UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Redes neurais para grafos e suas aplicações aos sistemas complexos

Guilherme Michel Lima de Carvalho

Dissertação de Mestrado do Programa Interinstitucional de Pós-Graduação em Estatística (PIPGEs)

ICMC USP SÃO CARLOS

Guilherme Michel Lima de Carvalho

Redes neurais para grafos e suas aplicações aos sistemas complexos

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP e ao Departamento de Estatística – DEs-UFSCar, como parte dos requisitos para obtenção do título de Mestre em Estatística – Programa Interinstitucional de Pós-Graduação em Estatística. *VERSÃO REVISADA*

Área de Concentração: Estatística

Orientador: Prof. Dr. Francisco Aparecido Rodrigues

USP – São Carlos
Junho de 2022

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

Lima de Carvalho, Guilherme Michel
L331r Redes neurais para grafos e suas aplicações aos sistemas complexos / Guilherme Michel Lima de Carvalho; orientador Francisco Aparecido Rodrigues. -- São Carlos, 2022.
92 p.

Dissertação (Mestrado - Programa Interinstitucional de Pós-graduação em Estatística) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2022.

1. Redes Neurais para Grafos. 2. Sistemas Complexos. 3. Aprendizado de Máquina. I. Aparecido Rodrigues, Francisco , orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Guilherme Michel Lima de Carvalho

Graph neural networks and its applications to complex systems

Master dissertation submitted to the Institute of Mathematics and Computer Sciences – ICMC-USP and to the Department of Statistics – DEs-UFSCar, in partial fulfillment of the requirements for the degree of the Master Interagency Program Graduate in Statistics. *FINAL VERSION*

Concentration Area: Statistics

Advisor: Prof. Dr. Francisco Aparecido Rodrigues

USP – São Carlos
June 2022

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia
Programa Interinstitucional de Pós-Graduação em Estatística

Folha de Aprovação

Defesa de Dissertação de Mestrado do candidato Guilherme Michel Lima de Carvalho, realizada em 08/04/2022.

Comissão Julgadora:

Prof. Dr. Francisco Aparecido Rodrigues (USP)

Prof. Dr. Pedro Luiz Ramos (PUC-Chile)

Prof. Dr. Diego Raphael Amancio (USP)

O Relatório de Defesa assinado pelos membros da Comissão Julgadora encontra-se arquivado junto ao Programa Interinstitucional de Pós-Graduação em Estatística.

Dedico este trabalho em memória ao meu pai e avô José Carlos de Carvalho.

AGRADECIMENTOS

Tantas foram as pessoas que me apoiaram durante este trajeto. Primeiramente, gostaria de agradecer ao meu orientador Francisco Aparecido Rodrigues por ter acompanhado esse processo e por ter me motivado a ter curiosidade e pensamento crítico sobre os assuntos. Além disso, também foi essencial para o desenvolvimento do projeto todas as disciplinas que cursei no programa de PIPGEs, em especial, Modelos de Regressão com o Diniz, Aprendizado de Máquina com o Izbicki e Processos dinâmicos em redes complexas com o Francisco Rodrigues. Sem esta base potencialmente não conseguiria ter caminhado até aqui.

Além disso, também quero agradecer a toda a estrutura do ICMC e do Departamento de Estatística da UFSCar que propiciaram ótimos locais para estudo e também ótimos livros.

Ademais, a todos os amigos que fiz durante o tempo que fiquei em São Carlos, em especial, aos colegas do PIPGEs pelas diversas horas de estudo que passamos juntos.

Agradeço também a minha mãe e avó Eunice Carvalho que apesar de não entender muito bem o que eu fiz e estudei nesse tempo, me forneceu todo o apoio emocional nos momentos mais difíceis.

Por fim, agradeço a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001, pelo apoio financeiro sem o qual essa pesquisa não teria possível .

RESUMO

GUILHERME MICHEL L. DE CARVALHO. **Redes neurais para grafos e suas aplicações aos sistemas complexos**. 2022. 90 p. Dissertação (Mestrado em Estatística – Programa Interinstitucional de Pós-Graduação em Estatística) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2022.

Sistemas complexos são compostos de diversos componentes que interagem entre si. Uma abordagem natural para estes tipos de sistemas é utilizando a abstração matemática de grafos. Em diversos contextos do mundo real é possível se utilizar técnicas de redes complexas para a modelagem desses sistemas. Nestes sistemas podem ocorrer processos dinâmicos como por exemplo a propagação de informação e a propagação de doenças. Neste trabalho consideramos a utilização de técnicas de redes neurais artificiais para dados estruturados como grafos com o objetivo de estudar a propagação de rumor em redes complexas e a detecção de estruturas de comunidades. Para o caso de propagação de rumor, foi proposto um modelo baseado em redes neurais para grafos com o objetivo de recuperar a origem de propagação em grafos artificiais com estruturas de comunidades e para a detecção de estruturas de comunidades foi avaliado o potencial do aprendizado de representações por redes neurais para grafos em comparação a algoritmos tradicionais da ciência de redes complexas.

Palavras-chave: Redes Neurais para Grafos, Sistemas Complexos, Aprendizado de Máquina.

ABSTRACT

GUILHERME MICHEL L. DE CARVALHO. **Graph neural networks and its applications to complex systems**. 2022. 90 p. Dissertação (Mestrado em Estatística – Programa Interinstitucional de Pós-Graduação em Estatística) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2022.

Complex systems are composed of several components that interact with each other. A natural approach for these types of systems is to use mathematical graph abstraction. In different contexts in the real world, it is possible to use complex network techniques to model these systems. In these systems, dynamic processes such as the spread of information and the spread of disease can occur. In this work we consider the use of artificial neural network techniques for graph-structured data in order to study the propagation of rumor in complex networks and the detection of community structures. For the proposed case of rumor, a model was developed based on graph neural networks for the porpuse of detected the source of the a rumour in graphs with community structure and for community dectection was evaluate the potential of graph neural networks in comparison to traditional methods of the network science.

Keywords: Graph Neural Networks, Complex systems, Machine Learning.

LISTA DE ILUSTRAÇÕES

Figura 1 – Evolução da dinâmica em uma rede com estrutura de comunidades 24
Figura 2 – Grafo artificial com estrutura de comunidade 25
Figura 3 – Grafo e sua representação em matriz de adjacências 27
Figura 4 – Grafo direcionado e sua representação em matriz de adjacências 28
Figura 5 – Rede ego-facebook coletada de alguns participantes de uma pesquisa na plataforma do facebook e a sua respectiva distribuição de grau, Fonte dos dados. 29
Figura 6 – Distribuição do grau de uma realização do modelo de rede ER 31
Figura 7 – Distribuição do grau de uma realização do modelo de rede BA 32
Figura 8 – Exemplo de uma realização do modelo LFR 34
Figura 9 – Integração numérica do sistema de equações diferenciais mean-field e a simulação estocástica na rede regular 40
Figura 10 – Exemplo da representação de um conjunto de dados com duas classes(vermelhos e verdes) em diferentes sistemas de coordenadas. (Figura inspirada em um exemplo do livro de Goodfellow (GOODFELLOW; BENGIO; COURVILLE, 2016)) 44
Figura 11 – Representação esquemática de uma MLP 45
Figura 12 – Exemplos de diferentes funções de ativação 46
Figura 13 – Exemplo de duas representações para matriz de adjacências para o mesmo grafo. Na imagem da esquerda uma ordem aleatória e na da direita os nós são ordenados de acordo com a comunidade a que pertencem 48
Figura 14 – Representação esquemática da atualização da representação entre camadas (Figura gerada utilizando o pacote Tikz, adaptado da coleção de Petar Veličković https://github.com/PetarV-/TikZ) 49
Figura 15 – Rede gerada pelo modelo LFR com $N = 1000$, $\tau_1 = 3$, $\tau_2 = 1.5$, $k_{min} = 10$, $k_{max} = 40$, $min_c = 100$, $max_c = 250$ e $\mu = 0.2$. 51
Figura 16 – Representações vetoriais para os vértices em um modelo de rede LFR(N=1000) usando camadas GCNs sem nenhum treinamento nos parâmetros treináveis e com dimensionalidade reduzida por t-SNE. 53
Figura 17 – Evolução da dinâmica em uma rede com estrutura de comunidades 57
Figura 18 – Evolução da dinâmica em uma rede LFR com comunidades de mesmo tamanho. 61
Figura 19 – Acurácia dos modelos em função dos passos de tempo para diferentes valores de μ 62

Figura 20 – Acurácia dos modelos em função da fração inicial de propagadores para diferentes valores de μ 64
Figura 21 – Acurácia dos modelos em função da taxa de propagação λ para diferentes valores de μ 65
Figura 22 – Acurácia dos modelos em função da taxa de recuperação α para diferentes valores de μ 66
Figura 23 – Evolução da dinâmica em uma rede LFR com comunidades de diversos tamanhos. 67
Figura 24 – Acurácia dos modelos em função dos passos de tempo para diferentes valores de μ 68
Figura 25 – Acurácia dos modelos em função da fração inicial de propagadores para diferentes valores de μ 69
Figura 26 – Acurácia dos modelos em função da taxa de propagação λ para diferentes valores de μ 70
Figura 27 – Acurácia dos modelos em função da taxa de recuperação α para diferentes valores de μ 71
Figura 28 – Representações vetoriais em 2 dimensões para a rede do modelo LFR mostrada na Figura 8. 75
Figura 29 – Redes geradas pelo modelo LFR com $N = 400$ e diferentes valores de μ. . . 77
Figura 30 – Representações vetoriais em redes LFR homogêneas obtidas com o modelo GCN e com dimensionalidade reduzida por t-SNE 78
Figura 31 – Comparação com métodos tradicionais para redes LFR com $N = 400$, $\tau_1 = 3$, $k_{min} = 10$, $k_{max} = 20$, $min_c = 100$, $max_c = 100$ e variando o parâmetro de mistura μ entre 0.01 e 0.9 79
Figura 32 – Redes heterogêneas geradas pelo modelo LFR com $N = 400$ e diferentes valores de μ. 80
Figura 33 – Representações vetoriais em redes LFR heterogêneas obtidas com o modelo GCN e com dimensionalidade reduzida por t-SNE 81
Figura 34 – Comparação com métodos tradicionais para redes LFR heterogênas com $N = 400$, $\tau_1 = 3$, $k_{min} = 10$, $k_{max} = 45$, $min_c = 25$, $max_c = 200$ e variando o parâmetro de mistura μ entre 0.01 e 0.9 82

LISTA DE ALGORITMOS

Algoritmo 1 – Simulação Maki-Thompson 39

LISTA DE ABREVIATURAS E SIGLAS

BA	Barabási-Albert
CNN	Convolutional Neural Network
ER	Erdős–Rényi
GAE	Graph Auto-Encoders
GCN	Graph Convolutional Networks
GNN	Graph Neural Networks
LFR	Lancichinetti–Fortunato–Radicchi
MLP	Multilayer Perceptron
NMI	Informação Mutua Normalizada

LISTA DE SÍMBOLOS

$\mathcal{G}$ — Grafo

$\mathcal{V}$ — Conjunto de vértices

$\mathcal{E}$ — Conjunto de arestas

$\mathbf{A}$ — Matriz de Adjacências

A_{ij} — Elemento i,j da Matriz de Adjacências $\mathbf{A}$

$\mathcal{L}$ — Função Perda (Loss)

$\oplus$ — Concatenação de vetores

SUMÁRIO

1 INTRODUÇÃO 23
1.1 Introdução 23
1.2 Organização do trabalho 26

2 REDES COMPLEXAS 27
2.1 Teoria das redes complexas 27
2.1.1 Definições e conceitos *27*
2.1.2 Medidas de centralidade e propriedades de redes complexas *28*
2.1.3 Modelos de Redes *30*
2.1.3.1 Modelo de Erdős–Rényi *30*
2.1.3.2 Modelo Barabási–Albert *31*
2.1.3.3 Modelo de configuração *33*
2.1.3.4 Modelo de Lancichinetti–Fortunato–Radicchi *33*

3 MODELAGEM MATEMÁTICA E SIMULAÇÃO DE PROCESSOS DINÂMICOS EM REDES COMPLEXAS 35
3.1 Introdução 35
3.2 Propagação de rumor 35
3.2.1 Abordagem mean-field para propagação de rumor em redes complexas *36*
3.3 Simulação estocástica da propagação em redes complexas 38

4 REDES NEURAIS 43
4.1 Introdução 43
4.1.1 Multilayer perceptron *45*
4.2 Redes Neurais para Grafos 47
4.2.1 Graph Neural Network (GNN) *49*
4.2.2 Graph Convolutional Networks (GCN) *50*
4.2.3 Representações vetoriais por GCN sem treinamento *51*
4.3 Representações para grafos inteiros 54
4.4 Funções Perda 54
4.4.1 Aprendizado supervisionado ou semi-supervisionado *54*
4.4.2 Aprendizado não-supervisionado *55*

5 RESULTADOS E DISCUSSÃO 57

5.1 Detecção da fonte de rumor em redes com estrutura de comunidades 57
5.1.1 Redes LFR com comunidades de tamanhos iguais 60
5.1.1.1 Impacto da quantidade de iterações da dinâmica 62
5.1.1.2 Impacto da fração inicial de propagadores 64
5.1.1.3 Impacto da taxa de propagação 65
5.1.1.4 Impacto da taxa de recuperação 66
5.1.2 Redes LFR com comunidades de tamanhos heterogêneos 67
5.1.2.1 Impacto da quantidade de iterações da dinâmica 68
5.1.2.2 Impacto da fração inicial de propagadores 69
5.1.2.3 Impacto da taxa de propagação 70
5.1.2.4 Impacto da taxa de recuperação 70
5.1.3 Comentários gerais e conclusão 71
5.2 Detecção de comunidades 72
5.2.1 Experimentos em redes LFR 76
5.2.1.1 Redes LFR com comunidades de tamanhos iguais 76
5.2.1.2 Redes LFR com comunidades de diversos tamanhos 80
5.2.2 Comentários gerais e conclusão 83

6 CONCLUSÃO 85

REFERÊNCIAS 87

CAPÍTULO

1

INTRODUÇÃO

1.1 Introdução

Sistemas complexos são compostos de diversos componentes que interagem entre si. Esses sistemas podem ser representados por redes em que os componentes do sistemas são os vértices e as arestas representam as interações dos componentes do sistema. Exemplos desses sistemas são: redes de interações entre proteínas (MARCOTTE *et al.*, 1999), interações entre pessoas (S; RM; RM, 1989), redes elétricas, internet, redes sociais entre outros (ALBERT; BARABÁSI, 2002). Esses sistemas geralmente possuem estrutura não homogênea de interações entre os seus componentes. Nesses sistemas podem ocorrer processos dinâmicos como por exemplo a propagação de doenças entre pessoas (LILJEROS; EDLING; AMARAL, 2001), troca de informação e conhecimento, propagação de vírus em computadores, entre outros. Por exemplo, numa rede social cada pessoa pode ser considerada um vértice e as arestas seriam as conexões de amizade que cada indivíduo tem com outros indivíduos, um processo dinâmico nesse contexto seria a propagação de rumores ou informações entre os indivíduos. Nesses sistemas complexos, principalmente nos de larga escala como redes sociais, é comum observar topologias complexas e estruturas extremamente heterogêneas nos grafos que os representam. Apesar de serem extremamente difíceis de modelar, os sistemas complexos atraíram atenção da comunidade científica dada a possibilidade de caracterizar fenômenos macroscópicos em termos da evolução dinâmica dos elementos básicos e iterações desses elementos no sistema (BARRAT; BARTHELEMY; VESPIGNANI, 2008).

Por outro lado, mais recentemente emergiram diversos trabalhos na tentativa de adaptar conceitos de redes neurais artificiais para dados estruturados como grafos (SCARSELLI *et al.*, 2009; KIPF; WELLING, 2017). Neste trabalho, procuramos utilizar as novas abordagens de redes neurais adaptadas para dados estruturados como grafos com aplicações aos sistemas complexos. Em particular, um grande problema que assola o mundo digital da atualidade é a propagação de rumores (CHOI *et al.*, 2020) em redes sociais que podem influenciar por exemplo

no resultado de eleições ou até mesmo na efetividade políticas públicas relacionadas a saúde das pessoas (BOVET; MAKSE, 2019). Motivado por estes problemas práticos que assolam a sociedade, um dos interesses do trabalho aqui presente é o de conseguir captar com precisão qual foi o grupo de indivíduos que começou a propagação de um rumor numa rede.

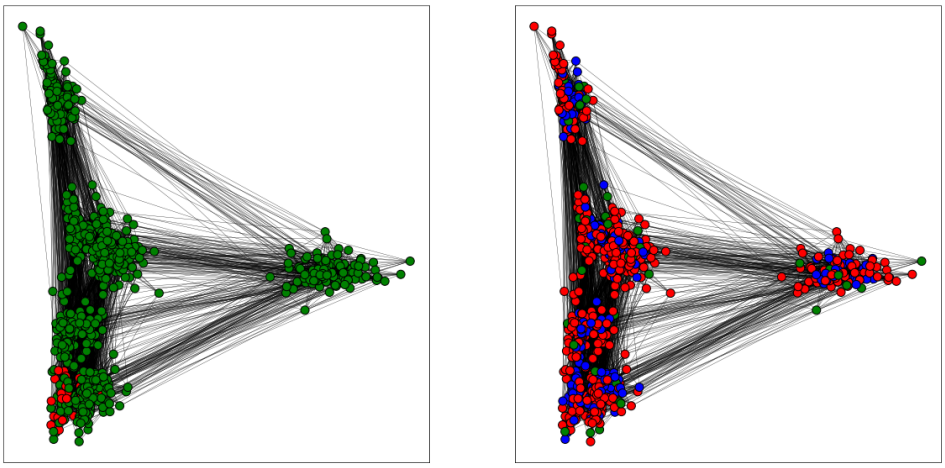

Figura 1 – Evolução da dinâmica em uma rede com estrutura de comunidades

Fonte: Elaborada pelo autor.

Para exemplificar o problema posto, na figura 1 é mostrado dois grafos. As cores dos vértices representam o estado em que o vértice está naquele momento, os vértices em verde estão no estado de ignorantes, ou seja, indivíduos que não sabem do rumor, os vértices em vermelho no estado de propagadores e os vértices em azul no estado de calados ou contidos. Observando o grafo da esquerda, boa parte dos vértices estão em verdes, ou seja, são ignorantes em relação ao rumor, já os vértices em vermelho são indivíduos que começam a propagar um rumor a partir de um agrupamento/comunidade do grafo.

No grafo representado na direita se passou um período de tempo e o rumor se propagou entre boa parte dos indivíduos do grafo, portanto se observa muitos vértices em vermelho, enquanto alguns permanecem em verde (ignorantes) e outros em azul (calados). O problema de interesse é: Dado o momento em que o rumor já se alastrou pela rede social, será que é possível determinar qual o agrupamento de indivíduos que começou a propagá-lo?

Outro problema de grande interesse na comunidade científica de redes complexas é o de encontrar agrupamentos de vértices em grafos. Esses agrupamentos podem ser chamados de comunidades e são definidos por grupos de vértices que se conectam mais entre si e menos com vértices de outro agrupamento ou comunidade.

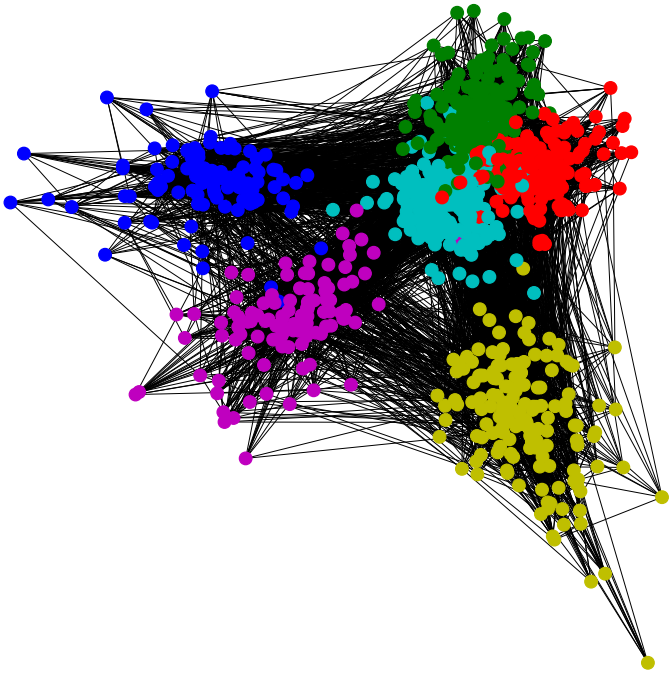

Figura 2 – Grafo artificial com estrutura de comunidade

Fonte: Elaborada pelo autor.

Na figura 2 é mostrado um grafo artificial que possui estruturas de comunidades. Os vértices que são da mesma comunidade estão pintados com a mesma cor. Neste exemplo o grafo possui um total de seis comunidades e um bom algoritmo de detecção de comunidades consegue detectá-las com precisão.

As contribuições desse trabalho podem ser resumidas da seguinte maneira: Em uma primeira aplicação de modelos de redes neurais para grafos mostramos que é possível recuperar a fonte de rumor partindo de comunidade em uma dinâmica de propagação, mesmo só tendo a informação de um instante em particular. Esta abordagem pode ser usada para reduzir o espaço de busca, usando pouca informação, dos propagadores iniciais de uma dinâmica em redes complexas com quantidade de vértices extremamente grande, como é o caso de redes sociais. Em um segundo momento exploramos o potencial de representação das redes neurais para grafos em detecção de comunidades a partir do conceito de Graph Auto-Encoders(GAE) (KIPF; WELLING, 2016) e também comparamos com algoritmos tradicionais de redes complexas.

1.2 Organização do trabalho

O restante trabalho está organizado da seguinte maneira:

- No capítulo 2 é introduzida a teoria de redes complexas. Incluindo assim, noções básicas de grafos, medidas de centralidade e modelos de redes.
- No capítulo 3 é abordada a modelagem matemática de processos de propagação em redes complexas e apresentamos uma abordagem para a simulação estocástica de processos de propagação.
- No capítulo 4 é introduzida uma discussão sobre redes neurais e em seguida a adaptação para dados estruturados como grafos.
- No capítulo 5 é apresentado as análises e resultados da pesquisa desenvolvida. Em um primeiro momento foi analisado o potencial de redes neurais para grafos para recuperar a origem de uma dinâmica. Em seguida, é verificado o potencial do aprendizado de representações para os vértices de grafos com o objetivo de detecção de comunidades em redes complexas.
- No capítulo 6 é apresentado a conclusão e possíveis investigações futuras.

CAPÍTULO

2

REDES COMPLEXAS

2.1 Teoria das redes complexas

A teoria das redes complexas é baseada na abstração matemática de grafo, durante o texto usaremos as palavras grafo e rede de forma a denotar o mesmo objeto.

2.1.1 Definições e conceitos

Um grafo $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ é definido por um conjunto de vértices/nós $\mathcal{V}$ e um conjunto de arestas/links $\mathcal{E}$ entre os vértices. Uma aresta que conecta o nó $u \in \mathcal{V}$ ao nó $v \in \mathcal{V}$ é denotada pelo par ordenado $(u, v) \in \mathcal{E}$. Usaremos N para denotar a quantidade de vértices no grafo, ou seja, $N = |\mathcal{V}|$. Uma forma de representar os grafos é por meio de uma matriz de adjacências $\mathbf{A}$, que é uma matriz de dimensão $N \times N$ onde $A_{ij} = 1$ se $(i, j) \in \mathcal{E}$ e 0 caso contrário.

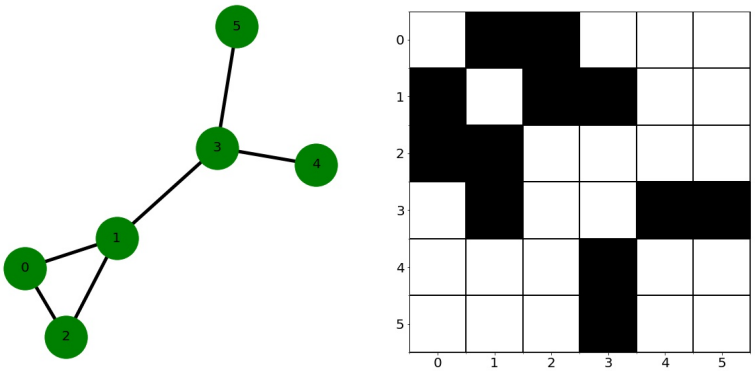

Figura 3 – Grafo e sua representação em matriz de adjacências

Fonte: Elaborada pelo autor.

Na Figura 3 está representado um grafo e a sua representação por matriz de adjacência, onde os quadrados pretos representam 1 e os brancos 0. Neste exemplo o grafo é não direcionado, ou seja, se $(i, j) \in \mathcal{E}$ e $(j, i) \in \mathcal{E}$. No entanto, existem também outros tipos de grafos que podem ser definidos, como por exemplo os grafos direcionados.

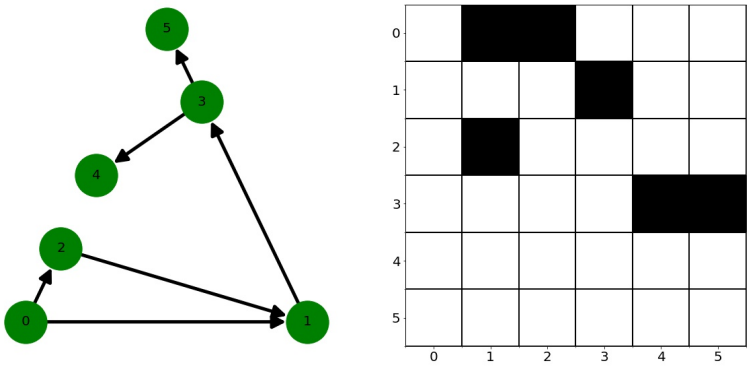

Figura 4 – Grafo direcionado e sua representação em matriz de adjacências

Fonte: Elaborada pelo autor.

Na Figura 4 é apresentado um exemplo de grafo direcionado e sua matriz de adjacências. Neste caso, a matriz de adjacência não é simétrica como no caso da Figura 3. Além dos grafos direcionados e não-direcionados, pode-se definir também grafos com pesos nas arestas, ou seja, em que o valor da conexão não é binário e sim um número real, geralmente positivo. Ademais, existem redes multilayer em que se teriam várias camadas de redes (COZZO *et al.*, 2018). No entanto, neste trabalho foram utilizadas redes não direcionadas. Na seção a seguir caracterizamos algumas medidas e propriedades para esse tipo de rede.

2.1.2 Medidas de centralidade e propriedades de redes complexas

Existem diversas informações contidas nos vértices que podem ser caracterizadas a partir de medidas. Uma medida básica nesse contexto é o grau que é simplesmente a quantidade de conexões ou vizinhos que um vértice possui e pode ser definido da seguinte maneira:

$$k_i = \sum_{j=1}^{N} A_{ij} \tag{2.1}$$

Tendo essas medidas para cada vértice é possível então definir uma distribuição empírica para os graus da rede. Sendo assim, define-se p_k a fração de vértices que possuem grau k e assim

obtém-se a distribuição (NEWMAN, 2003).

$$p_k = \frac{N_k}{N} \tag{2.2}$$

Sendo N_k o número de vértices que possuem grau k.

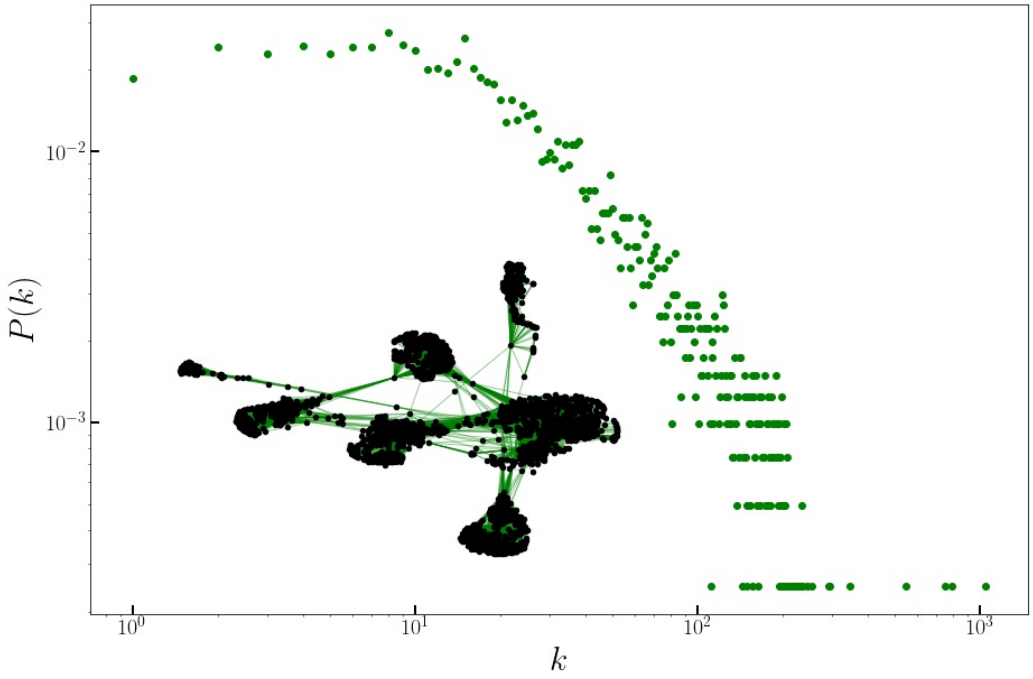

Figura 5 – Rede ego-facebook coletada de alguns participantes de uma pesquisa na plataforma do facebook e a sua respectiva distribuição de grau, Fonte dos dados.

Fonte: Elaborada pelo autor.

Observa-se na Figura 5 um heterogeneidade de grau muito alta de conexões na rede de amostra do facebook pela distribuição de grau. A escala do gráfico está em log-log e pode-se observar uma tendência a seguir uma lei de potência ($P(k) \sim k^{-\gamma}$).

Uma medida de interesse global para rede pode ser definida a partir dos graus, sendo o grau médio da rede:

$$\langle k \rangle = \frac{1}{N} \sum_{i=1}^{N} k_i \tag{2.3}$$

Além do grau existem diversas outras medidas para caracterizar uma rede, não abordaremos com detalhe, mas é possível encontrar uma revisão em (COSTA *et al.*, 2007).

2.1.3 Modelos de Redes

2.1.3.1 Modelo de Erdős–Rényi

O modelo de Erdős–Rényi (ER) foi o primeiro modelo de redes aleatórias proposto por Erdős e Rényi (ERDOS; RENYI, 1959). Na sua formulação original para criação de um grafo aleatório não direcionado $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ são inicializados $N = |\mathcal{V}|$ vértices em que as arestas entre todos os pares acontecem com probabilidade p e não acontecem com probabilidade $1 - p$. Como são N vértices, existe um total de $N(N-1)/2$ possibilidades de arestas e a probabilidade de observar um grafo $\mathcal{G}$ com N vértices e E arestas é dada por:

$$P(\mathcal{G}_{N,E}) = p^E (1-p)^{\frac{1}{2}N(N-1)-E} \tag{2.4}$$

Além disso, nessa construção o número médio de arestas geradas é dado por $\langle E \rangle = \frac{1}{2}N(N-1)p$. Como cada aresta contribui para o grau de dois vértices, o grau médio da rede é dado por (PASTOR-SATORRAS, 2007):

$$\langle k \rangle = \frac{2\langle E \rangle}{N} = (N-1)p \tag{2.5}$$

Também é possível obter uma expressão para a distribuição de grau do modelo ER:

$$P(k) = \binom{N-1}{k} p^k (1-p)^{N-1-k} \tag{2.6}$$

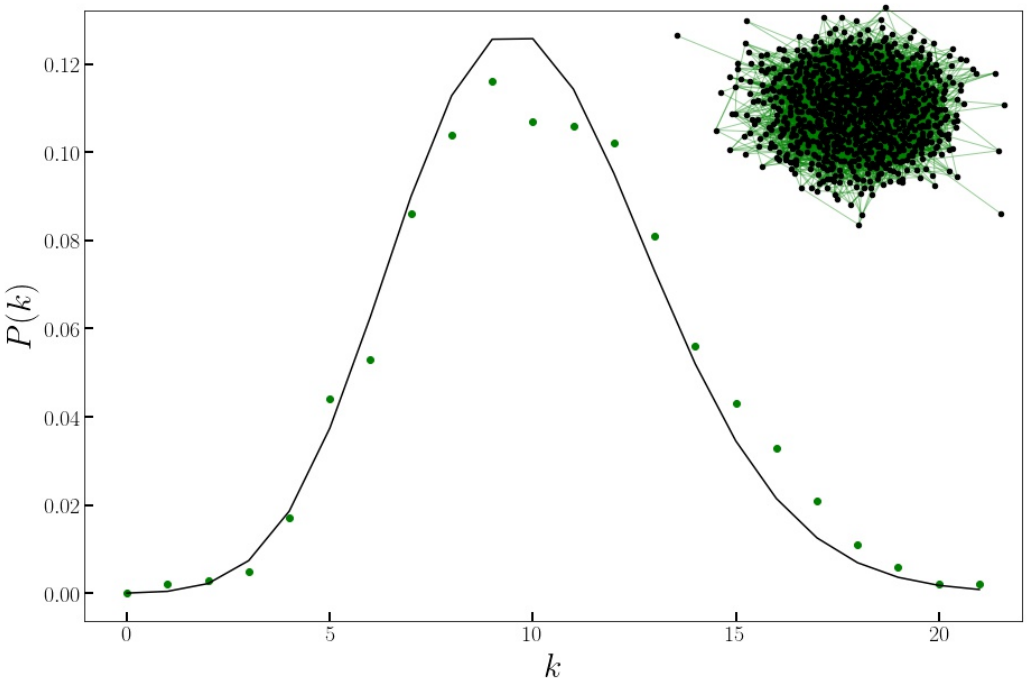

Figura 6 – Distribuição do grau de uma realização do modelo de rede ER

Fonte: Elaborada pelo autor.

Na Figura 6 é mostrado a distruição de grau de uma realização da rede ER para $\langle k \rangle = 10$, $p \approx 0.1$ e $N = 1000$.

2.1.3.2 Modelo Barabási–Albert

Muitas redes reais como a rede da Figura 5 apresentam uma distribuição do grau heterogênea com a essência de que alguns poucos vértices da rede possuem grau muito elevado em comparação ao restante. O modelo Barabási-Albert (BA) (ALBERT; BARABÁSI, 2002) é uma formulação para tentar simular redes com essas características observadas nas redes reais. A ideia por trás do método é pensar num processo dinâmico de formação dessas redes reais em que as novas arestas não são alocadas de forma aleatória e sim tendem a se conectar com vértices cujo o grau já é elevado. Para geração de redes BA se começa com um número m_0 de vértices conectados. A cada passo de tempo um novo vértice é adicionado com $m < m_0$ arestas conectadas com os vértices antigos do sistema. Essas novas arestas são conectadas com os antigos vértices do sistema com uma probabilidade proporcional aos graus dos vértices existentes no sistema, ou seja, a probabilidade de que o novo nó se conecte a um vértice i é dada por:

$$p_i = \frac{k_i}{\sum_j k_j} \tag{2.7}$$

É possível demonstrar que usando essas regras a rede gerada possui uma distribuição de grau lei de potência, $P(k) \sim k^{-3}$ (PASTOR-SATORRAS, 2007) .

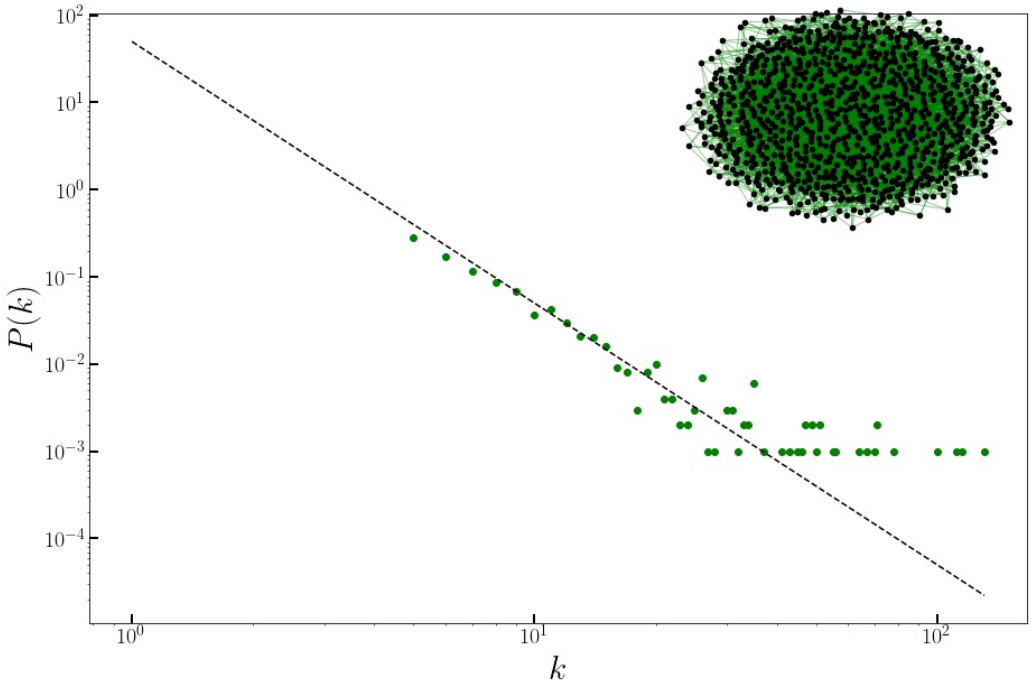

Figura 7 – Distribuição do grau de uma realização do modelo de rede BA

Fonte: Elaborada pelo autor.

Na figura 7 é mostrado a distribuição de grau de uma realização do modelo BA com $\langle k \rangle = 10$ e $N = 1000$.

2.1.3.3 Modelo de configuração

O modelo de configuração (CATANZARO; NÁ; PASTOR-SATORRAS, 2005) surgiu da necessidade de gerar redes heterogêneas que seguem alguma distribuição de grau pré-estabelecida. O método na sua formulação original funciona da seguinte maneira: Dado a quantidade de vértices N e a distribuição do grau $P(k) \sim k^{-\gamma}$

- Cada vértice i começa não conectado com ninguém e será associado a um grau k_i extraído da distribuição de probabilidade $P(k) \sim k^{-\gamma}$ e sujeito as restrições $k_{min} \leq k_i \leq N^{1/2}$ e $\sum_i k_i$ par de tal forma a evitar que sobre arestas desconectadas. Feito isto, se tem uma lista com os índices dos vértices e seus respectivos graus.

- A rede então é construída conectando vértices aleatoriamente de tal forma a ter $\sum_i k_i/2$ arestas no total e respeitando os graus estabelecidos anteriormente, além evitar múltiplas e auto-conexões.

2.1.3.4 Modelo de Lancichinetti–Fortunato–Radicchi

Muitas redes complexas apresentam estrutura de comunidades, ou seja, elementos de um subconjunto dos vértices que estão mais conectados entre si e menos com os outros subconjuntos. Devido a existência desse tipo de estrutura em redes reais, se tornou necessário algum modelo que simule características similares as redes observadas. Além disso, existem diversos algoritmos utilizados para encontrar esses módulos presentes na rede complexa. Uma maneira de comparar a performance dos métodos é utilizando algum modelo de referência (YANG; ALGESHEIMER; TESSONE, 2016). O modelo de Lancichinetti–Fortunato–Radicchi (LFR) (LANCICHINETTI; FORTUNATO; RADICCHI, 2008) foi criado pra suprir essas necessidades, tanto de simular redes similares as reais, quanto para comparação de métodos de detecção de comunidades. Uma implementação adaptada do modelo de LFR funciona da seguinte maneira (NetworkX developer team, 2014), o usuário dá como entrada o número de nós N, τ_1, τ_2, μ e o grau médio da rede $\langle k \rangle$ ou k_{min} e k_{max} e então são executados os seguintes passos:

- Cada nó do grafo tem um grau que é escolhido a partir de uma distribuição lei de potência com expoente τ_1. Caso o usuário não forneça, os extremos da distribuição do grau (k_{min} e k_{max}) são escolhidos de tal forma que o grau médio seja $\langle k \rangle$. Além disso, cada nó u terá uma quantidade de arestas $(1-\mu)k_u$ com vértices da mesma comunidade e μk_u com vértices de outras comunidades.

- Gerar tamanhos de comunidade de acordo com uma lei de potência com expoente τ_2. Se o usuário especificar min_c e max_c esses serão os intervalos dos tamanhos de comunidade geradas. Se não, os tamanhos mínimos e máximos serão o min_k e max_k da distribuição do grau. Além disso, as comunidades terão o tamanho de tal forma que a soma da quantidade de vértices delas seja igual a N.

- Cada nó é rotulado aleatoriamente a uma comunidade dentre as possíveis com a condição de que a comunidade tenha tamanho grande o suficiente para que a quantidade de arestas dentro da mesma comunidade $(1-\mu)k_u$.
- Cada nó u terá uma fração de arestas $(1-\mu)$ do seu grau com nós da sua comunidade e o restante com nós de outra comunidade, essa atribuição é feita de forma similar ao modelo de configuração.

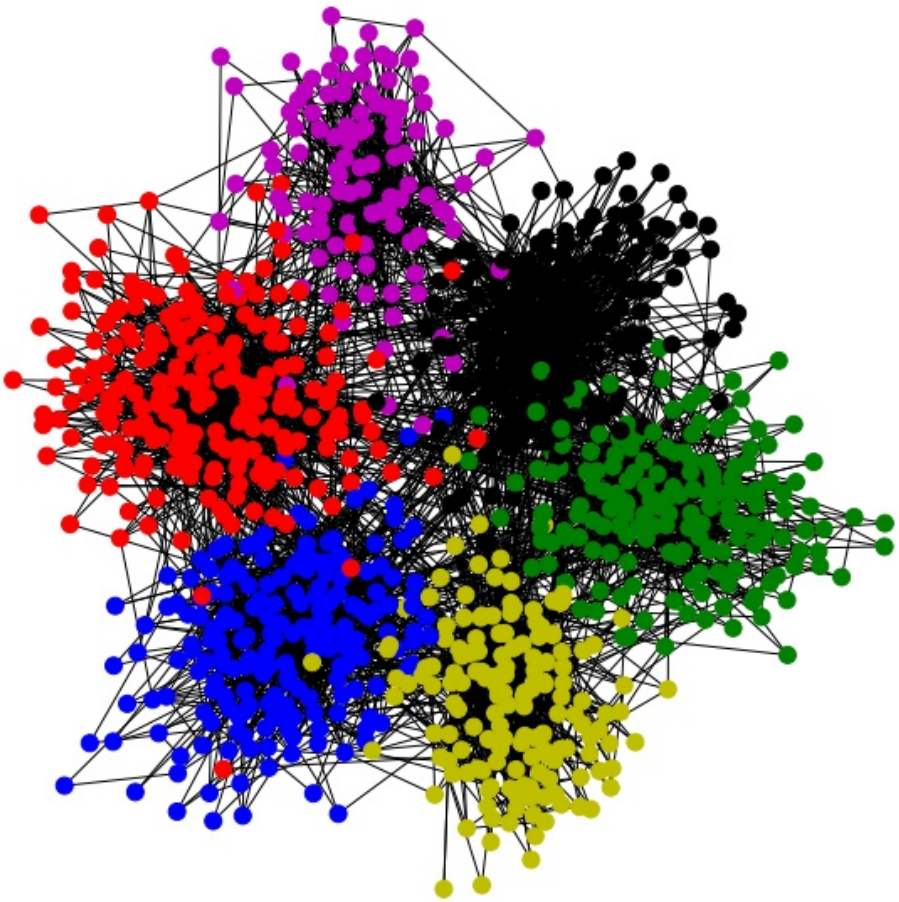

Figura 8 – Exemplo de uma realização do modelo LFR

Fonte: Elaborada pelo autor.

Na Figura 8 é mostrado um exemplo do modelo LFR para os parâmetros $N = 1000$, $\tau_1 = 1.5$, $\tau_2 = 3$, $k_{min} = 2$, $k_{max} = 30$, $min_c = 100$, $max_c = 250$ e $\mu = 0.10$. Os vértices estão coloridos de acordo com a comunidade que pertencem.

CAPÍTULO

3

MODELAGEM MATEMÁTICA E SIMULAÇÃO DE PROCESSOS DINÂMICOS EM REDES COMPLEXAS

3.1 Introdução

Para se entender redes complexas, a comunidade científica da área se concentra boa parte em entender a estrutura da rede e possíveis processos dinâmicos ocorrendo nessa rede. Muitos estudos surgem no sentido de compreender o impacto da estrutura no processo dinâmico e vice-versa. Exemplos de processos dinâmicos são propagação de doenças entre pessoas, propagação de rumores em redes sociais, falhas em cascatas, entre outros (BARRAT; BARTHELEMY; VESPIGNANI, 2008). Neste capítulo será introduzido sobre o tema da modelagem matemática da propagação de rumor e da simulação desse processo dinâmico de maneira estocástica em redes.

3.2 Propagação de rumor

Os primeiros modelos matemáticos para descrição do fenômeno de propagação de rumor/informação foram introduzido por Daley e Kendall (DALEY; KENDALL, 1964) e Maki e Thompson (MAKI; THOMPSON, 1973), sem considerar a estrutura de rede para a população. Para estes modelos é considerado uma abordagem semelhante a usada no contexto de propagação de doenças infecciosas. Considerando que os indivíduos da população podem estar em três diferentes estados: X - *Ignorantes*, Y - *Propagadores*, Z - *Calados*.

Em que X são os indivíduos da população que não ouviram/receberam o rumor, Y são os indivíduos que já receberam o rumor e podem propagar e Z os indivíduos que já sabem do rumor mas não estão dispostos a propagá-lo.

Sendo assim, são definidas as seguintes transições:

$$Y + X \xrightarrow{\lambda} Y + Y \tag{3.1}$$

$$Y + Y \xrightarrow{\alpha} Z + Y \tag{3.2}$$

$$Y + Z \xrightarrow{\alpha} Z + Z \tag{3.3}$$

Ou seja, na primeira regra (3.1) dois indivíduos no estado Y e estado X entram em contato e com uma taxa λ o indíviduo no estado X muda para o estado Y. De maneira similar pode-se definir as regras mostradas em (3.2) e (3.3).

3.2.1 Abordagem mean-field para propagação de rumor em redes complexas

Um dos primeiros trabalhos sobre a dinâmica de propagação de rumor em redes complexas foi introduzido por (MORENO; NEKOVEE; PACHECO, 2004). A abordagem mostrada a seguir tem origem nesses estudos. Suponha que existem N nós na rede, cada um desses nós podem estar em três estados diferentes, como definido anteriormente em 3.2. Considerando um sistema homogêneo, ou seja, em que todos os nós tem o mesmo grau pode-se descrever o modelo original de (MAKI; THOMPSON, 1973) em termos da densidade de indivíduos em cada estado, ou seja:

$$x(t) + y(t) + z(t) = 1 \tag{3.4}$$

A equação é definida a partir das proporções de indivíduos em um estado no tempo t. Sendo que $Y(t)$ é a quantidade de indivíduos no estado de *propagador* no tempo t, analogamente $X(t)$ é a quantidade de indivíduos no estado de *ignorante* no tempo t e por fim $Z(t)$ é a quantidade de indivíduos no estado de *calado* no tempo t. Sendo assim, as proporções são: $y(t) = Y(t)/N$, $x(t) = X(t)/N$, $z(t) = Z(t)/N$, em que N é a quantidade total de indivíduos no sistema.

É possível definir um conjunto de equações diferenciais que descreve a evolução do sistema ao longo do tempo:

$$\frac{dx(t)}{dt} = -\lambda \langle k \rangle x(t) y(t) \tag{3.5}$$

$$\frac{dy(t)}{dt} = \lambda \langle k \rangle x(t) y(t) - \alpha \langle k \rangle y(t)[y(t) + z(t)] \tag{3.6}$$

$$\frac{dz(t)}{dt} = \alpha \langle k \rangle y(t)[y(t) + z(t)] \tag{3.7}$$

A equação (3.5) descreve $x(t)$ decresce de forma proporcional a λ e o grau médio $\langle k \rangle$ e a fração de $x(t)$ e $y(t)$. A equação (3.6) descreve como a fração de *Propagadores* $y(t)$ aumenta de acordo com uma taxa proporcional a λ, $\langle k \rangle$ e as densidades $x(t)$ e $y(t)$. Por outro lado a fração de *Propagadores* $y(t)$ decresce de acordo com a própria classe a uma taxa de $\alpha \langle k \rangle$ multiplicado por $1 - x(t) = y(t) + z(t)$ e além disso, a proporção de indivíduos no estado Z cresce a uma taxa proporcional a α e ao grau médio $\langle k \rangle$ e aos indivíduos nos estados Y e Z (MORENO; NEKOVEE; PACHECO, 2004).

Note que o modelo descrito pelas equações (3.5), (3.6) e (3.7) é simplificado e parte da suposição de homogeneidade da rede em que a propagação está acontecendo, no entanto, mesmo com essa suposição pode-se obter bons resultados de predição sobre o futuro do estado do sistema em casos de redes mais heterogêneas, em que a suposição de que os vértices tem uma quantidade de conexões aproximadamente igual ao grau médio não vale.

3.3 Simulação estocástica da propagação em redes complexas

Para a simulação do processo dinâmico na estrutura de rede complexa será utilizada uma abordagem inspirada pelo algoritmo de Gillespie (GILLESPIE, 1976). Essa abordagem utilizada é adaptada da apresentada no artigo de (ARRUDA; RODRIGUES; MORENO, 2018) e que tem como inspirações o trabalho de (COTA; FERREIRA, 2017). Dada uma condição inicial para o processo, ou seja, os estados dos vértices no inicio do processo, o incremento de tempo é dado por:

$$\Delta t = \frac{1}{\lambda N_k + \alpha N_k} \tag{3.8}$$

Sendo N_k a quantidade de graus somados de todos os indivíduos no estado de *Propagadores*. Com probabilidade:

$$p_{propagar} = \frac{\lambda}{\lambda + \alpha} \tag{3.9}$$

Um vértice no estado de *Propagador* é escolhido proporcionalmente ao grau e transmite o rumor/informação para um vizinho escolhido de forma uniforme aleatória. E com a probabilidade complementar:

$$p_{calar} = \frac{\alpha}{\alpha + \lambda} \tag{3.10}$$

Um *Propagador* escolhido de forma proporcional ao seu grau se torna *Calado* se um vizinho seu escolhido de forma uniforme estiver no estado de *Calado* ou *Propagador*. Feito isto, esse conjunto de regras é iterado até um número de passos (ou tempo máximo) estipulado previamente ou até atingir o sistema atingir o estado absorvente, ou seja, não restar mais nenhum *Propagador* para transmitir a contaminação.

Um pseudo-código para a dinâmica é mostrado a seguir:

Algoritmo 1 – Simulação Maki-Thompson

Entrada: Grafo $\mathcal{G}$; Número inicial de propagadores N_i; λ; α; Quantidade de passos máximo $step_{max}$.

Saída: A dinâmica até o passo de tempo estipulado ou até não ter nenhum Propagador

$j \leftarrow 0$
enquanto $j \leq N_i$ **faça**:
    Escolhe aleatoriamente um vértice para ter o estado de propagador.
    $j \leftarrow j + 1$
fim enquanto
$time \leftarrow 0$
$steps \leftarrow 0$
enquanto $N_i > 0$ e $steps \leq step_{max}$ **faça**
    Gere um número uniforme entre 0 e 1 e salve em $rand_{prob}$
    $p_{progagar} \leftarrow \frac{\lambda}{\lambda+\alpha}$
    se ($p_{progagar} > rand_{prob}$) **então**
        Escolha um vértice *Propagador* aleatório de forma proporcional ao grau. Depois de ter escolhido o vértice escolhe um vizinho aleatoriamente de forma uniforme. Se esse vizinho escolhido for *Ignorante*, então ele se torna *Propagador*
        $N_i \leftarrow N_i + 1$
    senão
        Escolha um vértice *Propagador* aleatório de forma proporcional ao grau. Depois de ter escolhido o vértice escolhe um vizinho aleatoriamente de forma uniforme. Se o vizinho escolhido for *Propagador* ou *Calado*, então o vértice que tentou transmitir se torna *Calado*.
        $N_i \leftarrow N_i - 1$
    fim se
    $time \leftarrow time + \frac{1}{\lambda N_k + \alpha N_k}$ ▷ N_k é a quantidade de graus somadas dos *Propagadores*
    $steps \leftarrow steps + 1$
fim enquanto

Com o objetivo de checar a qualidade da simulação estocástica na rede complexa consideramos uma rede regular (com todos os vértices com o mesmo grau) de $N = 1000$ e $\langle k \rangle = 10$. Para integração numérica das equações diferenciais foi considerado $\langle k \rangle = 10$ com as condições iniciais de $y(0) = 0.4$, $x(0) = 0.6$ e $z(0) = 0$ e as taxas de $\lambda = 0.5$ e $\alpha = 0.2$.

Com essa simulação pretende-se verificar se a simulação estocástica da propagação está de acordo com o modelo teórico de evolução da dinâmica Maki-Thompson proposta por (MORENO; NEKOVEE; PACHECO, 2004). Consideramos utilizar uma rede regular para esta verificação, pois na rede regular a hipótese de homogeneidade do número de conexões dos vértices vale e portanto as equações do modelo Mean-field funcionam melhor.

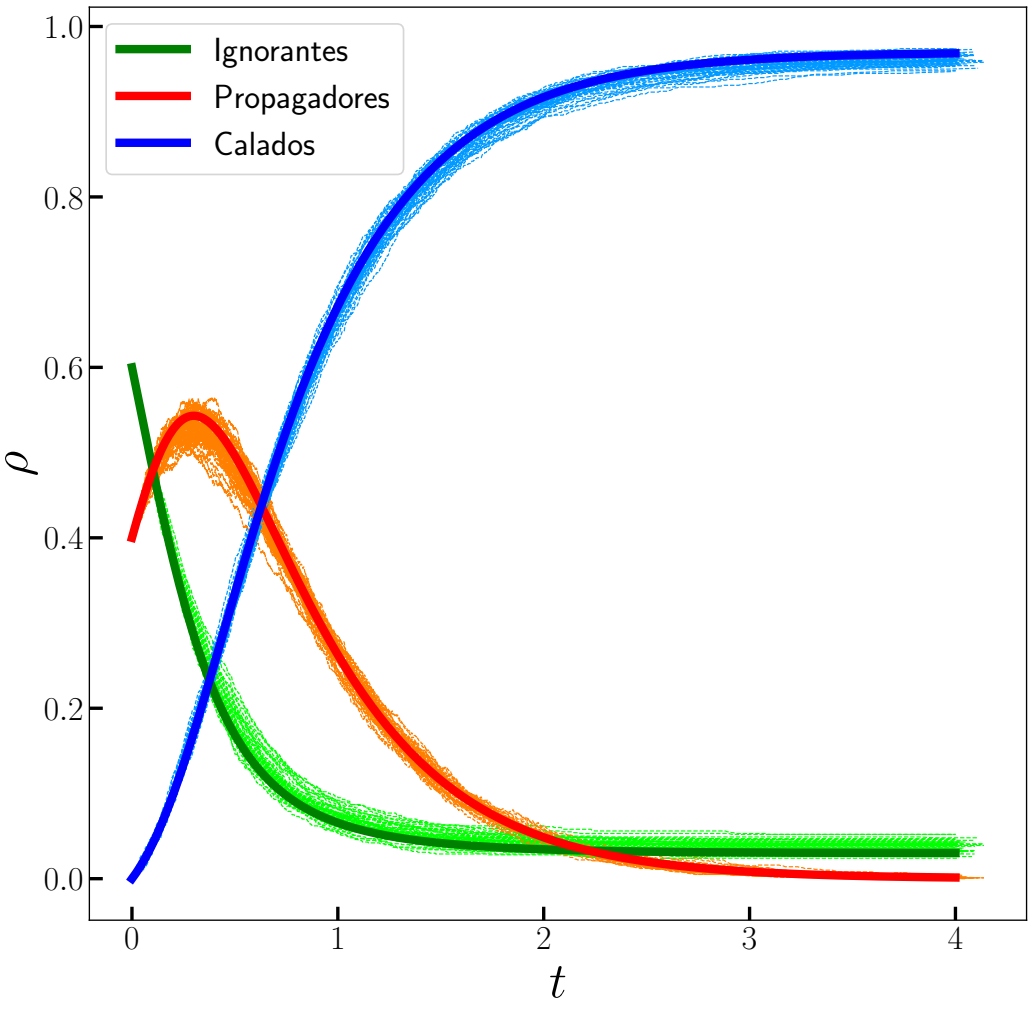

Figura 9 – Integração numérica do sistema de equações diferenciais mean-field e a simulação estocástica na rede regular

Fonte: Elaborada pelo autor.

Na Figura 9 foi considerada a integração numérica das equações mean-field em uma rede regular. As cores mais fortes representam a integração numérica enquanto as cores mais fracas representam uma realização da simulação estocástica. Foram feitas 50 simulações a partir das mesmas condições. Ademais, em cada simulação são escolhidos aleatoriamente 40% dos vértices para serem *propagadores* e as taxas são $\lambda = 0.5$ e $\alpha = 0.2$. Podemos observar que a simulação estocástica da propagação na rede se ajusta bem a integração numérica das equações diferenciais mean-field.

Na figura 9 foi considerado uma quantidade inicial de 40% vértices para iniciarem a propagação no estado de *propagador*, no entanto, com uma pequena alteração do código de simulação da dinâmica é possível escolher quais são os vértices que vão começar a propagação ao invés de escolher aleatoriamente.

CAPÍTULO

4

REDES NEURAIS

4.1 Introdução

As redes neurais artificiais são como modelos matemáticos inspirados pelo funcionamento do cérebro. Uma rede neural artificial mais simples, de uma maneira geral, pode ser entendida como uma composição de transformações afins seguidas de uma função de ativação não-linear. Neste modelo mais simples o objetivo é construir uma função que consiga receber um vetor de entrada que é caracterizado por variáveis importantes de algum estudo e a partir disso dar como saída, após algumas transformações, uma classificação ou regressão por exemplo. As variáveis preditoras do estudo podem ser pixels de imagens, características financeiras de uma pessoa, características biológicas de uma pessoa, variáveis sobre comportamento de pessoas online, entre outras coisas e as variáveis resposta pode ser se uma foto é um cachorro ou gato, a renda de uma pessoa, se a pessoa tem diabetes ou não ou se uma pessoa compraria ou não algum produto. Do ponto de vista de redes neurais o problema é resolvido a partir do aprendizado conjunto de *representações* e da tarefa proposta sendo ela regressão ou classificação. Um exemplo interessante sobre a questão da importância da representação dos dados é apresentado no livro (GOODFELLOW; BENGIO; COURVILLE, 2016) em que o objetivo é encontrar uma reta que consegue separar bem um conjunto de dados composto de duas classes. Suponha um conjunto de dados em que as observações podem pertencer a duas classes distintas, a vermelha ou a verde. Gerando dados artificiais em duas dimensões temos o seguinte gráfico:

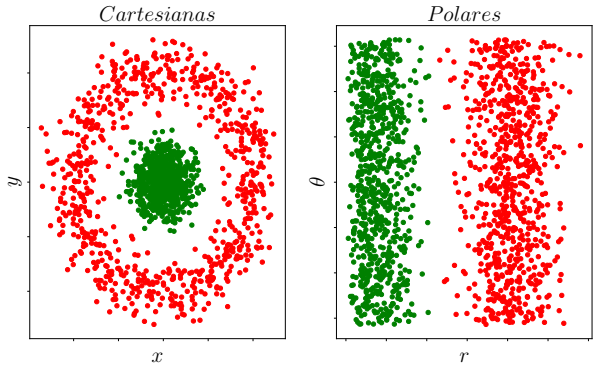

Figura 10 – Exemplo da representação de um conjunto de dados com duas classes(vermelhos e verdes) em diferentes sistemas de coordenadas. (Figura inspirada em um exemplo do livro de Goodfellow (GOODFELLOW; BENGIO; COURVILLE, 2016))

Fonte: Elaborada pelo autor.

Observa-se que na figura 10 a representação cartesiana (da esquerda) não é possível encontrar uma reta que consegue dividir os pontos vermelhos dos pontos verdes. No entanto, se transformarmos estes dados para um sistema de coordenadas polares a tarefa de separá-los se torna muito mais fácil, neste caso uma reta vertical conseguiria facilmente separar os pontos vermelhos dos verdes.

Essencialmente o que uma rede neural artificial simples faz é aplicação de funções para transformação de tal forma a obter diferentes representações para o conjunto de dados e com isso conseguir performar uma tarefa. Exemplos de aplicações são em reconhecimento de dígitos escritos a mão (LECUN *et al.*, 1998), classificação de imagens (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), processamento del linguagem natural (BENGIO *et al.*, 2003), dentre outras.

Neste capítulo introduzimos os modelos clássicos de redes neurais artificiais e também a adaptação para dados estruturados como grafos.

4.1.1 Multilayer perceptron

Como uma extensão da ideia do perceptron (ROSENBLATT, 1961) surgiu o Multilayer Perceptron (MLP) para tarefas em que não se é possível ter uma separação linear no espaço das variáveis (VAPNIK, 2000).

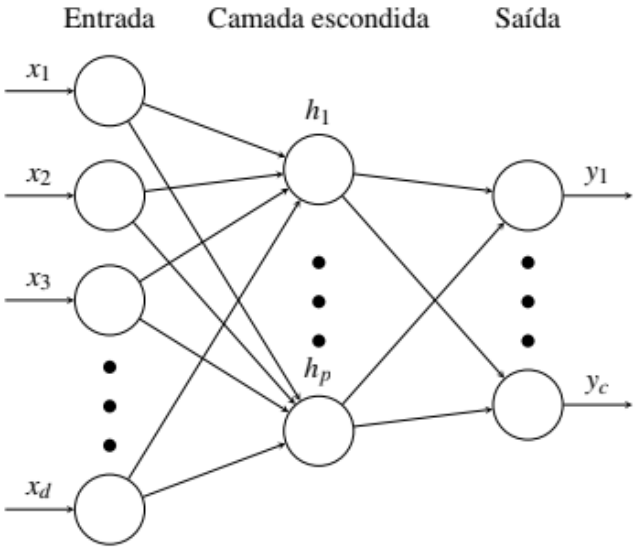

Figura 11 – Representação esquemática de uma MLP

Fonte: Elaborada pelo autor.

Formalmente no exemplo da Figura 11 temos uma função $f : \mathbb{R}^d \to \mathbb{R}^c$ que mapeia um vetor de entrada de dimensão d em um vetor de saída de dimensão c. Esse mapeamento acontece via transformações afins seguidas de uma função não-linear. Mais especificamente, no exemplo da figura 11 teríamos:

$$\mathbf{y} = \sigma_2(\mathbf{b}^{(2)} + \mathbf{W}^{(2)}(\sigma_1(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)})) \tag{4.1}$$

Onde $\mathbf{W}^{(1)} \in \mathbb{R}^{d \times p}$ é uma matriz de parâmetros ajustáveis e $\mathbf{b}^{(1)} \in \mathbb{R}^p$ é um vetor de parâmetros ajustáveis chamado viés e $\sigma_1(.)$ uma função não-linear que atua elemento a elemento. Portanto, na camada oculta temos:

$$\mathbf{h} = \sigma_1(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}) \tag{4.2}$$

Ademais, para obter $\mathbf{y}$ em função da representação $\mathbf{h}$ da camada escondida:

$$\mathbf{y} = \sigma_2(\mathbf{W}^{(2)}\mathbf{h} + \mathbf{b}^{(2)}) \tag{4.3}$$

Em que $\mathbf{W}^{(2)} \in \mathbb{R}^{p \times c}$ e $\mathbf{b}^{(2)} \in \mathbb{R}^p$ são parâmetros ajustáveis e $\sigma_2(.)$ uma função de ativação.

Como função de ativação pode-se usar por exemplo a $ReLU(x) = max(0,x)$, $sigmoid(x) = 1/(1+e^{-x})$ entre outras.

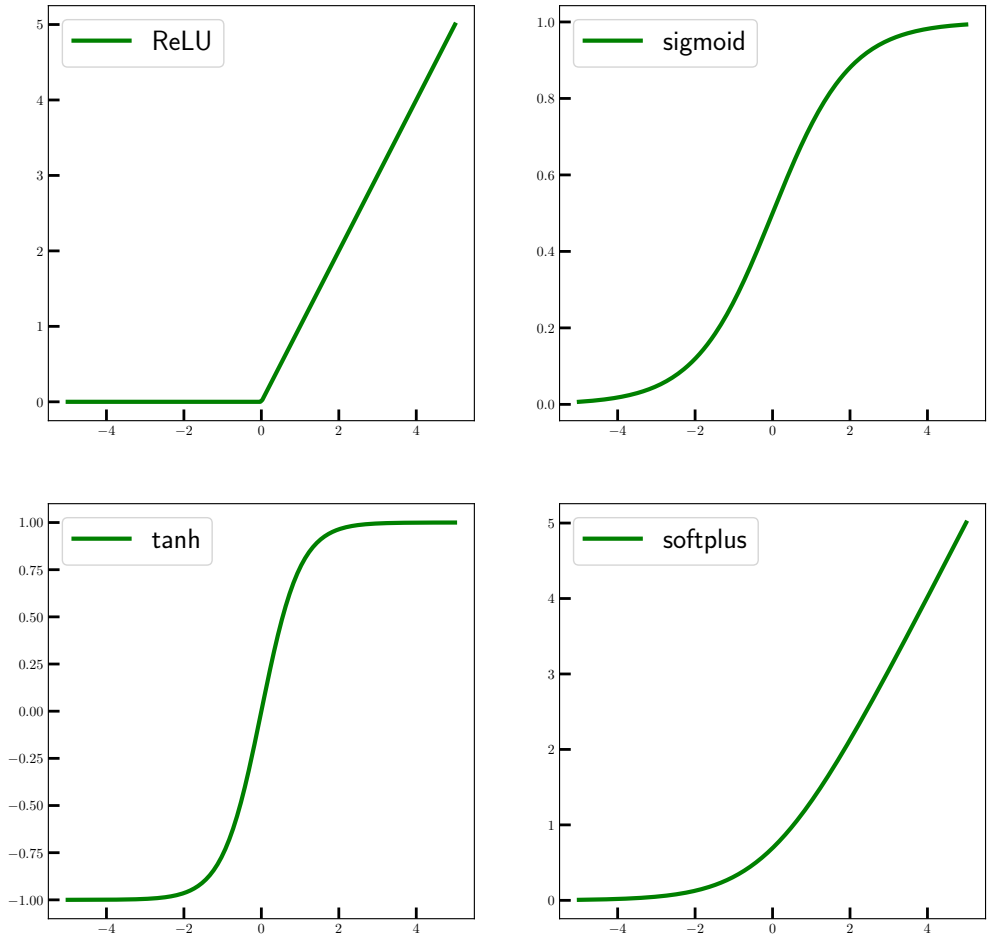

Figura 12 – Exemplos de diferentes funções de ativação

Fonte: Elaborada pelo autor.

O conjunto de parâmetros ajustáveis do modelo é dado por: $\theta = \{\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{b}^{(1)}, \mathbf{b}^{(2)}\}$. Para treinar os parâmetros deve-se definir uma função perda $\mathcal{L}$ que quantifica o quão bom está a predição. Por exemplo, se o problema for classificação pode-se usar como função perda a entropia cruzada. Para isso, considere um conjunto de amostras de treinamento $\mathcal{T} = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_n, y_n)\}$ em que $\mathbf{x}_i \in \mathbf{R}^d$ e $y_i \in \mathcal{C}$ em que $\mathcal{C}$ é o conjunto das possíveis classes. Supondo um vetor de entrada $\mathbf{x}_i$, ou seja, uma observação em particular e que desejamos obter uma classe predita para essa observação, deve-se transformar a saída em um vetor de probabilidades.

Para isto, usando a função softmax temos:

$$\hat{p}_{ij} = \frac{e^{y_{ij}}}{\sum_{k=1}^{|\mathcal{C}|} e^{y_{ik}}} \tag{4.4}$$

Em que $\hat{p}_{ij}$ é a probabilidade da observação i ser da classe j, para $j = 1, \cdots, |\mathcal{C}|$.

Como função perda pode-se utilizar a entropia cruzada:

$$\mathcal{L} = -\frac{1}{|\mathcal{T}|} \sum_{i=1}^{|\mathcal{T}|} \sum_{j \in \mathcal{C}} \mathbb{1}(y_k = j) log(\hat{p}_{ij}) \tag{4.5}$$

Inicialmente o conjunto de parâmetros θ é inicializado aleatoriamente usando alguma técnica de geração de números aleatórios (GLOROT; BENGIO, 2010) e para realizar o treinamento dos parâmetros é geralmente utilizado o algoritmo *backpropagation* (RUMELHART; HINTON; WILLIAMS, 1986) com variantes de otimizadores como o gradiente descendente estocástico (ROBBINS, 2007) ou Adam (KINGMA; BA, 2017).

O caso da figura 11 é um exemplo específico de MLP. No entanto, é possível ter diferentes quantidades de camadas escondidas e também diferentes tamanhos para as representações, gerando assim diferentes arquiteturas.

4.2 Redes Neurais para Grafos

As redes neurais aritificiais se mostraram poderosas nas mais diversas tarefas de regressão e classificação, popularizadas principalmente por um trabalho de classificação de imagens (KRIZHEVSKY; SUTSKEVER; HINTON, 2012). No entanto, as mais tradicionais como por exemplo MLP ou Convolutional Neural Network (CNN) não estão bem definidas para dados estruturados como grafo. Segundo (HAMILTON, 2020) uma primeira abordagem para classificação de grafos que se poderia adotar para definir uma rede neural para grafos seria simplesmente usar a matriz de adjacências concatenada do grafo como entrada. Neste caso cada grafo $\mathcal{G}$ seria mapeado em um espaço de dimensão $|\mathcal{V}| \times |\mathcal{V}|$ e então passado como entrada para uma MLP, como definido na equação abaixo:

$$\phi(\mathcal{G}) = \mathbf{MLP}(\mathbf{A}[1] \oplus \mathbf{A}[2] \oplus \cdots \oplus \mathbf{A}[|\mathcal{V}|]) \tag{4.6}$$

onde $\mathbf{A}[i] \in \mathbb{R}^{|\mathcal{V}|}$ é a linha i da matriz de adjacências e o simbolo $\oplus$ é a operação de concatenação de vetores.

No entanto, essa abordagem pode não funcionar muito bem pois a matriz de adjacências de um grafo não possui representação única. Como a rotulação dos nós do grafo é completamente arbitrária, para um grafo de tamanho N teríamos portanto $N!$ possibilidades.

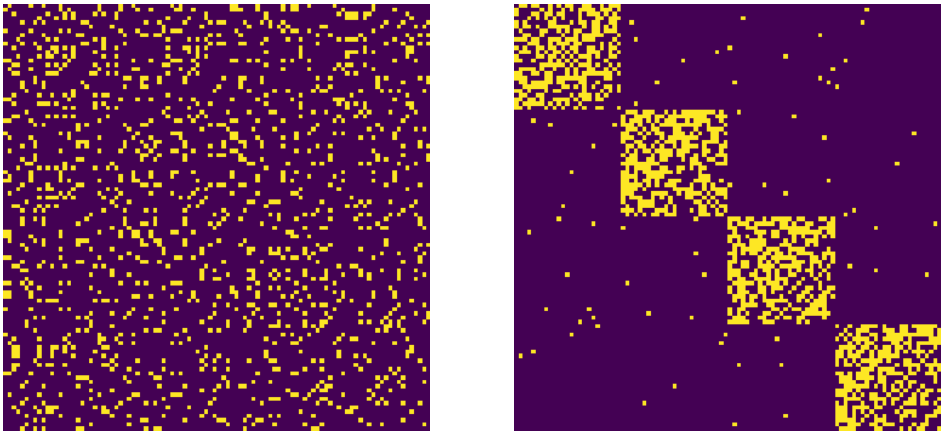

Figura 13 – Exemplo de duas representações para matriz de adjacências para o mesmo grafo. Na imagem da esquerda uma ordem aleatória e na da direita os nós são ordenados de acordo com a comunidade a que pertencem

Fonte: Elaborada pelo autor.

Na Figura 13 a matriz de adjacências a esquerda foi plotada em uma rotulação aleatória para os nós e a da direita os nós foram rotulados de acordo com a comunidade a que pertencem. Como pode-se observar, as duas matriz de adjacências possuem estrutura completamente diferente.

Para definir redes neurais para grafos é, portanto, necessário um novo tipo de arquitetura para aprendizado. Os primeiros trabalhos que usaram o termo *Graph Neural Networks (GNN)* e introduziram o aprendizado via redes neurais em grafos foram os (GORI; MONFARDINI; SCARSELLI, 2005; SCARSELLI *et al.*, 2009). Uma maneira de definir modelos de GNN é usando a noção de *message passing* entre os vértices. Ou seja, a ideia seria que os vértices possuem representações vetoriais e essas representações são usadas para trocar mensagens entre os vértices e suas conexões. Mais precisamente, suponha que queremos um modelo que recebe um grafo $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ e representações vetoriais $\mathbf{h}_u^{(0)}$ iniciais para todo $u \in \mathcal{V}$ e aprende representações vetoriais para os vértices de tal forma a performar alguma específica aplicação como predição de links (ZHANG; CHEN, 2018) ou classificação de nós (KIPF; WELLING, 2017). Durante cada iteração em um modelo de GNN temos uma representação vetorial $\mathbf{h}_u^{(k)}$ para cada vértice e esta representação é atualizada de acordo com alguma regra de agregação sobre os vértices vizinhos. Neste caso o índice k é usado para denotar o número da camada escondida do modelo. De forma geral a regra de atualização das representações pode ser escrita da seguinte

maneira (HAMILTON, 2020):

$$\mathbf{h}_u^{(k+1)} = UPDATE^{(k)}\left(\mathbf{h}_u^{(k)}, \mathbf{m}_{\mathcal{N}(u)}^{(k)}\right) \tag{4.7}$$

Onde $\mathbf{m}_{\mathcal{N}(u)}^{(k)}$ é a mensagem agregada da vizinhança do vértice u e $UPDATE$ é uma função diferenciável arbitrária.

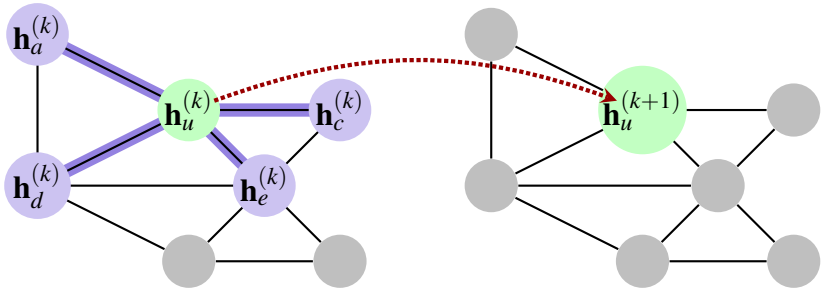

Figura 14 – Representação esquemática da atualização da representação entre camadas (Figura gerada utilizando o pacote Tikz, adaptado da coleção de Petar Veličković https://github.com/PetarV-/TikZ)

Fonte: Elaborada pelo autor.

Na Figura 14 está representado visualmente como funciona o método. O vértice u cuja representação vetorial na camada k é $\mathbf{h}_u^{(k)}$ recebe uma mensagem dos seus vizinhos a, b, c e e e com essas mensagens e a sua própria representação é feita alguma operação para gerar a nova representação vetorial $\mathbf{h}_u^{(k+1)}$.

4.2.1 Graph Neural Network (GNN)

No caso específico do modelo GNN de (SCARSELLI *et al.*, 2009) pode-se definir a operação de *message passing* da seguinte maneira simplificada:

$$\mathbf{h}_u^{(k+1)} = \sigma\left(\mathbf{W}_{self}^{(k+1)}\mathbf{h}_u^{(k)} + \mathbf{W}_{neigh}^{(k+1)}\sum_{v\in\mathcal{N}(u)}\mathbf{h}_v^{(k)} + \mathbf{b}^{(k+1)}\right) \tag{4.8}$$

Em que $\mathbf{W}_{self}^{(k+1)}, \mathbf{W}_{neigh}^{(k+1)} \in \mathbb{R}^{d^{(k+1)}\times d^{(k)}}$ são matrizes de parâmetros treináveis e $\mathbf{b}^{(k+1)} \in \mathbb{R}^{d^{(k+1)}}$, $\sigma(.)$ é uma função de ativação que atua elemento a elemento. Os inteiros $d^{(k)}$ e $d^{(k+1)}$ são usados para denotar a dimensão das representações vetoriais para os vértices das camadas k e $k+1$. Portanto, entre as camadas é possível alterar a dimensão das representações vetoriais para os nós dependendo da escolha específica das dimensões dos parâmetros treináveis.

Note que no modelo de atualização das representações descrito na equação (4.8) a mensagem agregada da vizinhança do vértice u pode ser escrita como:

$$\mathbf{m}_{\mathcal{N}(u)}^{(k)} = \sum_{v \in \mathcal{N}(u)} \mathbf{h}_v^{(k)} \tag{4.9}$$

Portanto, se tivermos um vértice que tem muitos vizinhos a mensagem agregada descrita na equação (4.9) pode resultar em vetores cujos elementos possuem valor alto. Este fato pode resultar em dificuldade no processo de otimização dos parâmetros treináveis e também prejudica a performance de maneira geral.

Uma característica interessante dos modelos de GNN em geral é que os parâmetros treináveis utilizados em cada camada são compartilhados. Ou seja, para atualizar as representação de todos os nós em uma camada específica k teremos os mesmos parâmetros $\mathbf{W}_{self}^{(k)}, \mathbf{W}_{neigh}^{(k)}$ e $\mathbf{b}^{(k)}$, $\forall\, v \in \mathcal{V}$. Sendo assim, é possível escrever a equação (4.8) na forma matricial para a atualização conjunta de todas as representações vetoriais para os vértices, omitindo o termo $\mathbf{b}^{(k+1)}$ por simplicidade:

$$\mathbf{H}^{(k+1)} = \sigma\left(\mathbf{H}^{(k)}\mathbf{W}_{self}^{(k+1)} + \mathbf{A}\mathbf{H}^{(k)}\mathbf{W}_{neigh}^{(k+1)}\right) \tag{4.10}$$

A equação (4.10) realiza a atualização das representações vetoriais para os vértices na camada k para a camada $k+1$ simultaneamente.

4.2.2 Graph Convolutional Networks (GCN)

O modelo de Graph Convolutional Networks (GCN) foi introduzido por (KIPF; WELLING, 2017). Este modelo é similar ao de (SCARSELLI *et al.*, 2009) com a diferença de utilizar uma normalização simétrica da operação de agregação e utilizar apenas uma matriz de parâmetros treináveis em cada camada. Pode-se definir a regra de atualização da seguinte maneira:

$$\mathbf{h}_u^{(k+1)} = \sigma\left(\mathbf{W}^{(k+1)} \sum_{v \in \mathcal{N}(u) \cup \{u\}} \frac{\mathbf{h}_v^{(k)}}{\sqrt{|\mathcal{N}(u)||\mathcal{N}(v)|}} + \mathbf{b}^{(k+1)}\right) \tag{4.11}$$

Em comparação com a equação 4.8, além de introduzir a normalização, a camada GCN possui menos parâmetros treináveis, sendo apenas a matriz $\mathbf{W}^{(k+1)} \in \mathbb{R}^{d^{(k+1)} \times d^{(k)}}$ e $\mathbf{b}^{(k+1)} \in \mathbb{R}^{d^{(k+1)}}$. Isso acontece devido ao fato de que a mensagem agregada dos vizinhos é incorporada com a mensagem do próprio vértice, basta observar que a soma das representações vetoriais acontece para $v \in \mathcal{N}(u) \cup \{u\}$. O termo de normalização simétrica $\frac{1}{\sqrt{|\mathcal{N}(u)||\mathcal{N}(v)|}}$ introduzido ajuda a reduzir instabilidades numéricas ou dificuldades no processo de otimização (HAMILTON, 2020).

A equação (4.11) também pode ser escrita na forma matricial já que os parâmetros da mesma camada são compartilhados, então é possível fazer a atualização das representações para todos os vértices de maneira conjunta:

$$\mathbf{H}^{(k+1)} = \sigma(\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{H}^{(k)}\mathbf{W}^{(k+1)}) \tag{4.12}$$

Em que $\tilde{\mathbf{A}} = \mathbf{A} + \mathbb{I}_N$ é a matriz de adjacências do grafo em questão com auto-loop e $\tilde{\mathbf{D}}$ é a matriz diagonal de graus dos vértices em que os elementos da diagonal são dados por $\hat{D}_{i,i} = \sum_j \tilde{A}_{i,j}$. Na equação (4.12) foi omitido o termo $\mathbf{b}^{(k+1)}$ por simplicidade.

4.2.3 Representações vetoriais por GCN sem treinamento

Para exemplificar o potencial do modelo de GCN considere uma rede do modelo LFR com $N = 1000$, $\tau_1 = 3$, $\tau_2 = 1.5$, $k_{min} = 10$, $k_{max} = 40$, $min_c = 100$, $max_c = 250$ e $\mu = 0.20$. Desta maneira se obtém a rede mostrada na figura abaixo, com um total de 6 comunidades.

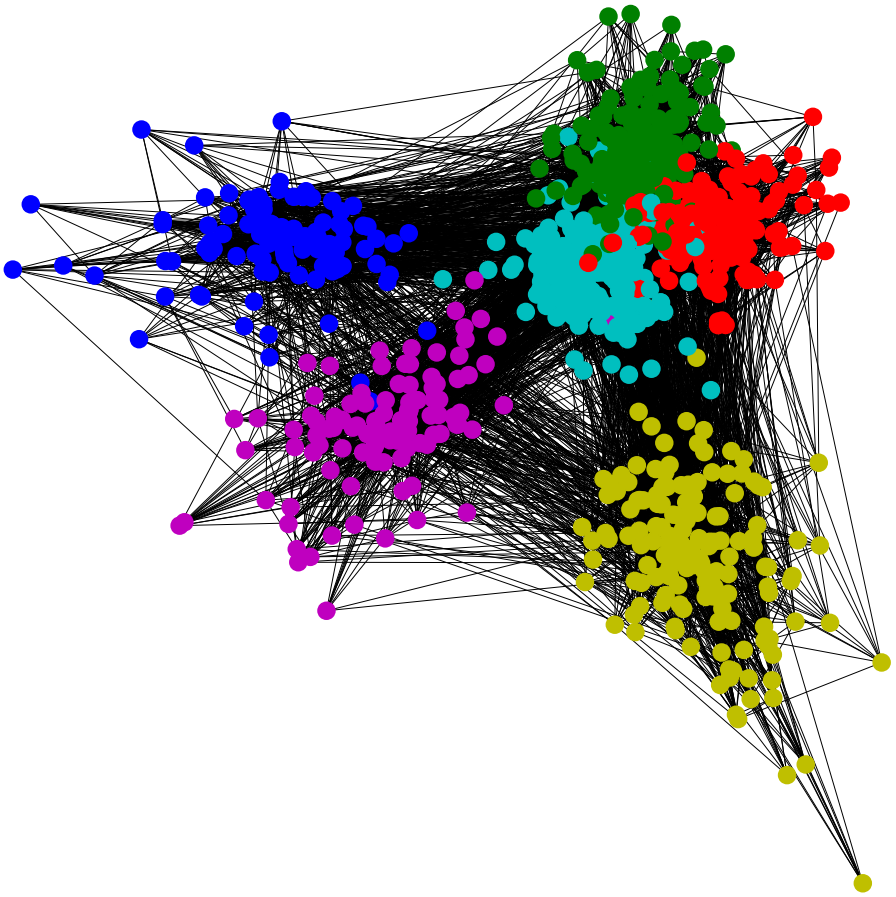

Figura 15 – Rede gerada pelo modelo LFR com $N = 1000$, $\tau_1 = 3$, $\tau_2 = 1.5$, $k_{min} = 10$, $k_{max} = 40$, $min_c = 100$, $max_c = 250$ e $\mu = 0.2$.

Fonte: Elaborada pelo autor.

Cada vértice da rede na figura 15 está colorido de acordo com a comunidade a que pertence. Como neste caso não existe nenhuma informação adicional sobre os vértices dessa rede, consideremos como representação vetorial inicial para os vértices a matriz identidade, ou seja, $\mathbf{H}^{(0)} = \mathbb{I}_N$. Portanto, para todo vértice $u \in \mathcal{V}$ temos $\mathbf{h}_u^{(0)} = \mathbf{x}_u$, sendo $\mathbf{x}_u$ a linha da matriz identidade no índice u. Sendo assim, definimos o seguinte modelo:

$$\mathbf{H}^{(1)} = \sigma\left(\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{H}^{(0)}\mathbf{W}^{(1)}\right) \tag{4.13}$$

$$\mathbf{H}^{(2)} = \sigma\left(\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{H}^{(1)}\mathbf{W}^{(2)}\right) \tag{4.14}$$

$$\mathbf{H}^{(3)} = \sigma\left(\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{H}^{(2)}\mathbf{W}^{(3)}\right) \tag{4.15}$$

$$\mathbf{H}^{(4)} = \sigma\left(\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{H}^{(3)}\mathbf{W}^{(4)}\right) \tag{4.16}$$

$$\mathbf{H}^{(5)} = \sigma\left(\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{H}^{(4)}\mathbf{W}^{(5)}\right) \tag{4.17}$$

Consideramos utilizar representações vetoriais de dimensão 200, ou seja, $\mathbf{H}^{(\ell)} \in \mathbb{R}^{200}$, $\ell \in \{1,2,3,4,5\}$. As matrizes de parâmetros treináveis do modelo são $\mathbf{W}^{(1)}$,$\mathbf{W}^{(2)}$,$\mathbf{W}^{(3)}$,$\mathbf{W}^{(4)}$ e $\mathbf{W}^{(5)}$. Os elementos das matrizes dessas matrizes são inicializados partir da geração de números aleatórios de uma distribuição uniforme, $\mathcal{U}(-\frac{\sqrt{6}}{p}, \frac{\sqrt{6}}{p})$ em que $p = \sqrt{2000}$. A escolha da distribuição tem como base um trabalho sobre o impacto dos valores iniciais para os parâmetros treináveis (GLOROT; BENGIO, 2010). Realizando as operações de *message passing* descritas nas equações (4.13), (4.14), (4.15), (4.16) e (4.17), sem nenhuma regra de treinamento para os parâmetros e então realizando uma redução de dimensionalidade nas representações vetoriais para os vértices em cada camada por t-SNE (MAATEN; HINTON, 2008) se obtém os seguintes gráficos:

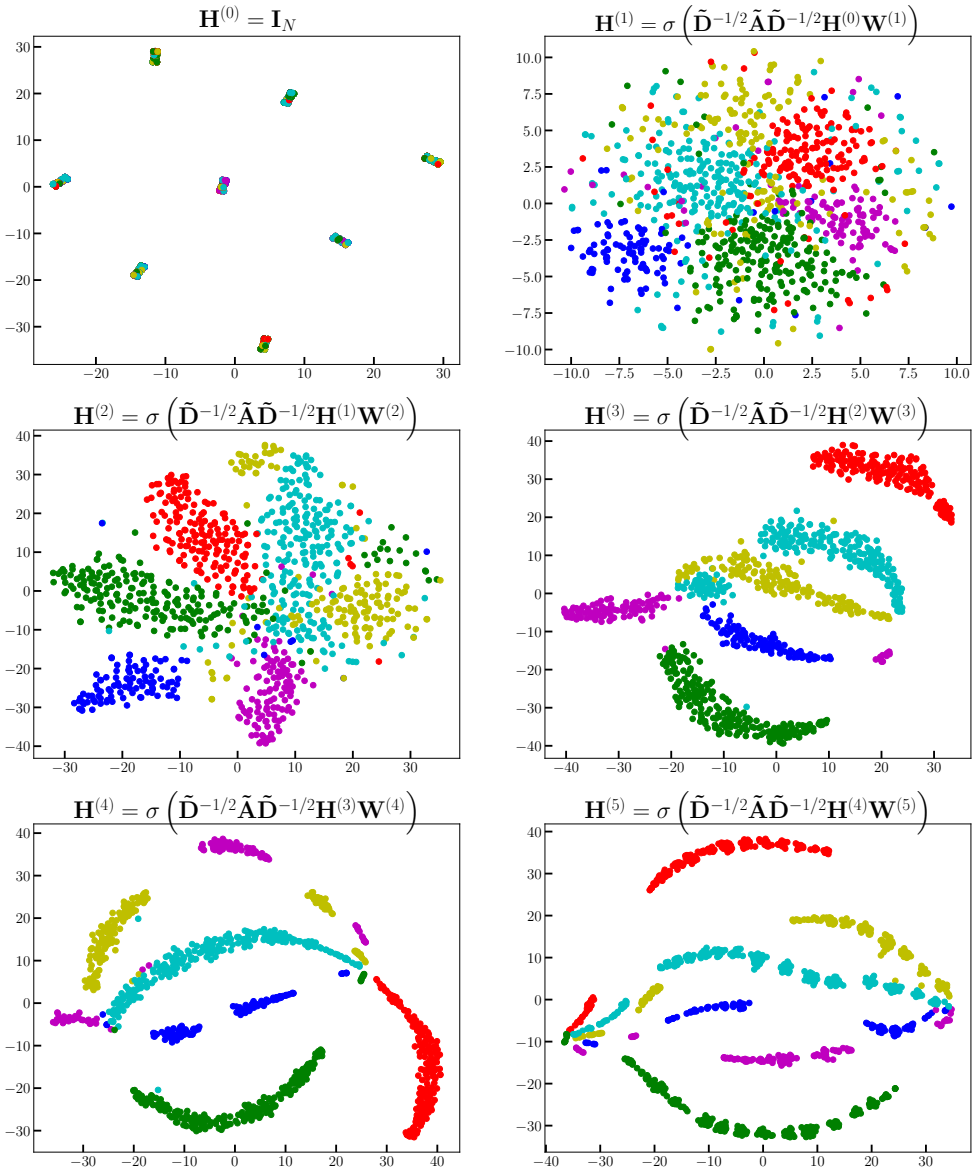

Figura 16 – Representações vetoriais para os vértices em um modelo de rede LFR(N=1000) usando camadas GCNs sem nenhum treinamento nos parâmetros treináveis e com dimensionalidade reduzida por t-SNE.

Fonte: Elaborada pelo autor.

Pode-se observar na representação ilustrada pela figura 16 que os nós que estão na mesma comunidade, representados pelas cores, tem uma representação vetorial similar entre si a partir das camadas GCNs. Este simples exemplo foi para mostrar visualmente o poder de representação usando a abordagem de GCN. Neste caso não foi feito nenhum ajuste nos parâmetros treináveis. No próximo capítulo abordaremos as possíveis tarefas de aprendizado em grafos e também alguns exemplos de funções perda.

4.3 Representações para grafos inteiros

As GNNs podem ser usadas também para tarefas a nível do grafo inteiro. Mais especificamente, temos um conjunto $\mathcal{D} = \{(\mathcal{G}_1, y_1), (\mathcal{G}_n, y_2), \cdots, (\mathcal{G}_n, y_n)\}$ de grafos e rótulos e o objetivo é usar os modelos GNN para classificação. Uma abordagem seria simplesmente usar como representação vetorial para o grafo inteiro a média de todas as representações dos vértices:

$$\mathbf{z}_{\mathcal{G}} = \frac{\sum_{v \in \mathcal{V}} \mathbf{h}_v}{|\mathcal{V}|} \tag{4.18}$$

Também é possível usar a soma ou fazer alguma outra abordagem mais elaborada (XINYI; CHEN, 2019) para agregar todas as representações vetoriais dos vértices de tal forma a obter a representação para o grafo como um todo.

4.4 Funções Perda

4.4.1 Aprendizado supervisionado ou semi-supervisionado

Existem diversas aplicações de modelos GNN em grafos, como citado anteriormente. As principais são classificação de nós (KIPF; WELLING, 2017; WEBER *et al.*, 2019), predição de links (ZHANG; CHEN, 2018), classificação de grafos (ZHANG *et al.*, 2018).

Por exemplo, se a tarefa específica for classificação binária de nós, pode-se usar como função perda a entropia cruzada:

$$\mathcal{L} = \sum_{v \in \mathcal{V}} y_v log(\sigma(\mathbf{z}_v^T \theta)) + (1 - y_v) log(1 - \sigma(\mathbf{z}_v^T \theta)) \tag{4.19}$$

Na equação (4.19) $\mathbf{z}_v$ é a representação vetorial da última camada em um modelo de GNN para o vértice v, θ é um vetor de parâmetros treináveis, σ pode ser a função sigmoid, $y_v \in \{0, 1\}$ representa a classe verdadeira do nó v. Além disso, $\sigma(\mathbf{z}_v^T \theta)$ vai ser um valor no intervalo $[0, 1]$ que representa a probabilidade predita para o nó v ser da classe 1 por exemplo.

Se o cenário for o de aprendizado semi-supervisionado para classificação de nós, teríamos apenas os rótulos de alguns nós e portanto a tarefa de aprendizado teria que levar em conta apenas

os nós que possuem rótulo. Sendo assim, alterando a equação (4.19) para um aprendizado semi-supervisionado implica que ao invés de considerar a função perda para $\forall\ v \in \mathcal{V}$, é considerado apenas nos nós que estão disponíveis os rótulos, ou seja, $\forall\ v \in \mathcal{V}_{train}$, em que $\mathcal{V}_{train} \subset \mathcal{V}$.

$$\mathcal{L} = \sum_{v \in \mathcal{V}_{train}} y_v log(\sigma(\mathbf{z}_v^T \theta)) + (1 - y_v) log(1 - \sigma(\mathbf{z}_v^T \theta)) \tag{4.20}$$

Note que mesmo a função perda da equação (4.20) sendo calculada apenas em um subconjunto do total de vértices, é possível ter as predições de nós que não foram usados para treinar os parâmetros do modelo. Na prática, com os modelos de GNN obteremos representações vetoriais para todos os nós, $\mathbf{z}_v$, $\forall\ v \in \mathcal{V}$, sendo assim, é possível calcular para todos qual a probabilidade de serem da classe 1 por exemplo.

Para tarefas de regressão em grafos inteiros, como por exemplo prever propriedades de moléculas, pode-se usar o erro quadrático da seguinte maneira (HAMILTON, 2020):

$$\mathcal{L} = \sum_{\mathcal{G}_i \in \mathcal{T}} ||MLP(\mathbf{z}_{\mathcal{G}_i}) - y_{\mathcal{G}_i}||_2^2 \tag{4.21}$$

Em que $\mathcal{T} = \{\mathcal{G}_1, \cdots, \mathcal{G}_n\}$ é o conjunto de grafos usados para o treinamento do modelo e $\mathcal{Y} = \{y_{\mathcal{G}_1}, \cdots, y_{\mathcal{G}_n}\}$ é o conjunto de valores reais associados a cada grafo. Além disso, $\mathbf{z}_{\mathcal{G}_i}$ é a representação vetorial dada por um modelo genérico de GNN para o nó i.

Já em relação a classificação de grafos é possível utilizar a entropia cruzada de forma similar a definida no caso de classificação de nós (4.19) da seguinte maneira:

$$\mathcal{L} = \sum_{\mathcal{G}_i \in \mathcal{T}} -log(softmax(\mathbf{z}_{\mathcal{G}_i}, \mathbf{y_i})) \tag{4.22}$$

4.4.2 Aprendizado não-supervisionado

Em aprendizado não supervisionado no caso de grafos seria quando não se tem nenhum rótulo dos vértices em caso de classificação ou regressão. Uma maneira para treinar os parâmetros dos modelos de GNN é através de *Auto-Encoders*, que nada mais é, neste caso, de uma tentativa de reconstrução do grafo original a partir das representações obtidas na saída de um modelo de GNN.

(KIPF; WELLING, 2016) introduziu o modelo de Graph Auto-Encoders (GAE) que consiste em treinar um modelo de GNN para predição de arestas. Mais formalmente, seja $\mathbf{H}^{(L)}$ a matriz de representações para todos os vértices na saída de um modelo de GNN com L camadas. Uma maneira de reconstruir a matriz de adjacências a partir das representações $\mathbf{H}^{(L)}$ é tomando o produto interno entre todos os pares de vértices e passando para uma sigmoid que vai mapear o

resultado do produto interno em um valor entre 0 e 1, que neste caso representa a probabilidade de conexão entre um par de vértices. Ou seja:

$$\tilde{\mathbf{A}} = \sigma(\mathbf{H}^{(L)}(\mathbf{H}^{(L)})^T) \tag{4.23}$$

Em que $\sigma(.)$ é a função sigmoid que atua elemento a elemento e $\tilde{\mathbf{A}}$ é a matriz de adjacências reconstruída a partir das representações vetoriais. Neste caso $\tilde{A}_{ij}$ representa a probabilidade predita de existência de conexão entre os vértices i e j. A intuição por trás deste método é que o produto interno entre vetores está relacionado com a medida de similaridade de cosseno, ou seja, se tivermos um par vértices conectados, a similaridade das representações vetoriais desses vértices deverá ser alta para que seja predito com êxito a existência de aresta. Como função perda neste caso é possível utilizar a entropia cruzada com pesos nas classes.

$$\mathcal{L} = -\sum_i \sum_j W_1(A_{ij} log(\tilde{A}_{ij})) + W_2(1 - A_{ij}) log(1 - \tilde{A}_{ij}) \tag{4.24}$$

Em que W_1 e W_2 são a proporção de arestas e não-arestas respectivamente no grafo original. É importante utilizar esses pesos na função perda porque em geral a matriz de adjacências de um grafo é muito esparsa em casos de redes reais, sendo assim, uma forma de pensar nesse problema é que existe um desbalanceamento grande entre a existência e a não-existência de arestas no grafo que se está tentando fazer a predição.

CAPÍTULO

5

RESULTADOS E DISCUSSÃO

5.1 Detecção da fonte de rumor em redes com estrutura de comunidades

Foi considerado o problema de detectar qual a fonte da propagação de rumor em uma rede com estrutura de comunidades cuja dinâmica começou em uma das comunidades com múltiplos *propagadores*. Supondo uma dinâmica de Maki-Thompson em que todos os propagadores iniciais pertencem a mesma comunidade de um grafo, o problema consiste então em conseguir identificar de qual comunidade a propagação foi iniciada dado que se passou um intervalo de tempo desde o inicio da propagação.

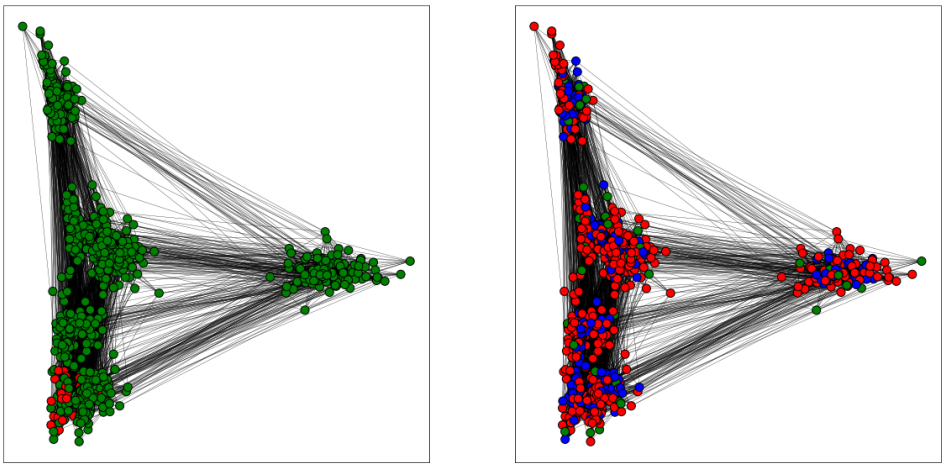

Figura 17 – Evolução da dinâmica em uma rede com estrutura de comunidades

Fonte: Elaborada pelo autor.

Na Figura 17 na figura da esquerda está representado um grafo do modelo LFR que possui 7 comunidades. Uma quantidade de vértices em vermelho de uma das 7 comunidades estão no estado de *propagadores* e começam a propagação do rumor no grafo. Ainda na figura 17 da direita, depois de uma quantidade de passos da dinâmica o rumor já se propagou para todos as comunidades do grafo e se observa que existem vértices vermelhos, do estado de *propagadores* em todas as comunidades do grafo e que também existem diversos vértices azuis que são os indivíduos no estado de *calados*. Seria interessante ter um modelo que para dado um momento no tempo em que um rumor já está totalmente propagado em um grafo, o modelo retorna a comunidade mais provável de ter iniciado a propagação. Para efeitos práticos isto poderia ser útil em redes sociais em que diversos rumores são propagados, muitas vezes de maneira coordenada dentro de um grupo específico de indivíduos. Retornar de forma imediata o grupo mais provável de ter começado a propagação pode ajudar a identificar os principais usuários do grupo que disseminam rumores e com essa identificação tornar-se possível a realização de alguma intervenção nesses usuários.

Existem trabalhos que consideram recuperar o primeiro propagador em simulações de Suscetível-Infectado-Recuperado (SIR) (SHAH *et al.*, 2020) fazendo uso de GNNs. Fundamentalmente a ideia do problema é similar a que será tratada aqui porém difere no sentido de que a dinâmica de interesse nesse trabalho é a dinâmica de propagação de rumor de Maki-Thompson e além disso o interesse aqui não é recuperar o primeiro propagador e sim recuperar o agrupamento ou comunidade em que diversos propagadores ao mesmo tempo iniciaram o processo de disseminação de um rumor/informação.

Para construção de um modelo baseado em GNN para este problema é preciso definir um conjunto de dados e um problema de classificação a partir desse conjunto. O conjunto de dados para um grafo em particular consiste no estado dos vértices em algum momento escolhido da dinâmica Maki-Thompson e o rótulo da comunidade que começou a propagação, ou seja, para um grafo específico $\mathcal{G}_j$ temos:

$$\mathcal{T}_{\mathcal{G}_j} = \{(\mathcal{G}_j, \mathbf{X}_1, y_1), \cdots, (\mathcal{G}_j, \mathbf{X}_n, y_n)\} \tag{5.1}$$

No conjunto (5.1) é definido por triplas em que $\mathcal{G}_j$ é o grafo em questão, $\mathbf{X}_i$ para $i \in \{1, \cdots, n\}$ são matrizes que contém a informação do estado de todos os vértices e y_i para $i \in \{1, \cdots, n\}$ são os rótulos das comunidades em que diversos propagadores começaram o rumor e por fim n é a quantidade de observações do conjunto. Nas matrizes $\mathbf{X}_i$, cada linha $x_{i,v}$ corresponde a um vetor que indica em qual estado o vértice v está, para $v \in \{1, \cdots, N\}$ em que N é a quantidade de vértices do grafo. Por exemplo, se o vértice v está no estado *ignorante* o vetor será $\mathbf{x}_{v,i} = [1,0,0]$, já se o vértice estiver no estado *propagador* então $\mathbf{x}_{v,i} = [0,1,0]$ e por fim se estiver no estado *calado* o vetor será $\mathbf{x}_{v,i} = [0,0,1]$.

O modelo baseado em GCN será definido de maneira simples. Na camada final é necessário que se retorne um vetor com a dimensão sendo igual ao número de comunidades que o grafo possui. Portanto, é necessário realizar algumas transformações após obter as representações vetoriais para os vértices do grafo. A arquitetura em que foram realizados os experimentos é definida da seguinte maneira:

$$\mathbf{h}_u^{(1)} = \mathbf{A}\mathbf{h}_u^{(0)} + \mathbf{b}^{(1)} \tag{5.2}$$

$$\mathbf{h}_u^{(2)} = \mathbf{h}_u^{(1)} + \sigma\left(\mathbf{W}^{(2)} \sum_{v \in \mathcal{N}(u) \cup \{u\}} \frac{\mathbf{h}_v^{(1)}}{\sqrt{|\mathcal{N}(u)||\mathcal{N}(v)|}} + \mathbf{b}^{(2)}\right) \tag{5.3}$$

$$\mathbf{h}_u^{(3)} = \mathbf{h}_u^{(2)} + \sigma\left(\mathbf{W}^{(3)} \sum_{v \in \mathcal{N}(u) \cup \{u\}} \frac{\mathbf{h}_v^{(2)}}{\sqrt{|\mathcal{N}(u)||\mathcal{N}(v)|}} + \mathbf{b}^{(3)}\right) \tag{5.4}$$

$$\mathbf{y}_u = \mathbf{R}\sigma(\mathbf{T}\mathbf{h}_u^{(3)} + \mathbf{s}) \tag{5.5}$$

$$\mathbf{z}_{\mathcal{G}} = \mathbf{y}_1 \oplus \mathbf{y}_2 \oplus \cdots \oplus \mathbf{y}_N \tag{5.6}$$

$$\hat{\mathbf{p}} = softmax(\mathbf{P}\mathbf{z}_{\mathcal{G}} + \mathbf{q}) \tag{5.7}$$

As camadas (5.2), (5.3), (5.4) e (5.5) estão definidas na visão de representações para os vértices do grafo, em que $u \in \mathcal{V}$ em que $\mathcal{V}$ é o conjunto de vértices do grafo. Já a camada (5.6) se trata de uma agregação das informações, para ter uma representação para o grafo todo e por fim a camada (5.7) é a transformação final para obter um vetor cuja dimensão é exatamente o número de comunidades que o grafo possui. Sendo assim, cada posição i deste vetor pode ser interpretada como a probabilidade do rumor ter começado na comunidade i. Uma observação relevante é que a camada (5.6) possui uma operação de concatenação que não é invariante a ordem. No entanto, como será um modelo definido para cada grafo, não existe nenhum problema em ter uma operação invariante. A única limitação que esta operação induz é a de que um modelo que foi feito para um grafo poderá não performar bem em um outro grafo diferente do qual o modelo foi treinado.

Na equação (5.2) a representação inicial para o vértice u é definida como sendo o vetor que indica o estado que o vértice se encontra na dinâmica, ou seja, para a observação i temos $\mathbf{h}_u^{(0)} = \mathbf{x}_{u,i}^T$ para todos os vértices u do grafo.

Como função perda para quantificar a qualidade das predições e definir o objetivo de ser otimizado foi utilizada a entropia cruzada:

$$\mathcal{L} = -\frac{1}{N_\tau} \sum_{k=1}^{N_\tau} \sum_{c \in \mathcal{C}} \mathbb{1}(y_k = c) log(\hat{p}_{c,k}) \tag{5.8}$$

N_τ é a quantidade de dados do conjunto de treinamento em que foi calculado a função perda. O valor $\hat{p}_{c,k}$ é a probabilidade da propagação ter começado na comunidade c para a observação k.

Para avaliar o modelo foi considerado dois cenários de experimentos. O primeiro consiste em redes LFR com quatro comunidades de tamanhos iguais. E o segundo cenário com redes LFR sem essa limitação de quantidade de comunidades e nem para os tamanhos serem iguais. A definição destes dois tipos de experimentos vem do fato de que uma heterogeneidade muito grande no grafo pode complicar a interpretação dos resultados, portanto com o primeiro cenário pode-se visualizar melhor o potencial do modelo.

5.1.1 Redes LFR com comunidades de tamanhos iguais

Para analisar o potencial do modelo baseado GCN em grafos com comunidades de quantidades de vértices iguais foi considerado utilizar o modelo LFR com os parâmetros:

- $N = 1000$, quantidade de vértices
- $\tau_1 = 3$, expoente da distribuição lei de potência para os graus.
- $k_{min} = 10$, quantidade mínima do grau de um vértice.
- $k_{max} = 20$, quantidade máxima do grau de um vértice.
- $min_c = 250$, quantidade de vértices miníma em uma comunidade.
- $max_c = 250$, quantidade de vértices máxima em uma comunidade.

Além dos parâmetros citados acima, o parâmetro de mistura μ foi variado no intervalo $[0.04, 0.6]$ de tal forma a criar 20 redes com diferentes intensidades de característica de agrupamento em comunidades. Desta maneira são criadas redes LFR similares a da figura abaixo:

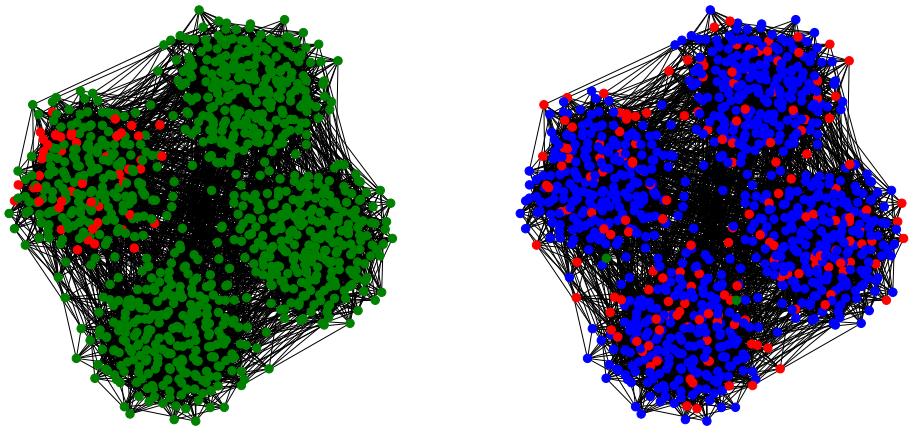

Figura 18 – Evolução da dinâmica em uma rede LFR com comunidades de mesmo tamanho.

Fonte: Elaborada pelo autor.

Na figura 18 os grafos representados são um grafo do modelo LFR com os parâmetros $N = 1000$, $\tau_1 = 3$, $k_{min} = 10$, $k_{max} = 20$, $min_c = 250$ e $\mu = 0.1$ em diferentes momentos de uma dinâmica Maki-Thompson. Como citado anteriormente, foram criados 20 grafos LFR variando o parâmetro de mistura. Para cada um dos grafos foi gerado um conjunto de dados com 2500 observações da maneira descrita pelo conjunto (5.1). Para gerar este conjunto foram simuladas dinâmicas Maki-Thompson com parâmetros nos seguintes intervalos:

- $\lambda \in [0.1, 0.9]$, (Taxa de propagação)
- $\alpha \in [0.1, 0.9]$, (Taxa de recuperação)
- $steps \in [1, 2500]$, (Quantidade de passos da dinâmica)
- $frac \in [0.1, 1]$, (Fração inicial de propagadores dentro de uma comunidade)

Os parâmetros da dinâmica definidos acima foram escolhidos aleatoriamente dentro do intervalo possível para então realizar a simulação Maki-Thompson.

Para o modelo baseado em GCN representado no conjunto de equações (5.2), (5.3), (5.4),(5.5), (5.6), (5.7) temos que a dimensão da representação inicial é 3, já que é o vetor que representa o estado em que o vértice u se encontra se encontra, ou seja, $\mathbf{h}_u^{(0)} \in \mathbb{R}^3$, já

a dimensão das representações das camadas GCN são 32, ou seja, $\mathbf{h}_u^{(1)}$ e $\mathbf{h}_u^{(2)} \in \mathbb{R}^{32}$. Na camada de saída, neste caso, a dimensão será 4, pois os grafos LFR gerados com os parâmetros definidos fornecem grafos com essa quantidade de comunidades. A função de ativação σ utilizada *ReLU*. O espaço de matrizes e vetores treináveis das camadas do modelo é dado por: $\theta = \{\mathbf{A}, \mathbf{b}^{(1)}, \mathbf{W}^{(2)}, \mathbf{b}^{(2)}, \mathbf{W}^{(3)}, \mathbf{b}^{(3)}, \mathbf{R}, \mathbf{T}, \mathbf{s}, \mathbf{P}, \mathbf{q}\}$. Para o treinamento dos parâmetros treináveis foi considerado otimizador Adam (KINGMA; BA, 2017) com taxa de aprendizado de 0.001 por 10 épocas.

Para entender a performance do modelo foram simuladas outras 600 observações independentes das usadas do treinamento do modelo utilizando alguns cenários possíveis para os parâmetros de simulação. Nas próximas sessões serão mostrados os resultados dos modelos treinados para cada grafo nas novas observações.

5.1.1.1 *Impacto da quantidade de iterações da dinâmica*

Para compreender o impacto do tempo desde o começo da propagação na predição do modelo, foram simuladas novas observações com os parâmetros fixos $frac = 0.2$ (fração inicial de propagadores que iniciou dentro da comunidade), $\alpha = 0.4$ (taxa de recuperação), $\lambda = 0.6$ (taxa de propagação) e além dos parâmetros fixos a quantidade de iterações da dinâmica foi variada sequencialmente no intervalo $[1, 2500]$.

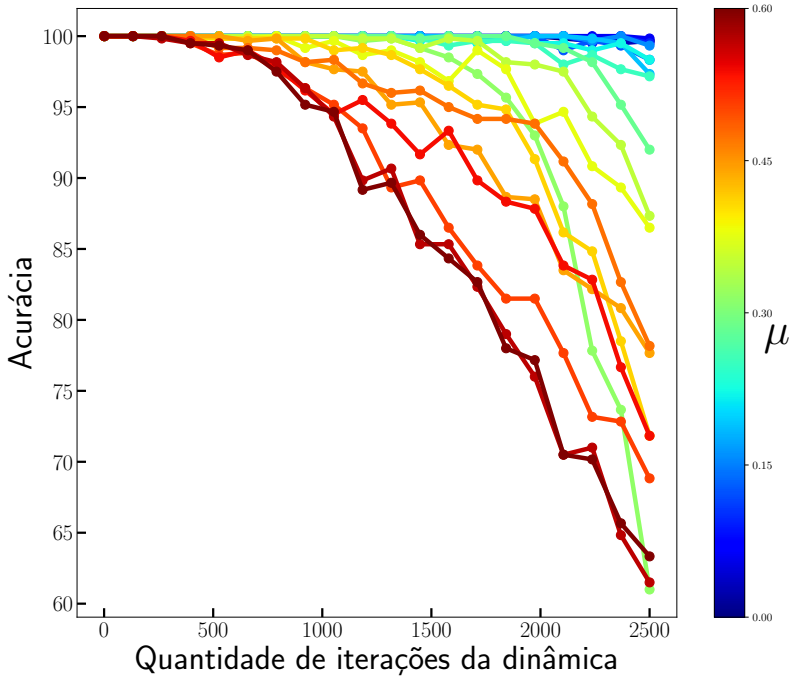

Figura 19 – Acurácia dos modelos em função dos passos de tempo para diferentes valores de μ

Fonte: Elaborada pelo autor.

Na figura 19 cada ponto é a média da acurácia do modelo para 600 observações simuladas nas mesmas condições. Cada uma das curvas corresponde a um modelo baseado em GCN e para as intensidades de cores variam de acordo com o parâmetro de mistura μ que representa o quão modular o grafo em que a dinâmica foi simulada, lembrando que maiores valores de μ (parâmetro de mistura) estão associados a um grafo com estrutura de comunidade menos bem definida. Pode-se observar ainda na figura 19 que caso o grafo tenha μ maior a performance dos modelos é degradada com maior intensidade para uma quantidade menor de iterações da dinâmica. Ou seja, caso o grafo possua uma menor definição clara de estrutura de comunidades a performance do modelo é prejudicada. Em aspectos práticos da dinâmica isso significa que caso alguns propagadores de uma comunidade comecem a propagar dentro de uma comunidade e os vértices dessa comunidade na verdade tem muitas conexões com vértices de outras comunidades a chance do rumor/informação pular para outra comunidade é maior, portanto neste caso o modelo baseado em GCN acaba errando um pouco mais.

5.1.1.2 Impacto da fração inicial de propagadores

Para compreender o impacto da quantidade inicial de propagadores dentro da mesma comunidade na performance do modelo, foram simuladas novas observações com os parâmetros fixos $\alpha = 0.4$ (taxa de recuperação), $\lambda = 0.6$ (taxa de propagação), $step = 1000$ (quantidade de iterações da dinâmica) e foi variado sequencialmente a fração inicial de propagadores no intervalo $[0.1, 0.9]$

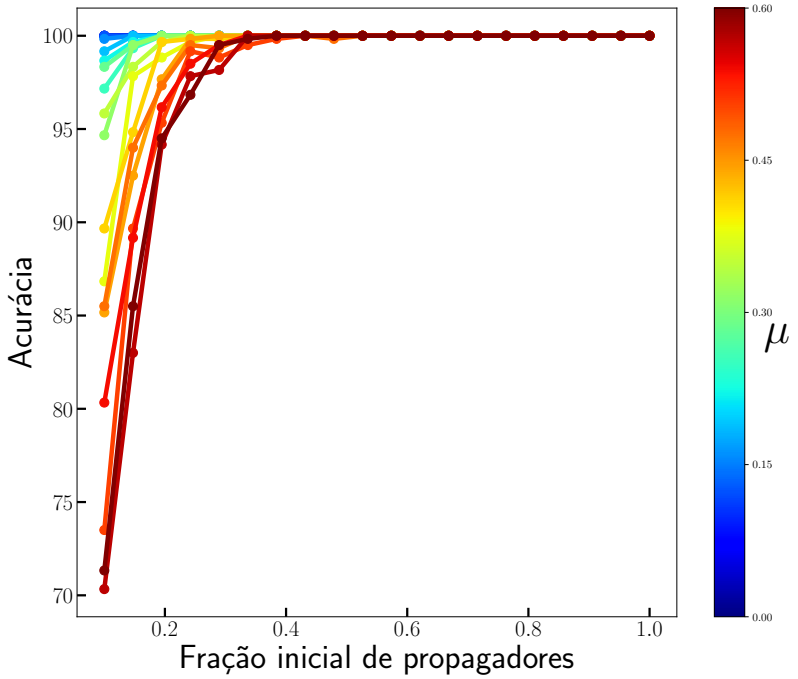

Figura 20 – Acurácia dos modelos em função da fração inicial de propagadores para diferentes valores de μ

Fonte: Elaborada pelo autor.

Na figura 20 cada ponto é a média da acurácia do modelo para 600 observações simuladas nas mesmas condições. Como pode-se observar para os casos de grafos mais próximos a cor vermelha a performance é prejudicada quando a fração inicial de propagadores é pequena. Ou seja, em termos práticos da dinâmica isso significa que com poucos vértices dentro de uma comunidade começando a propagar o rumor/informação em um grafo em que os vértices possuem muitas conexões fora da própria comunidade, a propagação tende a sair da própria comunidade em menos passos de tempo, resultando assim na performance prejudicada do modelo baseado em GCN. Além disso vale destacar que para o caso em que a fração inicial de propagadores se aproxima de 1, ou seja, todos os vértices da comunidade começam a propagar o rumor/informação, temos uma performance do modelo muito assertiva nesse cenário.

5.1.1.3 Impacto da taxa de propagação

Para compreender o impacto da taxa de propagação na performance do modelo, foram simuladas novas observações com os parâmetros fixos $\alpha = 0.4$ (taxa de recuperação), $frac = 0.1$ (fração inicial de propagadores), $step = 1000$ (quantidade de iterações da dinâmica) e foi variado sequencialmente a taxa de propagação λ intervalo $[0.1, 0.9]$.

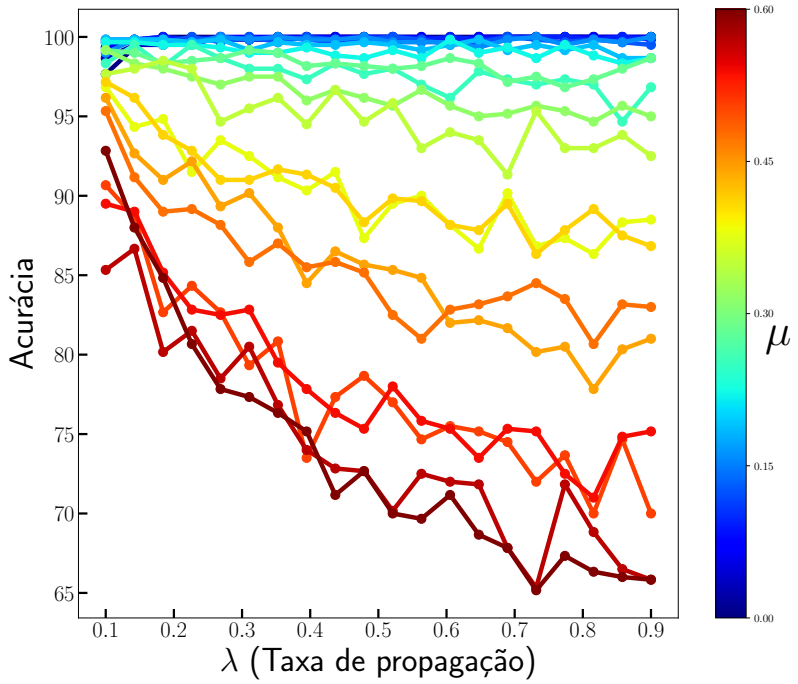

Figura 21 – Acurácia dos modelos em função da taxa de propagação λ para diferentes valores de μ

Fonte: Elaborada pelo autor.

Na figura 20 cada ponto é a média da acurácia do modelo para 600 observações simuladas nas mesmas condições. Nota-se que a performance do modelo é prejudicada também para valores altos do parâmetro de mistura μ que são as cores mais próximas ao vermelho. Além disso, nota-se que isso acontece para valores maiores da taxa de propagação λ. Em outras palavras, caso o grafo possua uma estrutura modular baixa e o rumor/informação se propaga com uma taxa elevada, então o modelo baseado em GCN tende a ter uma performance pior. Por outro lado, para redes cuja modularidade é alta, ou seja, nos casos de μ baixo, a taxa de propagação não mostra tanto impacto.

5.1.1.4 Impacto da taxa de recuperação

Para compreender o impacto da taxa de propagação na performance do modelo, foram simuladas novas observações com os parâmetros fixos $\lambda = 0.8$ (taxa de recuperação), $frac = 0.2$ (fração inicial de propagadores), $step = 1000$ (quantidade de iterações da dinâmica) e foi variado sequencialmente a taxa de recuperação α no intervalo $[0.1, 0.9]$.

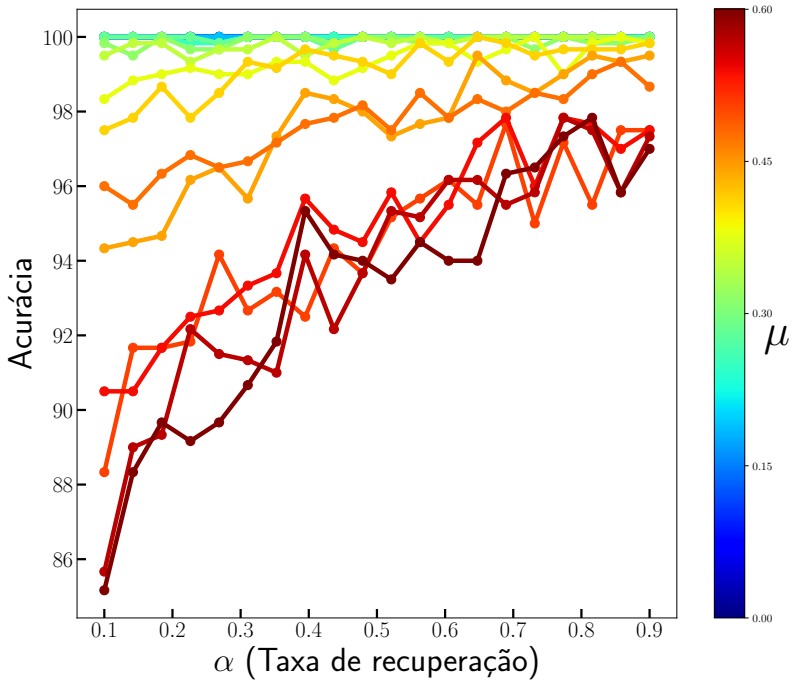

Figura 22 – Acurácia dos modelos em função da taxa de recuperação α para diferentes valores de μ

Fonte: Elaborada pelo autor.

Na figura 20 cada ponto é a média da acurácia do modelo para 600 observações simuladas nas mesmas condições. Nota-se que a performance do modelo é prejudicada também para valores altos do parâmetro de mistura μ que são as cores mais próximas ao vermelho. Além disso, nota-se que isso acontece para valores menores da taxa de recuperação α, lembrando que quanto maior essa taxa maior a chance de indivíduos que estão no estado de propagadores virarem calados. Em outras palavras, em grafos com pouca estrutura modular e a dinâmica possuindo uma taxa de recuperação baixa a performance do modelo baseado em GCN se degrada. Por outro lado, nos casos em que o parâmetro de mistura μ é baixo, não se observa esse impacto.

5.1.2 Redes LFR com comunidades de tamanhos heterogêneos

Para analisar o potencial do modelo baseado GCN em grafos com comunidades de quantidades de vértices heterogêneas foi considerado utilizar o modelo LFR com os parâmetros:

- $N = 1000$, quantidade de vértices
- $\tau_1 = 3$, expoente da distribuição lei de potência para os graus.
- $k_{min} = 10$, quantidade mínima do grau de um vértice.
- $k_{max} = 20$, quantidade máxima do grau de um vértice.
- $min_c = 50$, quantidade de vértices miníma em uma comunidade.
- $max_c = 250$, quantidade de vértices máxima em uma comunidade.

Além dos parâmetros citados acima, o parâmetro de mistura μ foi variado no intervalo $[0.04, 0.6]$ de tal forma a criar 20 redes com diferentes intensidades de característica de agrupamento em comunidades. Abaixo um exemplo da evolução da dinâmica em uma dos grafos gerados com essa configuração de parâmetros.

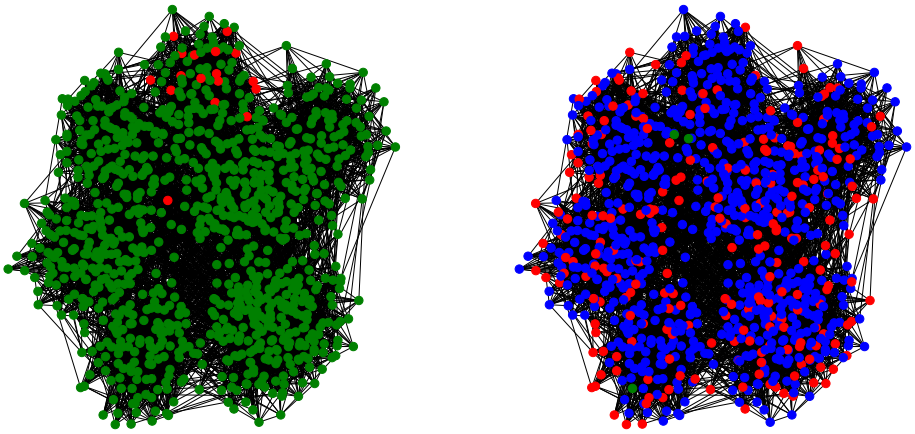

Figura 23 – Evolução da dinâmica em uma rede LFR com comunidades de diversos tamanhos.

Fonte: Elaborada pelo autor.

De maneira similar ao caso de grafos LFR com comunidades de tamanhos iguais, foram geradas observações para o treinamento do modelo baseado em GCN e serão avaliados os impactos dos diferentes parâmetros de simulação Maki-Thompson e de características estruturais do grafo LFR.

5.1.2.1 Impacto da quantidade de iterações da dinâmica

Para avaliação do modelo baseado em GCN em grafos LFR com comunidades de diversos tamanhos em termos de quantidade de iterações da dinâmica foram simuladas novas observações com parâmetros da dinâmica Maki-Thompson iguais aos citados na seção 5.1.1.1.

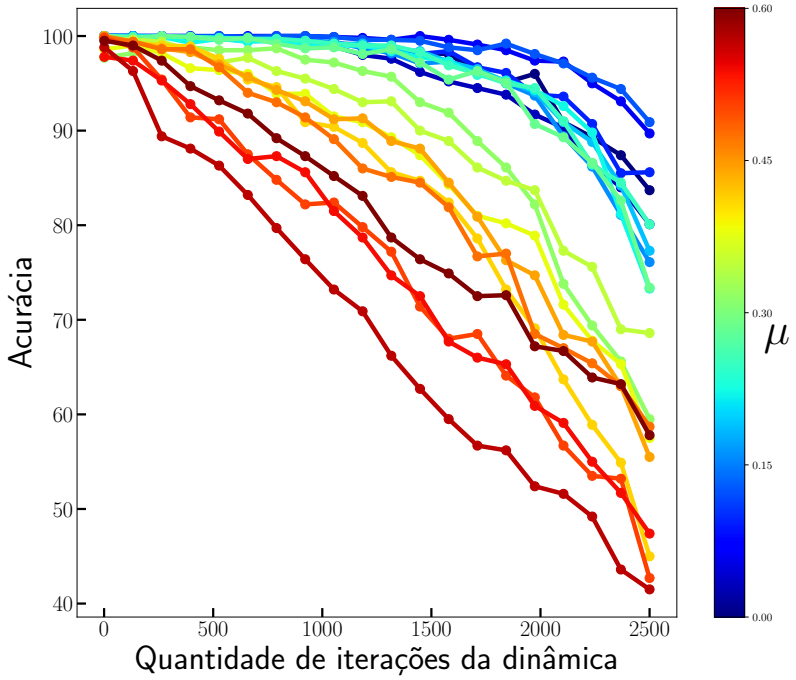

Figura 24 – Acurácia dos modelos em função dos passos de tempo para diferentes valores de μ

Fonte: Elaborada pelo autor.

Em comparação com o caso de grafos LFR com comunidades de tamanhos iguais, no caso da figura 24 em grafos mais heterogêneos observa-se que a performance do modelo baseado em GCN cai mais rapidamente para todos os casos do parâmetro de mistura μ.

5.1.2.2 Impacto da fração inicial de propagadores

Para avaliação do modelo baseado em GCN em grafos LFR com comunidades de diversos tamanhos com relação a fração inicial de propagadores foram simuladas novas observações com parâmetros da dinâmica Maki-Thompson iguais aos citados na seção 5.1.1.2.

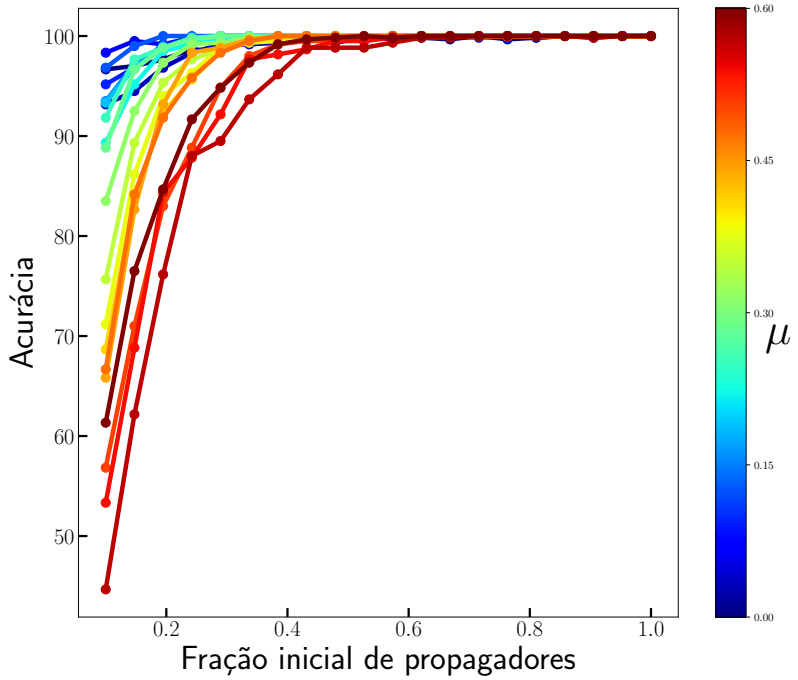

Figura 25 – Acurácia dos modelos em função da fração inicial de propagadores para diferentes valores de μ

Fonte: Elaborada pelo autor.

Na figura 25 observa-se um comportamento muito similar ao da figura 20, com a diferença que para o caso de grafos LFR com comunidades de diversos tamanhos a acurácia observada é menor nos piores cenários em que μ é alto e a fração inicial de propagadores é pequena. No caso aqui observado pode-se interpretar o comportamento da mesma maneira citada na seção 5.1.1.2.

5.1.2.3 Impacto da taxa de propagação

Para avaliação do modelo baseado em GCN em grafos LFR com comunidades de diversos tamanhos com relação a taxa de propagação foram simuladas novas observações com parâmetros da dinâmica Maki-Thompson iguais aos citados na seção 5.1.1.3.

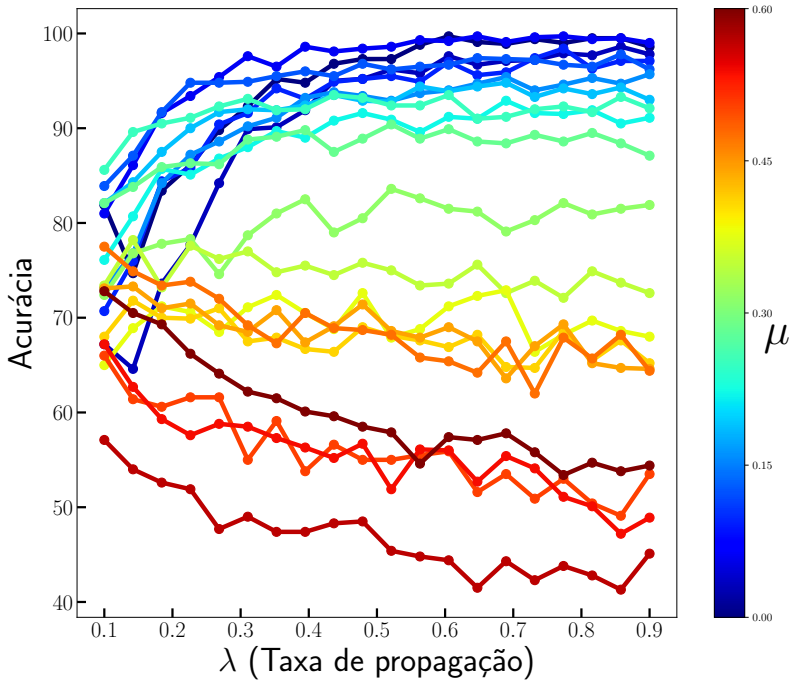

Figura 26 – Acurácia dos modelos em função da taxa de propagação λ para diferentes valores de μ

Fonte: Elaborada pelo autor.

Na figura 26 observa-se um comportamento diferente do observado na figura 21. Nos casos em que o parâmetro de mistura μ é pequeno e a taxa de propagação λ é baixa nota-se uma degradação da performance do modelo baseado em GCN. Neste caso, o padrão é um pouco diferente do observado para o caso homogêneo, não sendo trivial de interpretar o porque da degradação da performance para os casos em que μ é baixo.

5.1.2.4 Impacto da taxa de recuperação

Para avaliação do modelo baseado em GCN em grafos LFR com comunidades de diversos tamanhos com relação a taxa de recuperação foram simuladas novas observações com parâmetros da dinâmica Maki-Thompson iguais aos citados na seção 5.1.1.4.

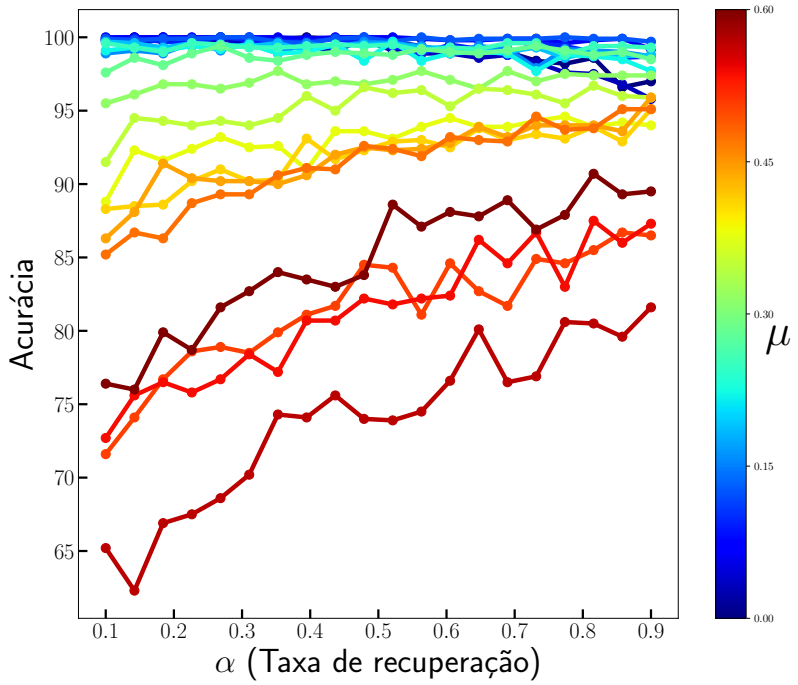

Figura 27 – Acurácia dos modelos em função da taxa de recuperação α para diferentes valores de μ

Fonte: Elaborada pelo autor.

Para a taxa de recuperação α é observado um padrão similar ao caso de redes LFR com comunidades de tamanhos homogêneos, com a diferença de que no caso presente na figura 27 a acurácia apesar de apresentar a mesma tendência, é um pouco menor.

5.1.3 Comentários gerais e conclusão

Com um modelo baseado em GCN foi possível criar um classificador que possui uma assertividade muito alta para os casos de dinâmicas Maki-Thompson simuladas em redes artificiais LFR. Observa-se que quando os grafos LFR possuem comunidades de mesmo tamanho a performance fica em geral maior do que para grafos LFR com comunidades de diversos tamanhos. Isso pode acontecer por vários motivos, incluindo o fato de que nos casos em que as comunidades são de diversos tamanhos também variava a quantidade de comunidades, sendo assim, em grafos com muitas comunidades o problema de classificação se complexifica pois existirão mais classes. Como extensão das simulações aqui realizadas, é possível analisar qual o impacto de outras características topológicas do grafo na acurácia do modelo preditivo, como por exemplo o grau médio dos propagadores iniciais, a modularidade do grafo, a quantidade de comunidades, dentre outras propriedades.

5.2 Detecção de comunidades

Para usar os modelos de GNN no contexto de detecção de comunidades é preciso definir uma regra de aprendizado. Existem trabalhos que levam em consideração um aprendizado supervisionado em detecção de comunidades (CHEN; LI; BRUNA, 2019), ou seja, se tem a informação sobre qual comunidade o vértice pertence. Além disso, existem outros trabalhos que levam em conta a informação parcial de alguns vértices e com isso fazer o ajuste dos parâmetros via aprendizado semi-supervisionado (LI; HAN; WU, 2018). No entanto, algoritmos tradicionais da comunidade de ciência de redes complexas como o *Label Propagation* (GARZA; SCHAEFFER, 2019), *Louvain* (BLONDEL *et al.*, 2008), *Girvan-Newman* (GIRVAN; NEWMAN, 2002), *FastGreedy* (CLAUSET; NEWMAN; MOORE, 2004) são não-supervisionados, ou seja, não se tem nenhuma informação sobre qualquer vértice da rede e a tarefa de detectar comunidades é feita com base somente na estrutura do grafo. Alguns trabalhos recentes tem explorado o potencial de extrair representações para os vértices de forma não-supervisionada utilizando GNNs com o objetivo de aumentar a performance dos modelos em tarefas de classificação de vértices (JIN *et al.*, 2020; ZHU; DU; YAN, 2020). Dado este contexto, usaremos um modelo baseado em GNN para extrair as representações vetoriais para os vértices e então utilizar as representações obtidas para encontrar comunidades no grafo. Seguindo a ideia usada em (KIPF; WELLING, 2016) para obter as representações para os vértices de maneira não-supervisionada que consiste na ideia de treinar o modelo GCN para prever a existência de arestas. Para isto, é utilizado uma ideia similar aos autoencoders (HINTON; ZEMEL, 1994) em redes neurais que utiliza representações latentes para reconstruir na saída algo similar ao que foi dado de entrada. No caso de grafos, isto pode ser feito tentando reconstruir a matriz de adjacências do grafo a partir das representações geradas pelo modelo. Ou seja, o objetivo é estimar uma matriz do seguinte formato:

$$\hat{\mathbf{A}} = \begin{bmatrix} \hat{p}_{11} & \hat{p}_{12} & \cdots & \hat{p}_{1N} \\ \hat{p}_{21} & \hat{p}_{22} & \cdots & \hat{p}_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \hat{p}_{N1} & \hat{p}_{N2} & \cdots & \hat{p}_{NN} \end{bmatrix} \tag{5.9}$$

Em que $\hat{p}_{ij}$ na matriz (5.12) representa a probabilidade estimada de existir conexão entre o vértice i e j. Para obter a matriz $\hat{\mathbf{A}}$ a partir das representações vetoriais de saída de um modelo baseado em GNN é necessário utilizar alguma medida de similaridade entre vetores como por exemplo um simples produto interno, que por sua vez, é relacionado com a similaridade de cosseno sem ter a normalização. Portanto, utilizando a saída de um modelo GCN temos:

$$\mathbf{H}^{(L)} = GCN(\mathbf{X}, \mathbf{A}) \tag{5.10}$$

Em que $\mathbf{H}^{(L)}$ na (5.10) é a matriz de representações para os vértices da rede obtida na última camada L. Neste caso, $\mathbf{H}^{(L)} \in \mathbb{R}^{N \times F}$ em que cada linha de $\mathbf{H}^{(L)}$ é composta pela representação de cada vértice $\mathbf{h}_v^{(L)}$ para todo $v \in \mathcal{V}$. Considerando a reconstrução da matriz de adjacências como sendo (KIPF; WELLING, 2016) :

$$\hat{\mathbf{A}} = \sigma(\mathbf{H}^{(L)}(\mathbf{H}^{(L)})^T) \tag{5.11}$$

Em que na (5.11) a função σ é a *sigmoid*, que por sua vez, é aplicada em cada elemento da matriz resultante da operação $\mathbf{H}^{(L)}(\mathbf{H}^{(L)})^T$. Escrevendo a (5.11) explicitamente temos:

$$\hat{\mathbf{A}} = \begin{bmatrix} \sigma(\mathbf{h}_1 \cdot \mathbf{h}_1^T) & \sigma(\mathbf{h}_1 \cdot \mathbf{h}_2^T) & \cdots & \sigma(\mathbf{h}_1 \cdot \mathbf{h}_N^T) \\ \sigma(\mathbf{h}_2 \cdot \mathbf{h}_1^T) & \sigma(\mathbf{h}_2 \cdot \mathbf{h}_2^T) & \cdots & \sigma(\mathbf{h}_2 \cdot \mathbf{h}_N^T) \\ \vdots & \vdots & \ddots & \vdots \\ \sigma(\mathbf{h}_N \cdot \mathbf{h}_1^T) & \sigma(\mathbf{h}_N \cdot \mathbf{h}_2^T) & \cdots & \sigma(\mathbf{h}_N \cdot \mathbf{h}_N^T) \end{bmatrix} \tag{5.12}$$

Na matriz (5.12) cada elemento $\sigma(\mathbf{h}_i \cdot \mathbf{h}_j)$ representa a probabilidade estimada $\hat{p}_{ij}$ de existir conexão entre o vértice i e j, sendo que $\mathbf{h}_i \cdot \mathbf{h}_j$ é o produto interno das representações vetoriais para vértice i e j.

Tendo definido a saída desejada se precisa de uma função para quantificar a qualidade da predição dada pela modelo e definir a regra do aprendizado, para isto é possível utilizar a entropia binária cruzada ponderada:

$$\mathcal{L} = -\sum_i \sum_j W_1(A_{ij} log(\hat{p}_{ij})) + W_2(1 - A_{ij}) log(1 - \hat{p}_{ij})$$

Em que W_1 e W_2 são constantes para ponderar o resultado da função perda. É necessário ponderar pois geralmente a matriz de adjacências de um grafo é muito esparsa, tendo então muitos casos em que não existe a conexão entre dois vértices e portanto o valor correspondente na matriz é nulo. Se não houver ponderação o modelo poderá convergir rapidamente para a predição $\hat{p}_{ij} = 0$ para todos os vértices i e j, sendo que se fizer isto acertará a maioria dos casos.

Os pesos são definidos da seguinte maneira:

$$W_1 = \frac{\sum_i \sum_j (1 - A_{ij})}{N^2} \tag{5.13}$$

$$W_2 = \frac{\sum_i \sum_j A_{ij}}{N^2} \tag{5.14}$$

Na equação (5.10) falta definir qual é a arquitetura do modelo *GCN*. Para isto, consideramos um modelo de GCN da seguinte maneira para obter a representação para um nó u:

$$\mathbf{h}_u^{(1)} = \mathbf{W}^{(1)}\mathbf{h}_u^{(0)} + \mathbf{b}^{(1)} \tag{5.15}$$

$$\mathbf{h}_u^{(\ell)} = \mathbf{h}_u^{(\ell-1)} + \sigma\left(\mathbf{W}^{(\ell)} \sum_{v \in \mathcal{N}(u)\cup\{u\}} \frac{\mathbf{h}_v^{(\ell-1)}}{\sqrt{|\mathcal{N}(u)||\mathcal{N}(v)|}} + \mathbf{b}^{(\ell)}\right) \tag{5.16}$$

Na equação (5.15) o expoente ℓ indica a camada de GCN, podendo variar de 2 até qualquer quantidade escolhida pelo usuário.

Além disso, nas representações da atual camada são somadas as representações vetoriais da camada anterior. Experimentos realizados utilizando este tipo de ideia mostraram uma melhor performance em casos de classificação (KIPF; WELLING, 2017).

Também pode-se escrever as equações (5.15) e (5.16) de forma matricial:

$$\mathbf{H}^{(1)} = \mathbf{H}^{(0)}\mathbf{W}^{(1)} + \mathbf{B}^{(1)} \tag{5.17}$$

$$\mathbf{H}^{(\ell)} = \mathbf{H}^{(\ell-1)} + \sigma\left(\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{H}^{(\ell-1)}\mathbf{W}^{(\ell)} + \mathbf{B}^{(\ell)}\right) \tag{5.18}$$

Como em muitas redes, como por exemplo nos modelos de redes artificiais, não se possui nenhum atributo associado a cada vértice para ser utilizado como representação inicial, deve-se então considerar alguma representação vetorial genérica, para assim poder treinar o modelo e obter representações mais interessantes. Sendo assim, neste caso pode-se atribuir como representação inicial simplesmente a matriz identidade, ou seja, $\mathbf{H}^{(0)} = \mathbf{I}_N$.

Depois de ter treinado o modelo GCN é então aplicado o método K-means (Lloyd, 1982) nas representações vetoriais para os vértices com o objetivo de obter as comunidades da rede.

Para avaliar a qualidade do método em detectar comunidades é comumente usado na literatura a Informação Mutua Normalizada (NMI) (AMELIO; PIZZUTI, 2015) que pode ser definida da seguinte maneira:

$$NMI(A,B) = \frac{-2\sum_{i=1}^{C_A}\sum_{j=1}^{C_B} N_{ij} log\left(\frac{N_{ij}N}{N_i N_j}\right)}{\sum_{i=1}^{C_A} N_i log\left(\frac{N_i}{N}\right) + \sum_{i=1}^{C_B} N_j log\left(\frac{N_j}{N}\right)} \tag{5.19}$$

Se as duas divisões em cluster dos conjuntos A e B forem equivalentes a medida NMI vale 1 e caso forem completamente independentes vale 0. Além disso, na equação 5.19 N é o número total de vértices no grafo, C_A é a quantidade de comunidades do conjunto A e C_B

a quantidade de comunidades do conjunto B. N_{ij} é a quantidade de vértices em comum da comunidade i do conjunto A e da comunidade j do conjunto B. N_i e N_j são o número total de vértices na comunidade i e j de seus respectivos conjuntos.

Realizando o treinamento na rede da Figura 8 com 4 camadas de GCN e escolhendo a dimensão da representação vetorial para os nós de 128 e utilizando o otimizador de Adam (KINGMA; BA, 2017) com taxa de aprendizado de 0.01 e então realizando uma redução de dimensionalidade por t-SNE obtemos o seguinte gráfico:

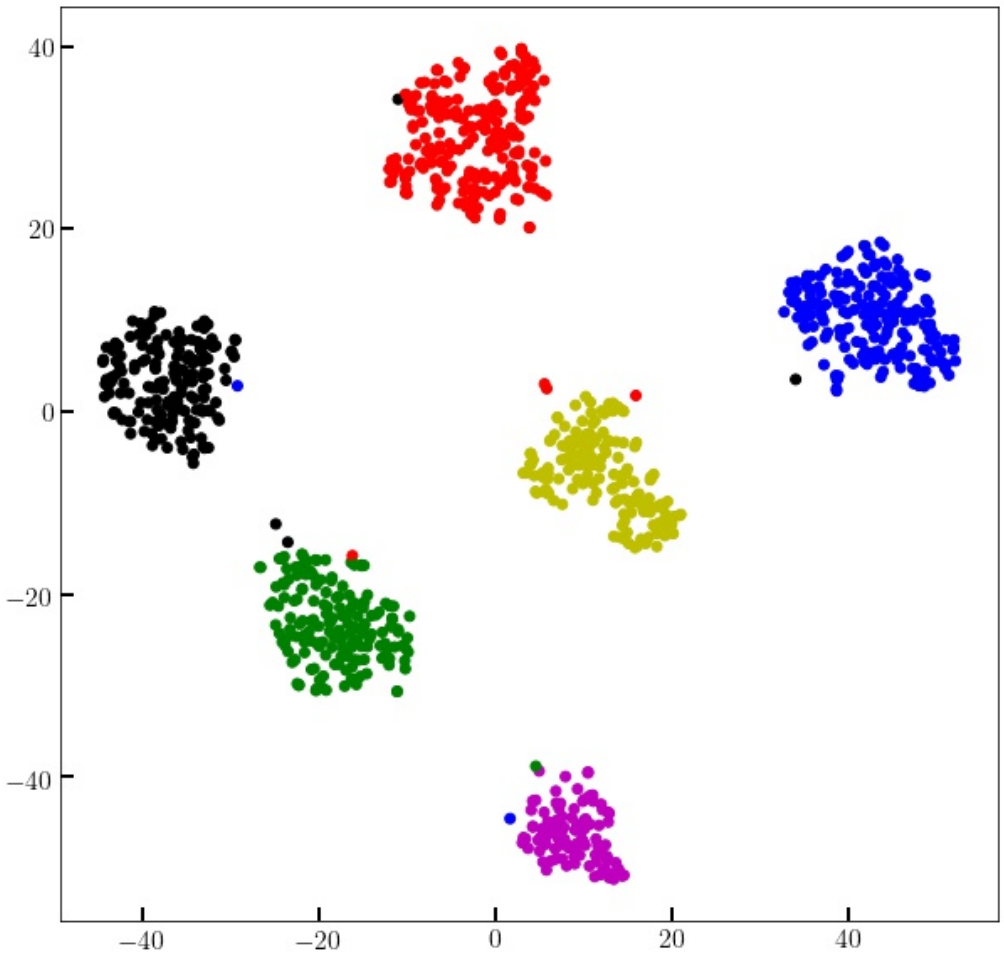

Figura 28 – Representações vetoriais em 2 dimensões para a rede do modelo LFR mostrada na Figura 8.

Fonte: Elaborada pelo autor.

Como pode-se observar na Figura 28 o modelo consegue representar bem as estruturas de comunidades presentes na rede, dado que vértices pertencentes a mesma comunidade apresentam uma representação vetorial similar.

5.2.1 Experimentos em redes LFR

Para avaliar a detecção de comunidades via GNN será dividido em dois segmentos de experimentos. O primeiro segmento será com redes do modelo LFR em que existam comunidades com a mesma quantidade de vértices e também com a quantidade de comunidades fixada em quatro. Já o segundo segmento será em redes LFR sem a limitação de ter apenas comunidades de mesma quantidade de vértices e nem a quantidade de comunidades limitada em quatro. A escolha experimental foi feita desta maneira tendo em vista explorar limitações que podem ser geradas pelo uso de um método de clustering após obter as representações, pois o método de clustering utilizado foi o K-means (Lloyd, 1982) e possui limitações tanto devido a agrupamentos de diferentes tamanhos e também em quantidade grande de grupos (HUANG, 2004).

5.2.1.1 Redes LFR com comunidades de tamanhos iguais

Para analisar o potencial do modelo baseado GCN em grafos com comunidades de quantidades de vértices iguais foi considerado utilizar o modelo LFR com os parâmetros:

- $N = 400$, quantidade de vértices
- $\tau_1 = 3$, expoente da distribuição lei de potência para os graus.
- $k_{min} = 10$, quantidade mínima do grau de um vértice.
- $k_{max} = 20$, quantidade máxima do grau de um vértice.
- $min_c = 100$, quantidade de vértices miníma em uma comunidade.
- $max_c = 100$, quantidade de vértices máxima em uma comunidade.

Além dos parâmetros definidos acima, o parâmetro de mistura μ que regula a fração de conexões entre as comunidades foi variado no intervalo 0.01 à 0.9, quanto menor o parâmetro mais definidas são as comunidades e quanto maior menos estrutura de comunidade possui o grafo. Para cada configuração de parâmetros foram geradas 10 grafos e os resultados reportados são uma média entre os grafos.

Na figura abaixo um exemplo de rede LFR é mostrado para cada configuração de parâmetros.

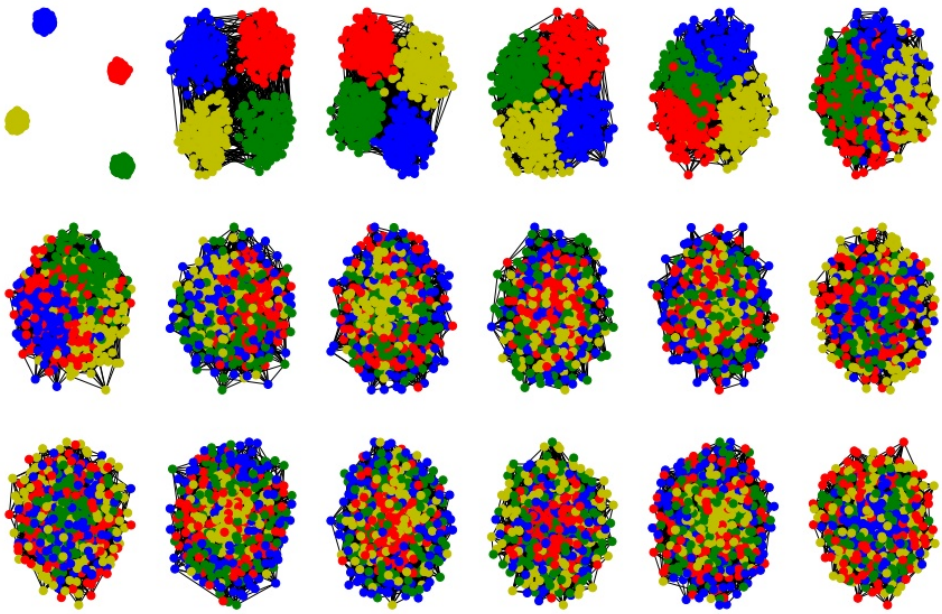

Figura 29 – Redes geradas pelo modelo LFR com $N = 400$ e diferentes valores de μ.

Fonte: Elaborada pelo autor.

Na figura 29 estão representadas 18 redes do modelo LFR. No canto esquerdo superior é uma rede com os parâmetros $N = 400$, $\tau_1 = 3$, $k_{min} = 10$, $k_{max} = 20$, $min_c = 100$, $max_c = 100$ e $\mu = 0.01$. Indo para a direita são mostradas redes LFR com os mesmos valores de parâmetros porém com o μ crescendo. No canto inferior direito é uma rede LFR com $\mu = 0.9$, neste caso a rede já perde a característica modular.

Com o objetivo de visualizar as representações obtidas no caso das redes mostradas na figura 29 foi reduzida a dimensão após o treinamento de um modelo GCN para cada rede e gerada a figura abaixo:

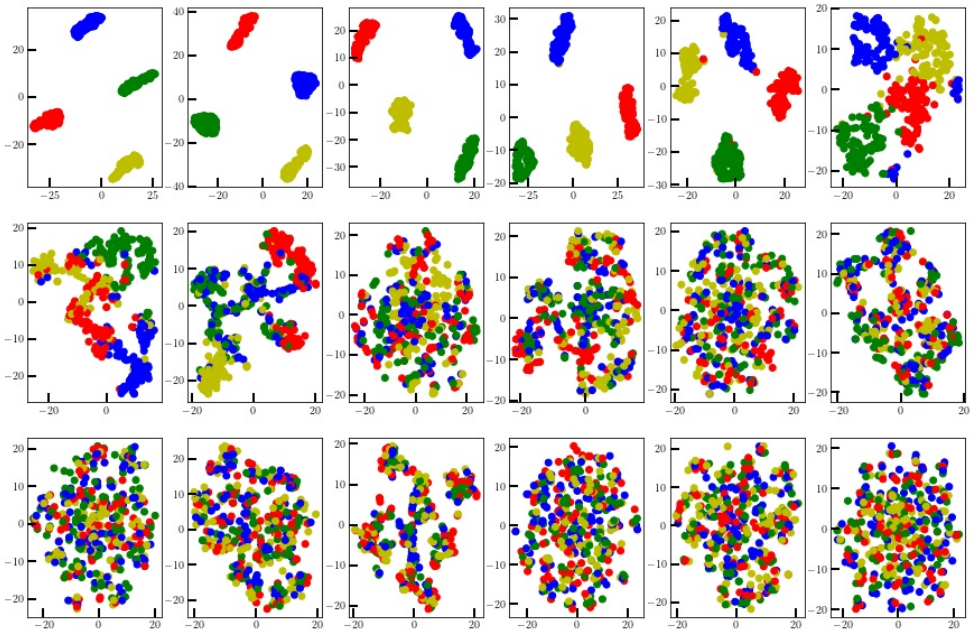

Figura 30 – Representações vetoriais em redes LFR homogêneas obtidas com o modelo GCN e com dimensionalidade reduzida por t-SNE

Fonte: Elaborada pelo autor.

Na figura 30 foi plotado em duas dimensões as representações obtidas para os grafos da figura 29 a partir do modelo baseado em GNN. Como pode-se observar visualmente que os vértices que pertencem a mesma comunidade possuem representações similares, pelo menos nos casos em que as comunidades são bem definidas (primeira linha de imagens).

Para obter as comunidades das redes artificiais LFR foi considerado um modelo descrito pelas equações 5.17 e 5.18 com um total de 3 camadas camadas GCN, ou seja, $\ell = 3$. Além disso, a dimensão das representações nas camadas foi definida como 32, ou seja, $\mathbf{H}^{(\ell)} \in \mathbb{R}^{32}$. Ademais, a quantidade de épocas foi definida como sendo 150 e a taxa de aprendizado como 0.01 no otimizador Adam (KINGMA; BA, 2017).

Comparando o modelo baseado em GNN com alguns métodos tradicionais (*Label Propagation* (GARZA; SCHAEFFER, 2019), *Louvain* (BLONDEL *et al.*, 2008), *Girvan-Newman* (GIRVAN; NEWMAN, 2002), *FastGreedy* (CLAUSET; NEWMAN; MOORE, 2004)) nas redes LFR usando a métrica NMI foi obtido os seguintes resultados:

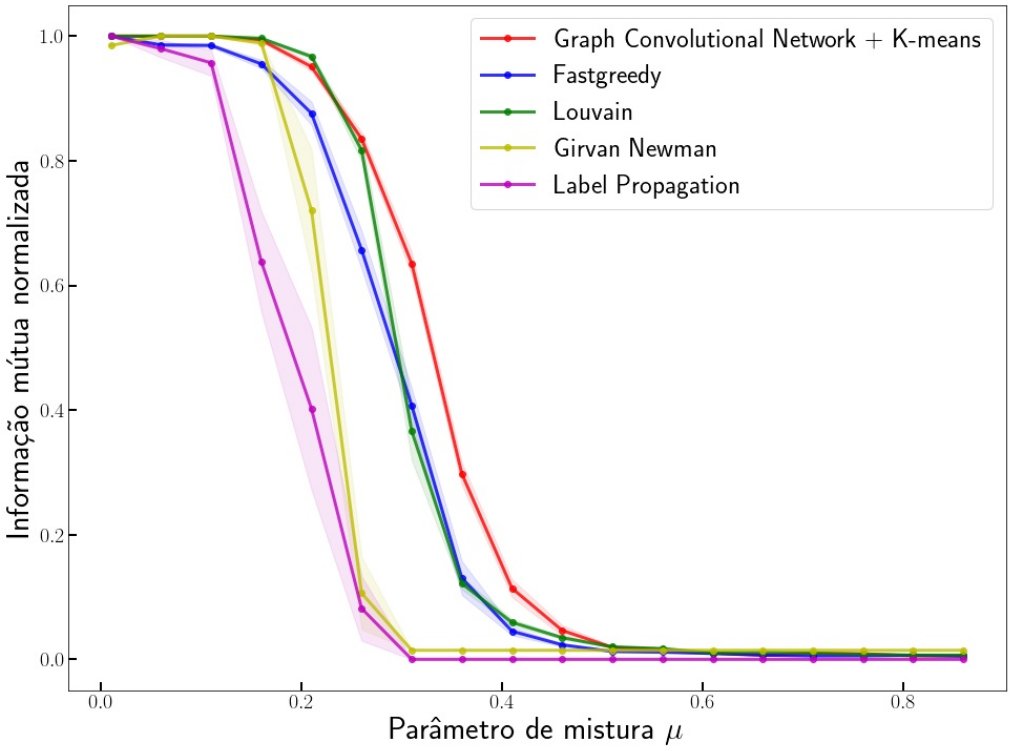

Figura 31 – Comparação com métodos tradicionais para redes LFR com $N = 400$, $\tau_1 = 3$, $k_{min} = 10$, $k_{max} = 20$, $min_c = 100$, $max_c = 100$ e variando o parâmetro de mistura μ entre 0.01 e 0.9

Fonte: Elaborada pelo autor.

Na figura 31 cada ponto é uma média da performance em termos de NMI de cada modelo em 10 grafos LFR gerados aleatoriamente. As sombras foram geradas a partir do erro padrão de cada grupo de observações. Como pode-se observar o modelo GCN + K-means em vermelho apresenta um desempenho competitivo com o método de Louvain em verde e superior ao restante. Para os casos em que o parâmetro de mistura é maior o modelo GCN tem uma performance um pouco superior a todos.

5.2.1.2 Redes LFR com comunidades de diversos tamanhos

Para analisar o potencial do modelo baseado GCN em grafos com comunidades de quantidades de vértices variada foi considerado utilizar o modelo LFR com os parâmetros:

- $N = 400$, quantidade de vértices
- $\tau_1 = 3$, expoente da distribuição lei de potência para os graus.
- $\tau_2 = 1.5$, expoente da distribuição lei de potência para a quantidade de vértices nas comunidades.
- $k_{min} = 10$, quantidade mínima do grau de um vértice.
- $k_{max} = 45$, quantidade máxima do grau de um vértice.
- $min_c = 25$, quantidade de vértices miníma em uma comunidade.
- $max_c = 200$, quantidade de vértices máxima em uma comunidade.

Com essa escolha de parâmetros e variando o parâmetro de mistura μ é possível obter redes com quantidades de comunidades diversas e também com quantidade variadas de vértices nas comunidades.

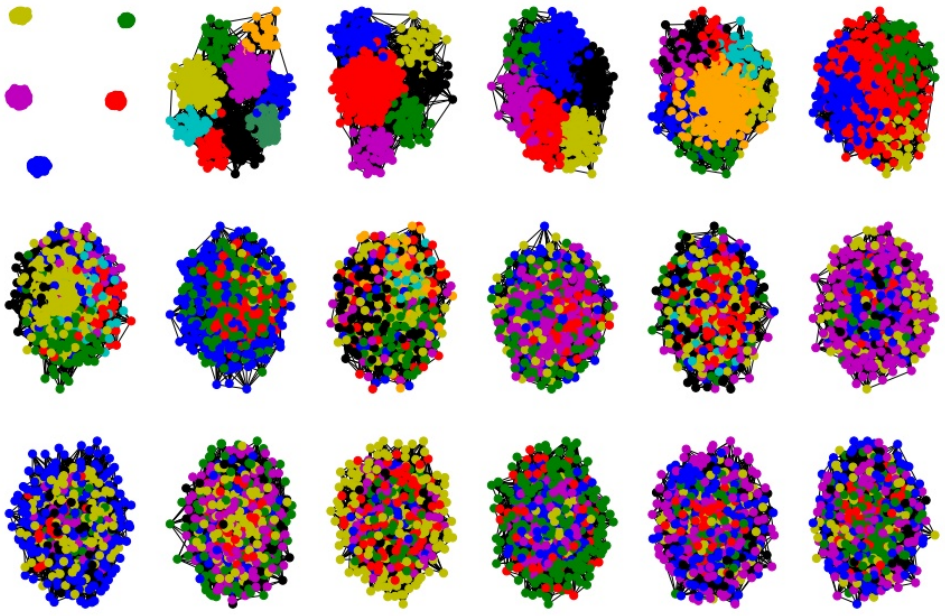

Figura 32 – Redes heterogêneas geradas pelo modelo LFR com $N = 400$ e diferentes valores de μ.

Fonte: Elaborada pelo autor.

Na figura 32 são mostradas 18 redes com os parâmetros $N = 400$, $\tau_1 = 3$, $\tau_2 = 1.5$, $k_{min} = 10$, $k_{max} = 45$, $min_c = 25$, $max_c = 200$ com o parâmetro de mistura μ sendo 0.01 na rede plotada no canto superior esquerdo e aumentando nos grafos seguintes até chegar em $\mu = 0.9$ no último grafo do canto inferior direito. Nota-se que com essa escolha de parâmetros é possível criar grafos com quantidades de comunidades distintas. Na figura 32 a quantidade de comunidades da esquerda para a direita e de cima para baixo é de 5, 9, 6, 6, 8, 4, 7, 4, 8, 5, 7, 6, 6, 6, 6, 5, 6, 6, diferentemente do experimento tratado na seção anterior em que a quantidade de comunidades era fixada em 4. Visualizando as representações vetoriais em dimensão reduzida para as redes da figura 32 obtém-se a seguinte figura:

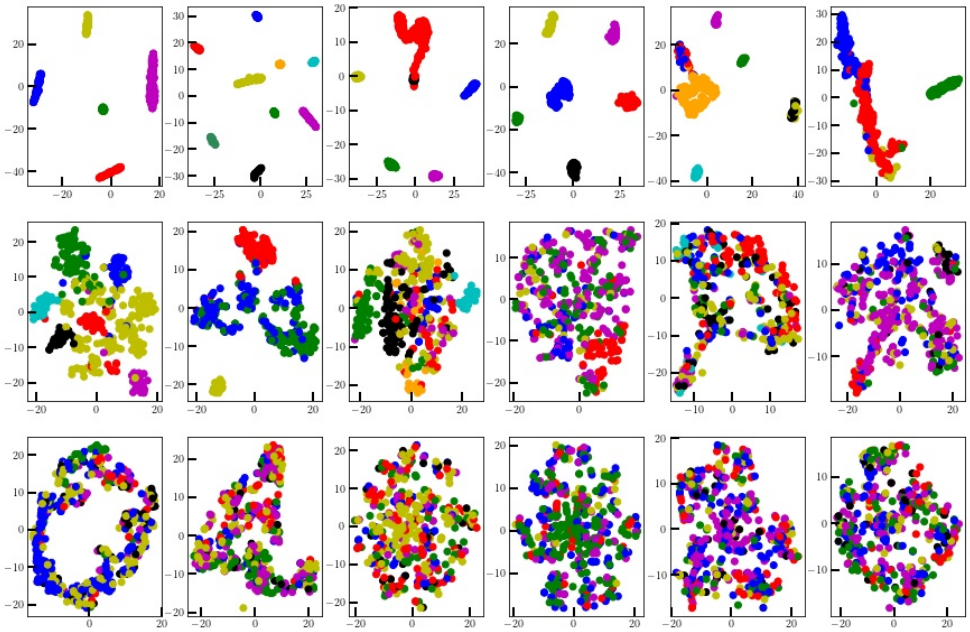

Figura 33 – Representações vetoriais em redes LFR heterogêneas obtidas com o modelo GCN e com dimensionalidade reduzida por t-SNE

Fonte: Elaborada pelo autor.

De forma análoga ao cenário de redes LFR com comunidades de tamanhos iguais, no caso de tamanhos diferentes foi considerado um modelo descrito pelas equações 5.17 e 5.18 com um total de 3 camadas camadas GCN, ou seja, $\ell = 3$. Além disso, a dimensão das representações nas camadas foi definida como 32, ou seja, $\mathbf{H}^{(\ell)} \in \mathbb{R}^{32}$. Ademais, a quantidade de épocas foi definida como sendo 150 e a taxa de aprendizado como 0.01 no otimizador Adam (KINGMA; BA, 2017). Comparando o modelo baseado em GCN com os métodos tradicionais no caso de redes heterogêneas obtém-se:

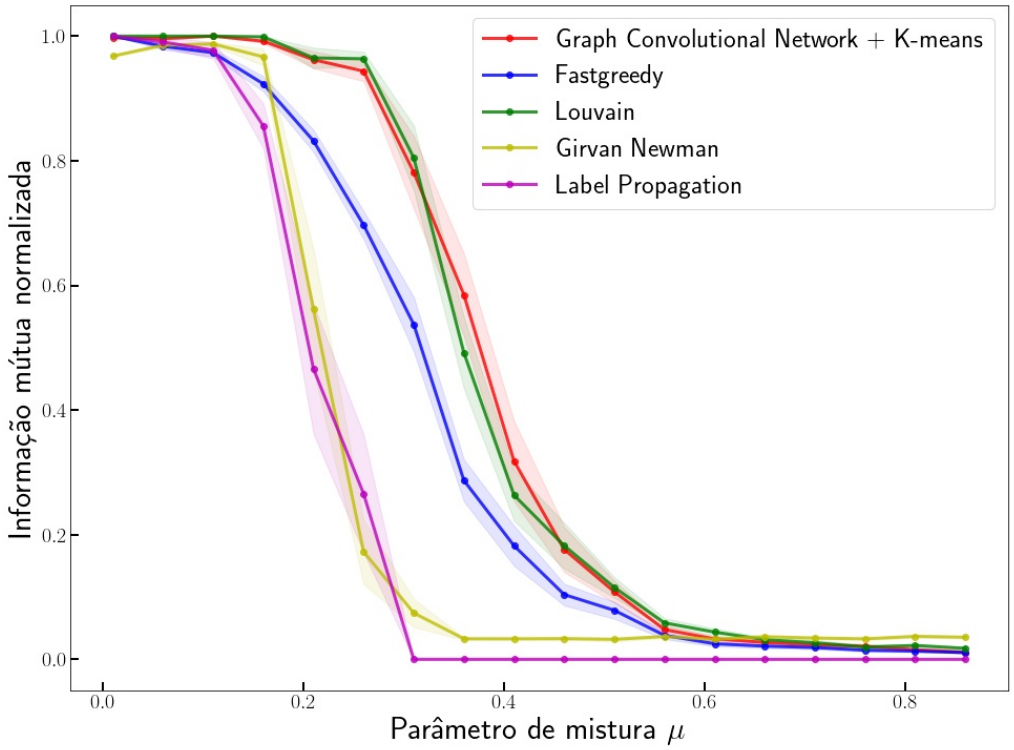

Figura 34 – Comparação com métodos tradicionais para redes LFR heterogênas com $N = 400$, $\tau_1 = 3$, $k_{min} = 10$, $k_{max} = 45$, $min_c = 25$, $max_c = 200$ e variando o parâmetro de mistura μ entre 0.01 e 0.9

Fonte: Elaborada pelo autor.

Na figura 34 cada ponto é uma média da performance dos modelos em 10 realizações diferentes de rede LFR. No cenário observado na figura 34 nota-se que o modelo GCN + K-means te um desempenho similar ao método de Louvain e superior aos outros métodos para o caso de redes heterogêneas LFR.

5.2.2 Comentários gerais e conclusão

De maneira geral o desempenho do modelo baseado em GNN foi competitivo ao melhor método (Louvain (BLONDEL *et al.*, 2008)) dentre os que foram escolhidos para a comparação. Uma comparação mais exaustiva utilizando redes LFR e suas diferentes possíveis combinações de parâmetros poderia explorar cenários em que um ou outro método seria favorecido. De maneira favorável ao método baseado em GNN pode-se destacar que além de obter representações vetoriais para os vértices de maneira consistente com a estrutura de comunidade, essas representações podem ser usadas para outros propósitos, como por exemplo classificação ou regressão em vértices em redes sociais. Além disso, existem redes sociais reais em que é possível obter uma representação vetorial inicial de acordo com características obtidas a partir de informações dos usuários e utilizar essas informações no modelo baseado em GNN pode ajudar no desempenho de encontrar comunidades. De maneira desfavorável ao método baseado em GNN pode-se destacar o alto custo computacional, fazendo com que seja muito mais demorado para se obter as comunidades. No cenário em que o número de vértices é extremamente grande, utilizar o método da maneira aqui descrita pode ser inviável. No entanto, é possível adaptar agregação de mensagem dos vértices para tornar mais escalável, como por exemplo no método GNNAutoScale (FEY *et al.*, 2021), e então treinar o modelo GNN para obter as representações da maneira não-supervisionada aqui descrita. Outro ponto um pouco desfavorável é que o método requer a escolha de vários parâmetros como por exemplo a quantidade de camadas GCN, a dimensão das representações, a taxa de aprendizado, o método de otimização, e todas essas escolhas complexificam o problema de detecção de comunidades e também podem impactar no desempenho obtido. Ademais, da maneira aqui descrita utilizar GNN para obter as comunidades requer saber a principio o número de comunidades que o grafo possui, pois as representações obtidas da maneira não-supervisionada são passadas para o método K-means. Em cenários reais de detecção de comunidades nem sempre isso é sabido, precisando então o usuário especificar ou utilizar algum método para inferir a quantidade de comunidades. Sendo assim, a performance do modelo pode ser prejudicada caso a quantidade de comunidades inferida ou proposta for diferente da real.

CAPÍTULO

6

CONCLUSÃO

Neste trabalho foram explorados dois cenários de aplicação de redes neurais para grafos. No primeiro cenário, um problema de classificação foi definido utilizando uma dinâmica de propagação de rumor (Maki-Thompson) (MAKI; THOMPSON, 1973; ARRUDA; RODRIGUES; MORENO, 2018) em grafos. Para isto, foi simulado a dinâmica de propagação tendo como propagadores iniciais vértices que percentem ao mesmo agrupamento no grafo. Tendo a história da simulação e o rótulo de qual agrupamento que iniciou a propagação é possível definir o problema de classificação. Sendo definido o problema de classificação, foi então proposto um modelo baseado em GCN para classificar qual é o agrupamento mais provável de ter começado uma dinâmica dado um momento no tempo em que um rumor já havia se alastrado pelo grafo. Observou-se que o modelo preditivo baseado em GCN apresentou uma acurácia elevada e foi visto alguns cenários em que potencialmente o modelo possui mais limitações. Em geral, os cenários em que a acurácia é degradada são em grafos em que as estruturas de comunidades não são tão bem definidas e a taxa de propagação da dinâmica é alta, enquanto a taxa de recuperação é baixa. Além disso, observou-se que em grafos em que as comunidades possuem diversos tamanhos a performance do modelo preditivo também é levemente prejudicada. Possíveis futuras investigações para esta aplicação podem ser feitas com o objetivo de explorar questões topológicas do grafo e como isso impacta na dinâmica Maki-Thompson e no modelo preditivo baseado em GCN. Ademais, podem ser explorados modificações nas camadas de agregação da mensagem como por exemplo a GAT (VELICKOVIC *et al.*, 2017).

No segundo cenário de aplicação foi analisado o potencial do aprendizado das representações latentes para os vértices para a tarefa de detecção de comunidades em grafos. Para isto, foram simuladas redes artificiais a partir do modelo LFR (LANCICHINETTI; FORTUNATO; RADICCHI, 2008) fixando algumas propriedades e variando o parâmetro de mistura. A métrica de avaliação de desempenho utilizada foi a informação mútua normalizada (AMELIO; PIZZUTI, 2015). Foram testados dois cenários possíveis para parâmetros no modelo LFR, sendo que em um deles eram produzidos grafos com quantidade de comunidades fixas e tamanhos iguais e no

segundo cenário grafos com quantidade de comunidades variável e tamanho de comunidades heterogêneos. Observou-se nos dois cenários em comparação com métodos clássicos de redes complexas, utilizar as representações por GCN apresenta uma performance competitiva com o melhor modelo dentre os que comparados. Os modelos tradicionais de redes complexas que foram utilizados na comparação são: *Louvain* (BLONDEL *et al.*, 2008), *Label Propagation* (GARZA; SCHAEFFER, 2019), *Girvan-Newman* (GIRVAN; NEWMAN, 2002), *FastGreedy* (CLAUSET; NEWMAN; MOORE, 2004). Dentre os modelos comparados o que mais se destacou foi o Louvain e obteve uma performance similar ao método baseado em GCN. Uma vantagem para o modelo baseado em GCN é que é possível facilmente utilizar variáveis associadas a cada um dos vértices do grafo e isso pode contribuir para aumentar a performance. No entanto, uma desvantagem é que o custo computacional é muito maior do que os métodos tradicionais de redes. Como futuras investigações também podem ser explorados outras camadas de agregação como por exemplo a GAT (VELICKOVIC *et al.*, 2017) e também podem ser testados em outros modelos de grafos artificiais como por exemplo os modelos estocásticos de bloco (HOLLAND; LASKEY; LEINHARDT, 1983).

REFERÊNCIAS

ALBERT, R.; BARABÁSI, A.-L. Statistical mechanics of complex networks. **Rev. Mod. Phys.**, American Physical Society, v. 74, p. 47–97, Jan 2002. Citado nas páginas 23 e 31.

AMELIO, A.; PIZZUTI, C. Is normalized mutual information a fair measure for comparing community detection methods? In: **2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)**. [S.l.: s.n.], 2015. p. 1584–1585. Citado nas páginas 74 e 85.

ARRUDA, G. F. de; RODRIGUES, F. A.; MORENO, Y. Fundamentals of spreading processes in single and multilayer complex networks. **Physics Reports**, v. 756, p. 1 – 59, 2018. ISSN 0370-1573. Fundamentals of spreading processes in single and multilayer complex networks. Citado nas páginas 38 e 85.

BARRAT, A.; BARTHELEMY, M.; VESPIGNANI, A. **Dynamical Processes on Complex Networks**. 1. ed. [S.l.: s.n.], 2008. Citado nas páginas 23 e 35.

BENGIO, Y.; DUCHARME, R.; VINCENT, P.; JANVIN, C. A neural probabilistic language model. **J. Mach. Learn. Res.**, JMLR.org, v. 3, n. null, p. 1137–1155, mar 2003. ISSN 1532-4435. Citado na página 44.

BLONDEL, V. D.; GUILLAUME, J.-L.; LAMBIOTTE, R.; LEFEBVRE, E. Fast unfolding of communities in large networks. **Journal of Statistical Mechanics: Theory and Experiment**, IOP Publishing, v. 2008, n. 10, p. P10008, oct 2008. Citado nas páginas 72, 79, 83 e 86.

BOVET, A.; MAKSE, H. Influence of fake news in twitter during the 2016 us presidential election. **Nature Communications**, v. 10, 01 2019. Citado na página 24.

CATANZARO, M.; NÁ, M. B.; PASTOR-SATORRAS, R. Generation of uncorrelated random scale-free networks. **Phys. Rev. E**, American Physical Society, v. 71, p. 027103, Feb 2005. Citado na página 33.

CHEN, Z.; LI, L.; BRUNA, J. Supervised community detection with line graph neural networks. **arXiv: Machine Learning**, 2019. Citado na página 72.

CHOI, D.; CHUN, S.; OH, H.; HAN, J.; KWON, T. Rumor propagation is amplified by echo chambers in social media. **Scientific Reports**, v. 10, 01 2020. Citado na página 23.

CLAUSET, A.; NEWMAN, M. E. J.; MOORE, C. Finding community structure in very large networks. **Physical Review E**, American Physical Society (APS), v. 70, n. 6, Dec 2004. ISSN 1550-2376. Citado nas páginas 72, 79 e 86.

COSTA, L. d. F.; RODRIGUES, F. A.; TRAVIESO, G.; BOAS, P. R. V. Characterization of complex networks: A survey of measurements. **Advances in Physics**, Informa UK Limited, v. 56, n. 1, p. 167–242, Jan 2007. ISSN 1460-6976. Citado na página 29.

COTA, W.; FERREIRA, S. C. Optimized gillespie algorithms for the simulation of markovian epidemic processes on large and heterogeneous networks. **Computer Physics Communications**, v. 219, p. 303 – 312, 2017. ISSN 0010-4655. Citado na página 38.

COZZO, E.; ARRUDA, G. F. de; RODRIGUES, F. A.; MORENO, Y. **Multiplex Networks**. 1st ed.. ed. [S.l.]: Springer International Publishing, 2018. (SpringerBriefs in Complexity). Citado na página 28.

DALEY, D.; KENDALL, D. Epidemics and rumours. **Nature**, v. 204, 1964. Citado na página 35.

ERDOS, P.; RENYI, A. On random graphs. **Publicationes Mathematicae**, v. 6, p. 290–297, 1959. Citado na página 30.

FEY, M.; LENSSEN, J. E.; WEICHERT, F.; LESKOVEC, J. Gnnautoscale: Scalable and expressive graph neural networks via historical embeddings. **CoRR**, abs/2106.05609, 2021. Citado na página 83.

GARZA, S. E.; SCHAEFFER, S. E. Community detection with the label propagation algorithm: A survey. **Physica A: Statistical Mechanics and its Applications**, v. 534, p. 122058, 2019. ISSN 0378-4371. Citado nas páginas 72, 79 e 86.

GILLESPIE, D. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. **Journal of Computational Physics**, v. 22, n. 4, p. 403 – 434, 1976. ISSN 0021-9991. Citado na página 38.

GIRVAN, M.; NEWMAN, M. E. J. Community structure in social and biological networks. **Proceedings of the National Academy of Sciences**, Proceedings of the National Academy of Sciences, v. 99, n. 12, p. 7821–7826, Jun 2002. ISSN 1091-6490. Citado nas páginas 72, 79 e 86.

GLOROT, X.; BENGIO, Y. Understanding the difficulty of training deep feedforward neural networks. In: **Proceedings of Machine Learning Research**. Chia Laguna Resort, Sardinia, Italy: JMLR Workshop and Conference Proceedings, 2010. v. 9, p. 249–256. Citado nas páginas 47 e 52.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [S.l.]: The MIT Press, 2016. ISBN 0262035618. Citado nas páginas 13, 43 e 44.

GORI, M.; MONFARDINI, G.; SCARSELLI, F. A new model for learning in graph domains. In: **Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.** [S.l.: s.n.], 2005. v. 2, p. 729–734 vol. 2. Citado na página 48.

HAMILTON, W. L. Graph representation learning. **Synthesis Lectures on Artificial Intelligence and Machine Learning**, Morgan and Claypool, v. 14, n. 3, p. 1–159, 2020. Citado nas páginas 47, 49, 50 e 55.

HINTON, G. E.; ZEMEL, R. Autoencoders, minimum description length and helmholtz free energy. In: COWAN, J.; TESAURO, G.; ALSPECTOR, J. (Ed.). **Advances in Neural Information Processing Systems**. [S.l.]: Morgan-Kaufmann, 1994. v. 6. Citado na página 72.

HOLLAND, P. W.; LASKEY, K. B.; LEINHARDT, S. Stochastic blockmodels: First steps. **Social Networks**, v. 5, n. 2, 1983. ISSN 0378-8733. Disponível em: <https://www.sciencedirect.com/science/article/pii/0378873383900217>. Citado na página 86.

HUANG, J. Z. Extensions to the k-means algorithm for clustering large data sets with categorical values. **Data Mining and Knowledge Discovery**, v. 2, p. 283–304, 2004. Citado na página 76.

JIN, W.; DERR, T.; LIU, H.; WANG, Y.; WANG, S.; LIU, Z.; TANG, J. **Self-supervised Learning on Graphs: Deep Insights and New Direction**. 2020. Citado na página 72.

KINGMA, D. P.; BA, J. **Adam: A Method for Stochastic Optimization**. 2017. Citado nas páginas 47, 62, 75, 79 e 82.

KIPF, T.; WELLING, M. Variational graph auto-encoders. **NIPS Workshop on Bayesian Deep Learning**, 2016. Citado nas páginas 25, 55, 72 e 73.

KIPF, T. N.; WELLING, M. Semi-supervised classification with graph convolutional networks. **5th International Conference on Learning Representations**, Set 2017. Citado nas páginas 23, 48, 50, 54 e 74.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. Imagenet classification with deep convolutional neural networks. **Neural Information Processing Systems**, v. 25, 01 2012. Citado nas páginas 44 e 47.

LANCICHINETTI, A.; FORTUNATO, S.; RADICCHI, F. Benchmark graphs for testing community detection algorithms. **Phys. Rev. E**, American Physical Society, v. 78, p. 046110, Oct 2008. Citado nas páginas 33 e 85.

LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, v. 86, n. 11, p. 2278–2324, 1998. Citado na página 44.

LI, Q.; HAN, Z.; WU, X.-M. Deeper Insights into Graph Convolutional Networks for Semi-Supervised Learning. In: AAAI. **The Thirty-Second AAAI Conference on Artificial Intelligence**. [S.l.], 2018. Citado na página 72.

LILJEROS, F.; EDLING, C.; AMARAL, L. The web of human sexual contacts. **Nature**, 2001. ISSN 907–908. Citado na página 23.

Lloyd, S. Least squares quantization in pcm. **IEEE Transactions on Information Theory**, v. 28, n. 2, p. 129–137, 1982. Citado nas páginas 74 e 76.

MAATEN, L. V. D.; HINTON, G. Visualizing data using t-sne. **Journal of Machine Learning Research 1**, 2008. Citado na página 52.

MAKI, D. P.; THOMPSON, M. **Mathematical models and applications**. Englewood Cliffs, N.J.: Prentice-Hall Inc, 1973. Citado nas páginas 35, 36 e 85.

MARCOTTE, E.; M, P.; HL, N.; RICE, D.; YEATES, T.; EISENBERG, D. Detecting protein function and protein-protein interactions from genome sequences. **Science**, Jul 1999. Citado na página 23.

MORENO, Y.; NEKOVEE, M.; PACHECO, A. F. Dynamics of rumor spreading in complex networks. **Phys. Rev. E**, American Physical Society, v. 69, p. 066130, Jun 2004. Citado nas páginas 36, 37 e 40.

NetworkX developer team. **NetworkX**. 2014. Disponível em: <https://networkx.github.io/>. Citado na página 33.

NEWMAN, M. E. The structure and function of complex networks. **Society for Industrial and Applied Mathematics**, v. 45, 2003. Citado na página 29.

PASTOR-SATORRAS, A. V. R. **Evolution and Structure of the Internet: A Statistical Physics Approach**. 1. ed. [S.l.]: Cambridge University Press, 2007. Citado nas páginas 30 e 32.

ROBBINS, H. A stochastic approximation method. **Annals of Mathematical Statistics**, v. 22, p. 400–407, 2007. Citado na página 47.

ROSENBLATT, F. Principles of neurodynamics: Perceptrons and the theory of brain mechanisms. **Tech. rep. Cornell Aeronautical Lab Inc Buffalo NY.**, 1961. Citado na página 45.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. **Nature**, v. 323, p. 533–536, 1986. Citado na página 47.

S, G.; RM, A.; RM, M. Networks of sexual contacts: implications for the pattern of spread of hiv. aids. **PMID**, 1989. ISSN 807-17. Citado na página 23.

SCARSELLI, F.; GORI, M.; TSOI, A. C.; HAGENBUCHNER, M.; MONFARDINI, G. The graph neural network model. **IEEE Transactions on Neural Networks**, v. 20, n. 1, p. 61–80, 2009. Citado nas páginas 23, 48, 49 e 50.

SHAH, C.; DEHMAMY, N.; PERRA, N.; CHINAZZI, M.; BARABASI, A.-L.; VESPIGNANI, A.; YU, R. **Finding Patient Zero: Learning Contagion Source with Graph Neural Networks**. 2020. Citado na página 58.

VAPNIK, V. N. **The Nature of Statistical Learning Theory**. 2nd. ed. [S.l.]: Springer, 2000. (Statistics for Engineering and Information Science). Citado na página 45.

VELICKOVIC, P.; CUCURULL, G.; CASANOVA, A.; ROMERO, A.; LIò, P.; BENGIO, Y. Graph attention networks. arXiv, 2017. Citado nas páginas 85 e 86.

WEBER, M.; DOMENICONI, G.; CHEN, J.; WEIDELE, D. K. I.; BELLEI, C.; ROBINSON, T.; LEISERSON, C. E. Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics. **CoRR**, 2019. Citado na página 54.

XINYI, Z.; CHEN, L. Capsule graph neural network. In: **International Conference on Learning Representations**. [S.l.: s.n.], 2019. Citado na página 54.

YANG, Z.; ALGESHEIMER, R.; TESSONE, C. A comparative analysis of community detection algorithms on artificial networks. **Scientific Reports**, v. 6, 08 2016. Citado na página 33.

ZHANG, M.; CHEN, Y. Link prediction based on graph neural networks. In: **Advances in Neural Information Processing Systems 31**. [S.l.]: Curran Associates, Inc., 2018. p. 5165–5175. Citado nas páginas 48 e 54.

ZHANG, M.; CUI, Z.; NEUMANN, M.; CHEN, Y. An end-to-end deep learning architecture for graph classification. In: **AAAI**. [S.l.: s.n.], 2018. Citado na página 54.

ZHU, Q.; DU, B.; YAN, P. **Self-supervised Training of Graph Convolutional Networks**. 2020. Citado na página 72.


ICMC
USP
SÃO CARLOS