Dados Internacionais de Catalogação na Publicação
Universidade Federal Rural de Pernambuco
Sistema Integrado de Bibliotecas
Gerada automaticamente, mediante os dados fornecidos pelo(a) autor(a)

M929m Moura Filho, Cláudio Márcio de Araújo
Uma metodologia para a avaliação de desempenho e custos do treinamento de redes neurais em ambientes de nuvem / Cláudio Márcio de Araújo Moura Filho. - 2024.
16 f. : il.

Orientadora: Erica Sousa.
Inclui referências.

Trabalho de Conclusão de Curso (Graduação) - Universidade Federal Rural de Pernambuco, Bacharelado em Ciência da Computação, Recife, 2024.

1. avaliação de desempenho. 2. computação em nuvem. 3. redes neurais. 4. custo financeiro. 5. metodologia. I. Sousa, Erica, orient. II. Título

CDD 004

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 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, 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, em inglês, *Deep Neural Networks* (*DNNs*), 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 as *Graphics Processing Units*(*GPUs*). Entretanto, essas unidades de processamento possuem um custo de aquisição 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: conceitos básicos na Seção 2. Os trabalhos relacionados serão mostrados na Seção 3 e a metodologia será apresentada na Seção 4. Por fim, as Seções 5 e 6 são destinadas a discutir os resultados dos experimentos realizados e a conclusão deste trabalho, respectivamente.

2. Conceitos básicos

Para melhor compreensão deste trabalho, alguns conceitos básicos acerca de computação em nuvem, redes neurais e avaliação de desempenho serão apresentados nesta seção.

Segundo o *National Institute of Standards and Technologies*(*NIST*), computação em nuvem é um modelo que habilita o acesso de maneira universal a um conjunto compartilhado de recursos (processadores, memória, disco, etc.) que podem ser provisionados com o mínimo de esforço de gerenciamento ou interação com o provedor de serviços [CORREIA 2011]. De maneira geral, os modelos de precificação para serviços de nuvem são os modelos de pagamento estático, ou fixo, e os modelos de pagamento dinâmico [WU;BUYYA;RAMAMOHANARAO 2020].

A avaliação de desempenho de sistemas computacionais consiste de um conjunto de técnicas classificadas como as baseadas em medição e as baseadas em modelagem. Essas técnicas de avaliação de desempenho permitem o planejamento de infraestruturas de nuvem conforme o tipo e volume da carga de trabalho[LILJA 2005][MENASCÉ; ALMEIDA 2005].

As *Deep Neural Networks* mais conhecidas pela sigla *DNNs* são uma classe de algoritmos de inteligência artificial capazes de resolver problemas complexos que envolvem reconhecimento de padrões, classificação e predição [LECUN 2015].

As *DNNs* adotam o algoritmo de *backpropagation* para se adaptar aos dados que passam pelas suas múltiplas camadas de processamento e, dessa maneira, elas podem aprender a interpretar os dados e consequentemente tornam-se capazes de encontrar soluções para os problemas como, reconhecimento de discurso, reconhecimento de objetos em imagens entre outras diversas possibilidades [LECUN 2015].

Redes neurais convolutivas, ou *Convolutional Neural Networks* (*CNNs*) em inglês, são um tipo especial de rede neural profunda que é capaz de lidar com imagens usadas principalmente para reconhecer objetos dentro de imagens e classificar a imagem ou destacar o objeto encontrado na mesma [LECUN 2015].

Uma aplicação importante das *CNNs* é no campo da medicina, onde as redes neurais podem ser utilizadas para identificar células cancerígenas em imagens. As redes neurais conseguem predizer corretamente mais de 90% das vezes quando uma imagem possui ou não o padrão de células cancerígenas [JAIN 2022].

Além do reconhecimento de imagens, as redes neurais também podem ser utilizadas para reconhecimento de discurso, como é o caso das redes neurais utilizadas para processamento de linguagem natural. O caso de maior destaque recente é o *ChatGPT*, uma ferramenta capaz de reconhecer entradas do usuário e fornecer respostas contextualizadas de acordo com as perguntas feitas para ela [YAZAN ALAHMED 2023].

Dessa forma, o planejamento de uma infraestrutura que permita o treinamento de redes neurais possibilita o rápido desenvolvimento de soluções eficazes para os diversos tipos de problemas que as mesmas se dispõe a resolver.

3. 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], [JACKSON 2010] e [CARNEIRO 2018]. 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.

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.

Outro ponto em comum entre os trabalhos mencionados é o fato de que os experimentos levam em conta diversos frameworks para redes neurais como *Caffe*, *TensorFlow*, *Torch* e *MXNet*, além das configurações de ambientes físicos, com exceção de [LIU 2018] onde os autores utilizam máquinas virtuais da *Amazon Web Services* (*AWS*) como ambiente computacional.

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.

Tabela 1 – Resumo dos trabalhos relacionados a redes neurais

Trabalho	***Datasets***	***Frameworks***	**Métricas**	**Modelo de Precificação**
JUVE 2010	-	-	TEMPO DE EXECUÇÃO DA TAREFA.	-
JACKSON 2010	-	-	TEMPO DE EXECUÇÃO DA TAREFA E *SUSTAINED SYSTEM PERFORMANCE.*	-
SHI 2016	MNIST e CIFAR-10.	Caffe, Torch, TensorFlow, MXNet e CNTK.	TEMPO/*BATCH.*	-
SHAMS 2017	ILSVRC 2012, CIFAR-10 e MNIST.	Caffe, Apache SINGA e TensorFlow.	TEMPO DE EXECUÇÃO E IMAGENS/MILISEGUNDO	-
WU 2019	MNIST, CIFAR-10 e ILSVRC 2012.	Caffe, Torch, TensorFlow e Theano.	TEMPO DE TREINAMENTO, TEMPO DE TESTE, ACURÁCIA, UTILIZAÇÃO DE CPU E UTILIZAÇÃO DE MEMÓRIA	-
LIU 2018	CIFAR-10 e ImageNet.	Caffe2, Chainer, TensorFlow, MXNet e CNTK.	ACURÁCIA, TEMPO DE TREINAMENTO IMAGENS/SEGUNDO	-
CARNEIRO 2018	MS-COCO.	TensorFlow.	TEMPO DE EXECUÇÃO E AVALIAÇÕES/SEGUNDO	-
ZHU 2018	ImageNet1K, IWSLT15, VOC 2007 , LibriSpeech, Downsampled ImageNet e Atari 2006.	TensorFlow, MXNet e CNTK.	VAZÃO, UTILIZAÇÃO DE OPERAÇÕES DE PONTO FLUTUANTE, UTILIZAÇÃO DE CPU, UTILIZAÇÃO DE MEMÓRIA E UTILIZAÇÃO DE GPU	-

ELSHAWI 2021	MNIST, CIFAR-10, CIFAR-100, SVHN, IMDB Reviews, Penn Treebank, Many things: English to Spanish e VOC 2012.	TensorFlow, Keras, PyTorch, MXNet, Theano e Chainer.	TEMPO DE TREINAMENTO, ACURÁCIA. UTILIZAÇÃO DE CPU, UTILIZAÇÃO DE MEMÓRIA E UTILIZAÇÃO DE GPU	-
XIE 2022	MNIST, CIFAR-10 e Flower Recognition.	TensorFlow, Pytorch e PaddlePaddle.	TEMPO DE TREINAMENTO, ACURÁCIA, UTILIZAÇÃO DE CPU, UTILIZAÇÃO DE GPU, UTILIZAÇÃO DE MEMÓRIA.	-
MEETU KANDPAL; GAHLAWAT; PATEL 2017	-	-	-	Estático, Dinâmico.
EDINAT 2018	-	-	-	Estático, Dinâmico.
ADEBAYO; NONSINDISO MANGANYELA; ADIGUN 2020	-	-	-	Estático, Dinâmico e Híbrido.
WU; BUYYA; RAMAMOHANARAO 2020	-	-	-	Baseados em Custo, Valor e Mercado(Dinâmico)

Como é possível observar a partir da Tabela 1, dos 11 trabalhos sobre avaliação de desempenho de redes neurais, 6 deles utilizam os *datasets* de *benchmark CIFAR-10* e 5 desses também utilizam o *MNIST*. Em relação aos *frameworks*, 8 dos 11 trabalhos utilizam o *TensorFlow*, 4 utilizam o *MXNet* e 4 utilizam o *Torch/PyTorch*.

Para os trabalhos relacionados à precificação de nuvem, nota-se que existe um consenso em relação às formas de cobrança, todos citam o modelo dinâmico como uma forma de cobrança utilizada pelas nuvens, principalmente pelo fato desse modelo ser justo tanto para o provedor quanto para o cliente.

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.

4. 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 baseada nos princípios de avaliação de desempenho de sistemas

computacionais apresentados em [JAIN 1991] e é composta de cinco atividades, são elas: Entendimento do Ambiente, Planejamento de Experimentos, Configuração do Ambiente, Medição e Análise de Métricas. Essas atividades propostas visam permitir a compreensão da natureza do ambiente bem como desenhar experimentos consistentes com o objetivo final de avaliar o treinamento de redes neurais.

Na Figura 1 é possível ver essas atividades e seus subprocessos.

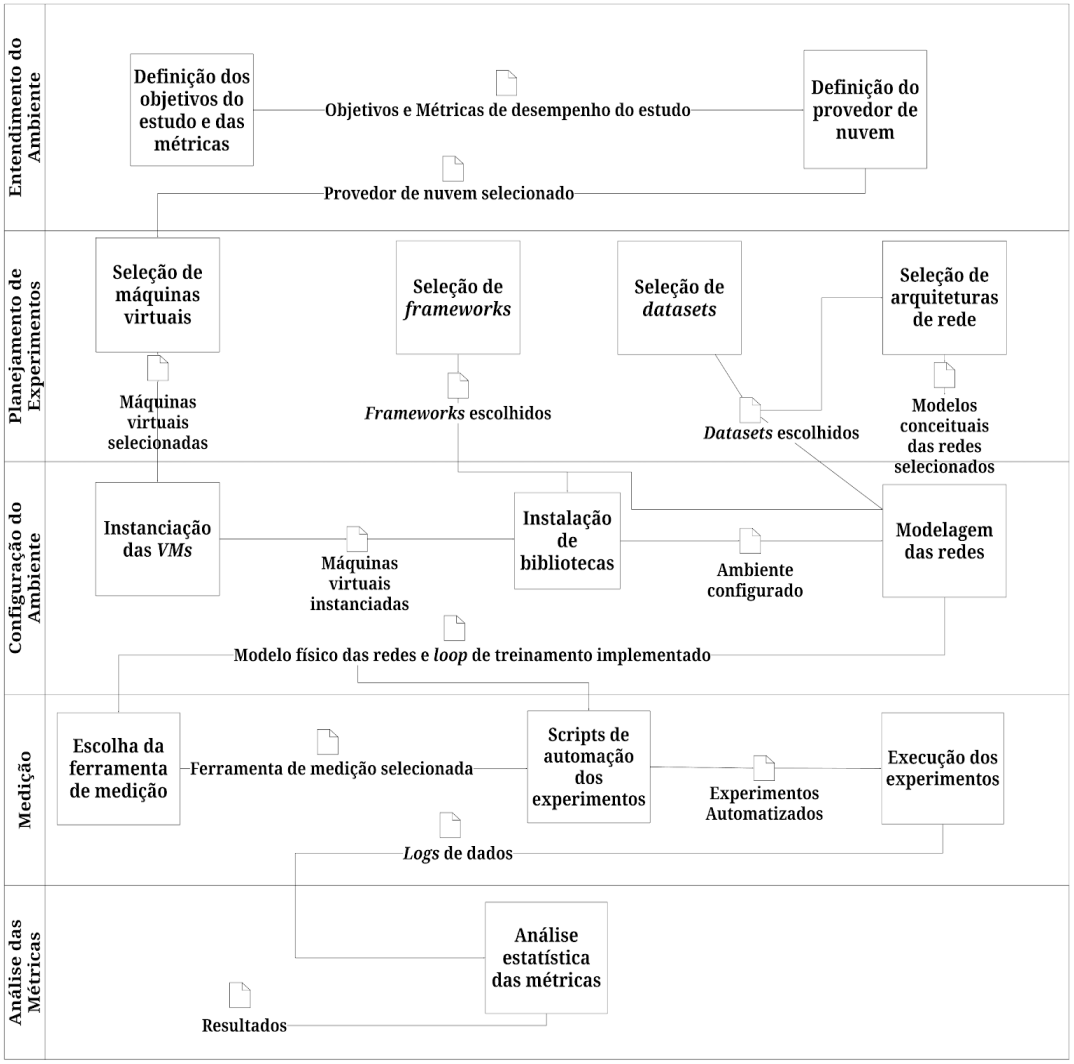

Figura 1 – Metodologia empregada

4.1 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 acurácia, já a métrica de custo pode são os custos dos recursos usados.

Depois da definição das métricas, 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.

4.2 Planejamento de experimentos

O objetivo do planejamento de experimentos é obter o máximo de informação com o mínimo de experimentos, economizando esforço considerável que seria gasto na coleta de dados. Uma análise adequada dos experimentos também ajuda a separar os efeitos de vários fatores que podem afetar o desempenho, além de permitir determinar se um fator tem um efeito significativo ou se a diferença observada é simplesmente devido ao acaso, variações causadas por erros de medição ou parâmetros que não foram controlados [JAIN 1991].

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* disponíveis para utilização, como por exemplo, *Caffe*, *Chainer*, *CNTK*, *TensorFlow*, *PyTorch*, 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, 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*.

4.3 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

instanciação de máquinas virtuais, instalação das bibliotecas necessárias para executar o treinamento das redes neurais e a criação dos modelos das redes neurais.

Os subprocessos dessa atividade estão fortemente relacionados com os subprocessos da atividade anterior (Seção 4.2), pois para instanciar as máquinas primeiro é necessário saber quais os tipos de máquinas serão utilizadas. Para instalar bibliotecas e configurar o loop de treinamento é imperativo conhecer quais *frameworks*, *datasets* e arquitetura de rede serão utilizados.

4.4 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. Garantir a qualidade dos dados medidos é essencial, nesse sentido, a repetição da execução dos cenários planejados é primordial.

4.5 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 coletadas e, opcionalmente, com a plotagem de gráficos das médias para facilitar a visualização dos resultados.

5. 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.

5.1 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, custo, utilização de *CPU*, utilização de memória, utilização de *GPU* e utilização da memória da *GPU*, sendo a primeira métrica, medida em segundos(s), o custo medido em Reais(R$), 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.

5.2 Planejamento de experimentos

Após a conclusão das primeiras atividades, que visam entender o ambiente, é necessário planejar os experimentos, ou seja, selecionar fatores e os níveis que serão considerados para criar cenários de experimentação. Esses fatores e seus níveis podem ser vistos na Tabela 2, e nas subseções adiante, como esses níveis foram selecionados.

Tabela 2 - 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*	-

5.2.1 Máquinas virtuais

Para executar experimentos é importante definirmos ambientes distintos para possibilitar uma comparação, tomando como base o que vem sendo feito nos trabalhos relacionados, podemos distinguir dois ambientes: o ambiente com apenas *CPU* e o ambiente com *CPU* e *GPU*.

Com o provedor definido e os tipos de ambientes em mente, resta então selecionar as especificações do ambiente. Para definir isso foi usado o valor do custo/hora das máquinas virtuais a fim de encontrar um valor razoável que não compromete-se tanto o orçamento disponível de U$200(duzentos dólares, ou em conversão direta na época, R$973) ao mesmo tempo em que um ambiente robusto o suficiente para executar a tarefa de treinar redes neurais fosse escolhido.

Ao final, dois ambientes se destacaram, o ambiente da máquina *F4s_v2* e o ambiente *NC4as_T4_v3*, onde o primeiro conta apenas com CPU e o segundo conta com *CPU* e *GPU*. Na Tabela 3 é possível visualizar mais detalhes das especificações de cada máquina virtual.

Tabela 3 - 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

5.2.2 Frameworks

Seguindo o padrão do estado da arte, foram selecionados três dos principais *frameworks* disponíveis, eles são os mais utilizados em trabalhos recentes na literatura e só por isso já eram bons candidatos a serem escolhidos, porém outro fator que pesa nessa escolha é a questão da usabilidade, existem muitos projetos de fora do meio acadêmico que utilizam esses *frameworks* como ferramenta fundamental de trabalho.

Os *frameworks* em questão são: *PyTorch, MXNet e TensorFlow*. Todos os 3 permitem uma rápida modelagem de redes neurais e são representantes do que se tem de moderno nesta área.

5.2.3 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.

5.2.3 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.

5.3 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. Todas as bibliotecas foram instaladas através do *pip* 23.3, o gerenciador de pacotes da linguagem *Python*, este por sua vez na versão 3.9.

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.

5.4 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 as métricas de utilização de recursos. O *loop* de treinamento emite sinais quando uma época ou o treinamento da rede neural acaba. As métricas de utilização de CPU, Memória, GPU, Memória da GPU e tempo são registradas em um arquivo de *log* com valores separados por vírgula, sempre que capturadas.

Com o tempo de treinamento é possível calcular a estimativa do custo total do treinamento do experimento utilizando a seguinte fórmula:

$$Custo_{vm}/hora \;\; x\, Tempo\_Treinamento$$

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 10 vezes.

É importante notar, que o treinamento de uma rede neural, por padrão, toma cerca de 30 a 100 épocas, contudo, isso é válido apenas quando queremos melhorar a eficácia da rede neural, porém, este trabalho não visa avaliar esta métrica.

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

A Tabela 4 mostra a configuração de cada cenário dos experimentos.

Tabela 4 - Cenários dos experimentos.

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

Para coletar os dados das métricas é necessário executar o treinamento com as configurações de cada cenário. A seguir são apresentadas figuras (Figuras 2 a 5) com gráficos das médias das métricas coletadas.

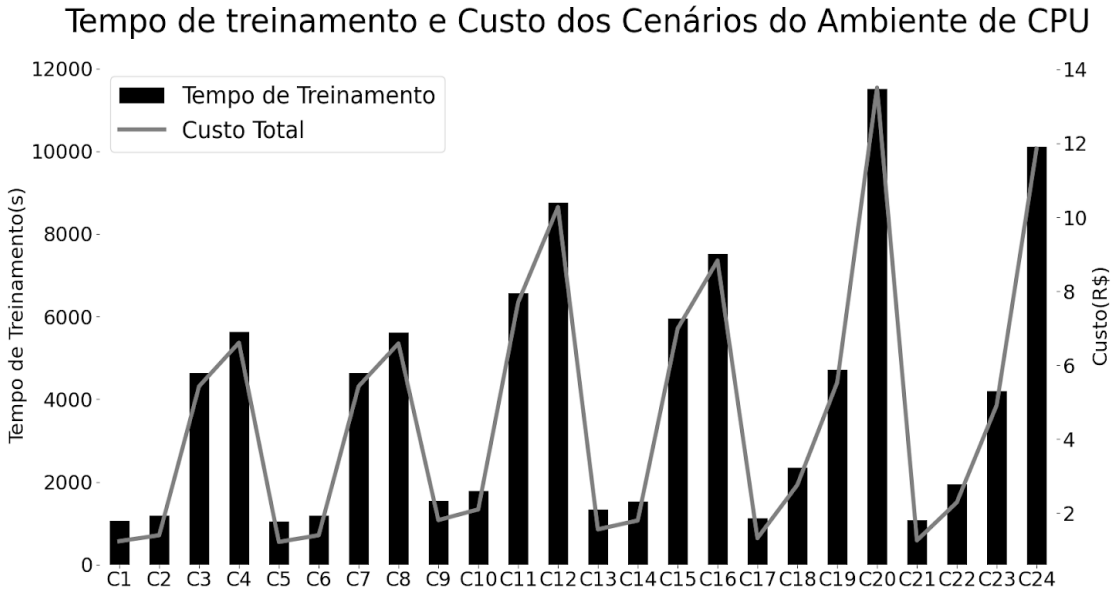

Figura 2 - Gráfico das médias de tempo de treinamento e custo do ambiente *CPU*.

As Figuras 2 mostra um gráfico de barras referente às médias de tempo de treinamento de cada cenário que utiliza apenas a CPU para treinar as redes neurais após

todas as 5 execuções do cenário, além disso essa figura possui um gráfico de linhas que mostram o custo total em reais(R$) de cada cenário após as execuções do cenário.

É possível notar no gráfico que as médias de tempo de treinamento para os cenários que utilizam a dimensão de imagem 64x64 são no mínimo cerca de 4000 segundos maior do que a mesma média para os cenários que utilizam a dimensão de imagem 32x32.

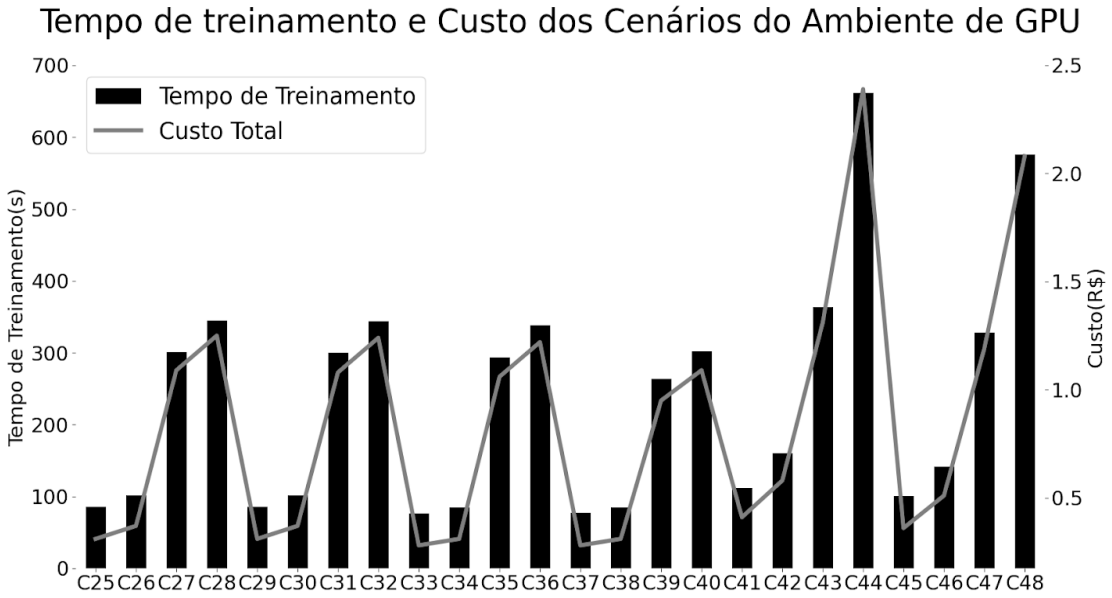

Figura 3 - Gráfico das médias de tempo de treinamento e custo do ambiente *GPU*.

A Figura 3, assim como a Figura 2, mostra um gráfico de barras referente às médias de tempo de treinamento de cada cenário que possui e utiliza a GPU para o treinamento das redes neurais após todas as execuções do cenário, essa figura também possui o gráfico de linhas que mostram o custo total em reais(R$) de cada cenário após as 5 execuções do cenário.

Os resultados apresentados nas Figuras 2 e 3 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 nas barras dos cenários que utilizam a Rede 2, esses cenários requerem no mínimo 10% mais tempo para treinar a rede.

O ambiente de *GPU* é pelo menos 10 vezes mais rápido que o ambiente de *CPU*. Apesar do valor do custo/hora do ambiente com *GPU* ser o triplo do ambiente de *CPU*, o custo total do ambiente de *CPU* foi cerca de 4 a 6 vezes maior que o do ambiente de *GPU*, Considerando os cenários com *TensorFlow*, o cenário C20 no ambiente de CPU custou cerca de R$14,00, enquanto o mesmo cenário no ambiente de GPU (C44) teve um custo aproximado de R$2,50.

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.

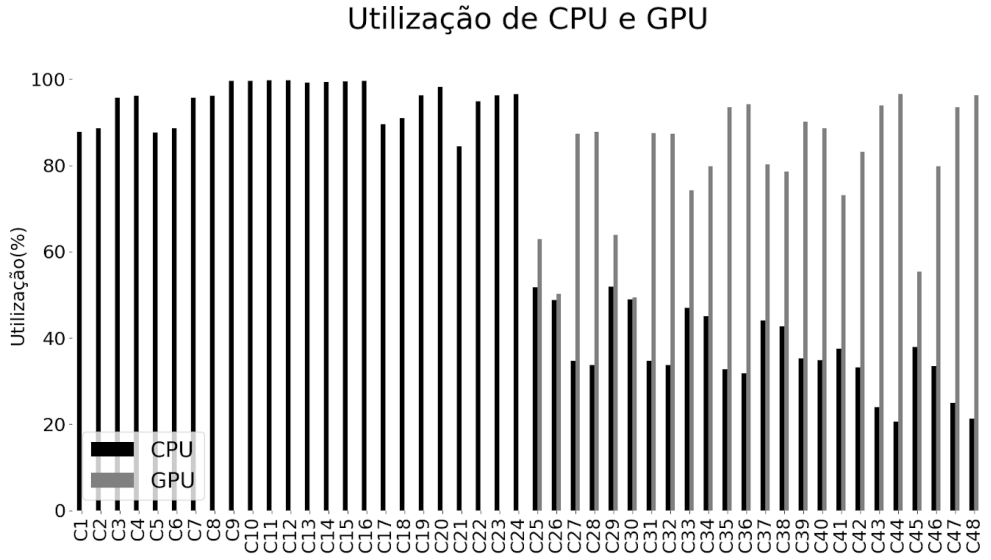

Figura 4 - Gráfico das médias de utilização de *CPU* e *GPU*

A Figura 4 exibe gráficos de barras das médias de utilização de *CPU* e utilização de *GPU* quando aplicável, após as 5 execuções dos experimentos de cada cenário.

Considerando a Figura 4, observa-se que o *PyTorch* é 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 40% a mais da *GPU* em relação a *CPU*.

A Figura 5, exibe gráficos de barras para utilização de memória e utilização de memória da GPU quando possível.

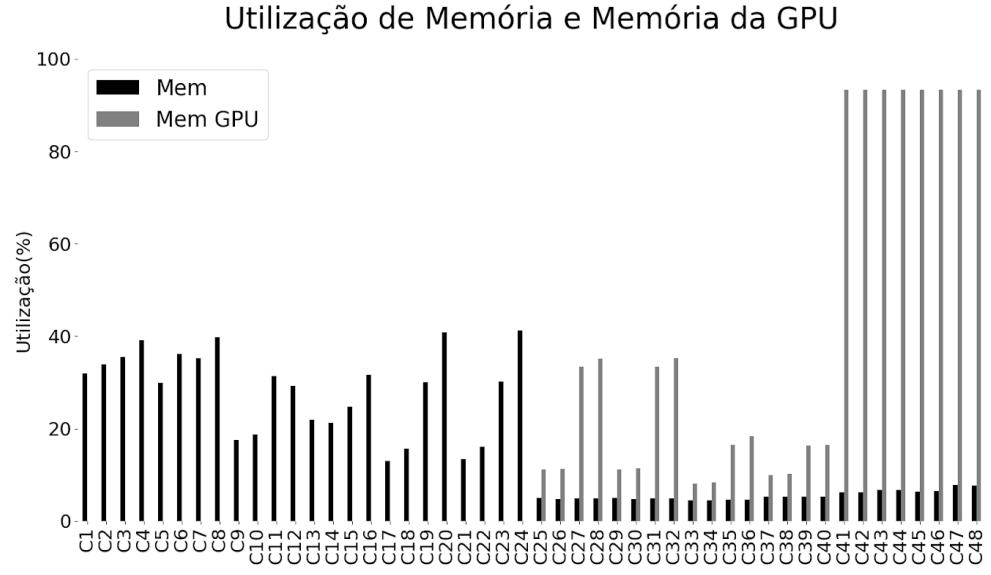

Figura 5 - Gráfico das médias de utilização de memória e memória da *GPU*

A Figura 5 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, a soma dos custos dos experimentos dá um valor aproximado de R$134,00, o que mostra que a equação apresentada para calcular o custo é bem precisa.

6. 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*), 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 aproveitado pelos diversos *frameworks* existentes como demonstrado nas métricas coletadas no estudo.

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-nuv em-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.

JAIN. The art of computer systems performance analysis : techniques for experimental design, measurement, simulation, and modeling. New York: Wiley, 1991.

JAIN, D. et al. Lung Cancer Detection Using Convolutional Neural Network. 11 nov. 2022.

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.

YAZAN ALAHMED et al. “How Does ChatGPT Work” Examining Functionality To The Creative AI CHATGPT on X’s (Twitter) Platform. 21 nov. 2023.

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