner's guide. [S.l.]: Cambridge University Press, 2005. 278p.

LIU, J. et al. "Usability Study of Distributed Deep Learning Frameworks For Convolutional Neural Networks." 2018.

MEETU KANDPAL; GAHLAWAT, M.; PATEL, K. R. Role of predictive modeling in cloud services pricing: A survey. 1 jan. 2017.

MENASCÉ, D. A.; ALMEIDA, V. A. F. Performance by Design: computer capacity planning by example. [S.l.]: Prentice Hall PTR, 2005. 462p.

OpenAI Five. 2018. Disponível em: <https://openai.com/research/openai-five>. Acesso em: 20/02/2024.

SHAMS, S. et al. Evaluation of Deep Learning Frameworks Over Different HPC Architectures. 1 jun. 2017.

SHI, S. et al. Benchmarking State-of-the-Art Deep Learning Software Tools. arXiv (Cornell University), 25 ago. 2016.

SIMONYAN, K. e ZISSERMAN, A. "Very Deep Convolutional Networks for Large-Scale Image Recognition." CoRR abs/1409.1556 (2014): n. pag.

WU, C.; BUYYA, R.; RAMAMOHANARAO, K. Cloud Pricing Models. ACM Computing Surveys, v. 52, n. 6, p. 1–36, 21 jan. 2020.

WU, Y. et al. A Comparative Measurement Study of Deep Learning as a Service Framework. IEEE Transactions on Services Computing, p. 1–1, 2019.

XIE, X. et al. Performance Evaluation and Analysis of Deep Learning Frameworks. 23 set. 2022.

ZHU, H. et al. TBD: Benchmarking and Analyzing Deep Neural Network Training. arXiv (Cornell University), 16 mar. 2018.