José Maria dos Santos Leal
Orientador: Rayner Gomes Sousa

Aplicação de Algoritmos de Aprendizagem de Máquina para o Fatiamento de Redes no Cenário 5G

Picos - PI
17 de Março de 2023

José Maria dos Santos Leal
Orientador: Rayner Gomes Sousa

Aplicação de Algoritmos de Aprendizagem de Máquina para o Fatiamento de Redes no Cenário 5G

Monografia submetida ao Curso de Bacharelado em Sistemas de Informação como requisito parcial para obtenção de grau de Bacharel em Sistemas de Informação.

Universidade Federal do Piauí
Campus Senador Helvídio Nunes de Barros
Bacharelado em Sistemas de Informação

Picos - PI
17 de Março de 2023

FICHA CATALOGRÁFICA
Serviço de Processamento Técnico da Universidade Federal do Piauí
Biblioteca José Albano de Macêdo

L435a Leal, José Maria dos Santos

Aplicação de algoritmos de aprendizagem de máquina para o fatiamento de redes no cenário 5G [recurso eletrônico] / José Maria dos Santos Leal – 2023.
46 f.

1 Arquivo em PDF
Indexado no catálogo *online* da biblioteca José Albano de Macêdo-CSHNB
Aberto a pesquisadores, com restrições da Biblioteca

Trabalho de Conclusão de Curso (Graduação) – Universidade Federal do Piauí, Bacharelado em Sistemas de Informação, Picos, 2023.
"Orientador: Me. Rayner Gomes Sousa"

1. Análise de desempenho. 2. Algoritimos. 3. Fatiamentos de redes. 4. Telefonia Móvel 5G . I. Sousa, Rayner Gomes. II. Título.

CDD 004.2

Emanuele Alves Araújo CRB 3/1290

APLICAÇÃO DE ALGORITMOS DE APRENDIZAGEM DE MÁQUINA PARA O FATIAMENTO DE REDES NO CENÁRIO 5G

JOSÉ MARIA DOS SANTOS LEAL

Monografia Aprovada como exigência parcial para obtenção do grau de Bacharel em Sistemas de Informação.

Data de Aprovação

Picos – PI, 17 de março de 2023

Rayner Gomes Sousa

Prof. Rayner Gomes Sousa

Deborah Maria Vieira Magalhães

Prof. Deborah Maria Vieira Magalhães

Romuere Rodrigues Veloso e Silva

Prof. Romuere Rodrigues Veloso e Silva

Agradecimentos

Agradeço primeiramente a Deus depois a minha mãe que me deu forças para chegar a este momento e sempre lutou para que eu tivesse uma educação melhor. Gostaria de agradecer também a todos os professores do cursos de Sistemas de Informação da UFPI campus CSHNB pelos conhecimentos passados que me possibilitaram ser um profissional e uma pessoa melhor, especialmente ao professor Rayner Gomes Sousa que me orientou durante esta jornada.

Frase.

Resumo

O 5G é a quinta geração de redes de telefonia, fornecendo suporte a ampla gama de aplicações e paradigmas com diferentes demandas de recursos da rede. Para garantir que todas essas demandas sejam atendidas, o 5G depende do serviço de criação de fatias de rede, que deve ser rápido e eficiente afim de atender todos os clientes da rede. Uma fatia é criada com base em uma requisição feita pelo usuário da rede, onde o provedor pode aceitar ou recusar a requisição de acordo com os recursos disponíveis da rede naquele momento. Para criar uma fatia de rede é necessário que o problema de mapeamento conhecido como Virtual Network Embedding (VNE) de complexidade NP-difícil seja resolvido da maneira mais otimizada possível. Este trabalho apresenta os resultados obtidos ao utilizar algoritmos de Aprendizado de Máquina durante o processo de criação de fatias de rede, comparando diferentes soluções. Além disso, propomos uma solução com parâmetros otimizados, chamada de GraphSAGE, ele utiliza os recursos da rede para realizar previsões com base nas informações agregadas das máquinas presentes na rede. Por fim, demonstramos o impacto causado pelo aprendizado de máquina na resolução do problema de VNE, analisando métricas como receita do provedor, custo do mapeamento, taxa de aceitação de requisições e a taxa de uso dos recursos da rede, onde o GraphSAGE obteve maior taxa de aceitação de requisições e receita para o provedor.

Palavras-chaves: Quinta Geração de Telefonia Móvel (5G); Aprendizado de Máquina; Fatiamento de redes; Redes virtuais.

Abstract

5G is the fifth generation of telephony networks, supporting a wide range of applications and paradigms with different demands on network resources. To ensure that all these demands are met, 5G depends on the service of creating network slices, which must be fast and efficient in order to serve all network customers. A slice is created based on a request made by the network user, where the provider can accept or reject the request according to the network resources available at that time. To create a network slice, it is necessary that the mapping problem known as Virtual Network Embedding (VNE) of NP-hard complexity be solved in the most optimal way possible. This work presents the results obtained when using Machine Learning algorithms during the process of creating network slices, comparing different solutions. In addition, we propose a solution with optimized parameters, called GraphSAGE, which uses network resources to perform forecasting based on aggregated information from the machines present in the network. Finally, we demonstrate the impact caused by machine processing in solving the VNE problem, analyzing metrics such as provider revenue, mapping cost, request reception rate and network resource usage rate, where GraphSAGE obtained the highest rate of receiving requests and revenues for the provider.

Keywords: Fifth Generation Mobile Networks (5G); Machine Learning; Network slicing; Virtual Networks.

Lista de ilustrações

Lista de tabelas

Glossário

5G Quinta Geração de Telefonia Móvel. 15, 16, 18, 33, 34

AM Aprendizado de Máquina. 16, 21–23, 27, 36–38, 40

CNN Convolutional Neural Networks. 24

D2D *Device to Device*. 15

GCN *Graph Convolutional Networks*. 23, 24, 34, 37, 40

GPU *Graphic Processing Unit*. 27, 29, 30, 34

GraphVINE Graph Neural Network Accelerated Virtual Network Embedding. 34

GRC *Global Resource Capacity*. 33

IA Inteligência Artificial. 22, 26, 27, 40

InP *Infrastructure Provider*. 18, 33

IoT Internet of Things. 15, 18

M2M *Machine to Machine*. 15

NeuroViNE *Neural Preprocessor for Your Virtual Network Embedding Algorithm*. 34

NFV *Network Function Virtualization*. 18

NS Network Slicing. 15, 18

NSaaS *Network Slicing as a Service*. 16, 18

QoS *Quality of Service*. 15, 18, 33, 40, 41

RNN Rede Neural Recorrente. 34

RS Rede de Substrato. 18

SDN *Software-Defined Networking*. 18

SLA *Service level Agreement*. 15, 19

SP *Service Provider*. 18

TCO *Total Cost of Ownership*. 15, 16

UFP *Unsplittable Flow Problem*. 21

VANET *Vehicular Ad Hoc Networks*. 15

VN *Virtual Network*. 18, 19, 30, 33

VNE *Virtual Network Embedding*. 16–21, 24, 26, 33–35

VNR *Virtual Network Requested*. 16, 18, 19

Lista de símbolos

α	Letra grega alpha
β	Letra grega beta
Σ	Letra grega sigma
Θ	Letra grega theta
$\forall$	Simbolo de para todo
$\leftarrow$	Simbolo para a esquerda
$\rightarrow$	Simbolo para a direita
$\cup$	Simbolo operação de união
$\subset$	Simbolo de relação para o subconjunto
$\in$	Simbolo de relação que indica o pertencimento
$\leq$	Simbolo de menor ou igual
$=$	Simbolo de igual

Sumário

1 Introdução

As redes de computadores operam de forma integrada, com vários sistemas distribuídos e heterogêneos, e cada aplicação requer diferentes parâmetros de *Quality of Service* (QoS) para cumprir sua missão. Para garantir a QoS, é necessário que sejam estabelecidos acordos de nível de serviço *Service level Agreement* (SLA), que devem ser suportados pela infraestrutura da rede (M.AGIWAL; ROY; SAXENA, 2016).

A diversidade de aplicativos e cenários de rede é devido a vários fatores, como a minimização e integração de circuitos digitais, a presença onipresente de redes sem fio, a disponibilidade de várias plataformas, como *Cloud Computing*, *Mobile Edge Computing* e "Conteinerização". Além disso, os novos paradigmas computacionais, como *Internet* das Coisas Internet of Things (IoT), *Device to Device* (D2D), *Machine to Machine* (M2M) e *Vehicular Ad Hoc Networks* (VANET), devem ser suportados pela Quinta Geração de Telefonia Móvel (5G).

A diversidade de aplicativos e cenários de redes se deve principalmente a: (i) minimização e integração de circuitos digitais; (ii) da presença ubíqua de redes sem fio (*wireless*); (iii) do rápido processo de implantação e disponibilidade das aplicações em razão da maturidade de várias plataformas como *Cloud Computing*, *Mobile Edge Computing* e *Containerization*; e (iv) metodologias com foco na diminuição do tempo entre o projeto e o uso, p.ex: metodologias ágeis. Ademais, recentes paradigmas computacionais que as novas gerações de redes, p.ex. Quinta Geração de Telefonia Móvel (5G), devem suportar são: IoT, comunicação D2D e M2M e VANET (TALEB et al., 2017).

Devido à diversidade de QoS, aplicações, cenários e paradigmas, os serviços de rede se tornam críticos na atualidade (FOUKAS et al., 2017). Portanto, é necessário criar uma solução que não se baseie em uma arquitetura única seguindo o modelo tradicional de rede *one-fits-all*, ou seja, uma arquitetura para todos. Esse modelo tradicional é inviável quando se considera a diversidade existente. A arquitetura atual da *Internet* enfrenta o problema de inflexibilidade, conhecido na literatura como "ossificação" (WU et al., 2020a). Por outro lado, o 5G foi projetado para lidar com o problema da ossificação na *Internet*, portanto, não segue o modelo *one-fits-all* (EJAZ et al., 2016).

O fatiamento de rede (ou Network Slicing (NS)) é uma tecnologia que provê um conjunto personalizado de recursos e serviços que atendem às necessidades específicas de uma ou mais aplicações, respeitando suas respectivas SLA. A requisição de uma fatia é feita por um cliente do provedor, a qual consta a abrangência da rede e suas características, tais como topologia, largura de banda, atraso (ou *delay*) e a confiabilidade. O custo da fatia (ou *slice*) é determinado pelos requisitos e pelo modelo de negócio do provedor, sendo financeiramente atraente, já que o cliente paga apenas pelo que utiliza, reduzindo assim *Total Cost of Ownership* (TCO). Com a oferta do fatiamento de rede como ser-

viço, conhecido por *Network Slicing as a Service* (NSaaS), as operadoras de rede móvel gerenciam todo o ciclo de vida das fatias de rede, proporcionando aos seus clientes uma redução significativa, também, de seu TCO. Essa solução é uma alternativa viável para os clientes que buscam reduzir custos e obter uma rede personalizada e adaptada às suas necessidades específicas (HAN; TAYADE; SCHOTTEN, 2017).

Para realizar o fatiamento de rede utiliza-se uma classe de algoritmos encontrados dentro da área de *Virtual Network Embedding* (VNE). O VNE consiste em um processo de mapear os nós e enlaces virtuais para nós e enlaces reais. Os nós e enlaces virtuais são solicitados por meio de pedido formal nomeado por *Virtual Network Requested* (VNR). O mapeamento é uma busca de possibilidades dentro do espaço de solução, sua complexidade é NP-Difícil por se tratar de um problema de otimização. Sua complexidade pode ser comprovada reduzindo-a ao problema *multiway separator*, conforme provado de diferentes maneiras (FISCHER et al., 2013a; VASSILARAS et al., 2017; YU et al., 2008; WU et al., 2020b). Os problemas de VNE de grandes instâncias são resolvidos com a adoção de heurísticas. Existem soluções determinísticas na literatura; no entanto, elas são usadas apenas com redes pequenas, o que não é o caso no 5G (FISCHER et al., 2013b; ZHU; AMMAR, 2006).

Levando em conta o grande grau de dinamismo presente nas redes, além de sua constante evolução, multidiversidade de aplicações que dependem do bom funcionamento das redes, torna-se cada vez mais complexo administrar e gerenciar os recursos da rede de forma automatizada (BOUTABA et al., 2018). Ademais, apesar dos notáveis resultados obtidos pelas heurísticas ao lidar com o VNE, novas abordagens usando aprendizado de máquina se mostram atrativas (SINGH et al., 2020; ABIDI et al., 2021; SSENGONZI; KOGEDA; OLWAL, 2022; DANGI; LALWANI, 2023). Dessa forma, vemos que a aplicação de técnicas de Aprendizado de Máquina (AM)(AYOUBI et al., 2018) são utilizadas na resolução do problema de VNE, promovendo a criação de uma rede flexível, que busca otimizar o controle de recursos e que atenda as mais distintas demandas.

1.1 Objetivos

O objetivo geral deste projeto é desenvolver uma nova solução para o problema de VNE, utilizando técnicas de AM e demonstrando sua eficiência na resolução desse tipo de problema.

1.2 Objetivos Específicos

Os objetivos específicos deste trabalho são:

- Avaliar a eficácia de algoritmos de AM na solução do problema de VNE, considerando a taxa de sucesso na alocação de requisições.

- Analisar o algoritmo mais eficiente para resolver o problema de VNE.
- Aperfeiçoar o processo de fatiamento da rede, tornando-o mais atrativo do que as heurísticas atualmente disponíveis.

1.3 Organização do Trabalho

Este trabalho está dividido em quatro capítulos. O Capítulo 1 contextualiza o trabalho. O Capítulo 2 apresenta os fundamentos teóricos da metodologia abordada, incluindo as tecnologias e itens envolvidos, bem como uma análise dos principais trabalhos relacionados. No Capítulo 3, é apresentada a arquitetura da solução desenvolvida, destacando suas características e funcionalidades. Finalmente, o Capítulo 4 discute os resultados obtidos com a metodologia proposta, apresentando conclusões e sugestões para trabalhos futuros.

2 Referencial Teórico

Nesta seção serão apresentados os principais conceitos inerentes ao projeto. Estes conceitos são divididos nos seguintes tópicos: (i) Sistema 5G e *Network Slicing*; (ii) *Virtual Network Embedding* - VNE; (iii) Formalização matemática do problema; (iv) Complexidade da VNE; (v) Bases de dados; e (vi) Algoritmos de aprendizagem.

2.1 Sistema 5G e Network Slicing

As redes 5G têm a ambição de suportar uma ampla gama de serviços e aplicativos. O 5G abre caminho para mercados verticais, como casas inteligentes, segurança e vigilância, transporte, industria, varejo e saúde. Cada locatário (*tenant*) definirá um conjunto de demandas que implicará no tipo de serviço oferecido pelo provedor 5G. É um fato que o 5G não projetará uma arquitetura para atender a todos os tipos de mercados verticais e diferentes aplicações. O 5G é projetado para oferecer alta flexibilidade, com base em *Software-Defined Networking* (SDN), *Network Function Virtualization* (NFV) e computação em nuvem para implantar uma rede virtual para cada locatário conforme necessário (BLANCO et al., 2017).

A infraestrutura 5G tem como componente crítico o serviço de fatiamento, que no sistema 5G é conhecido por NSaaS. Esse serviço é um mecanismo adotado para lidar com a diversidade de demandas e requisitos específicos de serviços e aplicações. O NS foi projetado como um elemento chave para permitir que o 5G suporte a IoT e outros mercados verticais. O NSaaS espera por requisições de rede virtual VNR que podem chegar a qualquer momento. Um VNR é um documento formal descrito por uma topologia de rede e demandas de QoS, por exemplo: banda, atrasos e confiabilidade (ZHOU et al., 2016).

O serviço NS é composto por um conjunto de atividades encadeadas para instanciar uma *Virtual Network* (VN), por exemplo: (i) alocação de recursos; (ii) instalação de novos serviços; (iii) orquestração de componentes de gerenciamento de rede; (iv) controles de sinalização; e outros. Dentre essas atividades, uma primordial é o mapeamento de redes virtuais para recursos físicos, matematicamente conhecido como VNE.

5G tem duas novas entidades: *Infrastructure Provider* (InP) e *Service Provider* (SP). O InP possui e gerencia a Rede de Substrato (RS) (dispositivos físicos: switches, rotas, enlaces, etc.), enquanto o SP se concentra em oferecer serviços personalizados aos clientes, podendo compartilhar vários RS. NS é um serviço prestado por um SP que visa a criação de uma VN sobre o substrato da rede. Na visão 5G, uma rede virtual é uma fatia (*slice*) e cada uma é independente e isolada. Um cliente requer uma VN para um SP por meio de uma VNR.

2.2 Virtual Network Embedding - VNE

Para efetuar o fatiamento de redes é imprescindível que a rede utilize uma classe de algoritmos conhecida por VNE. O VNE é um processo de mapeamento de um conjunto de nós virtuais para um agrupamento de nós reais e uma coleção de enlaces virtuais para um agrupamento de enlaces reais (VASSILARAS et al., 2017; FISCHER et al., 2013a; BAYS et al., 2016). A Figura 1 é uma ilustração do resultado do processo de mapeamento, a mesma é retirada do trabalho de (FISCHER et al., 2013b). No processo representando pela imagem, a rede física recebe duas VNRs com características distintas, em seguida os recursos que elas solicitam são alocadas na rede física

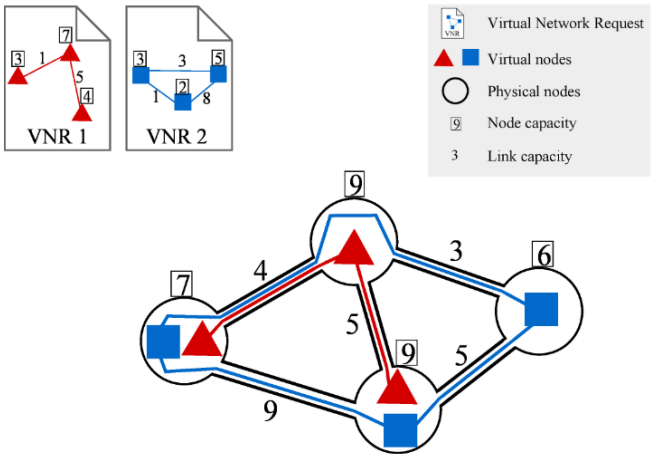

Figura 1 – Mapeamento de uma rede virtual para uma rede física. Figura retirada do trabalho de (FISCHER et al., 2013a).

O VNR está associado a um SLA específico da VN. Sob a visão das redes, os recursos associados a uma VN fazem parte dos recursos estabelecidos em sua respectiva fatia. Como o problema do VNE é NP-Difícil (ou *NP-Hard*), então o mesmo faz parte de uma classe de problemas combinatórios cuja solução boa não é calculada em tempo polinomial, ainda, a solução ótima é desconhecida. Dado que esse problema é de otimização e combinatorial, achar uma solução aceitável com um bom grau de eficácia requer o uso de heurísticas adaptadas. O trabalho de (ZHU; AMMAR, 2006) demonstra uma prova de que há um conflito (*(tradeoff)*) entre a alocação ótima de nós e a alocação ótima de enlaces, em muitos casos os dois não podem ser obtidos concomitantemente. A Figura 2 apresenta o *tradeoff* entre enlaces e nós citado anteriormente, onde seis fatias solicitam recursos da rede física, na seção "a" da imagem, o algoritmo mapeia conforme os recursos dos enlaces, entretanto, na seção "b" o mapeamento é feito com base no valor ótimo dos nós.

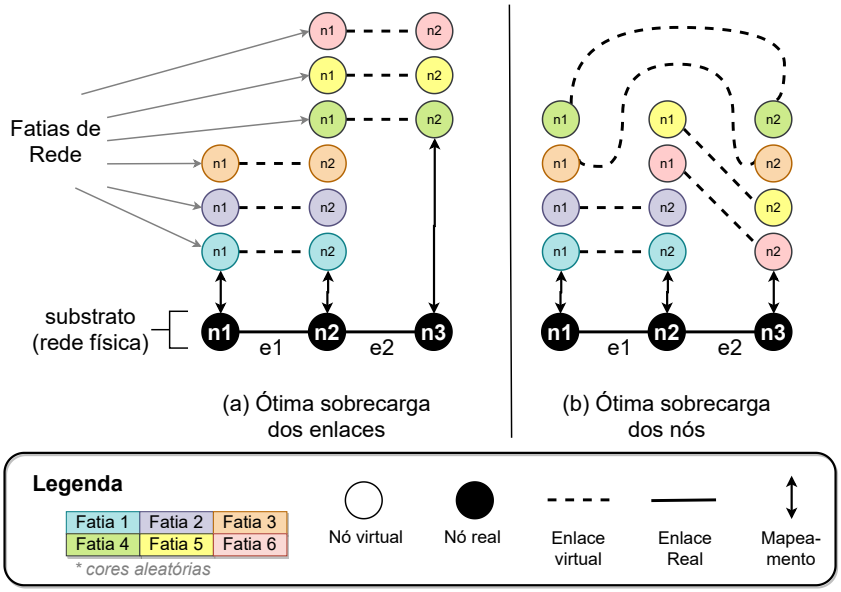

Figura 2 – Dilema entre a alocação ótimas de nós e enlaces.

2.3 Formalização Matemática

A formalização a seguir é uma forma abstrata e simplificada de definição do VNE. O problema de VNE pode ser representado em forma de grafo. Seja S = (V,A) a estrutura da rede, onde V representa um conjunto de vértices e "A" o conjunto de arestas da estrutura de rede. Seja, $R_i = (N_i, L_i)$ um agrupamento m requisições, de redes virtuais onde N_i e L_i representam um grupo de nós e enlaces virtuais de uma R_i e $0 \leq i \leq m$. Ademais, seja C um vetor de espaços para recursos e seja $cap : V \cup A \rightarrow R$ uma função que relaciona os recursos que estão disponíveis (capacidades) aos elementos da estrutura da rede, sejam eles um nó ou aresta. Por fim, para cada R_i, seja $dem_i : N_i \cup L_i \rightarrow R$ uma função que associa demandas a todos os elementos de todas as requisições.

Desse modo, VNE constitui-se de funções $f_i : N_i \rightarrow A_j$ e $g_i : L_i \rightarrow [ai, ..., aj] \subseteq A$. De forma que para cada R_i tal que $\forall n^i \in N^i : dem_i(n_i) \leq cap(f_i(n^i))$ e $\forall l^i \in L^i : \forall g_i(l^i) : dem_i(l^i) \leq cap(l)$. A função f_i corresponde ao mapeamento de nós virtuais e g_i corresponde ao mapeamento de enlaces virtuais. Juntos essas funções se tornam o mapeamento (*Embedding*) do R_i, podendo ser executadas separadamente. A Tabela 1 apresenta os conceitos descritos no texto. Uma restrição que pertence aos problemas de VNE é que dois nós virtuais dentro de uma mesma requisição não podem ser associados a um mesmo nó físico.

2.3.1 Complexidade da VNE

O problema VNE é de complexidade NP-Difícil, logo, não existe uma forma de resolvê-lo em tempo linear, apenas a checagem da solução em tempo polinomial. A complexidade do mapeamento exclusivo para os enlaces reduz para uma classe de problemas conhecidos

Tabela 1 – Terminologia usada na descrição de uma VNE

Termo	**Descrição**
S = (N,L)	S é um substrato da rede, que consiste de N nós e L enlaces
$R_i = (N^i, L^i)$	R_i denota a i^{th} solicitação de rede virtual, que consiste de N^i nós e L^i enlaces
$\dot{R} = \Pi_{j=1}^{m} R_j$	$\dot{R}$ contem vetores de recursos para todos os recursos de R_i ... R_m
cap: $N \cup L \rightarrow \dot{R}$	A função cap atribui uma capacidade a um elemento do substrato da rede (nó ou enlace)
$dem_i : N_i \cup L_i \rightarrow \dot{R}$	A função dem_i atribui uma demanda a um elemento da R_i (nó ou enlace)
$f_i : N^i \rightarrow N$	f_i é a função que mapeia um nó virtual da R_i para um nó de substrato (VNoM)
$g_i : L^i \rightarrow SN' \subseteq SN$	g_i é a função que mapeia um enlace virtual da R_i para um caminho na rede de substrato (VLiM)

como *Unsplittable Flow Problem* (UFP), que também possui complexidade NP-Difícil. Demonstrado por (ZHU; AMMAR, 2006), existe um dilema quando ocorre a tentativa de atender aos requisitos dos nós e enlaces simultaneamente. A Imagem 3 demonstra que o problema de VNE pode ser reduzido em dois caminhos, no primeiro buscamos o mapeamento ótimo de nós e no segundo, o mapeamento ótimo de enlances.

Ignorando as restrições dos nós e considerando apenas o problema de alocar de maneira otimizada um conjunto de enlaces virtuais equivale a resolver o UFP (CHOWDHURY et al., 2017). O problema clássico da mochila (knapsack) pode ser reduzido para UFP, tendo o UFP como uma só aresta e = (u, v) com capacidade c(e) = W, onde W é igual à capacidade da mochila. A demanda d_i tem o peso igual a p_i para o item i da mochila e (WU et al., 2020b).

2.4 Base de Dados

A base de dados desempenha um papel fundamental no treinamento de diversos modelos de AM, tornando a fase de formação dessa base essencial para o aperfeiçoamento dos processos de AM. Contudo, obter uma estrutura de rede verdadeira para coleta de dados reais pode ser bastante desafiador, especialmente diante do atual cenário em que há uma escassez de bases de dados disponíveis. Por essa razão, a geração de dados sintéticos tornou-se a opção mais factível para este trabalho. Para criar uma rede simulada, utilizamos a linguagem de programação *Python* em conjunto com o *framework* NetworkX [1], as características dessa rede serão descritas na Seção 3.7 deste trabalho.

[1] https://networkx.org/documentation/stable/index.html, acessado 03 de Março de 2023.

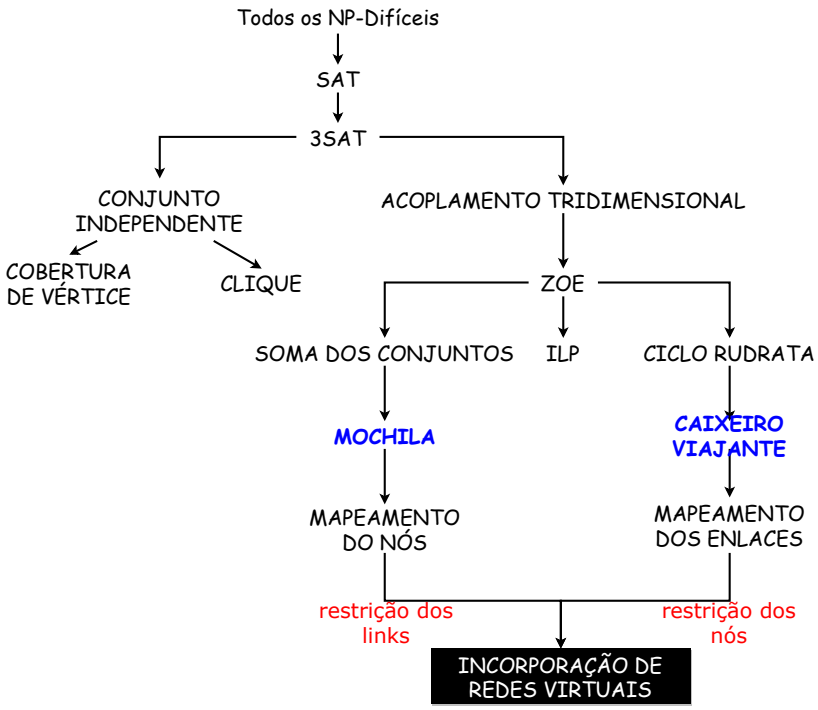

Figura 3 – Redução de problemas NP-Difícil.

2.5 Algoritmos de Aprendizagem de Máquina

A AM é uma subárea da Inteligência Artificial (IA) que permite que sistemas computacionais tomem decisões sem ou com pouca intervenção humana, por meio da análise e treinamento de modelos de dados. Seu uso é vasto e se estende por diversos setores, incluindo detecção de fraudes, filtragem de *spam*, descoberta de padrões de compras *online* e prevenção e tratamento de doenças. Devido à sua alta flexibilidade, a AM é amplamente utilizada em vários setores da economia para obter vantagens competitivas a partir da compreensão de seus dados. Conforme a pesquisa anual sobre o uso de IA realizada pela Partners (PARTNERS, 2021), apenas 4,1% das empresas entrevistadas não possuem aplicações relacionadas à IA. Portanto, o uso de técnicas de AM para gerenciar os recursos de rede é uma opção atrativa.

Os algoritmos de AM podem ser divididos em três classes distintas: aprendizagem supervisionada, aprendizagem não supervisionada e aprendizagem por reforço. Na aprendizagem supervisionada, os dados de entrada possuem rótulos. Na aprendizagem não supervisionada, os dados de entrada não possuem rótulos. Já na aprendizagem por reforço, ocorre um processo interativo em que o algoritmo considera o ambiente para determinar uma sequência correta de ações. Há uma "recompensa" para cada decisão correta e uma "penalidade" para cada decisão incorreta (AYOUBI et al., 2018; DOMEKE; CIMOLI; MONROY, 2022).

2.5.1 K-Means

O algoritmo *K-Means* representado pela Figura 4 é uma técnica de clusterização não supervisionada utilizada para agrupar dados baseado em suas características. O método compara cada valor de cada linha utilizando a distância para avaliar a distância entre cada ocorrência. Em seguida, o algoritmo calcula os centroides para cada grupo de dados. Durante a iteração, o valor de cada centroide é ajustado pela média dos valores de cada atributo das ocorrências que pertencem a esse centroide. Assim, o algoritmo produz K centróides e agrupa as ocorrências de acordo com sua distância dos centróides. Os motivos da escolha do *K-Means* foram que ele tem várias vantagens, incluindo brevidade, eficiência e celeridade,um baixo custo computacional (ABDULLAH et al., 2022), além disso, ele é um algoritmo que sofreu aplicações em outros trabalhos que lidam com o fatiamento de redes, como (SINGH et al., 2020) e (SAIBHARATH; MISHRA; HOTA, 2023).

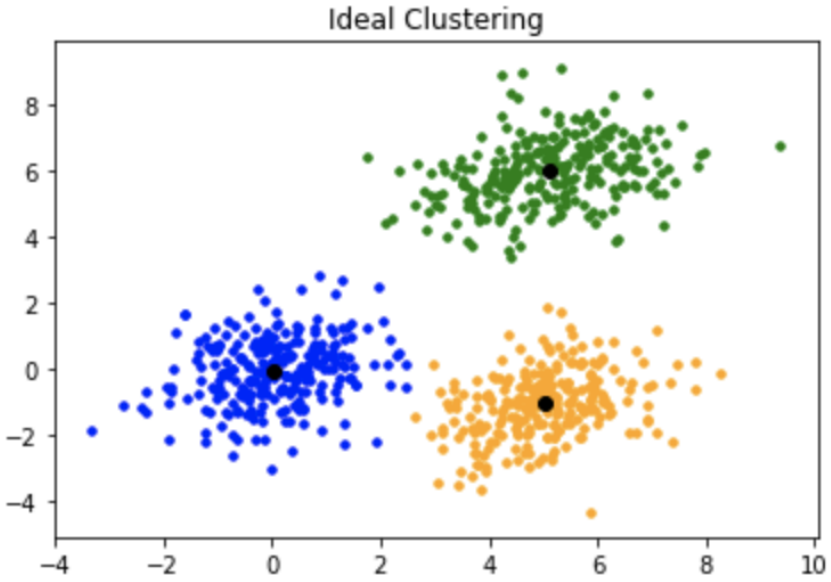

Figura 4 – Exemplo de uso do algoritmo K-Means com k=3 com cada grupo é representador por uma cor.

2.5.2 Redes Convolucionais de Grafos

As *Graph Convolutional Networks* (GCN) redes convolucionais de grafos, são algoritmos de AM aplicados a dados estruturados no formato de um grafo. As GCN aprendem a representação do grafo através de uma mesma transformação linear é aplicada a todos os vizinhos de um nó. Essas redes são amplamente utilizadas em diversas áreas, como sistemas de recomendação, processamento de imagens, previsão de tráfego e análise social. As GCN podem ser divididas em dois modelos de aprendizado, transdutivo e indutivo. No modelo transdutivo já possuímos os dados de teste e de treino do modelo, na abordagem

indutiva temos somente os dados de treinamento e aplicamos o modelo a um conjunto de dados de teste nunca visto.

A Figura 5 compara a estrutura uma entre as Convolutional Neural Networks (CNN) e uma GCNs, enquanto primeira só pode ser aplicada em dados euclidianos (regulares e estruturados), fugindo da estrutura dos dados reais, a segunda pode ser aplicada em conjuntos de dados irregulares (não euclidianos). Este tipo de estrutura possibilita que as GCN sejam aplicadas ao problema de VNE, observando que a rede física em si é uma grande estrutura de dados irregulares que sofrem alterações conforme seus recursos são alocados pelas requisições.

O funcionamento de uma GCN pode ser dividido em três etapas: entrada, propagação e transformação de características e saída. Como em uma rede neural convolucional simples, a GCN aprende uma nova representação para cada nó x_i em várias camadas, sendo subsequentemente usada como entrada em um classificador linear. Para a k-ésima camada de convolução do grafo, a entrada das representações de nós de todos os nós é denotada pela matriz $H^{(k-1)}$, e as representações de nó de saída são denotadas por H^k. A Equação 2.1 se refere à entrada dos recursos na GCN (WU et al., 2019).

Na GCN, a propagação dos recursos nas camadas ocultas h_i de cada nó v_i é feita com base em um cálculo utilizando vetores de recursos de sua vizinhança local. Esse cálculo pode ser reduzido a uma equação de multiplicação de matriz simples na Equação 2.2, em que $\mathbf{S}$ representa a matriz de adjacência normalizada dos nós do grafo. Na etapa de transformação de características, apresentada na Equação 2.3, que ocorre após a propagação, cada camada está associada a uma matriz de peso $\Theta^{(k)}$, e as representações obtidas são anteriormente transformadas linearmente. Por fim, a função de ativação ReLU é aplicada para obter novas representações do recurso $\mathbf{H}^k$ e gerar uma saída.

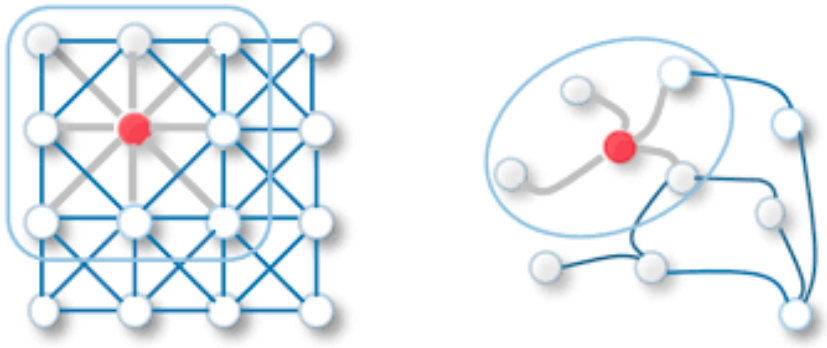

Figura 5 – Estrutura dos dados de uma CNN e GCN.

$$\mathbf{H}^{(0)} = \mathbf{X} \tag{2.1}$$

$$\tilde{\mathbf{H}}^{(k)} \leftarrow \mathbf{S}\mathbf{H}^{(k-1)} \tag{2.2}$$

$$\mathbf{H}^{(k)} \leftarrow ReLU(\tilde{\mathbf{H}}^{(k)}\Theta^{(k)}) \tag{2.3}$$

2.5.3 GraphSAGE

A maioria das estruturas de incorporação são inerentemente transdutivas, e só podem gerar incorporações para um único grafo estático. Essas abordagens transdutivas não generalizam eficientemente para nós invisíveis, por exemplo, em grafos em evolução, e não conseguem aprender a generalizar em diferentes grafos. Diante deste problema, escolhemos o GraphSAGE, que é um *framework* para aprendizado de representação indutiva em grafos grandes. Ele é usado para gerar representações vetoriais de baixa dimensão para nós e é especialmente útil em grafos que possuem informações ricas sobre atributos de nós (HAMILTON, 2017). Representações vetoriais de baixa dimensão de nós em grafos grandes têm inúmeras aplicações em aprendizado de máquina, como classificação de nós, agrupamento e previsão de *link*.

O GraphSAGE aproveita as informações de atributos do nó para gerar representações eficientes para dados não vistos anteriormente. A Figura 6 ilustra como o GraphSAGE funciona. Na primeira etapa o GraphSAGE analisa a amostra dos vizinhos do nó, na segunda etapa, ele agrega informações das características dos vizinhos, na etapa final ocorre a previsão do contexto do grafo e do rótulo com base nas informações agregadas anteriormente.

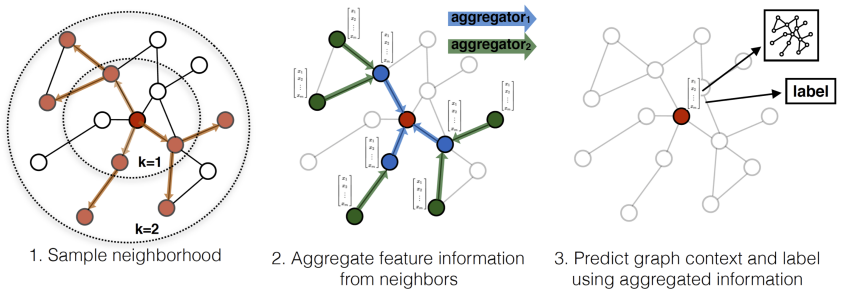

Figura 6 – Funcionamento do GraphSAGE.

2.6 Considerações Finais do Capítulo

Neste capítulo, apresentamos o contexto em que nosso projeto se encontra, juntamente com os conceitos e definições necessárias para sua compreensão. No próximo capítulo, descreveremos os aspectos de desenvolvimento e as funcionalidades que foram implementadas.

3 Desenvolvimento do Projeto

Este capítulo apresenta as principais tecnologias utilizadas na solução proposta neste trabalho, bem como a arquitetura adotada para alcançar o seu objetivo. Serão descritas as ferramentas e *frameworks* empregados, bem como suas respectivas funcionalidades e importâncias na solução final. Além disso, serão apresentados os principais componentes da arquitetura utilizada, ilustrando como eles interagem entre si para solucionar o problema de VNE de forma eficiente e escalável.

3.1 Modelo de Desenvolvimento

Durante o desenvolvimento da solução final, adotamos a metodologia *Scrum*, na qual foram realizadas *sprints* semanais, feitas aos Sábados e Domingos. No início de cada *sprint*, foi criado um *Sprint Backlog*, que definiu as tarefas a serem executadas ao longo da semana. Ao final de cada *sprint*, conduzimos uma reunião de revisão de *sprint*, na qual apresentamos os resultados alcançados e os obstáculos enfrentados.

3.2 Google Colaboratory

A IA tem sido amplamente aplicada em diversas áreas da computação, como redes de computadores, visão computacional e hardware. No entanto, à medida que essa tecnologia evolui, é necessário o uso de máquinas mais robustas para executar testes (GUNAWAN et al., 2020). Devido às exigências de hardware necessárias para execução local, a utilização do produto da *Google Research*, o *Google Colaboratory* [1] , tornou-se necessária, ela ocasionou a redução do tempo de execução do teste de 2 horas para 30 minutos. Essa ferramenta permite que qualquer pessoa escreva códigos em *Python* diretamente pelo navegador, sendo especialmente adequada para aprendizado de máquina e análise de dados, além de ser gratuita.

3.3 Python

A linguagem de programação *Python*[2] é uma das mais populares na atualidade, sendo conhecida por ser dinâmica, modular, interpretada e multi-paradigma (PYTHON, 2021). Com uma sintaxe simples e fácil entendimento, é amplamente utilizada por programadores, engenheiros, matemáticos e outros profissionais. A grande quantidade de bibliotecas

[1] https://https://colab.research.google.com/, acessado 03 de Março de 2023.
[2] https://docs.python.org/3/

nativas presentes na linguagem, aliada à possibilidade de instalação de novos pacotes, tornou o *Python* atraente para diversos setores do desenvolvimento *Web*, além de áreas como análise de dados e AM. De fato, em uma pesquisa realizada por (OVERFLOW, 2023) no ano de 2023, o *Python* foi apontado como a terceira linguagem de programação mais utilizada, com 48,25% de 83052 respostas. Devido a sua popularidade e versatilidade, o *Python* e outros *frameworks* presentes na linguagem foram escolhidos para serem utilizados neste projeto.

3.4 PyTorch

O *PyTorch*[3] é um *framework* de código aberto para AM usado em aplicações de IA. Ele oferece dois recursos de alto nível: computação tensor acelerada por meio de *Graphic Processing Unit* (GPU). O *PyTorch* define uma classe chamada Tensor para armazenar e operar em matrizes retangulares multidimensionais homogêneas de números. Os tensores *PyTorch* são semelhantes aos *arrays NumPy*, mas também podem ser operados em GPUs NVIDIA compatíveis com CUDA. Graças a esses recursos, o *PyTorch* foi escolhido para ajudar na construção de uma solução otimizada, acelerando os testes e comparações realizados.

3.5 Scikit-learn

Assim como o *PyTorch*, o *Scikit-learn* [4] é uma biblioteca de AM com código aberto. Ela oferece uma vasta gama de recursos eficientes para modelagem estatística, análise e mineração de dados, além de suporte para aprendizado supervisionado e não supervisionado. Sua construção foi baseada em outras bibliotecas científicas e numéricas do Python, como o *NumPy*[5], *SciPy*[6] e *Matplotlib*[7]. A facilidade de aplicação do *Scikit-learn* é um fator decisivo para sua escolha, ao eliminar a necessidade de implementar diversos algoritmos que serão usados na solução final.

3.6 Arquitetura da Solução

A Figura 7 apresenta a arquitetura da solução, esta arquitetura foi criada por (HABIBI et al., 2020), que segundo o autor, obteve resultados substanciais e pode ser dividida em cinco partes: *encoder*, *decoder*, *discriminator*, *clustering* e *embedding*. A quantidade de camadas presentes no *enconder*, *decoder* e *discriminator*. Na primeira etapa, o *encoder*

[3] https://pytorch.org, acessado 03 de Março de 2023.
[4] https://scikit-learn.org/stable/, acessado 03 de Março de 2023.
[5] https://numpy.org/doc/, acessado 03 de Março de 2023.
[6] https://docs.scipy.org/doc/scipy/, acessado 03 de Março de 2023.
[7] https://matplotlib.org/stable/index.html, acessado 03 de Março de 2023.

que têm 3 camadas, recebe a matriz de recursos X do servidor e a matriz ponderada A da rede física, em que os pesos representam a largura de banda, gerando uma distribuição que determinará as representações do servidor. A primeira camada do *encoder* possui como entrada o número de caraterísticas e as demais camadas possuem 16 neurônios, sendo cada uma dessas camadas uma *SAGEConv*, a qual é uma camada *GraphSAGE* projetada para ser aplicada em grafos grandes (HAMILTON, 2017). Os parâmetros da camada *SAGEConv* foram ajustados conforme a Tabela 2, objetivando um maior grau de generalização, otimização e flexibilidade para diferentes grafos. O *encoder* tem somente três parâmetros, que são números de neurônios para as camadas de entrada, oculta e saída com valores já descritos anteriormente. Em seguida aplicamos o algoritmo K-Means para agrupa os servidores em *clusters*.

Após determinar a distribuição de representação, uma amostra Z' é criada e alimentada no *decoder*, que consiste em um perceptron de quatro camadas de transformação linear. Os parâmetros desta camada são compostos somente pelo números de neurônios de entrada e saída, bem como um valor bias que recebe um valor "True". A camadas de entrada do *decoder* possuem 16 neurônios, já as camadas ocultas têm 16 e 8 neurônios respectivamente, por a camada de saída têm o tamanho do número de características. A função do *decoder* é avaliar a qualidade da codificação feita pelo *encoder* após decodificação e compara-las aos valores reais. Essas camadas recebem um tensor de entrada e aplica uma transformação linear a ele, mapeando-o para um espaço de recursos diferente, representado por um tensor de saída.

O *discriminator* é uma rede neural de 3 camadas de transformação linear com tamanhos de 16, 32 e 16 respectivamente, com a função de ativação ReLU, cujo objetivo é distinguir as amostras representativas da distribuição p(Z) das amostras geradas pelo *encoder*, melhorando as amostras produzidas pelo *encoder* com base em um processo de *feedback* interativo. Os parâmetros do *discriminator* são os mesmos que compõem o *decoder*, havendo diferença somente no números de neurônios. Em seguida, o *encoder* é treinado simultaneamente para gerar amostras que sigam a distribuição p(Z). A amostra Z' gerada pelo *encoder* é usada para agrupar os servidores físicos, em que servidores dentro do mesmo grupo são semelhantes entre si e diferentes dos outros grupos. Os servidores em um grupo fornecem recursos e largura de banda até um certo ponto para uma tarefa de mapeamento. Dessa forma, o custo do mapeamento não muda quando é necessário escolher entre dois servidores capazes dentro do mesmo grupo. O algoritmo utilizado para o agrupamento foi o *k-means*, com o número de *clusters* definido como 5, obtido através da aplicação do método de cotovelo.

A etapa final da solução consiste em realizar o *embedding* ou mapeamento da rede virtual na rede física, conforme descrito no Algoritmo 1. Quando uma rede virtual definida por G^t chega em um instante de tempo t, ela é enviada para o *GraphSAGE* com a rede física G_p e um servidor n. Essa função utiliza a rede física como referência para realizar

o mapeamento, empregando um mecanismo de busca em largura que se baseia em dois limites pré-definidos: $\alpha \times |N^t|$ e β. Esses limites são usados para restringir o número de servidores e a profundidade da pesquisa. Após cada mapeamento, a rede é retreinada, atualizando a matriz de representação e o *encoder*, visando alcançar um alto grau de generalização.

Parâmetro	**Descrição**	**Valor**
in_channels	Número de canais de entrada	Número de características.
hidden_channels	Número de canais de saída	16
out_channels	Número de canais de saída	16
aggr	Estratégia de agregação de mensagens (*mean, max, ou add*).	*mean*
normalize	Se deve normalizar a saída.	False
root_weight	Se deve usar um peso raiz.	True
project	Se deve projetar os dados de entrada.	True
bias	Se deve usar um viés.	False

Tabela 2 – Parâmetros da camada SAGEConv

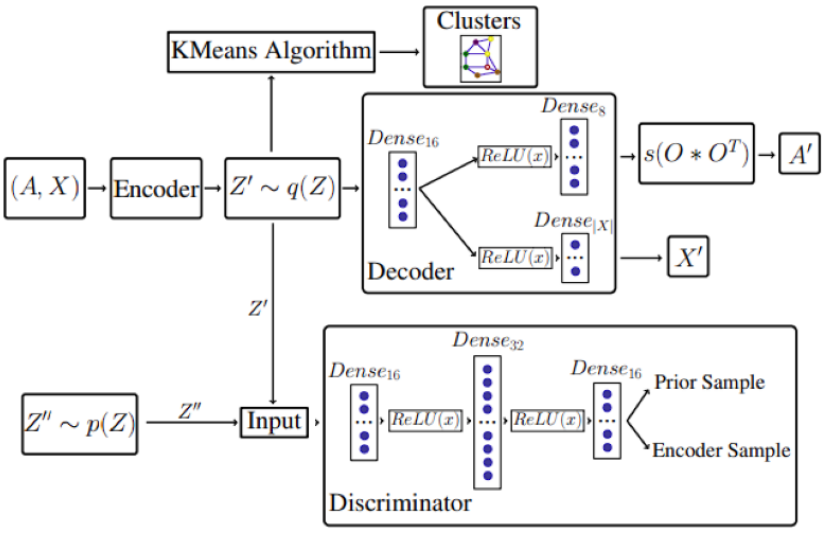

Figura 7 – Arquitetura da solução final.

3.7 Ambiente de Testes

Para realizar os testes, foi escolhida a plataforma *Google Colaboratory*, cujas especificações do ambiente de teste estão descritas na Tabela 3.7. A rede física utilizada consistiu em uma rede com 100 nós que está presente na Figura 8, estes nós simulam as máquinas da rede. Cada nó possui capacidade de CPU e GPU variando entre 100 e 400, e capacidade

Algorithm 1 Processo de mapeamento

mapeamento $((n, G^p(N^p, L^p), G^t(N^t, L^t), \alpha, \beta)$
  $q_p \leftarrow fila(n)$
  $q_t \leftarrow$fila prioritária(N^t) ▷ Prioriza nós virtuais com maior demanda de CPU
  enquanto *True* faça:
    $n^p \leftarrow$qp.remover()
    Marque n_p como visitado
    enquanto *True* faça:
      $n^t \leftarrow$qt.remover()
      se $\boldsymbol{g}(n^t) > \boldsymbol{f}(n^p)$ **então**
        Pare ▷ Recurso insuficiente
      se não há caminho para conectar n^t aos seus vizinhos **então**
        Pare ▷ Banda insuficiente
    se $dist(n^p, n) = \beta$ **então**
      continue
    counter $\leftarrow 0$
    para n'$\in$ vizinho(n^p) **e** counter $\leq (\alpha \times |N^t|)^{1/\beta}$ **faça**
      se n' não foi visitado **então**
        $qp.inserir(n')$
        counter $\leftarrow counter + 1$

de memória que varia entre 300 e 1200. As VNs utilizadas nos testes possuem diferentes números de nós, variando entre 4 e 10, com uma chance de conexão entre os nós de 70%. A CPU e a GPU têm distribuições entre 4 e 10, enquanto a memória tem uma distribuição entre 9 e 30. Cada VN tem um tempo de vida que varia entre 100 e 900 unidades de tempo. O teste dura 2000 unidades de tempo e 100 épocas de treino, com uma taxa de chegada de 2 requisições por unidade de tempo, este mesmo experimento e parâmetros foram aplicados em (HABIBI et al., 2020), com cada unidade tempo correspondendo a um ciclo completo de processamento.

Processador	*Intel(R) Xeon(R) CPU @ 2.00GHz*
Memória RAM	*12 GB*
HD	*100GB*
Placa de vídeo	*Nvidia Quadro K80*

Tabela 3 – Google Colab.

3.7.1 Grafo do substrato

A Figura 8 ilustra a complexidade do grafo utilizado para representar a infraestrutura física. Ela é o resultado do processo de criação segundo as configurações descritas anteriormente. O retângulo azul destaca a quantidade de arestas que incidem em cada nó. A Figura 9 apresenta a distribuição dos graus dos nós, com pode notar, a grande quantidade

tem cerca de 40 nós tem 15 graus, ou seja, 15 arestas incidentes. O grafo resultante tem 100 nós e 1.985 enlaces.

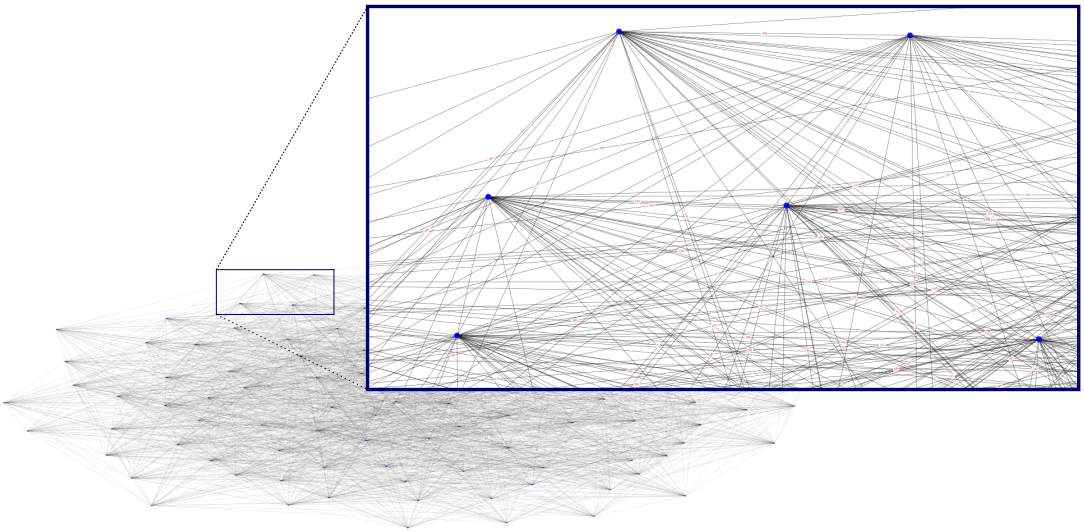

Figura 8 – Grafo do substrado da infraestrutra física.

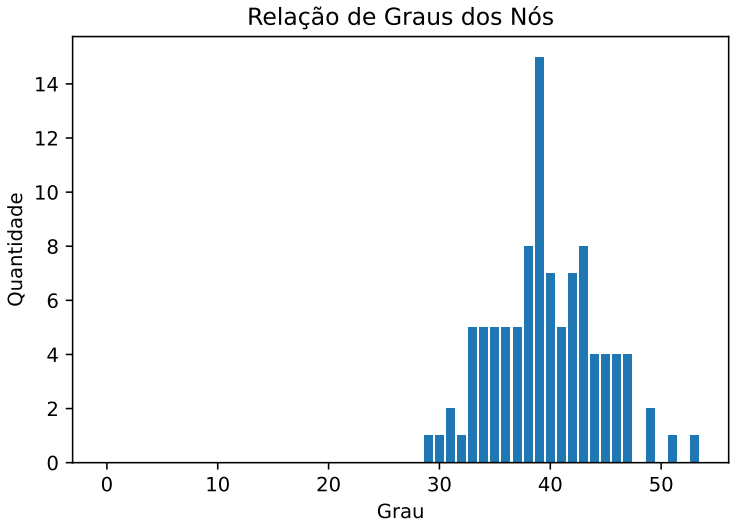

Figura 9 – Distribuição dos graus dos nós do grafo.

3.8 Métricas de Avaliação

Para analisar os resultados, foram definidas cinco métricas distintas. As variáveis utilizadas pelas fórmulas são denotadas na Tabela 4. A primeira métrica é a taxa de aceite (ta) (Equação 3.1), calculada como a razão entre o número de requisições aceitas, ou seja, mapeadas com sucesso, e o número total de requisições recebidas pelo servidor. A segunda métrica é a receita (Equação 3.2) que retorna a soma de todos os recursos das

requisições mapeadas com sucesso. A terceira métrica é o custo (Equação 3.3) que é a soma da multiplicação do número de arestas ocupadas por uma requisição pelos recursos de banda alocados. Como quarta e a quinta métricas (Equações 3.4 e 3.5) temos as taxas de utilização de CPU (tuc) e de banda (tub), respectivamente; elas refletem a capacidade do algoritmo em alocar o máximo possível de recursos da rede.

Tabela 4 – Variáveis utilizadas nas Equações sobre as Métricas de Avaliação.

Variável	**Definição**
VNR	Conjunto de todas as requisições.
R	Conjunto de todos os recursos.
M = (N^i, L^i)	Conjunto de todos as requisições mapeadas.
A	Conjunto de arestas físicas.
demais variáveis e funções	ver Tabela 1.

$$ta = \frac{\sum_{n=1}^{R_n}}{|VNR|} \tag{3.1}$$

$$receita = \sum_{n=1}^{|R|} R_n \tag{3.2}$$

$$custo = \sum_{n=1}^{|R|} R_n \times L_i, L_i \rightarrow [ai, ..., aj] \subseteq A \tag{3.3}$$

$$tuc = \frac{(1 - \sum_i^M f_i(N^i)}{\sum_i^{|N|} cap(N_i)} \tag{3.4}$$

$$tuc = \frac{(1 - \sum_i^M f_i(L^i)}{\sum_i^{|L|} cap(L_i)} \tag{3.5}$$

3.9 Coeficiente de Silhueta

Para avaliar a eficiência do algoritmo *K-Means* descrito na Seção 2.5.1, utilizamos o coeficiente de silhueta apresentado na Equação 3.6, que mede o grau de similaridade entre os *clusters* gerados pelo algoritmo; o valor a representa a distância média entre uma amostra e todos os outros pontos da mesma classe; enquanto o valor b representa a distância média entre uma amostra e os pontos do *clusters* mais próximo. O coeficiente de silhueta varia de -1 a 1, sendo que valores próximos a 1 indicam uma alta similaridade entre os membros do *clusters*.

$$coeficiente = \frac{b - a}{max(a, b)} \tag{3.6}$$

3.10 Trabalhos Relacionados

Foram aplicadas diversas soluções ao problema de VNE ao longo dos anos. No entanto, as soluções para redes menores que 50 nós não são aplicáveis ao contexto do 5G, onde vários parâmetros de QoS são considerados, ainda, é de se esperar uma infraestrutura com mais de 50 nós (GOMES; VIEIRA; CASTRO, 2022). A seguir, apresentamos algumas abordagens aplicadas ao problema de VNE.

A fim de ilustrar abordagens baseada em soluções exatas, heurísticas, em redes neurais e redes neurais gráficas, a seguir, apresentamos um resumo de alguns distintos trabalhos nessas respectivas abordagens. O trabalho de (GOMES; VIEIRA; CASTRO, 2022) faz uma revisão de meta-heurísticas aplicadas ao problema de fatiamento de redes no contexto do 5G. Como apontado pelo trabalho, após uma análise inicial de 115 trabalhos, há uma deficiência de enfrentamentos do problema de VNE usando redes neurais.

Solução Exata - As abordagens mais simples para o problema de VNE consistem em realizar o mapeamento no primeiro nó com capacidade disponível, como foi feito no trabalho de Yu (2008). Uma abordagem semelhante é a do *BestFit*, na qual os recursos solicitados pela VN são alocados nos nós com a maior capacidade de recursos disponível no momento. Apesar de sua simplicidade, essas soluções estão desatualizadas, especialmente considerando o tamanho e a quantidade de recursos das redes atuais, além da complexidade do problema presente neste cenário.

Heurística - A abordagem de (GONG et al., 2014), denominada *Global Resource Capacity* (GRC), busca maximizar a receita e minimizar o custo do InP por meio da aplicação de um balanceamento de carga guloso. O processo de mapeamento é realizado sequencialmente, onde um nó virtual é mapeado e, em seguida, é adotada a política do caminho mais curto para mapear cada *link* virtual. Embora o autor tente recriar o contexto de uma rede real, criando requisições e demandas de recursos de forma aleatória em uma

rede de 100 nós usando a ferramenta de simulação GT-ITM, a solução proposta considera apenas a banda e a capacidade dos nós, o que pode ser insuficiente para cenários reais.

Rede Neural - No que lhe concerne, o *Neural Preprocessor for Your Virtual Network Embedding Algorithm* (NeuroViNE) proposto em (BLENK et al., 2018) busca melhorar os algoritmos existentes de VNE ao reduzir o espaço de busca e pré-processar a instância do problema, extraindo subgrafos relevantes para a solução final. A solução é alimentada por uma Rede Neural Recorrente (RNN) nomeada *Hopfield*, que seleciona os melhores nós para o processo de mapeamento. Embora o autor tenha aplicado a solução em vários cenários e com diferentes modelos de distribuição de requisições, capacidade e recursos da rede, o trabalho considera apenas a capacidade dos nós e a banda presente na rede.

Rede Neural de Grafo - Já o trabalho de (HABIBI et al., 2020) utiliza GCN para agrupar os servidores e orientar o processo de incorporação em direção a um tempo de execução aprimorado e desempenho. A solução, nomeada de Graph Neural Network Accelerated Virtual Network Embedding (GraphVINE), é baseada em agrupamento e agregação para auxiliar no processo de mapeamento e considera, além da capacidade e da banda dos nós, a GPU existente em cada nó. Embora o autor busque simular a rede da maneira mais real possível, sem o uso de simuladores, o projeto ainda não é escalável o bastante para suportar o 5G.

Rede Neural de Grafo - GraphSAGE (Original e Atualizado) - O GraphSAGE original é aplicado na solução de (HAMILTON, 2017) para agregar as informações dos nós e realizar sua classificação, porém, ele não aplica seu algoritmo na solução do problema de VNE. Logo, podemos usar o GraphSAGE para resolver o problema do mapeamento, buscando gerar uma solução mais eficiente. No GraphSAGE atualizado, ajustamos seus parâmetros conforme a Tabela 2 para que ele obtenha um melhor desempenho do que sua versão original, colocamos os parâmetros *root_weight* e *project* como "True", o primeiro adiciona recursos do nó, a saída e o segundo aplica uma função linear seguida de uma função de ativação antes da agregação. Além disso, o GraphSAGE atualizado permite a projeção das representações antes da agregação.

Portanto, este projeto se torna oportuno ao apresentar uma nova abordagem que atenda a todos os pontos apresentados na Tabela 5. A Tabela apresenta uma comparação entre este trabalho e os trabalhos relacionados anteriormente citados, destacando os resultados obtidos e sem prejudicar nenhum trabalho mencionado.

Tabela 5 – Comparativo entre soluções para o problema de VNE

Algoritmo	CPU	Banda	GPU	Memória	Escalabilidade	Otimizado
NeuroVINE (BLENK et al., 2018)	Sim	Sim	Não	Não	Média	Não
GraphVINE (HABIBI et al., 2020)	Sim	Sim	Sim	Sim	Alta	Não
GraphSAGE (Original) (HAMILTON, 2017)	Sim	Sim	Sim	Sim	Alta	Não
GraphSAGE (Atualizado) (Este trabalho)	Sim	Sim	Sim	Sim	Alta	Sim

3.11 Proposta

Esta monografia apresentará uma visão abrangente sobre as diferentes soluções existentes para o problema de VNE, bem como propor uma nova abordagem para este desafio. O objetivo é desenvolver uma solução capaz de aumentar a taxa de mapeamento com baixo custo, além de solucionar o problema de VNE por meio de uma abordagem inexplorada até o momento. O resultado esperado é uma solução capaz de:

- Aumentar a taxa de mapeamento e reduzir o desperdício dos recursos da rede;
- Propor uma nova abordagem para solucionar o problema de VNE, explorando novas técnicas e ferramentas para melhorar a qualidade da solução e reduzir o tempo de execução;
- Maximizar a receita do provedor e otimizar o serviço para os clientes, buscando um equilíbrio entre o desempenho da rede e os custos de operação;
- Generalizar a solução para diferentes casos de uso e cenários, atendendo as demandas variadas dos clientes da rede e garantindo a flexibilidade e escalabilidade da solução;

4 Resultados

Este capítulo tem como proposta apresentar os resultados (Seção 4.1) obtidos neste trabalho, com base nas métricas mencionadas na Seção 3.8. A versão do algoritmo GraphSAGE aplicada em todas as comparações é a versão atualizada da solução.

4.1 Resultados

4.1.1 Coeficiente de Silhueta e Taxa de aceitação

Para avaliar a eficiência na criação dos *clusters*, utilizamos o coeficiente de silhueta, que resultou em um índice médio substancial de 0.72 (SCALDELAI; SANTOS; MATIOLI, 2022), ele foi obtido pelo somatório dos coeficientes e dividido pelo total de épocas de treino. Na Figura 10 apresentamos o gráfico da taxa de aceitação, métrica diretamente relacionada ao lucro do provedor, na Tabela 6 vemos a taxa de aceitação média de cada solução. Quanto mais requisições forem aceitas, menor será o custo e maior será a receita final. A análise do gráfico revela que os algoritmos que utilizaram AM em suas soluções obtiveram melhores resultados em relação aos demais. O GraphSAGE apresentou uma taxa de aceitação média maior com um aumento 2%, 15,4%, 45%, 20,3% e 19,7% maior em comparação com os algoritmos GraphVINE, NeuroVINE, GRC, BestFit e FirstFit, respectivamente. Além disso, o desempenho do FirstFit e do GRC foi semelhante.

Os comportamentos do GraphSAGE e GraphVINE são semelhantes pois ambas usam o método de propagar mensagens entre os nós do grafo para gerar representações dos nós, mas No GraphSAGE, a informação é propagada em camadas, onde cada camada é responsável por propagar as informações de cada nó para seus vizinhos, até que a informação se propague por todo o grafo. Já no GraphVINE, a informação é propagada através de um processo iterativo que combina as informações de cada nó com as informações de seus vizinhos em um conjunto de vinhetas, que são vetores que representam pequenas porções do grafo. Essas vinhetas são então usadas para gerar as representações dos nós.

Algoritmo	**Taxa de aceitação média**
GraphSAGE	0.823
GRC	0.373
GraphVINE	0.804
Best Fit	0.62
First Fit	0.626
NeuroVINE	0.669

Tabela 6 – Taxas de aceitação média dos algoritmos.

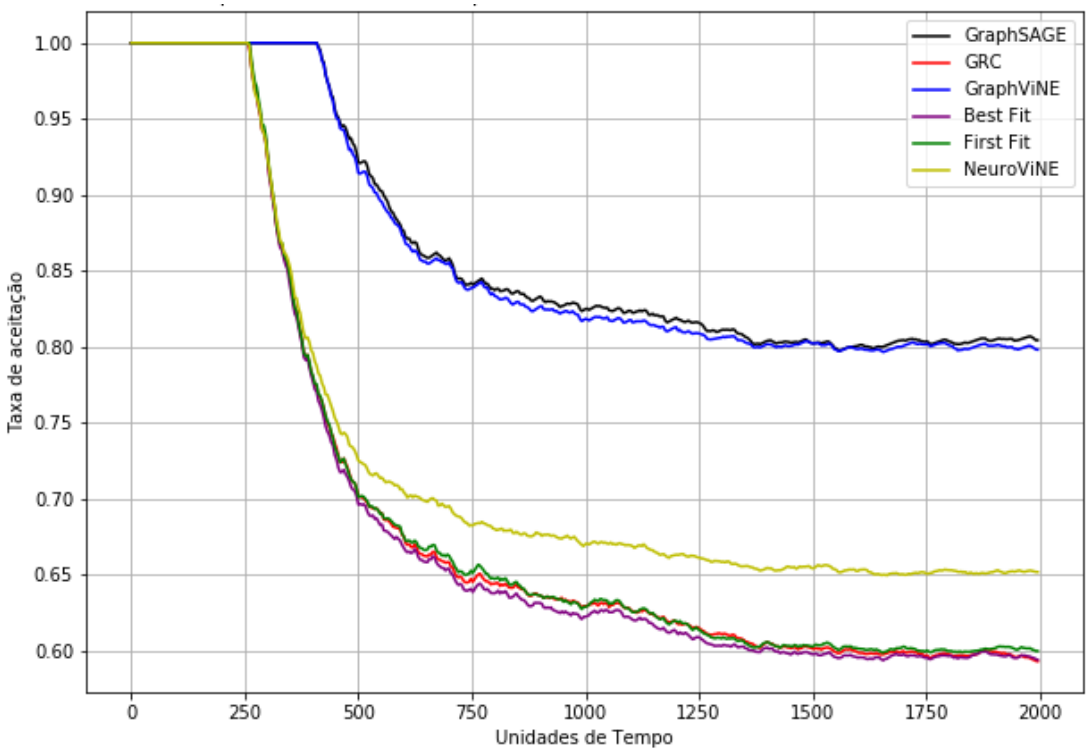

Figura 10 – Taxa de aceitação de cada algoritmo ao longo do tempo.

4.1.2 Receita e Custo

A receita é uma métrica importante relacionada diretamente à rede física, pois quanto mais recursos da rede forem alocados com sucesso, maior será a receita e o lucro do provedor. Na Figura 11 e na Tabela 7, podemos observar que os algoritmos que utilizam GCN obtiveram as maiores receitas, com um aumento de 44% na receita final em relação aos outros algoritmos. Em relação ao custo, quanto maior for o seu valor, maior é o gasto do provedor para realizar o mapeamento, podemos inferir que os algoritmos que usam AM possuem um custo semelhante a soluções mais simples (IRAWAN et al., 2020). No entanto, é importante analisar a métrica de custo de cada algoritmo, pois assim podemos avaliar a relação custo-benefício de cada solução. Na Figura 12, é possível observar o custo de cada algoritmo. Quando analisamos essas duas métricas juntas, verificamos que os algoritmos que utilizam AM apresentam uma relação custo-benefício superior em comparação às demais soluções.

4.1.3 Utilização de CPU e Banda

A taxa de utilização de CPU indica a capacidade que o algoritmo tem de alocar o máximo de recursos possíveis de um nó, apresentada na Figura 13 em conjunto com a Tabela 8, demonstra que em todas as soluções testadas, a taxa de utilização foi superior a 80%. Isso ocorre porque os algoritmos buscam utilizar o máximo de recursos disponíveis dentro da rede. Já em relação à taxa de utilização da banda, os algoritmos que utilizam

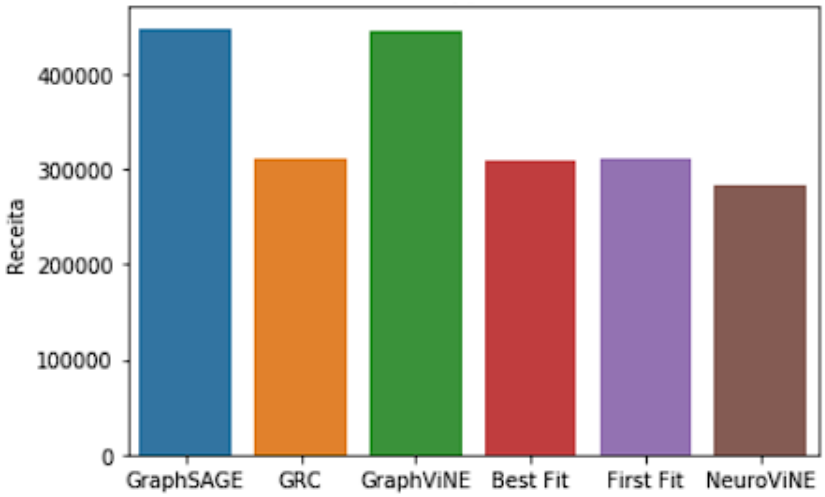

Figura 11 – Receita proveniente de cada solução.

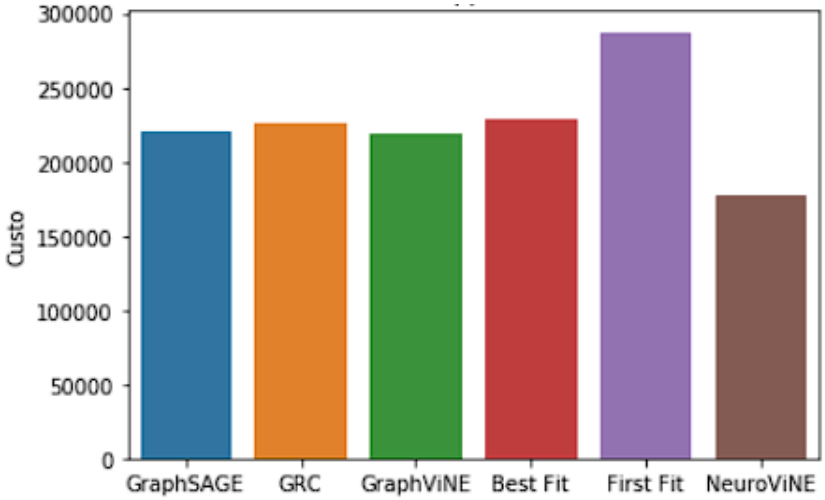

Figura 12 – Custo gerado por cada solução.

Algoritmo	**Receita**	**Custo**
GraphSAGE	447823	219865
GRC	310278	225549
GraphVINE	444273	218831
Best Fit	308823	228988
First Fit	310806	287405
NeuroVINE	283694	177263

Tabela 7 – Custo e receita dos algoritmos.

AM apresentaram as menores taxas, com valores inferiores a 10%, ocasionado pelo fator de que todas as soluções buscam a alocação ótima dos recursos CPU. Essa restrição entre a alocação ótima de recursos dos nós e enlaces deverá ser explorada em trabalhos futuros. Essa exploração deverá ser inicializada pela substituição do algoritmo que realiza a sub-rotina de mapeamento por um algoritmo mais eficiente que consiga mapear nós em diferentes sub-redes.

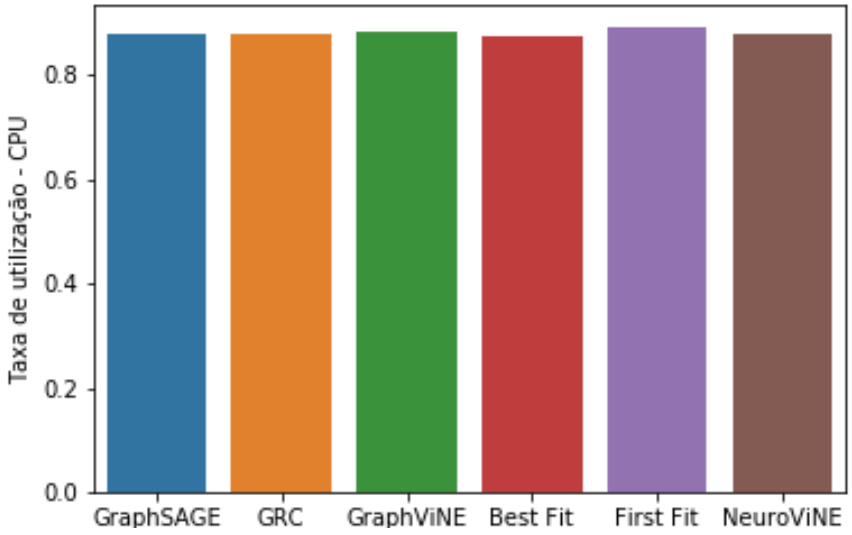

Figura 13 – Taxa de utilização utilização de CPU de cada algoritmo.

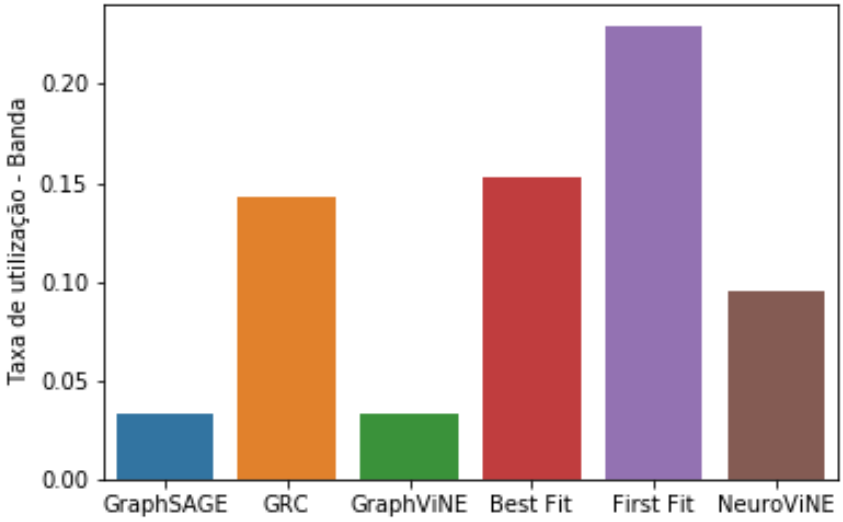

Figura 14 – Taxa de utilização de banda de cada algoritmo.

Algoritmo	**Taxa de uso da CPU**	**Taxa de uso da banda**
GraphSAGE	0.88	0.033
GRC	0.877	0.143
GraphVINE	0.881	0.034
Best Fit	0.87	0.15
First Fit	0.891	0.23
NeuroVINE	0.876	0.095

Tabela 8 – Taxa de uso da CPU e da banda dos algoritmos.

5 Conclusão e Trabalhos Futuros

O NeuroVINE e GraphVINE são modelos de *autoencoder* variacional treinados para reconstruir os dados de entrada enquanto aprendem uma representação de baixa dimensão dos dados de entrada. O GraphSAGE, por outro lado, é treinado para prever o rótulo de um nó ou outra saída específica da tarefa usando a representação do nó aprendida por meio da passagem de mensagens. Quanto ao aspecto de escalabilidade, o GraphSAGE por ser um processo indutivo se destaca por conseguir gerar saídas para dados de treino nunca antes vistos. Nossos testes revelaram uma vantagem do GraphSAGE em relação taxa de aceitação, que foi superior a 82,3% e a receita, com valor de 447823. Ademais, apesar da sofisticação do GraphSAGE, neste trabalho ajustamos alguns pontos que lhe concederam um melhoramento na sua performance, conforme a Tabela 2. Ressaltamos que a codificação dos algoritmos de alto nível, esses retirados dos respectivos artigos, foi um grande obstáculo durante a vigência do desenvolvimento.

Ao longo desta pesquisa, foram encontradas vários obstáculos, tais como a necessidade de encontrar computador com um *hardware* robusto, obter e aplicar o conhecimento relacionado a uma tecnologia recente. Este trabalho apresentou a avaliação de diferentes algoritmos que utilizam AM em suas soluções, bem como uma nova abordagem com a utilização do algoritmo GraphSAGE, que se demonstrou uma solução promissora. Após a análise dos gráficos, é possível inferir que as soluções que utilizam IA, especialmente GCN, apresentam melhores resultados em quase todas as métricas avaliadas, tendo os maiores valores em relação a taxa de aceitação de requisições e a receita, o que reforça ainda mais a aplicação de AM para solucionar o problema de mapeamento.

Existe uma lacuna para melhorias, como a utilização de heurísticas para otimizar o processo de mapeamento final e resolver o problema da sub-rede, que requer que o mapeamento de todos os nós esteja em uma mesma sub-rede, o que não reflete todas as situações da realidade. Outra questão a ser abordada é a utilização de banda, já que os algoritmos que utilizam AM têm as menores taxas de utilização da banda. Além disso, é necessário aplicar novos parâmetros de QoS para tornar a rede simulada mais próxima de uma rede real. Por fim, é importante que a solução seja aplicada em um ambiente de produção para avaliar seu desempenho em um cenário real. Com isso em mente, sugerimos as seguintes características para trabalhos futuros:

- Processo de Mapeamento: Utilizar o *GraphSAGE* em conjunto com algoritmos de mapeamento mais eficientes, como algoritmos genéticos.
- Resolver o Problema da Sub-rede: Encontrar uma solução para o mapeamento dos nós que não estejam em uma mesma sub-rede.

- Utilização de Banda: Encontrar uma forma de maximizar a utilização dos recursos de banda disponíveis na rede.
- Rede mais Complexa: Aplicar novos parâmetros de QoS para tornar a rede simulada mais próxima de uma rede real.
- Aplicação Prática: Aplicar a solução em uma rede real ou emulada e avaliar seu desempenho em um ambiente de produção.

Referências

ABDULLAH, D. et al. The application of k-means clustering for province clustering in indonesia of the risk of the covid-19 pandemic based on covid-19 data. *Quality & Quantity*, Springer, v. 56, n. 3, p. 1283–1291, 2022. Citado na página 23.

ABIDI, M. H. et al. Optimal 5g network slicing using machine learning and deep learning concepts. *Computer Standards & Interfaces*, Elsevier, v. 76, p. 103518, 2021. Citado na página 16.

AYOUBI, S. et al. Machine learning for cognitive network management. *IEEE Communications Magazine*, v. 56, n. 1, p. 158–165, 2018. Citado 2 vezes nas páginas 16 e 22.

BAYS, L. R. et al. Virtual network embedding in software-defined networks. In: *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*. [S.l.: s.n.], 2016. p. 10–18. Citado na página 19.

BLANCO, B. et al. Technology pillars in the architecture of future 5g mobile networks: Nfv, mec and sdn. *Computer Standards & Interfaces*, Elsevier, v. 54, p. 216–228, 2017. Citado na página 18.

BLENK, A. et al. Neurovine: A neural preprocessor for your virtual network embedding algorithm. In: IEEE. *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. [S.l.], 2018. p. 405–413. Citado na página 34.

BOUTABA, R. et al. A comprehensive survey on machine learning for networking: evolution, applications and research opportunities. *Journal of Internet Services and Applications*, Springer, v. 9, n. 1, p. 1–99, 2018. Citado na página 16.

CHOWDHURY, S. R. et al. ReViNE: Reallocation of Virtual Network Embedding to eliminate substrate bottlenecks. *Proceedings of the IM 2017 - 2017 IFIP/IEEE International Symposium on Integrated Network and Service Management*, IFIP, p. 116–124, 2017. Citado na página 21.

DANGI, R.; LALWANI, P. Harris hawks optimization based hybrid deep learning model for efficient network slicing in 5g network. *Cluster Computing*, Springer, p. 1–15, 2023. Citado na página 16.

DOMEKE, A.; CIMOLI, B.; MONROY, I. T. Integration of network slicing and machine learning into edge networks for low-latency services in 5g and beyond systems. *Applied Sciences*, MDPI, v. 12, n. 13, p. 6617, 2022. Citado na página 22.

EJAZ, W. et al. Internet of things (iot) in 5g wireless communications. *IEEE Access*, v. 4, p. 10310–10314, 2016. Citado na página 15.

FISCHER, A. et al. Virtual network embedding: A survey. *IEEE Communications Surveys and Tutorials*, v. 15, n. 4, p. 1888–1906, 2013. ISSN 1553877X. Citado 3 vezes nas páginas 8, 16 e 19.

FISCHER, A. et al. Virtual network embedding: A survey. *IEEE Communications Surveys Tutorials*, v. 15, n. 4, p. 1888–1906, 2013. Citado 2 vezes nas páginas 16 e 19.

FOUKAS, X. et al. Network slicing in 5g: Survey and challenges. *IEEE Communications Magazine*, IEEE, v. 55, n. 5, p. 94–100, 2017. Citado na página 15.

GOMES, R.; VIEIRA, D.; CASTRO, M. F. de. Application of meta-heuristics in 5g network slicing: A systematic review of the literature. *Sensors*, v. 22, n. 18, 2022. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/22/18/6724>. Citado na página 33.

GONG, L. et al. Toward profit-seeking virtual network embedding algorithm via global resource capacity. In: IEEE. *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*. [S.l.], 2014. p. 1–9. Citado na página 33.

GUNAWAN, T. S. et al. Development of video-based emotion recognition using deep learning with google colab. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, v. 18, n. 5, p. 2463–2471, 2020. Citado na página 26.

HABIBI, F. et al. Accelerating virtual network embedding with graph neural networks. In: IEEE. *2020 16th International Conference on Network and Service Management (CNSM)*. [S.l.], 2020. p. 1–9. Citado 3 vezes nas páginas 27, 30 e 34.

HAMILTON. Inductive representation learning on large graphs. *Advances in neural information processing systems*, v. 30, 2017. Citado 3 vezes nas páginas 25, 28 e 34.

HAN, B.; TAYADE, S.; SCHOTTEN, H. D. Modeling profit of sliced 5g networks for advanced network resource management and slice implementation. In: *2017 IEEE Symposium on Computers and Communications (ISCC)*. [S.l.: s.n.], 2017. p. 576–581. Citado na página 16.

IRAWAN, D. et al. Network slicing algorithms case study: Virtual network embedding. In: IEEE. *2020 14th International Conference on Telecommunication Systems, Services, and Applications (TSSA*. [S.l.], 2020. p. 1–5. Citado na página 37.

M.AGIWAL; ROY, A.; SAXENA, N. Next generation 5g wireless networks: A comprehensive survey. *IEEE Communications Surveys Tutorials*, v. 18, n. 3, p. 1617–1655, 2016. Citado na página 15.

OVERFLOW, S. *Most Popular Technologies*. 2023. Urlhttps://insights.stackoverflow.com/survey/2021section-most-popular-technologies-web-frameworks. Citado na página 27.

PARTNERS, N. *Releases 2021 Big Data and AI Executive Survey*. 2021. Disponível em: <https://www.businesswire.com/news/home/20210104005022/en/NewVantage-Partners-Releases-2021-Big-Data-and-AI-Executive-Survey>. Citado na página 22.

PYTHON, W. Python. *Python Releases Wind*, Citeseer, v. 24, 2021. Citado na página 26.

SAIBHARATH, S.; MISHRA, S.; HOTA, C. Joint qos and energy-efficient resource allocation and scheduling in 5g network slicing. *Computer Communications*, Elsevier, v. 202, p. 110–123, 2023. Citado na página 23.

SCALDELAI, D.; SANTOS, S. R. dos; MATIOLI, L. C. Índice de densidade da clusterização: Uma nova métrica para validação interna de agrupamentos. *Proceeding Series of the Brazilian Society of Computational and Applied Mathematics*, v. 9, n. 1, 2022. Citado na página 36.

SINGH, S. K. et al. Machine learning-based network sub-slicing framework in a sustainable 5g environment. *Sustainability*, Multidisciplinary Digital Publishing Institute, v. 12, n. 15, p. 6250, 2020. Citado 2 vezes nas páginas 16 e 23.

SSENGONZI, C.; KOGEDA, O. P.; OLWAL, T. O. A survey of deep reinforcement learning application in 5g and beyond network slicing and virtualization. *Array*, Elsevier, p. 100142, 2022. Citado na página 16.

TALEB, T. et al. On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration. *IEEE Communications Surveys Tutorials*, v. 19, n. 3, p. 1657–1681, 2017. Citado na página 15.

VASSILARAS, S. et al. The Algorithmic Aspects of Network Slicing. *IEEE Communications Magazine*, v. 55, n. 8, p. 112–119, 2017. ISSN 01636804. Citado 2 vezes nas páginas 16 e 19.

WU, F. et al. Simplifying graph convolutional networks. In: PMLR. *International conference on machine learning*. [S.l.], 2019. p. 6861–6871. Citado na página 24.

WU, H. et al. On Virtual Network Embedding: Paths and Cycles. *IEEE Transactions on Network and Service Management*, v. 4537, n. c, p. 1–14, 2020. ISSN 19324537. Citado na página 15.

WU, H. et al. On Virtual Network Embedding: Paths and Cycles. *IEEE Transactions on Network and Service Management*, v. 4537, n. c, p. 1–14, 2020. ISSN 19324537. Citado 2 vezes nas páginas 16 e 21.

YU, M. et al. Rethinking virtual network embedding: Substrate support for path splitting and migration. *Computer Communication Review*, v. 38, n. 2, p. 19–29, 2008. ISSN 01464833. Citado na página 16.

ZHOU, X. et al. Network Slicing as a Service : Enabling Enterprises ' Own Software-Defined Cellular Networks. *IEEE Communications Magazine*, v. 677, n. July, p. 146–153, 2016. ISSN 01406736. Citado na página 18.

ZHU, Y.; AMMAR, M. Algorithms for assigning substrate network resources to virtual network components. In: *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*. [S.l.: s.n.], 2006. p. 1–12. Citado 3 vezes nas páginas 16, 19 e 21.

Apêndices

APÊNDICE A – Apêndice

Trabalho submetido para o XXXI SIC: SEMINÁRIO DE INICIAÇÃO CIENTÍFICA - 2022, realizado na UFPI – Campus Ministro Petrônio Portella – Teresina – PI.

SiUFPI Seminários Integrados 2022

XXXI Seminário de Iniciação Científica

CERTIFICADO

Certificamos que **JOSE MARIA DOS SANTOS LEAL** (discente no Programa de Iniciação Científica Voluntária ICV/UFPI (IC)) e **RAYNER GOMES SOUSA** (orientador(a), UNIVERSIDADE FEDERAL DO PIAUÍ) concluíram, no período de 01/09/2021 a 31/08/2022, o trabalho intitulado **Aplicação de Algoritmos de Aprendizagem de Máquina para o Fatiamento de Redes no Cenário 5G**, apresentado na modalidade **COMUNICAÇÃO ORAL**, durante o **XXXI SIC: SEMINÁRIO DE INICIAÇÃO CIENTÍFICA - 2022**, realizado na UFPI – Campus Ministro Petrônio Portella – Teresina – PI.

TERMO DE AUTORIZAÇÃO PARA PUBLICAÇÃO DIGITAL NA BIBLIOTECA "JOSÉ ALBANO DE MACEDO"

Identificação do Tipo de Documento

( ) Tese

( ) Dissertação

(X) Monografia

( ) Artigo

Eu, **José Maria dos Santos Leal**, autorizo com base na Lei Federal nº 9.610 de 19 de Fevereiro de 1998 e na Lei nº 10.973 de 02 de dezembro de 2004, a biblioteca da Universidade Federal do Piauí a divulgar, gratuitamente, sem ressarcimento de direitos autorais, o texto integral da publicação Aplicação de Algoritmos de Aprendizagem de Máquina para o Fatiamento de Redes no Cenário 5G de minha autoria, em formato PDF, para fins de leitura e/ou impressão, pela internet a título de divulgação da produção científica gerada pela Universidade.

Picos-PI 26 de Março de 2023

José Maria dos Santos Leal

Assinatura

Assinatura