

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ
PRÓ-REITORIA DE PESQUISA, PÓS-GRADUAÇÃO E INOVAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Erik Jhones Freitas do Nascimento

Avanços em redes neurais para grafos em tarefas de predição de arestas

Fortaleza – CE

28 de julho de 2023

Erik Jhones Freitas do Nascimento

Avanços em redes neurais para grafos em tarefas de predição de arestas

Dissertação submetida ao Programa de Pós-Graduação em
Ciência da Computação do Instituto Federal de Educação,
Ciência e Tecnologia do Ceará, como requisito parcial
para obtenção do grau de Mestre em Ciência da Computação.

Linha de Pesquisa: Inteligência Artificial

Orientador: Prof. Dr. Amauri H. de Souza Jr

Coorientador: Prof. Dr. Diego P. P. Mesquita

Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE)

Pró-Reitoria de Pesquisa, Pós-Graduação e Inovação (PRPI)

Programa de Pós-Graduação em Ciência da Computação (PPGCC)

Fortaleza – CE

28 de julho de 2023

Dados Internacionais de Catalogação na Publicação
Instituto Federal do Ceará - IFCE
Sistema de Bibliotecas - SIBI
Ficha catalográfica elaborada pelo SIBI/IFCE, com os dados fornecidos pelo(a) autor(a)

- N244a Nascimento, Erik jhones Freitas do.
Avanços em redes neurais para grafos em tarefas de predição de arestas / Erik jhones Freitas do Nascimento. - 2023.
58 f. : il. color.
- Dissertação (Mestrado) - Instituto Federal do Ceará, Mestrado em Ciência da Computação, Campus Fortaleza, 2023.
Orientação: Prof. Dr. Amauri Holanda de Souza Junior.
Coorientação: Prof. Dr. Diego Parente Paiva Mesquita.
1. redes neurais para grafos. 2. predição de arestas. 3. calibração de confiança. I. Título.
- CDD 004
-



Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE)

Pró-Reitoria de Pesquisa, Pós-Graduação e Inovação (PRPI)

Programa de Pós-Graduação em Ciência da Computação (PPGCC)

Erik Jhones Freitas do Nascimento

Esta dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação, sendo aprovada pela Coordenação do Programa de Pós-Graduação em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia do Ceará e pela banca examinadora:

Orientador: Prof. Dr. Amauri H. de Souza Jr

Instituto Federal de Educação, Ciência e Tecnologia do Ceará
(IFCE)

Coorientador: Prof. Dr. Diego P. P. Mesquita

Fundação Getulio Vargas (FGV)

Prof. Dr. Joao Paulo Pordeus Gomes

Universidade Federal do Ceará (UFC)

Prof. Dr. Thiago Alves Rocha

Instituto Federal de Educação, Ciência e Tecnologia do Ceará
(IFCE)

Fortaleza – CE

28 de julho de 2023

*Este trabalho é dedicado ao meu Eu do passado e meu Eu do futuro,
que são os que realmente entendem a luta e glória de nunca desistir dos meus sonhos.*

.

Agradecimentos

Primeiramente quero agradecer ao Instituto Federal do Ceará (IFCE), a Fundação Cearense de Apoio ao Desenvolvimento Científico e Tecnológico (FUNCAP) e a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) por fornecer tanto incentivo e apoio a pesquisa de alta qualidade, apoio esse que me fez chegar até aqui. Foi um privilégio ser aluno e bolsista do IFCE.

Quero agradecer a minha melhor amiga e companheira, Ana Carolina Silva Abreu, por estar ao meu lado e sempre acreditar em mim e me ajudar a não enlouquecer e quebrar mentalmente quando as coisas ficaram difíceis. Também quero agradecer ao meu irmão, Kelvin Jonathas Freitas do Nascimento, que foi o cara que colocou comida na mesa durante os meus primeiros meses como mestrando e sem ele eu não teria tido tantos privilégios e poder me dedicar 100% ao mestrado.

Quero agradecer individualmente a cada um dos membros do meu grupo de pesquisa, grupo esse que vem me fazendo ir mais longe academicamente do que eu nunca pensei em chegar. Professor Doutor Amauri Holanda de Souza Júnior, por ter paciência comigo em diversos momentos difíceis, por respeitar minhas limitações, por me incentivar a superar essas limitações, e por me orientar tão bem durante esses anos. Professor Doutor Diego Parente Paiva Mesquita, por ser um excelente conselheiro e fornecedor de ideias e uma mente rápida para lidar com desafios teóricos tão frequentes nas nossas pesquisas. Mestra Tamara Arruda e ao Mestrando Gabriel Duarte por serem bons amigos dentro e fora do mundo da pesquisa, por ajudarem a aliviar a carga de trabalho, por serem ótimos debatedores de ideias (mesmo que pareçam malucas) e por acreditarem no futuro e potencial das nossas ideias.

E quero agradecer a mim mesmo, a pessoa que todo dia senta em uma cadeira desconfortável e pesquisa artigos científicos, que ler por horas e horas, que fica madrugadas em claro, que aguenta o estresse, agradecer por ser a pessoa que se exercita, se alimenta e continua a "nadar" nesse mar de descobertas. Quero me agradecer por ser dedicado e conseguir buscar e lutar por essa formação, por ir contra o caminho fácil de uma vida sem sentido e finalmente por estar aqui escrevendo essa dissertação.

*“Quando você elimina o impossível,
o que restar,
não importa o quão improvável seja,
deve ser a verdade”
(Sr Arthur Conan Doyle)*

Resumo

Redes neurais para grafos (*graph neural networks*, GNNs) consistem em generalizações de arquiteturas neurais clássicas (e.g., redes convolucionais) para dados estruturados na forma de grafos. Essas redes têm se tornado a abordagem mais promissora para aprendizado de representações para grafos, levando a resultados notáveis em diversas aplicações, incluindo geração de novas drogas, sistemas de recomendação, tradução automática, e geolocalização de usuário. As GNNs podem ser aplicadas para solucionar problemas de classificação de nó, classificação de grafos, geração de grafos, e predição de arestas. Neste trabalho, o foco será na tarefa de predição de arestas.

Neste trabalho são propostos avanços para a tarefa de predição de arestas ao serem apresentados dois modelos distintos. O primeiro é o *multi-hop linear graph variational autoencoder* (ML-GVAE) um modelo para predição de arestas que utiliza a combinação de múltiplos codificadores lineares para gerar uma representação latente de baixa dimensão mais precisa do grafo de entrada. O segundo é o método *on-and-off link scaling* (OOLS) um método de calibração *post hoc* para GNNs focado em aprender um vetor de temperatura baseado nas representações das arestas.

O ML-GVAE foi treinado utilizando três grafos famosos para predição de arestas e comparado contra três GNNs (i.e., VGAE, VAE, e L-GVAE). Os experimentos mostraram que o ML-GVAE, no geral, superou os métodos concorrentes. Onde o ganho mais expressivo foi para a predição das arestas no grafo PubMed com aumento de 1.9% na AUC. O OOLS foi aplicado em duas GNNs distintas treinadas com 6 grafos próprios para predição de aresta e ele foi comparado contra quatro outros calibradores. Os resultados encontrados mostram que o OOLS supera de forma majoritária todos os métodos concorrentes de calibração. Em especial, são apresentados resultados que mostram que o método OOLS gera uma calibração precisa sem afetar negativamente a AUC do modelo alvo.

Palavras-chave: redes neurais para grafos. predição de arestas. calibração de confiança.

Abstract

Graph neural networks (*graph neural networks*, GNNs) consist of generalizations of classical neural architectures (e.g., convolutional networks) for structured data in the form of graphs. These networks have become the most promising approach for learning graph representations, leading to remarkable results in several applications, including drug generation, recommendation systems, machine translation, and user geolocation. GNNs can be applied to solve node classification, graph classification, graph generation, and edge prediction problems. In this work, the focus will be on the edge prediction task.

In this work, advances are proposed for the edge prediction task by presenting two different models. The first is *multi-hop linear graph variational autoencoder* (ML-GVAE) a model for edge prediction that uses the combination of multiple linear encoders to generate a more accurate low-dimensional latent representation of the input graph. The second is the *on-and-off link scaling* (OOLS) method, a *post hoc* calibration method for GNNs focused on learning a temperature vector based on edge representations.

ML-GVAE was trained using three famous graphs for edge prediction and compared against three GNNs (i.e., VGAE, VAE, and L-GVAE). The experiments showed that ML-GVAE generally outperformed competing methods. Where the most expressive gain was for the prediction of edges in the PubMed graph with an increase of 1.9% in AUC. The OOLS was applied to two distinct GNNs trained with 6 own graphs for edge prediction and it was compared against four other calibrators. The results found show that OOLS outperforms most of all competing calibration methods. In particular, results are presented that show that the OOLS method generates an accurate calibration without negatively affecting the AUC of the target model.

Keywords: graph neural network. link prediction. confidence calibration.

Lista de ilustrações

Figura 1	–	Acima: Exemplo de uma imagem, estrutura regular, e sua representação como grafo. Note como a conectividade entre os nós segue um padrão bem definido. Abaixo: Exemplo de uma molécula, estrutura irregular e complexa, e o grafo que a representa.	29
Figura 2	–	O processo de passagem de mensagem e atualização de $\mathbf{H}$. Aqui, uma cor mais escura de um nó indica o recebimento das mensagens da camada anterior vindas dos seus vizinhos.	31
Figura 3	–	Fluxograma tarefa previsão de aresta. A matriz $\mathbf{H}$ resultante passa por uma função de <i>readout</i> para extração dos <i>logits</i>	32
Figura 4	–	Fluxograma MLGVAE. A etapa de codificação utiliza camadas SGCs empilhadas e combinadas utilizando concatenação ou soma.	37
Figura 5	–	Diagrama de confiança de uma GNN para predição de aresta não calibrada.	42
Figura 6	–	Diagramas de confiabilidade de GNNs para predição de aresta. Pode ser notada uma tendência geral de previsões super confiantes (gráficos abaixo da diagonal) para valores de $p_{uv} > 0.5$ (classe 1). No entanto, a tendência muda quando $p_{uv} < 0.5$, apresentando comportamento sub-confiante.	46
Figura 7	–	Comparação entre os diagramas de confiança do GraphSAGE treinado com o grafo Cora antes e depois da calibração com os métodos analisados. O OOLS possui o diagrama mais próximo da reta diagonal, o que indica uma melhor calibração.	50

Lista de tabelas

Tabela 1	– Previsão de aresta usando recursos de nó. ML-G(V)AEs consistentemente superam o (V)GAE baseado em GCN e o Linear G(V)AE em todos os casos.	38
Tabela 2	– Previsão de aresta sem recursos de nó. Nossos codificadores de multi-saltos lineares obtêm maior AUC e AP para todos os conjuntos de dados. A lacuna de desempenho é particularmente maior no Pubmed.	38
Tabela 3	– Estatísticas dos datasets utilizados nos nossos experimentos.	48
Tabela 4	– Performance (média e desvio padrão para AUC e Hits@20) das GNNs que foram calibradas. Em adicional também é apresentado o ECE antes da calibração. No geral, quanto maior o AUC do modelo, menor é o ECE. Isso é explicado pelo fato de modelos com AUC maiores terem uma menor entropia (NLL).	49
Tabela 5	– Erro de calibração esperado (média e desvio padrão). Em negrito estão destacados os melhores resultados do ECE encontrado para cada GNN.	49
Tabela 6	– Comparação do impacto que os métodos de calibração causam sobre a AUC das GNNs. No geral os métodos tradicionais tendem a causar uma ligeira redução na AUC. Enquanto o OOLS a deixa quase inalterada.	50
Tabela 7	– Hits@20 pós-calibração. Os menores valores estão destacados em negrito. BBQ , quanto Hits são os métodos mais nocivos para essas métricas. Enquanto o OOLS a melhora em vários cenários. . .	51

Lista de abreviaturas e siglas

AP	<i>Average precision</i>
RNA	Rede neural artificial
GNN	<i>Graph neural network</i>
ECE	<i>Expected calibration error</i>
BBQ	<i>Bayesian binning into quantiles</i>
AUC	<i>Area under the ROC curve</i>
ReLU	<i>Rectified linear unit</i>
MLP	<i>Multilayer perceptron</i>
SGC	<i>Simple graph convolution</i>
GCN	<i>Graph convolutional network</i>
VGAE	<i>Variational graph autoencoder</i>
LP	<i>Link prediction</i>
ML-GVAE	<i>Multihop linear graph variational autoencoder</i>
PEG	<i>Positional encoding graph</i>
SEAL	<i>Learning from subgraphs, embeddings and attributes for link prediction</i>
GraphSAGE	<i>Graph sample and aggregate</i>
VAE	<i>Variational autoencoder</i>
DRNL	<i>Double-radius node labelling</i>
DGCNN	<i>Deep graph convolutional neural network</i>
GDL	<i>Geometric deep learning</i>
OOLS	<i>On-and-off link scaling</i>
GATS	<i>Graph attention temperature scaling</i>
CaGCN	<i>Calibration graph convolutional network</i>

Lista de símbolos

Ψ	Redes neurais para grafos
G	Grafo
V	Conjunto de nós do grafo
E	Conjunto de arestas do grafo
A	Matriz de adjacência
$\tilde{A}$	Matriz de adjacência normalizada
I	Matriz identidade
X	Matriz de atributos dos nós
H	Matriz de representações
$\in$	Pertence
Ξ	Função de <i>readout</i>
σ	Função sigmoid
s	Vetor de logits
$\hat{c}$	Vetor de confianças
$\hat{q}$	Vetor de confianças calibradas
$\hat{p}$	Vetor de probabilidades
Θ	Parâmetros VGAE
Z	Espaço latente de baixa dimensão (VGAE)
W	Matriz de parâmetros
Φ	Parâmetros VGAE
$\odot$	Produto de Hadamard
U	Autovetores
Λ	Autovalores
Δ	Grafo Laplaciano
$\mathcal{M}$	Subconjunto de E
$\subset$	Contém

B_i Compartimento i (Bin)

$\mathbf{t}$ Vetor temperatura

σ_{SM} Função softmax

k Número de classes

$\mathbb{R}$ Conjunto dos reais

$\mathbf{m}_u^{(\ell)}$ Mensagens da camada ℓ

$\mathcal{N}(u)$ Vizinhança do nó u

Sumário

1	INTRODUÇÃO	25
1.1	Motivação e visão geral	25
1.2	Objetivos Geral e Específicos	27
1.3	Produção científica	27
1.4	Estrutura da Dissertação	27
2	FUNDAMENTAÇÃO TEÓRICA: PREDIÇÃO DE ARESTA UTILIZANDO REDES NEURAIS PARA GRAFOS	29
2.1	Redes neurais para grafos	30
2.2	Predição de aresta	31
2.2.1	Autocodificador variacional de grafos	32
2.2.2	<i>Graph sample and aggregate</i>	33
3	CONTRIBUIÇÃO I: MELHORANDO AUTOCODIFICADOR VARIACIONAL DE GRAFOS COM CONVOLUÇÕES SIMPLES DE MULTI-SALTOS	35
3.1	Introdução	35
3.2	Convoluções simples para grafos	35
3.3	Autocodificador (variacional) linear para grafos com multi-saltos	36
3.4	Experimentos	37
3.5	Resultados	37
3.6	Conclusão	38
4	FUNDAMENTAÇÃO TEÓRICA: CALIBRAÇÃO DE REDES NEURAIS PARA PREDIÇÃO DE ARESTAS	41
4.1	Calibração de confiança	41
4.2	Métodos de calibração	42
4.2.1	Histogram binning	42
4.2.2	<i>Isotonic regression</i>	43
4.2.3	<i>Temperature scaling</i>	43
4.2.4	<i>Bayesian binning into quantiles</i>	43
4.2.5	Pontos fracos dos calibradores tradicionais	44
5	CONTRIBUIÇÃO II: CALIBRADOR DE CONFIANÇA PARA GNNS DE PREDIÇÃO DE ARESTA	45
5.1	GNNS para predição de aresta são descalibradas?	45
5.2	<i>On-and-off link scaling</i>	45
5.3	Treinamento do calibrador	47
5.4	Experimentos	47
5.4.1	Conjuntos de dados	47
5.4.2	Baselines	48

5.5	Configurações dos experimentos	48
5.6	Resultados e discussões	48
5.7	Conclusão	51
6	CONCLUSÃO	53
	REFERÊNCIAS	55

1 Introdução

1.1 Motivação e visão geral

Aprendizado de máquina (*machine learning*, ML) é uma subárea da inteligência artificial que compreende técnicas estatísticas ou bio inspiradas para construção de modelos matemáticos capazes de lidar com incertezas a partir de dados (BISHOP; NASRABADI, 2006). Aprendizado de máquina, especialmente aprendizado profundo (*deep learning*), tem se tornado a principal abordagem para resolver problemas na área da inteligência artificial. Por exemplo, nos últimos anos, houve avanços significativos em aplicações como predição de novas drogas (XIONG et al., 2021), classificação de imagens clínicas (NASCIMENTO et al., 2020), detecção de discurso de ódio (SUN et al., 2019), detecção de ataques cibernéticos (HANDA; SHARMA; SHUKLA, 2019) e geração automática de jogos (NASCIMENTO et al., 2021).

Modelos tradicionais de aprendizado profundo são normalmente aplicados a problemas em que os dados assumem valores em domínios regulares, tais como imagens ou texto. Estender esses modelos para dados irregulares ou não-euclidianos, como grafos, nem sempre é uma tarefa trivial, e esse objetivo deu origem a uma nova área chamada de *geometric deep learning* (GDL) (BRONSTEIN et al., 2017a). Nesse contexto, as redes neurais para grafos (*graph neural networks*, GNNs) (SCARSELLI et al., 2008) surgiram e se destacam como modelos capazes de aprender representações e fornecer predições para dados estruturados na forma de grafos.

As GNNs se provaram muito proeminentes em solucionar problemas de classificação de nós (KIPF; WELLING, 2016a; VELIČKOVIĆ et al., 2017; SATORRAS; HOOGEBOOM; WELLING, 2021; WU et al., 2019), classificação de grafos (KIPF; WELLING, 2016a; HSU et al., 2022), e predição de arestas (ZHANG; CHEN, 2018; NASCIMENTO; SOUZA; MESQUITA, 2021; WANG et al., 2022; KIPF; WELLING, 2016b). Esse último, em específico, vem ganhando cada vez mais atenção dos pesquisadores devido ao grande leque de aplicações, o que inclui recomendação de conteúdo (CHEN; LI; HUANG, 2005; LIU, 2022; EL-KISHKY et al., 2022; ZHANG et al., 2022), previsão de tráfego (urbano, aéreo ou digital) (ZHOU et al., 2019; LU et al., 2020; JIN et al., 2022), grafos de conhecimento (ZEB et al., 2022), e prever interação entre proteínas (BALOGH et al., 2022). Com isso, é natural que os pesquisadores busquem trazer mais avanços para essa área para tornar as tarefas de predição de aresta mais precisas e confiáveis possível.

Quando falamos de precisão, métodos como codificador posicional para grafos (*positional encoder graph*, PEG) (WANG et al., 2022) e autocodificador variacional para grafos (*variational graph autoencoder*, VGAE) (KIPF; WELLING, 2016b) fazem parte do estado da arte para predição de arestas. VGAE, em específico, utiliza GNNs de passagem de mensagens de duas camadas conhecida como *graph convolutional network* (GCN) (KIPF; WELLING, 2016a) para gerar uma representação latente de baixa dimensão do grafo. No entanto, um dos problemas enfrentados pelo VGAE é o aumento considerável da complexidade do método quando aumentado o número de camadas das GCNs internas. Com isso em mente, a primeira contribuição desse trabalho é o desenvolvimento do método *multi-hop linear graph variational autoencoder* (ML-GVAE), uma GNN para predição de arestas baseada no modelo VGAE. O ML-GVAE utiliza uma GNN simples e linear conhecida como *simple graph convolution* (SGC) (WU et al., 2019) no lugar da GCN utilizada pelo método VGAE. Esses SGCs são combinados em camadas via soma e/ou concatenação para gerar a representação latente do grafo. Essa combinação de SGCs traz dois benefícios: o primeiro é a possibilidade de exploração

de uma vizinhança maior do grafo, o que permite uma maior capacidade preditiva; e o segundo é a baixa complexidade do modelo, uma vez que o SGC é linear.

No entanto, em muitas aplicações, especialmente em sistemas críticos (AMODEI et al., 2016), mais do que um modelo com boa precisão, também é altamente desejado um modelo confiável. Normalmente, um modelo confiável implica que ele deve saber quando é provável estar incorreto, ou seja, a confiança associada ao rótulo da classe predita deve refletir a verossimilhança da verdade fundamental (*ground truth*) (GUO et al., 2017). Por exemplo, em um cenário de direção autônoma, o sistema adotará a previsão fornecida pelo modelo somente quando este possuir alta confiança para sua previsão. Caso contrário, o poder de decisão será devolvido ao motorista, o sistema adotará outras estratégias mais seguras. Com o intuito de suavizar essas desconfiças associadas aos modelos de aprendizado de máquina, tem-se notado nos últimos anos um interesse crescente em tópicos como métodos de calibração de confiança (ZADROZNY; ELKAN, 2002; PLATT et al., 1999; NAEINI; COOPER; HAUSKRECHT, 2015; ZADROZNY; ELKAN, 2001).

Como GNNs para predição de arestas são, em geral, modelos profundos e pouco interpretáveis, isso as torna naturalmente pouco confiáveis (WANG et al., 2021; GUO et al., 2017). Para contornar essa falta de confiança, uma estratégia popular é utilizar métodos calibradores *post hoc*. A ideia é aumentar a confiança do modelo alterando as probabilidades associadas a cada amostra via transformações matemáticas.

Recentemente, houve um aumento de modelos propostos para calibrar GNNs, tais como *GCN as a calibration function* (CaGCN) (WANG et al., 2021) e *graph attention temperature scaling* (GATS) (HSU et al., 2022). Esses calibradores utilizam modelos neurais para transformar o espaço de saída das GNNs alvo de tal forma que o erro de calibração esperado (*expected calibration error*, ECE) seja o mínimo possível. No entanto, CaGCN e GATS são aplicados somente para GNNs especializadas em classificação de nós. Enquanto o primeiro utiliza uma GCN para gerar a calibração dos modelos, forçando a GNN de entrada a ser estruturada somente para classificação de nó. O segundo utiliza conceitos de mecanismos de atenção que são aplicados especificamente sobre o conjunto de nós do grafo. CaGCN e GATS assumem uma estrutura rígida nos dados de entrada, o que não permite uma generalização para a tarefa de predição de arestas. Além disso, uma grande desvantagem de usar os métodos tradicionais para calibrar GNNs é que eles não levam em consideração a estrutura topológica dos grafos (e.g., vizinhança dos nós), conseqüentemente, uma calibração imprecisa pode ser gerada, mesmo quando o ECE é baixo. As pesquisas realizadas na literatura, até o momento da escrita desta dissertação, não mostraram indícios de que existam métodos de calibração focados em GNNs para predição de arestas.

Dito isso, a segunda contribuição deste trabalho é a proposta de um novo método de calibração de GNNs para predição de arestas chamado de *on-and-off link scaling* (OOLS). O método OOLS utiliza as representações das arestas providas pela GNN alvo (das quais é sabido conter as informações estruturais do grafo) para aprender um vetor de temperatura utilizado para transformar as confianças originais em confianças calibradas.

Experimentos com o OOLS aplicado a duas GNNs para predição de arestas treinadas em seis conjuntos de dados mostraram que ele supera as abordagens tradicionais de calibração em termos de ECE. Notavelmente, o OOLS não só consegue realizar uma boa calibração, como também não afeta negativamente as métricas de avaliação das GNNs (i.e., AUC, Hits@20). Além disso, resultados empíricos mostram que ML-GVAE é competitivo com, se não melhor, seus concorrentes VGAE e VAE em termos de AUC e precisão.

1.2 Objetivos Geral e Específicos

Esta dissertação tem como objetivo geral *melhorar a capacidade preditiva e confiabilidade de redes neurais para grafos em tarefas de predição de arestas*.

Em seu sentido mais particular, os objetivos específicos desta dissertação são:

- Avaliar se a utilização de modelos lineares para grafos consegue melhorar a acurácia de abordagens baseadas em autocodificadores;
- Verificar se as predições para arestas obtidas a partir de GNNs que compõem o estado da arte são descalibradas;
- Desenvolver métodos de calibração para predições de arestas obtidas a partir de GNNs;
- Avaliar, na forma de resultados experimentais, a qualidade dos métodos propostos em problemas relevantes do mundo real.

1.3 Produção científica

Durante este projeto de mestrado, os seguintes trabalhos científicos foram aceitos e publicados:

- do Nascimento, E. J. F.; Souza, A. H.; Mesquita, D. (2021). **Improving Graph Variational Autoencoders with Multi Hop Simple Convolutions**. In ESANN;
- Pereira, T. A., Nascimento, E. J. F.; Mesquita, D.; Souza, A. H. **ConveXplainer for Graph Neural Networks**. IN: 11th Brazilian Conference on Intelligent Systems (BRACIS), 2022.;
- Duarte, G. J. D. S., Pereira, T. A.; Nascimento, E. J. F.; Mesquita, D.; de Souza Jr, A. H. **How do loss functions impact the performance of graph neural networks?** IN: 15th Congresso Brasileiro de Inteligência Computacional (SBIC), 2021.;
- Pereira, T. A.; Nascimento, E. J. F.; Lucas E. Resck; Mesquita, D.; Souza, A. H. **Distill n'Explain: explaining graph neural networks using simple surrogates**. IN: 26th International Conference on Artificial Intelligence and Statistics (AISTATS), 2023.

1.4 Estrutura da Dissertação

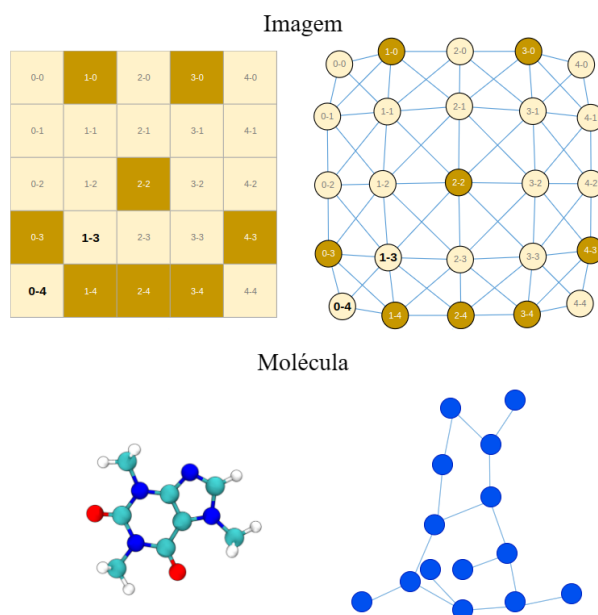
O restante do trabalho está organizado da seguinte forma: no Capítulo 2 é apresentada a fundamentação teórica das redes neurais para grafos (GNN). Além disso, são descritas as principais GNNs para predição de arestas da literatura. Em seguida, no Capítulo 3, é apresentado o método ML-GVAE. Neste capítulo também são apresentados os resultados alcançados e são feitas comparações contra outros modelos de GNNs. No Capítulo 4 são discutidas algumas técnicas de calibração tradicionais para redes neurais. Aqui também são discutidos conceitos como erro de calibração, diagramas de confiança e os pontos fracos desses métodos. No capítulo 5 é apresentado o calibrador OOLS para GNNs de predição de arestas. Também são fornecidas discussões que sustentam a eficácia do método. Finalmente, as conclusões obtidas são descritas no Capítulo 6.

2 Fundamentação teórica: predição de aresta utilizando redes neurais para grafos

Um grafo é caracterizado por uma coleção de nós e uma coleção de arestas entre esses nós (ZHANG; CHARTRAND, 2006). Dada a natureza não-regular dos grafos (i.e., não possuem representação intrínseca no espaço euclidiano), eles são uma modelagem extremamente poderosa e geral de dados do mundo real.

Com grafos é possível modelar dados regulares como imagens, textos, e frames de arquivos de vídeos, uma vez que esses tipos de objetos possuem uma conectividade bem definida entre seus componentes. Pegue os frames de um vídeo como exemplo. Eles podem ser imaginados como um grafo caminho, onde cada nó é um frame, e cada aresta conecta dois frames próximos. No entanto, dada a natureza dinâmica dos grafos, podemos representar perfeitamente qualquer fenômeno mais complexo, como relacionamentos entre pessoas (LIU et al., 2022), interações entre átomos (LI et al., 2022), redes de citação de conteúdo (WEN et al., 2022), redes de computadores (PRADHYUMNA; SHREYA et al., 2021), mapas militares (NIE; ZENG; MENG, 2020), além de sistemas críticos como sistemas de segurança, saúde e bancário (GAO et al., 2022). A Figura 1 ilustra alguns exemplos de estruturas que podem ser representadas pelos grafos.

Figura 1 – Acima: Exemplo de uma imagem, estrutura regular, e sua representação como grafo. Note como a conectividade entre os nós segue um padrão bem definido. Abaixo: Exemplo de uma molécula, estrutura irregular e complexa, e o grafo que a representa.



Fonte: Sanchez-Lengeling et al. (2021) e modificado pelo Autor.

Notações. Formalmente, um grafo não-direcionado $G = (V, E)$ é definido por dois conjuntos, sendo eles $V = \{1, \dots, n\}$, que representa o conjunto de entidades (nós), e $E \subseteq V \times V$, que representa o conjunto de relações (arestas) entre os nós. O conjunto de arestas E é geralmente representado como uma matriz de

adjacência $\mathbf{A} \in \mathbb{R}^{n \times n}$, onde $A_{ij} = 1$ se $(i, j) \in E$, e $A_{ij} = 0$ caso contrário. A matriz diagonal de graus $\mathbf{D}$ do grafo G é expressa como $D_{ii} = \sum_j A_{ij}$. Não raro, os nós do grafo G possuem atributos (características) que são expressos por uma matriz $\mathbf{X} \in \mathbb{R}^{n \times d}$. Normalmente é utilizada a versão normalizada da matriz $\mathbf{A}$ com *self-loops* definida como $\tilde{\mathbf{A}} = (\mathbf{D} + \mathbf{I})^{-1/2}(\mathbf{A} + \mathbf{I})(\mathbf{D} + \mathbf{I})^{-1/2}$, onde $\mathbf{I}$ é a matriz identidade.

Dado que é possível representar estruturas irregulares e complexas utilizando grafos, fica claro que não podemos utilizar redes neurais convencionais para processá-los e obter um resultado satisfatório. Dados estruturados como grafos possuem diversos atributos como homofilia, grau médio, centralidade de nós, triangularidade, etc. Tais atributos não são perfeitamente modelados por redes neurais convencionais, uma vez que elas consideram cada amostra do conjunto $\mathbf{X}$ como sendo eventos independentes. Portanto, quando lidamos com dados em formato de grafos, é necessário utilizar uma classe de redes neurais exclusivamente projetada para lidar com esse tipo de dados. A essa classe de redes neurais é dado o nome de redes neurais para grafos (*graph neural network*, GNN) (ZHANG; CHARTRAND, 2006; GILMER et al., 2017; HAMILTON, 2020).

2.1 Redes neurais para grafos

Uma rede neural para grafos (*graph neural network*, GNN) é uma transformação otimizável em todos os atributos do grafo (nós, arestas, contexto global) de modo que as simetrias do grafo (invariâncias de permutação) são preservadas (GILMER et al., 2017). A principal classe de GNNs utiliza um mecanismo de passagem de mensagem no qual as mensagens vetoriais são trocadas entre os nós e atualizadas com redes neurais (HAMILTON, 2020). Uma GNN de passagem de mensagem é construída em camadas, onde, em cada camada, é atualizada uma matriz $\mathbf{H}$ de mensagens (*embeddings*) dos nós do grafo. De modo geral, cada linha de $\mathbf{H}$ está associada a um nó u . A mensagem $\mathbf{h}_u^{(\ell)}$ é formada agregando as informações da vizinhança $\mathcal{N}(u)$ do nó u provenientes da camada $\ell - 1$. A função de atualização da matriz $\mathbf{H}$ é definida como na Equação 2.2:

$$\mathbf{H}^{(\ell)} = \text{UPDATE}^{(\ell-1)}(\mathbf{H}^{(\ell-1)}, \text{AGGREGATE}^{(\ell-1)}(\mathbf{A}, \mathbf{H}^{(\ell-1)})) \quad (2.1)$$

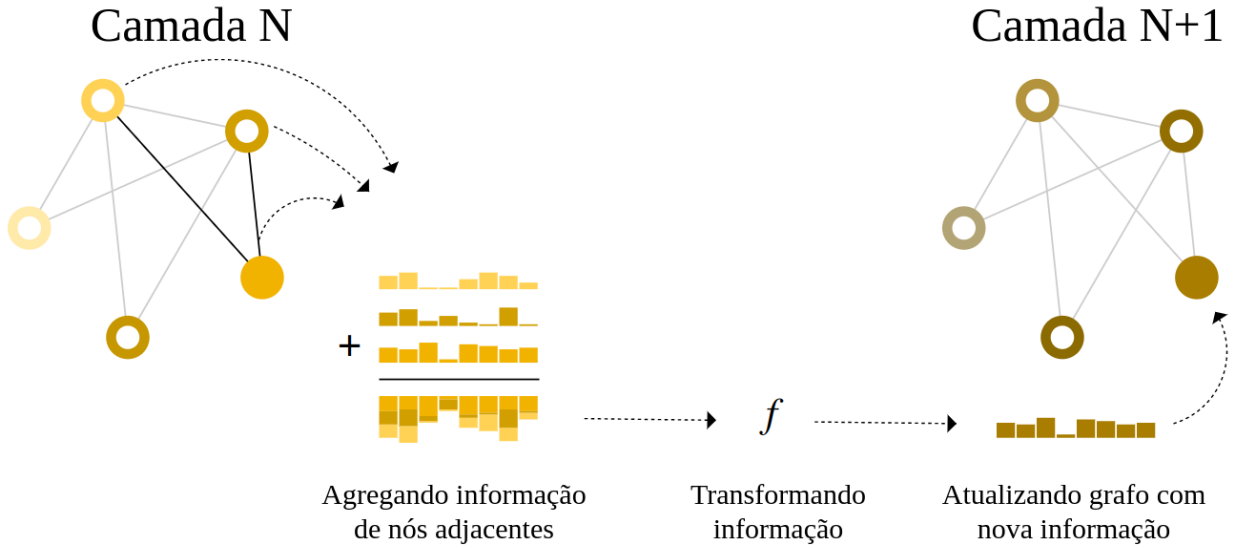
$$= \text{UPDATE}^{(\ell-1)}(\mathbf{H}^{(\ell-1)}, \mathbf{M}^{(\ell-1)}). \quad (2.2)$$

onde as funções UPDATE e AGGREGATE são funções diferenciáveis, e a u -ésima linha da matriz $\mathbf{M}^{(\ell)}$, representada como $\mathbf{m}_u^{(\ell)}$, corresponde à mensagem agregada para o nó u na camada ℓ .

O processo de passagem de mensagem é melhor ilustrado na Figura 2. Na camada ℓ , o modelo recebe as mensagens da camada anterior, essas mensagens serão processadas e passadas para a camada seguinte. Aqui, as mensagens são agregadas via soma, transformadas via redes neurais artificiais (RNAs), e atualizadas via função de atualização.

As GNNs podem ser aplicadas a uma grande diversidade de tarefas de classificação de nós (KIPF; WELLING, 2016a; VELIČKOVIĆ et al., 2017; WU et al., 2019), classificação de grafos (ZHANG et al., 2018; IVANOV; BURNAEV, 2018) ou predição de arestas (*Link prediction*, LP)(KIPF; WELLING, 2016b; HAMILTON; YING; LESKOVEC, 2017; WANG et al., 2022; ZHANG; CHEN, 2018). Daqui em diante será focado apenas na tarefa de LP, uma vez que esse é o tema desta dissertação.

Figura 2 – O processo de passagem de mensagem e atualização de $\mathbf{H}$. Aqui, uma cor mais escura de um nó indica o recebimento das mensagens da camada anterior vindas dos seus vizinhos.



Fonte: Sanchez-Lengeling et al. (2021) e modificado pelo Autor.

2.2 Predição de aresta

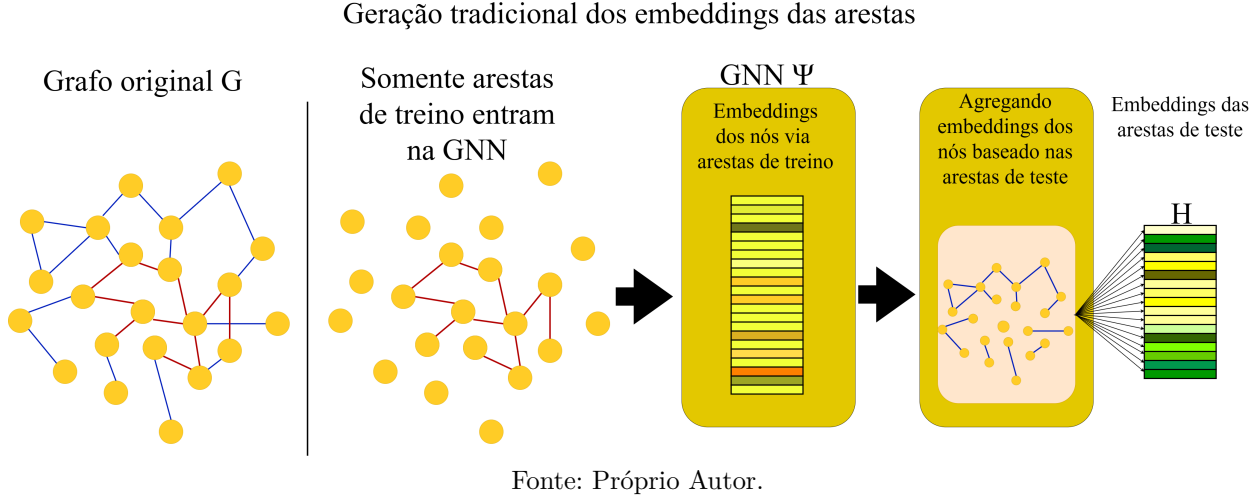
Predição de aresta (*link prediction*, LP) é um tópico de pesquisa muito importante no campo de GNNs, e um problema chave para dados estruturados como grafos. A tarefa de LP consiste em prever se uma aresta candidata (u, v) existirá, ou não, em um grafo G (LIBEN-NOWELL; KLEINBERG, 2003). Geralmente é atribuída probabilidade $p_{uv} \in [0, 1]$ para determinar a existência da aresta. Dentre os muitos problemas do mundo real que podem ser modelados como LP, podem ser citados recomendação de conteúdo (CHEN; LI; HUANG, 2005; LIU, 2022; EL-KISHKY et al., 2022; ZHANG et al., 2022), previsão de tráfego (urbano, aéreo ou digital) (ZHOU et al., 2019; LU et al., 2020; JIN et al., 2022), grafos de conhecimento (ZEB et al., 2022), e prever interação entre proteínas (BALOGH et al., 2022).

De forma simples, a tarefa de LP pode ser estruturada de acordo seguintes passos: considere as arestas no grafo como sendo exemplos positivos; amostre um número m de arestas inexistentes como sendo exemplos negativos; divida as arestas positivas e negativas em conjuntos de treino, teste e validação; treine algum modelo LP GNN com as arestas positivas; avalie o modelo utilizando alguma métrica como área sob a curva (AUC) ou HIts@k.

De modo mais formal, considere um grafo não-direcionado $G = (V, E)$ e uma aresta candidata (u, v) para a qual se quer associar uma probabilidade de ocorrência $P((u, v) \in E) = p_{uv}$. Para calcular p_{uv} , o primeiro passo consiste aplicar uma GNN para obter *embeddings* de nós para u e v , denotados por $\mathbf{h}_u^{(\ell)}$ e $\mathbf{h}_v^{(\ell)}$, respectivamente. Em seguida, os *embeddings* dos nós são combinados para formar *embeddings* de arestas via uma função ϕ de agregação invariante a ordem (e.g., produto de Hadamard), i.e., $\mathbf{h}_{uv} = \phi(\mathbf{h}_u^{(\ell)}, \mathbf{h}_v^{(\ell)})$. Finalmente, a representação de aresta $\mathbf{h}_{uv}$ é enviada para uma rede neural MLP (multilayer perceptrons) cuja saída corresponde ao *logit* s_{uv} associado a p_{uv} — i.e., $s_{uv} = \text{MLP}(\mathbf{h}_{uv})$, e $p_{uv} = \sigma(s_{uv})$, em que $\sigma(\cdot)$ denota a função sigmoide logística.

Como foi mencionado anteriormente, GNNs aplicadas à predição de aresta são avaliadas utilizando duas principais métricas: *area under the curve of the receiver operating characteristic* (AUC). A AUC mostra

Figura 3 – Fluxograma tarefa previsão de aresta. A matriz $\mathbf{H}$ resultante passa por uma função de *readout* para extração dos *logits*.



a performance de um modelo classificador para todos os valores de limiares de classificação (um limiar comumente utilizado é 0.5). A AUC é a medida da capacidade de um classificador binário de distinguir entre as classes e é usada como um resumo da curva ROC. Quanto maior a AUC, mais precisa é a GNN. Hits@k mensura a razão entre o total de verdadeiros positivos acima do vigésimo maior verdadeiro negativo e o total de verdadeiros positivos. Baseado na descrição do Hits@k, é fato que ele é bem sensível às transformações das probabilidades do modelo. Como será visto no Capítulo 4, métodos *post hoc* podem modificar negativamente os valores dessas métricas.

Neste capítulo são descritas duas das principais GNNs para predição de arestas, sendo elas o autocodificador variacional de grafos (*variational graph autoencoder*, VGAE) (KIPF; WELLING, 2016b), e *graph sample and aggregate*, (GraphSAGE) (HAMILTON; YING; LESKOVEC, 2017).

2.2.1 Autocodificador variacional de grafos

O autocodificador variacional de grafos (*variational graph autoencoder*, VGAE) (KIPF; WELLING, 2016b) é formado por uma rede codificadora e um decodificadora, com respectivos parâmetros Θ e Φ . O último parametriza a função de verossimilhança $p_{\Phi}(G_i|\mathbf{Z}_i)$ de um grafo G_i . O primeiro parametriza a distribuição variacional $q_{\Theta}(\mathbf{Z}_{1:S}) = \prod_{i=1}^S q_{\Theta}(\mathbf{Z}_i|G_i)$ que aproxima a posteriori Bayesiana:

$$p_{\Phi}(\mathbf{Z}_1, \dots, \mathbf{Z}_S | G_1, \dots, G_S) \propto \prod_{i=1}^S p_{\Phi}(G_i | \mathbf{Z}_i) p(\mathbf{Z}_i) \quad (2.3)$$

onde $p(\mathbf{Z}_1), \dots, p(\mathbf{Z}_S)$ são priores definidas pelo usuário e $\mathbf{Z}_i$ é uma representação de baixa dimensão do grafo original.

Os parâmetros (Θ, Φ) são ajustados via minimização da divergência de Kullback-Leibler (KL), uma medida que quantifica a diferença entre duas distribuições de probabilidade, entre as distribuições variacional

e posteriori na Equação 2.3:

$$\begin{aligned} (\Theta^*, \Phi^*) &=_{\Theta, \Phi} D_{\text{KL}} [q_{\Theta}(\mathbf{Z}_{1:S}|G_{1:S}) \| p_{\Phi}(\mathbf{Z}_{1:S}|G_{1:S})] \\ &\triangleq_{\Theta, \Phi} \mathbb{E}_{q_{\Theta}} \left[\log \frac{q_{\Theta}(\mathbf{Z}_{1:S}|G_{1:S})}{p_{\Phi}(\mathbf{Z}_{1:S}|G_{1:S})} \right] \end{aligned} \quad (2.4)$$

que é intratável, mas pode ser resolvido de forma eficiente maximizando o *evidence lower-bound* (ELBO):

$$\mathcal{L}(\Theta, \Phi) = \sum_{i=1}^S \mathbb{E}_{q_{\Theta}(\mathbf{Z}_i|G_i)} [-\log p_{\Phi}(G_i|\mathbf{Z}_i)] + D_{\text{KL}} [q_{\Theta}(\mathbf{Z}_i|G_i) \| p(\mathbf{Z}_i)] \quad (2.5)$$

Kipf e Welling (2016b) modelam $q_{\Theta}(\mathbf{Z}_i|G_i)$ como uma gaussiana $\mathcal{N}(\boldsymbol{\mu}_i, \text{diag}(\boldsymbol{\sigma}_i))$ e usam duas camadas *Graph Convolutional Networks* (GCNs) (KIPF; WELLING, 2016a) para o codificador. Sendo que uma camada GCN computa $\boldsymbol{\mu}_i \in \mathbb{R}^{n_i \times d}$ dado G_i , enquanto a outra retorna $\boldsymbol{\sigma}_i \in \mathbb{R}^{n_i}$. A verossimilhança $p_{\Phi}(G_i|\mathbf{Z}_i)$ é modelada como uma distribuição elementar de Bernoulli sobre a matriz de adjacência $\mathbf{A}_i$, com logits $\mathbf{Z}_i \mathbf{Z}_i^{\top}$.

2.2.2 Graph sample and aggregate

O *graph sample and aggregate* (GraphSAGE) (HAMILTON; YING; LESKOVEC, 2017) é um método de aprendizado de representação de grafos que permite extrair representações de nós em um grafo, levando em consideração a estrutura local do grafo. O método usa amostragem aleatória e agregação de informações dos nós vizinhos para calcular as representações.

Na etapa de amostragem, durante a iteração ℓ , para cada nó $v_i \in V$, é amostrado um subconjunto S_i^{ℓ} dos vizinhos de v_i . Essa amostragem pode ser feita de forma aleatória ou baseada em algum critério específico, como vizinhos mais próximos ou vizinhos ponderados. Cada vizinho $v_j \in S_i^{\ell}$ é representado por $h_j^{\ell-1}$, a representação do nó v_j na iteração anterior.

Na etapa de agregação, a representação $\mathbf{h}_u$ de um nó u é obtida ao serem agregados todos vetores de representações dos nós da vizinhança do nó u .

$$\mathbf{h}_u = f_{agg}(\mathbf{h}_v | v \in \mathcal{N}(u)) \quad (2.6)$$

onde f_{agg} é uma função diferenciável, ela pode ser tão simples quanto uma função de média (HAMILTON; YING; LESKOVEC, 2017) ou tão complexa quanto uma rede neural MLP, e $\mathcal{N}(u)$ é a vizinhança de u .

Depois de obter uma representação agregada para o nó u com base em seus vizinhos, u é atualizado usando uma combinação de sua representação anterior $\mathbf{h}_u^{l-1}$ e a representação agregada $\mathbf{h}_u$.

$$\mathbf{h}_u^l = f_{upd}(\mathbf{h}_u, \mathbf{h}_u^{l-1}) \quad (2.7)$$

onde, novamente, f_{upd} é uma função diferenciável, como uma rede neural MLP ou mesmo uma simples média, e l é o número de camadas do modelo.

A amostragem da estratégia de vizinhos no GraphSAGE produz um desempenho impressionante na tarefa de rotulagem de nós em várias redes de grande escala. Para aprender os parâmetros do agregador e atualizador, a função de perda do GraphSAGE é definida como:

$$J = - \sum_{v_i \in L} \log(p_i) \quad (2.8)$$

onde J é a função de custo, v_i é um nó rotulado, p_i é a probabilidade de atribuir a classe correta ao nó v_i . A probabilidade p_i é calculada usando a função softmax nas representações dos nós. Suponha que h_i^ℓ seja a representação final do nó v_i após ℓ iterações do GraphSage. A probabilidade p_i é calculada como:

$$p_i = \frac{\exp(z_i)}{\sum_{c=1}^C \exp(z_i^c)} \quad (2.9)$$

onde z_i é o vetor de pontuação associado ao nó v_i , z_i^c é a pontuação para a classe c calculada a partir do vetor de representação h_i^ℓ . A pontuação z_i^c pode ser obtida aplicando uma transformação linear aos vetores de representação h_i^ℓ , seguida de uma função de ativação específica para a tarefa. Por exemplo, pode-se usar uma camada densa *fully connected* seguida de uma função de ativação como ReLU ou sigmoid para obter as pontuações.

O objetivo da função de custo é minimizar o valor J , ou seja, ajustar as representações dos nós de forma a maximizar a probabilidade correta de classificação para os nós rotulados. É importante ressaltar que a função de custo pode variar dependendo da tarefa e das técnicas específicas aplicadas no GraphSage. Diferentes algoritmos de aprendizado, como regressão logística ou aprendizado por pares, podem usar diferentes funções de custo para otimizar o desempenho em tarefas específicas.

3 Contribuição I: melhorando autocodificador variacional de grafos com convoluções simples de multi-saltos

Neste capítulo, será apresentada uma das principais contribuições desta pesquisa, uma versão melhorada do método VGAE (KIPF; WELLING, 2016b) que utiliza menos parâmetros e é mais expressiva. Ao longo do capítulo, serão apresentadas justificativas fortes para o desenvolvimento do método, bem como resultados empíricos que sustentam a eficácia desta abordagem se comparada com os métodos concorrentes.

3.1 Introdução

Autocodificadores variacionais (*variational autoencoders*, VAEs) para grafos consistem em um codificador que projeta os grafos em um espaço latente de baixa dimensão e um decodificador que mapeia esse espaço latente de volta ao espaço original. Embora existam muitos tipos de VAEs (KIPF; WELLING, 2016b; SIMONOVSKY; KOMODAKIS, 2018; GROVER; ZWEIG; ERMON, 2019; JIN; BARZILAY; JAAKKOLA, 2018), a maioria dos VAEs para grafos emprega GNNs (KIPF; WELLING, 2016b; MICHELI, 2009) para o codificador. Recentemente, muitos trabalhos (WU et al., 2019; HUANG et al., 2021; CHEN et al., 2020; MESQUITA et al., 2020) mostraram que GNNs simples podem alcançar desempenho competitivo em uma variedade de tarefas de aprendizado de grafos. No contexto dos VAEs para grafos (VGAE), (SALHA et al., 2020) demonstram empiricamente que codificadores lineares de um salto (*hop*) funcionam de maneira semelhante a arquiteturas baseadas em GNNs multicamadas (KIPF; WELLING, 2016b). Aqui, o termo salto ou *hop* refere-se a vizinhança dos nós, 1-*hop* significa que estamos lidando com a vizinhança de tamanho 1 dos nós.

Neste capítulo é apresentada uma melhoria para os codificadores VGAEs (KIPF; WELLING, 2016b) aproveitando as informações de vizinhança de multi-saltos, que foram extraídas utilizando várias convoluções simples para grafos (SGCs) (WU et al., 2019). São propostas duas maneiras de combinar essas informações: concatenar as saídas do SGC e somá-las. Como esses VGAEs são projetados principalmente para lidar com tarefas de reconstrução (por exemplo, predição de aresta), então esta análise também foi estendida para autocodificadores (*autoencoders*, AEs) determinísticos. Os resultados empíricos demonstram que os métodos propostos superam VGAEs para previsão de aresta em conjuntos de dados de referência populares (Cora, Citeseer e Pubmed). Os métodos propostos também superam o modelo linear de um salto em (SALHA et al., 2020), o que indica que a combinação de informações de vários saltos é benéfica para tarefas de reconstrução.

3.2 Convoluções simples para grafos

É possível utilizar a autodecomposição da matriz Laplaciana do grafo para definir convoluções nos domínios do grafo (BRONSTEIN et al., 2017b). Sejam $\mathbf{U}$ e $\mathbf{\Lambda}$ os autovetores e autovalores do grafo Laplaciano $\mathbf{\Delta} := \mathbf{I} - \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2}$, respectivamente. A transformada de Fourier de um sinal de canal $\mathbf{X} \in \mathbb{R}^{n \times d}$ nos vértices do grafo é dada por $\hat{\mathbf{X}} = \mathbf{U}^\top \mathbf{X}$, e seu inverso é $\mathbf{X} = \mathbf{U} \hat{\mathbf{X}}$. Usando essas operações, é definida a convolução do grafo entre um sinal $\mathbf{X}$ e um filtro g como:

$$\mathbf{g} \star \mathbf{X} = \mathbf{U} ((\mathbf{U}^\top \mathbf{g}) \odot (\mathbf{U}^\top \mathbf{X})) = \mathbf{U} \hat{\mathbf{G}} \mathbf{U}^\top \mathbf{X} \quad (3.1)$$

onde $\hat{\mathbf{G}} = \text{diag}(\hat{g}_1, \hat{g}_2, \dots, \hat{g}_n)$ denota os coeficientes do filtro espectral $\hat{g}_i$.

Na prática, a abordagem não paramétrica na Equação 3.1 é indesejável, pois possui inúmeros parâmetros e não produz filtros localizados (BRUNA et al., 2014). Uma solução é aproximar o filtro com polinômios dos autovalores laplacianos, de forma que os polinômios do laplaciano levam à $\mathbf{U} \mathbf{\Lambda}^k \mathbf{U}^\top = \mathbf{\Delta}^k$.

Seja $\mathbf{H}^{(0)} = \mathbf{X}$; na camada ℓ , uma GNN espectral polinomial genérica calcula

$$\mathbf{H}^{(\ell)} = \text{ReLU} \left(\sum_{k=0}^K \mathbf{\Delta}^k \mathbf{H}^{(\ell-1)} \mathbf{\Theta}_k^{(\ell)} \right), \quad (3.2)$$

onde $\mathbf{\Theta}^{(\ell)} \in \mathbb{R}^{d_{\ell-1} \times d_\ell}$ são os coeficientes dos filtros espectrais. A saída $\mathbf{H}^{(L)} \in \mathbb{R}^{n \times d_L}$ após a camada L compreende representações para cada nó em G .

Wu et al. (2019) simplifica a camada da Equação 3.2 de quatro maneiras: i) adotando $K = 1$; ii) configurando $\mathbf{\Theta}_0^{(\ell)} = \mathbf{0}$ e $\mathbf{\Theta}_1^{(\ell)} = \mathbf{I}$; iii) removendo a função ReLU; iv) substituindo $\mathbf{\Delta}$ por $\tilde{\mathbf{A}}$. Empilhando camadas L e aplicando uma transformação linear, é obtida a *convolução simplificada de grafos* (SGC):

$$\mathbf{H} = \tilde{\mathbf{A}}^L \mathbf{X} \mathbf{\Theta}. \quad (3.3)$$

3.3 Autocodificador (variacional) linear para grafos com multi-saltos

Para melhorar o desempenho dos VGAEs, foram modificados seus codificadores por I) substituindo os GCNs de duas camadas por SGCs; e II) usando SGCs de ordem diferente para extrair efetivamente informações multi-salto. Este método é referido como autocodificador variacional linear para grafos com multi-saltos (*multi-hop linear graph variational autoencoder*, ML-GVAE). Além disso, foi proposto combinar os múltiplos SGCs de duas maneiras: somando e concatenando suas saídas. Os codificadores lineares resultantes são:

$$\text{ML-GVAE (sum):} \quad \boldsymbol{\mu}_i = \sum_{c=1}^C \tilde{\mathbf{A}}_i^c \mathbf{X}_i \mathbf{\Theta}_\mu^{(c)}, \quad \log \sigma_i = \sum_{c=1}^C \tilde{\mathbf{A}}_i^c \mathbf{X}_i \mathbf{\Theta}_\sigma^{(c)}; \quad (3.4)$$

$$\text{ML-GVAE (cat):} \quad \boldsymbol{\mu}_i = \left\| \sum_{c=1}^C \tilde{\mathbf{A}}_i^c \mathbf{X}_i \mathbf{\Theta}_\mu^{(c)} \right\|, \quad \log \sigma_i = \left\| \sum_{c=1}^C \tilde{\mathbf{A}}_i^c \mathbf{X}_i \mathbf{\Theta}_\sigma^{(c)} \right\|; \quad (3.5)$$

onde $\|$ é a operação de concatenação e i refere-se a algum nó e c é o total de camadas.

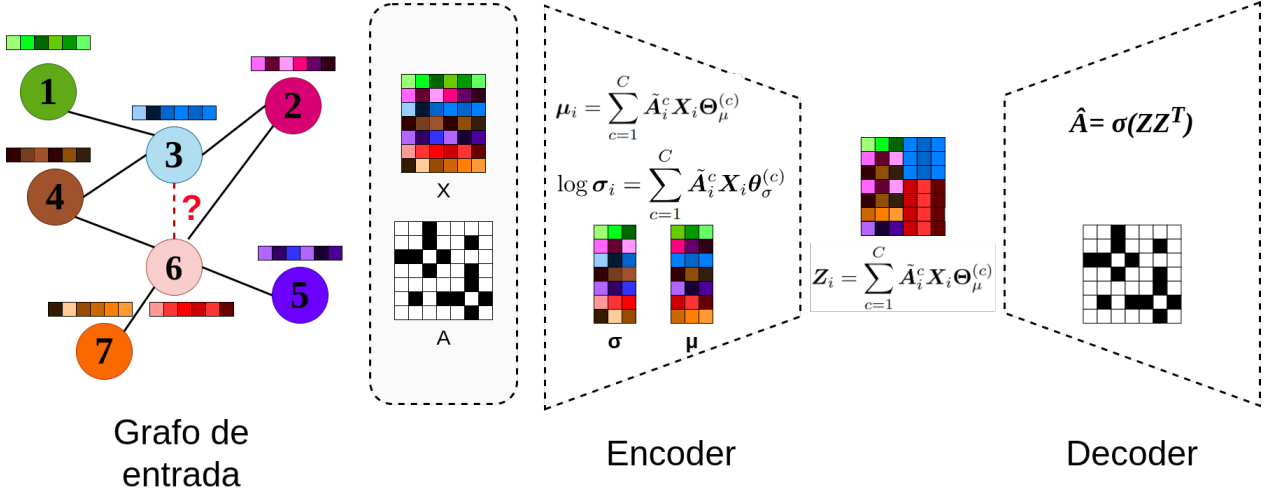
Apesar da natureza linear do codificador, vale ressaltar que a otimização de ML-GVAEs ainda é um problema não convexo. A questão principal aqui é a relação entre a distribuição variacional (Gaussiana) e os σ'_i s. Para controlar o cenário de perdas, também foram propostas versões determinísticas do ML-GVAE, que não depende da injeção de ruído:

$$\text{ML-GAE (sum):} \quad \mathbf{Z}_i = \sum_{c=1}^C \tilde{\mathbf{A}}_i^c \mathbf{X}_i \mathbf{\Theta}_\mu^{(c)}; \quad (3.6)$$

$$\text{ML-GAE (cat):} \quad \mathbf{Z}_i = \left\| \sum_{c=1}^C \tilde{\mathbf{A}}_i^c \mathbf{X}_i \mathbf{\Theta}_\mu^{(c)} \right\|. \quad (3.7)$$

Outra alternativa usual para controlar o cenário de perda é substituir σ'_i s por um hiperparâmetro escalar σ e usar gaussianas isotrópicas para a distribuição variacional. Nesse caso, é necessário apenas uma única rede de codificador. Observe que, ao contrário dos VAEs, os autocodificadores determinísticos não podem gerar amostras. No entanto, isso não é um impedimento para a previsão da aresta, na qual a reconstrução é o objetivo. Na Figura 4 é apresentada uma visão geral do método ML-VGAE. Os logits das novas arestas são obtidas via combinação dos valores de $\mathbf{Z}$.

Figura 4 – Fluxograma MLGVAE. A etapa de codificação utiliza camadas SGCs empilhadas e combinadas utilizando concatenação ou soma.



Fonte: Próprio Autor.

3.4 Experimentos

Na etapa de experimentos, foi seguida a mesma configuração de (SALHA et al., 2020). Portanto, as propostas apresentadas foram avaliadas em tarefas de predição de arestas e foram utilizadas as redes de citações Cora, Citeseer e Pubmed como *benchmarks*. Foram executados experimentos em duas versões de cada conjunto de dados: com e sem recursos de nó. Para o último, foi substituída a matriz de características por uma matriz de identidade.

Os ML-GVAEs foram comparados com (V)GAE com codificadores baseados em GCN (KIPF; WELLING, 2016a) e Linear (V)AE com codificadores de um salto (SALHA et al., 2020). Os ML-GVAEs combinam informações de vizinhança de 2, 3 e 4 saltos ($C = 4$ e não foram utilizadas informações de 1 salto). Além disso, foi utilizada a Área sob a Curva ROC (AUC) e a Precisão Média (*average precision* AP) como métricas. Os experimentos foram repetidos 20 vezes. Para cada método, foram relatados a média e o desvio padrão como resultados.

3.5 Resultados

Na Tabela 1 e na Tabela 2 são mostrados que o método ML-GVAE (soma) e ML-VAE (cat) superam os outros métodos para todos os conjuntos de dados. A lacuna de desempenho entre ML-G(V)AEs e concorrentes é particularmente perceptível no conjunto de dados PubMed, com e sem atributos de nós. Uma possível explicação é que os métodos propostos são os únicos a aproveitar informações de vizinhanças de tamanho três

Tabela 1 – Previsão de aresta usando recursos de nó. ML-G(V)AEs consistentemente superam o (V)GAE baseado em GCN e o Linear G(V)AE em todos os casos.

Modelo	Cora c/ atributos		CiteSeer c/ atributos		PubMed c/ atributos	
	AUC	AP	AUC	AP	AUC	AP
GCN VAE ($L = 2$)	91.6±0.92	92.6±0.91	90.7±1.01	92.0±0.97	94.6±0.51	94.8±0.42
GCN VAE ($L = 3$)	90.5±0.94	91.7±0.88	88.6±0.95	90.2±0.81	92.7±1.02	93.3±0.91
GCN AE ($L = 2$)	91.2±0.78	92.4±0.71	89.7±1.39	90.3±1.62	96.2±0.36	96.2±0.25
GCN AE ($L = 3$)	89.1±1.18	90.9±1.01	87.3±1.74	89.6±1.52	94.8±0.41	95.4±0.26
Linear GVAE	92.5±0.97	93.6±0.68	90.6±0.86	91.8±0.77	94.9±1.10	94.5±0.89
Linear GAE	92.0±0.93	93.3±0.86	91.5±1.17	92.9±0.97	95.8±0.20	95.8±0.17
ML-GVAE (sum)	93.1±0.35	93.5±0.77	92.8±0.61	93.3±0.74	98.1±0.92	98.3±0.16
ML-GVAE (cat)	93.5±0.74	94.2±0.17	93.6±0.83	93.5±0.87	96.7±0.29	96.7±0.11
ML-GAE (sum)	92.4±0.72	93.4±0.67	92.2±0.96	93.2±0.72	97.1±0.67	97.2±0.12
ML-GAE (cat)	93.7±0.37	94.1±0.35	93.4±0.95	94.0±0.53	96.9±0.70	97.0±0.34

Fonte: Autor.

Tabela 2 – Previsão de aresta sem recursos de nó. Nossos codificadores de multi-saltos lineares obtêm maior AUC e AP para todos os conjuntos de dados. A lacuna de desempenho é particularmente maior no Pubmed.

Modelo	Cora		CiteSeer		PubMed	
	AUC	AP	AUC	AP	AUC	AP
GCN VAE ($L = 2$)	84.5±1.05	88.8±1.04	77.5±1.24	82.8±0.91	84.1±0.31	88.0±0.22
GCN VAE ($L = 3$)	84.5±1.42	87.6±1.08	79.3±1.78	83.7±1.13	84.1±0.47	88.2±0.31
GCN AE ($L = 2$)	84.8±1.10	88.4±0.82	78.2±1.69	83.8±1.24	82.5±0.64	87.4±0.38
GCN AE ($L = 3$)	84.6±1.22	87.6±1.11	78.6±1.74	82.8±1.43	83.3±0.98	87.6±0.68
Linear GVAE	84.7±1.24	88.2±1.02	78.9±1.34	83.3±0.99	84.0±0.28	87.9±0.25
Linear GAE	83.2±1.13	87.5±0.95	77.0±1.81	83.0±1.25	81.8±0.32	87.5±0.28
ML-GVAE (sum)	85.5±0.21	89.1±0.12	78.6±0.35	83.5±0.43	90.1±0.42	91.5±0.54
ML-GVAE (cat)	86.3±0.51	89.8±0.61	79.9±0.49	84.7±0.71	86.8±0.22	88.1±0.34
ML-GAE (sum)	85.2±0.16	88.9±0.58	78.3±0.16	83.4±0.05	82.1±0.56	87.9±0.18
ML-GAE (cat)	84.8±0.12	89.1±0.32	77.8±0.18	83.2±0.11	83.7±0.58	88.1±0.63

Fonte: Autor.

e quatro, uma vez que (V)AEs baseados em GCN e (V)GAEs lineares empregam GCNs de duas camadas como codificadores.

Vale ressaltar que o número de camadas utilizados para treinar o VGAE e Linear (GAE) (duas camadas) estão intimamente ligadas ao fato de que é comprovado de que, para esses grafos em específico, duas camadas são a profundidade ótima para resolver o problema. Um número maior de camadas no VGAE faria o método sofrer do fenômeno de *oversmoothing* e isso prejudicaria as predições. ML-GVAE não sofre com o *oversmoothing* devido a combinação com informações de camadas mais rasas (de valores 2 e 3).

3.6 Conclusão

Neste capítulo, foi analisado se a combinação de modelos simples e lineares melhoram a capacidade preditiva de GNNs baseados em autocodificadores. Em seguida foi apresentado o método ML-GVAE, uma versão melhorada do VGAE (KIPF; WELLING, 2016b). Essas análises são referentes a primeira contribuição desta dissertação: aumentar a acurácia de GNNs para predição de arestas.

De fato, ao combinar várias camadas SGCs (WU et al., 2019) para o codificador, o ML-GVAE foi capaz

de alavancar informações de vários saltos. Os resultados mostraram que ML-GVAE supera consistentemente VGAEs e sua variante simplificada (SALHA et al., 2020) (Linear VGAE), em tarefas de predição de arestas. Além disso, ML-GVAE é indiscutivelmente mais rápido que o VGAE, pois podemos pré-calcular os recursos propagados antes do treinamento/teste.

4 Fundamentação teórica: calibração de redes neurais para predição de arestas

As GNNs para predição de arestas modernas alcançaram uma alta performance em solucionar diversos problemas do mundo real como previsão de tráfego (urbano, aéreo ou digital) (ZHOU et al., 2019; LU et al., 2020; JIN et al., 2022), recomendação de conteúdo (LIU, 2022; EL-KISHKY et al., 2022; ZHANG et al., 2022), grafos de conhecimento (ZEB et al., 2022), e prever interação entre proteínas (BALOGH et al., 2022). No entanto, para muitas tarefas críticas, além de uma alta acurácia também é desejado que o modelo possua uma estimativa de incerteza confiável (HSU et al., 2022). Por exemplo, em ambulâncias que possuem GPS com previsão de tráfego, se a GNN do GPS não é capaz de prever com confiança se os trajetos disponíveis estarão ou não congestionados, o GPS precisa alertar o motorista para tomar a decisão de qual percurso seguir.

De forma mais específica, uma GNN precisa garantir uma medida de confiança calibrada associada às suas previsões. Em outras palavras, a probabilidade associada aos rótulos das classes previstas deve refletir a probabilidade da verdade fundamental (*ground truth*) (GUO et al., 2017). Por exemplo, considere uma tarefa em que é desejado classificar imagens como corujas, gatos e cães. Suponhamos que um modelo GNN Ψ retornou probabilidades de 80% de ser gato para 100 amostras quaisquer. Se o modelo está bem calibrado, então espera-se que 80% das amostras que meu modelo associou uma probabilidade de 80% de ser gato, sejam de fato da classe gato. Da mesma forma, se o mesmo modelo atribuiu probabilidade de 15% de ser gato para as amostras de outro conjunto, então espera-se que 15% dessas amostras sejam de fato da classe gato. Se isso ocorrer, o modelo Ψ é dito calibrado.

Este capítulo fornece inicialmente uma visão geral sobre calibração de confiança em redes neurais, e como calcular o grau de confiança de um modelos. Em seguida são discutidos alguns métodos tradicionais para calibração de redes neurais (*isotonic regression*, *temperature scaling*, *Bayesian binning into quantiles*, *histogram binning*) e os motivos que os tornam impróprios para calibração de modelos GNN para predição de arestas.

4.1 Calibração de confiança

Neste trabalho, é considerado o problema de calibração de confiança para tarefas de predição de aresta. Considere um grafo $G = (V, E)$. Cada aresta $i = (u, v) \in E$ possui um rótulo $y = 1$ associado a ela, arestas que não estão em E recebem rótulo $y = 0$. O objetivo da tarefa de predição de aresta é prever se uma determinada aresta i pertence ou não ao grafo G . Modelos GNNs realizam essa tarefa atribuindo probabilidades $\hat{p}_i$ para cada aresta i , onde $\hat{y}_i = 1$ se $\hat{p}_i \geq 0.5$, e $\hat{y}_i = 0$ caso contrário. A confiança de cada aresta é definida como $\hat{c}_i = \max_y \hat{p}_i(y)$. Formalmente, uma GNN está perfeitamente calibrada se: $\forall c \in [0, 1], \mathbb{P}(y_i = \hat{y}_i | \hat{c}_i = c) = c$ (GUO et al., 2017). No restante deste trabalho, será utilizado $\hat{q}_i$ para se referir a uma confiança $\hat{c}_i$ calibrada.

Erro de calibração esperado (*Expected Calibration Error*, *ECE*) calcula a diferença na expectativa entre confiança e precisão. Essa diferença na expectativa pode ser calculada como: $\mathbb{E}_p[|\mathbb{P}(y_i = \hat{y}_i | \hat{c}_i = c_i) - c|]$. Uma forma aproximada de calcular a equação acima é particionando as previsões do modelo em compartimentos (bins) separados com base em suas pontuações de confiança associadas. Assim, para um dado subconjunto

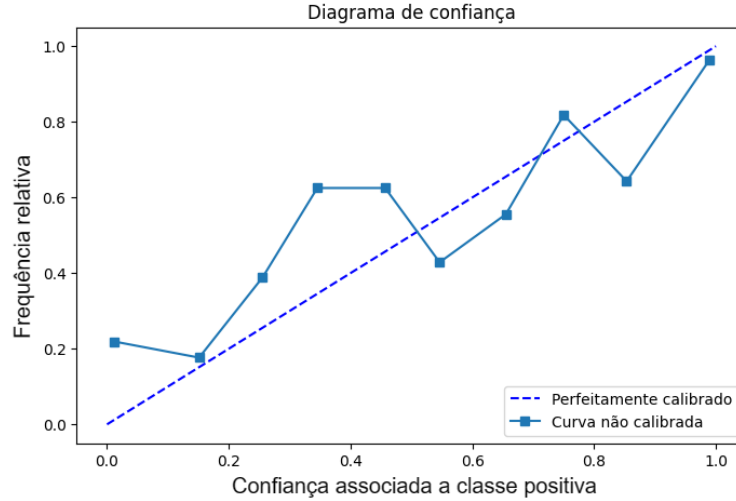
de arestas $\mathcal{M} \subset E$, nós temos $(B_1, \dots, B_N)$, onde $B_n = i \in \mathcal{M} | \frac{n-1}{N} < \hat{c}_i \leq \frac{n}{N}$. Em seguida, a diferença entre a confiança média e a precisão pode ser calculada dentro de cada bin. O ECE calcula a diferença entre a confiança média e a precisão dentro de cada bin e, em seguida, obtém uma média ponderada desses valores com base no tamanho relativo de cada bin.

$$\text{ECE} = \sum_{n=1}^N \frac{|B_n|}{|\mathcal{M}|} |\text{acc}(B_n) - \text{conf}(B_n)| \quad (4.1)$$

onde $\text{acc}(B_n) = \frac{1}{|B_n|} \sum_{i \in B_n} \mathbf{1}(\hat{y}_i = y_i)$ e $\text{conf}(B_n) = \frac{1}{|B_n|} \sum_{i \in B_n} \hat{c}_i$.

Diagramas de Confiabilidade são gráficos que permitem visualizar o quão bem calibrado o modelo está. Esses diagramas descrevem a frequência relativa no eixo y e a confiança média associada a classe positiva no eixo x. As medidas médias de confiança e frequência relativa dentro de cada bin são plotadas no diagrama, formando um gráfico de linha. Aqui, a calibração perfeita resultaria em uma linha diagonal no diagrama de confiabilidade, onde a confiança é igual à frequência relativa dentro de cada bin.

Figura 5 – Diagrama de confiança de uma GNN para predição de aresta não calibrada.



Fonte: Próprio Autor.

4.2 Métodos de calibração

4.2.1 Histogram binning

O textithistogram binning (ZADROZNY; ELKAN, 2001) é um método de calibração não paramétrico simples. Resumindo, todas as predições não calibradas $\hat{p}_i$ são divididas em bins mutuamente exclusivos $B_1, \dots, B_N$. A cada bin é atribuída uma pontuação calibrada θ_n ; ou seja, se $\hat{p}_i$ é atribuído ao bin B_n , então $\hat{q}_i = \theta_n$. No momento do teste, se a predição $\hat{p}_i$ cair no bin B_n , então a predição calibrada $\hat{q}_i$ é θ_n . Mais precisamente, para um N adequadamente escolhido (geralmente pequeno), primeiro definimos limites do bin $0 = a_1 \leq \dots \leq a_{N+1} = 1$, onde o bin B_n é definido pelo intervalo (a_n, a_{n+1}) . Normalmente, os limites do bin são escolhidos para serem intervalos de igual comprimento ou para equalizar o número de amostras em cada bin. As previsões θ_i são escolhidos para minimizar a perda quadrática binária:

$$\min_{\theta_1, \dots, \theta_N} \sum_{n=1}^N \sum_{i=1}^k \mathbf{1}(a_n \leq \hat{p}_i < a_{n+a})(\theta_n - y_i)^2 \quad (4.2)$$

onde $\mathbf{1}$ é a função indicadora. Dados os limites fixos dos bins, a solução para Equação 4.2 resulta em θ_n que corresponde ao número médio de amostras de classes positivas no bin B_n .

4.2.2 Isotonic regression

A *isotonic regression* (ZADROZNY; ELKAN, 2002) é indiscutivelmente o método de calibração não paramétrico mais comum. Ela aprende uma função constante por partes f para transformar saídas não calibradas; $\hat{q}_i = f(\hat{p}_i)$. Especificamente, a regressão isotônica produz f para minimizar a perda quadrática $\sum_{i=1}^k (f(\hat{p}_i) - y_i)^2$. Como f é restrito a ser constante por partes, podemos escrever o problema de otimização como:

$$\begin{aligned} \min_{\substack{a_1, \dots, a_{N+1} \\ \theta_1, \dots, \theta_N}} \sum_{n=1}^N \sum_{i=1}^k \mathbf{1}(a_n \leq \hat{p}_i < a_{n+a})(\theta_n - y_i)^2, \\ \text{sujeito a } 0 = a_1 \leq \dots \leq a_{N+1} = 1, \\ \theta_1 \leq \dots \leq \theta_N \end{aligned} \quad (4.3)$$

onde N é o número de intervalos; $a_1, \dots, a_{N+1}$ são os limites do intervalo; e $\theta_1, \dots, \theta_N$ são os valores da função. Sob esta parametrização, a *isotonic regression* é uma generalização estrita do método *histogram binning* em que os limites do bin e as predições do bin são otimizados em conjunto.

4.2.3 Temperature scaling

O *temperature scaling* (GUO et al., 2017) usa um único parâmetro escalar $T > 0$ para todas as classes. Dado o vetor logit z_i , a nova previsão de confiança é:

$$\hat{q}_i = \max_k \sigma_{SM} \left(\frac{z_i}{T} \right)^k \quad (4.4)$$

onde σ_{SM} é a função *softmax* e k é o número de classes (no caso de predição de arestas $k = 2$).

T é chamado de temperatura e “suaviza” o *softmax* (ou seja, aumenta a entropia de saída) com $T > 1$. Com $T \rightarrow \infty$, a probabilidade $\hat{q}_i$ se aproxima de $1/K$, o que representa a incerteza máxima. Com $T = 1$, recuperamos a probabilidade original $\hat{p}_i$. Com $T \rightarrow 0$, a probabilidade cai para uma massa pontual (ou seja, $\hat{q}_i = 1$). T é otimizado em relação a *negative log likelihood* (NLL) no conjunto de validação. Como o parâmetro T não altera o máximo da função *softmax*, a previsão de classe $\hat{y}_i$ permanece inalterada. Em outras palavras, o *temperature scaling* não afeta a precisão do modelo. O modelo é equivalente a maximizar a entropia da distribuição de probabilidade de saída sujeita a certas restrições nos logits.

4.2.4 Bayesian binning into quantiles

Bayesian binning into quantiles (BBQ) (NAEINI; COOPER; HAUSKRECHT, 2015) é uma extensão do *histogram binning* usando a média do modelo bayesiano. Essencialmente, o BBQ marginaliza todos os

possíveis esquemas de binning para produzir $\hat{q}_i$. Mais formalmente, um esquema de binning s é um par (N, I) onde N é o número de bins e I é uma partição correspondente de $[0, 1]$ em intervalos disjuntos $(0 = a_1 \leq a_2 \leq \dots \leq a_{N+1} = 1)$. Os parâmetros de um esquema de binning são $\theta_1, \dots, \theta_N$. Sob esta estrutura, o *histogram binning* e a *isotonic regression* produzem um único esquema de binning, enquanto o BBQ considera um espaço S de todos os esquemas de binning possíveis para o conjunto de dados de validação D . O BBQ realiza a média bayesiana das probabilidades produzidas por cada esquema:

$$\mathbb{P}(\hat{q}_i | \hat{p}_i, D) = \sum_{s \in S} \mathbb{P}(\hat{q}_i, S = s | \hat{p}_i, D) = \sum_{s \in S} \mathbb{P}(\hat{q}_i | \hat{p}_i, S = s, D) \mathbb{P}(S = s | D) \quad (4.5)$$

onde $\mathbb{P}(\hat{q}_i | \hat{p}_i, S = s, D)$ é a probabilidade calibrada usando o esquema de binning s . Usando uma priore uniforme, o peso $\mathbb{P}(S = s | D)$ pode ser derivado usando a regra de Bayes:

$$\mathbb{P}(S = s | D) = \mathbb{P}(D | S = s) / \sum_{s' \in S} \mathbb{P}(D | S = s') \quad (4.6)$$

Os parâmetros $\theta_1, \dots, \theta_N$ podem ser vistos como parâmetros de N distribuições binomiais independentes. Portanto, colocando um Beta a priori em $\theta_1, \dots, \theta_N$, podemos obter uma expressão de forma fechada para a verossimilhança marginal $\mathbb{P}(D | S = s)$. Isso nos permite calcular $\mathbb{P}(\hat{q}_i | \hat{p}_i, D)$ para qualquer entrada de teste.

4.2.5 Pontos fracos dos calibradores tradicionais

Quando aplicados para GNNs, os métodos tradicionais de calibração de confiança ignoram o efeito da topologia do grafo, ou seja, esses métodos não podem efetivamente aprender informações úteis para GNNs (e.g., vizinhos comuns, index de Katz, etc), o que irá inevitavelmente gerar erros durante a calibração. Além disso, muitos desses métodos tradicionais exploram funções de calibração somente no espaço linear, enquanto é bem conhecido que espaços não-lineares contêm transformações de funções muito mais complexas que são capazes de realizar uma melhor calibração de GNNs mais complicadas. Finalmente, esses métodos geralmente afetam negativamente a pontuação das métricas como AUC, Hits@k da GNN original, enquanto uma boa pontuação é ainda um requisito básico para muitas aplicações (HSU et al., 2022). Métodos como *histogram binning*, e *isotonic regression*, colapsam as probabilidades para um espaço restrito de valores, assim, essas abordagens podem afetar negativamente a métrica Hits@k.

5 Contribuição II: calibrador de confiança para GNNs de predição de aresta

Neste capítulo é apresentado um novo calibrador de confiança de GNNs para predição de arestas que visa tanto alcançar um baixo ECE, quanto suprir as desvantagens dos calibradores tradicionais apresentados no capítulo anterior. Esse novo calibrador é chamado de *on-and-off link scaling* (OOLS). O OOLS não só leva em consideração as informações topológicas do grafo, mas também busca transformações dos logits em um espaço não-linear, o que o torna mais adequado do que os métodos tradicionais para calibrar GNNs complexas. O OOLS consiste de duas etapas principais: geração das representações (*embeddings*) das arestas $\mathbf{H}$ e otimização de um vetor de temperatura $\mathbf{t}$.

Antes de entender como o OOLS funciona em seus detalhes, primeiro é necessário explicar o que leva as GNNs serem descalibradas. A Seção 5.1 lista alguns dos principais fatores que influenciam a calibração das GNNs. Entender esses fatores permite uma compreensão mais detalhada das contribuições que um modelo calibrador pode trazer.

5.1 GNNs para predição de aresta são descalibradas?

No contexto de redes neurais para imagens e sequências, Guo et al. (2017) destacam dois fatores principais que afetam a calibração de redes neurais. O primeiro fator diz respeito a profundidade e largura do modelo. Quanto mais profundo e largo o modelo, maior é a descalibração. O segundo fator que afeta a calibração dos modelos é a utilização de *batch normalization*. Apesar de não estar 100% claro como ele afeta a calibração dos modelos, Guo et al. (2017) afirmam que variações na taxa de aprendizado da função de *batch normalization* causam variações na calibração final do modelo.

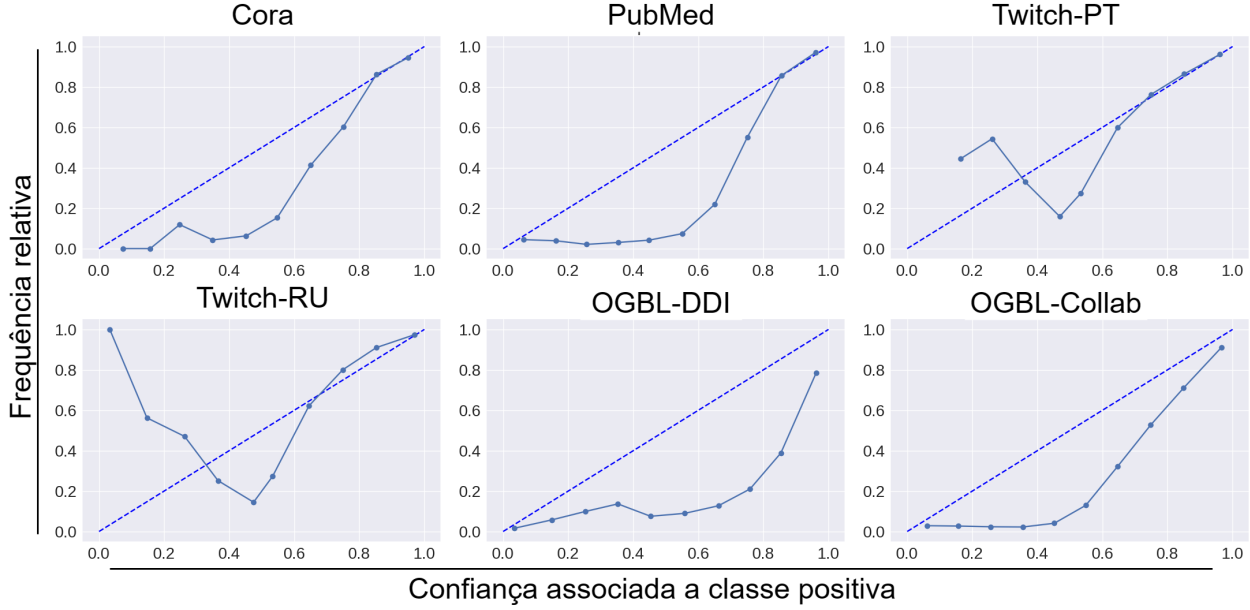
Com relação a modelos para grafos para classificação de nó, Hsu et al. (2022) apresentam a distância entre os elementos do conjunto de teste e os elementos do conjunto de treino como um dos principais fatores responsáveis pela descalibração de GNNs. No entanto, aspectos relacionados à calibração de GNNs para predição de arestas ainda não foram explorados na literatura.

A Figura 6 apresenta diagramas de confiabilidade para predição de aresta. Nela, é possível observar que as predições são significativamente descalibradas. Na maioria dos conjuntos de dados, é observado que para amostras preditas como classe 1, o modelo apresenta comportamento super-confiante, ao passo que para classe 0, o modelo é sub-confiante. Isso sugere a adoção de um calibrador baseado em temperatura que dependa da predição fornecida pela GNN.

5.2 *On-and-off link scaling*

Seja $\mathbf{H} = \text{GNN}(\mathbf{X}, \mathbf{A})$ tal que sua u -ésima linha $\mathbf{h}_u$ é o embedding para o nó u . Seguindo o padrão na literatura de GNNs, definimos o embedding de uma aresta (u, v) agregando $\mathbf{h}_u$ e $\mathbf{h}_v$ usando uma função ϕ , i.e., $\mathbf{h}_{uv} = \phi(\mathbf{h}_u, \mathbf{h}_v)$. No caso em que a aresta (u, v) é não-direcionada, ϕ é invariante à ordem em que os embeddings de nó são fornecidos — i.e., $\phi(\mathbf{h}_u, \mathbf{h}_v) = \phi(\mathbf{h}_v, \mathbf{h}_u)$. Para prever a ocorrência de uma aresta (u, v) , é calculada a log-probabilidade s_{uv} usando uma MLP, i.e., $s_{uv} = \text{MLP}(\mathbf{h}_{uv})$.

Figura 6 – Diagramas de confiabilidade de GNNs para predição de aresta. Pode ser notada uma tendência geral de previsões super confiantes (gráficos abaixo da diagonal) para valores de $p_{uv} > 0.5$ (classe 1). No entanto, a tendência muda quando $p_{uv} < 0.5$, apresentando comportamento sub-confiante.



Fonte: Próprio Autor.

Suponha que deseja-se prever a ocorrência de (u, v) . Suponha ainda que $A_{uv} = 0$ e deixe que $\mathbf{A}^+$ seja definida como:

$$A_{ij}^+ = \begin{cases} A_{ij}, & \text{if } (i, j) \neq (u, v) \\ 1, & \text{caso contrário.} \end{cases}$$

Deixe então que $\mathbf{H}^+ = \text{GNN}(\mathbf{X}, \mathbf{A}^+)$, de maneira similar à que descrevemos anteriormente. Deixe ainda que $\mathbf{h}_{uv}^+ = \phi(\mathbf{h}_u^+, \mathbf{h}_v^+)$. Intuitivamente, quando $\mathbf{h}_{uv}^+$ e $\mathbf{h}_{uv}$ diferem (em algum sentido) por uma grande margem, adicionar essa aresta conflita com a informação prévia que temos sobre o grafo. Quando nossa previsão $\hat{y} = \mathbb{1}[s_{uv} > 0]$ é positiva, observar uma margem larga entre $\mathbf{h}_{uv}^+$ e $\mathbf{h}_{uv}$ deveria diminuir nossa confiança. Quando $\hat{y} = 0$, no entanto, a mesma observação reforça a noção de que a aresta (u, v) é *alien* ao grafo que estamos analisando. Note que esse raciocínio está alinhado com a intuição que é tida sobre a Figura 6. Com isso em mente, é proposto o OOLS, um método de calibração baseado em *temperature scaling*, no qual calculamos a temperatura t_{uv} como uma função da margem entre $\mathbf{h}_{uv}$ e $\mathbf{h}_{uv}^+$.

Mais especificamente, são propostas duas formas de medir a discrepância (margem) entre $\mathbf{h}_{uv}$ e $\mathbf{h}_{uv}^+$: usando a distância Euclidiana $\|\mathbf{h}_{uv} - \mathbf{h}_{uv}^+\|_2$ e a diferença $\mathbf{h}_{uv} - \mathbf{h}_{uv}^+$. No caso da distância, a temperatura é calculada como

$$t_{uv} = \begin{cases} \text{MLP}_c(\|\mathbf{h}_{uv} - \mathbf{h}_{uv}^+\|_2) & \text{se } s_{uv} > 0 \\ \text{MLP}_c(-\|\mathbf{h}_{uv} - \mathbf{h}_{uv}^+\|_2) & \text{caso contrário} \end{cases},$$

onde a rede neural MLP_c , que é a MLP para calibração, deve retornar um vetor $\mathbf{t} \in \mathbb{R}^+$. Para tal, sua última camada pode ser equipada com uma ReLU ou uma Softplus.

No caso da diferença entre *embeddings*, a temperatura t_{uv} é calculada de forma similar como

$$t_{uv} = \begin{cases} \text{MLP}_c(\mathbf{h}_{uv} - \mathbf{h}_{uv}^+) & \text{se } s_{uv} > 0 \\ \text{MLP}_c(\mathbf{h}_{uv}^+ - \mathbf{h}_{uv}) & \text{caso contrário} \end{cases}.$$

Para obter uma probabilidade calibrada $\hat{q}_{uv}$ para uma aresta (u, v) , o *logit* s_{uv} é dividido pela temperatura t_{uv} de modo que $\hat{q}_{uv} = \sigma(s_{uv}/t_{uv})$. Além disso, nos experimentos, a margem (distância ou diferença) é tratada como hiper-parâmetro.

5.3 Treinamento do calibrador

Para aprender os parâmetros da MLP_c é minimizada uma soma ponderada entre 1) a *negative log likelihood* (NLL) do modelo calibrado no conjunto de treino; 2) uma penalidade que incentiva baixa confiança para predições incorretas e alta confiança para predições corretas; e 3) o erro esperado de calibração ECE. Mais especificamente, a NLL é dada por

$$\mathcal{L}_{\text{NLL}} = - \sum_{i=1}^{|D_{\text{train}}|} \sum_{k=1}^2 y_{i,k} \log(\hat{q}_{i,k}) \quad (5.1)$$

onde $|D_{\text{train}}|$ é o tamanho do conjunto de treino. Já o termo de regularização (WANG et al., 2021) é dado por:

$$\mathcal{L}_{\text{CAL}} = \frac{1}{n} \left(\sum_{i=1}^{|\text{cor}|} (1 - \hat{q}_{i,m}^{(\text{cor})} + \hat{q}_{i,s}^{(\text{cor})}) + \sum_{i=1}^{|\text{inc}|} (\hat{q}_{i,m}^{(\text{inc})} - \hat{q}_{i,s}^{(\text{inc})}) \right), \quad (5.2)$$

onde $|\text{cor}|$ e $|\text{inc}|$ são o número de arestas correta e incorretamente preditas e $\hat{q}_{i,m}$ e $\hat{q}_{i,s}$ são a maior e a segunda maior probabilidade calibrada, respectivamente. Por fim, a função objetivo resultante é dada por $\mathcal{L} = \mathcal{L}_{\text{NLL}} + \mathcal{L}_{\text{CAL}} + \lambda \text{ECE}$, onde λ é um hiper-parâmetro de regularização. Além disso, vale ressaltar que é utilizado o conjunto de validação para seleção de modelo.

5.4 Experimentos

Nesta seção são descritos os conjuntos de dados, as configurações dos experimentos e os resultados alcançados. Foram consideradas duas GNNs de predição de arestas para serem calibradas. A fim de se obter um resultado mais preciso possível, cada GNN foi treinada 5 vezes seguindo uma validação cruzada. Além disso, cada um dos cinco modelos GNNs treinados foi calibrado 5 vezes utilizando os calibradores analisados. Isso resulta em 25 execuções para cada calibrador para uma única GNN. Por fim, os resultados foram reportados via média e desvio padrão em porcentagem (%).

5.4.1 Conjuntos de dados

Os conjuntos de dados Cora, Citeseer, PubMed (SEN et al., 2008) são grafos de citações onde os nós representam os documentos de diferentes domínios e as arestas são links de citação entre esses documentos. Os conjuntos de dados Twitch (RU) e Twitch (PT) (ROZEMBERCZKI; ALLEN; SARKAR, 2021) são grafos de redes usuário-usuário onde os nós correspondem aos usuários do Twitch e as arestas correspondem as amizades mútuas. Os atributos dos nós são jogos preferidos, localização e hábitos de streaming. O conjunto de dados Chameleon (ROZEMBERCZKI; ALLEN; SARKAR, 2021) é um grafo de redes de páginas da Wikipedia sobre tópicos envolvendo camaleões. Os nós representam artigos da Wikipédia em inglês, as arestas refletem

os links mútuos entre eles. Os recursos do nó indicam a presença de substantivos específicos nos artigos e o tráfego mensal médio entre outubro de 2017 a novembro de 2018.

Tabela 3 – Estatísticas dos datasets utilizados nos nossos experimentos.

Dataset	nós	arestas	atributos
Cora	2708	10556	1433
PubMed	19717	88648	500
Citeseer	3327	9104	3703
Twitch-RU	4385	67148	128
Twitch-PT	1912	57905	128
Chameleon	2277	39730	2325

5.4.2 Baselines

O OOLS é comparado com quatro calibradores: *isotonic regression* (ZADROZNY; ELKAN, 2002), *histogram binning* (ZADROZNY; ELKAN, 2001), *temperature scaling* (GUO et al., 2017), e BBQ (NAEINI; COOPER; HAUSKRECHT, 2015). Para garantir uma comparação válida, as configurações dos experimentos deste trabalho seguem de perto as diretrizes e a configuração de avaliação dos trabalhos originais. Assim ambos os métodos foram treinados com o conjunto de treino, e testados com o conjunto de teste. Além disso, todos os calibradores foram aplicado em duas arquiteturas de GNNs para predição de arestas: *graph sample and aggregate* (GraphSAGE) (HAMILTON; YING; LESKOVEC, 2017), e *variational graph autoencoder* (VGAE) (KIPF; WELLING, 2016b). Isso permite avaliar a robustez e o desempenho dos calibradores em GNNs de diferentes complexidades.

5.5 Configurações dos experimentos

Para o treinamento das GNNs os conjuntos de dados foram divididos em 80%/10%/10%/ (treino, teste e validação). Para o treinamento de todas as GNNs foi utilizado o otimizador Adam (KINGMA; BA, 2014) com uma taxa de aprendizado de 1×10^{-2} com *weight decay* de 1×10^{-4} . Todos os calibradores avaliados foram treinados utilizando o conjunto de treino e os resultados reportados foram adquiridos utilizando o conjunto de teste. O OOLS é modelado com base em uma MLP de duas camadas com arquitetura $(d, 50, 128)$, onde d representa o número de colunas da matriz $\mathbf{H}^+$. OOLS também foi otimizado utilizando Adam com taxa de aprendizado 1×10^{-4} e *weight decay* de 5×10^{-8} . Para o cálculo do ECE, foi escolhido um número de *bins* $N = 15$. Todas as implementações foram desenvolvidas utilizando a linguagem *Python* 3.10 e as bibliotecas *PyTorch* (PASZKE et al., 2019) e *Torch Geometric* (FEY; LENSSEN, 2019). Todos os experimentos foram executados usando um computador 64-bits Windows 11, CPU intel Core™ i7-12700H 12ª geração, 16GB RAM DDR5, e placa de vídeo NVIDIA GeForce RTX™ 3060 com 6GB GDDR6.

5.6 Resultados e discussões

Nesta seção, é mostrado o desempenho dos modelos que foram calibrados (foi notado que modelos com AUC mais alto tendem a ter naturalmente um ECE mais baixo). Em seguida são fornecidos os resultados do modelo calibrador OOLS para todas as GNNs em termos de ECE. Esses resultados são então comparados com outras quatro *baselines* tradicionais de calibração. Nesta seção os modelos também são comparados baseados no grau de impacto que eles causam no AUC e Hits@20 do modelo GNN alvo. Por fim são apresentadas as conclusões.

Resultados de predição das GNNs: O desempenho das GNNs que foram calibradas é mostrado na Tabela 4. Na tabela são reportados a média e desvio padrão em porcentagem dos resultados em termos de AUC e Hits@20. Além disso, o ECE antes da calibração também é reportado. Pode ser notado que o modelo GraphSAGE apresenta, no geral, maiores valores de ECE, o que o torna mais descalibrado. Um dos motivos para esse fenômeno é a largura relativamente grande desse modelo (se comparado aos demais), o que confirma os impactos dos fatores apresentados previamente nesta seção.

Tabela 4 – Performance (média e desvio padrão para AUC e Hits@20) das GNNs que foram calibradas. Em adicional também é apresentado o ECE antes da calibração. No geral, quanto maior o AUC do modelo, menor é o ECE. Isso é explicado pelo fato de modelos com AUC maiores terem uma menor entropia (NLL).

	Model	AUC	Hits@20	ECE
cora	VGAE	89.7 ± 0.50	54.7 ± 4.40	17.0 ± 0.76
	SAGE	66.9 ± 1.68	25.0 ± 1.81	21.2 ± 1.74
cite	VGAE	86.3 ± 2.37	50.9 ± 5.21	20.0 ± 0.67
	SAGE	78.1 ± 1.47	38.0 ± 2.92	22.5 ± 1.89
pub	VGAE	95.3 ± 0.37	37.8 ± 2.92	12.4 ± 0.64
	SAGE	68.5 ± 2.28	16.7 ± 1.99	18.7 ± 1.42
PT	VGAE	83.7 ± 0.26	19.4 ± 0.24	16.3 ± 0.20
	SAGE	73.0 ± 0.68	11.3 ± 1.41	25.8 ± 2.84
RU	VGAE	86.3 ± 0.30	24.8 ± 0.20	16.9 ± 0.75
	SAGE	70.2 ± 1.37	9.92 ± 2.47	18.0 ± 2.13
chame	VGAE	97.2 ± 1.17	62.0 ± 2.81	7.5 ± 1.63
	SAGE	53.6 ± 3.31	23.8 ± 0.88	13.9 ± 1.61

Resultados Erro de calibração esperado (ECE): O desempenho do OOLS em comparação aos métodos concorrentes de calibração é exibido na Tabela 5. Aqui **Descal**, **Iso**, **Temp**, **BBQ** e **Hist** referem-se ao ECE dos modelos descalibrados, calibrados com *isotonic regression*, *temperature scaling*, BBQ, e *histogram binning*, respectivamente. **Embeddings+MLP** significa que foram utilizados os *embeddings* originais da GNN como entrada da MLP do OOLS. O OOLS (signed) superou os *baselines* em quase todos os cenários avaliados, onde em especial, a calibração do método GraphSAGE para os datasets cora e citeseer foi a mais expressiva. Tanto para o VGAE, quanto para o GraphSAGE, a calibração do grafo Twitch RU não obteve resultados tão expressivos se comparada aos métodos concorrentes.

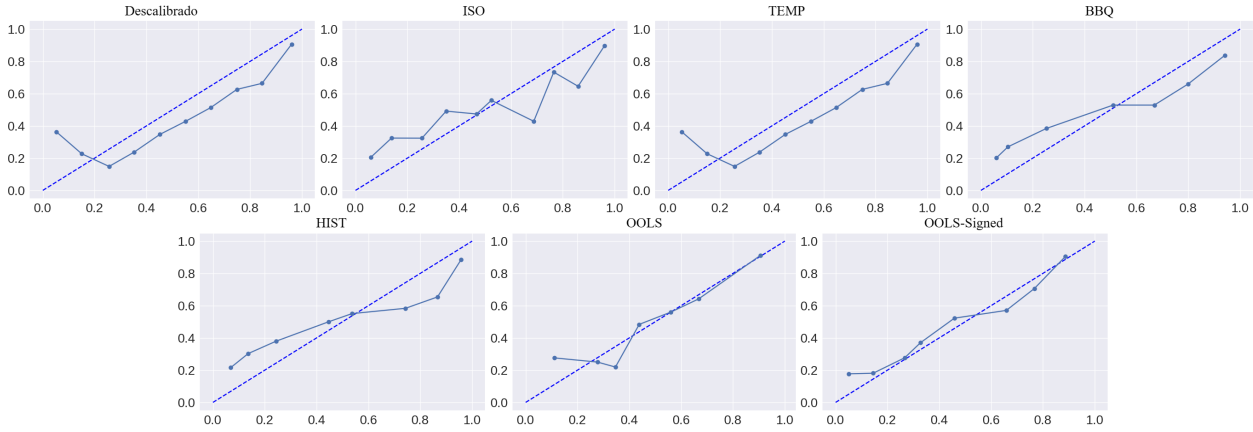
Tabela 5 – Erro de calibração esperado (média e desvio padrão). Em negrito estão destacados os melhores resultados do ECE encontrado para cada GNN.

	Model	Descal	Iso	Temp	BBQ	Hist	Embeddings+MLP	OOLS-Signed	OOLS-Unsigned
cora	VGAE	17.0 ± 0.76	6.88 ± 1.72	12.6 ± 1.79	7.09 ± 1.09	6.14 ± 0.76	4.91 ± 1.60	3.33 ± 0.67	3.32 ± 0.58
	SAGE	21.2 ± 1.74	15.3 ± 1.84	11.2 ± 1.47	14.6 ± 1.10	15.6 ± 1.19	3.58 ± 1.29	1.84 ± 0.48	2.47 ± 0.53
cite	VGAE	20.0 ± 0.67	5.52 ± 1.81	18.5 ± 1.86	5.14 ± 1.29	5.02 ± 1.36	4.92 ± 1.06	4.19 ± 0.65	4.26 ± 0.43
	SAGE	22.5 ± 1.89	18.8 ± 1.14	12.8 ± 1.56	18.6 ± 1.90	17.8 ± 1.82	2.62 ± 0.62	3.79 ± 1.98	1.67 ± 1.01
pub	VGAE	12.4 ± 0.64	3.04 ± 1.52	20.8 ± 1.86	2.22 ± 1.59	2.20 ± 1.97	3.01 ± 1.16	2.81 ± 0.76	1.85 ± 0.37
	SAGE	18.7 ± 1.42	4.93 ± 1.41	13.9 ± 1.72	4.95 ± 1.57	4.83 ± 1.32	4.16 ± 1.47	3.01 ± 1.3	2.99 ± 1.12
PT	VGAE	16.3 ± 0.20	1.84 ± 1.10	16.1 ± 1.45	1.72 ± 1.13	1.66 ± 1.24	4.81 ± 1.56	1.50 ± 0.44	1.19 ± 0.21
	SAGE	25.8 ± 2.84	2.17 ± 1.37	23.9 ± 1.53	2.08 ± 1.14	1.90 ± 1.42	4.06 ± 1.75	1.53 ± 0.89	1.72 ± 0.81
RU	VGAE	16.9 ± 0.75	2.15 ± 1.30	12.0 ± 1.53	2.02 ± 1.37	1.55 ± 1.09	2.22 ± 1.1	4.12 ± 1.32	4.19 ± 0.97
	SAGE	18.0 ± 2.13	2.22 ± 1.32	16.7 ± 1.15	2.35 ± 1.26	2.34 ± 1.19	4.56 ± 1.99	3.32 ± 1.06	2.92 ± 1.23
chame	VGAE	7.5 ± 1.63	2.83 ± 1.26	6.3 ± 1.46	2.84 ± 1.22	2.77 ± 1.29	3.57 ± 1.19	1.07 ± 0.34	1.79 ± 0.56
	SAGE	13.9 ± 1.61	3.46 ± 1.44	12.0 ± 1.64	3.59 ± 1.76	3.23 ± 1.56	3.99 ± 1.45	2.46 ± 0.86	2.25 ± 1.59

Em adição, na Figura 7 são exibidos alguns diagramas de confiança para o modelo GraphSAGE treinado com o grafo Cora. Note como a linha azul de calibração do OOLS fica mais próxima a linha diagonal.

Esse comportamento indica uma melhor calibração.

Figura 7 – Comparação entre os diagramas de confiança do GraphSAGE treinado com o grafo Cora antes e depois da calibração com os métodos analisados. O OOLS possui o diagrama mais próximo da reta diagonal, o que indica uma melhor calibração.



Fonte: Próprio Autor.

Resultados AUC e Hits@20 pós calibração: Na Tabela 6 são exibidos os resultados da métrica AUC pós calibração. O objetivo é avaliar como os calibradores afetam essa métrica. Como pode ser observado, o *temperature scaling* não gera qualquer mudança na AUC, isso se deve ao fato desse calibrador utilizar um valor único de temperatura para todas as confianças. No geral, o OOLS produz algumas melhoras na AUC se comparado tanto a original, quanto a resultante da aplicação dos outros calibradores. Essa é uma consequência direta da otimização da temperatura do OOLS via NLL, uma vez que essa otimização funciona na prática como um processo de suavização do modelo.

Tabela 6 – Comparação do impacto que os métodos de calibração causam sobre a AUC das GNNs. No geral os métodos tradicionais tendem a causar uma ligeira redução na AUC. Enquanto o OOLS a deixa quase inalterada.

	Model	Iso	Temp	BBQ	Hist	Embeddings+MLP	OOLS-Signed	OOLS-unsigned
cora	VGAE	89.5 ± 0.54	89.7 ± 0.50	89.6 ± 0.68	88.9 ± 0.58	89.1 ± 2.52	89.6 ± 1.65	89.8 ± 1.76
	SAGE	67.0 ± 1.76	66.9 ± 1.68	66.3 ± 2.89	64.3 ± 1.46	66.9 ± 1.78	67.3 ± 1.09	67.3 ± 1.48
cite	VGAE	86.1 ± 2.35	86.3 ± 2.37	86.2 ± 2.40	85.6 ± 2.87	86.0 ± 2.85	86.6 ± 1.34	86.7 ± 1.56
	SAGE	77.9 ± 1.19	78.1 ± 1.47	76.1 ± 2.01	69.4 ± 1.58	77.6 ± 2.10	78.9 ± 1.97	78.3 ± 1.60
pub	VGAE	95.3 ± 0.34	95.3 ± 0.37	95.2 ± 0.37	94.7 ± 0.38	95.2 ± 0.98	95.6 ± 1.25	95.7 ± 1.67
	SAGE	69.8 ± 2.12	68.5 ± 2.28	69.9 ± 1.48	68.0 ± 1.96	68.6 ± 1.86	68.0 ± 1.93	68.9 ± 1.25
PT	VGAE	83.3 ± 0.17	83.7 ± 0.26	83.6 ± 0.12	83.6 ± 0.11	82.5 ± 1.56	83.9 ± 1.01	83.8 ± 1.41
	SAGE	72.0 ± 0.69	73.0 ± 0.68	72.5 ± 0.84	72.2 ± 0.95	72.7 ± 1.58	73.2 ± 1.14	73.5 ± 1.35
RU	VGAE	86.6 ± 1.94	86.3 ± 0.30	86.6 ± 0.42	85.6 ± 0.29	86.2 ± 2.12	86.9 ± 0.31	86.7 ± 1.32
	SAGE	69.1 ± 1.38	70.2 ± 1.37	69.3 ± 1.10	69.9 ± 1.08	68.3 ± 1.25	70.6 ± 1.06	70.2 ± 1.32
chame	VGAE	97.1 ± 0.17	97.2 ± 0.17	97.0 ± 0.22	96.7 ± 0.15	97.5 ± 1.00	97.1 ± 1.50	97.8 ± 1.02
	SAGE	52.1 ± 1.55	53.6 ± 3.31	52.5 ± 0.64	52.2 ± 0.96	53.6 ± 2.99	53.6 ± 1.19	53.5 ± 1.14

Na Tabela 7 são apresentados os valores da métrica Hits@20 para os modelos pós calibrados. Nessa tabela, os números destacados em negrito referem-se aos piores valores resultantes. Pode ser notado que o método *histogram binning* é o que mais afeta negativamente os valores de Hits@20 das GNNs. Isso se deve ao fato de que esse calibrador colapsa todos os valores das probabilidades contidos em um bin para a média desse bin. Como consequência, arestas negativas podem ter os seus valores de confiança igualados aos valores de confiança das arestas positivas. *Isotonic regression* e BBQ podem sofrer do mesmo problema. Por outro

lado, o OOLS não gera perdas na métrica Hits@20, na verdade ele a melhora em alguns casos. Isso se deve ao fato do OOLS ser otimizado via NLL, e foi mostrado que quanto menor os valores de NLL, melhor vai ser a calibração e a precisão do modelo.

Tabela 7 – Hits@20 pós-calibração. Os menores valores estão destacados em negrito. **BBQ**, quanto **Hits** são os métodos mais nocivos para essas métricas. Equanto o OOLS a melhora em vários cenários.

	Model	Iso	Temp	BBQ	Hist	Embeddings+MLP	OOLS-Signed	OOLS-unsigned
cora	VGAE	50.3 $\pm$ 4.58	54.7 $\pm$ 4.40	48.5 $\pm$ 5.97	36.8 $\pm$ 1.72	52.9 $\pm$ 1.56	54.9 $\pm$ 1.05	54.7 $\pm$ 1.21
	SAGE	22.4 $\pm$ 3.76	25.0 $\pm$ 1.81	21.1 $\pm$ 2.52	15.4 $\pm$ 1.12	24.9 $\pm$ 1.98	26.8 $\pm$ 1.19	25.6 $\pm$ 1.30
cite	VGAE	43.6 $\pm$ 6.80	50.9 $\pm$ 5.21	49.1 $\pm$ 5.73	33.9 $\pm$ 2.82	50.9 $\pm$ 1.79	52.1 $\pm$ 1.12	51.6 $\pm$ 1.19
	SAGE	36.7 $\pm$ 3.50	38.0 $\pm$ 2.92	33.1 $\pm$ 4.36	27.9 $\pm$ 10.1	38.1 $\pm$ 2.23	40.1 $\pm$ 1.59	38.3 $\pm$ 1.51
pub	VGAE	35.0 $\pm$ 2.96	37.8 $\pm$ 2.92	6.93 $\pm$ 13.8	0.0 $\pm$ 0.0	37.9 $\pm$ 2.09	37.8 $\pm$ 1.02	37.9 $\pm$ 1.06
	SAGE	14.5 $\pm$ 3.02	16.7 $\pm$ 1.99	15.1 $\pm$ 2.31	14.4 $\pm$ 3.54	16.5 $\pm$ 1.68	17.6 $\pm$ 1.03	17.0 $\pm$ 1.25
PT	VGAE	13.5 $\pm$ 0.21	19.4 $\pm$ 0.24	0.04 $\pm$ 0.0	0.04 $\pm$ 0.0	19.5 $\pm$ 1.63	19.6 $\pm$ 1.12	19.9 $\pm$ 1.41
	SAGE	9.49 $\pm$ 2.18	11.3 $\pm$ 1.41	0.0 $\pm$ 0.0	0.0 $\pm$ 0.0	11.8 $\pm$.98	17.2 $\pm$ 1.14	17.5 $\pm$ 1.35
RU	VGAE	23.0 $\pm$ 1.94	24.8 $\pm$ 0.02	0.01 $\pm$ 0.02	0.02 $\pm$ 0.03	24.7 $\pm$ 1.21	24.4 $\pm$ 0.31	24.7 $\pm$ 1.32
	SAGE	3.98 $\pm$ 2.75	9.92 $\pm$ 2.47	0.0 $\pm$ 0.0	0.0 $\pm$ 0.0	9.93 $\pm$ 2.52	11.5 $\pm$ 1.02	10.9 $\pm$ 1.32
chame	VGAE	52.7 $\pm$ 8.91	62.0 $\pm$ 2.81	0.0 $\pm$ 0.0	0.0 $\pm$ 0.0	63.1 $\pm$ 1.60	64.0 $\pm$ 1.01	62.5 $\pm$ 1.00
	SAGE	19.1 $\pm$ 2.40	23.8 $\pm$ 0.88	22.0 $\pm$ 2.63	22.2 $\pm$ 3.05	23.6 $\pm$ 2.08	23.6 $\pm$ 1.20	23.5 $\pm$ 1.59

5.7 Conclusão

Neste capítulo foram apresentados fatores que influenciam a calibração de GNNs para predição de aresta. Além disso, foram explorados novos meios de geração de *embeddings* mais informativos, o que levou a construção do método calibrador OOLS. Além disso também foram destacados a motivação para criação do OOLS, a descrição dos conjuntos de dados utilizados, a configuração utilizada nos experimentos, e os resultados alcançados.

O OOLS foi avaliado utilizando a métrica *expected calibration error* (ECE). Além disso, também foram apresentados resultados para as métricas AUC e Hits@20. Foi realizada uma comparação com quatro métodos de calibração presentes na literatura. Todos os resultados apresentados confirmam a eficácia do OOLS em calibrar GNNs, bem como sua capacidade de não afetar negativamente a precisão desses modelos.

6 Conclusão

Neste trabalho, foram fornecidos avanços para predição de arestas com GNNs ao serem propostos dois modelos: 1) ML-VGAE, um método variante do VGAE que utiliza menos parâmetros e é tão expressivo quanto. Esse método utiliza combinações de múltiplos autocodificadores lineares para gerar as representações das arestas. Essa característica permite o ML-VGAE explorar vizinhanças maiores dos nós enquanto mantém uma complexidade baixa; 2) OOLS, um calibrador de confiança específico para GNNs em tarefas de predição de arestas. Esse método utiliza variantes das representações das arestas do grafo para alimentar uma rede neural que gera um vetor de temperatura responsável por transformar as probabilidades originais fornecidas pela GNN em probabilidades calibradas.

As propostas foram avaliadas de formas diferentes. O ML-GVAE foi avaliado através da tarefa de predição de aresta em alguns grafos populares contra os métodos VGAE e VAE. Os resultados, em termos de AUC e precisão, foram comparados variando atributos dos nós. Os experimentos mostraram que o ML-VGAE superou totalmente os métodos concorrentes. Além disso, argumentos intuitivos foram fornecidos para sustentar a utilização de uma abordagem de combinação de múltiplos autocodificadores lineares.

Já o calibrador OOLS foi avaliado na calibração de duas GNNs para predição de arestas treinadas com 6 grafos (*datasets*) diferentes. Os resultados apresentados demonstram a viabilidade do calibrador, uma vez que ele mostra um desempenho, no mínimo, equivalente aos calibradores tradicionais concorrentes. Além disso, os experimentos comprovaram que o nosso calibrador não afeta negativamente as métricas de avaliação das GNNs, o que é importante, uma vez que uma alta acurácia ainda é requisito essencial para muitas tarefas envolvendo predição de arestas.

Referências

- AMODEI, D. et al. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016. Citado na página 26.
- BALOGH, O. M. et al. Efficient link prediction in the protein–protein interaction network using topological information in a generative adversarial network machine learning model. *BMC bioinformatics*, BioMed Central, v. 23, n. 1, p. 1–19, 2022. Citado 3 vezes nas páginas 25, 31 e 41.
- BISHOP, C. M.; NASRABADI, N. M. *Pattern recognition and machine learning*. [S.l.]: Springer, 2006. v. 4. Citado na página 25.
- BRONSTEIN, M. M. et al. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, IEEE, v. 34, n. 4, p. 18–42, 2017. Citado na página 25.
- BRONSTEIN, M. M. et al. Geometric deep learning: Going beyond Euclidean data. *IEEE Signal Processing Mag.*, v. 34, n. 4, p. 18–42, 2017. Citado na página 35.
- BRUNA, J. et al. Spectral networks and locally connected networks on graphs. In: *ICLR*. [S.l.: s.n.], 2014. Citado na página 36.
- CHEN, H.; LI, X.; HUANG, Z. Link prediction approach to collaborative filtering. In: *Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL '05)*. [S.l.: s.n.], 2005. p. 141–142. Citado 2 vezes nas páginas 25 e 31.
- CHEN, M. et al. Simple and deep graph convolutional networks. In: *ICML*. [S.l.: s.n.], 2020. Citado na página 35.
- EL-KISHKY, A. et al. Twihin: Embedding the twitter heterogeneous information network for personalized recommendation. In: *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. [S.l.: s.n.], 2022. p. 2842–2850. Citado 3 vezes nas páginas 25, 31 e 41.
- FEY, M.; LENSSEN, J. E. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019. Citado na página 48.
- GAO, C. et al. Graph neural networks for recommender system. In: *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. [S.l.: s.n.], 2022. p. 1623–1625. Citado na página 29.
- GILMER, J. et al. Neural message passing for quantum chemistry. In: PMLR. *International conference on machine learning*. [S.l.], 2017. p. 1263–1272. Citado na página 30.
- GROVER, A.; ZWEIG, A.; ERMON, S. Graphite: Iterative generative modeling of graphs. In: *ICML*. [S.l.: s.n.], 2019. Citado na página 35.
- GUO, C. et al. On calibration of modern neural networks. In: PMLR. *International conference on machine learning*. [S.l.], 2017. p. 1321–1330. Citado 5 vezes nas páginas 26, 41, 43, 45 e 48.
- HAMILTON, W.; YING, Z.; LESKOVEC, J. Inductive representation learning on large graphs. *Advances in neural information processing systems*, v. 30, 2017. Citado 4 vezes nas páginas 30, 32, 33 e 48.
- HAMILTON, W. L. Graph representation learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, Morgan and Claypool, v. 14, n. 3, p. 1–159, 2020. Citado na página 30.
- HANDA, A.; SHARMA, A.; SHUKLA, S. K. Machine learning in cybersecurity: A review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Wiley Online Library, v. 9, n. 4, p. e1306, 2019. Citado na página 25.

- HSU, H. H.-H. et al. What makes graph neural networks miscalibrated? *arXiv preprint arXiv:2210.06391*, 2022. Citado 5 vezes nas páginas 25, 26, 41, 44 e 45.
- HUANG, Q. et al. Combining label propagation and simple models out-performs graph neural networks. In: *ICLR*. [S.l.: s.n.], 2021. Citado na página 35.
- IVANOV, S.; BURNAEV, E. Anonymous walk embeddings. In: PMLR. *International conference on machine learning*. [S.l.], 2018. p. 2186–2195. Citado na página 30.
- JIN, J. et al. A gan-based short-term link traffic prediction approach for urban road networks under a parallel learning framework. *IEEE Transactions on Intelligent Transportation Systems*, IEEE, v. 23, n. 9, p. 16185–16196, 2022. Citado 3 vezes nas páginas 25, 31 e 41.
- JIN, W.; BARZILAY, R.; JAAKKOLA, T. Junction tree variational autoencoder for molecular graph generation. In: *ICML*. [S.l.: s.n.], 2018. Citado na página 35.
- KINGMA, D. P.; BA, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. Citado na página 48.
- KIPF, T. N.; WELING, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. Citado 4 vezes nas páginas 25, 30, 33 e 37.
- KIPF, T. N.; WELING, M. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016. Citado 7 vezes nas páginas 25, 30, 32, 33, 35, 38 e 48.
- LI, X.-S. et al. Multiphysical graph neural network (mp-gnn) for covid-19 drug design. *Briefings in Bioinformatics*, Oxford University Press, v. 23, n. 4, p. bbac231, 2022. Citado na página 29.
- LIBEN-NOWELL, D.; KLEINBERG, J. The link prediction problem for social networks. In: *Proceedings of the twelfth international conference on Information and knowledge management*. [S.l.: s.n.], 2003. p. 556–559. Citado na página 31.
- LIU, G. An ecommerce recommendation algorithm based on link prediction. *Alexandria Engineering Journal*, Elsevier, v. 61, n. 1, p. 905–910, 2022. Citado 3 vezes nas páginas 25, 31 e 41.
- LIU, J. et al. Fa-gnn: Filter and augment graph neural networks for account classification in ethereum. *IEEE Transactions on Network Science and Engineering*, IEEE, v. 9, n. 4, p. 2579–2588, 2022. Citado na página 29.
- LU, B. et al. Spatiotemporal adaptive gated graph convolution network for urban traffic flow forecasting. In: *Proceedings of the 29th ACM International conference on information & knowledge management*. [S.l.: s.n.], 2020. p. 1025–1034. Citado 3 vezes nas páginas 25, 31 e 41.
- MESQUITA, D. et al. Rethinking pooling in graph neural networks. In: *NeurIPS*. [S.l.: s.n.], 2020. Citado na página 35.
- MICHELI, A. Neural network for graphs: A contextual constructive approach. *IEEE Transactions on Neural Networks*, v. 20, n. 3, p. 498–511, 2009. Citado na página 35.
- NAEINI, M. P.; COOPER, G.; HAUSKRECHT, M. Obtaining well calibrated probabilities using bayesian binning. In: *Proceedings of the AAAI conference on artificial intelligence*. [S.l.: s.n.], 2015. v. 29, n. 1. Citado 3 vezes nas páginas 26, 43 e 48.
- NASCIMENTO, E. J. F. do et al. Procedural generation of isometric racetracks using chain code for racing games. In: IEEE. *2021 20th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*. [S.l.], 2021. p. 136–143. Citado na página 25.
- NASCIMENTO, E. J. F. do et al. Classificando graus de pterígio utilizando aprendizado de máquina. *Journal of Health Informatics*, v. 12, 2020. Citado na página 25.
- NASCIMENTO, E. J. F. do; SOUZA, A. H.; MESQUITA, D. Improving graph variational autoencoders with multi-hop simple convolutions. In: *ESANN*. [S.l.: s.n.], 2021. Citado na página 25.

- NIE, K.; ZENG, K.; MENG, Q. Knowledge reasoning method for military decision support knowledge graph mixing rule and graph neural networks learning together. In: IEEE. *2020 Chinese Automation Congress (CAC)*. [S.l.], 2020. p. 4013–4018. Citado na página 29.
- PASZKE, A. et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, v. 32, 2019. Citado na página 48.
- PLATT, J. et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, Cambridge, MA, v. 10, n. 3, p. 61–74, 1999. Citado na página 26.
- PRADHYUMNA, P.; SHREYA, G. et al. Graph neural network (gnn) in image and video understanding using deep learning for computer vision applications. In: IEEE. *2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC)*. [S.l.], 2021. p. 1183–1189. Citado na página 29.
- ROZEMBERCZKI, B.; ALLEN, C.; SARKAR, R. Multi-scale attributed node embedding. *Journal of Complex Networks*, Oxford University Press, v. 9, n. 2, p. cnab014, 2021. Citado na página 47.
- SALHA, G. et al. Simple and effective graph autoencoders with one-hop linear models. In: *ECML-PKDD*. [S.l.: s.n.], 2020. Citado 3 vezes nas páginas 35, 37 e 39.
- SANCHEZ-LENGELING, B. et al. *A gentle introduction to graph neural networks*. 2021. Disponível em: <<https://distill.pub/2021/gnn-intro/>>. Citado 2 vezes nas páginas 29 e 31.
- SATORRAS, V. G.; HOOGEBOOM, E.; WELLING, M. E (n) equivariant graph neural networks. In: PMLR. *International conference on machine learning*. [S.l.], 2021. p. 9323–9332. Citado na página 25.
- SCARSELLI, F. et al. The graph neural network model. *IEEE transactions on neural networks*, IEEE, v. 20, n. 1, p. 61–80, 2008. Citado na página 25.
- SEN, P. et al. Collective classification in network data. *AI magazine*, v. 29, n. 3, p. 93–93, 2008. Citado na página 47.
- SIMONOVSKY, M.; KOMODAKIS, N. GraphVAE: Towards generation of small graphs using variational autoencoders. In: *ICANN*. [S.l.: s.n.], 2018. Citado na página 35.
- SUN, C. et al. How to fine-tune bert for text classification? In: SPRINGER. *Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings 18*. [S.l.], 2019. p. 194–206. Citado na página 25.
- VELIČKOVIĆ, P. et al. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017. Citado 2 vezes nas páginas 25 e 30.
- WANG, H. et al. Equivariant and stable positional encoding for more powerful graph neural networks. *arXiv preprint arXiv:2203.00199*, 2022. Citado 2 vezes nas páginas 25 e 30.
- WANG, X. et al. Be confident! towards trustworthy graph neural networks via confidence calibration. *Advances in Neural Information Processing Systems*, v. 34, p. 23768–23779, 2021. Citado 2 vezes nas páginas 26 e 47.
- WEN, Y. et al. Session-based recommendation with gnn and time-aware memory network. *Mobile Information Systems*, Hindawi, v. 2022, 2022. Citado na página 29.
- WU, F. et al. Simplifying graph convolutional networks. In: PMLR. *International conference on machine learning*. [S.l.], 2019. p. 6861–6871. Citado 5 vezes nas páginas 25, 30, 35, 36 e 38.
- XIONG, J. et al. Graph neural networks for automated de novo drug design. *Drug Discovery Today*, Elsevier, v. 26, n. 6, p. 1382–1393, 2021. Citado na página 25.

- ZADROZNY, B.; ELKAN, C. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In: *Icml*. [S.l.: s.n.], 2001. v. 1, p. 609–616. Citado 3 vezes nas páginas 26, 42 e 48.
- ZADROZNY, B.; ELKAN, C. Transforming classifier scores into accurate multiclass probability estimates. In: *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. [S.l.: s.n.], 2002. p. 694–699. Citado 3 vezes nas páginas 26, 43 e 48.
- ZEB, A. et al. Complex graph convolutional network for link prediction in knowledge graphs. *Expert Systems with Applications*, Elsevier, v. 200, p. 116796, 2022. Citado 3 vezes nas páginas 25, 31 e 41.
- ZHANG, M.; CHEN, Y. Link prediction based on graph neural networks. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 2018. p. 5165–5175. Citado 2 vezes nas páginas 25 e 30.
- ZHANG, M. et al. An end-to-end deep learning architecture for graph classification. In: *Proceedings of the AAAI conference on artificial intelligence*. [S.l.: s.n.], 2018. v. 32, n. 1. Citado na página 30.
- ZHANG, M. et al. Dynamic graph neural networks for sequential recommendation. *IEEE Transactions on Knowledge and Data Engineering*, IEEE, 2022. Citado 3 vezes nas páginas 25, 31 e 41.
- ZHANG, P.; CHARTRAND, G. *Introduction to graph theory*. [S.l.]: Tata McGraw-Hill, 2006. Citado 2 vezes nas páginas 29 e 30.
- ZHOU, Y. et al. A reliable traffic prediction approach for bike-sharing system by exploiting rich information with temporal link prediction strategy. *Transactions in GIS*, Wiley Online Library, v. 23, n. 5, p. 1125–1151, 2019. Citado 3 vezes nas páginas 25, 31 e 41.