

UNIVERSIDADE FEDERAL DO RIO GRANDE - FURG
CENTRO DE CIÊNCIAS COMPUTACIONAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO
CURSO DE MESTRADO EM ENGENHARIA DE COMPUTAÇÃO

Dissertação de Mestrado

**Redes Neurais em Grafos: um benchmarking de métodos
de classificação de documentos de engenharia**

Vanessa Telles da Silva

Dissertação de Mestrado apresentada ao Programa
de Pós-Graduação em Computação da Universi-
dade Federal do Rio Grande - FURG, como requi-
sito parcial para a obtenção do grau de Mestre em
Engenharia de Computação

Orientador: Prof. Dr. Nelson Duarte Filho
Co-orientador: Prof. Dr. Marcelo Rita Pias

Rio Grande, 2021

Ficha Catalográfica

S586r Silva, Vanessa Telles da.

Redes neurais em grafos: um *benchmarking* de métodos de classificação de documentos de engenharia / Vanessa Telles da Silva. – 2021.

123 f.

Dissertação (mestrado) – Universidade Federal do Rio Grande – FURG, Programa de Pós-Graduação em Computação, Rio Grande/RS, 2021.

Orientador: Dr. Nelson Duarte Filho.

Coorientador: Dr. Marcelo Rita Pias.

1. *Deep learning* 2. Grafos 3. Semi-supervisionado 4. Redes Neurais I. Duarte Filho, Nelson II. Pias, Marcelo Rita III. Título.

CDU 004

Catálogo na Fonte: Bibliotecário José Paulo dos Santos CRB 10/2344



MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO RIO GRANDE
CENTRO DE CIÊNCIAS COMPUTACIONAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO
CURSO DE MESTRADO EM ENGENHARIA DE COMPUTAÇÃO

DISSERTAÇÃO DE MESTRADO

Redes Neurais em Grafos: Um benchmarking de dados e métodos de classificação de nodos

Vanessa Telles da Silva

Banca examinadora:

Prof. Dr. Ricardo Matsumura de Araujo

Prof. Dr. Paulo Lilles Jorge Drews Junior

Prof. Dr. Marcelo Rita Pias
Coorientador

Prof. Dr. Nelson Lopes Duarte Filho
Orientador

*Dedico esta pesquisa a Noeli e Dalila.
Mulheres excepcionais que fizeram de mim a pessoa que sou hoje.*

AGRADECIMENTOS

Gostaria de agradecer minha família por sempre terem me incentivado e acreditado em meu potencial, principalmente nos momentos em que eu mesma não acreditei. Em meio as adversidades que passamos encontrei as forças para continuar seguindo meus sonhos, independente da dificuldade ou de quão inalcançáveis eles aparentam ser. Especialmente para Richard, Ingrid, Nicolly, Isabella, Laura e Helena, os quais espero poder servir como exemplo de resiliência, perseverança e também de curiosidade e teimosia.

Agradeço ao meu namorado Gabriel por aguentar todas as crises de ansiedade, por ouvir as inúmeras lamentações proferidas madrugada adentro e por sempre oferecer, do seu jeitinho, palavras de conforto, incentivo e merecidos puxões de orelha. Sem você esse trabalho não seria possível.

Ao meu tio avô Paulo por ter acreditado em meu potencial, por ter desempenhado o papel de exemplo em nossa família e por ter sido o maior apoiador em minha vida acadêmica.

Agradeço também a meus sogros e cunhado, pelo acolhimento e força que passaram durante toda essa jornada. Levo comigo os conselhos, momentos de descontração e desabafos que me fizeram seguir em frente nos dias em que eu pensava em desistir. Vocês fazem parte tanto do meu crescimento profissional, quanto pessoal e por isso sou eternamente grata.

Agradeço a todos os professores que contribuíram com a minha trajetória acadêmica compartilhando de suas sabedorias e experiencias. Especialmente aos professores Nelson Duarte Filho, Marcelo Rita Pias e Sílvia Silva da Costa Botelho, pelos ensinamentos, incentivo, paciência e por terem exigido de mim muito mais do que eu sequer imaginava ser capaz de oferecer.

E por fim agradeço aos colegas que tive oportunidade de trabalhar e compartilhar conhecimentos, desafios, frases de motivação, surtos, noites em claro, maratonas de programação, pequenas vitórias e a cereja do bolo.

*Some lights are ambition
Some lights are wanderings
Lights of people
Each one of them is precious
On a dark night, don't feel alone
Like stars, we shine
Don't disappear
Because you are one great existence
Let us shine.*

— MIKROKOSMOS, BTS

RESUMO

SILVA, Vanessa Telles da. **Redes Neurais em Grafos: um benchmarking de métodos de classificação de documentos de engenharia**. 2021. 123 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal do Rio Grande - FURG, Rio Grande.

A criação de novos dados cresce em um ritmo acelerado, trazendo benefícios para a sociedade. No entanto, o volume de dados, tanto estruturado quanto não estruturado, se mostra um desafio considerável em entregar acesso imediato dos dados aos usuários. A Transformação Digital propõe avanços quanto ao manuseio de dados digitais e documento impressos na gestão centrada em documentos nas mais diversas áreas de atuação. Essa tarefa é essencial pois é responsável por informar e guiar os membros da equipe em projeto na área de construção, montagem e em engenharia. É comum que organizações dessa natureza trabalhem com centenas, ou até milhares, de documentos técnicos, usualmente impressos, como plantas industriais, ordens de compra para peças, certificados de qualidades, ensaios com material e outros tipos. Com demanda atual da indústria por eficiência, a inspeção manual dos documentos técnicos é custosa. Neste contexto, aplicações de cenários reais se beneficiariam com a implementação de um sistema automático de classificação que consiga lidar com o grande volume de dados existentes. As soluções propostas atualmente tem como base a utilização de um grande volume de dados e acabam negligenciando o fato de que estes documentos podem estar interligados. O trabalho aqui proposto tem como objetivo aprofundar a compreensão quanto ao desempenho de Redes Neurais baseada em Grafos (GNN), com uma abordagem simples e eficiente para a classificação de documentos. Os resultados deste trabalho indicam que modelos de GNN com aprendizado semi-supervisionado podem ser escolhas de projeto problemáticas quando os dados da aplicação tem alto grau de relacionamentos e estes elos advêm de relações indiretas. Mostrando a necessidade de se repensar as metodologias atuais em projetos de aprendizado de máquina, trazendo a discussão para o espaço de dados. O trabalho propõe implementar a exploração do *Data Space* aos passos que compõem projetos de GNN. Essa abordagem alinha-se à tendência atual do aprendizado de máquina centrado no dado (*data-centric*).

Palavras-chave: Deep learning, Grafos, Semi-supervisionado., Redes Neurais.

ABSTRACT

SILVA, Vanessa Telles da. **Graph Neural Networks: a benchmarking of document classification methods**. 2021. 123 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal do Rio Grande - FURG, Rio Grande.

Newly available data has grown fast in the past few years, bringing benefits to society. However, the sheer volume of data, structured or unstructured, poses a considerable challenge in providing immediate access to end-users. Digital transformation, in many organizations, promises to advance the handling of digital and legacy printed data as part of document-centered management. This is a crucial task to inform and guide staff members in construction, engineering, and built environments. It is not unusual for organisations to deal with hundreds or thousands of printed technical documents such as industrial plant design, purchase orders, quality certificates, material tests, etc. Manually inspecting these technical documents is costly in today's industry demand for efficiency. This real-life application would benefit tremendously from an automated classification system that could sift through a sheer amount of data in document files. Currently, proposed solutions rely on a vast amount of labeled data and often overlook the inherent inter-link among documents. This work sheds some light on this problem space. It aims at deepening understanding of the performance of Graph Neural Networks (GNN) as a simple yet powerful approach to accurate document classification. Contributions of the work are as follows: performance validation of a few GNN-based document classifiers, devising a novel engineering document dataset, and accuracy profiling of GNN classifier built out of the novel dataset. Results suggested that GNN models with semi-supervised learning can be problematic design choices in cases that data is highly connected and these links come from indirect relationships. It shows is time to rethink "de facto" machine learning methodologies as part of a revisit to data space. This work attempts to propose Data Space exploration steps to be added to future GNN projects workflow. Such an approach is aligned with recent calls for a Data-Centric A.I. mindset.

Keywords: Deep Learning, Graph, Semi-supervised, Neural Networks.

LISTA DE FIGURAS

1	Exemplos de páginas que compõem um típico <i>databook</i>	20
2	Exemplo das ligações internas e externas entre <i>databooks</i>	21
3	Exemplo das ligações em artigos.	22
4	Exemplo de visualização de dados através da estrutura de grafos. . . .	27
5	Arquitetura de uma RNA.	29
6	Layout de uma GCN.	33
7	Metodologia geral do trabalho proposto.	46
8	<i>Dataset</i> utilizado para retirar as informações de conexões entre os <i>databooks</i>	53
9	Arquivos que compõem o dataset EPC.	55
10	Modelo GCN: Acurácia média durante o processo de treinamento. . .	62
11	Modelos GCN: Comparação de <i>loss</i> no treinamento e validação. . .	63
12	Modelo Cheb: Valores de acurácia durante o processo de treinamento.	64
13	Modelos Cheb: Comparação entre os valores de <i>loss</i> para o conjunto de treino e validação.	64
14	Modelo ARMA: Valores de acurácia no processo de treinamento. . .	65
15	Modelos ARMA: Comparação entre as curvas de <i>loss</i>	66
16	Modelo GAT: Valores de acurácia durante o processo de treinamento. .	67
17	Modelos GAT: Comparação entre curvas <i>loss</i> de treino e validação. .	67
18	Valores de acurácia de teste variando a função de ativação	68
19	Valores de acurácia para dados de teste variando a função de <i>loss</i> . .	71
20	Valores de acurácia de teste variando os valores de drop-out.	74
21	EPC: Valores de acurácia durante o processo de treinamento.	79
22	EPC: <i>Loss</i> no treinamento versus teste.	80
23	<i>GNN Design Space</i>	82
24	Resultados para o dataset EPC com conexões do tipo Registro.	83
25	Resultados para o dataset EPC com conexões do tipo Lote.	84
26	Resultados para o dataset EPC - grafo com elos Corrida.	85
27	Resultados para o dataset EPC com conexões do tipo Fornecedor. . .	86
28	Resultados para o dataset EPC com conexões do tipo Certificado. . .	86
29	Grafo do dataset Cora.	90
30	Grafo do dataset Citeseer.	91
31	Grafo do dataset Pubmed.	91

32	Grafo do dataset EPC com conexão do tipo Certificado.	92
33	Grafo do dataset EPC com conexão do tipo Corrida.	92
34	Grafo do dataset EPC com conexão do tipo Fornecedor.	93
35	Grafo do dataset EPC com conexão do tipo Lote.	93
36	Grafo do dataset EPC com conexão do tipo Registro.	94
37	Métricas Transitivity e Average Clustering.	96
38	Métricas degree e k-cores.	97
39	Proporção de classes em cada dataset.	98
40	Proposta do GNN Data Space.	100
41	Valores de acurácia de teste variando as épocas.	119
42	Modelo GCN: Duração dos processos de treino.	120
43	Duração dos processo de de treino variando a função de ativação. . .	120
44	Duração dos processo de de treino variando a função de loss	121
45	Duração dos processos de treino variando o Drop out.	121
46	Resultados para o dataset Cora.	122
47	Resultados para o dataset Citeseer.	122
48	Resultados para o dataset Pubmed.	122
49	Resultados para o dataset EPC com tipo de conexão Registro com aprendizagem supervisionada.	123

LISTA DE TABELAS

1	Informações sobre os databases utilizados.	49
2	Quantidades de instâncias em cada classe.	49
3	Quantidades de features em cada label.	50
4	Quantidades de instâncias em cada classe do dataset EPC.	52
5	Quantidades de features em cada classe do dataset EPC.	53
6	Separação dos dados utilizada.	57
7	Experimentos com os modelos de <i>baseline</i> e configuração padrão de GNNs.	60
8	Valores de acurácia média da literatura após 200 épocas de treinamento.	62
9	Valores de acurácia de teste variando a função de ativação.	69
10	Valores de <i>loss</i> encontrados para o conjunto de validação variando a função de ativação.	69
11	Valores de <i>loss</i> encontrados para o conjunto de treino variando a função de ativação.	69
12	Valores de acurácia (dados de teste) variando a função de <i>loss</i>	72
13	Valores de <i>loss</i> obtidos com o conjunto de validação variando a função de <i>loss</i>	72
14	Valores de <i>loss</i> encontrados para o conjunto de treino variando a função de <i>loss</i>	72
15	Valores de <i>loss</i> encontrados para o conjunto de validação variando o drop-out.	73
16	Valores de <i>loss</i> encontrados para o conjunto de treino variando o drop-out.	74
17	EPC: Valores de acurácia de teste e duração do treino.	81
18	Configurações com os melhores resultados encontrados no conjunto de teste para cada dataset.	87
19	Valores da métrica degree.	95
20	Valores das métricas Cluster Transitivity e Average Clustering.	95
21	Valores das métricas K-cores.	96
22	Valores de acurácia de teste para o dataset Cora.	114
23	Valores de acurácia de teste para o dataset Citeseer.	115
24	Valores de acurácia de teste para o dataset Pubmed.	115
25	Valores de acurácia de teste variando os valores de drop out.	116

26	Cora: Duração em minutos do processo de treino variando o número de épocas.	116
27	Citeseer: Duração em minutos do processo de treino variando o número de épocas.	117
28	Pubmed: Duração em minutos do processo de treino variando o número de épocas.	117
29	Duração em minutos do processo de treino variando a função de ativação.	118
30	Duração em minutos do processo de treino variando a função loss. . .	118
31	Duração em minutos do processo de treino variando o valor de drop out.	118

LISTA DE ABREVIATURAS E SIGLAS

ARMA	Autoregressive Moving Average Model
CHEB	Chebyshev
CNN	Convolutional Neural Network
DIG	Domain Insight Graphs
DTI	Drug Target Interactions
ELU	Exponential Linear Unit
EPC	Engineering, Procurement and Construction
GAT	Graph Attention Networks
GCN	Graph Convolutional Networks
GNN	Graph Neural Networks
GRM	Graph Reasoning Model
HTG	Hierarchical Topic Graph
IDC	International Data Corporation
KNN	K-Nearest Neighbord
LSTM	Long Short Term Memomry
OCR	Optical Character Recognition
NLL	Negative Log Likelihood
ONU	Organização das Nações Unidas
RNA	Redes Neurais Artificiais
RBM	Restricted Boltzmann Machines
ReLU	Rectified Linear Unit
RLU	Remaining Useful Life
PIPA	People in Photo Album
PISC	People in Social Context
PyG	PyTorch Geometric
SGD	Stochastic Gradient Descent

SVM	Support Vector Machines
VQA	Visual Question Answering
XML	Extensible Markup Language

SUMÁRIO

1	INTRODUÇÃO	17
1.1	Projeto <i>Análise de Textos Técnicos - ATT</i>	18
1.1.1	<i>Databooks</i>	19
1.2	Motivação	20
1.3	O Problema de Interesse	22
1.3.1	Aprendizado Semi-Supervisionado	23
1.3.2	Aprendizado Semi-Supervisionado em Grafos	23
1.4	Objetivos	24
1.5	Estrutura do Trabalho	24
2	FUNDAMENTAÇÃO TEÓRICA	26
2.1	Teoria dos Grafos	26
2.2	K-nearest neighbors (KNN)	27
2.3	Redes neurais Artificiais	28
2.4	Graph Neural Networks (GNNs)	30
2.4.1	Graph Convolutional Networks (GCNs)	32
2.4.2	Cheb	34
2.4.3	ARMA	35
2.4.4	GAT	37
2.4.5	Resumindo os modelos de GNN	39
3	REVISÃO BIBLIOGRÁFICA	41
4	METODOLOGIA	45
5	PREPARAÇÃO DOS DADOS	48
5.1	Datasets de Citação	48
5.2	Dataset EPC	50
5.2.1	Features	51
5.2.2	Labels (rótulos)	52
5.2.3	Relacionamentos	53
5.2.4	Arquivos Resultantes	54
6	MODELOS	56
6.1	Modelos de controle	56
6.2	Arquiteturas de GNN	56
6.3	Calibração dos Modelos	57
6.3.1	Modelos <i>Baseline</i> e GNN	58

6.3.2	Configurações de GNNs	58
7	EXPERIMENTOS E DISCUSSÕES	59
7.1	<i>Baseline</i> e Calibração dos modelos GNN	59
7.1.1	Experimentos com épocas	61
7.1.2	Experimentos por Função de ativação	68
7.1.3	Experimentos por Função de loss	71
7.1.4	Experimentos por Drop-out	73
7.1.5	Discussão	76
7.2	Classificação de Documentos EPC	78
7.2.1	GraphGym	81
7.2.2	Discussão	86
8	EXPLORANDO OS DADOS	89
8.1	Visualização dos Grafos	89
8.2	Análise dos Grafos	94
8.3	Características dos <i>datasets</i>	97
8.4	Discussão	98
9	CONCLUSÃO	101
9.1	Contribuições	103
9.2	Recomendações para Trabalhos Futuros	104
	REFERÊNCIAS	105
	ANEXO A TABELAS	114
	ANEXO B FIGURAS	119

1 INTRODUÇÃO

Na era atual de constante desenvolvimento tecnológico, a criação de dados cresce em um ritmo acelerado (Fang et al. 2020). Essa abundância de dados disponível facilita o aprendizado e a qualidade de vida que se tem hoje. Entretanto, ao mesmo tempo, o imenso volume de dados torna cada vez mais difícil o acesso rápido e ordenado a informações precisas (Mao et al. 2019). Além da quantidade, o formato no qual esses dados são armazenados é crucial e afeta diretamente o seu manuseio (Glaubitz 2004), muitas vezes sendo a principal barreira para utilização de certas abordagens. Não obstante, os dados em sua totalidade tendem a ser redundantes, o que acaba por complicar ainda mais a recuperação das informações de forma eficiente (Szulc et al. 2017).

Em outras palavras, pode-se afirmar que, não somente o “como” identificar informações virou um problema, mas também o “modo” de relacionar e cruzar os dados para obter novas informações também assim se tornou (Rajaraman & Ullman 2011) (Hovy et al. 2013) (Schenker et al. 2003).

Neste âmbito, vale ressaltar que, apesar dos avanços da computação nos dias de hoje, os formatos de dados utilizados nos documentos impressos ainda são mecanismos presentes que atuam como peça central para gerenciar, informar e instruir diversas aplicações (Richter et al. 2018). Tendo em vista que documentos de caráter oficial e com grande grau de relevância tendem ainda a serem conservados em cópias físicas, em papel, de forma a criarem evidências tangíveis (Thongkor & Amornraksa 2017).

Estima-se que, para imprimir documentos, apenas um funcionário de escritório usa em média 10.000 folhas de papel por ano (Illinois Library 2021) e passa de 30% a 40% do seu tempo de trabalho em busca de documentos previamente impressos (Forbes 2021). Segundo o *Record Nations*, corporações gastam em média mais de 120 bilhões de dólares por ano em formulários impressos (Record Nations 2021).

A área de engenharia não é uma exceção neste contexto. Organizações dessa natureza trabalham com centenas ou até milhares de documentos técnicos como plantas industriais, ordens de compra, certificados de qualidades, certificados de material e outros tipos (Amor & Clift 1997), os quais usualmente são apresentados de forma física, impressos. O gerenciamento cuidadoso desses documentos é essencial pois fornece os

meios necessários para coordenar atividades fundamentais para conclusão de projetos de construção, montagem e até mesmo para processos de compra e venda de produtos (Zheng et al. 2019).

Um dos principais desafios quanto ao gerenciamento de informação em documentos industriais se dá pelo fato de que a formalização e visualização de dados provenientes destes, diferente das advindas de banco de dados, são consideradas tarefas complexas devido ao conteúdo das informações usualmente possuírem naturezas diversificadas, não sendo estruturadas. Isso dificulta abordagens de buscas mais usuais como, por exemplo, a busca de sentenças por palavras chaves (Matsubayashi et al. 2018). Quando a fonte de dados é proveniente de papel (Dahl & Sørensen 2019), como muitas vezes é o caso de relatórios de engenharia, manuais, documentos técnicos como plantas, esquemáticos e outros, estes desafios são somados a etapas de pré processamento e digitalização utilizando ferramentas, como por exemplo o OCR (*Optical Character Recognition*).

Uma vez que a quantidade de documentos gerados em um determinado ambiente seja elevada, tarefas como a classificação de forma manual se tornam extremamente custosas e improdutivas (Liu et al. 2017), gerando a necessidade de sistemas que sejam capazes de categorizar grandes volumes de documentos de forma automatizada. Esta é uma etapa crucial que exige níveis de atenção e cuidado elevados, pois uma definição errônea quanto ao tipo do documento acaba por prejudicar todo o processo de busca, recuperação, exploração e retirada de conhecimento atrelados aquele documento. Muitas vezes esta etapa mal elaborada pode prejudicar a especificação de possíveis relacionamentos (ou elos de conectividade) entre documentos que de certa forma auxiliam na tarefa de classificação.

1.1 Projeto *Análise de Textos Técnicos - ATT*

Durante a realização de obras de construção e engenharia, documentos do tipo EPC (Engineering, Procurement and Construction) (Kabirifar & Mojtahedi 2019) devem passar por um processo de fiscalização realizado por profissionais capacitados e licenciados, os quais manualmente analisam cada página, conferindo completude, conformidade e buscando por anormalidades relacionadas à má-fé e fraudes.

O processo de completude visa verificar se todos os arquivos que deveriam estar presente nos documentos EPC de fato foram encontrados. Por sua vez, conformidade corresponde a verificar se as informações técnicas contidas nos arquivos estão dentro dos limites aceitáveis estabelecidos pelas normativas técnicas específicas de cada elemento envolvido. Já a busca por anormalidades visa encontrar situações que estejam fora dos padrões de normalidade, podendo configurar-se sob a forma de erros nos documentos como a falta de um carimbo, de uma assinatura ou possíveis situações de má-fé, como a reutilização de um certificado antigo seja novamente utilizado para certificar um novo

lote de matéria-prima. Esta análise de anormalidades é um processo de exploração baseado nas experiências e vivências dos inspetores licenciados, o que faz dela uma prática subjetiva, sem uma definição formal, devendo ser minuciosamente investigada.

A abordagem manual de inspeção é lenta, custosa e inconveniente, no caso de grande obras, podendo ultrapassar dez mil páginas de análise. Em decorrência disso, a FURG vem realizando atividades de PD&I, sob a forma de um projeto, visando implementar um sistema de software capaz de adquirir, digitalizar, interpretar e avaliar documentos relativos as atividades de EPC.

Assim, o Projeto ATT visa abordar essas situações, desde o início do processo de inspeção técnica, passando pela digitalização dos documentos, convertendo os dados escaneados no formato imagem para um formato digital editável, que possibilite o tratamento das informações neles contidos. Com os documentos digitalizados, as páginas dos mesmos são separadas e submetidas a algoritmos responsáveis pela extração das informações que fazem parte do domínio de entendimento de formulários (*form-understanding*). Com essas informações extraídas, elas devem ser estruturadas em um formato hierárquico de modo que ao serem armazenadas mantenham as relações existentes entre os dados nelas contidos. Dessa forma facilitando a interpretação do contexto ao qual eles pertencem. Tendo os dados brutos extraídos e armazenados, eles são disponibilizados em uma plataforma de visualização e manipulação, onde os inspetores podem realizar buscas, explorar e visualizar os dados dos documentos e os seus relacionamentos.

1.1.1 Databooks

Os documentos técnicos focos nesta dissertação são denominados *databooks*. Tratam-se de conjuntos de arquivos que formam a documentação mínima necessária referente aos processos de compra, fabricação, construção ou montagem de peças e equipamentos para a indústria.

Na Figura 1 ilustram-se exemplos de páginas que compõem um databook. Como se pode observar na Figura, em um único *databook* são encontrados diversos ensaios, certificados de qualidade e testes com *layouts* distintos que variam de acordo com a sua natureza e fornecedor. Tratam-se de documentos que pertencem a classe dos chamados documentos semi-estruturados ¹. Documentos semi-estruturados possuem uma parte padronizada (estruturada), a qual permanece com a mesma configuração, e uma parte a qual tem seu layout variável (não-estruturada).

Durante o processo de extração de informações relevantes dos documentos semi-estruturados, um dos desafios encontrados se dá pela necessidade do sistema conseguir distinguir automaticamente quais os tipos de ensaios, testes e certificados estão presentes

¹ Documentos semi-estruturados neste contexto são formulários em que o posicionamento dos campos de dados varia de documento para documento. Por exemplo, o endereço do fornecedor do parafuso estojo pode estar localizado no canto superior esquerdo ou no rodapé do formulário. Entretanto, este dado precisa ser extraído como o endereço do fornecedor

Figura 1: Exemplos de páginas que compõem um típico *databook*.

(a) Capa.

(b) Página 1.

CERTIFICADO DE QUALIDADE

Nº: [REDACTED] NFº: [REDACTED]

Certificamos para os devidos fins que, os produtos constantes em nosso(s) Pedido Interno(s) [REDACTED] do dia 22/02/2018, em nome da empresa [REDACTED], foram fabricados com matéria prima qualificada e dentro dos mais rigorosos padrões de qualidade e conformidade.

Nº do pedido do cliente (ordem de compra): [REDACTED] Descrição do cliente: [REDACTED]

Item: [REDACTED] 0800/ BARRA ROSCADA 2" BUN X 1000mm - NAT ASTM A193 Gr. B16 (OTO B) / OS Condição: [REDACTED]

Certificado nº: [REDACTED] lote nº: [REDACTED] do fornecedor: [REDACTED]

ANÁLISE QUÍMICA (%)

Material: [REDACTED]

	C	Si	Mn	P	S	Cr	Ni	Mo	Cu	Al	B	Ti	Bi	Zn	Fe	N	V
0.04	0.22	0.38	0.21	0.008	0.005	0.22	0.007										

Observações: V: 0.20

PROPRIEDADES MECÂNICAS

Dimensões (mm)	Seção (mm²)	Lim. de Escoamento (MPa)	Lim. de Resistência (MPa)	Alongamento (%)	Extensão (%)	Dureza	Carga Aplicada
788.71		561.73	561.73	11.40	56.63	27/28/29	

REQUISITOS ADICIONAIS

Requisito	Especificado	Encontrado	Laudo
Microestrutura	ASTM E187	Matéria Reversa	Ap
Descarbonização	ASTM A562F2038	Não	Ap
Avaliação de Rastreabilidade do Sistema	Lote 556612		Ap
Análise Química (Química Total)	ASTM A193 Gr. B16		Ap
Análise Química (Química Total)	ASTM A193 Gr. B16		Ap
Análise de Inclusões	ASTM E437	Matéria Reversa	Ap
Microestrutura	ASTM E437	Matéria Reversa	Ap
Análise de Inclusões	ASTM E437	Matéria Reversa	Ap
Macrografia	ASTM E187	S15V1C1	Ap
Descarbonização	ASTM A562	OK	Ap
Macrografia	ASTM E187	S15V1C1	Ap
Tensão	502 a 550°C	502°C	Ap
Resistência	Máx. 550°C	720°C	Ap
Dureza	Máx. 20 HRC	28.00/28.3 HRC	Ap
Tração	ASTM A193 Gr. B16		Ap
Composição Química	ASTM A193 Gr. B16		Ap

MEIOS DE CONTROLE UTILIZADOS

Cod.	Instrumento	Certif. nº	Validade
PC-45	Paquímetro (0-150 mm)	QMS12317	jun/2018
TR-133	Trena (3 metros)	QMS13417	jun/2018
CR-104P	Cal. Anal. (0-2" x 1/8")	QMS44918	jun/2019
CR-104	Cal. Anal. (0-2" x 1/8")	QMS44918	jun/2019

Observações

Requisito	Especificado	Encontrado
INSP. DIMENSIONAL	100% REALIZADA	100% APROVADA
IDENT. MARCAÇÃO	2" BUN	BAIRO RELEVIO

Atestamos e Qualitamos os produtos citados neste de 22/02/2018

(c) Página 2.

(d) Página 3.

RELATÓRIO DE ENSAIOS METALOGRAFICOS

Nº: [REDACTED] RE Nº: [REDACTED]

Data de Emissão: 03/05/2018

Identificação da amostra: Barra Roscada 2" BUN / Corpo de Prova Representativo

Restabelecimento: Lote [REDACTED] - Condição [REDACTED]

Norma aplicável: ASTM A193/16 Grau B16

1. OBJETIVO

Realizar ensaios metalográficos na amostra fornecida conforme solicitação do cliente.

2. ENSAIOS E RESULTADOS

2.1. EXAME METALOGRAFICO

Preparação: [REDACTED] / AZURE E 511

Método de Ensaio: ASTM E 770 / 5607

Ampliação: 500 vezes

Ampliação: 500 vezes

Ampliação: 500 vezes

Condições de antebraço / ensaio: Temperatura de 20 °C - Umidade: 67%

Equipamento e instrumento utilizados: Microscópio MET-700 - Comprimento de calibração: 1802/181 - Validade: 03/2020

Entrega: Relatório Digital (PDF) - Certificado de validação: nº 046818-01 - Validade: 06/2019

Entregas: Emerson Great Dias - Técnico em Metalurgia

Identificação: Lote [REDACTED] - Condição [REDACTED]

Análise Micrográfica:

Consiste em uma estrutura homogênea e uniforme de matéria reversa.

RELATÓRIO DE INSPEÇÃO DIMENSIONAL

Nº: 07212018

Data: 15/05/2018

Folha: 1/1

Cliente: [REDACTED] PCS: [REDACTED] PI: [REDACTED]

Descrição do Material: BARRA ROSCADA 1.318" BUN X 1000 mm - ASTM A193 - Gr. B16 CLASSE 2

Norma de Referência: ASME B1.1 / ASME B19.2.1

Procedimento: PC-001

Revisão: 00

Croqui

Item **Características** **Especificado** **Encontrado** **AP** **RE**

01	Comprimento = (L1)	Mínimo: 994.412 mm	Máximo: 1005.588 mm	995.10 mm	1001.15 mm	OK
02	Rosca	Calibrador Anel - Passa	CR - 19P - ajustado	OK		
03	Rosca	Calibrador Anel - Não Passa	CR - 19NP - ajustado	OK		
04	Ø de Rosca (Nominal)	1.318" BUN	1.318" BUN	OK		
05	Cavitação	J = 8854	J = 8854	OK		
06	Visual	Inseto de Retrabos	Inseto de Retrabos	OK		
07	Chamfe	Referência 30°	Referência 30°	OK		

Quantidade: 20 peças

Quantidade Amostras: 11

5 peças

Notas:

1) A análise dimensional foi realizada conforme amostragem do lote realizado.

Conforme NBR 8428 N II B16 NQA 2.5

Resultado da Inspeção:

Aprovado ☒ Reprovado ☐ RNC ☐

Inspeção **Controle de Qualidade** **Cliente**

Data: 15/05/18

Data: 15/05/18

Data: 15/05/18

Fonte: Elaborado pelo autor

no *databook*.

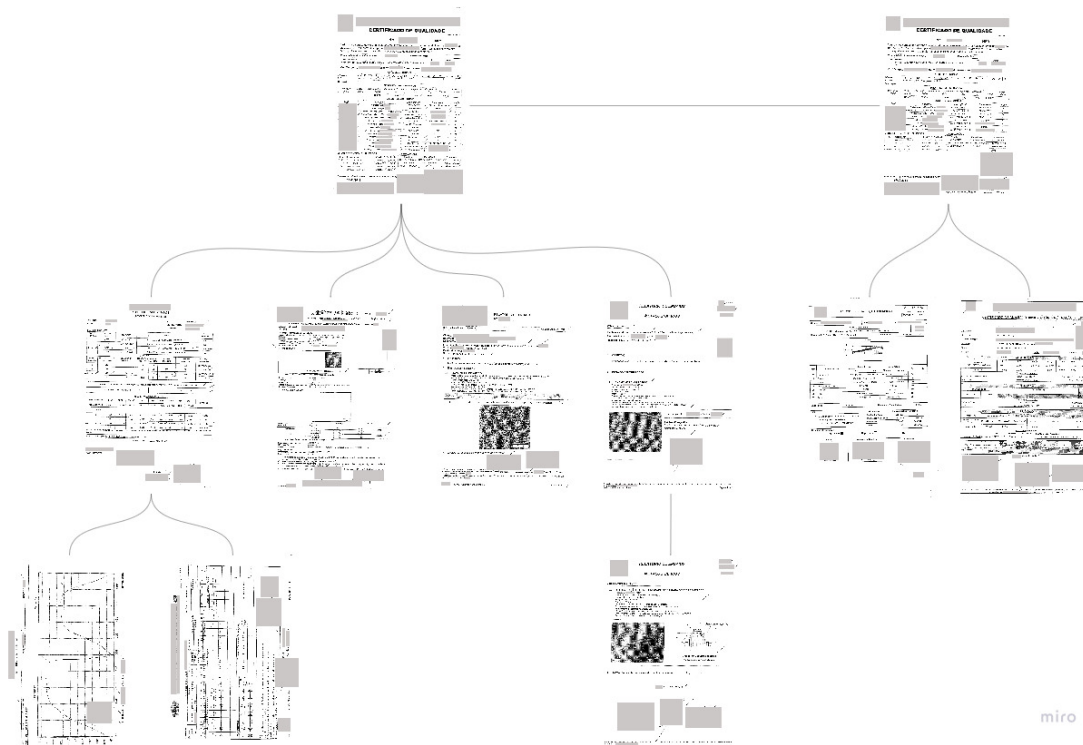
1.2 Motivação

Como mencionado, os documentos relativos a EPC tratados no Projeto ATT possuem diversos tipos de informações. Sendo que essas informações configuram possíveis ligações (ou elos). Tais elos podem ser ligações entre ensaios, testes realizados, etc, com

outros arquivos pertencentes aos *databooks*.

Essas ligações podem assumir dois tipos de origem, como mostra a Figura 2, onde se tem as ligações internas, ou seja, as existentes dentro de um *databook* (ex. a identificação única de um teste) e as ligações externas, entre um ou mais *databooks*, como por exemplo, compartilhar do mesmo fornecedor, fazer parte da mesma ordem de compra, entre outros.

Figura 2: Exemplo das ligações internas e externas entre *databooks*.



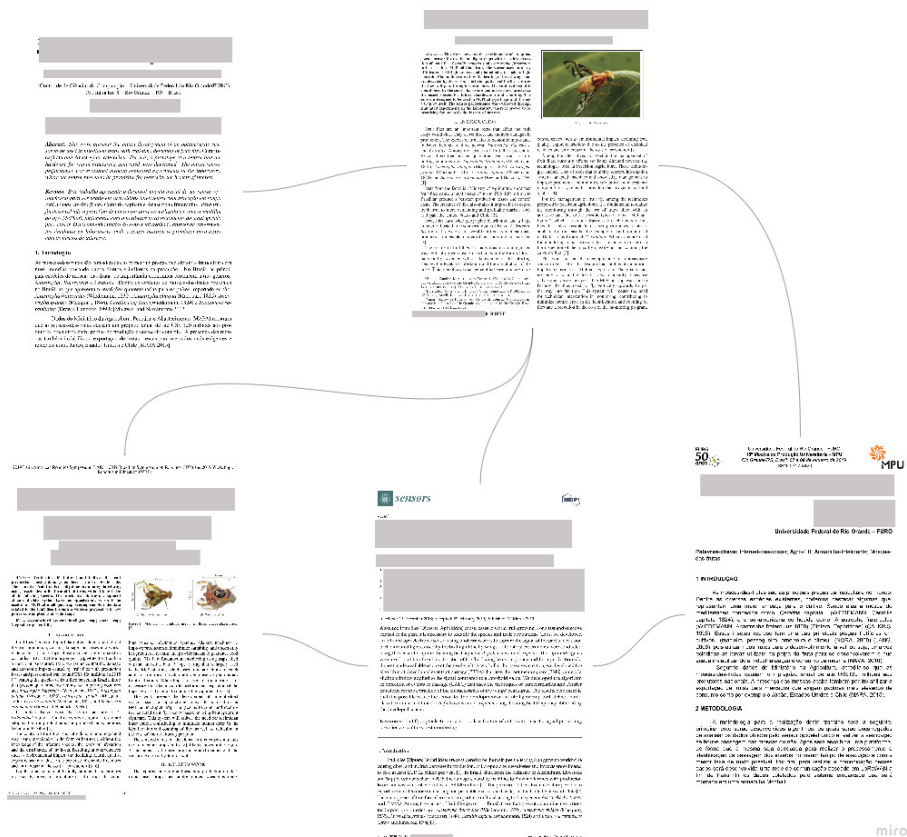
Fonte: Elaborado pelo autor

A configuração interligada de documentos permite a utilização de uma emergente classe de métodos, os quais propõem explorar possíveis relacionamentos entre as entidades, de forma que as conexões existentes podem contribuir para a tarefa de classificação das mesmas. Neste caso, para a classificação das páginas dos documentos que fazem parte de um *databook*.

A necessidade de realizar a classificação de documentos conectados é recorrente em diversas áreas de atuação. Por exemplo, na Figura 2 deseja-se classificar o tipo de teste realizado para mostrar propriedades físicas e químicas do material de uma peça. A Figura 3 apresenta outro tipo de exemplo, o qual artigos científicos podem ser classificados de acordo com as áreas de conhecimento que melhor representam os seus conteúdos.

Em ambos exemplos, para se obter um processo de classificação automatizado é necessário construir um modelo classificador que, a partir de exemplos de documentos rotulados, consiga aprender uma representação média dos padrões presentes em cada documento. Ao lidar com dados provenientes de aplicações como os relativos as atividades de EPC, pode-se obter exemplos rotulados, relativos as classes que se deseja utilizar na

Figura 3: Exemplo das ligações em artigos.



Fonte: Elaborado pelo autor

classificação dos documentos. Porém, o numero de exemplos disponíveis é limitado, uma vez que adquiri-los requer um processo trabalhoso e exaustivo de anotação manual, que muitas vezes é humanamente impossível de ser realizado. Entretanto, exemplos não rotulados, em geral, estão disponíveis em um volume maior, como é o caso dos documentos de EPC. Além de serem abundantes, esses documentos não rotulados possuem um grau de heterogeneidade elevada. Isto se coloca como pano de fundo motivador deste trabalho.

1.3 O Problema de Interesse

Baseado na natureza da aplicação vislumbrada, pode-se definir o problema de interesse como análogo a realizar classificação de documentos numa situação em que existe uma quantidade muito pequena de **dados rotulados** e uma grande quantidade de **dados não rotulados**. Sendo que esse problema pode ser tratado por uma abordagem de aprendizado de máquina denominada semi-supervisionada (em inglês, *Semi-supervised learning*).

Adotando essa abordagem, propõe-se fazer uso da combinação de uma pequena amostra de dados rotulados (*labelled*) com uma grande quantidade de dados sem rótulos (*unlabelled*) para realizar o treinamento de um modelo classificador (Van Engelen & Hoos

2020). Esse aprendizado semi-supervisionado tem por objetivo viabilizar aplicações reais que apresentam severas limitações quanto aos procedimentos necessários a anotação de dados.

1.3.1 Aprendizado Semi-Supervisionado

Contextualizando formalmente, o aprendizado semi-supervisionado considera L e U como o número de instâncias (exemplos) rotuladas (*labelled*) e não-rotuladas (*unlabelled*), respectivamente. Considerando que $x_{1:L}$ e $x_{L+1:L+U}$ denotam vetores de *features* de instâncias rotuladas e não rotuladas, respectivamente (Yang et al. 2016).

Os rótulos $y_{1:L}$ são disponibilizados. Baseando-se nas instâncias rotuladas e não rotuladas, o problema de aprendizado semi-supervisionado é definido como o aprendizado de um classificador $f : x \rightarrow y$.

Existem dois paradigmas de aprendizado semi-supervisionado:

- a. O transdutivo, que objetiva aplicar o classificador f apenas nas instâncias utilizadas durante o treinamento, com isto não generalizando para instâncias desconhecidas;
- b. O aprendizado indutivo, que tem o intuito de obter um classificador parametrizável f que pode generalizar para instâncias que não foram utilizadas no treinamento.

1.3.2 Aprendizado Semi-Supervisionado em Grafos

O princípio adotado na combinação de métodos baseados em grafos e aprendizado semi-supervisionado se dá pela suposição de que instâncias próximas tendem a ter o mesmo rótulo. Neste tipo de abordagem, além dos conjuntos de nodos rotulados e não rotulados, também se deve informar ao modelo o grafo correspondente aos dados, denotado pela matriz A . Essa matriz encontra-se descrita na Equação 1, sendo composta por a_{ij} elementos, os quais indicam uma conexão entre as instâncias i e j , podendo estas serem rotuladas ou não rotuladas.

$$A = (L + U) \times (L + U). \quad (1)$$

O grafo A pode ser construído a partir de redes e elos entre as instâncias, como por exemplo, os fornecedores de um material para uma peça mecânica, ou a rede de citações entre documentos. O grafo no contexto de aprendizado semi-supervisionado representa informação complementar que não se encontra presente nos vetores de *features*. Por exemplo, as arestas no grafo correspondes a *hyperlinks* entre documentos (Yang et al. 2016).

A função de *loss* utilizada é descrita pela Equação 2, na qual o primeiro componente refere-se a uma função de *loss* padrão de aprendizado supervisionado e o segundo componente representa a regularização Laplaciana do grafo, a qual atribui uma penalidade

elevada quando nodos (instâncias) similares com grande w_{ij} são classificados como tendo rótulos diferentes $f(x_i) \neq f(x_j)$.

$$\sum_{i=1}^L l(y_i, f(x_i)) + \lambda \sum_{i,j} a_{ij} \|f(x_i) - f(x_j)\|^2 \quad (2)$$

Segundo a literatura, este tipo de abordagem mostra-se vantajosa em problemas onde há uma abundância de dados não rotulados e o processo de rotular os mesmos é custoso, demorado e até impraticável (Sanches 2003).

1.4 Objetivos

Tem-se como objetivo geral deste trabalho apresentar um estudo comparativo para investigar o desempenho de métodos de redes neurais em grafos (GNNs) ao abordar o problema de classificação de documentos técnicos em um cenário real, por meio do aprendizado semi-supervisionado.

Para alcançar este objetivo, os seguintes objetivos específicos tornaram-se necessários:

1. Reproduzir os resultados obtidos na literatura para os modelos de redes neurais em grafos (GNNs) escolhidos.
2. Realizar uma comparação dos modelos GNNs com intuito de criar um *perfil de calibragem* de parâmetros para os modelos validados.
3. Criar um *dataset* próprio com as características internas de documentos semi-estruturados de EPC e estabelecer os relacionamentos inter-documentos na forma de um grafo.
4. Realizar uma validação quantitativa de um classificador para documentos EPC.
5. Determinar as limitações no processo de treinamento e validação de modelos GNNs, e propor tanto melhorias quanto ajustes.

1.5 Estrutura do Trabalho

Esta dissertação está organizada da seguinte forma. No capítulo 2 é apresentada uma breve fundamentação teórica, introduzindo conceitos necessários para o entendimento do trabalho realizado como os aspectos de Redes Neurais em Grafos (*Graph Neural Networks* - GNNs). No Capítulo 3 é feita uma revisão bibliográfica com uma breve apresentação quanto a trabalhos relacionados. O capítulo 4 apresenta a metodologia geral utilizada durante o desenvolvimento deste trabalho. Seguido pelos capítulos 5 e 6, os quais, respectivamente, descrevem a preparação dos dados e dos modelos utilizados. No

capítulo 7 são apresentados e discutidos os resultados obtidos pelos experimentos realizados. Já no capítulo 8 é apresentado um processo de exploração quanto as características e estruturas dos dados. Por fim, no capítulo 9 são apresentadas a conclusão e sugestões de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são resumidamente abordados alguns conceitos de base teórica que suportam o desenvolvimento do trabalho aqui proposto. Uma introdução é feita à teoria de grafos, ao algoritmo K-nearest neighbors e sobre arquiteturas de *Graph Neural Networks*, ou Redes Neurais em Grafos, a qual é seguida pela descrição dos aspectos construtivos e os princípios de funcionamento de cada uma das arquiteturas de redes escolhidas para o desenvolvimento da dissertação.

2.1 Teoria dos Grafos

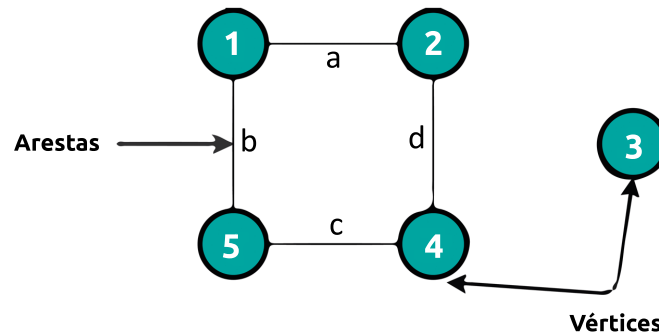
Diversas situações reais podem ser descritas em função de uma representação gráfica composta por um conjunto de pontos com arestas fazendo a ligação entre dois desses pontos (Bondy et al. 1976). Esse tipo de representação é amplamente utilizada na descrição de redes físicas como circuitos elétricos, rodovias, interconexão de computadores, descrição de moléculas orgânicas como a estrutura de proteínas, ou até mesmo a descrição de relações menos tangíveis como relacionamentos sociológicos, *databases*, fluxo de controle de um programa de computador, relacionamentos dos usuários em uma rede social e muitos outros exemplos. Qualquer sistema que possa ser representado usando essa descrição de pontos e conexões (ou elos/relacionamentos) entre eles pode ser chamado de grafo (Gross & Yellen 2004).

Na matemática, a teoria dos grafos foi primeiramente desenvolvida no século XVIII (Petrella 2011). A origem é geralmente creditada à publicação feita por Euler em 1736, na qual, ao representar o problema das pontes de *Koningsberg* através de pontos e as ligações entre eles, conseguiu comprovar que era impossível cruzar as sete pontes em uma caminhada contínua e retornar ao ponto de partida sem atravessar duas vezes por qualquer uma das pontes (Euler 1741).

Um Grafo é conceitualmente definido como uma estrutura matemática utilizada para representar relações entre pares de objetos (West et al. 1996), de forma que o Grafo G é descrito como $G = (V, E)$, onde V representa o conjunto de vértices (ou nodos) e E representa as arestas (ou links) entre eles (Gross & Yellen 2004). Na Figura 4 tem-se

a representação do que seria um grafo com os seus respectivos conjuntos V e E . Os pontos e ligações no problema das pontes de *Koningsberg* referem-se aos vértices/nodos e arestas/links, respectivamente.

Figura 4: Exemplo de visualização de dados através da estrutura de grafos.



$$G = (V, E) \text{ onde: } \begin{aligned} V &= \{1, 2, 3, 4, 5\} \\ E &= \{a, b, c, d\} \end{aligned}$$

Fonte: Elaborado pelo autor

Grafos são extremamente vantajosos para entender as características de diversos conjuntos de dados nas mais variadas áreas de aplicação como a da saúde, governamental, ciência, finanças e outras (Robinson et al. 2015).

Representar sistemas na forma de grafos auxilia não apenas na análise visual do problema, assim como o mesmo pode ser abordado sob o ponto de vista matemático e computacional (Riaz & Ali 2011) (Godsil & Royle 2013). Isso decorre do fato de que grafos podem ser facilmente descritos por uma matriz, geralmente chamada de matriz de adjacências, o que possibilita a aplicação de técnicas algébricas associadas a matrizes. Com isso possibilitando o estudo das propriedades estruturais do grafo (Godsil & Royle 2013). Também a representação de grafos utilizando matrizes é computacionalmente relevante, tendo em vista que matrizes são estruturas bem conhecidas e utilizadas por métodos computacionais, havendo assim uma vasta gama de algoritmos disponíveis, atrelados a como melhor manipulá-las (Deo 2017) (Kepner & Gilbert 2011).

2.2 K-nearest neighbors (KNN)

Na sua instância de aprendizado de máquina supervisionado, este método é utilizado tanto para a tarefa de classificação quanto para a tarefa de regressão (Peterson 2009). K-nearest Neighbors, também conhecido como KNN, durante o processo de treinamento de um modelo classificador utiliza como entrada uma sequência de exemplos rotulados com intuito de aprender uma função capaz de classificar novas entradas não rotuladas e desta fornecer como saída a classe prevista, com qualidade dentro de um nível aceitável

de acurácia (Murphy 1990).

Os dados rotulados fazem parte do conjunto de dados de treino, os quais são usados para realizar as previsões. Cada previsão de uma instância é feita através de uma busca por todo o conjunto de treino até encontrar um número K das instâncias mais semelhantes (vizinhos). Ou seja, é um classificador no qual o aprendizado se baseia na similaridade entre os dados, inferindo que se há similaridade, logo pode-se assumir que os dados estão provavelmente próximos uns dos outros (Cover & Hart 1967).

Para determinar essa semelhança, comumente é utilizada a distância euclidiana entre as amostras (Qian et al. 2004). Na Equação 3, pode-se observar a distância euclidiana entre as instâncias p_i e p_j , onde para $K = 1$ tem-se que n representa os n atributos que descrevem as instância.

$$d = \sqrt{\sum_{K=1}^n (p_{ik} - p_{jk})^2} \quad (3)$$

Então, basicamente, a partir dos resultados das distâncias do K vizinhos o método apresenta resultado da predição.

A escolha do valor para K varia para cada aplicação e é feito de forma empírica. Porém $K = 5$ é uma configuração padrão utilizada quando o método é aplicado pela primeira vez, já que valores muito baixos de K podem levar a *outliers* (pontos fora da curva) e valores altos tornam o algoritmo computacionalmente custoso.

2.3 Redes neurais Artificiais

A origem das redes neurais artificiais (RNA's) remonta, primeiramente, ao trabalho de (McCulloch & Pitts 1943) no qual foi apresentado o que seria o primeiro modelo de um neurônio artificial, seguido pela concepção do perceptron por (Rosenblatt 1958) e posteriormente do adaline por (Widrow & Hoff 1960).

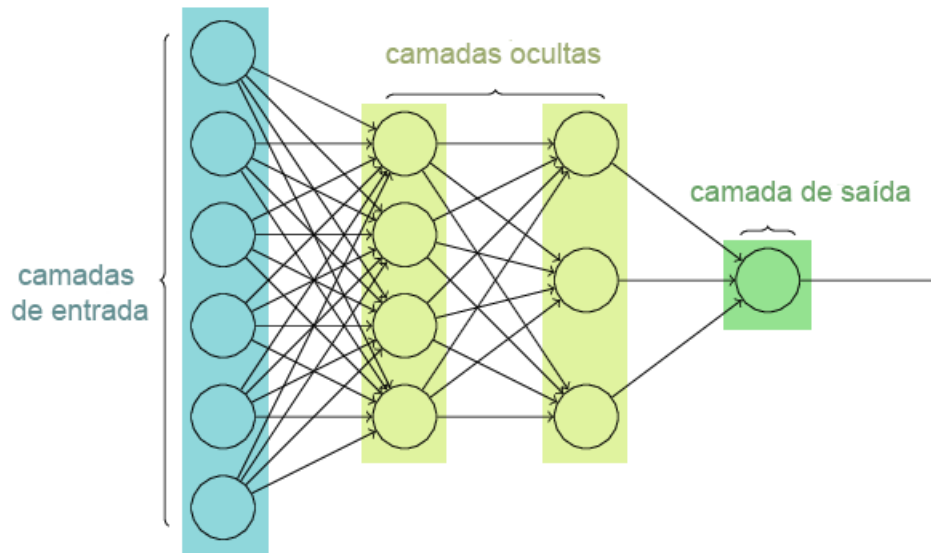
O processo de aprendizado no cérebro humano requer constantes ajustes na relação sináptica entre os neurônios (Stergiou & Siganos 2010), de forma análoga, os algoritmos de aprendizado empregados em redes neurais baseiam-se na forma a qual os neurônios se comunicam e se modificam de acordo com a experiência adquirida ao longo do aprendizado (Sejnowski 2020).

Diz-se que uma rede neural artificial é um modelo computacional biologicamente inspirado, composta por neurônios artificiais, de forma que o processamento de um único neurônio seja a combinação linear das entradas com os seus respectivos pesos, seguido pela combinação linear desses dados através da aplicação de uma determinada função matemática aos dados (função de ativação) (Krogh 2008). O ajuste destes pesos é realizado por um processo chamado treinamento ou aprendizado. Ou seja, a rede neural artificial é treinada a partir de um conjunto de exemplos específicos para o treino, de forma que

tenham o conhecimento sobre o problema em consideração (Binoti et al. 2014). Uma vez treinada, o algoritmo de aprendizagem faz a tentativa de generalização desses dados e aprende o conhecimento adquirido através dos parâmetros calculados para a rede, os quais são chamados de pesos (Rauber 2005).

Para realizar o processo de aprendizagem a arquitetura de RNA's básica se divide em três tipos de componentes chamados de camadas. Primeiramente tem-se a *camada de entrada* responsável por receber os dados de entrada, seguida pelas camadas internas (*hidden layers*) que realizam a extração dos padrões e por fim a camada de saída responsável por apresentar a saída resultante das outras camadas da rede. Existem diversas variações de arquiteturas de RNA's disponíveis, a Figura 5 demonstra uma das arquiteturas mais simples, uma rede neural artificial multi-camada com *feed forward propagation*.

Figura 5: Arquitetura de uma RNA.



Fonte: Elaborado pelo autor

Sendo os pesos sinápticos w , os valores das entradas x , a quantidade de entradas n e a quantidade de neurônios i , para cada sinapse tem-se:

$$s_i = \sum_{j=1}^n w_{ij}x_j. \quad (4)$$

Adicionando a constante chamada bias, responsável por centralizar a curva da função de ativação em um valor conveniente, tem-se:

$$s_i = \sum_{j=1}^n w_{ij}x_j + b_i. \quad (5)$$

Logo após, o resultado do somatório pela função de ativação, a qual tem como objetivo limitar a amplitude de saída do neurônio. A Equação 6 demonstra uma das possíveis

funções de ativação utilizadas em RNAs, neste caso a título de simplificação foi escolhida a função ReLu, definida como a função que retorna 0 caso a entrada seja negativa, e retorna o valor original de entrada nos casos de valores positivos.

$$y_i = \max(0, s_i). \quad (6)$$

Por fim, uma função *loss*, calculada na parte do treinamento chamada de propagação para frente (*forward propagation*), representa a diferença entre a saída obtida e a saída esperada. A matriz de pesos é ajustada pela etapa do processo de treinamento que ocorre na direção reversa (*backward propagation*). Por exemplo, utilizando o método de otimização *Stochastic gradient descent* (SGD), o processo de treinamento toma decisões de direcionamento na busca pelos pontos de mínimo associados à função *loss*.

A capacidade de autoaprendizagem das RNA's permite que as mesmas produzam melhores resultados à medida que uma maior quantidade de dados sejam disponibilizados. Assim abrindo caminho para a solução de problemas os quais seriam difíceis de serem resolvidos manualmente por seres humanos.

2.4 Graph Neural Networks (GNNs)

Embora os algoritmos usuais de redes neurais consigam captar padrões ocultos de maneira eficiente em diversos tipos de entrada de dados, principalmente quando se tem dados os quais podem ser representados em uma estrutura de blocos 2D (*grid-like*) como são os casos de problemas de classificação de imagem (Perez & Wang 2017) e segmentação semântica (Garcia-Garcia et al. 2017), há um desafio interessante quanto a complexidade imposta a esses algoritmos quando os mesmos utilizam-se de dados com estruturas em um domínio irregular (Wu et al. 2020)(Veličković et al. 2017).

Geralmente, em problemas complexos como o de redes sociais (Viégas & Donath 2004), redes de computadores (Internet) (Chaparro & Eberle 2015), estruturas de proteínas (Kannan & Vishveshwara 1999), *e-commerce* (Huang et al. 2004) e outros, devido a natureza irregular das suas estruturas, o conjunto de dados de entrada são melhor representados na forma de grafos e não através de um espaço métrico (Jiang et al. 2019).

Existem certas propriedades desses tipos de dados que os tornam inadequados para análises usuais envolvendo RNA's, já que neste ramo de aprendizagem de máquina considera-se a premissa de que as instâncias de um conjunto são independentes umas das outras. Quando se lida com grafos essa suposição não é mais válida pois cada instância, agora representada como um nó (vértice), está relacionada a outras instâncias por conexões de diversos tipos e pesos diferentes (Wu et al. 2020). Grafos também podem ter tamanhos variáveis com os quais algoritmos usuais não conseguem lidar sem que seja necessário empregar métodos como *padding* (quantidade de *pixels* adicionados a uma imagem quando ela está sendo processada por uma Rede Neural Convolutacional) e *resi-*

zing (redimensionar as imagens em tamanhos menores ou maiores enquanto treina a rede neural convolucional).

A fim de superar essas limitações, trabalhos recentes focam no projeto de novas arquiteturas de redes neurais que possam lidar com esses dados de relacionamentos de maneira eficiente. Chama-se essas arquiteturas de Redes Neurais em Grafos, ou *Graph Neural Networks* do inglês.

Introduzido pela primeira vez na literatura por (Gori et al. 2005), posteriormente aprofundado por (Scarselli et al. 2008) e (Gallicchio & Micheli 2010) *Graph Neural Networks* (GNNs) consiste em um processo iterativo o qual propaga os estados dos nodos do grafo até o equilíbrio, seguido por uma rede neural a qual é encarregada de produzir uma saída para cada nodo com base em seu respectivo estado (Xu et al. 2018).

Sabe-se que em um grafo cada nodo é naturalmente definido por suas características, seus nodos vizinhos e um rótulo atribuído a ele. Essencialmente, uma aplicação típica da arquitetura GCN é a tarefa de classificação de um nodo, ou aresta ou de um grafo ou sub-grafo.

É possível utilizar o aprendizado supervisionado em que grande parte do conjunto de dados disponíveis rotulados são empregados no treinamento da rede. Entretanto, as GNNs favorecem às aplicações em que é difícil ou até impossível obter dados rotulados em quantidade suficiente para garantir um modelo com acurácia razoável. As redes neurais em grafos tendem a utilizar o aprendizado semi-supervisionado para explorar dados não rotulados e com isto melhorar o desempenho do modelo treinado. Um grande número de algoritmos de aprendizado semi-supervisionado, otimiza de forma conjunta durante o treinamento duas funções objetivas: uma função *loss* “modo supervisionado” sobre os dados rotulados e outra função “modo não-supervisionado” sobre ambos os dados rotulados e não-rotulados. A função da Equação 2, já descrita na seção 1.3.2, é a soma ponderada da *loss* “supervisionada” e um termo de regularização Laplaciana em grafo, em que atribui uma penalidade grande quando nodos (instâncias) similares com grande w_{ij} são classificados como tendo rótulos diferentes $f(x_i) \neq f(x_j)$.

Nessa configuração, cada nodo v é caracterizado por um vetor de características (*features*) x_v , pelas características das arestas conectadas ao nodo $x_{co[v]}$, o *embedding* dos nodos adjacentes $h_{ne[v]}$, e pelas características dos nodos adjacentes $x_{ne[v]}$. Sendo f a função paramétrica que representa a dependência do nodo na sua vizinhança, chamada função de transferência local. O objetivo se dá por aprender o $h_v \in \mathbb{R}^s$, vetor de dimensão s responsável por armazenar a informação de vizinhança do nodo v para com todo os demais nodos.

$$h_v = f(x_v, x_{co[v]}, h_{ne[v]}, x_{ne[v]}). \quad (7)$$

Aplicando o teorema do ponto fixo de Banach (Khamsi & Kirk 2011) na Equação 7,

fica-se com:

$$H^{t+1} = F(H^t, X). \quad (8)$$

Onde H e X representam, respectivamente, a concatenação de todos os h e x , assim como H^t denota a t -ésima iteração de H .

A saída da rede neural em grafos o_v , a qual representa o rótulo (*label*) dos nodos, é adquirida passando o estado h_v , bem como a característica (*feature*) x_v , para uma função de saída g chamada de função local de saída.

$$o_v = g(h_v, x_v). \quad (9)$$

O próximo passo corresponde em como aprender os parâmetros de f e g . Uma vez que p corresponde ao número de nodos supervisionados, a perda pode ser descrita como:

$$loss = \sum_{i=1}^p (t_i - o_i). \quad (10)$$

Quanto ao aprendizado, o algoritmo utilizado é baseado na estratégia de gradiente descendente, na qual primeiramente os estados H_v^t são atualizados de forma iterativa na Equação 7 até um tempo T no qual $H(T) \approx H$ de forma que se chegue a Equação 8. Após, o gradiente de pesos W é calculado a partir da perda e, por fim, os pesos são atualizados de acordo com o valor do gradiente calculado.

2.4.1 Graph Convolutional Networks (GCNs)

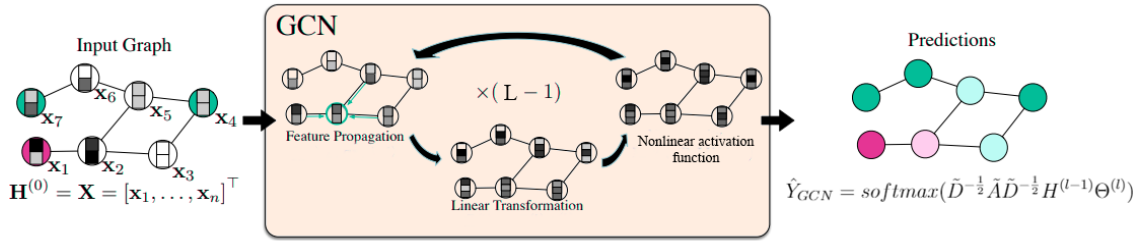
Em razão da aplicação de Redes Neurais Convolucionais (*Convolutional Neural Networks (CNNs)*) na resolução de problemas reais, como os de visão computacional (Gopalakrishnan et al. 2017) e processamento de linguagem natural (Gehring et al. 2016), ter apresentado um desempenho promissor, cada vez mais pesquisadores demonstram o interesse em conduzir trabalhos com o intuito de empregar operações convolucionais em dados com estruturas irregulares (Zhang, Tong, Xu & Maciejewski 2019).

A fim de suprir essa demanda, (Kipf & Welling 2016) propôs um modelo de abordagem espectral chamado de Rede Convolucional em Grafos (*Graph Convolutional Network GCN*), uma rede neural com várias camadas e filtros de convoluções espectrais de primeira ordem, seguidos por uma função de ativação não linear com a finalidade de aprender padrões em dados representados na forma de grafos (Wu et al. 2019).

A Figura 6 demonstra as transformações que a arquitetura GCN realiza em suas características ao longo das l – *camadas*, assim como a aplicação do classificador.

GCN é definido como abordagem espectral por trabalhar com a representação de *spectral networks* proposta por (Bruna et al. 2013), na qual a operação de convolução é definida pelo cálculo da decomposição da laplaciana de um grafo no domínio de *Fourier*.

Figura 6: Layout de uma GCN.



Fonte: Modificado de (Wu et al. 2019)

Essa operação de convolução pode ser definida como a multiplicação de um sinal escalar $x \in \mathbb{R}^N$ para cada nodo por um filtro $g_\theta = \text{diag}(\theta)$ parametrizado por $\theta \in \mathbb{R}^N$ no domínio de frequência (*Fourier*):

$$g_\theta \star x = U g_\theta U^\top x. \quad (11)$$

Sendo U a matriz normalizada de autovetores da matriz laplaciana L do grafo, onde:

$$L = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}} = U \Lambda U^\top. \quad (12)$$

I_N representa a matriz identidade, A a matriz de adjacência do grafo, D a matriz de *degree* (grau) do grafo e Λ a matriz diagonal dos autovetores.

De maneira semelhante a uma CNN convencional, a GCN opera de forma que após aprendida, a nova representação da característica x_i (*features*) é passada como entrada para um classificador linear. A representação de todos os nodos de entrada é generalizada pela matriz X , enquanto $H^{(l)}$ representa a saída para a camada de convolução l da rede. Logo cada camada da rede pode ser descrita como uma função não linear::

$$H^{(l+1)} = f(H^{(l)}, A). \quad (13)$$

A escolha da função f , bem como o modo como a mesma é parametrizada, são os fatores fundamentais os quais distinguem um modelo de rede do outro.

Para o desenvolvimento deste trabalho foi implementado o *multilayer* GCN proposto por (Kipf & Welling 2016), no qual em cada camada da rede a representação dos nodos é atualizada em três etapas, iniciando pela propagação da característica pela vizinhança, a transformação linear e então ativação através da função de ativação não linear (Wu et al. 2019). Este modelo possui a regra de propagação descrita na Equação 14.

$$H^{(l+1)} = \sigma \left(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l)} W^{(l)} \right). \quad (14)$$

No qual $\tilde{D} = \sum_j \tilde{A}_{ij}$, enquanto W^l representa a matriz de pesos de cada camada (treinada usando otimizador gradiente descendente), $\tilde{A} = A + I_n$ e σ representa a função

de ativação não linear, neste caso devido as suas características de esparsidade e probabilidade reduzida de gradiente de fuga, fora escolhida a função ReLU.

Para classificação de nodos, a última camada da GCN irá prever o rótulo usando uma camada *softmax*, retornando como saída para n nodos $\hat{Y} \in \mathbb{R}^{n \times C}$, o qual representa a probabilidade de o nodo pertencer a classe C .

Sendo Θ a matriz de filtro dos parâmetros, a predição de classe $\hat{Y}$ para um GCN de l -camadas pode ser descrita como:

$$\hat{Y}_{GCN} = \text{softmax} (\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l-1)} \Theta^{(l)}). \quad (15)$$

2.4.2 Cheb

Analisando de forma isolada, abordagens como a demonstrada na Equação 11 se tornam computacionalmente custosas para grafos em larga escala devido a multiplicação $\mathcal{O}(N^2)$ complexa no domínio de *Fourier* da matriz de autovetor U . A fim de evitar decomposições espectrais custosas é comum optar pela utilização de filtros polinomiais, tendo em vista que os mesmos possuem uma resposta de impulso finita (Tremblay et al. 2018) desta forma permitindo a implementação de abordagens com base, por exemplo, no polinômio de *Chebyshev*.

Em seu trabalho, (Hammond et al. 2011) propôs que o parâmetro $g_\theta(\Lambda)$ pode ser aproximado por uma expansão truncada em termos de polinômios de *Chebyshev* $T_k(x)$ para a K ordem, de forma que a Equação 11 seja reescrita como:

$$g_\theta \star x \approx \sum_{k=0}^K \theta_k T_k \tilde{L}x. \quad (16)$$

De forma que o processo de normalização do laplaciano para tornar a base ortogonal é descrito por:

$$\tilde{L} = \frac{2}{\lambda_{max}} L - I_N. \quad (17)$$

Sendo $\theta \in \mathbb{R}^K$ um vetor com os coeficientes de Chebyshev, os autovalores pertencentes ao intervalo $[-1, 1]$ e λ_{max} o maior autovalor de L (Nilsson 2020).

Pode-se dizer, que os polinômios de *Chebyshev* de ordem K são recursivamente definidos na base T como:

$$\begin{cases} T_0(x) = 1 \\ T_1(x) = x \\ T_k(x) = 2x T_{k-1}(x) - T_{k-2}(x) \end{cases} \quad (18)$$

Assumindo que o laplaciano seja uma matriz esparsa com $\mathcal{O}(1)$ elementos diferentes de zero em cada linha (Levie et al. 2018), devido a recursividade a filtragem usando esse

polinômio possui um custo de $\mathcal{O}(K|\mathcal{E}|)$ operação, sendo $\mathcal{E}$ o conjunto das arestas do grafo.

Utilizando do resultado relatado em (Hammond et al. 2011), (Defferrard et al. 2016) propôs a Cheb, uma rede neural convolucional a qual utiliza a parametrização de filtros pelo polinômio de *Chebyshev*, assim removendo a necessidade de computar os autovetores da Laplaciana do grafo, logo diminuindo o custo computacional.

Em questões práticas, pode-se dizer que Cheb e o modelo GCN proposto por (Kipf & Welling 2016) são semelhantes, tendo em vista que GCN nada mais é do que uma Cheb de primeira ordem, na qual se tem como parâmetros $\lambda_{max} = 2$ e $K = 1$. Ou seja, ao limitar o intervalo de uma convolução à semelhança de 1ª ordem, a métrica utilizada para determinar a similaridade entre 2 nodos acaba por ser baseada na vizinhança imediata do próprio nodo. Essa limitação torna a arquitetura linear em relação ao Laplaciano, sendo o objetivo principal reduzir o risco de sobreajuste (*overfitting*) em vizinhanças locais do grafo (Kipf & Welling 2016).

Apesar da eficiência computacional, filtros polinomiais, possuem uma série de desvantagens que impactam diretamente na eficiência do modelo quanto ao problema no qual está sendo empregado. Esses filtros são sensíveis a mudanças nas características dos nodos, na estrutura dos grafos ou quando a mesma não é conhecida (Isufi et al. 2016), como também não conseguem modelar mudanças bruscas na resposta de frequência (Tremblay et al. 2018).

Especialmente na Cheb, os autovalores são contraídos para a banda $[-1, 1]$ o que faz com que os agrupamentos (*clusters*) dos mesmos se tornem muito concentrados de forma que a resposta de frequência do filtro *Chebyshev* acaba por ser incapaz de separar os autovalores individuais nos *clusters*. Este tipo de comportamento é comumente encontrado em grafos que apresentam estruturas de comunidade como, por exemplo, os grafos de redes sociais (Levie et al. 2018). Nestas situações precisa-se empregar outra classe de filtros, que seja capaz de trabalhar com bandas de frequência mais estreitas, como é o caso dos filtros ARMA .

2.4.3 ARMA

ARMA, sigla para Modelos Autorregressivos de Médias Móveis, representam uma família de filtros na qual combina-se os modelos autoregressivos (AR) com os modelos de médias móveis (MA) afim de produzir um filtro com resposta para diversas variedades de frequências e que possa ser aplicada de forma universal em qualquer estrutura de grafo (Isufi et al. 2016) (Narang et al. 2013). Isso é possível pois os coeficientes dos filtros são projetados de forma independente da estrutura, logo sendo possível ser empregado em grafos os quais são variáveis com o tempo ou que até mesmo nos casos em a estrutura não é conhecida a priori.

Em (Bianchi et al. 2019) é proposto um novo modelo de GNN baseado nos filtros

ARMA. Aproveitando-se da não dependência explícita dos autovalores de L que estes filtros possuem, os filtros ARMAS tornam-se mais robustos a perturbações no grafo, desta forma, permitindo a generalização para grafos não vistos durante o treinamento como para grafos com topologias diferentes.

Utilizando-se das abordagens e definições de processamento de sinal em grafos, um filtro é um operador linear que modifica os componentes do sinal X , o qual representa as instâncias de todas as características, na base da matriz laplaciana L de forma que de acordo com a resposta, h age em cada autovalor λ . Na etapa de treinamento, os coeficientes dos filtros são otimizados afim de minimizar uma função de erro/perda (*loss function*) (Levie et al. 2019).

A resposta de um filtro ARMA de ordem K é dada como:

$$h_{ARMA_K}(\lambda) = \frac{\sum_{k=0}^K p_k \lambda^k}{1 + \sum_{k=0}^K q_k \lambda^k}. \quad (19)$$

No qual p_k representa a ordem do modelo auto-regressivo e q_k representa a ordem do modelo de média móvel. Podem-se reescrever a Equação 19 como:

$$\bar{X} = \frac{(\sum_{k=0}^K p_k L^k)}{1 - \sum_{k=0}^K q_k L^k}. \quad (20)$$

Na GNN proposta por (Bianchi et al. 2019) utilizou-se a aplicação recursiva já adotada em processamento de sinal em grafos, na qual para um filtro $ARMA_1$ tem-se a resposta demonstrada na Equação 21.

$$\bar{X}^{(t+1)} = aM\bar{X}^{(t)} + bX. \quad (21)$$

Na qual a e b são os coeficientes do filtro e M corresponde a:

$$M = \frac{1}{2}(\lambda_{max} - \lambda_{min})I - L. \quad (22)$$

A resposta à frequência do filtro $ARMA_1$ é definida como:

$$h_{ARMA_1}(\mu_m) = \frac{r}{\mu_m - p}. \quad (23)$$

Com $r = -\frac{b}{a}$ e $p = \frac{1}{a}$.

A variável μ_m representa o m autovalor de M enquanto λ_m representa o m autovalor de L . De forma que:

$$\mu_m = (\lambda_{max} - \lambda_{min})/2 - \lambda_m. \quad (24)$$

A recursividade da Equação 21 para $ARMA_1$ é então atualizada com camadas de GCS (*Graph Convolutional Skip*), obtendo:

$$\bar{X}^{(t+1)} = \sigma(\tilde{L} \cdot \bar{X}^{(t)} W^{(t)} + X V^{(t)}). \quad (25)$$

Sendo $W^{(t)}$ e $V^{(t)}$ parâmetros de treino e a matriz laplaciana $\tilde{L} = I - L$ obtida ao inferir $\lambda_{max} = 2$ e $\lambda_{min} = 0$.

Por fim tem-se que a saída das camadas é dado por:

$$\bar{X} = \frac{1}{K} \sum_{k=1}^K \bar{X}_k^{(T)}. \quad (26)$$

2.4.4 GAT

Graph Attention Network (GAT) define uma arquitetura de rede neural a qual incorpora mecanismos de atenção de maneira compartilhada a todas as arestas do grafo na sua etapa de propagação, seguido por uma estratégia de “auto-atenção” (*self-attention*) (Zhang, James & Liu 2019) com a finalidade de computar os estados ocultos de cada nodo a partir de seus vizinhos (Zhou et al. 2018).

Tem como objetivo principal suprir as limitações dos métodos com base em convolução propostos anteriormente na literatura, como por exemplo o modelo de GCN proposto por (Kipf & Welling 2016) cujo qual possui dependência direta com a estrutura do grafo, o que prejudica a generalização do mesmo.

O modelo GAT fora proposto pela primeira vez por (Veličković et al. 2017). Em seu estudo o modelo foi construído empilhando camadas de atenção (*attention layers*), responsáveis por computar os coeficientes de atenção. Através desse empilhamento o modelo permite atribuir implicitamente importâncias diferentes aos nodos do grafo (Kosaraju et al. 2019), ou seja permite especificar diferentes pesos para os diversos nodos em uma vizinhança sem que haja a necessidade de realizar operações matriciais custosas ou de ter o conhecimento prévio quanto a estrutura global do grafo.

A entrada para as camadas de atenção do modelo GAT são dadas pelo conjunto de características (*features*) do nodo $h = \{\vec{h}_1, \vec{h}_2, \dots, \vec{h}_N\}$, $\vec{h}_i \in \mathbb{R}^F$, com F sendo o número de características em cada nodo e N o número de nodos no grafo. A camada de atenção irá produzir um novo conjunto de características h' como saída.

Uma transformação linear, parametrizada pela matriz de pesos $W \in \mathbb{R}^{F' \times F}$, é aplicada a cada nodo, logo após aplica-se *self attention* nos nodos de forma que o mecanismo de atenção compartilhado $a : \mathbb{R}^{F'} \times \mathbb{R}^{F'} \rightarrow \mathbb{R}$ computa os coeficientes de atenção, obtendo então a Equação 27, a qual indica a importância das características do nodo j para o nodo i .

$$e_{ij} = a(W\vec{h}_i, W\vec{h}_j). \quad (27)$$

Por sua vez, a estrutura do grafo é inserida no mecanismo de atenção por um tipo de

“mascaramento” (*masked attention*), a qual computa apenas os e_{ij} do nodo $j \in \mathcal{N}_i$, sendo $\mathcal{N}_i$ a vizinhança do nodo i no grafo. Logo o modelo permite que cada nodo atenda a todos os outros nodos. Os coeficientes de atenção são então normalizados para todas as opções de j através da função *softmax*.

$$\alpha_{ij} = \text{softmax}_j(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}_i} \exp(e_{ik})}. \quad (28)$$

No modelo proposto por (Veličković et al. 2017) ao aplicar a função LeakyReLU (com entrada negativa de $\alpha = 0.2$) ao mecanismo de atenção a , o qual no trabalho proposto representa uma rede neural de camada única e do tipo *feedforward* parametrizada pelo vetor de pesos $\vec{a} \in \mathbb{R}^{2F'}$, obtende-se os coeficientes de atenção parametrizados α_{ij} do nodo j para o i , como mostra a Equação 29.

$$\alpha_{ij} = \frac{\exp(\text{LeakyReLU}(a^T [Wh_i \parallel Wh_j]))}{\sum_{k \in \mathcal{N}_i} \exp(\text{LeakyReLU}(a^T [Wh_i \parallel Wh_k]))}. \quad (29)$$

Com esses coeficientes computados as características para cada nodo são obtidas após aplicar-se uma função não linear σ , obtendo como resultado a Equação 30.

$$\vec{h}'_i = \sigma \left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} W \vec{h}_j \right). \quad (30)$$

A fim de estabilizar o processo de aprendizagem a camada de atenção da rede GAT proposta utiliza um esquema denominado em inglês de *multi-head attention* (Vaswani et al. 2017). Basicamente, K mecanismos de atenção independentes executam a transformação proposta na Equação 30, concatenando as *features* de forma a obter como saída uma nova representação destas *features* demonstrada na Equação 31. No caso em que o esquema *multi-head* é aplicado na camada final da rede, ao invés de efetuar a concatenação é feito o calculo da média resultando na Equação 32.

$$\vec{h}'_i = \parallel_{k=1}^K \sigma \left(\sum_{j \in \mathcal{N}_i} \alpha_{ij}^k W^k \vec{h}_j \right). \quad (31)$$

$$\vec{h}'_i = \sigma \left(\frac{1}{K} \sum_{k=1}^K \sum_{j \in \mathcal{N}_i} \alpha_{ij}^k W^k \vec{h}_j \right). \quad (32)$$

Onde α_{ij}^k são os coeficientes de atenção normalizados computados pelo k mecanismo de atenção e $\parallel$ representa a concatenação.

Uma das principais diferenças entre os modelos GAT e GCN se dá pela forma como representação das características da vizinhança são adquiridas e coletadas (Zhang, James & Liu 2019). GAT propõe substituir a operação de convolução empregada em GCN pelo seu mecanismo de atenção.

Em questões computacionais se comparado com métodos anteriores, a proposta

GAT é considerada eficiente. Isto devido ao cálculo dos pares nodo-vizinho ser paralelizável (Zhou et al. 2018) e não se utilizar de operações matriciais custosas. Contudo a paralelização sobre as arestas pode levar a redundância, uma vez que é comum vizinhanças se sobreporem em grafos (Veličković et al. 2017). Já a complexidade por camada do GAT com K heads computando F' features é expressa por $\mathcal{O}(K | \mathcal{V} | FF' + K | \mathcal{E} | F')$, onde F representa o número de features de entrada, $|\mathcal{V}|$ o número de nodos e $|\mathcal{E}|$ o número de arestas do grafo (Guo et al. n.d.).

2.4.5 Resumindo os modelos de GNN

Sumarizando, entende-se que o Cheb é um modelo de rede neural convolucional que se baseia na teoria espectral de grafos, de forma que realiza a parametrização de filtros convolucionais localizados pelo polinômio de Chebyshev. Por sua vez, o GCN é um modelo baseado em grafos o qual se utiliza desses filtros de convoluções espectrais. Desta forma, o GCN pode ser descrito como um modelo Cheb o qual possui o intervalo de convolução limitado à semelhanças de 1ª ordem, ou seja, a similaridade entre dois nodos é dada pelas informações correspondentes à vizinhança imediata dos mesmos.

Já o modelo ARMA combina modelos autoregressivos com modelos de médias móveis para produzir uma família de filtros capazes de entregar respostas de diversas frequências de modo que se torna possível a sua utilização de maneira universal em estruturas de grafos diversas. Enquanto o modelo GAT faz uso de uma abordagem baseada em mecanismos de atenção, assumindo valores de importância diferentes para nodos que fazem parte da mesma vizinhança.

Em questão de aplicações, o modelo GCN, apesar de ser uma variação do Cheb, é mais utilizado na literatura tendo em vista que pela limitação de vizinhança em sua implementação o mesmo tende a apresentar menos chance de cair em uma situação de sobre ajuste como também acaba por ser computacionalmente mais eficiente se comparado ao Cheb.

Entretanto, ambos modelos (GCN e Cheb) utilizam filtros polinomiais, os quais não são indicados para certas aplicações, já que os mesmos são sensíveis a mudanças nas características do grafo e não conseguem trabalhar com bandas de frequência mais estreitas. Para essas situações o modelo ARMA apresenta uma melhor escolha já que o mesmo consegue modelar mudanças bruscas em relação a resposta em frequência. Por sua vez, o modelo GAT também tem como objetivo suprir as limitações advindas de métodos que possuem dependência direta com a estrutura do grafo. Através de sua abordagem de atenção, além da generalização o mesmo também promete ser mais eficiente computacionalmente, trazendo como *trade-off* uma possível redundância devido ao processo de paralelização das operações matriciais dos pares nodo-vizinho.

Dentre os quatro modelos apresentados, uma característica comum a todos refere-se à possibilidade de serem utilizados em uma abordagem que emprega uma quantidade re-

duzida de dados durante a etapa de treinamento da rede (semi-supervisionado). Situação essa que condiz com a encontrada em aplicações que lidam com a classificação de documentos como foram descritas neste trabalho anteriormente.

3 REVISÃO BIBLIOGRÁFICA

Classificação de documentos é uma área de intensa pesquisa (Kalra & Lal 2016), principalmente quando há interconexão entre as fontes de dados (Kettouch 2017).

Ao utilizar abordagens que possibilitem manter a estrutura de conexões diretas e indiretas entre os dados, (Eto 2019) aliadas a técnicas que permitam realizar o manuseio de um grande volume de dados de forma eficiente, tem-se uma solução a qual possibilita uma melhor extração, recuperação e classificação das informações resultantes. Esse é o caso das abordagens que utilizam algoritmos de inteligência artificial baseados na estrutura de grafos (Strohman et al. 2007).

A representação de informações na forma de grafos possibilita empregar este tipo de estrutura em soluções de distintos propósitos e áreas de conhecimento, como são apresentadas em trabalhos como o realizado por (Zhao et al. 2019). Neste caso, um modelo baseado na teoria dos grafos é proposto com o desenvolvimento de uma GCN temporal (T-GCN) para modelar a rede de estradas urbanas. Este é um problema complexo devido às restrições da estrutura topológica da rede de estradas e à lei da mudança dinâmica ao longo do tempo.

Em (Li et al. 2019) é apresentado um modelo que combina LSTM (*long short term memory*) e redes neurais convolucionais para realizar a predição do “tempo de vida útil restante” (*Remaining useful life* - RLU) de máquinas. A validação do modelo foi feita utilizando o conjunto de dados de simulação de degradação do motor turbofan fornecido pela NASA demonstrando que, comparado com o uso isolado de LSTM ou CNN, a previsão do modelo proposto foi superior.

Já (Wang et al. 2018) desenvolveu o *Graph Reasoning Model* (GRM) com o intuito de incorporar o que pode-se considerar como “conhecimento óbvio” e a semântica contextual de objetos em uma cena para reconhecer o relacionamento social que acontece naquele ambiente. O GRM basicamente consiste em um modelo que propaga a informação do nodo pelo grafo afim de explorar as interações entre as pessoas de interesse e os objetos da cena. Em seguida um grafo de atenção é usado para medir a importância de cada nodo de forma adaptativa, assim decidindo quais objetos são relevantes para o reconhecimento da cena. Por fim foram realizados experimentos em dois datasets de *benchmark* o *People*

in Social Context (PISC) e People in Photo Album Relation (PIPA-Relation), comparando seus respectivos resultados com os modelos existentes no estado da arte.

Em (Teney et al. 2017) foi desenvolvida uma rede neural profunda para processar uma estrutura, na forma de grafo, referente a uma cena, com o intuito de inferir respostas a um problema de VQA (*visual questions answering*). O modelo proposto foi treinado com *datasets* de “cenas abstratas”, tendo demonstrando resultados promissores. A acurácia do estado da arte foi elevada de 71,2% para 74,4% na configuração de múltiplas escolhas. Já na versão “balanceada” do *dataset* aumentou de 34.7% para 39.1%.

Por sua vez, (Ahmed et al. 2016) propõe um *framework* supervisionado para pre-dizer relacionamentos entre informações de usuários na rede social Twitter, utilizando a representação da rede na forma de grafos. Neste caso, para as arestas foram definidos dois tipos de link, os explícitos chamados de *friend edges* (dois nodos que visivelmente possuem uma conexão de amizade entre eles) e os implícitos chamados de *action edges* (quando dois nodos possuem uma conexão não explícita, uma ação como uma menção, uma resposta a um comentário ou um *retweet*). As características que usam as *friends edges* são classificadas em características de conectividade ou características de comunidade, enquanto as que usam as *action features* são chamadas de características de interação ou características de confiança. Ahmed et al. (2016) utilizou algoritmos de aprendizado de máquina supervisionado, aplicados a um *dataset* com 2974 usuários do *twitter*, onde 13 foram randomicamente selecionados e a sua lista de amizades percorrida. Considerando apenas a relação de seguidor, foram obtidos 88458 relacionamentos de amizades entre esses usuários, com uma média de 30 amigos por usuário.

Já em (Szekely et al. 2015) foi apresentada uma solução para um problema global que viola diretamente os direitos humanos. Para tal, foi utilizado um DIG, sigla para (*domain insight graphs*), num trabalho conjunto envolvendo quinze organizações, com o objetivo de desenvolver um sistema que faça uso de abordagens baseadas em grafos para auxiliar no combate ao tráfico de pessoas, através da identificação da rede de atores que se encontra por trás de tais atos.

Diversas aplicações apropriam-se da facilidade que a representação visual de grafos oferece para um melhor entendimento do problema, incorporando-a como ponto chave de suas soluções. Em (Lee & On 2015) podem-se observar isso. Nele, com intuito de conscientizar a população quanto ao posicionamento público dos membros pertencentes a ONU, é proposto um sistema que utiliza algoritmos de mineração para criar um “grafo social” com as medidas de similaridade encontradas nas informações públicas de votos de cada membro da organização. Esse sistema é encapsulado para funcionar em um navegador com uma interface intuitiva, na qual o usuário pode visualizar e analisar de modo visual os padrões de votos encontrados.

Além disso na área de fármacos, destacam-se os modelos para busca automática por possíveis interações entre químicos e proteínas (Crichton et al. 2018). Problemática essa

conhecida como *drug target interactions* (DTI). Esses tipos de trabalhos possuem um papel fundamental no reposicionamento de fármacos. Através dos mesmos podem-se prever as interações que certos componentes químicos terão sobre proteínas, dessa forma auxiliando os pesquisadores a encontrar potenciais usos para os fármacos que estão sendo desenvolvidos (Ashburn & Thor 2004). Em uma rede DTI usualmente são utilizados métodos como RBM (*Restricted Boltzmann Machines*) (Wang & Zeng 2013) e SVM (*Support Vector Machines*) (Ding et al. 2013). No entanto, aplicações com *Graph Neural Network* (Torng & Altman 2019) (Lim et al. 2019) (Zhao et al. 2020) tem recebido interesse nos últimos três anos.

Verifica-se principalmente que nos casos em que as informações encontram-se puramente na forma de documentos, escaneados e digitalizados os quais são interconectados, que citam e são citados por outros, ou que de alguma forma possuam algum relacionamento entre si, essas relações acabam por estabelecerem pontes de conexão as quais geram um conhecimento compartilhado que apenas pode ser extraído levando essa conectividade em consideração, como pode-se observar em Wang et al. (2016). Para situações em que os dados são disponibilizados desta maneira, nos últimos anos pesquisadores tendem a fazer uso de métodos que utilizam a teoria de grafos, seja ao nível representação hierárquica dos dados ou no processo de classificação baseada nos relacionamentos entre documentos (Ostendorff et al. 2019) (Rinaldi et al. 2021).

Em Mujtaba et al. (2018) foi desenvolvido um sistema de representação totalmente automatizado de documentos baseado em grafos, chamado CGDR, para classificação dos relatórios de autópsia forense. O estudo utilizou 1500 arquivos para classificar 16 tipos de relatórios representando as informações como um conjunto de nodos e usando classificação hierárquica de texto. As características conceituais, baseadas em termos e nomenclatura da medicina, foram extraídas e então representadas por meio de grafos, ao invés da usual representação por *bag-of-words*. Arranjadas no formato, as informações serviram como recursos para treinar um classificador de texto. No entanto, o modelo proposto não é a solução ideal quando é necessário classificar todos os tipos existentes de relatórios. Isso se dá pois eles ainda dependem de especialistas para identificar as características úteis para cada tipo de autópsia disponível. Não há nenhum algoritmo de aprendizado profundo ou abordagem adaptativa implementada para contornar essa situação e melhorar o desempenho do CGDR. As informações devem ser entregues no formato ideal para que o sistema funcione adequadamente.

Por outro lado, em Nugroho et al. (2019) é proposta uma abordagem híbrida, usando sete tipos de *kernels* para grafos em conjunto com o algoritmo SVM a para a classificação automática de patentes. Os três *datasets* utilizados foram divididos em conjunto de treino e validação, correspondendo a 80% e 20% dos dados, respectivamente. Métricas de acurácia, precisão, *recall* e F1 foram utilizadas a fim de avaliar o desempenho dos classificadores.

Enquanto Zhang et al. (2020) propôs um método de classificação chamado TextING o qual treina uma GNN com intuito de representar as relações entre as palavras e um *Gated Graph Neural Networks* para propagar as informações dos nodos para seus vizinhos e em seguida agregá-las em um *embedding* de um documento. O método foi testado em quatro *datasets* e comparados com três tipos diferentes de métodos.

É necessário observar sobretudo, que as abordagens mencionadas acabam por necessitarem de uma grande quantidade de dados rotulados para seu processo de classificação, fator preocupante tendo em vista que em situações reais este tipos de dados são escassos.

Em virtude disso, Yao et al. (2019) propôs um método chamado de Text Graph Convolutional Networks (Text GCN), o qual tem a finalidade de realizar a classificação de texto em casos os quais se tem disponível um número limitado de amostras rotuladas. Foram utilizados ao total cinco *datasets* para o estudo comparativo sendo eles 20NG, R8, R52, Ohsumed e MR, obtendo resultados melhores que métodos do estado da arte nos quatro primeiros *datasets*. Também fazendo uso de GCN, Wang et al. (2020) integra o método com uma abordagem de Hierarchical Topic Graph (HTG) para realizar classificação de documentos. Foram utilizados 11 métodos de base e cinco *datasets* para comparações, obtendo o melhor resultado na maioria dos *datasets*.

Vale-se destacar a existência de métodos baseados em grafos com o nome semelhante a GCN, porém que constituem outras abordagens com objetivos diferentes, como em Sahu et al. (2019) que utiliza Graph Convolutional Neural Network (GCNN) aplicado ao processo de extração das inter-relações entre palavras.

Tal qual em Yao et al. (2019), os modelos propostos por Kipf & Welling (2016), Defferrard et al. (2016), Veličković et al. (2017) e Bianchi et al. (2019) também utilizam abordagens de aprendizado semi-supervisionadas.

4 METODOLOGIA

Este trabalho propôs-se a realizar um estudo comparativo sobre o desempenho de métodos de redes neurais em grafos (GNNs). Para tanto, fez-se uma comparação de arquiteturas GNNs selecionadas na literatura e treinadas a partir de *datasets* públicos de citações de artigos científicos. Por meio deste estudo comparativo, também foi criado um *perfil de calibragem* de parâmetros para os modelos validados. Esta metodologia de comparação (ou *benchmarking* em inglês) utiliza um fluxo de trabalho centrado em modelos (*model-centric*), também chamado de validação com *black-box*. Neste caso, os passos seguem um fluxo padronizado em que se utiliza dados que forem possíveis de coletar (ex. em *datasets* públicos) e desenvolve-se um modelo com desempenho suficiente para lidar com o ruído nos dados.

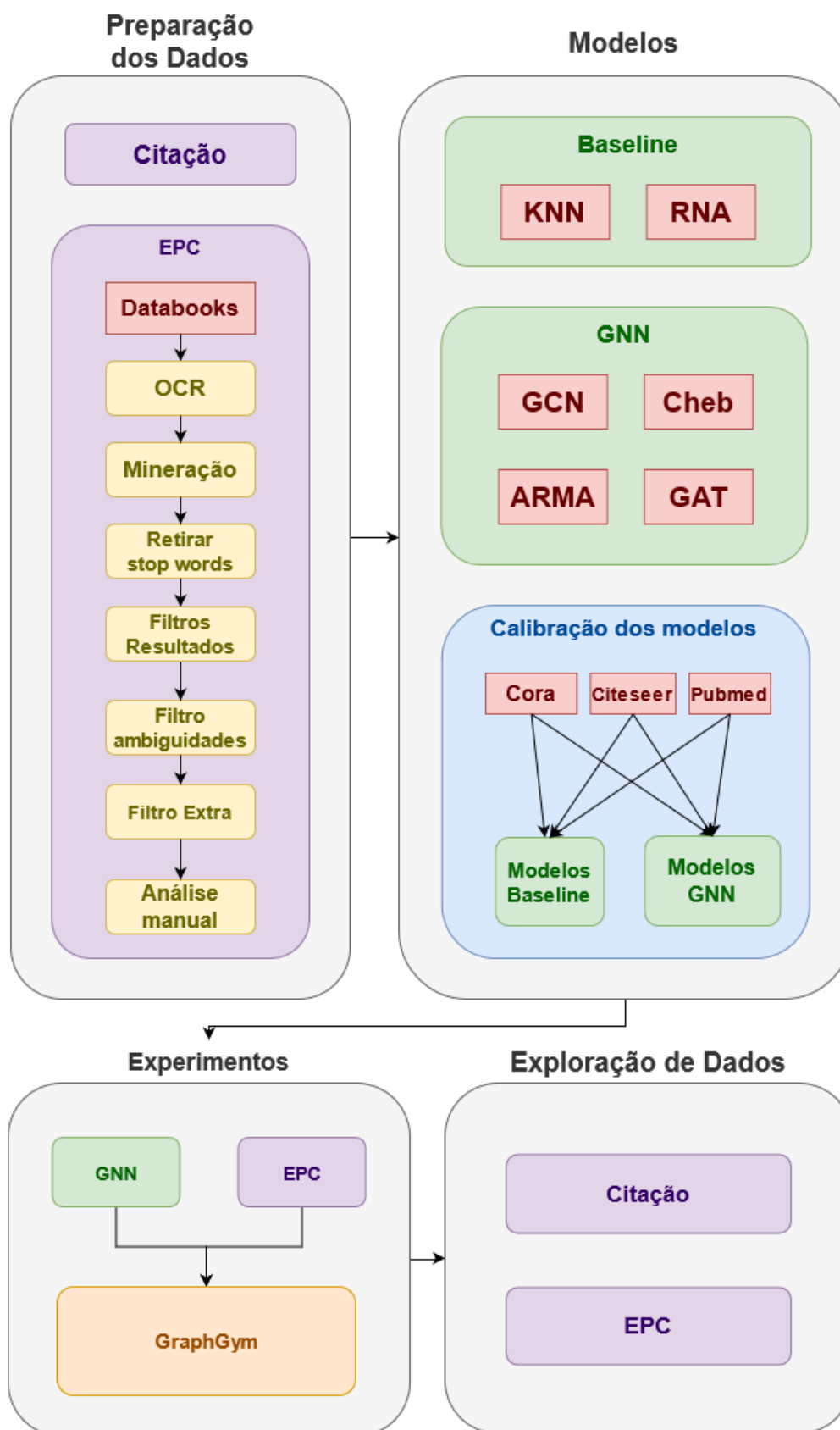
A grande vantagem desta metodologia está no fato de que sua padronização pode garantir um fluxo de desenvolvimento sistematizado para novas aplicações. Andrew Ng (2021) define o ciclo de um projeto de aprendizado de máquina contendo as etapas:

- a. Definição do escopo;
- b. Coleta dos dados;
- c. Treinamento do modelo;
- d. Implantação no ambiente da aplicação;

No caso da metodologia centrada no modelo, fixa-se o dado na forma dos *datasets* e iterativamente se faz melhorias no código/modelo. A metodologia utilizada durante a dissertação foi centrada no modelo e está dividida em quatro partes, como mostra a Figura 7.

A primeira etapa corresponde à preparação dos dados, a qual é dividida em duas partes, Citação e EPC. Na parte Citação foi realizado um processo de entendimento e compreensão das características e componentes dos *datasets* para *benchmarking* escolhidos. Na parte EPC foi então realizada a construção do dataset relativo ao Projeto ATT mencionado na seção 1.1. A partir de uma série de sub-passos desenvolveu-se um *dataset* próprio, contendo as informações sobre os *databooks* EPC.

Figura 7: Metodologia geral do trabalho proposto.



Fonte: Elaborado pelo autor

Após, tem-se a etapa referente aos Modelos. Nesta parte da metodologia foram escolhidos dois modelos de *baseline* e quatro modelos de redes neurais em grafos (GNNs). Em um primeiro momento foi feita uma análise comparativa quanto aos resultados dos modelos *baseline* e de GNN, a fim de validar a abordagem de grafos para o problema de classificação de documentos. Após, os quatro modelos de redes neurais em grafos foram submetidos a uma etapa chamada de Calibração, com o objetivo de realizar uma análise dos métodos e seus parâmetros de ajustes (otimização de hiperparâmetros). Este perfil de calibração é importante para desenvolver o trabalho de treinamento e validação dos modelos GNNs na problemática de classificação de documentos de EPC.

A etapa de Experimentos EPC visa:

- a. Estabelecer os relacionamentos entre documentos de EPC através do *dataset* próprio do Projeto ATT;
- b. Validar quantitativamente um classificador para este tipo de documentos;

O método GNN que apresentou o melhor desempenho na etapa de calibração foi utilizado para treinar o classificador. Com o objetivo de validar os resultados obtidos nos primeiros experimentos, foram feitos novos experimentos com o modelo e o *dataset*, desta vez aplicando-os na plataforma de Stanford chamada GraphGym (You et al. 2020).

A partir dos resultados obtidos durante este processo foi necessário retornar aos dados para explorá-los de forma mais aprofundada, com a finalidade de compreender e justificar os resultados. Esta análise centrada nos dados (*data-centric*) reconhece que a consistência dos dados é talvez o elemento mais importante de um projeto de aprendizado de máquina. Para tanto, é necessário utilizar ferramentas de visualização e “corte” dos dados (expressão em inglês *slice & dice*) para melhorar a qualidade dos mesmos.

A ideia central é fixar o código e iterativamente melhorar o dado (Andrew Ng 2021). Este processo permitirá que múltiplos modelos obtenham desempenhos satisfatórios.

A etapa de preparação de dados será explorada em detalhes no Capítulo 5 e, por sua vez, a etapa referente ao modelo será explorada no Capítulo 6.

5 PREPARAÇÃO DOS DADOS

Dois tipos de *datasets* foram utilizados neste trabalho. O primeiro é composto por dados de *benchmarking* comumente empregados na literatura e o segundo é um *dataset* próprio desenvolvido a partir dos *databooks* utilizados no Projeto ATT. Este capítulo traz informações mais detalhadas acerca destes *datasets*.

5.1 Datasets de Citação

Foram utilizados três *datasets* com dados reais de redes de citações usualmente utilizados para *benchmarkings* de classificação de nodos - Cora (Lu & Getoor 2003), Citeseer (Sen et al. 2008) e Pubmed (Namata et al. 2012). Em todos eles os nodos representam os documentos (artigos científicos), as arestas representam a citação de um artigo ao outro e rótulo (*label*) representa o assunto ou área de conhecimento do artigo.

Os *datasets* passaram pelo processo de *stemming* e remoção de *stop word* em casos em que a frequência de ocorrência é menor do que dez. As palavras resultantes correspondem as características (*features*) de cada nodo, ou seja, as palavras chaves que descrevem o conteúdo do artigo. Estas palavras foram representadas utilizando o método de codificação *one-hot* binário (Yang et al. 2016). Já o *label rate* representa o número de nodos rotulados utilizados para treinamento dividido pelo total de nodos em cada *dataset*.

O conjunto de dados Cora contém 2708 artigos sobre aprendizado de máquina divididos em sete classes, a saber: *Case Based*, *Genetic Algorithms*, *Neural Networks*, *Probabilistic Methods*, *reinforcement Learning*, *Rule Learning* e *Theory*.

Por outro lado, o conjunto de dados Citeseer possui 3327 artigos científicos divididos em seis classes sendo elas: *Agents*, *AI*, *DB*, *IR*, *ML* e *HCI*.

Enquanto o Pubmed contém 19717 artigos sobre diabetes divididos em três classes: *Mellitus Experimental*, *Diabetes Mellitus Type 1* e *Diabetes Mellitus Type 2*.

As informações quanto a quantidade de nodos, arestas e características de cada *dataset* foram resumidas na Tabela 1. A quantidade de instâncias em cada classe e a quantidade de *features* assinaladas como presentes em cada rótulo, nos três *datasets*, são apresentados nas Tabelas 2 e 3, respectivamente.

Tabela 1: Informações sobre os databases utilizados.

Dataset	Nodos	Arestas	Features	Classes	Label Rate
Cora	2708	5278	1433	7	0.052
PubMed	19717	44324	500	3	0.003
CiteSeer	3327	4552	3703	6	0.036

Tabela 2: Quantidades de instâncias em cada classe.

Modelo	Label	Quantidade	(%)
Cora	Neural_Networks	818	0.302068
	Genetic_Algorithms	818	0.302068
	Probabilistic_Methods	426	0.157312
	Genetic_Algorithms	418	0.154357
	Theory	351	0.129616
	Case_Based	298	0.110044
	Reinforcement_Learning	217	0.080133
	Rule_Learning	180	0.066470
Citeseer	DB	701	0.211655
	IR	668	0.201691
	Agents	596	0.179952
	ML	590	0.178140
	HCI	508	0.153382
	AI	249	0.075181
Pubmed	0	4103	0.208095
	1	7739	0.392504
	2	7874	0.399402

Percebe-se na Tabela 2 um desbalanço no número de instâncias em cada classe, situação típica em projetos de aprendizado de máquina. Esta situação deve ser contextualizada dentro do problema de classificação de documentos por meio do aprendizado semi-supervisionado.

Tabela 3: Quantidades de features em cada label.

Modelo	Label	Quantidade	(%)
Cora	Neural_Networks	1286	0.896167
	Probabilistic_Methods	1048	0.730314
	Genetic_Algorithms	1030	0.717770
	Theory	987	0.687805
	Case_Based	896	0.624390
	Reinforcement_Learning	829	0.577700
	Rule_Learning	752	0.524042
Citeseer	DB	2698	0.728205
	IR	2691	0.726316
	Agents	2504	0.675843
	ML	2571	0.693927
	HCI	2357	0.636167
	AI	1979	0.534143
Pubmed	0	496	0.992
	1	499	0.998
	2	500	1.000

5.2 Dataset EPC

O *Dataset EPC* foi desenvolvido utilizando *databooks EPC* de peças compradas pela empresa parceira do Projeto ATT. Foram utilizados 83 arquivos no formato pdf, os quais contabilizam um total de 240 *databooks* que pertencem ao mesmo fornecedor primário. Ou seja, aquele que está em contato direto com a empresa e forneceu a peça já pronta para o uso.

Como durante uma obra de engenharia são utilizados um número elevado de peças, para o dataset desenvolvido foram utilizados apenas *databooks* de parafusos estoijos, porcas sextavadas e barras roscadas de um fornecedor específico, o qual possui uma boa reputação, devido a sua organização em processos, segundo a empresa parceira.

Em cada *databook* encontram-se três tipos diferentes de organizações. Como já mencionado, tem-se o fornecedor primário, aquele que está em contato direto e fornece a peça pronta. Após, na sequência hierárquica, tem-se o fornecedor secundário o qual fornece a matéria-prima para a manufatura da peça. Por último tem-se os laboratórios responsáveis pelos ensaios e testes de conformidade exigida pelos conjuntos de normativas em ambas, a matéria prima e a peça final manufaturada.

O processo de desenvolvimento do dataset foi realizado de forma semelhante ao utilizado para o *dataset* Cora, com a inclusão de alguns novos passos devido à natureza e contexto das informações. Em questões de estrutura de armazenamento das informações, o *dataset* foi desenvolvido de modo que cada página represente um nodo, onde cada nodo

possui sua identificação única, suas características (*features*) e um rótulo de classe associado.

5.2.1 Features

Os *databooks* foram digitalizados através de *scanner*. Os arquivos resultantes do escaneamento passaram por processo de conversão para obtenção de textos na forma de caracteres, através da ferramenta *tesseract OCR*. Essa ferramenta é responsável por transformar o conteúdo dos documentos em formato de imagem para um PDF com uma camada de texto. Desta forma, o documento final pode ser selecionável e permite buscas no seu conteúdo. Durante o OCR também é gerado um arquivo chamado HOCR, o qual descreve os elementos encontrados na imagem em um formato XML, respeitando uma hierarquia de texto (páginas, linhas, palavras). Todas as palavras foram convertidas para caixa baixa (minúsculo) e qualquer tipo de acentuação também foi removida.

De posse desse arquivo, realizou-se um processo de identificação das palavras presentes no mesmo, o que resultou um total de 3,810,302 palavras, em Inglês ou Português, em todos os 240 *databooks*.

No próximo passo, foram retiradas as palavras vazias, chamadas de *stop words*. Aquelas que são utilizadas apenas como complemento de uma sentença e irrelevantes para o processamento do texto, como por exemplo os pronomes “de, para, como”, o artigo “*the*”, a preposição “*with*”, etc. Após essa filtragem obteve-se um total de 3,265,352 palavras.

Foram então aplicados pequenos filtros para retirar palavras que representam os resultados de um teste, como por exemplo uma análise química. Os mesmos não servem como *features* para um documento, visto que mudam conforme o estado do material ou peça analisada. Assim sendo, foram retirados todos os caracteres numéricos presentes dentre o conjunto de palavras restantes, bem como aquelas com número de caracteres menor do que quatro, restando um total de 1,707,681 palavras.

Devido ao número ainda muito grande de palavras restante, decidiu-se por aplicar um filtro responsável por tratar ambiguidades, nesse caso representadas por palavras repetidas. Salienta-se que, em virtude das falhas existentes no processo de OCR, não é aconselhável retirar apenas as palavras idênticas. Isso se deve a que, por exemplo, palavras como “porca” e “p0rca” podem ser a mesma palavra. Por este motivo, foram aplicados filtros utilizando métodos de similaridade como *Sequence Matcher* e a Distância de *Levenshtein*. Os filtros de ambiguidade ajudaram a reduzir o espaço de palavras para um total de 46571 palavras.

Devido ao processo de detecção e aquisição automática das palavras, ao analisar uma amostra de 1000 palavras foi possível perceber erros no processo de OCR. Certas palavras foram detectadas juntas e concatenadas em uma só. Uma vez que o processo de aquisição das *features* no sistema será dado de forma automática, a saída para resolver este problema deu-se através de um filtro básico, no qual optou-se pela remoção dessas palavras conca-

tenadas. Observando o conteúdo dos *databooks*, e levando em consideração a expertise dos inspetores envolvidos no projeto, foram retiradas todas palavras com 13 caracteres ou mais. Obtendo assim 34794 palavras.

Por fim, foi realizada uma análise manual das palavras restantes, com o propósito de assegurar uma maior qualidade quanto as palavras escolhidas. As mesmas estão foram categorizadas de acordo com a classe as quais pertencem, de forma que, após retirar palavras repetidas entre classes, chegou-se ao total de 134 palavras escolhidas como representantes das *features* para cada tipo de documento no *dataset*.

O processo de análise manual foi realizado em conjunto com inspetores, profissionais especializados na área, os quais exercem a função de consultores no projeto ATT.

5.2.2 Labels (rótulos)

Durante o processo de reconhecimento e entendimento dos *databooks* foram observados ao total 60 tipos distintos de arquivos existentes. Destes, 11 foram escolhidos como os mais importantes, segundo inspetores especialistas no assunto. Com isto, definiu-se os rótulos deste *dataset*.

Ao analisar os *datasets* de citações pode-se perceber que as classes com menos instâncias possuem valores entre 7% e 10% do total de instâncias.

Com o objetivo de aproximar o *dataset EPC* de um formato semelhante ao de citações, foram então escolhidas as cinco primeiras classes. Desta forma, a classe com menor número de instâncias representa 10.93% dos nodos no *dataset EPC*.

Para compor o *dataset* foram selecionados os testes de DU (dureza), DE (descarbonetação), MA (macrografia), IN (análise de inclusões) e QU (composição química).

A relação entre quantidade de instâncias por classe e de rótulos por nodo neste novo *dataset* são apresentadas nas Tabelas 4 e 5, respectivamente.

Tabela 4: Quantidades de instâncias em cada classe do dataset EPC.

Modelo	Label	Quantidade	(%)
EPC	DU	264	0.231579
	DE	255	0.223684
	MA	224	0.196491
	IN	214	0.187719
	QU	183	0.160526

Tabela 5: Quantidades de features em cada classe do dataset EPC.

Modelo	Label	Quantidade	(%)
EPC	DU	113	0.843284
	DE	110	0.820896
	MA	117	0.873134
	IN	96	0.716418
	QU	103	0.768657

5.2.3 Relacionamentos

Utilizou-se um grafo de relacionamentos como parte do *dataset* a ser utilizado no treinamento das redes neurais em grafos. Para fazer o levantamento das relações entre os *databooks* foi utilizado um conjunto de dados previamente desenvolvido no âmbito do Projeto ATT. Este arquivo, no formato JSON, contém informações que relacionam a capa dos *databooks* com suas demais páginas, como ilustrado na Figura 8. As demais páginas de um *databook* referem-se a diversos tipos de documentos, incluindo certificados de qualidade, resultados de ensaios de esforço em peças, testes de propriedades físicas e químicas de material.

Figura 8: *Dataset* utilizado para retirar as informações de conexões entre os *databooks*.

```

"Databook": "searchable_BOOK_PI_ ",
"Linha": [
{
  "id": "1",
  "Registros": [
    {
      "Registro_id": " ",
      "Nro_Paginas": "2",
      "Page": [
        {
          "Page_number": "1",
          "Requisito": "Mi ",
          "Registro": "078 ",
          "Corrida": "51 ",
          "lote_n": "55 ",
          "Certificado_n": "",
          "Obs": ""
        },
        {
          "Page_number": "2",
          "Requisito": "De ",
          "Registro": " ",
          "Corrida": "",
          "lote_n": "",

```

Fonte: Elaborado pelo autor

Foi realizado um processo de mineração neste arquivo com o objetivo de retirar as informações que representem ligações diretas.

Ao total foram escolhidos cinco tipos de ligações, sendo elas o número de Registro, Certificado, Lote e Corrida pertencentes a um mesmo fornecedor, totalizando 1, 460, 874 ligações. Das quais 856008 ligações são do tipo Certificado, 276411 de Fornecedor, 160796 de Corrida, 127061 de Lote e 40796 de Registro.

As ligações foram levantadas de forma manual, considerando o que se pode chamar de ligação simples, ou seja, quando dois nodos possuem, por exemplo, o mesmo número de certificado uma conexão é estabelecida entre eles. Estes tipos de elos entre documentos foram validados pelos inspetores da empresa parceira no Projeto ATT, especialistas no domínio dos dados e da aplicação.

5.2.4 Arquivos Resultantes

Foram então criado os arquivos denominados *content* e *cite* como partes integrantes do *Dataset EPC*. O arquivo *content* possui uma linha para cada um dos 1140 nodos (documentos), onde em cada linha tem-se a identificação única (ID) do nodo, a lista para a codificação *one-hot* contendo as 134 *features* seguido pela última coluna de rótulo.

Já o arquivo *cite* possui o ID do nodo origem e o ID do nodo destino para indicar a existência de uma conexão. Para cada tipo de elo foram criados arquivos *cite*. Por exemplo, o arquivo *cite_corrida*, apresentado na Figura 9b, contém a lista de conexões entre os documentos (nodos) seguindo o elo número de corrida, o qual é uma informação com alto grau de importância quanto ao processo de rastreamento de material e peças manufaturadas na indústria.

Figura 9: Arquivos que compõem o dataset EPC.

(a) Content.

0	689	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	...	0	0	0	DE	
1	645	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	...	0	0	0	DU	
2	649	0	0	0	0	1	1	0	1	1	1	0	0	0	0	0	0	...	0	0	0	QU	
3	2888	0	0	0	0	1	1	0	1	0	1	0	0	0	0	0	0	...	0	0	0	QU	
4	1026	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	...	0	0	0	MA	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1137	1201	0	0	0	1	0	0	0	1	0	1	1	0	0	0	0	1	0	0	...	0	0	IN
1138	2909	0	0	0	0	0	1	0	1	0	1	0	1	0	0	0	0	1	...	0	0	0	DU
1139	795	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	...	1	0	0	DE	

(b) Cite.

0	689	645
1	689	649
2	689	2888
3	689	1026
4	689	1566
...	...	...
160791	795	2021
160792	795	712

Fonte: Elaborado pelo autor

6 MODELOS

Neste capítulo serão brevemente apresentados os modelos selecionados. Primeiramente, são apresentadas as configurações dos modelos de baseline. Após são apresentados os modelos de GNN e por fim a descrição do processo de calibração, referente a otimização dos hiperparâmetros utilizada para explorar os modelos.

6.1 Modelos de controle

Foram escolhidos dois métodos de controle. Um método que não utiliza o aprendizado de máquina profundo e outro método que não utiliza os princípios de projeto de uma arquitetura GNN. Portanto, foram escolhidos o algoritmo de aprendizado supervisionado KNN (*K-nearest-neighbor*), o qual já foi brevemente introduzido na Seção 2.2, e uma Rede Neural Artificial, Seção 2.3, a qual utiliza como entrada apenas o one-hot das features. Ou seja, o conteúdo do arquivo content mencionado anteriormente e entrega como saída a probabilidade de cada classe para os nodos.

No algoritmo KNN foram utilizados três valores para k , sendo eles: 3, 6 e 7. Por outro lado, a Rede Neural foi construída de forma a ter duas camadas, *drop-out rate* de 0.5, função de ativação ReLU, 16 unidades internas e *cross-entropy* como a função *loss*.

6.2 Arquiteturas de GNN

Foram escolhidos quatro arquiteturas de redes neurais em grafos, usualmente aplicadas à tarefa de classificação de nodos. O objetivo de cada modelo a ser criado é predizer o “assunto” de um documento baseando-se nas características internas do documento (*features*) e nas interconexões do mesmo com outros documentos do *dataset*.

A escolha dos modelos se deu de forma que os mesmos supram totalmente ou agreguem valor quanto as falhas uns dos outros, de forma a entregar diferentes abordagens para a problemática em questão. No total, para realizar a análise proposta neste trabalho foram escolhidos quatro modelos: *Graph Convolutional Network* (Kipf & Welling 2016), *Cheb* (Defferrard et al. 2016), *ARMA* (Bianchi et al. 2019) e *Graph Attention Networks* (Veličković et al. 2017).

Desses, três utilizam de estratégias espectrais para a resolução, enquanto GAT utiliza a abordagem de atenção (*attention-based*). O GCN é uma versão de primeira ordem do Cheb, enquanto o ARMA é uma derivação de GCN no qual foram aplicados filtros específicos em suas camadas de convolução. A escolha dos mesmos foi realizada de forma a representar os primeiros esforços e os métodos mais utilizados para trabalhar com grafos em redes neurais relatados na literatura. Também contemplam duas classes distintas de soluções, as espectrais e as baseadas em mecanismos de atenção.

6.3 Calibração dos Modelos

Para realizar os experimentos foram usadas as mesmas configurações propostas por (Yang et al. 2016) para aprendizado semi-supervisionado, no qual cada *dataset* foi dividido em vinte nodos rotulados de cada classe para o conjunto de treinamento e separando, assim como em (Kipf & Welling 2016), valores fixos de 500 nodos para o conjunto de validação e 1000 nodos para o conjunto de testes. É importante ressaltar que esta partição dos dados não usual sob a ótica do aprendizado supervisionado, é de fato padrão para os problemas de aprendizado semi-supervisionado. Neste caso, tem-se dados não-rotulados que também são utilizados nas GNNs como parte do aprendizado.

A Tabela 6 resume os valores utilizados em cada conjunto.

Tabela 6: Separação dos dados utilizada.

Dataset	Treino(rotulados)	Validação	Teste	Não-rotulados
Cora	140	500	1000	1068
Citeseer	120	500	1000	1707
Pubmed	60	500	1000	18157

A implementação dos quatro modelos foi feita através da biblioteca de *deep learning PyTorch Geometric* (PyG) proposta por (Fey & Lenssen 2019). Para cada modelo foi realizada uma série de experimentos alternando apenas uma configuração por vez, entregando as mesmas condições de execução para todos os modelos escolhidos. Cada experimento foi executado cinco vezes.

Devido às limitações quanto ao hardware disponível, os experimentos foram realizados utilizando o *Google Colaboratory*, um ambiente computacional na nuvem baseado em *notebooks Jupyter*. A fim de avaliar o desempenho dos modelos foram usadas as métricas: média simples e desvio padrão da acurácia, resultados da *loss* no treinamento e validação, e a duração de cada teste.

6.3.1 Modelos *Baseline* e GNN

Primeiramente foram realizados experimentos variando a época de treinamento até o valor máximo de 200, mantendo as configurações padrões descritas na literatura para cada modelo. Para os modelos GCN, Cheb e ARMA utilizou-se a função de ativação *ReLU*. O modelo GAT seguiu as configurações especificadas em (Veličković et al. 2017) em que utiliza-se a função de ativação *eLU*.

Os resultados de validação dos modelos utilizando as configurações padrões, assim como na literatura, foram então comparados com os modelos denominados de *baseline*. O algoritmo KNN foi executado com três diferentes valores de K , sendo eles 3, 6 e 7, escolhidos de forma empírica. Por outro lado, a rede neural utilizada possui duas camadas, 16 unidades internas, função de ativação ReLU e *cross entropy* como função *loss*.

6.3.2 Configurações de GNNs

Para o processo de exploração das configurações, três tipos de experimentos foram realizados. O primeiro corresponde em mudar a função de ativação, desta forma, variando de acordo com as funções: ELU, ReLU e softmax. Após, foram feitos experimentos com dois tipos de função *loss*, a saber: *Negative Log Likelihood (NLL)* e *Cross Entropy*. Por fim, foram realizados experimentos alterando o valor do parâmetro de *drop-out rate*, realizando-se um total de cinco experimentos para cada modelo. *Drop-out rate* teve a seguinte variação de valor: 0.5, 0.6, 0.7, 0.8 e 0.9.

7 EXPERIMENTOS E DISCUSSÕES

Neste capítulo serão apresentados os resultados obtidos nos experimentos realizados durante a pesquisa. Em um primeiro momento, é feita a comparação entre os modelos de *baseline* escolhidos e os quatro modelos de GNN. Após são demonstrados os valores das métricas de desempenho para os modelos de GNN obtidos nos experimentos e os publicados na literatura. Em seguida, nos experimentos para a etapa de calibração, no qual é feita uma análise por época para cada modelo, precedida pelos experimentos variando a função de ativação, função *loss* e diferentes valores para o *drop-out*.

Algumas tabelas e figuras adicionais relacionadas aos experimentos podem ser encontradas nos Anexos A e B, respectivamente.

7.1 *Baseline* e Calibração dos modelos GNN

Durante esta fase, primeiramente, foram feitos experimentos com propósito comparativo entre os modelos de *baseline*, controle e GNNs.

Todos os modelos GNNs fizeram uso do particionamento dos dados para aprendizado semi-supervisionado, utilizando, desta forma, 20 nodos por classe para o treino da rede, 500 nodos para validação e 1000 nodos para teste, como proposto em (Yang et al. 2016).

A Tabela 7 mostra os resultados de acurácia média dos dados de teste e a duração obtida (em minutos) para 200 épocas de treinamento de GNNs. Os experimentos utilizaram *datasets* públicos.

Tabela 7: Experimentos com os modelos de *baseline* e configuração padrão de GNNs.

Dataset	Modelo	Acurácia	Duração (min)
Cora	KNN (k=3)	0.289±0.000	0.482
	KNN (k=6)	0.332±0.000	0.480
	KNN (k=7)	0.390±0.000	0.481
	RNA	0.554± 0.004	12.38
	GCN	0.814±0.006	1.935
	Cheb	0.805±0.009	10.428
	ARMA	0.822±0.006	6.328
	GAT	0.831±0.005	8.698
Citeseer	KNN (k=3)	0.333±0.000	0.885
	KNN (k=6)	0.379±0.000	0.895
	KNN (k=7)	0.368±0.000	0.897
	RNA	0.532±0.012	17.45
	GCN	0.710±0.009	2.241
	Cheb	0.698±0.011	8.361
	ARMA	0.708±0.008	16.803
	GAT	0.708±0.005	9.361
Pubmed	KNN (k=3)	0.581±0.000	0.086
	KNN (k=6)	0.534±0.000	0.086
	KNN (k=7)	0.514±0.000	0.085
	RNA	0.640±0.005	9.76
	GCN	0.788±0.007	1.531
	Cheb	0.781±0.005	10.921
	ARMA	0.787±0.003	18.986
	GAT	0.790±0.003	13.396

Pode-se perceber que, como era esperado, o algoritmo KNN apresentou os resultados mais baixos de acurácia em todos os *datasets*, independente do valor de K utilizado. A rede neural (RNA), apesar de apresentar uma acurácia média maior que a acurácia do KNN, ainda assim mostrou um resultado insatisfatório.

Cabe ressaltar, que os melhores resultados de desempenho para os modelos *baseline* foram observados no uso do *dataset* Pubmed. O KNN obteve resultados de 28% a 39% de acurácia nos *datasets* Cora e Citeseer, enquanto no Pubmed chegou a obter acurácia de 58%. Este resultado pode ter devido ao fato de que o algoritmo KNN, utiliza aprendizado baseado em instância. Como o *dataset* Pubmed possui um número grande de instâncias e a tarefa de classificação emprega apenas três classes podem ter contribuído para o melhor desempenho frente aos outros. Também dentro do *dataset* Pubmed observa-se a maior acurácia para uma rede neural (RNA) chegando a 64%.

Todos modelos utilizados como *baseline* apresentaram valores médios de acurácia inferiores aos quatro modelos que utilizam redes neurais em grafos (GNNs). Desta forma

em relação ao aprendizado profundo, entregar aos modelos apenas as informações referentes as *features* dos documentos, descartando a informação de conexões com nodos vizinhos e as informações advindas deles (grafo), acaba por apresentar uma performance de classificação inferior aos modelos que fazem uso dessas informações extras. O KNN, que apesar de utilizar de certa forma as informações da vizinhança, acaba por não ser eficiente nestes datasets tendo em vista que sua classificação é dada pela similaridade entre os dados.

A análise quanto a duração acabou por não trazer grandes revelações, como já era previsto. Ao executar o algoritmo KNN, o mesmo levou menos de um minuto para entregar o resultado por conta de sua simplicidade de implementação. Enquanto a rede neural levou em média 13 minutos para ser treinada, valor este superior aos valores entregues por alguns dos outros modelos avaliados. Quanto aos modelos de GNN, de maneira geral, os mesmos possuem tempos de duração próximos, havendo um distanciamento inicial a partir de 50 épocas de treino o qual se torna expressivo por volta de 100 épocas. Os valores de duração do processo de treinamento podem ser encontrados nos Anexos A e B.

Observa-se que o treinamento dos modelos KNN teve duração mais curta para o *dataset* Pubmed. Uma possível explicação deve-se ao fato de que neste caso apenas 60 nodos/documentos (20 por classe) foram utilizados durante o treinamento. O tamanho deste conjunto de dados é quase duas vezes menor que os dados para treinamento dos *datasets* Citeseer e Cora.

Uma vez feitas as comparações entre os modelos (*baseline* versus GNNs), iniciou-se um processo chamado de calibração dos modelos, o qual teve como objetivo a exploração e entendimento do funcionamento dos modelos de GNN mediante a variação de alguns parâmetros.

7.1.1 Experimentos com épocas

O primeiro experimento se deu ao avaliar os valores médios de acurácia durante a fase de treinamento, fixando o número de épocas em 200.

Os valores de acurácia média obtidos em 200 épocas foram então comparados com os resultados de acurácia publicados na literatura para os métodos conforme apresentado na Tabela 8.

Cada modelo GNN foi avaliado separadamente conforme suas curvas de acurácia e *loss*, com o intuito de auxiliar no entendimento individual dos princípios de funcionamento.

7.1.1.1 Modelo GCN

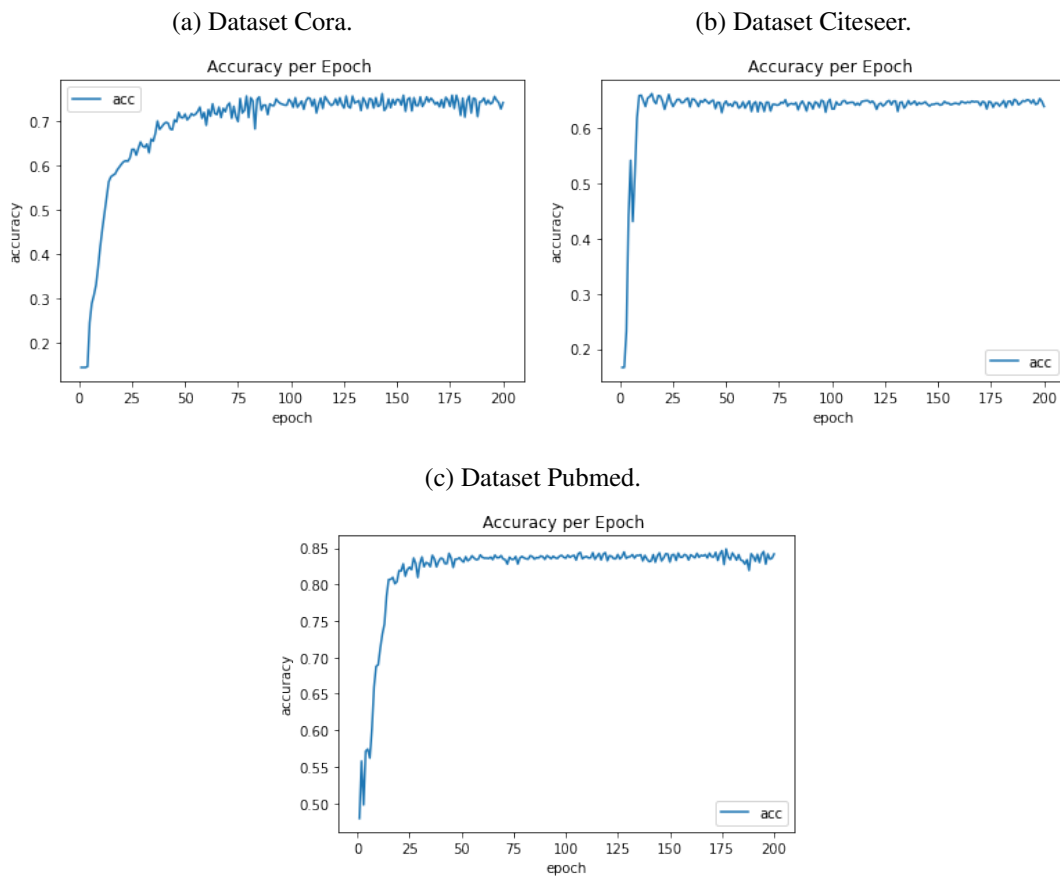
Na Figura 10 tem-se a acurácia média durante o processo de treinamento obtidos para os três *datasets* ao serem empregados no modelo GCN.

Pode-se observar que o modelo convergiu por volta de 25 épocas (ou menos) de trei-

Tabela 8: Valores de acurácia média da literatura após 200 épocas de treinamento.

Método	Cora	Citeseer	Pubmed
GCN (Kipf & Welling 2016)	0.81	0.70	0.79
Cheb (Defferrard et al. 2016)	0.79	0.69	0.74
ARMA (Bianchi et al. 2019)	0.83	0.72	0.78
GAT(Veličković et al. 2017)	0.83	0.72	0.79

Figura 10: Modelo GCN: Acurácia média durante o processo de treinamento.



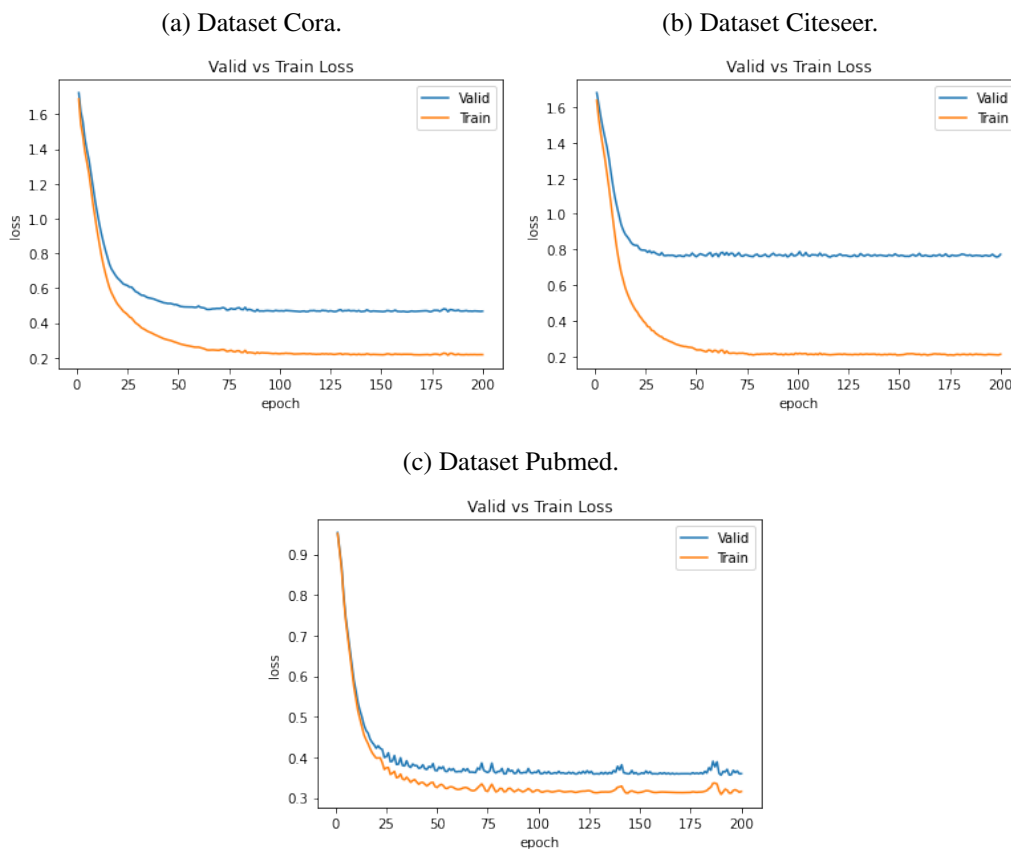
Fonte: Elaborado pelo autor

namento para *datasets* utilizados, atingindo acurácia média superior a 70% em todos os casos. Durante a etapa de teste, foram reportados acurácias de 0.814 ± 0.006 , 0.710 ± 0.009 e 0.78 ± 0.005 para o Cora, Citeseer e Pubmed, respectivamente.

Estes valores de acurácia estão compatíveis aos reportados na publicação que propõe a rede GCN (ver Tabela 8). Resultados com valores de acurácia para dados de teste obtidos variando o número de épocas na faixa de 1 a 200 encontram-se no Anexo A. Na Figura 11 tem-se o contraste das curvas de *loss* para treinamento e validação.

Pode-se avaliar que os valores *loss* diminuem com o tempo, atingindo valores abaixo de 1. As curvas de validação e treino apresentam um decaimento expressivo e convergem próximas uma da outra. Apesar do *dataset* citeseer apresentar uma diferença maior entre as curvas *loss* treino e *loss* validação, ainda assim não representam situações de subajuste ou sobreajuste. O *underfitting*, ou subajuste, refere-se à situação em que o “modelo não tem capacidade de aprender”. Overfitting, sobreajuste, surge nos casos em que o modelo “memoriza os dados” e desta forma não cria mecanismos de generalização.

Figura 11: Modelos GCN: Comparação de *loss* no treinamento e validação.



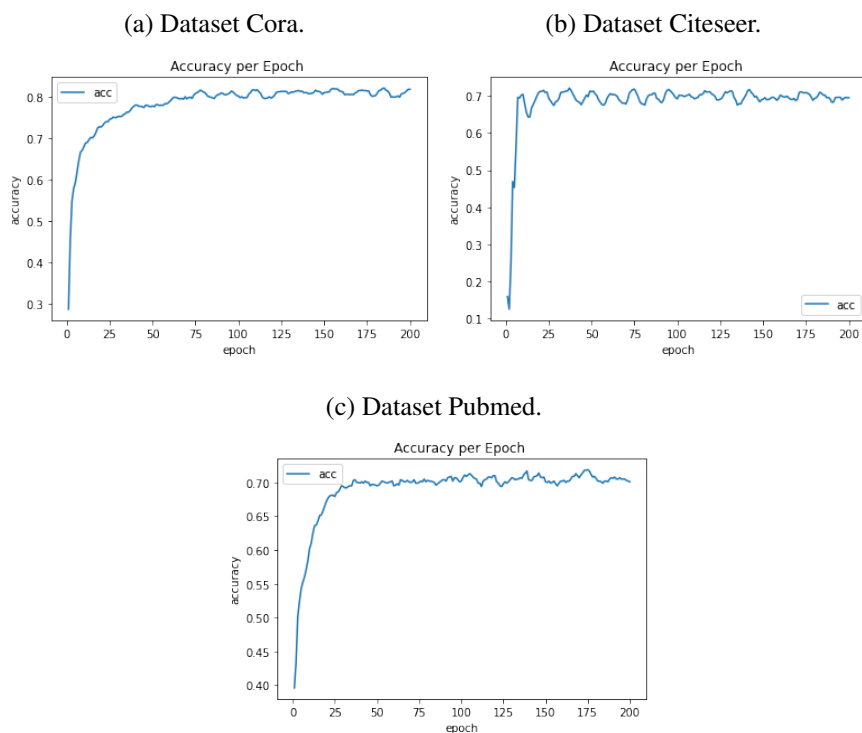
Fonte: Elaborado pelo autor

7.1.1.2 Modelo Cheb

Na Figura 12 tem-se os valores de acurácia para um modelo Cheb treinado. É possível observar que o processo de minimização da função *loss*, assim como no caso do GCN, convergiu por volta de 25 épocas para os três *datasets*. Durante a fase de testes deste modelo (200 épocas), obteve-se acurácias de 0.805 ± 0.009 (Cora), 0.699 ± 0.012 (Citeseer) e 0.782 ± 0.006 (Pubmed). Desta forma, estes resultados estão coerentes com os valores obtidos na literatura Tabela 8).

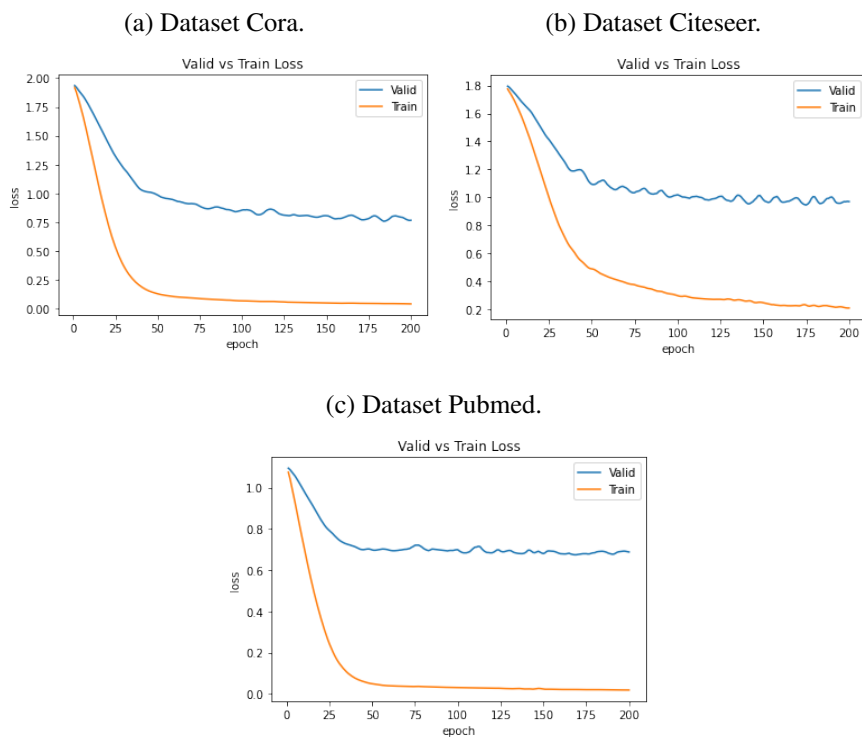
A Figura 13 apresenta as curvas de *loss* de treino e validação. Nota-se, neste caso, que as curvas apresentam um maior distanciamento uma da outra. Estas situações são preocupantes e podem caracterizar um possível *overfitting* do modelo.

Figura 12: Modelo Cheb: Valores de acurácia durante o processo de treinamento.



Fonte: Elaborado pelo autor

Figura 13: Modelos Cheb: Comparação entre os valores de loss para o conjunto de treino e validação.

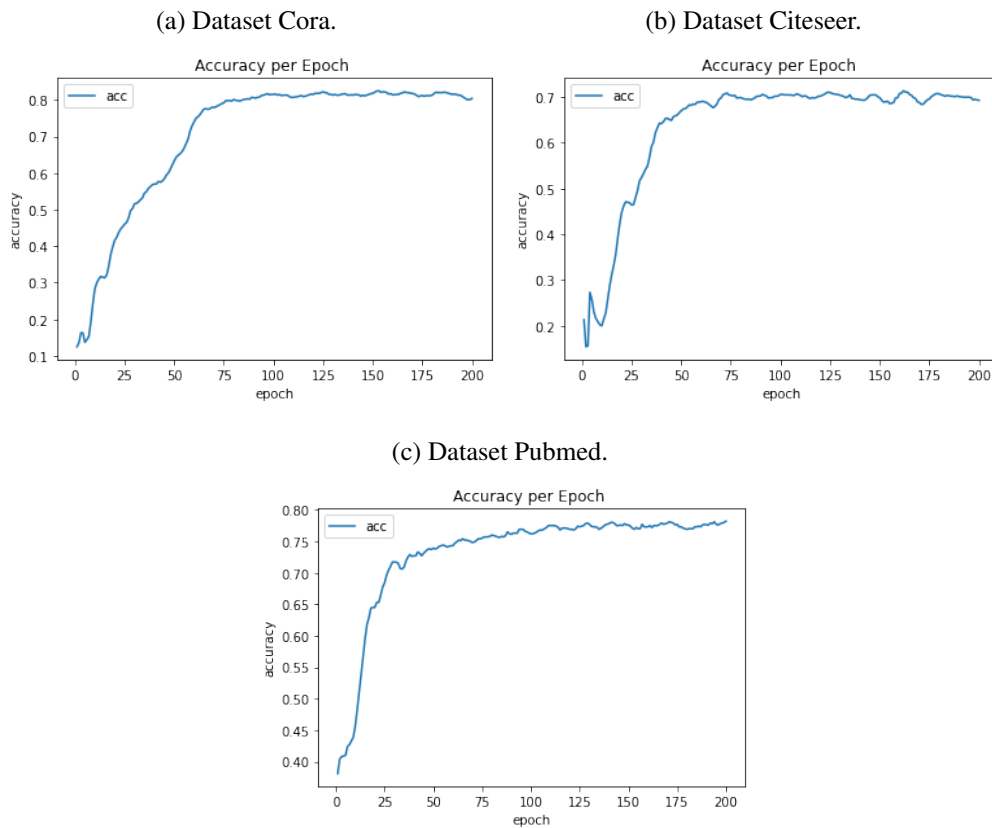


Fonte: Elaborado pelo autor

7.1.1.3 Modelo ARMA

A Figura 14 apresenta os valores de acurácia média obtidos nos experimentos com o treinamento e validação de um modelo ARMA.

Figura 14: Modelo ARMA: Valores de acurácia no processo de treinamento.



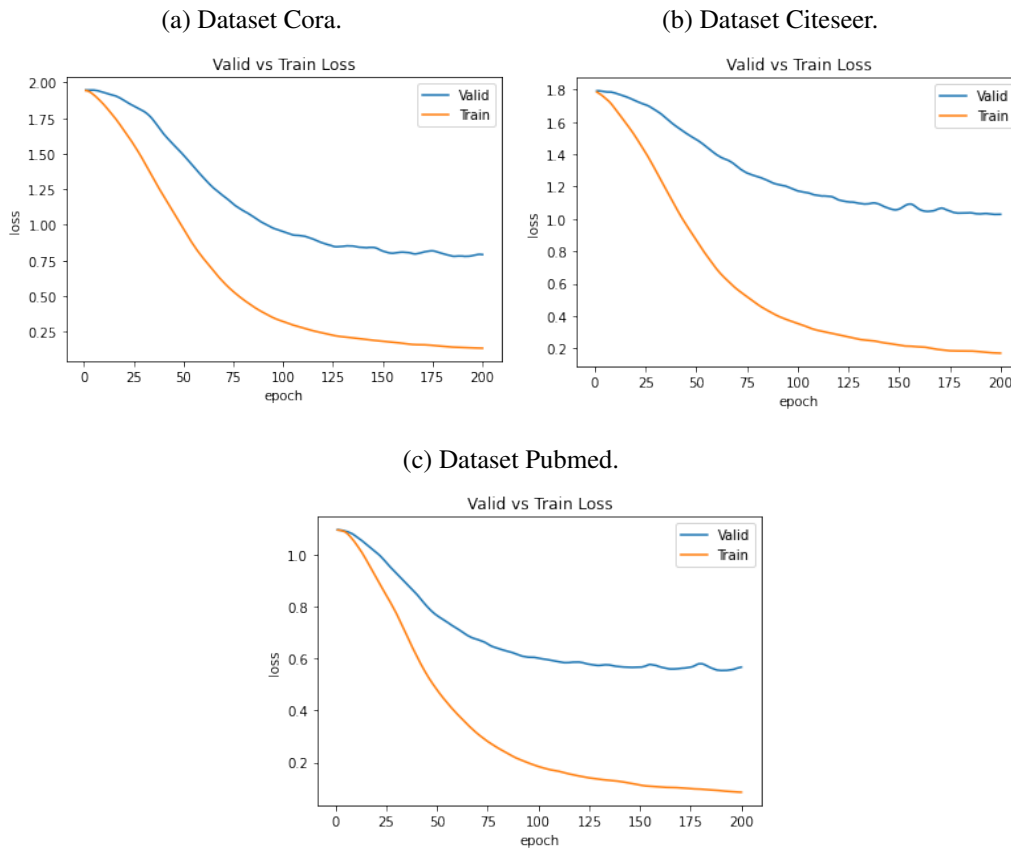
Fonte: Elaborado pelo autor

Diferente dos modelos apresentados anteriormente, o ARMA aparenta convergir por volta de 50 épocas de treinamento, atingindo valores de acurácia média próximos a 70%. O modelo ARMA também demonstra uma pequena instabilidade durante as épocas iniciais para o dataset Citeseer.

O modelo ARMA com 200 épocas de treinamento atingiu uma acurácia (dados de teste) de 0.822 ± 0.006 , 0.708 ± 0.008 e 0.787 ± 0.003 , para o Cora, Citeseer e Pubmed, respectivamente. De acordo com a Tabela 8, os resultados obtidos nestes experimentos estão coerentes. Uma pequena diferença de aproximadamente 2% foi observada no caso dos dados Citeseer.

Pode-se observar na Figura 15 que as curvas de *loss* para treino e validação apresentam uma característica diferente se comparadas com as curvas dos modelos apresentados anteriormente. Apesar da acurácia “convergir” em torno de 50 épocas, os valores de *loss* para treinamento atingem patamares próximos aos modelos anteriores por volta de 100 épocas. E pode-se também observar que, assim como no modelo Cheb, existe um distanciamento entre as duas curvas.

Figura 15: Modelos ARMA: Comparação entre as curvas de *loss*.



Fonte: Elaborado pelo autor

7.1.1.4 Modelo GAT

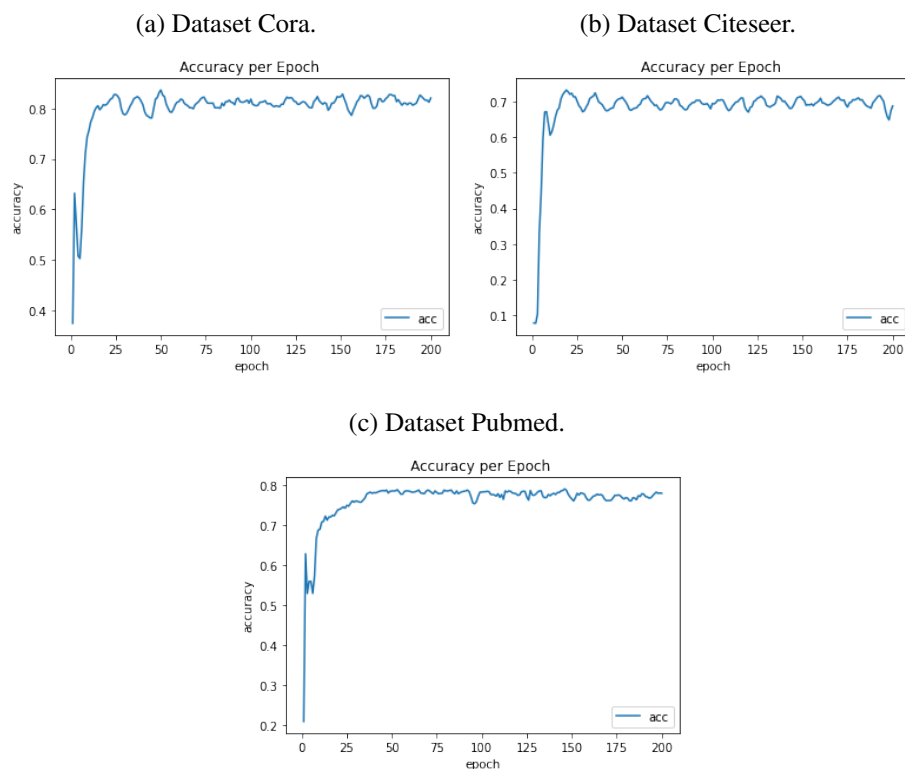
A Figura 16 apresenta os resultados de acurácia para um modelo GAT treinado com os *datasets* de *benchmarking*. O treinamento converge por volta de 25 épocas e apresenta uma certa variação durante as épocas iniciais.

Nos *datasets* Cora e Citeseer percebe-se um padrão de variação que se mantém até o ponto final do treinamento. O desempenho do modelo é satisfatório, apresentando valores de acurácia média de em torno de 80% para o Cora e Pubmed. No caso do Citeseer, a acurácia média é 70%.

O modelo GAT, treinado com 200 épocas, apresentou acurácias para os dados de teste: 0.831 ± 0.005 (Cora), 0.708 ± 0.005 (Citeseer) e 0.790 ± 0.003 (Pubmed). Novamente, os resultados obtidos encontram-se compatíveis com os obtidos na literatura (Tabela 8).

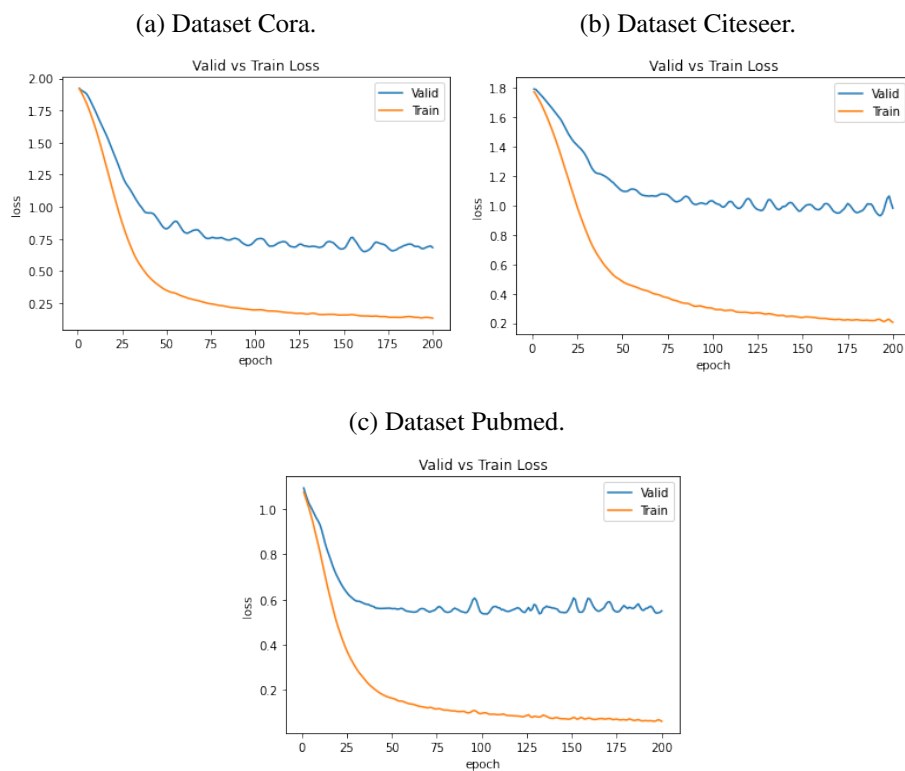
Já na Figura 17 tem-se as curvas de *loss* para treinamento e validação do modelo GAT. Novamente, percebe-se valores de *loss* superiores a 1 (caso do Citeseer). Observa-se um distanciamento das treino e validação, o que poderia indicar uma possível situação de *overfitting*.

Figura 16: Modelo GAT: Valores de acurácia durante o processo de treinamento.



Fonte: Elaborado pelo autor

Figura 17: Modelos GAT: Comparação entre curvas *loss* de treino e validação.



Fonte: Elaborado pelo autor

7.1.2 Experimentos por Função de ativação

Os próximos resultados foram obtidos de experimentos realizados em que altera-se a função de ativação utilizada na primeira camada, nos quatro tipos de redes. No total, foram escolhidas três funções para esta validação, sendo duas já utilizadas em alguns dos modelos e uma terceira com o objetivo de ampliar as opções existentes na comparação. Este experimento teve como objetivo explorar as diferenças entre o uso da função ReLU e ELU pelos modelos em suas arquiteturas originais, de forma a ser uma tentativa quanto a compreensão dessas escolhas. Foram utilizadas então a função ReLU, eLU e Softmax em experimentos com 200 épocas de treinamento, o valor de Drop-Out de 0.5 e NLL como a função de *loss*.

A Figura 18 demonstra de forma visual os valores de acurácia para o conjunto de teste dos três *datasets*. Já a Tabela 9 apresenta os valores da média e os respectivos desvio padrão.

Por sua vez as Tabelas 10 e 11, respectivamente, apresentam os valores mais baixos de *loss* obtidos com os conjuntos de validação e treino.

Figura 18: Valores de acurácia de teste variando a função de ativação



Fonte: Elaborado pelo autor

Como pode ser observado na Figura 18, quanto aos valores de acurácia obtidos para os dados no experimento, exceto com a função softmax, o modelo GCN obteve uma diferença ínfima em relação aos resultados com três *datasets*. A função ELU resultou em

Tabela 9: Valores de acurácia de teste variando a função de ativação.

Dataset	Função Ativação	GCN	GAT	Cheb	ARMA
CORA	ReLU	0.814±0.006	0.831±0.005	0.805±0.009	0.822±0.,006
	ELU	0.815±0.006	0.831±0.005	0.802±0.012	0.823±0.005
	Softmax	0.379±0.107	0.312±0.051	0.639±0.047	0.166±0.064
CITESEER	ReLU	0.710±0.009	0.712±0.005	0.698±0.011	0.708±0.008
	ELU	0.717±0.005	0.708±0.005	0.703±0.,011	0.711±0.006
	Softmax	0.261±0.086	0.198±0.033	0.268±0.054	0.121±0.055
PUBMED	ReLU	0.788±0.007	0.789±0.005	0.781±0.005	0.787±0.003
	ELU	0.795±0.003	0.790±0.003	0.783±0.005	0.781±0.038
	Softmax	0.628±0.,091	0.431±0.063	0.633±0.062	0.318±0.110

Tabela 10: Valores de *loss* encontrados para o conjunto de validação variando a função de ativação.

Dataset	Função Ativação	GCN	GAT	Cheb	ARMA
CORA	ReLU	0.8815	0.7293	0.8045	0.7696
	ELU	0.8396	0.7000	0.8239	0.7713
	Softmax	1.8276	1.8590	1.4323	1.9281
CITESEER	ReLU	1.0977	1.0376	1.0652	1.0862
	ELU	1.0549	0.9926	1.0584	1.0985
	Softmax	1.7370	1.7419	1.7091	1.7852
PUBMED	ReLU	0.5938	0.6087	0.5842	0.5688
	ELU	0.5661	0.5954	0.5794	0.5730
	Softmax	0.9904	1.0448	0.8833	1.0758

Tabela 11: Valores de *loss* encontrados para o conjunto de treino variando a função de ativação.

Dataset	Função Ativação	GCN	GAT	Cheb	ARMA
CORA	ReLU	0.3178	0.2299	0,0666	0,1587
	ELU	0.2787	0.1862	0.0604	0.1615
	Softmax	1.8098	1.9569	1.0802	1.9605
CITESEER	ReLU	0.4042	0.3614	0.1121	0.3237
	ELU	0.3442	0.2818	0.1048	0.3465
	Softmax	1.7005	1.8141	1.4908	1.7986
PUBMED	ReLU	0.1990	0.3007	0.0497	0.0391
	ELU	0.1568	0.2767	0.0472	0.0499
	Softmax	0.9528	1.1327	0.7312	1.0777

$0.795 \pm 0,003$ de acurácia para o *dataset* Pubmed, ou seja, 1% a mais de acurácia.

Em questão aos valores de *loss* encontrados, particularmente a função ELU apresentou valores inferiores em ambos conjuntos de validação e treino. Com a maior diferença presente no conjunto de validação sendo a entregue ao *dataset* Cora, onde obteve-se 0.8396 contra 0.8815 na ReLU.

De maneira geral, analisando tanto a acurácia nos dados de teste quanto os valores de *loss* (validação e treino) é possível identificar a função ELU como a que apresentou o melhor desempenho para a rede GCN. Este resultado difere do apresentado no artigo original do GCN, em que empregou-se a função ReLu.

O modelo Cheb, com exceção da função softmax, apresentou pouca diferença nos valores de acurácia de teste, considerando as funções ReLU e ELU para os três *datasets*. Desta vez a função ELU gerou um acurácia média menor para o *dataset* Citeseer (i.e. 0.703 ± 0.011 em contraste com 0.698 ± 0.011 no caso de ReLU). Também atingiu-se desvio padrão igual ou inferior. Apenas no *dataset* Cora a função ReLU obteve uma acurácia 0.003 maior, o que ainda representa uma diferença marginal.

O menor valor encontrado de *loss* para treino e validação foram respectivamente de 0.5794 e 0.00472, ambos para o *dataset* Pubmed.

As diferenças encontradas entre ELU e ReLU são ínfimas, ainda menores que as encontradas nos experimentos com o modelo GCN. Então, apesar de ELU aparentar ter um desempenho melhor, como a diferença é pequena ambas as funções configuram bons resultados para o modelo Cheb.

Tem-se o mesmo comportamento no ARMA, no qual os maiores valores de acurácia foram obtidos com o *dataset* Cora e os menores para o Citeseer - de forma similar aos modelos anteriores. Entretanto, cabe salientar que os valores de *loss* para os conjuntos de dados (validação e treino) são inferiores quando a função ReLu é utilizada.

Como os resultados de acurácia não demonstraram grandes discrepâncias quanto as funções de ativação ELU e ReLU, a escolha quanto a melhor função se da através dos valores de *loss*. Neste caso a função ReLU apresentou um melhor desempenho. Esse resultado corrobora o proposto no artigo que descreve este tipo de rede (Bianchi et al. 2019).

De forma análoga aos métodos apresentados anteriormente, para o GAT pode-se observar os mesmos padrões de comportamento quanto aos resultados de acurácia para os dados de teste com variação das funções de ativação.

Porém desta vez tem-se uma situação peculiar. No caso do Cora, ambas as funções ReLU e ELU obtiveram os mesmos valores de acurácia de 0.831 ± 0.005 , enquanto há uma inversão nos outros dois *datasets*. O desempenho dos modelos para o *dataset* Citeseer apresentou sua melhor acurácia de 0.712 ± 0.005 (função ReLU) e 0.790 ± 0.003 (função ELU) para o *dataset* Pubmed.

Ao analisar os valores de *loss* para treinamento e validação, nota-se então que, em

ambos conjuntos de dados e nos três dataset utilizados, a função ELU obteve valores menores de *loss*. Este resultado confirma o apresentado no artigo original que propõe a rede GAT.

7.1.3 Experimentos por Função de *loss*

Nesta seção são apresentados os resultados dos experimentos realizados, variando o tipo de função de *loss*, sendo escolhidas a função NLL e a função cross entropy.

Os experimentos foram realizados com 200 épocas de treinamento, o valor de 0.5 de drop-out e as funções de ativação utilizadas por padrão no *Pytorch Geometric* para cada modelo, as quais já foram descritas anteriormente.

Na Figura 19 é demonstrado de forma visual os valores de acurácia obtidos para o conjunto de dados para teste. A Tabela 12 apresenta esses valores e os seus desvios padrão. As Tabelas 13 e 14, por sua vez, descrevem os resultados de *loss* para os conjuntos de validação e treinamento.

Figura 19: Valores de acurácia para dados de teste variando a função de *loss*



Fonte: Elaborado pelo autor

A partir da Figura 19, para o GCN, nota-se que visualmente a diferença entre os valores de acurácia de teste entregues por ambas as funções é pequena, os valores são próximos com a maioria apresentando menos de 1% de diferença. Sendo esse valor de 1% encontrados no dataset Citeseer para os modelos GCN e ARMA.

Tabela 12: Valores de acurácia (dados de teste) variando a função de *loss*.

Dataset	Loss function	GCN	GAT	Cheb	ARMA
CORA	NLL	0.814±0.006	0.831±0.005	0.805±0.009	0.822±0.006
	cross_entropy	0.813±0.008	0.831±0.004	0.806±0.011	0.824±0.006
CITeseer	NLL	0.710±0.009	0.708±0.005	0.699±0.012	0.708±0.008
	cross_entropy	0.711±0.007	0.709±0.006	0.698±0.011	0.710±0.007
PUBMED	NLL	0.788±0.005	0.790±0.003	0.782±0.006	0.787±0.003
	cross_entropy	0.789±0.007	0.790±0.003	0.782±0.006	0.787±0.003

Tabela 13: Valores de *loss* obtidos com o conjunto de validação variando a função de *loss*.

Dataset	Loss Function	GCN	GAT	Cheb	ARMA
CORA	NLL	0.8815	0.7000	0.8045	0.7696
	cross_entropy	0.8788	0.6979	0.8094	0.7690
CITeseer	NLL	1.0977	0.9926	1.0652	1.0862
	cross_entropy	1.0984	0.9906	1.0649	1.1000
PUBMED	NLL	0.5938	0.5954	0.5842	0.5688
	cross_entropy	0.5913	0.5958	0.5824	0.5653

Tabela 14: Valores de *loss* encontrados para o conjunto de treino variando a função de *loss*.

Dataset	Loss Function	GCN	GAT	Cheb	ARMA
CORA	NLL	0.3178	0.1862	0.0666	0.1587
	cross_entropy	0.3145	0.1855	0.0667	0.1622
CITeseer	NLL	0.4042	0.2818	0.1121	0.3237
	cross_entropy	0.3998	0.2779	0.1117	0.3488
PUBMED	NLL	0.1990	0.2767	0.0497	0.0391
	cross_entropy	0.1929	0.2785	0.0499	0.0419

Em dois dos três datasets, sendo eles o Cora e Pubmed, a função NLL obteve valores de acurácia semelhantes com desvio padrão inferiores a função cross entropy.

Os valores de *loss* também são próximos uns dos outros em ambos os conjuntos de validação e treino, com os menores valores sendo encontrados para o dataset Citeseer. O modelo Cheb também teve alterações mínimas em seus valores de acurácia com dados de teste. Os valores de desvio padrão variaram conforme o *dataset*, porém ainda assim os resultados mostraram-se muito próximos uns dos outros.

O mesmo ocorreu com os valores de *loss* para os conjuntos de validação e treino,

os quais apresentam valores praticamente iguais em todos os *datasets*, fornecendo os menores valores para o Citeseer, em ambos os conjuntos (treinamento e validação).

De forma análoga, no modelo ARMA tem-se a mesma situação que nos modelos anteriores.

Os valores de acurácia e seus respectivos desvios padrão são semelhantes, chegando a obter exatamente 0.787 ± 0.003 nas duas funções para o *dataset* Pubmed.

No GAT, mais uma vez é possível perceber os padrões encontrados nos modelos anteriores. Obteve valores de acurácia de desvio padrão semelhantes para as duas funções, fornecendo exatamente 0.790 ± 0.003 para ambas funções no *dataset* Pubmed, assim como no modelo ARMA.

7.1.4 Experimentos por Drop-out

Nesta seção serão apresentados os experimentos realizados alterando os valores de drop-out em um intervalo de 0.5 até o valor extrapolado de 0.9.

Os experimentos foram feitos com 200 épocas de treino, função NLL como loss e as funções de ativação padrão para cada modelo conforme são descritos na biblioteca Pytorch Geometric.

A Figura 20 demonstra os valores de acurácia encontrados para o conjunto de teste nos três dataset disponíveis quando aplicados aos quatro modelos de GNN utilizados neste trabalho. A Tabela 25, encontrada no Anexo A, disponibiliza esses valores de acurácia em conjunto com seus respectivos desvios padrão. Enquanto tem-se nas Tabelas 15 e 16, os valores de loss para os conjuntos de validação e treino.

Tabela 15: Valores de loss encontrados para o conjunto de validação variando o drop-out.

Dataset	Drop-Out	GCN	GAT	Cheb	ARMA
CORA	0.5	0.8815	0.7000	0.8045	0.7696
	0.6	0.8892	0.6993	0.7887	0.7859
	0.7	0.9162	0.8117	0.7780	0.8091
	0.8	0.9686	1.1933	0.7723	0.8749
	0.9	1.1705	1.8764	0.7899	1.0810
CITeseer	0.5	1.0977	0.9926	1.0652	1.0862
	0.6	1.1183	0.9912	1.0663	1.1244
	0.7	1.1484	1.0945	1.0817	1.1585
	0.8	1.2109	1.3637	1.1094	1.2515
	0.9	1.4045	1.7595	1.2121	1.4383
PUBMED	0.5	0.5938	0.5954	0.5842	0.5688
	0.6	0.5993	0.5961	0.5818	0.5645
	0.7	0.6089	0.6088	0.5857	0.5658
	0.8	0.6298	0.6419	0.5961	0.5749
	0.9	0.6896	0.8452	0.6390	0.6096

Figura 20: Valores de acurácia de teste variando os valores de drop-out.



Fonte: Elaborado pelo autor

Tabela 16: Valores de loss encontrados para o conjunto de treino variando o drop-out.

Dataset	Drop-out	GCN	GAT	Cheb	ARMA
CORA	0.5	0.3178	0.1862	0.0666	0.1587
	0.6	0.3312	0.1863	0.0714	0.1799
	0.7	0.3644	0.3139	0.0815	0.2111
	0.8	0.4391	0.8340	0.1093	0.2977
	0.9	0.7297	1.8531	0.1891	0.5859
CITeseer	0.5	0.4042	0.28180	0.1121	0.3237
	0.6	0.4361	0.2807	0.1219	0.3975
	0.7	0.4945	0.4335	0.1472	0.4651
	0.8	0.6208	0.9510	0.2073	0.6582
	0.9	1.0100	1.7458	0.4413	1.0513
PUBMED	0.5	0.199	0.2767	0.0497	0.0391
	0.6	0.2085	0.2785	0.0543	0.0478
	0.7	0.2258	0.3078	0.0651	0.0480
	0.8	0.2666	0.3743	0.0945	0.0668
	0.9	0.3768	0.7275	0.1971	0.1394

Como é possível notar na Figura 20, os valores de acurácia encontrados para o modelo GCN assumem uma configuração com baixa variação, praticamente constante em todos os valores de drop-out no dataset Cora e apresentando leve decaídas de menos de 1% nos outros dois dataset com desvios padrão semelhantes. No dataset Cora seu melhor valor de acurácia (0.817 ± 0.007) foi encontrado ao utilizar 0.8 como drop-out, para o Citeseer tem-se 0.710 ± 0.008 com 0.6 de drop-out enquanto para o Pubmed foi encontrado 0.78 ± 0.005 com 0.5 de drop-out. Ao analisar-se os valores de loss encontrados para os conjuntos de validação e treino, em todos os três datasets ocorreu um aumento dos valores de loss conforme o valor de drop-out foi sendo incrementado.

Assim como no GCN, o modelo Cheb apresentou acurácias de teste com pouca variações, se mantendo com valores semelhantes durante todas as opções de drop-out testadas. Há uma pequena melhora no dataset Cora, passando de 0.805 ± 0.009 com drop-out de 0.5 para 0.819 ± 0.007 com drop-out de 0.8. No dataset Pubmed os valores de acurácia foram reduzindo conforme o drop-out foi aumentando, enquanto no Citeseer aconteceu uma situação peculiar. Inicialmente, as acurácias foram reduzindo, porém o valor de 0.699 ± 0.0012 foi encontrado tanto para o drop-out de 0.5 quanto de 0.9.

Os valores de loss para ambos os conjuntos de treino e validação foram aumentando conforme o valor de drop-out era incrementado. Com exceção do dataset Cora que no conjunto de validação teve uma queda a partir de 0.6 de drop-out até voltar ao valor inicial com drop-out de 0.9.

Já o modelo ARMA apresentou alguns resultados diferenciados. Para o dataset Cora houve um crescimento no valor de acurácia, obtendo o valor de 0.84 ± 0.007 para 0.7 de drop-out, o qual se manteve para 0.8 e novamente foi incrementado atingindo 0.825 ± 0.008 para 0.9. No Citeseer houve o incremento no drop-out 0.7, porém foi seguido pelo decaimento nos valores subsequentes. Por sua vez, o Pubmed manteve um decaimento conforme os valores de drop-out foram incrementados.

Quanto aos valores de loss para validação e treino, a mesma situação ocorrida nos outros modelos se repete para o modelo ARMA.

No modelo GAT também são encontrados situações diferentes aos outros modelos em questão de acurácia de teste, sendo essas diferenças visíveis na Figura 20. Para o dataset Pubmed há o decaimento dos valores, chegando a uma redução de 5.1% e um aumento no desvio padrão no experimento com 0.9 de drop-out. Situação semelhante é encontrada no dataset Citeseer, dessa vez o decaimento encontrado foi de 10.3%. Enquanto no Cora, apesar de haver o decaimento a partir de 0.7, o modelo obteve 0.831 ± 0.004 com o valor de 0.6, desta forma configurando a maior acurácia com o menor desvio padrão para este dataset.

De forma análoga aos demais modelos, o GAT também seguiu o padrão encontrado para os valores de loss nos conjuntos de validação e treino. Conforme o drop-out é incrementado os valores de loss também aumentam.

7.1.5 Discussão

Os resultados obtidos nos primeiros experimentos indicam que os modelos GNNs apresentam melhor desempenho (acurácia média) em relação aos modelos escolhidos como *baseline*. A rede neural convencional (RNA) demonstrou resultado satisfatório para o *dataset* Pubmed. Entretanto, esse resultado ainda pode ser considerado pontual.

Em virtude disso, demonstra-se que em aplicações com dados que possuam característica e distribuição semelhantes às presentes nos *datasets* estudados, para uma maior acurácia na tarefa de classificação de nodos, recomenda-se o uso de modelos que se utilizam de informações individuais de cada nodo e de informações da vizinhança (conexões entre nodos).

Já na etapa de calibração dos modelos foi realizada uma análise individual para cada modelo com a variação de um grupo de parâmetros.

Para o modelo GCN pode-se constatar que, nos três *datasets* de *benchmarking*, o modelo converge em torno de 25 épocas de treinamento, resultando em acurácias superiores a 70%. As curvas de *loss* para treino e validação decaem com o avanço de épocas de treinamento. A curva de validação mantém-se acima da de treino, desta forma, configurando o comportamento esperado. Ao variar as funções de ativação, os piores e os melhores resultados, tanto de acurácia de treino quanto de valores de *loss* foram obtidos utilizando as funções softmax e ELU, respectivamente. No entanto, há pouca diferença entre os resultados para as funções ReLU e ELU, recomendando-se o uso de ambas em aplicações. De forma análoga, as funções de *loss* NLL e Cross Entropy apresentaram resultados semelhantes, mostrando serem ambas boas escolhas.

Os resultados dos experimentos que variaram os valores para o parâmetro *drop-out* foram diferentes em cada *dataset*. Os valores de *loss* para treinamento e validação aumentaram com o aumento do valor de *drop-out*. As diferenças entre as acurácias são inferiores a 1%.

Em consequência disso, pode-se dizer que o valor de 0.5 representa uma boa escolha de design para o modelo.

O GCN apresentou um desempenho melhor quando treinado e validado com o *dataset* Cora.

O modelo Cheb também converge em 25 épocas de treinamento com acurácias semelhantes ao modelo anterior, porém ao observar as curvas de *loss* nota-se um distanciamento entre a curva de validação e treino, principalmente no *dataset* Pubmed. Este comportamento não é o ideal e pode representar uma situação de *overfitting* no modelo.

As funções de ativação ReLU e ELU apresentaram comportamentos praticamente semelhantes no modelo Cheb criado. O mesmo pode ser dito para a função de *loss*, em que ocorreu pouca ou nenhuma diferença significativa. No caso dos experimentos *drop-out*, o modelo Cheb não se beneficiou com a variação dos valores. Desta forma, constata-se que *drop-out* de 0.5 representa a melhor escolha.

De maneira geral, nestes experimentos, o modelo Cheb apresentou o pior desempenho dentre os quatro modelos. Este resultado é coerente já que é o modelo mais antigo. Este modelo foi também usado como base na construção da proposta de rede GCN.

Diferente dos modelos anteriores, o ARMA converge em 50 épocas de treinamento, atingindo praticamente valores de acurácia semelhantes ao GCN. No aspecto curvas de *loss*, o modelo precisou de mais épocas de treinamento para atingir valores semelhantes de *loss* entregues pelos modelos anteriores. Um resultado importante é que ocorreu um distanciamento entre a curva de treino e validação e de treino (se comparado com o GCN).

Nos experimentos com função de ativação pode-se observar que a ReLU, apesar de ter resultados de acurácia de teste próximos aos resultados para ELU, obteve-se valores de *loss* menores, logo sendo a melhor escolha para este modelo. Quanto à função de *loss*, não há diferenças substanciais para realizar uma escolha entre as duas opções, de fato, ambas funções entregam o mesmo valor de acurácia de teste no dataset Pubmed.

Além disso, percebe-se que o modelo ARMA apresenta tempos mais longos para o processo de treinamento nos *datasets* Pubmed e Citeseer. Esse tempo possivelmente se deve pelas características dos *datasets*. O Pubmed tem-se um número elevado de arestas. O Citeseer apresenta uma quantidade de *features* maior, ainda comparado com os valores encontrados no Cora.

Por fim o modelo GAT converge em 25 épocas de treinamento. Em certos experimentos obteve-se os maiores valores de acurácia com os dados de teste. As curvas de *loss* decaem conforme o treinamento avança, com a curva de validação acima do treino. Ao variar a função de ativação percebeu-se que a função ELU apresenta menores valores de *loss*. Ao variar os tipos de função de *loss*, os resultados indicam que pode-se utilizar ambas as funções *NLL* e *Cross Entropy*.

Através dos resultados dos experimentos nesta fase de calibração, pode-se chegar às seguintes observações:

1. O modelo Cheb apresentou os resultados menos satisfatórios nos experimentos.
2. O modelo ARMA mostrou-se com um desempenho aceitável. Entretanto, o tempo de convergência foi bem elevado.
3. Ao mesmo tempo que o GAT atingiu acurácias aceitáveis, este modelo apresentou valores de *loss* elevados.
4. A partir dos resultados encontrados, de maneira geral, foi possível constatar que o modelo GCN apresentou o melhor desempenho, mantendo resultados satisfatórios e consistentes para as validações realizadas.

Desta forma, a rede GCN (Kipf & Welling 2016) mostrou-se mais adequada para ser aplicada nas validações em cima do *dataset* EPC.

7.2 Classificação de Documentos EPC

Os documentos EPC, por exemplo os utilizados no projeto ATT, contêm diversas informações que configuram possíveis ligações (ou elos) entre os ensaios e testes realizados, certificados e outros arquivos pertencentes aos *databooks*. O *dataset* EPC, descrito na seção 5.2, contempla exatamente os dados das atividades de documentação relacionadas à engenharia, compra e construção na indústria de Óleo&Gás.

Esta seção tem por objetivo de realizar uma validação quantitativa de um classificador para documentos EPC, treinado a partir do *perfil de calibragem* pré-estabelecido. Uma vez que os modelos foram explorados durante o processo de calibração, deu-se início aos experimentos para criar e validar um classificador de documento EPC. Para isto, utiliza-se a rede GCN treinada e validada com o *dataset* EPC.

Os experimentos aplicaram a rede GCN após avaliar as configurações propostas no processo de calibração (função de ativação, *loss* e *drop-out*). Quanto ao particionamento do *dataset*, as frações utilizadas (*split*) para o aprendizado semi-supervisionado em (Yang et al. 2016), não puderam ser aplicadas em seus valores absolutos por conta de insuficiência de dados. Desta forma foi necessário uma adaptação no fracionamento, utilizando uma estratégia relativa, e desta forma, sem perda de generalidade.

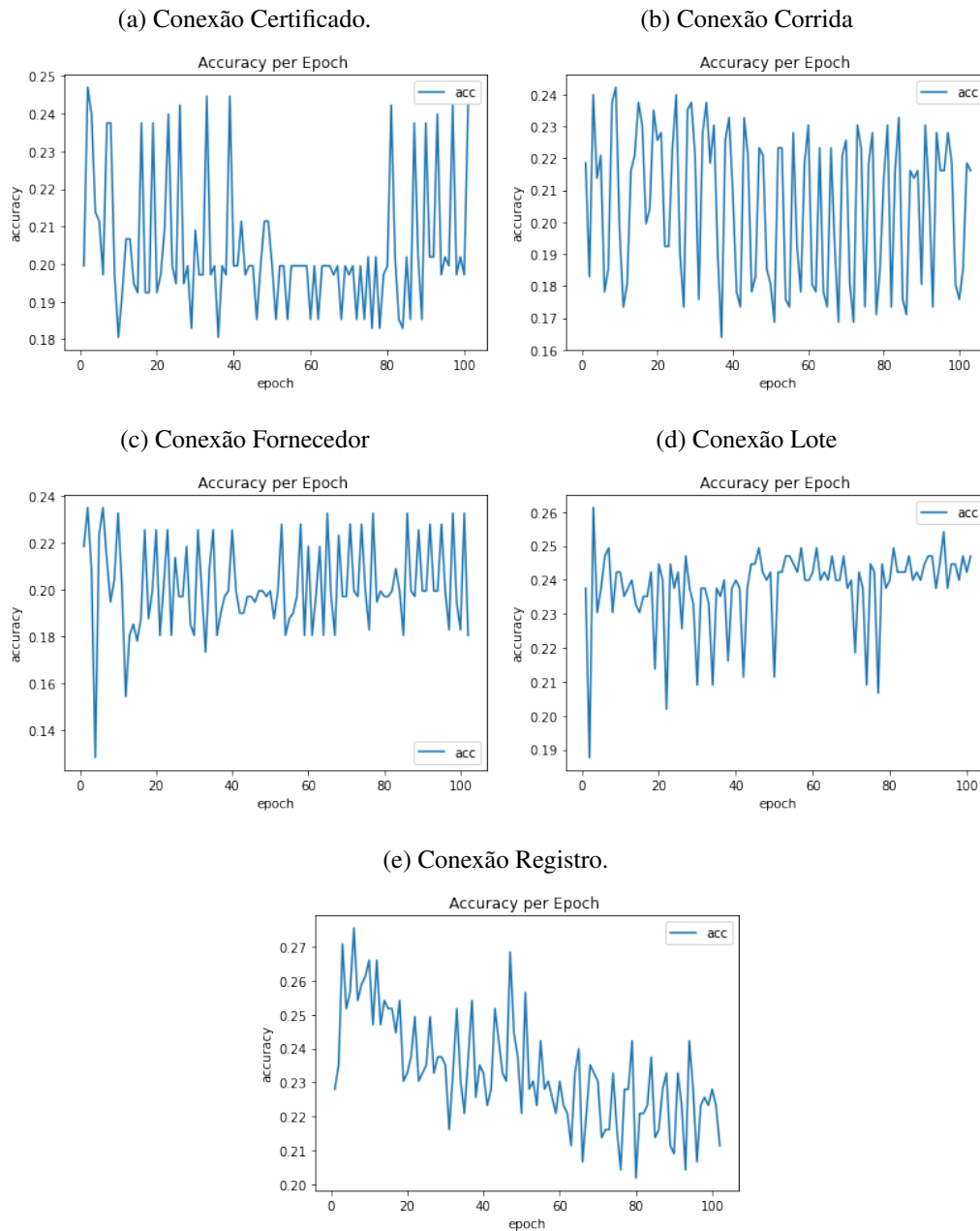
A estratégia utiliza as mesmas frações percentuais das empregadas nas validações de calibração. Por exemplo, 1000 nodos no conjunto de teste dos *datasets* correspondem a aproximadamente entre 30% e 36% do número total dos nodos. Transferindo este percentual ao *dataset* EPC, foram separados 419 nodos para o conjunto de teste (36% dos nodos disponíveis). O conjunto de treino foi criado com 20 nodos para cada classe, totalizando 100 nodos rotulados para o treinamento. Neste fracionamento, ainda restaram 621 nodos não rotulados que também são utilizados para o aprendizado semi-supervisionado.

Os experimentos foram realizados utilizando um conjunto de cinco grafos, cada um contendo ligações entre nodos (documentos) de um tipo específico: Certificado, Corrida, Fornecedor, Lote e Registro.

Todos os experimentos executaram com código implementado utilizando a biblioteca *Pytorch Geometric*. Utilizou-se a versão premium do Google Colaboratory, o qual disponibiliza uma máquina com CPU Intel Xeon CPU 2.30GHz com 13GB de RAM e uma GPU Tesla T4 com 16GB de memória.

A Figura 21 apresenta os valores de acurácia durante o processo de treino para os experimentos realizados utilizando os cinco tipos de ligações disponíveis no *dataset* EPC. Cada experimento foi executado utilizando os dados de calibração obtidos na Seção 7.1. Para o treinamento do classificador EPC, função de ativação ELU, função de *loss cross entropy* e o valor 0.5 de *drop out*. Como o modelo apresentou convergência antes de 100 épocas, para evitar o treino excessivo (*overtraining*) decidiu-se por utilizar 100 épocas de treino ao invés de 200.

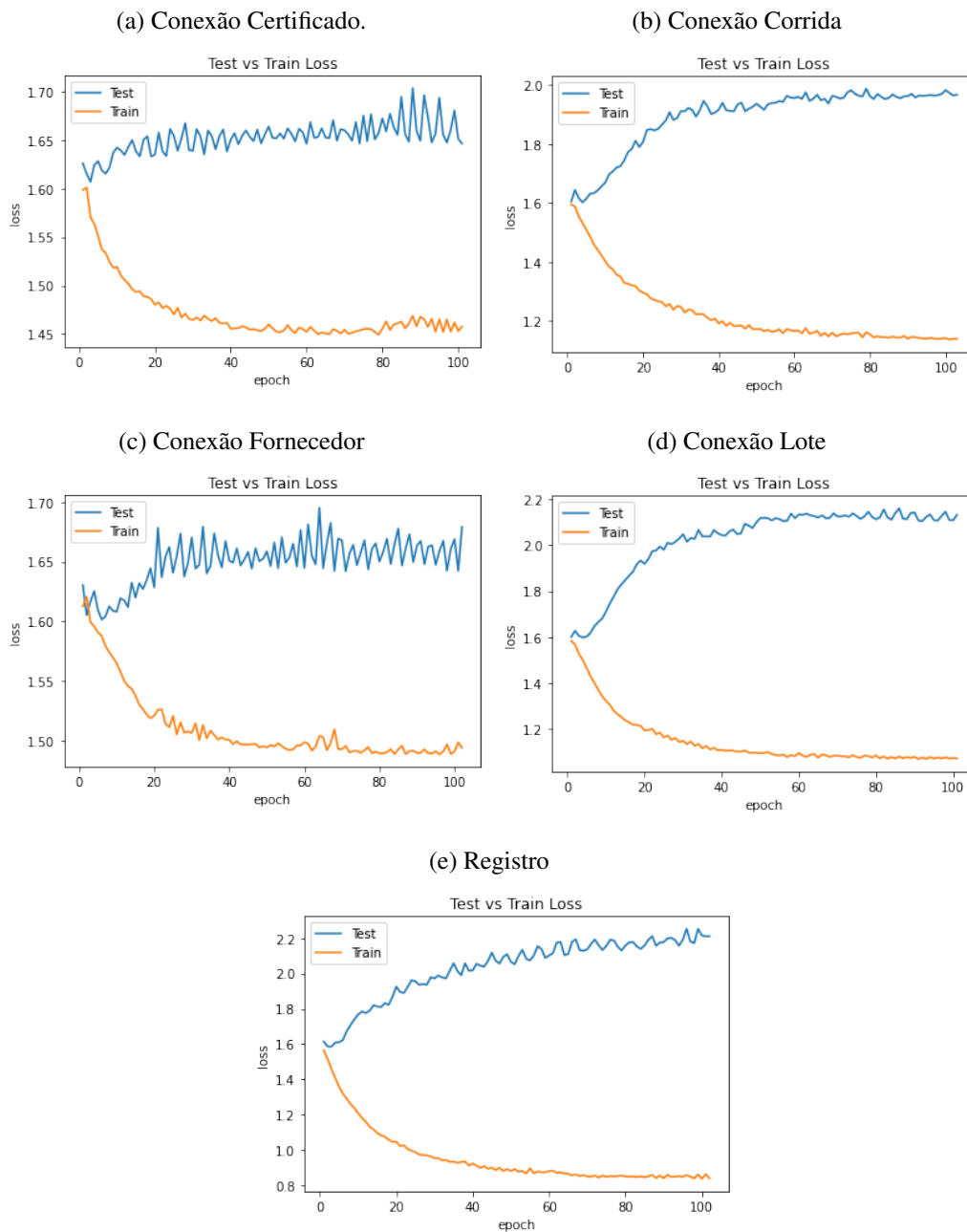
Figura 21: EPC: Valores de acurácia durante o processo de treinamento.



Fonte: Elaborado pelo autor

Pode-se perceber que em todos os cinco experimentos a acurácia apresenta uma grande variação e acaba não ultrapassando o valor de 0.27 (27%). É interessante ressaltar que o experimento para o grafo com conexão do tipo Registro gerou valores de acurácia que diminuem conforme o número de épocas aumenta.

A Figura 22 mostra as curvas de *loss* para o conjunto de treino e teste. Novamente, em todos os tipos de ligações no grafo a curva *loss* para o treinamento diminui enquanto a curva *loss* relacionada aos dados de teste aumenta com o número de épocas. Este comportamento é o oposto do que se esperava atingir e apresenta fortes indicações de que o modelo não convergiu.

Figura 22: EPC: *Loss* no treinamento versus teste.

Fonte: Elaborado pelo autor

A Tabela 17 contém os valores de acurácia obtidos para o conjunto de teste e a duração do processo de treinamento do modelo GCN separados pelos tipos de conexão. Como pode-se observar o modelo entregou acurácias baixas. O experimento com modelo utilizando o grafo com conexão por Certificado apresentou acurácia de 0.197 ± 0.027 e o grafo conectado por Registro 0.254 ± 0.010 , a pior e a melhor acurácia, respectivamente.

Os resultados obtidos nestes experimentos foram consideravelmente abaixo do esperado para modelos de GNN. Conforme foi observado na Seção 7.1, durante os experimentos realizados o modelo GCN manteve um comportamento estável, resultando em valores

Tabela 17: EPC: Valores de acurácia de teste e duração do treino.

Dataset	Conexão	Acurácia	Duração (min)
EPC	Certificado	0.197 ± 0.027	12.620
	Corrida	0.234 ± 0.05	2.114
	Fornecedor	0.226 ± 0.007	3.950
	Lote	0.221 ± 0.013	2.312
	Registro	0.254 ± 0.010	0.294

de acurácia acima de 0.60.

Tendo em vista que o dataset EPC foi desenvolvido utilizando dados de documentos em um cenário real e seguiu o mesmo aspecto construtivo dos *datasets* de *benchmarking* Cora e Citeseer, surge então a pergunta:

- Como os resultados de acurácia para o classificador EPC estão abaixo do esperado?

Uma série de verificações básicas (*sanity checking*), que perdurou por duas semanas inteiras, foram realizadas no *dataset* EPC e nos códigos Python dentro dos vários Google colab notebooks, respondendo as seguintes colocações:

- A lista de *features* compostas por palavras dos documentos foi construída corretamente? A técnica de *one-hot encoding* gerou dados consistentes?
- A lista de *features* está corretamente armazenada no arquivo nomeado *content*? Amostras aleatórias de linhas do arquivo *content* corresponde as *features*, ou palavras, encontradas nos respectivos documentos originais?
- As conexões nos grafos foram criadas corretamente? As conexões estão devidamente representadas no arquivo *cite*? As conexões seguem o formato dos *datasets* Cora, Citeseer e Pubmed?

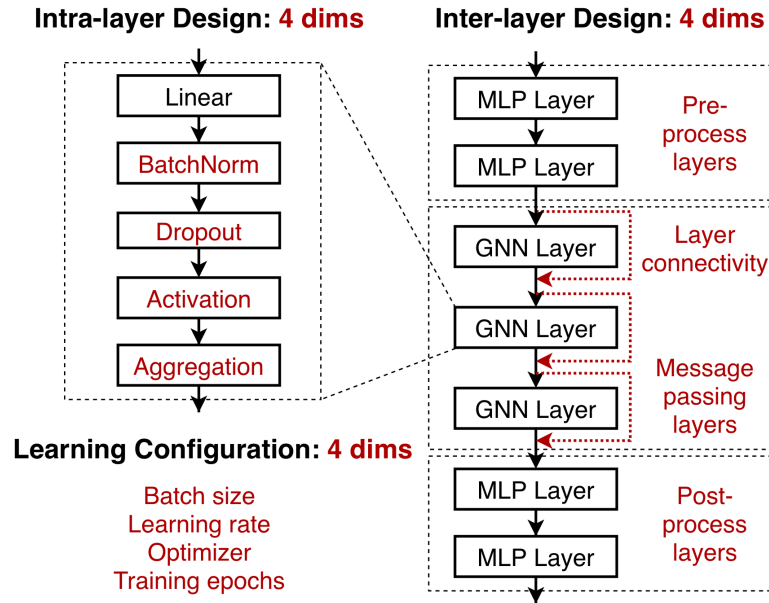
Após um período extenso de verificações, partiu-se para um processo complementar de exploração de hiperparâmetros. Utilizou-se para isto o Graphgym (You et al. 2020), um framework desenvolvido pela Universidade de Stanford, criado especificamente para validar novos projetos de redes GNNs.

7.2.1 GraphGym

GraphGym é uma plataforma proposta por (You et al. 2020) com o objetivo de projetar e avaliar arquiteturas de Redes Neurais em Grafos (Graph Neural Networks - GNNs) e seu desempenho quando aplicadas em diversas tarefas como classificação de nodos e arestas. O *framework* trabalha com três principais componentes: espaço de projeto (*design space*), espaço das tarefas (*task space*) e espaço de avaliação de projeto (*design space evaluation*).

O componente de espaço de projeto refere-se à aspectos da arquitetura de uma GNN, sendo eles o projeto de aspectos dentro de uma camada (*intra-layer design*), entre as camadas (*inter-layer design*) e configuração de aprendizagem (*learning configuration*). Existem quatro combinações para cada componente, gerando ao total 12 dimensões, o que resulta em 315, 000 projetos possíveis, como mostra a Figura 23.

Figura 23: *GNN Design Space*.



Fonte: (You et al. 2020)

Por sua vez, o espaço de tarefas no Gymgraph propõe uma métrica de similaridade entre os resultados obtidos pelo modelo ao ser empregado a um conjunto de tarefas diversificado. Essa similaridade é calculada ao medir a *Kendall rank correlation* (Abdi 2007) dos desempenhos de um conjunto fixo de arquiteturas GNN. Ao total são consideradas 32 tarefas, sendo estas: 12 tarefas de classificação de nodos sintéticos, 8 tarefas de classificação de grafos sintéticos, 6 tarefas de classificação em cenário real e 6 classificação de grafo.

O espaço de avaliação de projeto (*design space evaluation*) tem como objetivo avaliar e entregar algum tipo de informação relevante a partir dos resultados obtidos pelos outros componentes. Como, ao combinar ambos espaço de projeto e de tarefas resulta-se em mais de 10 milhões de combinações, para isto, desenvolveu-se então um procedimento de avaliação controlado de busca aleatória.

Para os experimentos no dataset EPC utilizou-se o componente de espaço de projeto (*design space*). A fim de validar os experimentos da Seção 7.1, escolheu-se variar no espaço interno de camada a opção de *drop out* (0.5 a 0.7) e a função de ativação (eLU e ReLU). A configuração de aprendizagem utilizou o otimizador Adam e SGD. O treinamento foi realizado com 50 épocas.

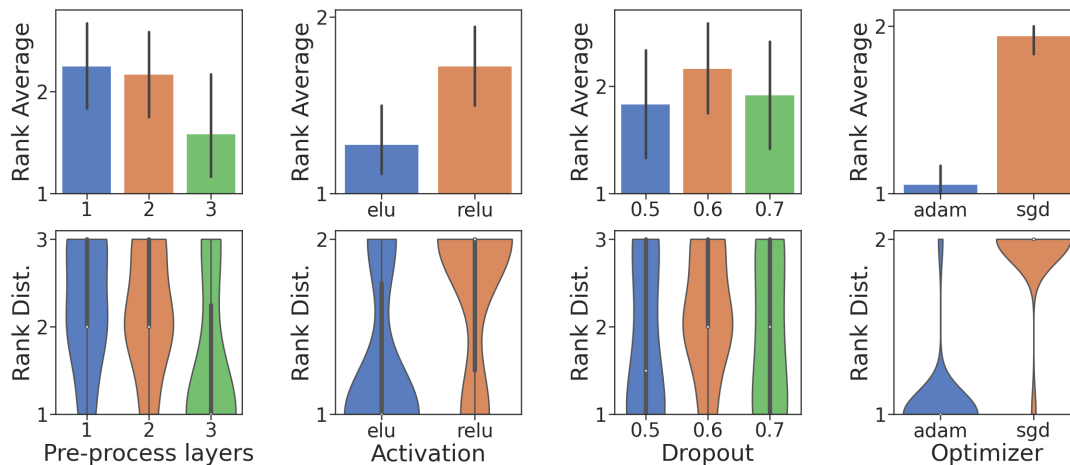
Como não existe a opção de variar a função *loss*, optou-se por adicionar mais um tipo de variação à configuração do experimento nos aspectos internos da camada com o

acréscimos de camadas de pré processamento, testando 1,2 e 3 camadas. Ao total o experimento executou de forma automatizada o número de 36 combinações de configurações.

Nas Figuras 24, 25, 26, 27 e 28 são apresentados os resultados do conjunto de teste (EPC) para cada uma de suas conexões. Primeiro, tem-se os gráficos de barra que trazem a classificação média para cada opção de projeto, no qual um valor baixo representa uma melhor escolha de projeto. Um *rank* equivalente a 1 significa que a opção sempre alcança o melhor desempenho dentre as outras opções disponíveis. Logo após tem-se os gráficos no formato de violino, os quais representam a distribuição da escolha para cada opção de projeto, os quais são utilizados para avaliar com qual frequência uma opção é classificada como 1 ou não.

A Figura 24 apresenta os resultados para a conexão do tipo Registro no grafo. Analisando cada sub-figura individualmente, percebe-se que o valor de 3 camadas de pré-processamento teve o *rank* mais próximo do valor 1 pelo gráfico de barras. No gráfico de violino confirma-se este valor, pois a opção demonstra a maior distribuição de escolhas perto do *rank* de 1. Em outras palavras, esta situação refere-se ao fato que ao utilizar 3 camadas de pré-processamento os resultados tendem a ser próximos de um rank próximo de 1.

Figura 24: Resultados para o dataset EPC com conexões do tipo Registro.



Fonte: Elaborado pelo autor

A próxima sub-figura refere-se a função de ativação, a qual neste caso demonstrou que a função ELU visivelmente possui o ranking mais próximo de 1. O gráfico de violino, novamente confirma esta informação ao mostrar que a base do violino, que representa maior frequência de casos com ranking 1, é visivelmente maior na função ELU do que na função ReLU.

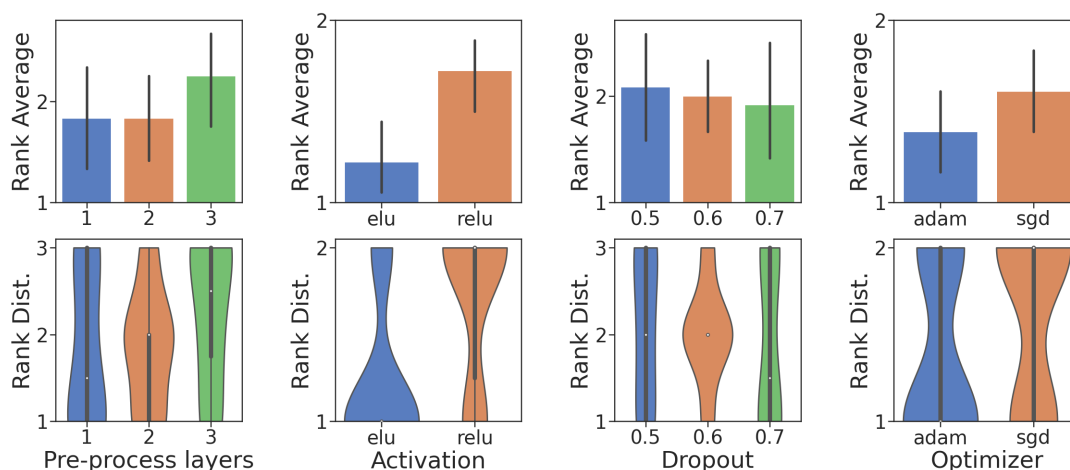
Quanto a opção de drop-out os gráficos assumem estruturas visuais diferentes, uma vez que há três opções para variação. Analisando apenas o gráfico de barras tanto o valor de 0.6 como de 0.7 apresentam resultados próximos. Neste caso o gráfico de violino serve para “desempatar” este contexto, uma vez que o valor de 0.5 apresenta uma

maior distribuição no ranking de 1, passando por um afinilamento no ranking 2. Enquanto o valor de 0.7 mantém-se praticamente constante em todos os rankings, revelando distribuições semelhantes. Assim pode-se concluir que o valor de 0.5 é a opção com maior frequência de ranking 1 e a melhor escolha para drop-out. O optimizer demonstra claramente em ambos subfiguras que a opção Adam é a melhor, pois tem uma classificação mais próxima e com a maior frequência de 1.

Este experimento teve o tempo de duração de 57 minutos e 24 segundos.

A Figura 25 refere-se aos resultados do dataset EPC com elos do tipo Lote, que no contexto EPC é a informação de rastreabilidade de uma matéria-prima e produto. Neste caso já na primeira sub-figura é possível observar um empate quanto as opções de 1 e 2 camadas de pré-processamento. O desempate é feito observando o gráfico de violino, no qual demonstra que apesar das duas opções aparentarem ter a mesma distribuição no ranking 1, a segunda opção possui uma maior frequência para o ranking 2, desta forma ela fica em segundo lugar em termos de frequência. Conclui-se que a primeira opção é a melhor escolha de projeto neste caso.

Figura 25: Resultados para o dataset EPC com conexões do tipo Lote.



Fonte: Elaborado pelo autor

A Subfigura correspondente a função de ativação é semelhante ao encontrado na conexão de Registro. A função ReLU é a melhor escolha.

Para drop-out, no gráfico de barras tem-se as três opções próximas do rank 2, enquanto no gráfico de violino os valores 0.5 e 0.7 possuem distribuições semelhantes para o ranking 1. Levando em consideração ambas as informações fornecidas pelo gráfico, pode-se considerar o valor de 0.7 sendo a melhor escolha.

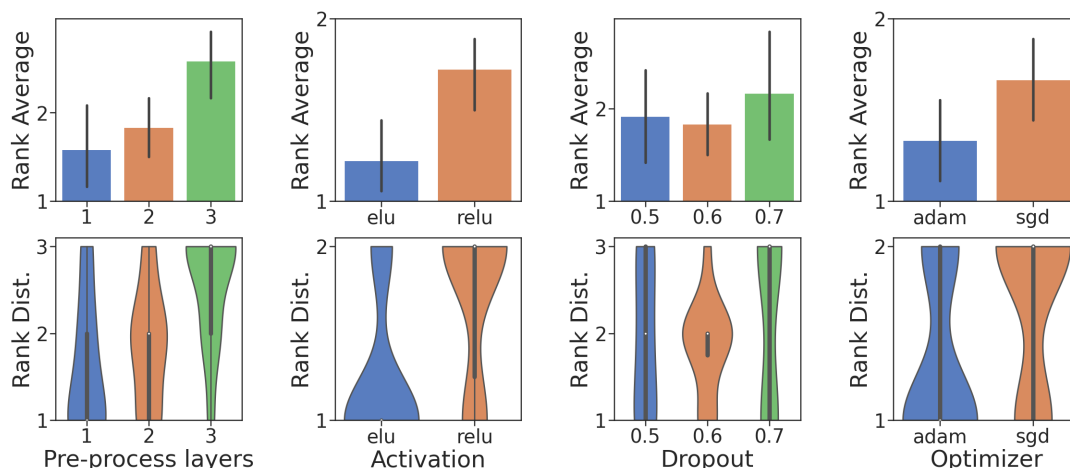
A Subfigura que explora o tipo de otimizador não é tão expressiva quanto ao analisado anteriormente. Porém ainda torna-se claro que a opção Adam está mais próxima e possui mais distribuição no *rank* 1.

Este experimento teve o tempo de duração de 5 horas e 10 minutos, com código executando no Google Colab na máquina disponibilizada.

A Figura 26 mostra os resultados para o grafo com conexão do tipo Corrida (i.e. informação complementar de rastreabilidade). Nela as melhores escolhas tendem a utilizar apenas 1 camada de pré-processamento, a função de ativação ELU, drop-out de 0.6 e otimizador Adam.

Este experimento teve o tempo de duração de 4h50min com execução na máquina disponibilizada pelo Google Colab.

Figura 26: Resultados para o dataset EPC - grafo com elos Corrida.



Fonte: Elaborado pelo autor

Enquanto na Figura 27, para definir a melhor opção de camadas de pré-processamento da rede é preciso combinar a análise das informações de ambos os gráficos, pois apenas o gráfico de barras não demonstra muita diferença entre 1 e 3 camadas. Contudo o gráfico violino demonstra que a terceira opção apresenta maior frequência no *rank* 2, logo escolher utilizar 1 camada apenas demonstra ser a melhor escolha.

Análise semelhante é feita para o valor de drop-out, o qual ao combinar as informações de ambos os gráficos conclui-se que o valor de 0.7 apesar de ser mais próximo de 1, possui uma maior frequência de aparecer em resultados com ranking 2 o que faz com que o valor de 0.6 pareça mais promissor de ser utilizado.

A função de ativação que apresenta resultados com *rank* 1 mais frequentemente é a ReLU. Enquanto o SGD é a melhor escolha de otimizador.

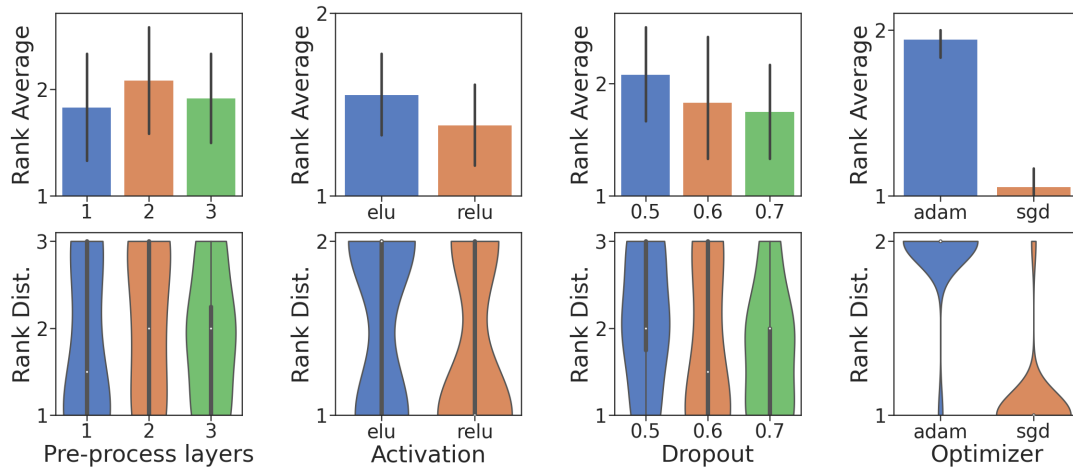
Este experimento teve o tempo de duração de 2 horas e 52 minutos.

Por fim, a Figura 28 refere-se aos resultados do tipo de conexão Certificado, as quais apresenta escolhas aparentemente bem definidas de acordo com ambos os gráficos, exceto para as opções de drop-out.

Para este experimento as melhores configurações são: 2 camadas de pré-processamento, função de ativação ReLU e SGD. O valor de drop-out apresentou resultados semelhantes, permanecendo entre 0.5 e 0.6. O valor de 0.5 é mais próximo do *rank* 1, enquanto 0.6 possui menor frequência em configurações com rank 2.

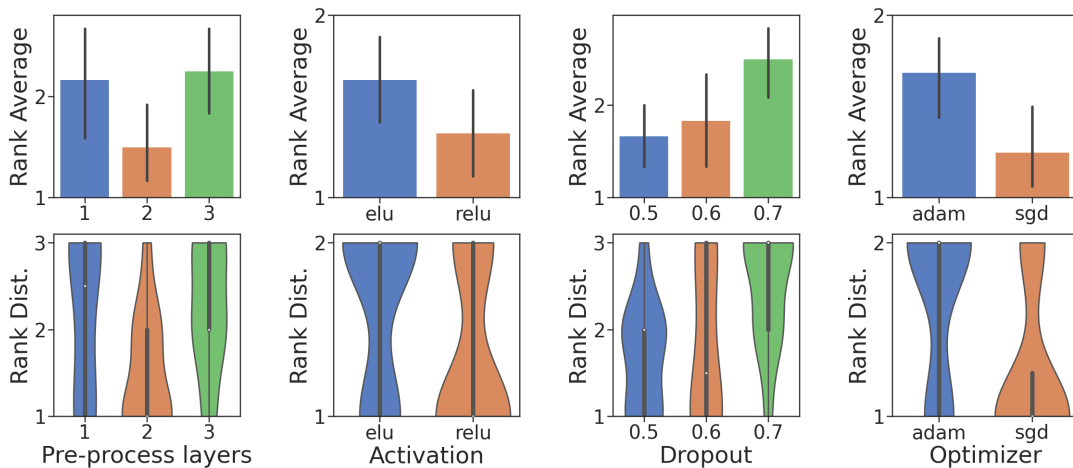
Este experimento teve o tempo de duração de 19 horas e 7 minutos.

Figura 27: Resultados para o dataset EPC com conexões do tipo Fornecedor.



Fonte: Elaborado pelo autor

Figura 28: Resultados para o dataset EPC com conexões do tipo Certificado.



Fonte: Elaborado pelo autor

A Tabela 18, contém as configurações que retornaram os melhores valores de acurácia para o conjunto de treino em cada dataset.

Os resultados na forma de gráficos que foram gerados pelo GraphGym para os datasets Cora, Citeseer e Pubmed encontram-se no Anexo B. Já o tempo de execução de seus experimentos são 1h17min, 1h03min e 7h20min, respectivamente.

Somando o tempo de duração de todos os experimentos, levou-se 42 horas e 36 minutos para concluir todos experimentos. O GraphGym explorou um universo extenso de configurações, com validações de modelos mediante a variação de vários parâmetros de projeto (*design space*).

7.2.2 Discussão

Os resultados de desempenho utilizando as variantes de grafos do *dataset* EPC ficaram abaixo da expectativa para um modelo GCN. Este modelo obteve desempenho razoável

Tabela 18: Configurações com os melhores resultados encontrados no conjunto de teste para cada dataset.

Dataset	Conexão	l_pre	Act	D_O	Optim	Epoch	Acurácia	Loss
EPC	Certificado	2	relu	0.7	sgd	49	0.235 ± 0.014	1.615
	Corrida	1	relu	0.6	sgd	49	0.263 ± 0.019	1.601
	Fornecedor	1	elu	0.5	sgd	49	0.233 ± 0.009	1.608
	Lote	2	elu	0.7	adam	49	0.271 ± 0.006	1.705
	Registro	2	relu	0.7	adam	49	0.276 ± 0.022	1.587
Cora	-	3	elu	0.6	adam	49	0.885 ± 0.008	0.419
Citeseer	-	3	elu	0.7	adam	49	0.770 ± 0.008	0.789
Pubmed	-	1	relu	0.5	adam	49	0.829 ± 0.014	0.522

durante os experimentos de calibração, apresentando uma acurácia mínima de 64% nos experimentos com hiperparâmetros e *datasets*. Para verificar esses resultados, executou-se uma nova bateria de experimentos utilizando a plataforma GraphGym.

Levando-se em conta o observado no GraphGym, percebe-se que nem sempre a escolha esperada pode ser a melhor. Por exemplo, o *dataset* EPC com conexões do tipo Fornecedor mostrou-se ter resultados melhores com função de ativação ReLU. Entretanto, o maior valor de acurácia foi obtido com uma configuração que utiliza a função ELU. Este fato se deve pois os gráficos apresentam a classificação média das configurações, ou seja, eles entregam as configurações que tendem a apresentar melhor resultados.

Os valores de acurácia e *loss* entregues pela plataforma para todas as variações do dataset EPC são condizentes com os experimentos realizados anteriormente. Portanto, pode-se confirmar os resultados anteriores.

Ainda segundo o GraphGym, para o dataset EPC o melhor resultado dentre as opções foi 0.276 ± 0.022 utilizando o grafo com conexão de Registro, seguido por 0.271 ± 0.006 para conexão de Lote. É importante ressaltar que ambas opções representam as variações do *dataset* com os menores números de ligações.

Destes experimentos, o ponto importante é a discrepância de resultados obtidos pelas variações do *dataset* EPC em comparação aos resultados obtidos com os *datasets* de *benchmarking* na Seção 7.1.

O *dataset* EPC foi considerado como os dados “fixos” de entrada. Os parâmetros do modelo foram variados iterativamente e validados durante o processo que chamou-se de calibração, diante dos resultados obtidos observa-se que talvez esta abordagem pode não ter sido a mais apropriada para lidar com o ruído nos dados. Esta metodologia, denominada de I.A. Centrada no Modelo (*model-centric*), permite obter escala, padronizar trabalho e automatizar o fluxo por software. A I.A. Centrada no Modelo recentemente sofreu críticas (Andrew Ng 2021) pois a “tunagem desenfreada do modelo” em busca

de desempenho, de certa forma negligência a natureza dos dados. Torna-se necessário garantir dados de qualidade em todas as etapas do projeto de aprendizado de máquina.

Diante disto decidiu-se por retornar aos dados e realizar um processo de exploração mais aprofundado das características dos *datasets* utilizados neste trabalho. O intuito é encontrar as possíveis diferenças estruturais, as quais possam auxiliar o entendimento do porquê há uma diferença de desempenho tão acentuada entre os resultados de GNNs sobre os dados EPC e os *datasets* Cora, Citeseer e Pubmed.

8 EXPLORANDO OS DADOS

Neste capítulo é apresentado um processo de exploração dos dados a fim de encontrar as diferenças estruturais entre os *datasets* de *benchmarking* e as variações do dataset EPC. São apresentados o desenho da estrutura de cada *dataset*, como também análises para melhor caracterização dos grafos, explorando métricas no nível de nodo e de conexão. Por fim, é feita uma breve discussão seguida pela proposta do *Data Space*, como uma etapa adicional para trabalhar os dados como parte do ciclo de vida de um projeto de aplicação com redes neurais em grafos.

A metodologia adotada neste trabalho, e considerada “padrão” na maioria dos trabalhos relacionados, focou inicialmente no modelo como se fosse uma caixa-preta que recebe dados de entrada, otimiza parâmetros internos da caixa e gera uma saída.

Este capítulo tornou-se necessário à medida que o resultado esperado de acurácia não se materializou para o novo modelo desenvolvido. Portanto, esta discussão passou a ser necessária, e agora alinha-se de forma mais ampla à tendência da metodologia centrada no dado (*A.I. Data-Centric*), em que os dados e modelos podem variar nas etapas de projeto.

8.1 Visualização dos Grafos

As Figuras 29 e 30 mostram a representação visual dos *datasets* Cora e Citeseer. Cada cor representa a classe à qual aquele nodo pertence.

Visualmente, nota-se que ambos *datasets* possuem uma concentração de nodos de diferentes classes no centro da imagem, com pequenas comunidades ao redor. Destaca-se também que o Citeseer demonstra possuir um grande número de comunidades isoladas.

A Figura 31 apresenta a visualização do *dataset* Pubmed. A análise deste grafo mostra-se difícil por conta da grande concentração de nodos no centro.

Por outro lado, as Figuras 32, 33, 34, 35 e 36 referem-se a visualização dos diferentes grafos do *dataset* EPC. Observa-se que os grafos possuem uma aparência mais conectada, sem a concentração central de nodos observada nos *datasets* de *benchmarking*.

Em alguns dos grafos, como por exemplo no grafo baseado na conexão Fornecedor, é possível verificar a existência de comunidades bem definidas, mas que ainda assim se

mantém conectadas umas com as outras.

Nota-se também, que em todas as variações existem alguns nodos isolados sem nenhum tipo de conexão. Essa situação ocorre principalmente no *dataset* com conexão do tipo Registro.

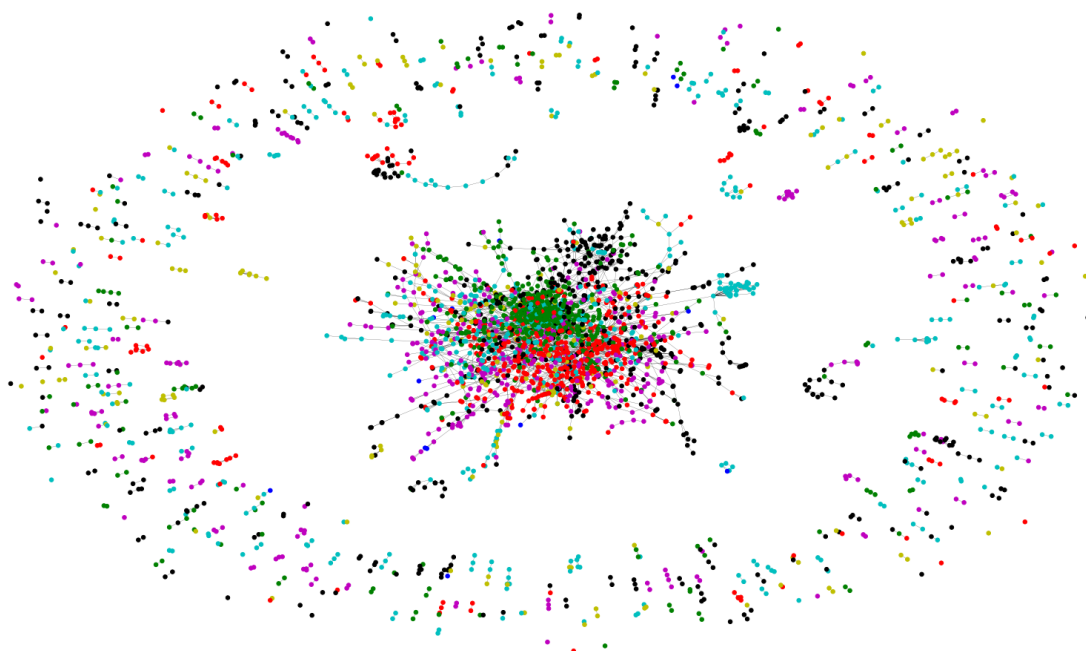
O *dataset* Pubmed tem uma grande quantidade de nodos, praticamente 10 vezes maior que os outros *datasets*. Neste caso, optou-se por deixá-lo de fora das análises, visto que o mesmo poderia criar uma tendência de *outlier* nos resultados.

Figura 29: Grafo do dataset Cora.



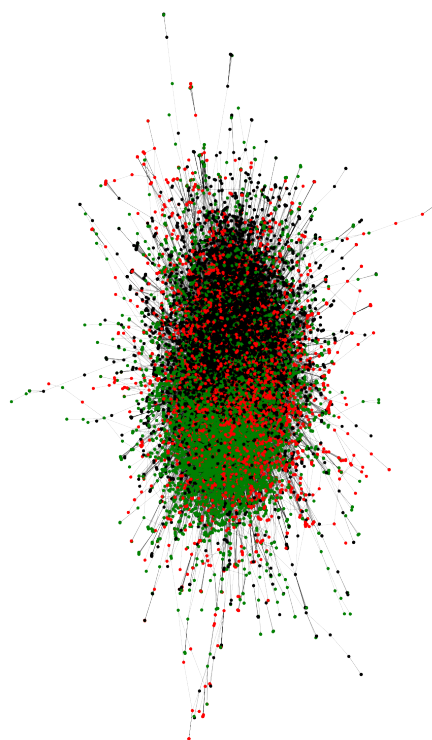
Fonte: Elaborado pelo autor

Figura 30: Grafo do dataset Citeseer.



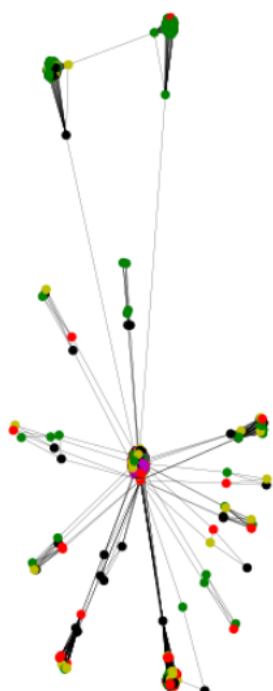
Fonte: Elaborado pelo autor

Figura 31: Grafo do dataset Pubmed.



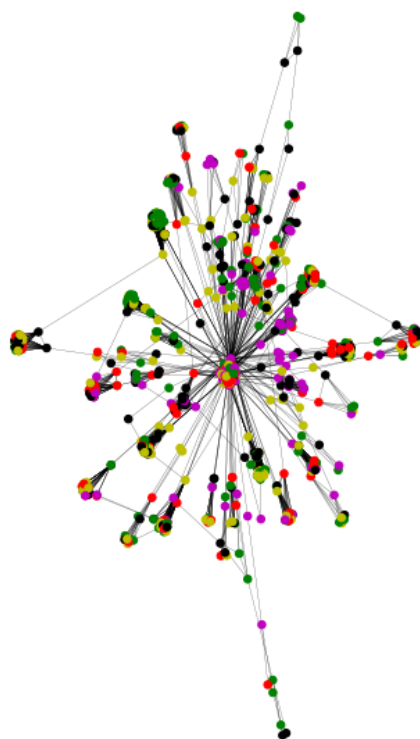
Fonte: Elaborado pelo autor

Figura 32: Grafo do dataset EPC com conexão do tipo Certificado.



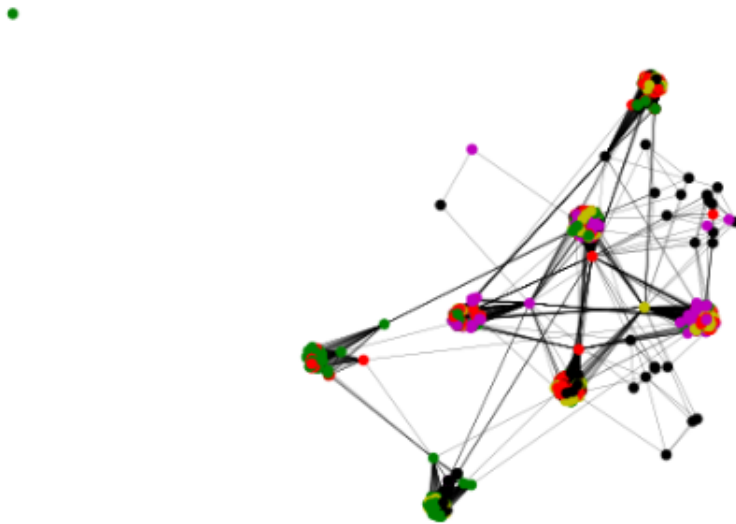
Fonte: Elaborado pelo autor

Figura 33: Grafo do dataset EPC com conexão do tipo Corrida.



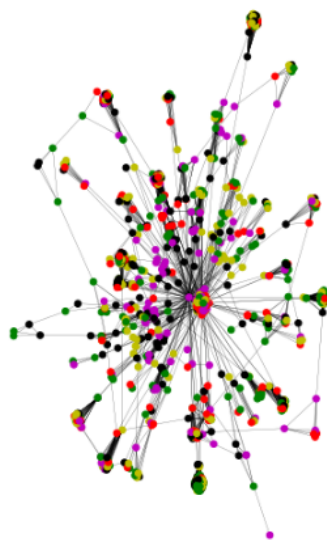
Fonte: Elaborado pelo autor

Figura 34: Grafo do dataset EPC com conexão do tipo Fornecedor.



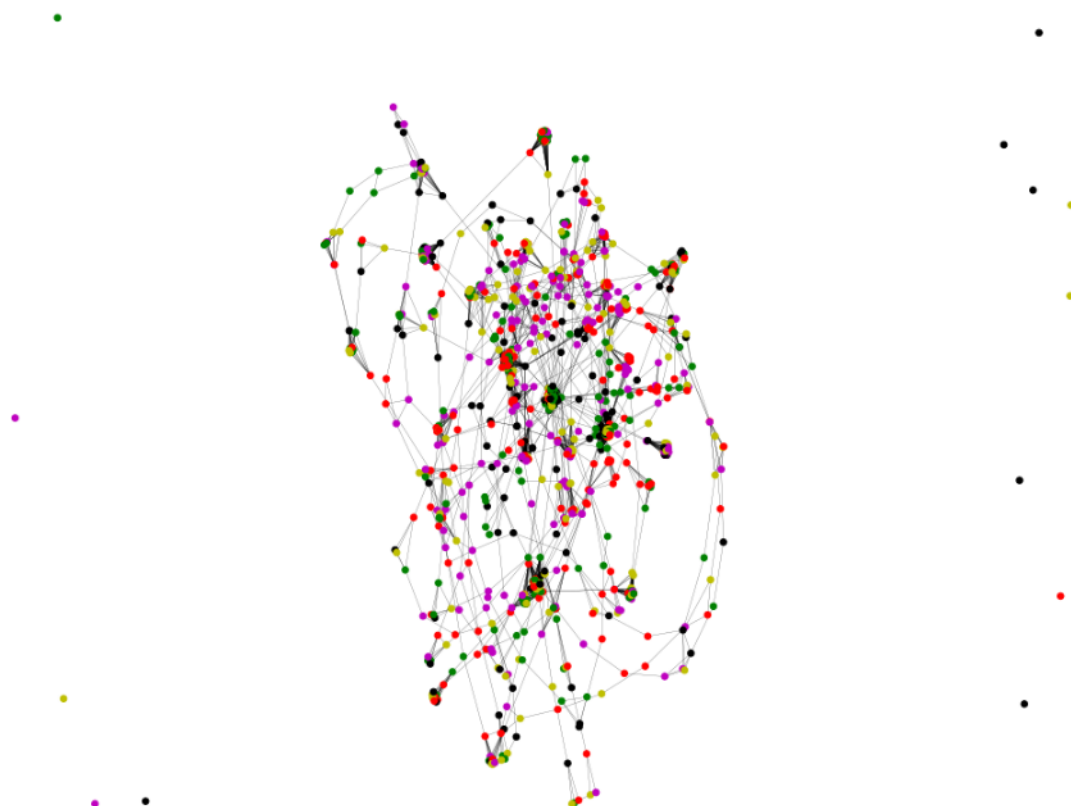
Fonte: Elaborado pelo autor

Figura 35: Grafo do dataset EPC com conexão do tipo Lote.



Fonte: Elaborado pelo autor

Figura 36: Grafo do dataset EPC com conexão do tipo Registro.



Fonte: Elaborado pelo autor

8.2 Análise dos Grafos

Este estudo teve por início avaliar um conjunto de métricas de caracterização de propriedades e topologias dos grafos. Primeiramente, analisou-se o grau médio dos nodos (*node degree*), uma medida de centralidade básica que indica a importância do nodo para o grafo e refere-se à quantidade de conexões que o nodo apresenta. Neste caso foram contabilizadas tanto as conexões que chegam no nodo (*in-degree*) quanto às ligações que saem do nodo (*out-degree*).

Conforme a Tabela 19 percebe-se que, com exceção do Cora, os *datasets* tem grau mínimo de 0. Este resultado indica que existem nodos isolados. O grau máximo de nodo no Cora tem valor próximo ao encontrado no EPC (Registro). Porém, ao observar o grau médio constata-se que os *datasets* de *benchmarking* possuem em média menos de 5 conexões por nodo. Em contraste, os diferentes tipos de grafos EPC, com exceção da conexão Registro, possuem grau médio superior a 100 nodos.

É interessante observar que no grafo EPC-Certificado tem-se o grau máximo de nodo de 954. Desta forma, considerando o número total de nodos neste *dataset*, pode-se verificar que um nodo conecta-se com 83.68% dos nodos no *dataset*.

Tabela 19: Valores da métrica degree.

Dataset	Conexão	Qtd nodos	Degree Min	Degree Max	Degree médio
EPC	Certificado	1140	0	954	751.001
	Corrida	1140	0	518	141.605
	Fornecedor	1140	0	476	242.968
	Lote	1140	0	465	111.975
	Registro	1140	0	186	36.061
Cora	-	2708	1	168	3.898
Citeseer	-	3327	0	99	2.736

A Tabela 20 mostra os resultados de agrupamento médio de nodos (*Average Clustering*). Esta métrica indica o grau com que os nodos tendem a agrupar em comunidades. O cálculo utiliza a fração de todos os triângulos possíveis presentes no Grafo. A Tabela também mostra os resultados para a transitividade de agrupamento (*Cluster Transitivity*), definida como o coeficiente de agrupamento médio dos nodos e suas conexões dentro da própria comunidade.

Novamente há diferenças perceptíveis entre as métricas para os *datasets*. O *dataset* EPC (e variantes) possui valores elevados tanto para transitividade quanto para agrupamento médio. Os *datasets* Cora e Citeseer tem valores menores para as duas métricas.

No caso do EPC-Certificado tem-se um agrupamento médio e um transitividade próximos de 1. Isso confirma os resultados de que este grafo é praticamente conectado.

Tabela 20: Valores das métricas Cluster Transitivity e Average Clustering.

Dataset	Conexão	Transitivity	Avg Clustering
EPC	Certificado	0.999	0.981
	Corrida	0.995	0.924
	Fornecedor	0.997	0.978
	Lote	0.994	0.896
	Registro	0.993	0.784
Cora	-	0.093	0.240
Citeseer	-	0.130	0.141

Ao analisar-se métricas referente ao agrupamento em grafos, é comum também avaliar-se o número de “cliques” no grafo. Cliques em um grafo correspondem a sub-grafos nos quais seus nodos são totalmente conectados entre si, representando a parte mais densa. Dependendo do tamanho do grafo, a métrica maior número de cliques (*Largest Clique number*) pode ser computacionalmente difícil de calcular, uma vez que é considerada um problema *NP-hard* (Pardalos & Xue 1994).

Em situações práticas, usualmente o cálculo de k -cores é utilizado como uma alternativa. A métrica K-cores remove todos os nodos com grau menor do que um valor K , retornando subgrafos que contenham nodos com K ligações ou mais.

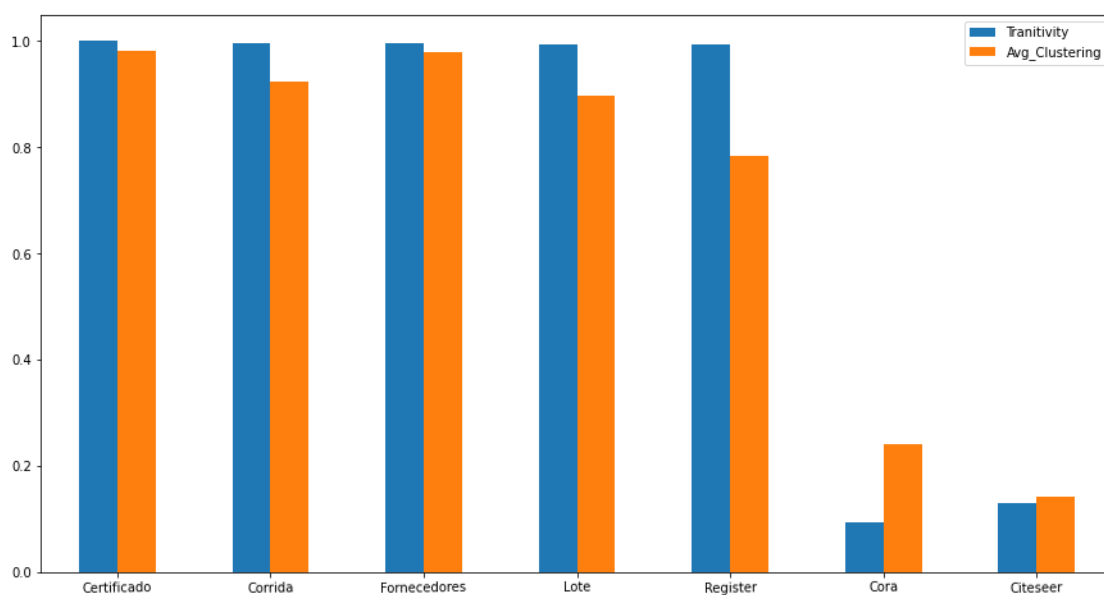
De acordo com a Tabela 21, os *datasets* de *benchmarking* possuem valores inferiores se comparados com os encontrados para as variações do EPC. Resultado coerente que vai de encontro com as métricas já mencionadas.

Tabela 21: Valores das métricas K-cores.

Dataset	Conexão	K-core
EPC	Certificado	922
	Corrida	376
	Fornecedor	408
	Lote	330
	Registro	163
Cora	-	174
Citeseer	-	18

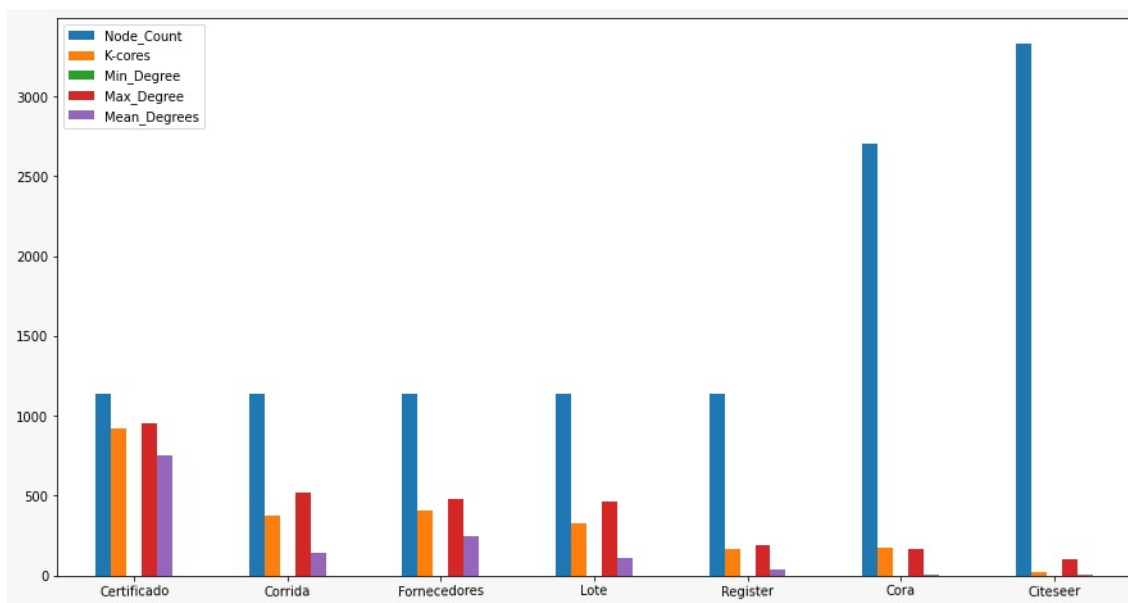
As Figuras 37 e 38 resumem os resultados encontrados utilizando as métricas selecionadas.

Figura 37: Métricas Transitivity e Average Clustering.



Fonte: Elaborado pelo autor

Figura 38: Métricas degree e k-cores.



Fonte: Elaborado pelo autor

8.3 Características dos *datasets*

Complementando as análises com algumas métricas em grafos, esta seção explora as características de construção do *dataset*.

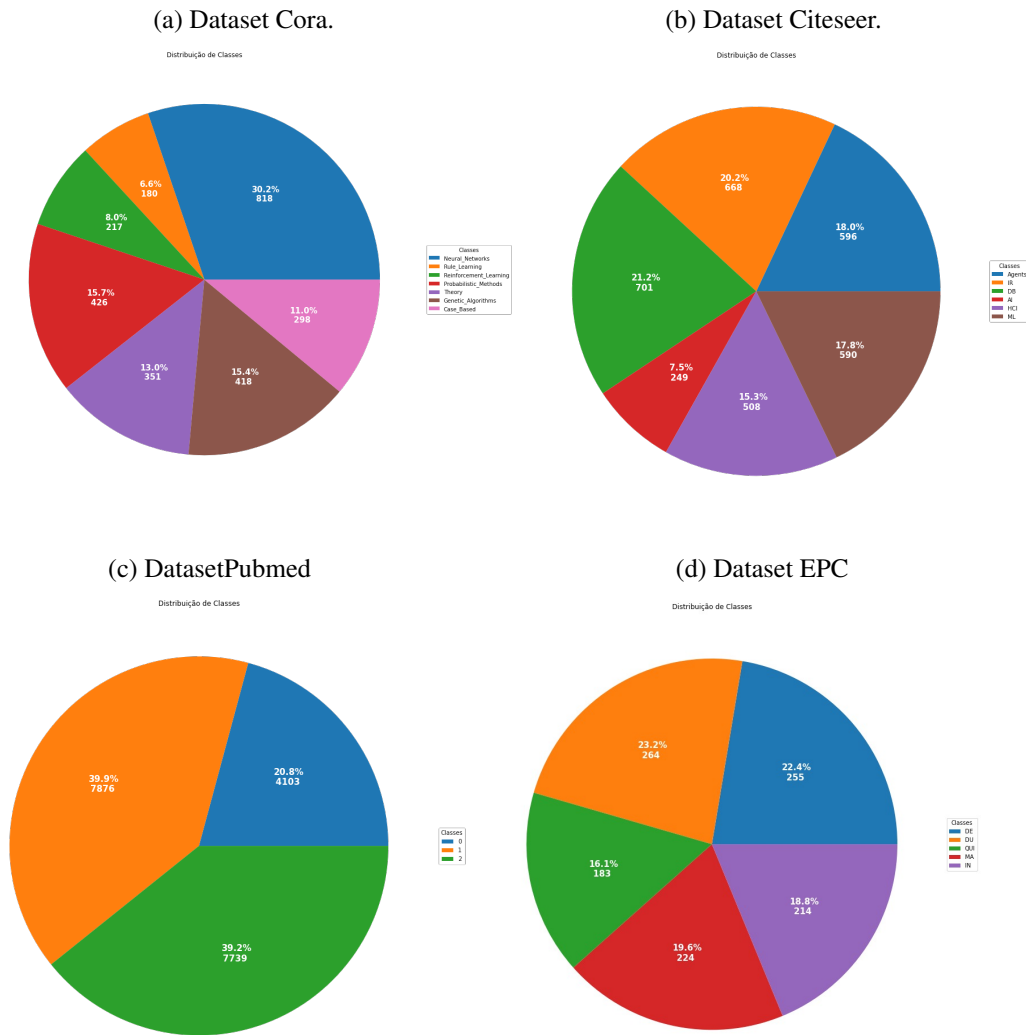
Cada nodo (ou documento) possui um conjunto de *features* específico dentre todas as disponíveis. Embora dois nodos pertençam à mesma classe, não necessariamente possuem as mesmas *features*. Por exemplo, no *dataset* EPC há documentos da mesma classe com testes feitos por fornecedores diferentes, em que cada empresa tem seu próprio *layout* de documento. As listas de *features* destes documentos devem ser diferentes por conta de diferenças de campos e dados preenchidos.

A fim de investigar essa diferença foram separadas amostras contendo 10 nodos de mesma classe no Cora e EPC-Registro. No caso do Cora encontrou-se que em média 97% das *features* encontradas nas amostras são iguais. No caso do EPC 86% das *features* são iguais.

A próxima análise deu-se em levantar a quantidade de relações existentes entre nós com a mesma classe (intra-relações) e entre nodos de classes diferentes (inter-relações). No *dataset* Cora foram encontradas 4418 intra-relações e 1011 inter-relações. No caso do EPC-Registro foram encontradas 11246 intra-relações e 29352 inter-relações.

Por fim foi feito o levantamento dos percentuais das classes em cada *dataset*, conforme mostra a Figura 39.

Figura 39: Proporção de classes em cada dataset.



Fonte: Elaborado pelo autor

8.4 Discussão

A proporção de cada classe nos *datasets* não trouxe grandes revelações que possam auxiliar na justificativa do desempenho não satisfatório do modelo GCN EPC. Isto se deve ao fato que, na construção do EPC a escolha final das classes foi feita com base nas frações encontradas no *dataset* Cora para manter equilíbrio entre as classes.

A conectividade no grafo é possivelmente o aspecto que mais apresentou diferença entre os *datasets*. Os grafos do EPC são altamente conectados conforme indicou as métricas de agrupamento (*clustering*). O grafo EPC-Certificado apresenta o maior valor de grau dentre os outros grafos, conectando 83.68% do grafo. Considerando que este grafo (EPC-Certificado) apresentou a pior acurácia nos experimentos realizados, observa-se que o desempenho do modelo GCN pode ser impactado por grafos altamente conectados.

Segundo a análise das *features*, percebeu-se que o EPC ao ser comparado com o Cora,

apresenta uma variação de quase cinco vezes maior quanto ao conjunto de *features* para nodos de mesma classe. Isto indica que há a possibilidade de uma das razões para o qual modelo não aprende seja devido à estas diferenças. Uma maneira para tentar contornar essa situação seria incrementar a quantidade de nodos, desta forma o modelo estaria exposto de forma mais recorrente à estas diferenças o que o ajudaria a encontrar o padrão correto para a classificação.

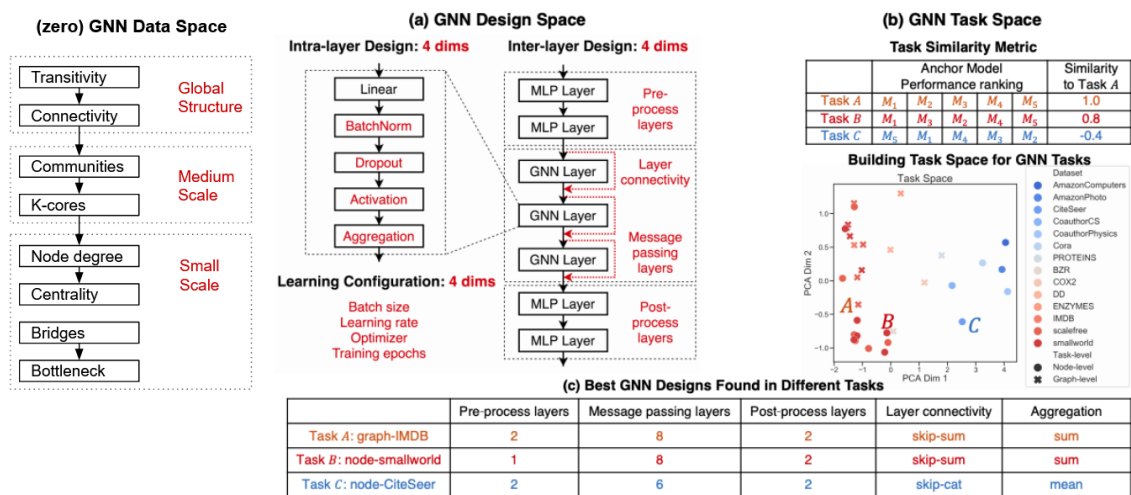
De acordo com as intra e inter-relações, nota-se que o *dataset* de *benchmarking* contém 81% de ligações entre nodos de mesma classe enquanto o EPC contém 27% de ligações entre nodos de mesma classe. Deste resultado, pode-se concluir que o modelo GCN tem um melhor desempenho quando utilizado em dados que contenham ligações diretas, de nodos semelhantes, do que quando aplicado em dados com ligações indiretas.

É possível então verificar que as conexões, tanto quantitativamente como qualitativamente, aparentam ter relação com o desempenho observado nos modelos GCNs criados e validados neste trabalho. Este resultado pode indicar também que o uso de uma abordagem de aprendizado semi-supervisionado com GNNs é favorável para aplicações específicas. Entretanto, os dados de entrada, seja na forma de grafos ou em outra estrutura representacional, precisam respeitar certas propriedades. Por exemplo, é importante estabelecer o número de ligações por nodo, formação de agrupamentos, conectividade do grafo e outras. A oportunidade encontra-se na descobertas destas propriedades para propor melhorias nas mesmas que retro-alimentam o processo de criação e validação de modelos. Pode-se então considerar este processo como o equivalente na metodologia GNN às transformações já utilizadas em dados de imagens nas aplicações de visão computacional (ex. *data augmentation*, *cleaning*, *adversial networks*, etc).

Por todos estes aspectos, com o propósito de evitar situações como a ocorrida durante a pesquisa, propõe-se a adição da lista de análises apresentadas neste capítulo como o primeiro passo para criar-se uma metodologia GNN centrada no dado. Todas as análises realizadas foram implementadas junto ao código do GraphGym. De certa forma, tais métricas adicionam mais espaço na ferramenta, o qual foi denominada de *GNN Data Space* que compreende o conjunto mínimo de métricas como node degree, average clustering, transitivity e k-core.

A Figura 40 mostra onde o espaço de dados (*Data Space*) situa-se na metodologia de criação de modelos do GraphGym.

Figura 40: Proposta do GNN Data Space.



Fonte: Elaborado pelo autor

9 CONCLUSÃO

A abundância de dados disponíveis atualmente, ao mesmo tempo que facilita, também acaba por dificultar os processos de classificação de documentos. Dados de diversas fontes tendem à heterogeneidade, principalmente quando existem relações explícitas e implícitas entre os mesmos. A diversidade dos dados pode influenciar a forma como as informações são adquiridas e interpretadas.

Neste contexto, o presente trabalho realizou um estudo comparativo que avaliou o desempenho de métodos de redes neurais em grafos (GNNs) para problemática de classificação automática de documentos. Os métodos utilizam *features* extraídas do conteúdo textual de cada documento, combinadas com uma estrutura de grafo representando os relacionamentos inter-documentos.

O treinamento de um modelo com desempenho satisfatório mostrou-se desafiador com o aprendizado de máquina semi-supervisionado, o qual representa o contexto mais próximo da realidade, em que um *dataset* rotulado é uma exceção, ou seja, há a disponibilidade de poucas instâncias rotuladas e uma abundância de instâncias não rotuladas.

Primeiramente, a fim de auxiliar no entendimento do trabalho desenvolvido, foi apresentada uma breve fundamentação teórica, introduzindo conceitos básicos de teoria de grafos e dos métodos considerados como *baseline*, a saber K-nearest-neighbor e redes neurais artificiais. Ainda neste capítulo, são trazidos a teoria e o princípio de funcionamento de *Graph Neural Networks* e dos modelos escolhidos. Após, no capítulo de Revisão Bibliográfica foram elencados projetos que fazem uso da representação de dados através da estrutura de grafos, somados à técnicas de inteligência artificial para resolver problemas análogos ao aqui em foco. Ao longo da pesquisa por projetos dessa natureza, percebeu-se a eficiência deste tipo de abordagem em áreas de importância social e industrial, assim justificando a escolha das mesmas.

Também foram apresentadas a metodologia geral proposta, compreendendo as etapas de preparação dos dados, escolha dos modelos utilizados, e a validação por meio de trabalho experimental.

Na primeira parte dos experimentos (calibração), verificou-se que os modelos *baseline* (KNN e RNA) apresentaram acurácias inferiores aos modelos de redes neurais em

grafos (GNNs). Esse resultado promissor indicou a relevância da informação de relacionamentos (ex. nodos vizinhos no grafo) para o aprendizado sobre um número limitado de instâncias rotuladas. A escolha e uso dos *datasets* públicos com dados de artigos e citações apresentou-se como crucial nesta etapa.

No decorrer destes experimentos, tornou-se clara a dificuldade do Cheb em atingir a acurácia obtida na validação dos outros modelos GNNs. Este resultado já era previsto, pois por ser um dos trabalhos pioneiros na área, pode-se dizer que o mesmo serviu como uma espécie de base para o desenvolvimento dos métodos subsequentes. O GCN, por exemplo, é uma versão de primeira ordem do Cheb.

O inverso pode-se afirmar para o ARMA, método mais recente dos avaliados. O treinamento desta rede exigiu duas vezes mais épocas para atingir resultados de acurácia compatíveis com os outros modelos. O pequeno ganho em desempenho que o ARMA atingiu em certas configurações pode não justificar o provável custo computacional (i.e. tempo de execução). No final, o modelo GCN destacou-se como o mais consistente em termos de acurácia, nas configurações de parâmetros utilizadas.

Para dar prosseguimento ao estudo comparativo, criou-se um *dataset* próprio, de forma a representar o contexto da aplicação real em desenvolvimento no Projeto ATT. De forma a manter a consistência e reprodutibilidade, utilizou-se a metodologia de criação do Cora, resultando em uma estrutura de dados compatível com os *datasets* públicos de citações.

Os resultados obtidos com a validação do modelo GCN treinado com o *dataset* EPC mostraram-se insatisfatórios, uma vez que a acurácia média foi três vezes menores que as obtidas com os modelo GCN nos *datasets* de *benchmarking*. Pelas curvas de *loss*, observou-se que o modelo GCN teve dificuldade em convergir e, com isto, prejudicou a sua capacidade de generalização para o conjunto de teste.

Uma segunda abordagem de validação consistiu em utilizar a ferramenta GraphGym, desenvolvida especificamente para a explorar o espaço de projeto de redes neurais em grafos (*design space*). Os resultados insatisfatórios de acurácia se confirmaram. Os experimentos com 36 combinações de projeto para o GCN mostraram a ineficácia dos modelos criados em classificar documentos EPC.

Este resultado negativo motivou a revisitação aos dados originais. A etapa de exploração de dados demonstrou a importância de dedicar tempo de P&D analisando, visualizando, transformando, limpando e “curando” os *datasets*. Este projeto, de certa forma, seguiu a metodologia padrão da “I.A centrada no modelo”, o qual não se mostrou vantajosa.

O retorno aos dados contempla a conhecida, e cada vez mais esquecida, “fração de etapas” em que 80% do tempo em projetos de aprendizado de máquina é dedicado à preparação de dados.

Observou-se neste análise dos dados que o tipo e número de conexões nos grafos

do *dataset* EPC podem explicar o baixo desempenho do modelo GCN. As métricas de grafo computadas trazem informações importantes sobre a natureza dos *datasets* utilizados. Salienta-se que o Cora, Citeseer e Pubmed possuem um menor número de conexões entre nodos. A fração de conexões é maior para os nodos da mesma classe (rótulo). Por exemplo, apenas 19% dos nodos conectados no Cora não pertencem à mesma classe.

Logo pode-se indicar que os modelos GNN apresentam um desempenho melhor em dados (i.e. grafos) que não sejam altamente conectados e que apresentem uma maior quantidade de ligações diretas do que indiretas.

Esses resultados demonstram então que em abordagens que utilizem GNN, antes de prosseguir com os passos metodológicos descritos no GraphGym, existe a oportunidade de se utilizar um passo importante de exploração dos dados. Este passo, proposto com o nome de *GNN Data Space* permite caracterizar os *datasets* utilizando um conjunto de métricas e visualização em grafos. Apesar de simplificada, esta etapa metodológica alinha-se ao conceito da I.A. centrada no dado (*Data-centric A.I.*).

Por outro lado, existe uma alternativa para utilizar-se GNNs com o *dataset* EPC. Como demonstrado na Figura 49, encontrada no Anexo B, o modelo GCN consegue entregar acurácias satisfatórias de até 60% com o *dataset* EPC-Registro. Neste cenário, utilizou-se o fracionamento dos dados na proporção convencional de 80% e 20%, para treino e teste, respectivamente. Desta forma demonstrando que, dependendo da disponibilidade e características dos dados rotulados, ainda há vantagens ao se utilizar abordagens de aprendizado supervisionado se comparada com as abordagens de aprendizado semi-supervisionado.

O debate é amplo neste caso e remete à situação de escolha do tipo de aprendizado considerando o *trade-off* entre acurácia, disponibilidade de dados, e obviamente o custo do trabalho de anotação destes dados.

Em síntese, o presente trabalho traz indícios de resultados experimentais de que modelos de GNN com aprendizado semi-supervisionado podem ser escolhas de projeto problemáticas quando os dados da aplicação tem alto grau de relacionamentos e estes elos advêm de relações indiretas. Para contornar este tipo de situação recomenda-se fazer o uso do que chamou-se de *GNN Data Space* para caracterizar de forma mais precisa os dados de entrada.

9.1 Contribuições

As principais contribuições deste trabalho são a seguir resumidas:

- Entendimento acerca do desempenho, em termos de acurácia, de classificadores GNNs, a partir da comparação com classificadores considerados clássicos na literatura.

- Desenvolvimento de um *dataset* EPC para representar documentos e seus relacionamentos, por meio de um conjunto de *features* e um grafo, sendo tais documentos comumente encontrados na área de engenharia da indústria de óleo e gás.
- Resultados do trabalho experimental, que adotou o modelo treinado com o *dataset* EPC e os parâmetros calibrados, mostraram a necessidade de um melhor entendimento acerca da estrutura dos grafos envolvidos.
- Um *framework* de exploração do espaço de dados, denominado *Data Space*, é proposto como o primeiro passo para a abordagem do problema de classificação com aprendizado semi-supervisionado e GNNs, evitando, desta forma, a “armadilha” do fluxo de trabalho conhecido por *Black Box*.

9.2 Recomendações para Trabalhos Futuros

Como trabalhos futuros, para aprimorar a qualidade do *dataset* desenvolvido sugere-se gerar novos arquivos do tipo *cite* para o EPC. Desta vez pretende-se criar as conexões por meio de combinações dos elos de rastreio, formando duplas de rastreabilidade e desta forma criando ligações compostas. Através deste processo pode ser possível conseguir um menor número de ligações, sem perder a importância da relação de conectividade entre os documentos. Esta observação, obtida durante a etapa de exploração dos dados, permitirá novos experimentos para validar a melhoria do desempenho de modelos perante a novos tipos de ligações. Ainda quanto ao *dataset*, sugere-se trocar a representação das *features* de one-hot encoding para outro tipo de *word embedding*, como por exemplo *word2vec*.

Sugere-se também aplicar modelos que explorem o contexto das ligações (*Relational GCN*), ou seja, que atribuam pesos e importância diferentes a partir do tipo da ligação. Desta forma, esta abordagem possibilita a utilização em conjunto de dados com ligações simples já existentes no EPC, como também explorar o novo tipo de ligação sugerida.

Recomenda-se também incorporar mais dados ao *dataset* próprio, adicionando *datatables* de fornecedores primários diversificados, desta forma introduzindo ainda mais diversidade para tornar os dados mais ricos.

Por fim, tem-se a proposta de estruturar as métricas, implementar e validar o *framework* correspondente ao *GNN Data Space* mencionado no Capítulo 8, conforme proposto na Figura 40. Esta nova camada de verificação no fluxo de projeto pode contribuir com o *design* de novas aplicações baseadas em redes neurais em grafos. Desta forma, acredita-se que o *Data Space* diminua o tempo de projeto e evite algumas armadilhas da abordagem “mais mecânica e automatizada” da I.A centrada em modelo.

REFERÊNCIAS

- Abdi, H. (2007), ‘The kendall rank correlation coefficient’, *Encyclopedia of Measurement and Statistics*. Sage, Thousand Oaks, CA pp. 508–510.
- Ahmed, C., ElKorany, A. & Bahgat, R. (2016), ‘A supervised learning approach to link prediction in twitter’, *Social Network Analysis and Mining* **6**(1), 24.
- Amor, R. & Clift, M. (1997), Documents as an enabling mechanism for concurrent engineering in the construction industry, in ‘1st international conference on Concurrent Engineering in Construction, CEC’, Vol. 97.
- Andrew Ng (2021), ‘Mlops: From model-centric to data-centric ai’, <https://www.deeplearning.ai/wp-content/uploads/2021/06/MLOps-From-Model-centric-to-Data-centric-AI.pdf>. Acessado em: 22 de Agosto de 2021.
- Ashburn, T. T. & Thor, K. B. (2004), ‘Drug repositioning: identifying and developing new uses for existing drugs’, *Nature reviews Drug discovery* **3**(8), 673.
- Bianchi, F. M., Grattarola, D., Alippi, C. & Livi, L. (2019), ‘Graph neural networks with convolutional arma filters’, *arXiv preprint arXiv:1901.01343* .
- Binoti, D. H. B., da Silva Binoti, M. L. M. & Leite, H. G. (2014), ‘Configuração de redes neurais artificiais para estimação do volume de árvores’, *Revista Ciência da Madeira (Brazilian Journal of Wood Science)* **5**(1), 10–12953.
- Bondy, J. A., Murty, U. S. R. et al. (1976), *Graph theory with applications*, Vol. 290, Macmillan London.
- Bruna, J., Zaremba, W., Szlam, A. & LeCun, Y. (2013), ‘Spectral networks and locally connected networks on graphs’, *arXiv preprint arXiv:1312.6203* .
- Chaparro, C. & Eberle, W. (2015), Detecting anomalies in mobile telecommunication networks using a graph based approach, in ‘The Twenty-Eighth International Flairs Conference’.

- Cover, T. & Hart, P. (1967), ‘Nearest neighbor pattern classification’, *IEEE transactions on information theory* **13**(1), 21–27.
- Crichton, G., Guo, Y., Pyysalo, S. & Korhonen, A. (2018), ‘Neural networks for link prediction in realistic biomedical graphs: a multi-dimensional evaluation of graph embedding-based approaches’, *BMC bioinformatics* **19**(1), 176.
- Dahl, C. M. & Sørensen, E. N. (2019), ‘Document digitization and machine learning’.
- Defferrard, M., Bresson, X. & Vandergheynst, P. (2016), Convolutional neural networks on graphs with fast localized spectral filtering, in ‘Advances in neural information processing systems’, pp. 3844–3852.
- Deo, N. (2017), *Graph theory with applications to engineering and computer science*, Courier Dover Publications.
- Ding, H., Takigawa, I., Mamitsuka, H. & Zhu, S. (2013), ‘Similarity-based machine learning methods for predicting drug–target interactions: a brief review’, *Briefings in bioinformatics* **15**(5), 734–747.
- Eto, M. (2019), ‘Extended co-citation search: Graph-based document retrieval on a co-citation network containing citation context information’, *Information Processing & Management* **56**(6), 102046.
- Euler, L. (1741), ‘Solutio problematis ad geometriam situs pertinentis’, *Commentarii academiae scientiarum Petropolitanae* pp. 128–140.
- Fang, J., Xu, D. et al. (2020), ‘How the internet influences the development of modern market economy’, *American Journal of Industrial and Business Management* **10**(05), 1002.
- Fey, M. & Lenssen, J. E. (2019), Fast graph representation learning with PyTorch Geometric, in ‘ICLR Workshop on Representation Learning on Graphs and Manifolds’.
- Forbes (2021), ‘The sustainable impact of a paperless office’, <https://www.forbes.com/sites/forbestechcouncil/2021/05/11/the-sustainable-impact-of-a-paperless-office/?sh=3cbe7f371095>. Acessado em: 03 de Agosto de 2021.
- Gallicchio, C. & Micheli, A. (2010), Graph echo state networks, in ‘The 2010 International Joint Conference on Neural Networks (IJCNN)’, IEEE, pp. 1–8.
- Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V. & Garcia-Rodriguez, J. (2017), ‘A review on deep learning techniques applied to semantic segmentation’, *arXiv preprint arXiv:1704.06857*.

- Gehring, J., Auli, M., Grangier, D. & Dauphin, Y. N. (2016), 'A convolutional encoder model for neural machine translation', *arXiv preprint arXiv:1611.02344*.
- Glaubitz, J. C. (2004), 'Convert: a user-friendly program to reformat diploid genotypic data for commonly used population genetic software packages', *Molecular Ecology Notes* **4**(2), 309–310.
- Godsil, C. & Royle, G. F. (2013), *Algebraic graph theory*, Vol. 207, Springer Science & Business Media.
- Gopalakrishnan, K., Khaitan, S. K., Choudhary, A. & Agrawal, A. (2017), 'Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection', *Construction and Building Materials* **157**, 322–330.
- Gori, M., Monfardini, G. & Scarselli, F. (2005), A new model for learning in graph domains, in 'Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.', Vol. 2, IEEE, pp. 729–734.
- Gross, J. L. & Yellen, J. (2004), *Handbook of graph theory*, CRC press.
- Guo, H., Yan, R., Feng, Y., Gao, X. & Zhu, Z. (n.d.), 'Simplifying graph attention networks with source-target separation'.
- Hammond, D. K., Vandergheynst, P. & Gribonval, R. (2011), 'Wavelets on graphs via spectral graph theory', *Applied and Computational Harmonic Analysis* **30**(2), 129–150.
- Hovy, E., Navigli, R. & Ponzetto, S. P. (2013), 'Collaboratively built semi-structured content and artificial intelligence: The story so far', *Artificial Intelligence* **194**, 2–27.
- Huang, Z., Chung, W. & Chen, H. (2004), 'A graph model for e-commerce recommender systems', *Journal of the American Society for information science and technology* **55**(3), 259–274.
- Illinois Library (2021), 'Green libraries: Paper use and recycling', <https://guides.library.illinois.edu/green-libraries/purchasing/paper>. Acessado em: 02 de Agosto de 2021.
- Isufi, E., Loukas, A., Simonetto, A. & Leus, G. (2016), 'Autoregressive moving average graph filtering', *IEEE Transactions on Signal Processing* **65**(2), 274–288.
- Jiang, B., Lin, D., Tang, J. & Luo, B. (2019), Data representation and learning with graph diffusion-embedding networks, in 'Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)'.

Kabirifar, K. & Mojtahedi, M. (2019), 'The impact of engineering, procurement and construction (epc) phases on project performance: A case of large-scale residential construction project', *Buildings* **9**(1), 15.

Kalra, M. & Lal, N. (2016), Data mining of heterogeneous data with research challenges, in '2016 Symposium on Colossal Data Analysis and Networking (CDAN)', IEEE, pp. 1–6.

Kannan, N. & Vishveshwara, S. (1999), 'Identification of side-chain clusters in protein structures by a graph spectral method', *Journal of molecular biology* **292**(2), 441–464.

Kepner, J. & Gilbert, J. (2011), *Graph algorithms in the language of linear algebra*, SIAM.

Kettouch, M. S. (2017), A new approach for interlinking and integrating semi-structured and linked data, PhD thesis, Anglia Ruskin University.

Khamsi, M. A. & Kirk, W. A. (2011), *An introduction to metric spaces and fixed point theory*, Vol. 53, John Wiley & Sons.

Kipf, T. N. & Welling, M. (2016), 'Semi-supervised classification with graph convolutional networks', *arXiv preprint arXiv:1609.02907*.

Kosaraju, V., Sadeghian, A., Martín-Martín, R., Reid, I., Rezatofighi, H. & Savarese, S. (2019), Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks, in 'Advances in Neural Information Processing Systems', pp. 137–146.

Krogh, A. (2008), 'What are artificial neural networks?', *Nature biotechnology* **26**(2), 195–197.

Lee, M. & On, B.-W. (2015), 'Social graph visualization techniques for public data', *Journal of the HCI Society of Korea* **10**(1), 5–17.

Levie, R., Huang, W., Bucci, L., Bronstein, M. M. & Kutyniok, G. (2019), 'Transferability of spectral graph convolutional neural networks', *arXiv preprint arXiv:1907.12972*.

Levie, R., Monti, F., Bresson, X. & Bronstein, M. M. (2018), 'Cayleynets: Spectral graph cnns with complex rational filters'.

Li, J., Li, X. & He, D. (2019), 'A directed acyclic graph network combined with cnn and lstm for remaining useful life prediction', *IEEE Access* **7**, 75464–75475.

Lim, J., Ryu, S., Park, K., Choe, Y. J., Ham, J. & Kim, W. Y. (2019), 'Predicting drug–target interaction using a novel graph neural network with 3d structure-embedded graph representation', *Journal of chemical information and modeling* **59**(9), 3981–3988.

- Liu, J., Chang, W.-C., Wu, Y. & Yang, Y. (2017), Deep learning for extreme multi-label text classification, *in* 'Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval', pp. 115–124.
- Lu, Q. & Getoor, L. (2003), Link-based text classification, *in* 'IJCAI Workshop on Text Mining and Link Analysis'.
- Mao, X., Yang, H., Huang, S., Liu, Y. & Li, R. (2019), 'Extractive summarization using supervised and unsupervised learning', *Expert Systems with Applications* **133**, 173–181.
- Matsubayashi, K., Yamashita, A., Nonaka, H. & Konno, Y. (2018), A research on document summarization and presentation system based on feature word extraction from stored informations, *in* '2018 Conference on Technologies and Applications of Artificial Intelligence (TAAI)', IEEE, pp. 60–63.
- McCulloch, W. S. & Pitts, W. (1943), 'A logical calculus of the ideas immanent in nervous activity', *The bulletin of mathematical biophysics* **5**(4), 115–133.
- Mujtaba, G., Shuib, L., Raj, R. G., Rajandram, R., Shaikh, K. & Al-Garadi, M. A. (2018), 'Classification of forensic autopsy reports through conceptual graph-based document representation model', *Journal of biomedical informatics* **82**, 88–105.
- Murphy, O. J. (1990), 'Nearest neighbor pattern classification perceptrons', *Proceedings of the IEEE* **78**(10), 1595–1598.
- Namata, G., London, B., Getoor, L., Huang, B. & EDU, U. (2012), Query-driven active surveying for collective classification, *in* '10th International Workshop on Mining and Learning with Graphs', Vol. 8.
- Narang, S. K., Gadde, A. & Ortega, A. (2013), Signal processing techniques for interpolation in graph structured data, *in* '2013 IEEE International Conference on Acoustics, Speech and Signal Processing', IEEE, pp. 5445–5449.
- Nilsson, A. (2020), On the experimental transferability of spectral graph convolutional networks, Technical report.
- Nugroho, B., Aritsugi, M., Otachi, Y. & Manabe, Y. (2019), 'Combined graph kernels for automatic patent classification: A hybrid approach', *World Patent Information* **57**, 18–24.
- Ostendorff, M., Bourgonje, P., Berger, M., Moreno-Schneider, J., Rehm, G. & Gipp, B. (2019), 'Enriching bert with knowledge graph embeddings for document classification', *arXiv preprint arXiv:1909.08402*.
- Pardalos, P. M. & Xue, J. (1994), 'The maximum clique problem', *Journal of global Optimization* **4**(3), 301–328.

Perez, L. & Wang, J. (2017), ‘The effectiveness of data augmentation in image classification using deep learning’, *arXiv preprint arXiv:1712.04621* .

Peterson, L. E. (2009), ‘K-nearest neighbor’, *Scholarpedia* **4**(2), 1883.

Petrella, J. R. (2011), ‘Use of graph theory to evaluate brain networks: a clinical tool for a small world?’.

Qian, G., Sural, S., Gu, Y. & Pramanik, S. (2004), Similarity between euclidean and cosine angle distance for nearest neighbor queries, in ‘Proceedings of the 2004 ACM symposium on Applied computing’, pp. 1232–1237.

Rajaraman, A. & Ullman, J. D. (2011), *Mining of massive datasets*, Cambridge University Press.

Rauber, T. W. (2005), ‘Redes neurais artificiais’, *Universidade Federal do Espírito Santo* p. 29.

Record Nations (2021), ‘How much paper is used in one day?’, <https://www.recordnations.com/2016/02/how-much-paper-is-used-in-one-day/5>. Acesso em: 03 de Agosto de 2021.

Riaz, F. & Ali, K. M. (2011), Applications of graph theory in computer science, in ‘2011 Third International Conference on Computational Intelligence, Communication Systems and Networks’, IEEE, pp. 142–145.

Richter, T., Escher, S., Schönfeld, D. & Strufe, T. (2018), Forensic analysis and anonymisation of printed documents, in ‘Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security’, pp. 127–138.

Rinaldi, A. M., Russo, C. & Tommasino, C. (2021), ‘A semantic approach for document classification using deep neural networks and multimedia knowledge graph’, *Expert Systems with Applications* **169**, 114320.

Robinson, I., Webber, J. & Eifrem, E. (2015), *Graph databases: new opportunities for connected data*, ”O’Reilly Media, Inc.”.

Rosenblatt, F. (1958), ‘The perceptron: a probabilistic model for information storage and organization in the brain.’, *Psychological review* **65**(6), 386.

Sahu, S. K., Christopoulou, F., Miwa, M. & Ananiadou, S. (2019), ‘Inter-sentence relation extraction with document-level graph convolutional neural network’, *arXiv preprint arXiv:1906.04684* .

- Sanches, M. K. (2003), Aprendizado de máquina semi-supervisionado: proposta de um algoritmo para rotular exemplos a partir de poucos exemplos rotulados, PhD thesis, Universidade de São Paulo.
- Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M. & Monfardini, G. (2008), ‘The graph neural network model’, *IEEE Transactions on Neural Networks* **20**(1), 61–80.
- Schenker, A., Last, M., Bunke, H. & Kandel, A. (2003), Classification of web documents using a graph model, in ‘Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.’, IEEE, pp. 240–244.
- Sejnowski, T. J. (2020), *A Revolução do Aprendizado Profundo*, Alta Books.
- Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B. & Eliassi-Rad, T. (2008), ‘Collective classification in network data’, *AI magazine* **29**(3), 93–93.
- Stergiou, C. & Siganos, D. (2010), ‘Neural networks. 1996’.
- Strohman, T., Croft, W. B. & Jensen, D. (2007), Recommending citations for academic papers, in ‘Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval’, pp. 705–706.
- Szekely, P., Knoblock, C. A., Slepicka, J., Philpot, A., Singh, A., Yin, C., Kapoor, D., Natarajan, P., Marcu, D., Knight, K. et al. (2015), Building and using a knowledge graph to combat human trafficking, in ‘International Semantic Web Conference’, Springer, pp. 205–221.
- Szulc, I., Stencel, K. & Wiśniewski, P. (2017), Using genetic algorithms to optimize redundant data, in ‘International Conference: Beyond Databases, Architectures and Structures’, Springer, pp. 165–176.
- Teney, D., Liu, L. & van Den Hengel, A. (2017), Graph-structured representations for visual question answering, in ‘Proceedings of the IEEE conference on computer vision and pattern recognition’, pp. 1–9.
- Thongkor, K. & Amornraksa, T. (2017), Digital image watermarking for printed and scanned documents, in ‘Ninth International Conference on Digital Image Processing (ICDIP 2017)’, Vol. 10420, International Society for Optics and Photonics, p. 104203O.
- Torng, W. & Altman, R. B. (2019), ‘Graph convolutional neural networks for predicting drug-target interactions’, *Journal of Chemical Information and Modeling* **59**(10), 4131–4149.
- Tremblay, N., Gonçalves, P. & Borgnat, P. (2018), Design of graph filters and filterbanks, in ‘Cooperative and Graph Signal Processing’, Elsevier, pp. 299–324.

- Van Engelen, J. E. & Hoos, H. H. (2020), ‘A survey on semi-supervised learning’, *Machine Learning* **109**(2), 373–440.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. & Polosukhin, I. (2017), Attention is all you need, in ‘Advances in neural information processing systems’, pp. 5998–6008.
- Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P. & Bengio, Y. (2017), ‘Graph attention networks’, *arXiv preprint arXiv:1710.10903* .
- Viégas, F. B. & Donath, J. (2004), Social network visualization: Can we go beyond the graph, in ‘Workshop on social networks, CSCW’, Vol. 4, pp. 6–10.
- Wang, S., Tang, J., Aggarwal, C. & Liu, H. (2016), Linked document embedding for classification, in ‘Proceedings of the 25th ACM international on conference on information and knowledge management’, pp. 115–124.
- Wang, Y. & Zeng, J. (2013), ‘Predicting drug-target interactions using restricted boltzmann machines’, *Bioinformatics* **29**(13), i126–i134.
- Wang, Z., Chen, T., Ren, J., Yu, W., Cheng, H. & Lin, L. (2018), ‘Deep reasoning with knowledge graph for social relationship understanding’, *arXiv preprint arXiv:1807.00504* .
- Wang, Z., Wang, C., Zhang, H., Duan, Z., Zhou, M. & Chen, B. (2020), Learning dynamic hierarchical topic graph with graph convolutional network for document classification, in ‘International Conference on Artificial Intelligence and Statistics’, PMLR, pp. 3959–3969.
- West, D. B. et al. (1996), *Introduction to graph theory*, Vol. 2, Prentice hall Upper Saddle River, NJ.
- Widrow, B. & Hoff, M. E. (1960), Adaptive switching circuits, Technical report, Stanford Univ Ca Stanford Electronics Labs.
- Wu, F., Zhang, T., Souza Jr, A. H. d., Fifty, C., Yu, T. & Weinberger, K. Q. (2019), ‘Simplifying graph convolutional networks’, *arXiv preprint arXiv:1902.07153* .
- Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C. & Philip, S. Y. (2020), ‘A comprehensive survey on graph neural networks’, *IEEE Transactions on Neural Networks and Learning Systems* .
- Xu, K., Hu, W., Leskovec, J. & Jegelka, S. (2018), ‘How powerful are graph neural networks?’, *arXiv preprint arXiv:1810.00826* .

- Yang, Z., Cohen, W. & Salakhudinov, R. (2016), Revisiting semi-supervised learning with graph embeddings, *in* ‘International conference on machine learning’, PMLR, pp. 40–48.
- Yao, L., Mao, C. & Luo, Y. (2019), Graph convolutional networks for text classification, *in* ‘Proceedings of the AAAI conference on artificial intelligence’, Vol. 33, pp. 7370–7377.
- You, J., Ying, Z. & Leskovec, J. (2020), ‘Design space for graph neural networks’, *Advances in Neural Information Processing Systems* **33**.
- Zhang, C., James, J. & Liu, Y. (2019), ‘Spatial-temporal graph attention networks: A deep learning approach for traffic forecasting’, *IEEE Access* **7**, 166246–166256.
- Zhang, S., Tong, H., Xu, J. & Maciejewski, R. (2019), ‘Graph convolutional networks: a comprehensive review’, *Computational Social Networks* **6**(1), 11.
- Zhang, Y., Yu, X., Cui, Z., Wu, S., Wen, Z. & Wang, L. (2020), ‘Every document owns its structure: Inductive text classification via graph neural networks’, *arXiv preprint arXiv:2004.13826* .
- Zhao, L., Song, Y., Zhang, C., Liu, Y., Wang, P., Lin, T., Deng, M. & Li, H. (2019), ‘T-gcn: A temporal graph convolutional network for traffic prediction’, *IEEE Transactions on Intelligent Transportation Systems* .
- Zhao, T., Hu, Y., Valsdottir, L. R., Zang, T. & Peng, J. (2020), ‘Identifying drug–target interactions based on graph convolutional network and deep neural network’, *Briefings in Bioinformatics* .
- Zheng, P., Chen, C.-H. & Shang, S. (2019), ‘Towards an automatic engineering change management in smart product-service systems—a dsm-based learning approach’, *Advanced Engineering Informatics* **39**, 203–213.
- Zhou, J., Cui, G., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C. & Sun, M. (2018), ‘Graph neural networks: A review of methods and applications’, *arXiv preprint arXiv:1812.08434* .

ANEXO A TABELAS

Tabela 22: Valores de acurácia de teste para o dataset Cora.

Epoch	GCN	GAT	Cheb	ARMA
1	0,206±0,075	0,23±0,081	0,251±0,060	0,186±0,068
5	0,518±0,100	0,483±0,150	0,488±0,106	0,232±0,095
10	0,637±0,082	0,661±0,103	0,593±0,083	0,321±0,109
20	0,715±0,046	0,753±0,044	0,731±0,044	0,483±0,110
30	0,754±0,031	0,788±0,023	0,768±0,027	0,611±0,096
40	0,784±0,013	0,793±0,025	0,779±0,022	0,698±0,074
50	0,784±0,013	0,802±0,016	0,785±0,022	0,744±0,058
60	0,796±0,016	0,807±0,013	0,796±0,015	0,777±0,054
70	0,802±0,012	0,809±0,013	0,801±0,014	0,8±0,032
80	0,806±0,010	0,811±0,012	0,798±0,017	0,809±0,017
90	0,809±0,008	0,812±0,011	0,801±0,016	0,812±0,009
100	0,811±0,008	0,816±0,011	0,801±0,018	0,815±0,011
200	0,814±0,006	0,831±0,005	0,805±0,009	0,822±0,006

Tabela 23: Valores de acurácia de teste para o dataset Citeseer.

Epoch	GCN	GAT	Cheb	ARMA
1	0,204±0,069	0,210±0,061	0,237±0,057	0,183±0,049
5	0,432±0,093	0,356±0,144	0,389±0,102	0,220±0,072
10	0,533±0,090	0,503±0,117	0,491±0,093	0,277±0,102
20	0,630±0,04	0,629±0,066	0,589±0,054	0,382±0,079
30	0,658±0,030	0,678±0,033	0,632±0,037	0,478±0,085
40	0,684±0,018	0,689±0,023	0,656±0,020	0,555±0,071
50	0,684±0,018	0,699±0,022	0,668±0,020	0,604±0,060
60	0,693±0,013	0,704±0,012	0,672±0,019	0,640±0,051
70	0,697±0,012	0,704±0,013	0,677±0,017	0,662±0,040
80	0,700±0,011	0,703±0,013	0,678±0,019	0,679±0,042
90	0,704±0,010	0,705±0,011	0,684±0,019	0,682±0,036
100	0,705±0,010	0,706±0,010	0,689±0,015	0,691±0,025
200	0,710±0,009	0,708±0,005	0,699±0,012	0,708±0,008

Tabela 24: Valores de acurácia de teste para o dataset Pubmed.

Epoch	GCN	GAT	Cheb	ARMA
1	0,427±0,128	0,383±0,108	0,435±0,078	0,400±0,097
5	0,647±0,061	0,647±0,084	0,648±0,048	0,529±0,111
10	0,687±0,037	0,707±0,030	0,686±0,039	0,618±0,084
20	0,711±0,026	0,714±0,011	0,730±0,018	0,681±0,064
30	0,733±0,016	0,719±0,007	0,749±0,01	0,720±0,048
40	0,7590±0,012	0,736±0,006	0,758±0,011	0,751±0,021
50	0,759±0,012	0,756±0,005	0,768±0,007	0,763±0,028
60	0,769±0,009	0,770±0,006	0,772±0,008	0,773±0,024
70	0,775±0,009	0,783±0,005	0,774±0,008	0,778±0,008
80	0,781±0,008	0,787±0,005	0,775±0,007	0,778±0,012
90	0,783±0,007	0,788±0,004	0,775±0,007	0,780±0,009
100	0,785±0,007	0,788±0,004	0,777±0,007	0,781±0,006
200	0,788±0,005	0,790±0,003	0,782±0,006	0,787±0,003

Tabela 25: Valores de acurácia de teste variando os valores de drop out.

Dataset	Drop Out	GCN	GAT	Cheb	ARMA
CORA	0.5	0,814±0,006	0,831±0,005	0,805±0,009	0,822±0,006
	0.6	0,814±0,006	0,831±0,004	0,811±0,012	0,821±0,025
	0.7	0,816±0,007	0,830±0,005	0,815±0,01	0,824±0,007
	0.8	0,817±0,007	0,830±0,008	0,819±0,007	0,824±0,007
	0.9	0,815±0,011	0,767±0,034	0,819±0,008	0,825±0,008
CITeseer	0.5	0,710±0,009	0,708±0,005	0,699±0,012	0,708±0,008
	0.6	0,710±0,008	0,708±0,006	0,697±0,012	0,711±0,007
	0.7	0,708±0,009	0,710±0,006	0,696±0,016	0,708±0,009
	0.8	0,705±0,009	0,713±0,008	0,697±0,013	0,708±0,010
	0.9	0,698±0,017	0,605±0,075	0,699±0,012	0,697±0,021
PUBMED	0.5	0,788±0,005	0,790±0,003	0,782±0,006	0,787±0,003
	0.6	0,787±0,006	0,790±0,004	0,782±0,007	0,786±0,008
	0.7	0,786±0,007	0,791±0,003	0,781±0,005	0,785±0,004
	0.8	0,783±0,007	0,787±0,005	0,778±0,007	0,782±0,004
	0.9	0,776±0,011	0,739±0,013	0,770±0,008	0,776±0,005

Tabela 26: Cora: Duração em minutos do processo de treino variando o número de épocas.

Epoch	GCN	GAT	Cheb	ARMA
1	0,0117	0,0183	0,0350	0,0150
5	0,0600	0,0833	0,1683	0,0633
10	0,1217	0,1617	0,3450	0,1267
20	0,2333	0,3283	0,6833	0,2500
30	0,3550	0,4967	1,0183	0,3750
40	0,6000	0,6100	1,3550	0,5017
50	0,6083	0,7850	1,6450	0,6083
60	0,7167	0,9383	2,0233	0,7700
70	0,8133	1,0833	2,3300	0,8933
80	0,9250	1,2483	2,6067	1,0200
90	1,0517	1,3783	2,7533	1,1033
100	1,7300	2,6250	5,5500	1,9000
200	1,9350	8,6983	10,4283	6,3283

Tabela 27: Citeseer: Duração em minutos do processo de treino variando o número de épocas.

Epoch	GCN	GAT	Cheb	ARMA
1	0,0150	0,0200	0,0400	0,0317
5	0,0633	0,0867	0,1933	0,1500
10	0,1267	0,1767	0,3883	0,3150
20	0,2483	0,3433	0,7750	0,6617
30	0,3667	0,5150	1,1467	1,0050
40	0,6167	0,6750	1,5417	1,3400
50	0,6167	0,8367	1,8683	1,6350
60	0,7300	1,0200	2,2950	2,0150
70	0,8417	1,1800	2,6933	2,3550
80	0,9683	1,3483	3,0667	2,6933
90	1,0717	1,5150	3,4317	3,0100
100	1,2733	2,7500	7,5333	5,2417
200	2,2417	9,3617	8,3617	16,8033

Tabela 28: Pubmed: Duração em minutos do processo de treino variando o número de épocas.

Epoch	GCN	GAT	Cheb	ARMA
1	0,0150	0,0283	0,0400	0,0400
5	0,0667	0,1250	0,2067	0,1900
10	0,1283	0,2567	0,4067	0,3800
20	0,2483	0,5167	0,8150	0,7667
30	0,3600	0,7800	1,2250	1,1517
40	0,6033	1,0367	1,6283	1,5400
50	0,6033	1,2700	1,9717	1,8900
60	0,7417	1,5217	2,4283	2,3050
70	0,8417	1,7800	2,7083	2,6933
80	0,9483	2,0400	2,7617	3,0750
90	1,0717	2,2900	2,8350	3,4600
100	0,9600	5,7117	5,7817	4,9433
200	1,5317	13,3967	10,9217	18,9867

Tabela 29: Duração em minutos do processo de treino variando a função de ativação.

Dataset	Activation Function	GCN	GAT	Cheb	ARMA
CORA	ReLU	1,9350	6,1916	3,7616	6,3283
	ELU	1,3816	8,6983	3,3833	4,6550
	Softmax	0,9383	6,1916	3,1866	7,4500
CITeseer	ReLU	2,2416	7,1650	4,7466	16,8033
	ELU	1,4783	9,3616	4,4616	11,5466
	Softmax	0,9533	7,0016	3,6900	11,6366
PUBMED	ReLU	1,1266	10,8766	4,0383	18,9866
	ELU	1,2316	13,3966	4,0733	15,9466
	Softmax	0,9933	10,8966	3,9866	15,9083

Tabela 30: Duração em minutos do processo de treino variando a função loss.

Dataset	Loss Function	GCN	GAT	Cheb	ARMA
CORA	nll	1,9350	8,6983	10,4283	6,3283
	cross_entropy	1,4533	6,2900	3,3966	4,9333
CITeseer	nll	2,2416	9,3616	8,3616	16,8033
	cross_entropy	1,6200	7,1433	4,7566	12,7833
PUBMED	nll	1,5316	13,3966	10,9216	18,9866
	cross_entropy	1,1916	11,1783	4,0683	16,2350

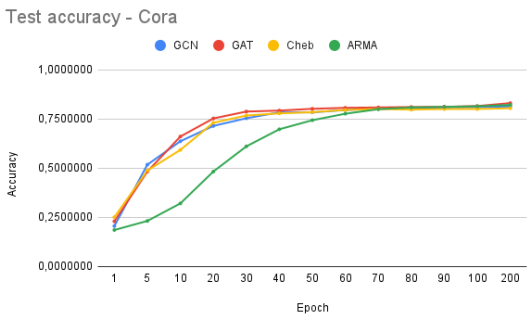
Tabela 31: Duração em minutos do processo de treino variando o valor de drop out.

Dataset	Drop Out	GCN	GAT	Cheb	ARMA
CORA	0.5	1,9350	8,6983	10,4283	6,3283
	0.6	1,4116	6,1433	3,4433	4,6933
	0.7	1,8883	8,3900	3,8433	6,2433
	0.8	2,0500	9,0966	3,8583	6,3133
	0.9	1,5383	6,4766	3,7683	4,8033
CITeseer	0.5	2,2416	9,3616	8,3616	16,8033
	0.6	1,6350	7,0183	4,7983	11,7200
	0.7	2,2400	9,1800	5,1850	16,7800
	0.8	2,3766	9,2366	5,3916	16,7833
	0.9	1,5550	7,2800	5,3733	12,0716
PUBMED	0.5	1,5316	13,3966	10,9216	18,9866
	0.6	1,1800	10,9200	4,0550	16,0233
	0.7	1,5483	13,2883	4,1950	18,9216
	0.8	1,6200	13,2900	4,2250	18,9333
	0.9	1,2833	11,2733	4,3433	16,1283

ANEXO B FIGURAS

Figura 41: Valores de acurácia de teste variando as épocas.

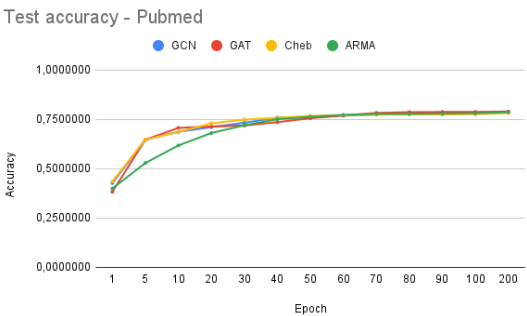
(a) Dataset Cora.



(b) Dataset Citeseer.

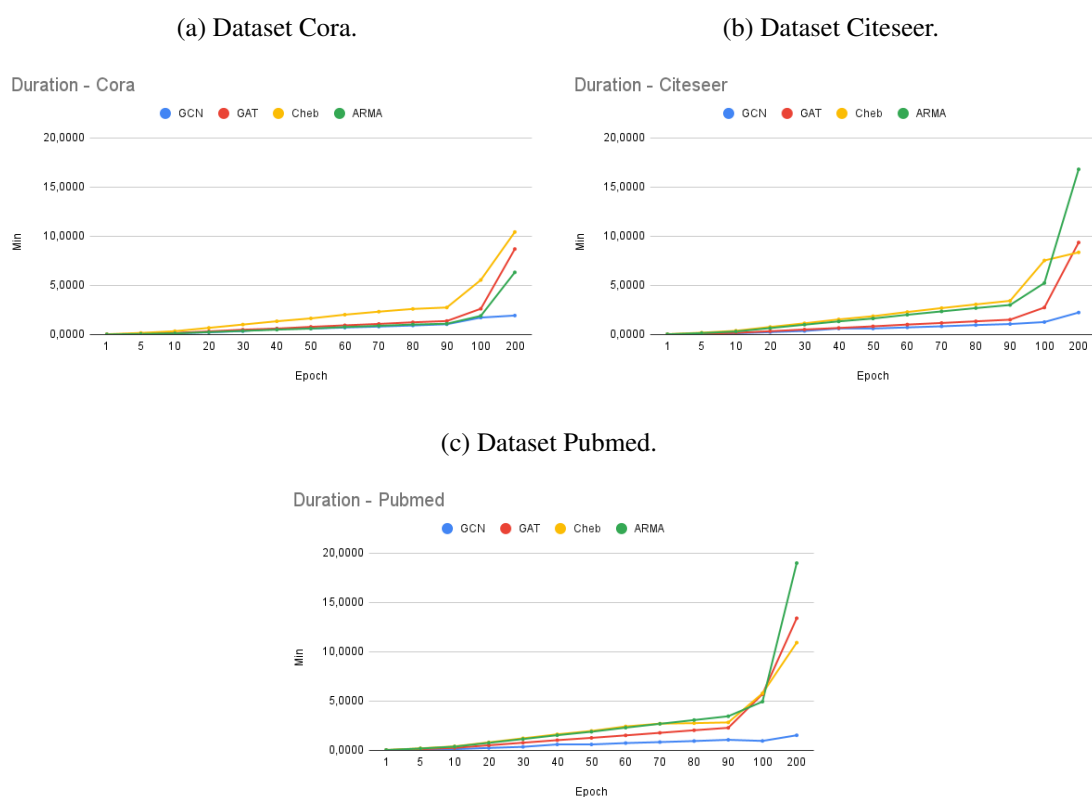


(c) Dataset Pubmed.



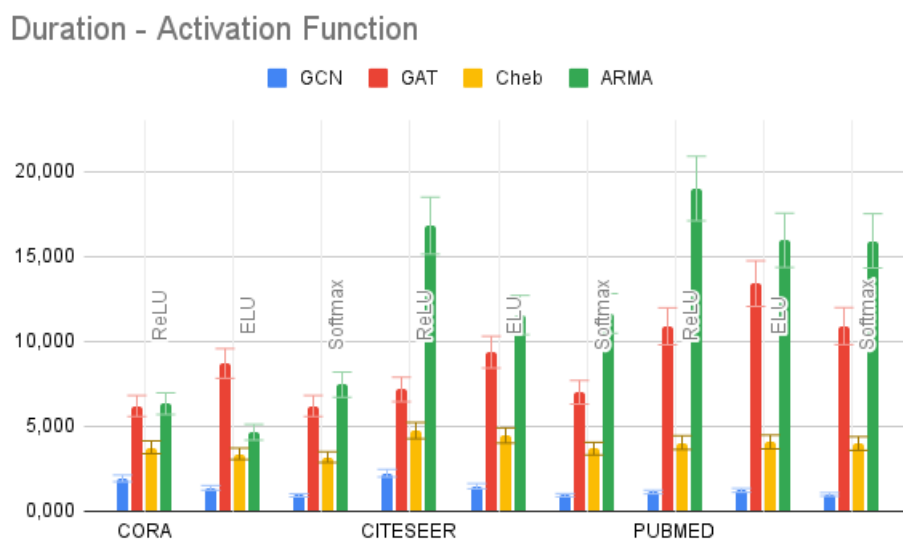
Fonte: Elaborado pelo autor

Figura 42: Modelo GCN: Duração dos processos de treino.



Fonte: Elaborado pelo autor

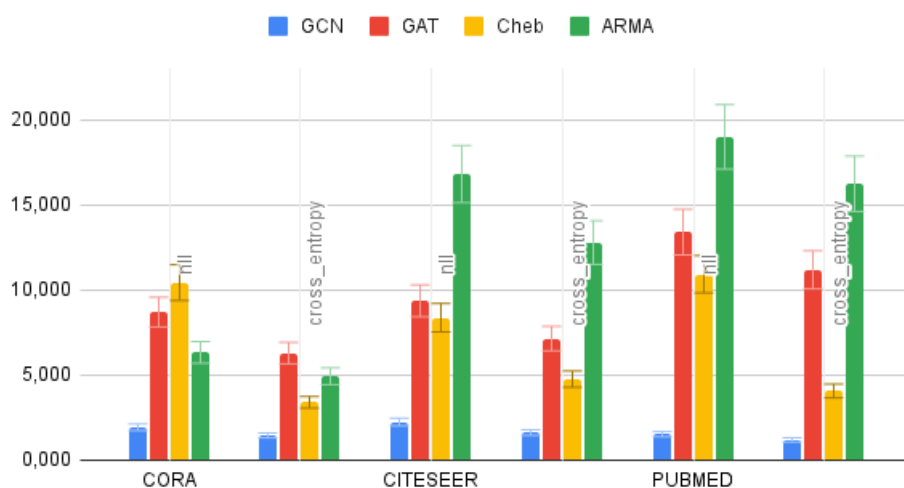
Figura 43: Duração dos processo de de treino variando a função de ativação.



Fonte: Elaborado pelo autor

Figura 44: Duração dos processo de de treino variando a função de loss

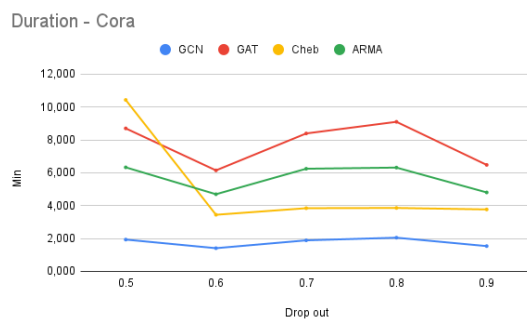
Duration - Loss Function



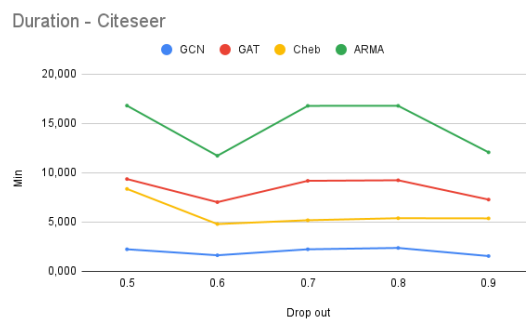
Fonte: Elaborado pelo autor

Figura 45: Duração dos processos de treino variando o Drop out.

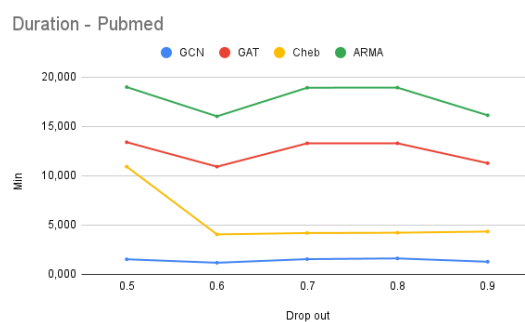
(a) Dataset Cora.



(b) Dataset Citeseer.

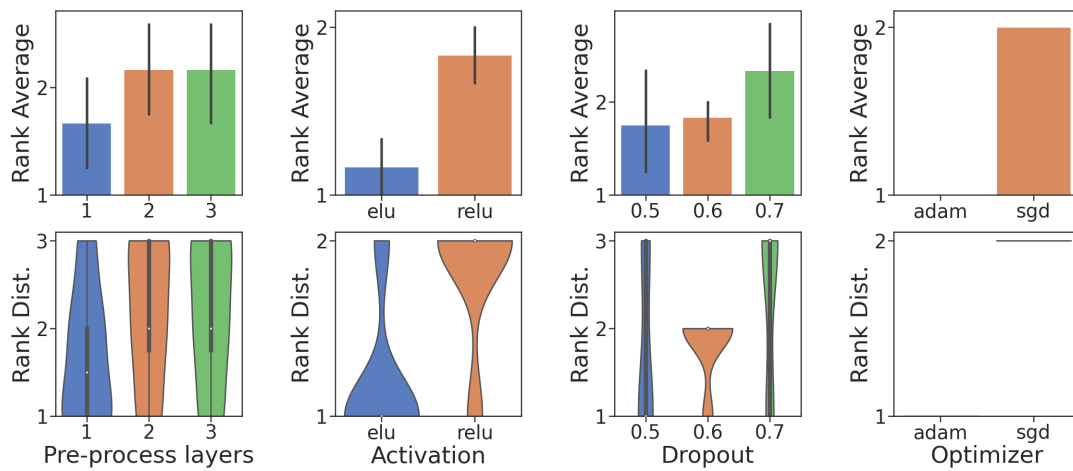


(c) Dataset Pubmed.



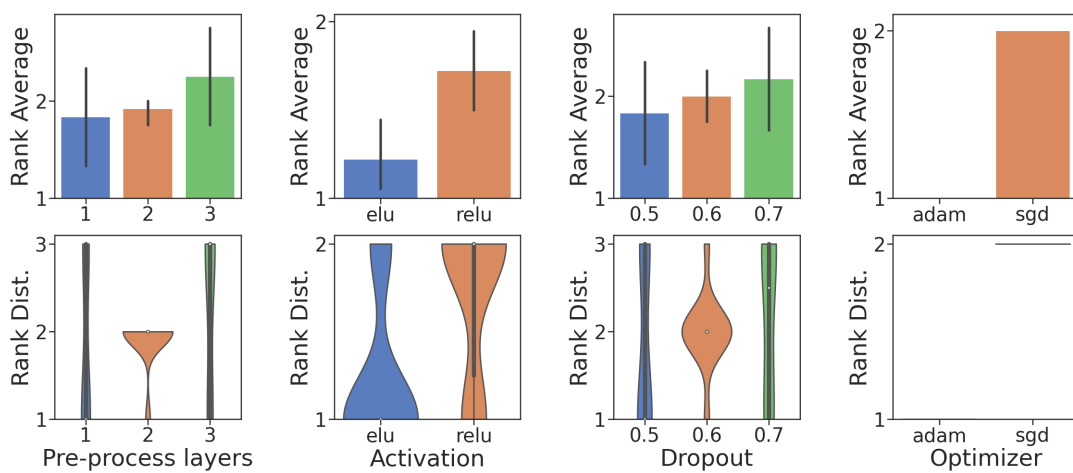
Fonte: Elaborado pelo autor

Figura 46: Resultados para o dataset Cora.



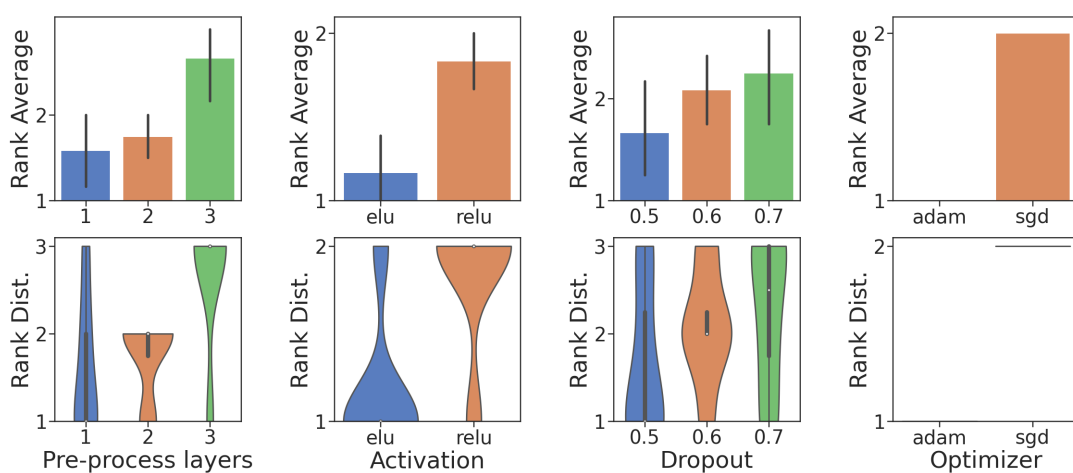
Fonte: Elaborado pelo autor

Figura 47: Resultados para o dataset Citeseer.



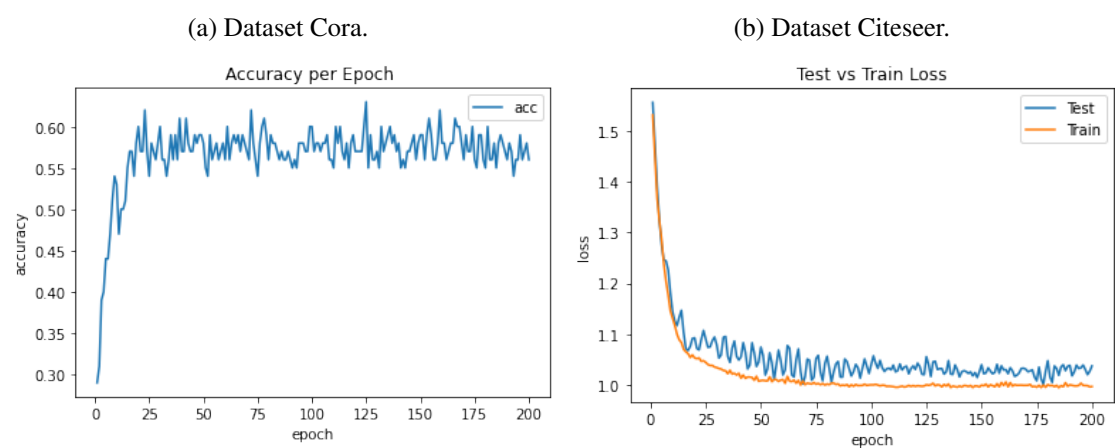
Fonte: Elaborado pelo autor

Figura 48: Resultados para o dataset Pubmed.



Fonte: Elaborado pelo autor

Figura 49: Resultados para o dataset EPC com tipo de conexão Registro com aprendizagem supervisionada.



Fonte: Elaborado pelo autor