UNIVERSIDADE FEDERAL DO OESTE DO PARÁ – UFOPA
INSTITUTO DE ENGENHARIA E GEOCIÊNCIAS – IEG
BACHARELADO INTERDISCIPLINAR EM CIÊNCIA E TECNOLOGIA

REBECA TRAVASSOS PRAIA

PROCESSAMENTO DE DADOS COM ALGORITMOS DE INTELIGÊNCIA ARTIFICIAL NA PLATAFORMA NVIDIA JETSON NANO COM VISTAS AO DESENVOLVIMENTO DE SISTEMAS DE DEEP LEARNING E INTERNET DAS COISAS

SANTARÉM
2022

REBECA TRAVASSOS PRAIA

PROCESSAMENTO DE DADOS COM ALGORITMOS DE INTELIGÊNCIA ARTIFICIAL NA PLATAFORMA NVIDIA JETSON NANO COM VISTAS AO DESENVOLVIMENTO DE SISTEMAS DE DEEP LEARNING E INTERNET DAS COISAS

Trabalho de Conclusão de Curso apresentado ao Bacharelado Interdisciplinar em Ciência e Tecnologia para obtenção do grau de Bacharel em Ciência e Tecnologia na Universidade Federal do Oeste do Pará, Instituto de Engenharia e Geociências.

Orientador: Anderson Alvarenga De Moura Meneses

SANTARÉM

2022

PROCESSAMENTO DE DADOS COM ALGORITMOS DE INTELIGÊNCIA ARTIFICIAL NA PLATAFORMA NVIDIA JETSON NANO COM VISTAS AO DESENVOLVIMENTO DE SISTEMAS DE DEEP LEARNING E INTERNET DAS COISAS

Rebeca Travassos Praia[1] – rebeca.praia@discente.ufopa.edu.br
Marla Teresinha Barbosa Geller[1,2] – marla.geller@gmail.com
Anderson Alvarenga de Moura Meneses[1] – anderson.meneses@ufopa.edu.br
[1] Laboratório de Inteligência Computacional – Universidade Federal do Oeste do Pará - Santarém, PA, Brasil
[2] Curso de Sistemas de Informação – Centro Universitário Luterano de Santarém – CEULS/ULBRA – Santarém, PA, Brasil

Resumo. *No presente trabalho, foi investigado o módulo Jetson Nano da Nvidia, destinado a aplicações de Inteligência Artificial (IA) para acoplamento a sistemas de Internet das Coisas (Internet of Things, IoT). O módulo Jetson Nano possui uma GPU (Graphics Processing Unit) de menor porte, com poder de processamento reduzido, mas que pode atender a necessidades básicas de treinamentos de Redes Neurais Artificiais (RNAs) profundas, a serem executadas em nós-folhas de sistemas de IoT. Mais especificamente, o objetivo deste trabalho é avaliação da performance da Plataforma Nvidia Jetson Nano, sob uma abordagem Edge Computing, por meio do treinamento de algoritmos de aprendizado de máquina na classificação de um conjunto de dados de Marketing Bancário, utilizando técnicas de ciência de dados no objetivo de predizer se o cliente aceitará ou não a proposta de investimento do banco. A avaliação do desempenho e eficiência da placa foram observadas pelo tempo de execução da aplicação e do resultado do treinamento dos algoritmos. Portanto, a integração de edge computing e IA com base nos resultados práticos deste artigo pode fornecer aos pesquisadores a possibilidade de incorporar vários modelos de aprendizado no sistema embarcado Jetson para desenvolver aplicações diversificadas em diversos cenários.*

Palavras-chave: *Nvidia Jetson Nano, Algoritmos, Análise de Dados, Aprendizado de Máquina, Aprendizagem Profunda.*

1. INTRODUÇÃO

É amplamente acreditado que a capacidade de executar algoritmos de IA na computação de borda (Edge Computing) será crucial para alcançar determinadas aplicações que abrangem diversos componentes de inteligência computacional em soluções que demandam processamento na ponta (Mittal, 2019). Na grande maioria dos programas de IA processados na nuvem ocorre problema de latência sendo isso não mais tolerável em certas aplicações em que não se permite esperar as respostas (Al-Turjimant, 2019). Mover o processamento diretamente para computação de borda é uma decisão que pode trazer benefícios para aplicações

de IA. Isto se deve a uma série de fatores, como por exemplo a baixa latência, maior eficiência na largura de banda, proteção de privacidade e conectividade (Huh & Seo, 2019).

Em geral, dispositivos de borda baseados na arquitetura ARM de boa capacidade de processamento são especialmente promissores para acelerar algoritmos de AM, devido ao seu baixo consumo energético e alta eficiência (Bernardo et. al, 2021). Existem muitos sistemas de computador de placa única que oferecem possibilidades de desenvolvimento de *hardware* e *software* e incluem CPU/GPU (Suzen et al. 2020). Em meados de março de 2019 surgiu a placa básica Nvidia Jetson Nano que atende essas exigências computacionais.

A Nvidia Jetson Nano possui menor custo e tem potencial inovador, que oferece uma plataforma otimizada tendo recursos de computação para IA na borda (Kurniawan, 2021). A Jetson é um módulo de processamento integrado com uma CPU (Unidade de Central de Processamento) ARM 57, possui uma Unidade de Processamento Gráfico (GPU) núcleos de processamento ideal para lidar com várias tarefas simultâneas (Valladares et. al, 2021) como pré-processar os dados, treinar e inferir uma Rede Neural Artificial (RNA) profunda (Chollet, 2018) e tem como principal objetivo criar soluções de Internet das Coisas (IoT) (Serpanos e Wolf, 2017).

A plataforma Jetson da Nvidia foi idealmente projetada para a inferência de modelos de AM (Mittal, 2019), não sendo comum o uso desse tipo de arquitetura para treinamento de AM, pois treinar uma rede neural envolve muito mais computação do que inferir de uma rede pré-treinada (Fang, 2020). Em suma, durante o treinamento, uma rede neural deve passar pela etapa de propagação e retropropagação e é computacionalmente caro, pois requer muitos cálculos de gradientes e atualizações de variáveis de peso (Haykin, 2001), sendo as placas como a NVIDIA Jetson otimizadas para multiplicação rápida de matrizes, por isso são eficazes durante a inferência, mas não tanto para treinamento.

Por esse motivo, justifica-se a busca por técnicas com o objetivo de acelerar aplicações de redes neurais na plataforma Jetson (Mittal, 2019). Estruturas de aprendizado profundo como Caffe, Tensorflow e Pythorch são otimizadas para serem executadas mais rapidamente em GPUs (Nvidia, 2019). Os frameworks aproveitam os recursos de processamento paralelo de uma GPU, acelerando as tarefas de treinamento e inferência e consequentemente ter um desempenho de fidelidade aprimorado.

Diante desse contexto, a pesquisa busca avaliar a viabilidade do uso do dispositivo Jetson Nano para treinamento de algoritmos de AM. Como experimento, fez-se necessário a criação de dois modelos computacionais para um conjunto de dados com mais de 75 mil registros em uma tarefa de classificação, sendo treinado em uma placa computacional compacta sem apoio de estruturas otimizadas. O conjunto de dados utilizados foi o *Bank Marketing* retirado da fonte *UCI Machine Learning Repository*, na qual esses dados estão relacionados com uma campanha de marketing direto (por telefone) de uma instituição portuguesa (Moro et al., 2014). O objetivo da classificação é prever se o cliente aceitará ou não a proposta de um depósito à prazo.

Tem-se como objetivo geral do trabalho a avaliação da performance da Jetson em relação ao tempo de execução dos algoritmos, além de, compará-los entre si em termos de acurácia e técnica de validação cruzada, trazendo a possibilidade se ter essa arquitetura como uma alternativa viável e mais recente, não somente para realização de modelos pré-treinados ou inferência, embora seja preferível, mas também para a fase de treinamento desses algoritmos.

O restante do artigo está estruturado da seguinte forma: na seção 2 são apresentados os trabalhos relacionados que tem objetivos semelhantes ao desta pesquisa. Na 3 seção é apresentado o referencial teórico com conceitos correlacionados a este trabalho. Em seguida, a 4 etapa do trabalho mostra a metodologia adotada para a execução da aplicação. E por fim, na seção 5 encontram-se respectivamente a conclusão e relação das referências da pesquisa.

2. TRABALHOS RELACIONADOS

Esta seção aborda de forma resumida trabalhos científicos correlacionados a este artigo.

Na UFOPA, pesquisas acadêmicas vêm sendo realizadas com a finalidade de monitoramento de consumo de energia elétrica. Varão et. al (2017) desenvolveram uma plataforma de monitoramento remoto de energia elétrica ao utilizar ferramentas Open Source para aquisição e gerenciamento de dados de sensores sendo visualizados em um gráfico na página web em tempo real. Varão et. al (2018) deram continuidade a linha de pesquisa implementando um sistema de monitoramento de corrente elétrica de um bebedouro localizado no LABIC[1] na UFOPA. Com isso, o usuário é capaz de observar os dados em um determinado período, por exemplo, e comparar com dados de período anteriores afim de verificar padrões e ter noção do consumo de energia para uma melhor gestão energética eficiente. Outros avanços incluem a modelagem em Unified Modeling Language (UML) de um sistema voltado para IoT, bem como a predição de série temporal de consumo de energia com IA (Geller & Meneses, 2021).

O trabalho realizado por da Silva et al. (2021) buscou soluções para sustentabilidade e redução de custos através do framework EnergySaver desenvolvido pelo LABIC[1], onde todos os detalhes da sua implementação são descritos minuciosamente em um manual localizado na plataforma ARXIV. O desenvolvimento do EnergySaver, que visa monitorar o consumo de energia elétrica, desde a captura de dados até a previsão de consumo incluiu tecnologias de código aberto aplicadas a IoT, sistemas embarcados e rede de Memória de curto Prazo Longa (LSTM). Os resultados obtidos pela série temporal ficaram próximas do valor real, porém foi necessária uma pesquisa mais afundo sobre seu desempenho estatístico da rede LSTM.

Em um trabalho recente, da Silva et. al (2022) propuseram a avaliação do desempenho de redes LSTM em previsão de consumo de energia elétrica em série temporal, para um módulo de previsão de um sistema de IoT. Como experimento utilizaram três séries temporais e compararam o LSTM com algoritmos de Gradiente de Impulso Extremo (XGBoost) e Floresta Aleatória (FA). O modelo LSTM teve um desempenho estatisticamente superior ao XGBoost e FA, pois, foi comprovado que o LSTM apresentou previsão precisa para monitoramento e estimativa do consumo de energia, sendo aplicável a eficiência energética e à tomada de decisão, destacando que este módulo de previsão com redes LSTM está acoplado à estrutura do EnergySaver.

3. REFERENCIAL TEÓRICO

Esta seção aborda os principais conceitos constitutivos do artigo, como IA, ciência de dados, AM, aprendizado profundo, bem como apresenta especificações técnicas do sistema integrado Jetson Nano.

3.1 Inteligência artificial

Com os setores cada vez mais orientados por dados, a demanda pela tecnologia de IA é crescente, desde os processos de automação até modelos que simulam tipos de inteligência para os mais diversos propósitos. As aplicações dessas ferramentas estão relacionadas a tecnologias do nosso cotidiano, redes sociais ou e-commerce, e aplicações que fogem do senso comum, como cibersegurança ou astrofísica. (Uzinski; Abreu; Oliveira, 2020). Em tese, a IA em sua forma mais elementar é uma área da ciência da computação que faz as máquinas funcionarem de forma inteligente, na prática de automatizar processos cognitivos (Chollet, 2018).

3.2 Ciência de dados

Atualmente, a Ciência de Dados e o AM são considerados grandes segmentos de computação no mundo (Feltrin, 2019). Os profissionais especializados em estatística colaboram

com sólida teoria de análise de dados enquanto pesquisadores e engenheiros da computação contribuem com novas capacidades e possibilidades tecnológicas (de Oliveira; Guerra; McDonell, 2018). Para Feltrin (2019) tarefas de processamento em Ciência de Dados utilizam metodologias analíticas, matemáticas, estatísticas e técnicas necessárias para interpretar os dados, validar hipóteses, simular cenários, criar modelos de decisão, gerar aproximações e fazer previsões sobre o futuro. Assim, pesquisadores (as) das mais variadas áreas do conhecimento podem usufruir desse conjunto de técnicas para desenvolver suas pesquisas.

3.3 Aprendizado de máquina e aprendizado profundo

Como um subconjunto da IA, o AM utiliza de algoritmos para criar modelos de forma autônoma a partir de dados alimentados para que eles possam aprender sem serem explicitamente programados e fornecer previsões probabilísticas (Géron, 2019). Zocca et al. (2017), descrevem uma das técnicas principais de AM, os algoritmos de aprendizado supervisionado, também chamado de análise preditiva, usa algoritmos para treinar um modelo para encontrar padrões em um conjunto de dados com rótulos e recursos. Posteriormente, o mesmo usa o modelo treinado para prever os rótulos nos recursos de um novo conjunto de dados. Um dos mais utilizados algoritmos de AM são as Redes Neurais Artificiais (RNAs) (Nunes; Hernane; Andrade, 2016). Há atualmente várias arquiteturas de RNAs, porém, neste trabalho utilizou-se a técnica Redes Perceptron de múltiplas camadas (PMC) que são caracterizadas pela presença de pelo menos uma camada oculta de neurônios, situada entre a camada de entrada e a respectiva camada de saída (Nunes; Hernane; Andrade, 2016). Essas unidades intermediárias de uma rede MLP funcionam como detectores de características gerando uma codificação interna dos padrões de entrada, que é utilizada para a definição da saída da rede (Braga; Ludemir; Carvalho, 2000).

O termo aprendizado profundo (Deep Learning, DL) de acordo com Zocca et al., (2017), se aplica a uma classe de algoritmos de redes neurais profundas que podem usar diferentes algoritmos de treinamento (*Backpropagation*) e ajuste de pesos, e não estão limitados a redes neurais de retropropagação e feed-forward clássico. Basicamente, o aprendizado profundo é definido como uma classe de técnicas de AM em que as informações são processadas em camadas hierárquicas, para entender representações e recursos dos dados em níveis crescentes de complexidade.

3.4 Jetson Nano

A Nvidia traz para o mercado o Kit de Desenvolvimento da Plataforma Jetson com poder computacional de 472 GFLOPS para executar algoritmos IA, de sistema operacional Linux distribuição Ubuntu. Possui módulo com entrada gigabit ethernet integrado, 4 GB de memória, sistema de 64 bits e atende todas as características de processamento de áudio e vídeo. Tecnicamente, possui um processador quad-core ARM -A57 e uma GPU que é a placa gráfica do computador com 128 núcleos Cuda com base na arquitetura Maxwell (Nvidia, 2019).

4. MATERIAIS E MÉTODOS

O módulo Nvidia Jetson Nano descrito na seção 3.4 é utilizado no âmbito deste estudo e os componentes adicionais necessários para o desenvolvimento do projeto incluem: microSD card (64GB), Fonte de alimentação barril (5V/4A), cabo HDMI para VGA, Minicomputador Jetson Nano, monitor, teclado, mouse e um cabo Ethernet Rj 45.

A metodologia foi dividida em três etapas: A primeira é referente a montagem e conexão dos equipamentos citados acima, além do processo de configuração do sistema e ambiente Jetson Nano. A segunda etapa é o desenvolvimento de dois modelos de AM utilizando técnicas

de ciência de dados para aprimoramento dos dados. E a terceira etapa trata da avaliação do desempenho de treinamento dos algoritmos na Plataforma Nvidia Jetson Nano.

4.1 Procedimentos de configuração do Sistema Jetson

Após a conexão dos equipamentos, foi preparado o software essencial para colocar o Jetson em execução e realizar a configuração do sistema com a Jetson Nano Developer. A instalação do programa no Jetson Nano foi através de um cartão MicroSD que contém uma suíte de desenvolvimentos incluindo todos os componentes do Jetpack, como Kernel, multimídias, Bootloader, além do Sistema Operacional Ubuntu Linux. Para isso, foram baixados dois programas o SD Memory Card Formatter para formatar o cartão microSD e atualizar o arquivo de imagem Nvidia Jetpack, e o Etcher para adicionar o sistema operacional para o cartão SD que foram inseridos no Jetson Nano e conectados os periféricos na placa para realizar o primeiro boot.

4.2 Procedimentos de configuração do ambiente Jetson

Para desenvolver os dois modelos computacionais, realizar o pré-processamento e preparar a base de dados para a classificação, realizou-se a instalação do pacote Python 3.6 e suas bibliotecas e módulos complementares como: Pandas, Numpy, Matplotlib, Seaborn e ScikitLearn, e as estruturas de aprendizado profundo, Tensorflow e APi Keras. Para codificação utilizou-se o *Visual Studio Code*, compatível com o sistema da Jetson Nano.

5. PROCEDIMENTO DA APLICAÇÃO

Os métodos aplicados no desenvolvimento do modelo computacional consistem em 4 etapas. A Fig. 1 representa um panorama geral do processo da aplicação a partir dos dados obtidos.

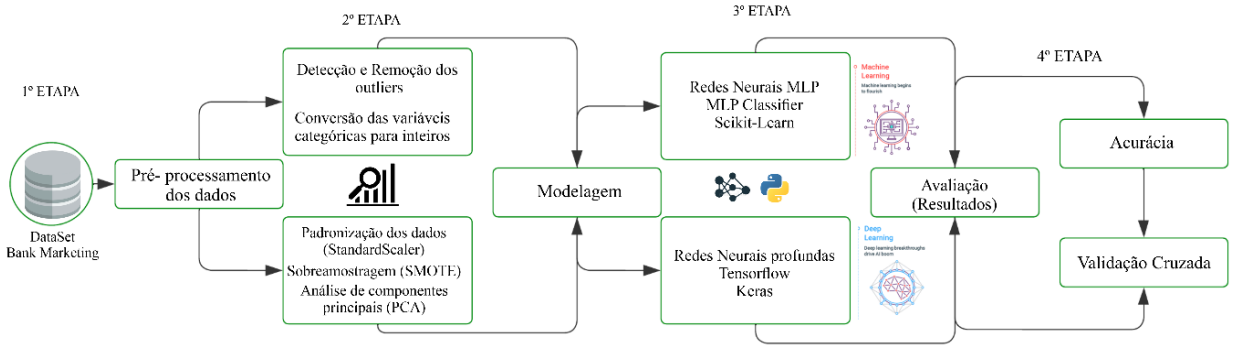


Figura 1 - Etapas da metodologia da criação dos modelos

Nas próximas subseções seguem as descrições de cada etapa desenvolvida para aplicação do modelo computacional.

5.1 Obtenção dos dados

O Dataset utilizado para o treinamento dos algoritmos foi o *Bank Marketing* uma base de dados pública disponibilizada no site *UCI Machine Learning Repository*, separada em dois grupos, um de clientes que aceitaram a campanha do banco assinando o depósito à prazo, e a outra de clientes que não se inscreveram. O objetivo é construir um modelo de AM que aprenda os padrões desconhecidos, classificando se o cliente irá assinar depósitos mais longos ou não.

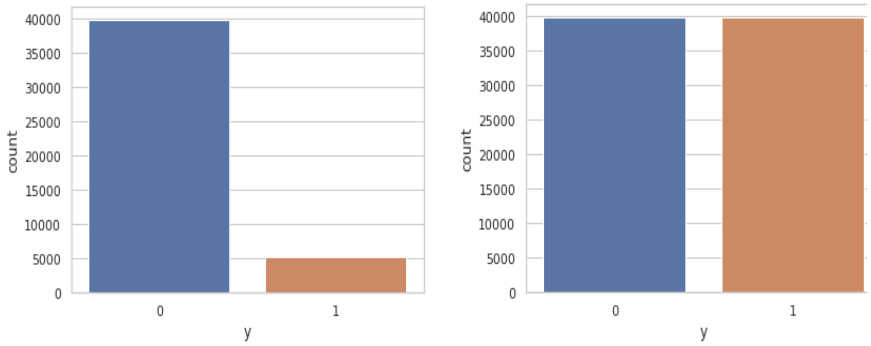
O conjunto de dados são compostos por 45211 amostras e 17 atributos, uma quantidade razoável de instâncias que serão consideradas para o treinamento na placa Jetson Nano.

5.2 Pré-processamento dos dados

Inicialmente buscou-se valores ausentes no conjunto de dados, o que não ocorreu. Em seguida, fez-se a detecção de valores discrepantes, encontrando-se muitos pontos de dados dispersos em algumas variáveis numéricas, que foram removidos, restando 45092 dos 45211 iniciais. Converteu-se as variáveis categóricas para tipo numérico utilizando a função *LabelEncoder* e *OrdinalEncode* e fez-se a reescala dos dados utilizando a função *StandardScaler*, que normaliza os dados, oferecendo dados padronizados ao classificador.

Com a análise exploratória dos dados verificou-se dados desbalanceados, onde o classificador se comporta com tendência para a classe majoritária, causando má classificação da classe menor o que pode afetar no resultado do modelo. Nessa base haviam 39843 registros da classe 0 e apenas 5249 da classe 1. Para resolver esse problema utilizou-se a técnica SMOTE (*Synthetic Minority Over-sampling Technique*), que cria artificialmente novas observações para a classe minoritária, até chegar a mesma proporção da classe super-representada (Chawla et al., 2002). Na Fig. 2 pode-se observar claramente antes e depois do uso da técnica de sobreamostragem, que ao balancear a amostra aumentou o número total de registros para 79686.

Ressalta-se ainda que após a definição da variável alvo (y), restaram 16 atributos no dataset, que são as variáveis preditoras. Além de, pela orientação do próprio dataset houve a remoção do atributo '*duration*', restando 15 atributos. Por último a técnica de PCA (Análise de Componentes Principais) foi utilizada para fazer o redimensionamento de recursos selecionando os atributos mais importantes que capturam o máximo de informações sobre o conjunto de dados (Jaadi, 2022). Com isso, o PCA ajustou os atributos para 13.



(a) (b)

Figura 2 – (a) Desequilíbrio das classes sem o SMOTE e (b) Equilíbrio das classes com SMOTE

Após o processo de codificação da técnica SMOTE, houve uma melhora nas métricas de avaliação do modelo, como mostra a Tabela 1:

Tabela 1 – Métricas antes e depois da aplicação do SMOTE

Relatório das Métricas antes do SMOTE	Relatório das Métricas depois do SMOTE
Accuracy = 0.85	Accuracy = 0.88
Recall = 0.30	Recall = 0.95
F1score = 0.32	F1score = 0.89

5.3 Desenvolvendo os modelos

As RNAs foram implementadas na linguagem Python pelo fato de ser uma linguagem de alto nível, clara e flexível, possibilitando resultados em pouco tempo.

A divisão do conjunto de dados em treinamento e teste implementado foi realizada pelo método train_test_split da biblioteca Sklearn determinando uma divisão de 80% para treinamento e 20% para teste, resultando em X_train (63748, 13) e X_test (15938, 13). Essa configuração foi a mesma para criação dos dois tipos de modelo.

A tabela 2 apresenta as configurações para o modelo de Rede Neural MLP e para o modelo Deep Learning:

Tabela 2 – Hiperparâmetros utilizados

Parametrização do modelo MLP

Input layer	Hidden layer	Output Layer	Solver	Activation	Max_iter
50 neurônios	200 neurônios	50 neurônios	adam	ReLU	1000
Parametrização do modelo deep learning					
1ª Layer dense	2ª Layer dense	3ª Input Layer	Optmizer	Activation	Loss function
256 neurônios	128 neurônios	10 neurônios Softmax	adam	Sigmoid ReLU	Categorical Crossentropy

A Jetson Nano obteve comportamento satisfatório durante sua fase de treinamento, apesar da memória RAM acabar excedendo sua capacidade, nesse caso a memória *Swap* (Memória secundária) evitou de o sistema ficar sobrecarregado, permitindo prosseguir com o treinamento dos algoritmos.

5.4 Métrica de avaliação e performance de treinamento na Jetson Nano

Foram feitos apenas dois treinamentos com o modelo de Rede Neural MLP. A rede neural teve um tempo de execução de aproximadamente 50 minutos para os dois treinamentos, e com acurácia de 87% e na validação cruzada alcançando a média dos testes de 76%. Na Fig. 3 pode-se observar que não houve diferença relevante nos testes com a mudança do parâmetro 'max_iter' no processo da validação cruzada e pouca no tempo de execução.

1ª Treinamento - MLP		**2ª Treinamento - MLP**	
Acurácia	87.70%	Acurácia	87.70%
pontuação treinamento	0.9655	pontuação treinamento	0.9655
pontuação teste	0.8770	pontuação teste	0.8770
Erro quadrático médio	0.12291	Erro quadrático médio	0.12291
Cross Validation		**Cross Validation**	
Kfold = 5		Kfold = 5	
Máx. Iterações = 1000		Máx.Iterações = 2000	
Teste 1ª	76.20%	Teste 1ª	76.20%
Teste 2ª	76.18%	Teste 2ª	76.18%
Teste 3ª	75.89%	Teste 3ª	75.89%
Teste 4ª	76.45%	Teste 4ª	76.45%
Teste 5ª	75.53%	Teste 5ª	75.53%
Média	76.05%	Média	76.05%
Desvio Padrão	(+/- 0.0031%)	Desvio Padrão	(+/- 0.0031%)
Tempo de Execução	00:50min:22s	Tempo de Execução	00:52min:51s

Figura 3 – Treinamento do algoritmo MLP

Foram realizados seis treinamentos para o modelo *deep learning*, e em cada execução o resultado da acurácia variou consideravelmente conforme a alteração dos seguintes parâmetros**:** tamanho do lote (*Batch_size*) e o número de épocas (*Epochs*).

Para os dois primeiros treinamentos, manteve-se o mesmo valor de lotes, porém com o número de épocas diferentes, o 2ª treinamento teve um resultado melhor na validação cruzada, com pouca diferença no resultado da acurácia, mas com tempo de execução maior (Fig. 4).

1° Treinamento	
Acurácia	70.25%
Cross Validation	
Kfold = 5 Batch_Size = 10 Epochs = 5	
Teste 1ª	71.01%
Teste 2ª	72.33%
Teste 3ª	73.19%
Teste 4ª	73.87%
Teste 5ª	74.27%
Média	72.94%
Desvio Padrão	(+/- 1.17%)
Tempo de Execução	00:34min:42s

2° Treinamento	
Acurácia	70.92%
Cross Validation	
Kfold = 5 Batch_Size = 10 Epochs = 10	
Teste 1ª	73.51%
Teste 2ª	75.55%
Teste 3ª	76.78%
Teste 4ª	78.59%
Teste 5ª	79.64%
Média	76.69%
Desvio Padrão	(+/- 2.03%)
Tempo de Execução	1h:10min:54s

Figura 4 - Resultado do 1ª e 2ª treinamento

Para o 3ª e 4ª treinamento (Fig. 5), aumentou-se a quantidade dos lotes, podendo-se observar que com maior valor do *batch_size* a dinâmica de treinamento foi mais rápida, porém o modelo não atingiu um bom desempenho. Isso implica no tamanho de lotes menores, onde o treinamento ocorre mais lentamente, mas pode convergir mais rapidamente e consequentemente obter um melhor resultado.

3ª Treinamento	
Acurácia	68.28%
Cross Validation	
Kfold = 5 Batch_Size = 30 Epochs = 5	
Teste 1ª	70.29%
Teste 2ª	71.95%
Teste 3ª	72.17%
Teste 4ª	72.28%
Teste 5ª	73.00%
Média	71.94%
Desvio Padrão	(+/- 0.90%)
Tempo de Execução	00:13min:42s

4ª Treinamento	
Acurácia	70.80%
Cross Validation	
Kfold = 5 Batch_Size = 30 Epochs = 10	
Teste 1ª	72.14%
Teste 2ª	74.02%
Teste 3ª	75.16%
Teste 4ª	76.95%
Teste 5ª	77.35%
Média	75.12%
Desvio Padrão	(+/- 1.92%)
Tempo de Execução	00:24min:47s

Figura 5 - Resultado do 3ª e 4ª treinamento

Por fim, obteve-se melhores resultados nos dois últimos treinamentos (Fig. 6), em específico para o 6ª treinamento chegando a 80%, na acurácia e 86% na validação cruzada. Determinou-se de maneira arbitrária o número de épocas para alcançar um desempenho ideal. Com isso, observou-se empiricamente que o modelo melhorou com mais épocas de treinamento estando em uma proporção adequada com o batch_size até certo ponto. O treinamento pela validação cruzada demanda mais processamento e consequentemente o tempo de execução maior. Porém essa técnica é uma maneira bem prática para se ter uma noção real do desempenho do modelo.

5° Treinamento	
Acurácia	77.24%
Cross Validation	
Kfold = 5 Batch_Size = 10 Epochs = 50	
Teste 1ª	81.16%
Teste 2ª	83.66%
Teste 3ª	85.61%
Teste 4ª	86.57%
Teste 5ª	86.80%
Média	84.76%
Desvio Padrão	(+/- 2.11%)
Tempo de Execução	05:40min:13s

6° Treinamento	
Acurácia	80.22%
Cross Validation	
Kfold = 5 Batch_Size = 10 Epochs = 100	
Teste 1ª	84.22%
Teste 2ª	86.72%
Teste 3ª	87.35%
Teste 4ª	88.36%
Teste 5ª	87.83%
Média	86.92%
Desvio Padrão	(+/- 1.40%)
Tempo de Execução	11:09min:49s

Figura 6 - Resultado do 5ª e 6ª treinamento

6. CONCLUSÃO

Com o recente avanço no aprendizado profundo, tem havido um interesse crescente em aplicar algoritmos de aprendizado profundo a dispositivos de borda. Ao contrário dos computadores tradicionais, os sistemas de computador de placa única são um sistema de alto desempenho em sua arquitetura simples.

De acordo com os resultados, foi possível evidenciar o desempenho promissor da Plataforma Nvidia Jetson Nano para o treinamento de algoritmos na tarefa de classificação, se mantendo constante por mais de 11 horas sem intervenções do sistema.

Conclui-se que a placa Jetson Nano mesmo com suas limitações computacionais teve a capacidade de treinar algoritmos de AM localmente em uma quantidade significativa de dados, mesmo que seja lento comparado com computadores de alto desempenho ou servidor para realizar o treinamento. A Jetson Nano é uma opção atraente pelo fato de ter nós de borda capaz de processar os dados, como também de oferecer um ambiente otimizado tendo recursos de computação para IA, além de ter um formato pequeno e de baixo consumo energético o que a torna perfeita para cenários com restrições de peso e potência. Até onde se sabe, este é o primeiro trabalho que avaliou a possibilidade de treinamento de algoritmos de AM com dados e não imagens na Plataforma Nvidia Jetson Nano.

Reconhecimentos
Os autores agradecem ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e ao Programa Institucional de Bolsas de Iniciação Científica (PIBIC) pelo apoio financeiro.

REFERÊNCIAS

Al-Turjman, F. (2019), "*Edge Computing: From Hype to Reality*", 1º ed., Springer International Publishing, Turquia.

Bernardo, F; Yokoyama, A; Schulze, B; Ferro, M. (2021). *Avaliação do Consumo de Energia para o Treinamento de Aprendizado de Máquina utilizando Single-board computers baseadas em ARM*. In: Simpósio em Sistemas Computacionais de Alto Desempenho (WSCAD). Belo Horizonte. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação, 2021. p. 60-71. DOI: https://doi.org/10.5753/wscad.2021.18512.

Braga, P.A; Ludemir, B.T; Carvalho, F.L.P.A, D. (2000), "*Redes Neurais Artificiais: Teoria e aplicações*", 1º ed LTC, Rio de Janeiro.

Chawla, N., Bowyer, K., Hall, L., & Kegelmeyer, P. (2002). *SMOTE: Synthetic Minority Over-sampling TEchnique. Journal of Artificial Intelligence Research*, 16, 321-357. https://doi.org/10.1613/jair.953

Chollet, F. (2018), "*Deep Learning with Python*",1ª ed., Manning Publication, Shelter Island.

de Oliveira, P.F; Guerra, S; McDonnel, R. (2018) "*Ciência de Dados com R*": *Introdução* (2ª ed.) BPAD, Brasília.

Fang, Y., Shalmani, S.M., & Zheng, R. (2020). CacheNet: *A Model Caching Framework for Deep Learning Inference on the Edge*. ArXiv, abs/2007.01793.

Feltrin, F.B. (2019), "*Ciência de Dados e Aprendizado de Máquina*: *Uma abordagem prática as redes neurais aritificiais*", 1º ed.

Geller, M. T. B; Meneses, A. A. M. *Modelling IoT Systems with UML: A Case Study for Monitoring and Predicting Power Consumption.* American Journal of Engineering and Applied Sciences, v. 14, p. 81-93, 2021.

Géron, A. (2019), "*Mãos à Obra*": *Aprendizado de Máquina com Scikit-Learn & TensorFlow: Conceitos, ferramentas e técnicas para a construção de sistemas inteligentes* (1ª ed.) Alta Books, Rio de Janeiro.

Haykin, S. *Redes Neurais: Princípios e Prática* (2ª ed.). Porto Alegre: Bookman, 2001. 898 p

Huh, J. and Seo, Y., "*Understanding Edge Computing: Engineering Evolution With Artificial Intelligence*," in IEEE Acesso, vol. 7, pp. 164229-164245, 2019, doi: 10.1109/ACCESS.2019.2945338.

Jaadi, Zakaria. "*A step-by-step Explanation of Principal Component Analysis (PCA)*" (2022). Disponível em: https://builtin.com/data-science/step-step-explanation-principal-component-analysiss/. Acesso em 5 de mai de 2022.

Kurniawan, A. (2021), "*IoT Projects with NVIDIA Jetson Nano: AI-Enabled Internet of Things Projects for Beginners*", 1□ ed., Apress, New York.

Mittal. S. A. *Survey on optimized implementation of deep learning models on the NVIDIA Jetson platform* J. Syst. Archit., 97 (2019), pp. 428-442, 10.1016/j.sysarc.2019.01.011.

Moro, S., Cortez, P and P. Rita, P. *A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems*, Elsevier, 62:22-31, June 2014.

Nunes, S.I; Hernane, S.D; Andrade, F.R. (2016), "*Redes Neurais Artificiais para Engenharia e Ciências Aplicadas": Fundamentos teóricos e aspectos práticos* (2ª ed.) Artlib, São Paulo.

Nvidia (2019). *Nvidia Deep Learning Institute*, may 2019. Disponível em: https://courses.nvidia.com/courses/course-v1:DLI+S-RX-02+V2/info. Acesso em 12 de Novembro de 2020.

Santos, c. a. m.; Silva, d. g.; Geller, m. t. b.; Varao, d. f. s.; Bentes, j.; Moura, m. s. s.; Teixeira, y. b.; Meneses, a. a. m. *EnergySaver* 1.0. 2020.

Serpanos, D., & Wolf, M. (2017), "*Internet das Coisas (IoT) sistemas: arquiteturas, algoritmos, metodologias*", 1º ed Springer, Switzerland.

Silva, D.G.; Geller, M.T.B.; Moura, M.S.S.; Meneses, A.A.M. "*A deep learning prediction module for the IoT system EnergySaver for monitoring and estimating power consumption*," In: 16th Conference on Sustainable Development of Energy, Water and Environment Systems, 2021, Dubrovnik. Conference Proceedings, 2021.

Silva, D. G.; Geller, M. T. B.; Moura, M.S.S.; Meneses, A. A. M. *Performance evaluation of LSTM neural networks for consumption prediction*. e-Prime - Advances in Electrical Engineering, Electronics and Energy, v. 2, p. 100030, 2022.

Suzen, A. A., Duman, B., and S¸ en, B. (2020). *Benchmark analysis of jetson tx2, jetson nano and raspberry pi using deep-cnn*. In 2020 Int. Cong. on Human- Computer Interaction, Optimization and Robotic Applications, pages 1–5. IEEE.

Uzinski, J.C; de Abreu, C.C.E; de Oliveira, B.R. (2020), "*Aplicações de Inteligência Artificial e Ciência de Dados*" (1ª ed.) Pantanal, Mato Grosso.

Valladares, S., Toscano, M., Tufiño, R., Morillo, P., & Vallejo-Huanga, D. (2021). *Performance evaluation of the Nvidia Jetson Nano through a real-time machine learning application*. In International Conference on Intelligent Human Systems Integration (pp. 343-349). Springer, Cham.

Varão, D.F.S; Teixeira, Y.B.; Bentes, J.; Meneses, A.A.M. *Desenvolvimento de um Protótipo de Sistema IoT em Linguagem Python com Vistas ao Monitoramento de Consumo Elétrico*. In: Simpósio de Computação do Oeste do Pará, 2018, Santarém. Anais do Simpósio de Computação do Oeste do Pará, 2018

Varão, D.F.S.; Teixeira, Y.B.; Bentes, J.; Meneses, A.A.M. *Desenvolvimento de uma Plataforma de IoT para Monitoramento de Consumo de Energia Elétrica*. In: I Simpósio de Computação do Oeste do Pará – SCOOP 2017, Santarém. Livro de Resumos – SCOOP 2017.

Zocca, V; Spacagna, G; Slater, D; Roelants, P. (2017), "*Python Deep Learning": Next generation techniques to revolutionize computer vison, AI, speech and data analysis* (1ª ed.) Packt, Birminggham- Mumbai.

DATA PROCESSING WITH ARTIFICIAL INTELLIGENCE ALGORITHMS ON THE NVIDIA JETSON NANO PLATFORM FOR THE DEVELOPMENT OF DEEP LEARNING SYSTEMS AND THE INTERNET OF THINGS

Abstract. *In this paper, we investigated Nvidia's Jetson Nano module, intended for Artificial Intelligence (AI) applications for coupling to Internet of Things (IoT) systems. The Jetson Nano module has a smaller GPU (Graphics Processing Unit) with reduced processing power, but which can meet basic needs for deep Artificial Neural Networks (ANNs) training to be run on leaf nodes of IoT systems. More specifically, the objective of this work is to evaluate the performance of the Nvidia Jetson Nano Platform, under an Edge Computing approach, by training machine learning algorithms on the classification of a Banking Marketing dataset using data science techniques in order to predict whether the customer will accept the bank's investment proposal or not. The evaluation of the performance and efficiency of the board was observed by the running time of the application and the result of the algorithms training. Therefore, the integration of edge computing and AI based on the practical results of this paper may provide researchers with the possibility of incorporating various learning models in the Jetson embedded system to develop diversified applications in various scenarios.*

Keywords: *Nvidia Jetson Nano; Algorithms; Data Analysis; Machine Learning; Deep Learning.*