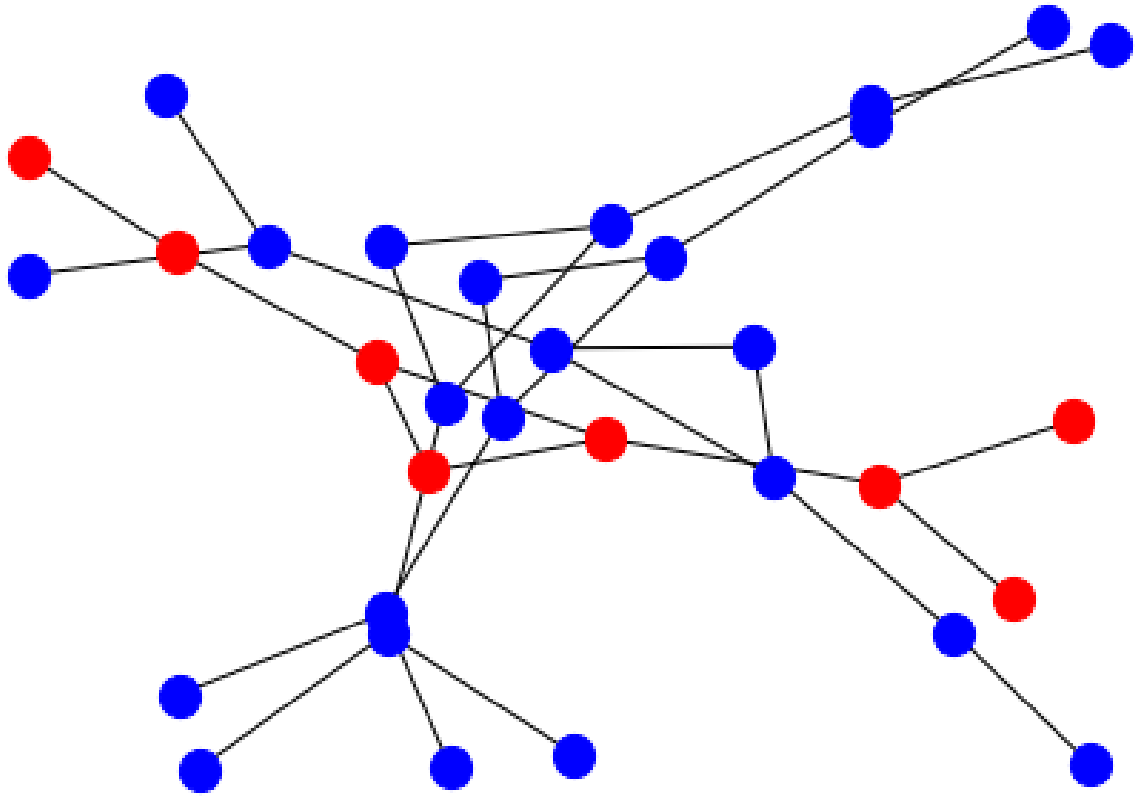
ARQUITETURAS GNN

ALUNA : ANNA PATRICIA PACHAS MANRIQUE

PROFESSOR : RENATO ROCHA



ÍNDICE

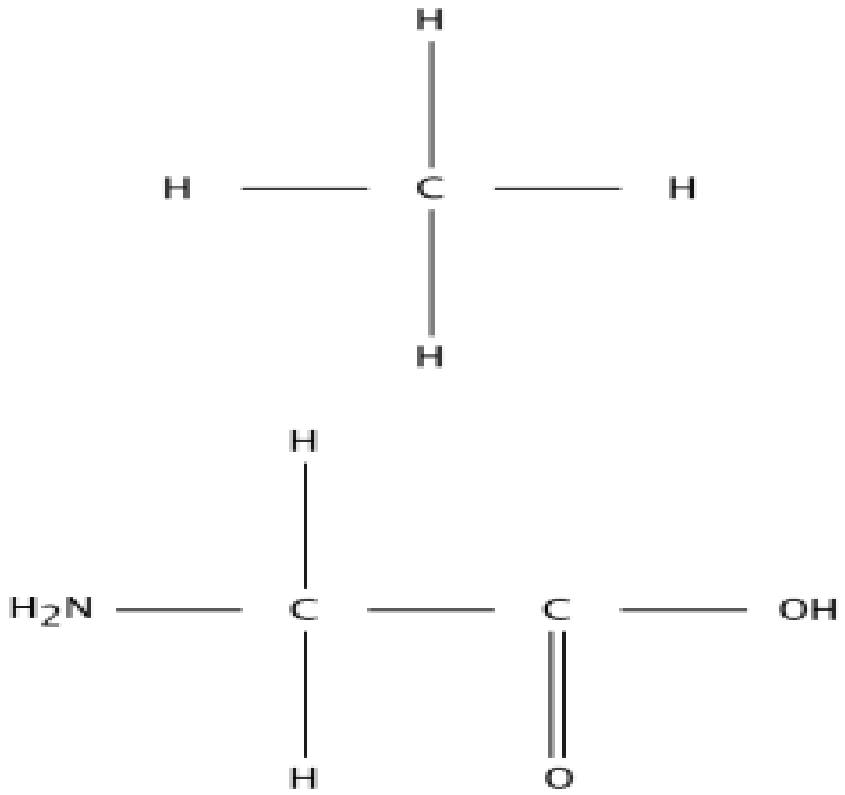
INTRODUÇÃO

- Redes neurais de grafos (GNNs) tem crescido no campo da inteligência artificial devido a sua capacidade de ingerir **tipos de dados relativamente não estruturados (grafos)** como **dados de entrada**.
- Alguns **elementos da arquitetura GNN** é conceitualmente **semelhante** às redes neurais tradicionais ,outros elementos representam um **afastamento das técnicas tradicionais** de aprendizagem profunda.
- **A arquitetura de aprendizagem** que tem foi projetada para **processar os referidos grafos** e a rede neural de grafos chamadas **(GNN).**

INTRODUÇÃO

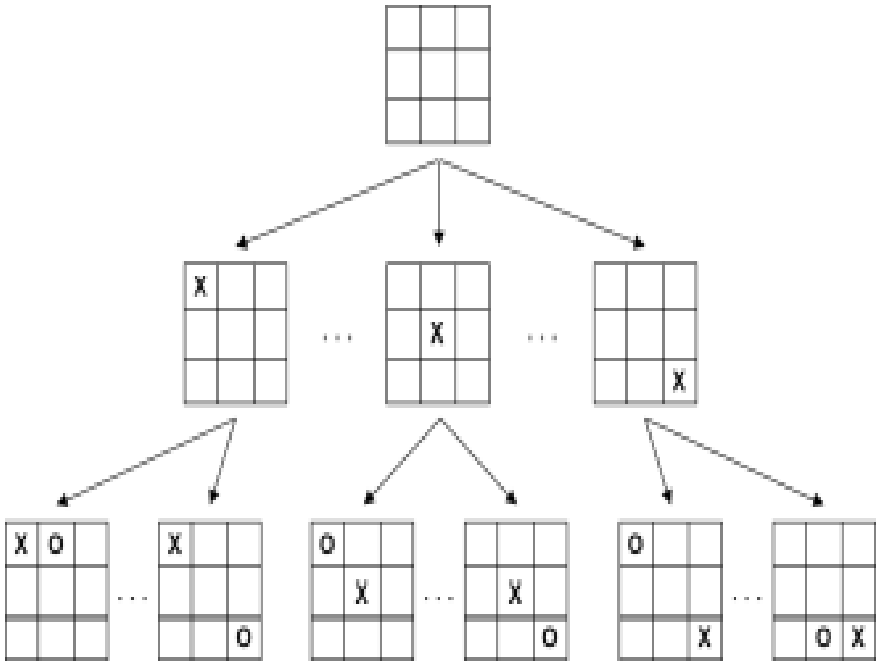
- **Grafo** é um conjunto de **vértices** distintos (representando itens ou entidades) e estão unidos pelas **arestas** (representando relacionamentos).
- Os grafos alimentados em uma **GNN não tem requisitos estruturais estritos** ; o numero de **vértices e arestas** entre os grafos de entrada pode **mudar**. Já algoritmos baseados em **NN** precisam de **entradas estruturadas** com dimensões estritamente definidas.
- Desta forma, GNNs **pode lidar com dados não estruturados** , uma propriedade que os torna **valiosos** em certos domínios de **problemas** onde os **dados do grafo** são abundantes.

INTRODUÇÃO



- **QUIMICA**
- Metano (superior) e glicina (inferior) estruturas moleculares representadas como gráficos.
- As arestas representam ligações eletroquímicas e vértices representam núcleos atômicos ou simples moléculas.

INTRODUÇÃO

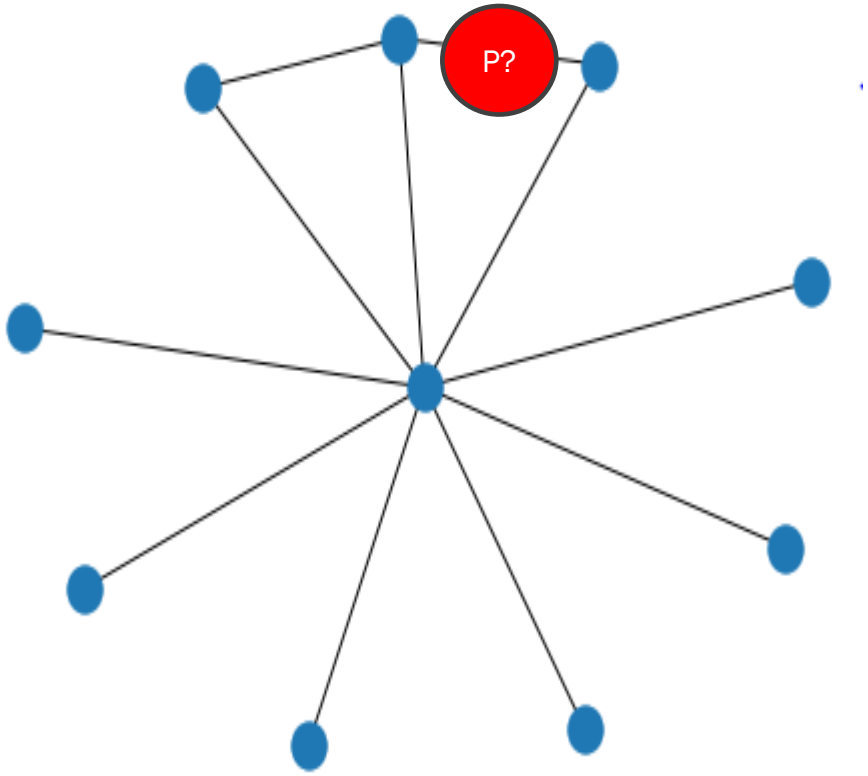


- **JOGOS**
- Uma árvore de jogo pode ser representada como um gráfo.
- Os **vértices são estados do jogo** e as **arestas** direcionadas representam ações que nos levam de **um estado** a outro.

OBJETIVOS OU TAREFAS

- O que a gente quer descobrir nos grafos ?

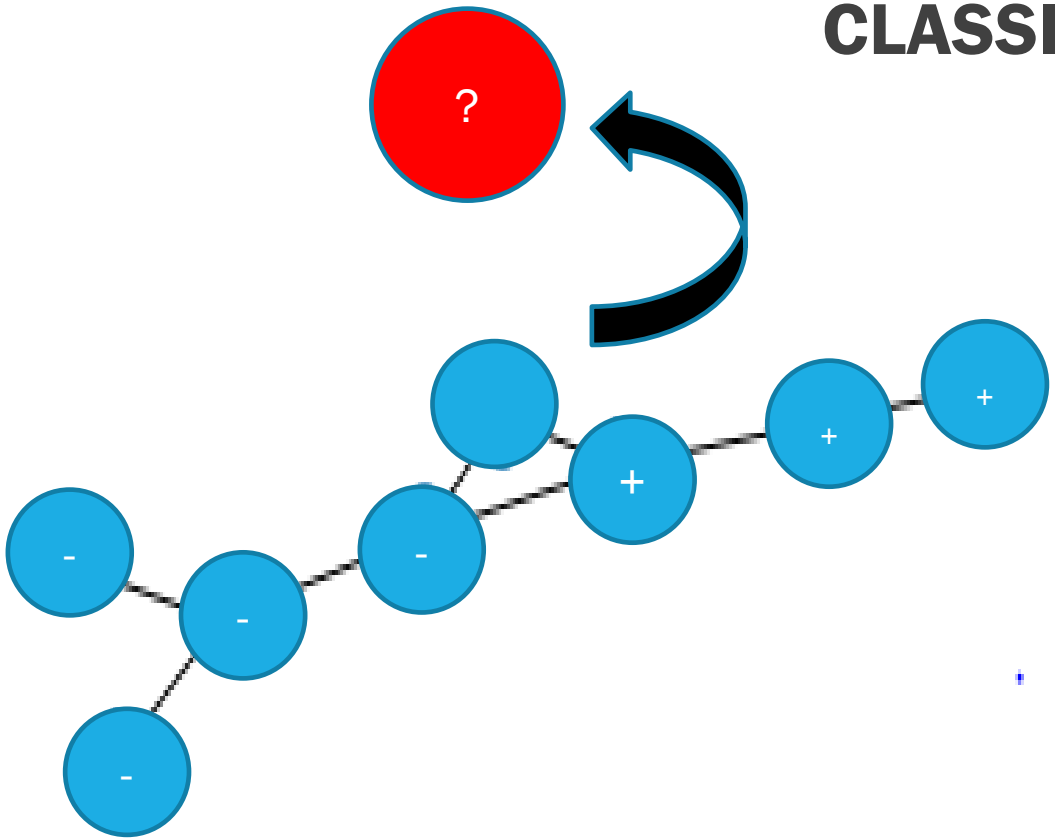
PREDIÇAO LINK



- A previsão de link é uma tarefa para estimar a **probabilidade de links** entre nós em um gráfo

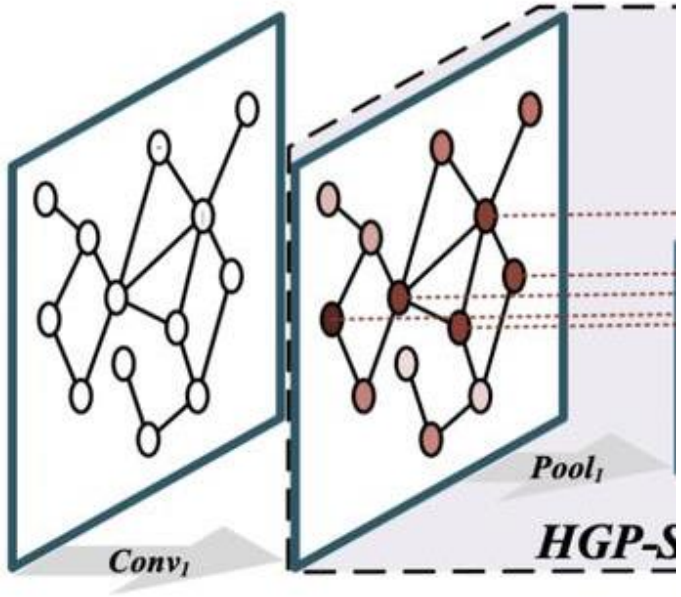
- Em uma rede social queremos saber com as informaçoes dadas , a **probabilidade de duas pessoas serem amigas** por exemplo

CLASSIFICAÇAO DE NÓ



- A tarefa de **classificação de nós** é aquela em que o algoritmo deve **determinar a rotulagem de algums nós observando os rótulos de seus vizinhos.**
- Os modelos de classificação de nós têm como **objetivo prever propriedades de nós** não existentes (conhecidas como propriedade de destino) com base em outras propriedades de nós.
- Em um **composto molecular** qual seria a carateristica de um determinado **átomo(nó)**

CLASSIFICAÇAO GRAFOS



- A que grafo cada um deles pertence?

A detecção de comunidade

- **A detecção de comunidade** é um dos problemas fundamentais na análise de redes, onde o objetivo é encontrar grupos de nós que sejam, em certo sentido, **mais semelhantes entre si do que com os outros nós**.

REDES E GRAFOS

Domínio do grafo

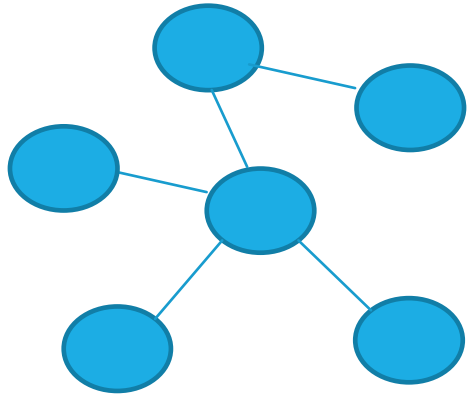
- Número de vértices (V)
- Número de Arestas(L)
- Grau do nó i (k_i)
- Distribuição de graus (P_k)
- matriz de adjacência
- Recursos
- Vizinhanças
- Estados (mudam no tempo. iteratividade)
- Embeddigs

REDES E GRAFOS - DOMÍNIO DO GRAFO

- **Elementos básicos da teoria dos grafos**, assim como os conceitos-chave necessários para entender como os GNNs são formulados e operam
- **A estrutura de dados** de entrada principal considerada é o grafo.
- grafos são formalmente definidos como um conjunto de vértices, e o conjunto de **arestas** entre esses **vértices**: formalmente, **G = G (V, E).**
- Os grafos são apenas uma forma de **codificar dados**, e, dessa forma, **cada propriedade** de um grafo representa algum **elemento real**, ou conceito nos dados.
- Compreender como os grafos podem ser usados para **representar conceitos complexos** é a **chave** para a modelagem de muitos problemas

REDES E GRAFOS - DOMÍNIO DO GRAFO

- **Número de Vértices (V)**
  - Representam itens, entidades ou objetos, que podem ser descritos naturalmente por **atributos** e seus relacionamentos com outros itens, entidades ou objetos. O conjunto de N vértices é V e ó único vértice no conjunto é v.
  - Em um grafo que representa uma rede social, **os vértices** pode representar **pessoas** .
  - **não ha requisito** para que todos **os vértices sejam homogêneos** em sua construção



REDES E GRAFOS - DOMÍNIO DO GRAFO

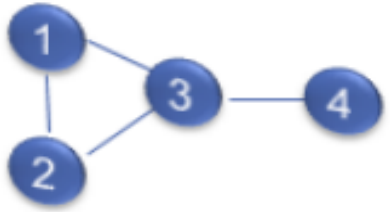
- **Número de Arestas(L)**
  - Representam e caracterizam as **relações** que existem entre itens, entidades ou objetos.
  - Em um grafo representando uma estrutura molecular, as arestas podem representar a **ligação eletroquímica** entre doiś átomos

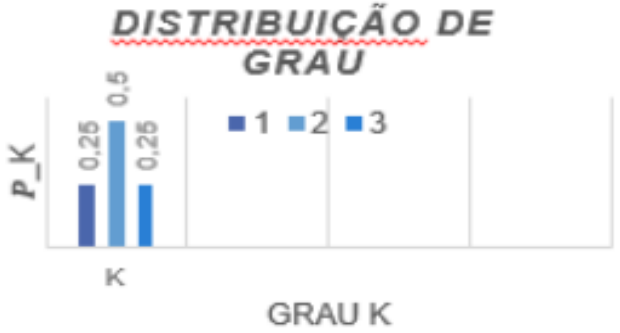
- **Grau do nó i (K_i)**
  - Cada nó tem um grau, representando o **número de ligações que tem com outros nós**.
  - O grau de um nó pode representar o **numero de amigos que uma pessoa** tem numa rede de amizade , por exemplo.
  - O numero total de ligações L, pode ser expressa como a soma dos graus dos nós dividido por 2,

$$L = \frac{1}{2}\sum_{i=1}^{N}(K_i)$$

REDES E GRAFOS – DOMINIO DO GRAFO

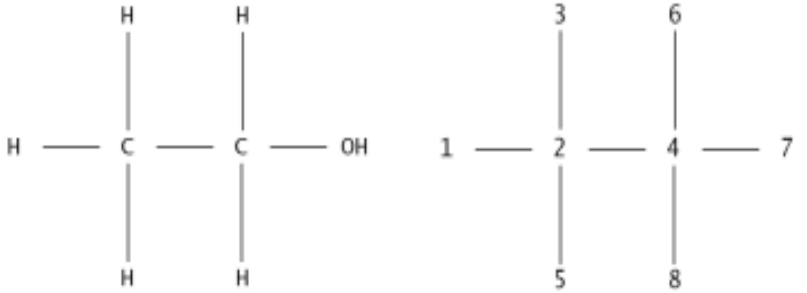
- Distribuição de graus (P_k)
  - Fornece a **probabilidade** de que um nó selecionado aleatoriamente na rede tenha **grau k.**
  - $P_k = \frac{N_k}{N}$





REDES E GRAFOS - DOMÍNIO DO GRAFO

- **matriz de adjacência**
  - É uma matriz NxN na qual :
  - $\begin{cases} A_{ij} = 1, \text{se existe a aresta entre o nó i e j} \\ A_{ij} = 0, \text{se não existe a aresta entre o nó i e j} \end{cases}$

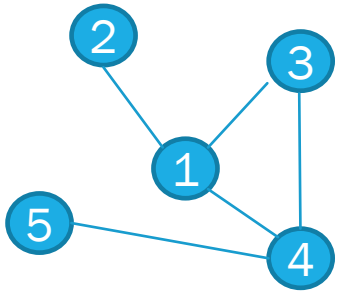


$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

REDES E GRAFOS - DOMÍNIO DO GRAFO

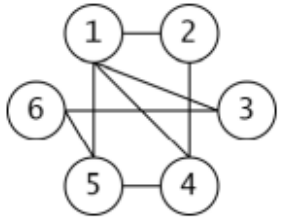
- matriz Laplaciana (L)
- L =D -A

$$\mathbf{L} = \mathbf{D} - \mathbf{A} = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} - \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 3 & -1 & -1 & -1 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 2 & 1 & 0 \\ -1 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix}$$



REDES E GRAFOS -DOMÍNIO DO GRAFO

- **Recursos**
  - Na IA, as **características** são simplesmente **atributo**s quantificáveis que caracterizam um fenômeno que esta em estudo.
  - No domínio do grafo, os recursos podem ser usados para **caracterizar vértices ou arestas**.
  - Estendendo nosso exemplo de rede social, podemos ter **recursos** para cada pessoa **(vértice) quantifica a idade, popularidade e uso de mídia social da pessoa .** Da mesma forma, podemos ter um **recurso** para cada **relacionamento** que quantifica quão bem duas pessoas se conhecem, ou o tipo de **relacionamento que tem (familiar, colega, etc.).**
  - recursos pode ser para : **vértice ou aresta**, então eles são representados por vetores de recursos numéricos referido como $\mathbf{V_i^F}$ e $\mathbf{e_{i,j}^F}$ e respectivamente.



vidx = 1
feat_name = Anton
feat_country = Australia
feat_age = 22

vidx = 2
feat_name = Bram
feat_country = Belgium
feat_age = 31

vidx = 6
feat_name = Fjalar
feat_country = Finland
feat_age = 9

vidx = 3
feat_name = Carl
feat_country = Canada
feat_age = 66

vidx = 5
feat_name = Edjo
feat_country = Egypt
feat_age = 26

vidx = 4
feat_name = Dagny
feat_country = Denmark
feat_age = 41

os **recursos** de cada vértice V_i^F

REDES E GRAFOS -DOMÍNIO DO GRAFO

- **Vizinhanças**
- Definido como **subgraficos dentro de um grafo**, as vizinhancas representam diferentes grupos de vertices e arestas.
- As vizinhanças podem **crescer a partir de um único vértice de forma iterativa** considerando os **vértices anexados** (via arestas) à vizinhança atual.
- **A vizinhança cresce de** $\mathbf{v_i}$ depois que uma iteração é referenciada neste texto como a vizinhança direta, ou pelo conjunto de índices vizinhos ne[v_i].
- Uma vizinhança pode ser definida sujeita a **determinado vértice** e critérios de recurso de borda.

REDES E GRAFOS -DOMÍNIO DO GRAFO

- **Estados (mudam no tempo, iteratividade)**
- **Codifique** as informações representadas em uma determinada vizinhança em **torno de um vértice** (incluindo o vértice da vizinhança, recursos de borda e estados).
- Estados podem ser considerados **ocultos vetores de recursos**.
- Na pratica os **estados** são criados aplicando i**terativamente uma função de extração de recursos aos estados da iteração anterior**, com os estados da iteração posteriores incluindo todas as informações necessárias para realizar
  - a classificação,
  - regressão
  - ou algum outro calculo de saída em um determinado vértice.

REDES E GRAFOS -DOMÍNIO DO GRAFO

- **Embeddigs**
  - Definido simplesmente como **representações compactadas.**
  - Se **reduzirmos o grande vetores recurso** associados **a vértices e arestas em embeddings de baixa dimensão**, torna-se possível para classifica-los com modelos de ordem inferior(ou seja, se pudermos tornar um conjunto de dados linearmente separável).
  - Embeddings podem ser **criados (ou aprendidos**) **para vértices, arestas, vizinhanças ou gráfos.**
  - Os embeddings tambem são chamados de **representações, codificações, vetores latentes ou de alto nível vetores de recursos dependendo do contexto**.

ARQUITECTURAS

- a) Recorrentes (RGNNs)
- b) Convolucionais (CGNNs)
- c) Autoencoder (GAEs)).

GRAFOS RECORRENTES DE REDES NEURAIS (RGNN)

- **O passe para frente (forward)**
  - **Transição**
  - **Saída.**

RGNN

- Em uma **Rede Neural padrão**, camadas sucessivas trabalham para **extrair recursos** de uma entrada.
- No caso da **classificação** de imagem simples, depois de ser processado por **camadas sequenciais**, o resultado dos recursos podem, então, ser fornecidos a uma **camada softmax ou neurônio´único** com o objetivo **de classificação, regressão**, etc.
- Da mesma forma, os primeiros trabalhos do **GNN** visavam **extrair** representações **de recursos de alto nível a partir de grafos** usando operações sucessivas de **extração de características** e, em seguida, funções de saída
- A **aplicação recursiva** de um **extrator de recursos**, ou rede de codificação,´e o que da ao **RGNN** seu nome.

RGNN

- **O passe para frente**
- O passe para frente RGNN ocorre em duas etapas principais:
- **Transição**
  - A primeira etapa se concentra **na computação de alto nível vetores** de recursos ocultos para cada vértice no gráfico de entrada.
  - Este calculo é realizado por um **função de transição**
- **Saída**
  - A segunda etapa esta relacionada ao **processamento dos vetores de recursos ocultos em resultados úteis;**
  - usando uma **função de saída**

RGNN

- **Transição**
- O processo de transição considera **a vizinhança de cada vértice v** em um grafo e produz uma representação oculta para cada um desses vizinhos
- Diferentes vértices no grafo pode ter **diferentes números de vizinhos**, o processo de transição emprega um **somatório sobre os vizinhos**, produzindo assim um vetor de tamanho consistente para cada vizinho.
- Esta representação´e muitas vezes referida como o **estado do vértice**
-
  - $\mathbf{v}_i^{\mathbf{F}}$ as características do vértice v , em torno do qual a vizinhança esta centrado
  - $e_{\mathbf{i},j}^{\mathbf{F}}$ as características das arestas que unem v_{i} aos seus vértices vizinhos v_{j} .
  - $\mathbf{v}_j^{\mathbf{F}}$ as características dos Vizinhos de v_{i}
  - $h_j^{\mathrm{k}-1}$ o estado anterior de Vizinhos de v_{i} .

RGNN

- **Transição**
- Formalmente, a função de transição f é usada no cálculo recursivo do k^{th} estado do vértice
- $h_i^k = \sum_{j \in ne[v_i]} = f\left(v_i^F, \epsilon_{ij}^F, v_{j,}^F, h_j^{k-1}\right)$
- onde todos $\boldsymbol{h_i^o}$ são definidos na inicialização
- Ele aceita **quatro vetores de recursos** que todos têm um comprimento definido, **independentemente** de qual **vértice** do gráfico está sendo considerado, independentemente da **iteração.**
- Isso significa que a **função de transição pode ser aplicada recursivamente, até um estado estável é alcançado para todos os vértices** no gráfico de entrada.
- Se **f** é um **mapa de contração**, o teorema **ponto fixo de Banach** garante que os valores **de** $\boldsymbol{h_i^k}$ irá convergir para **valores estáveis** exponencialmente rápido, independentemente da inicialização de h_i^0

RGNN
TEOREMA BANACH

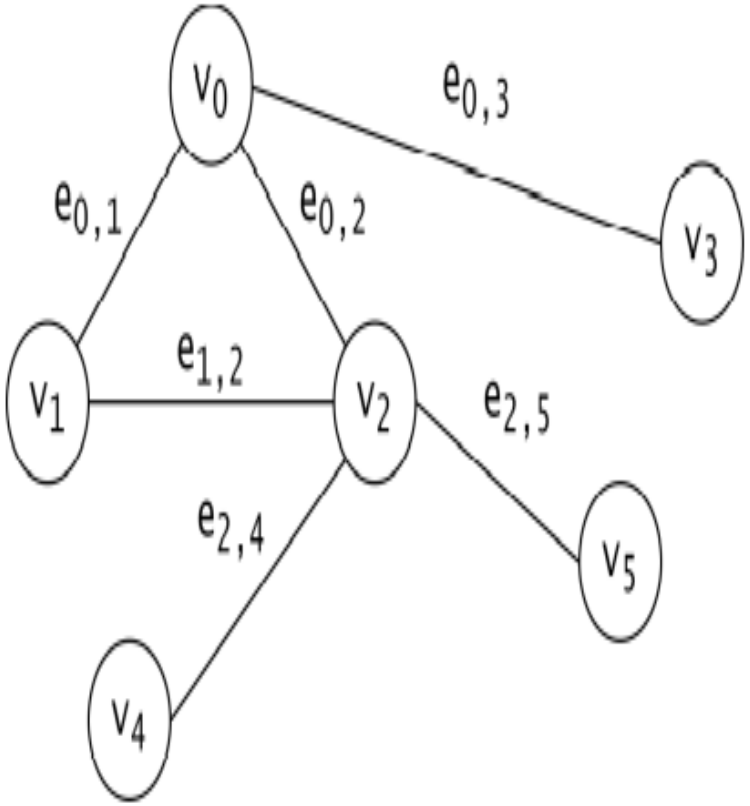
Contração

- Seja (M,d) um espaço métrico
- Uma função f: M →M é chamada de Contração sobre M se existe um numero real positivo k<1 tal que :
- d (f(x), f(y)) ≤ K d(x,y) , $\forall x, y \in M$

Teorema Ponto Fixo Banach

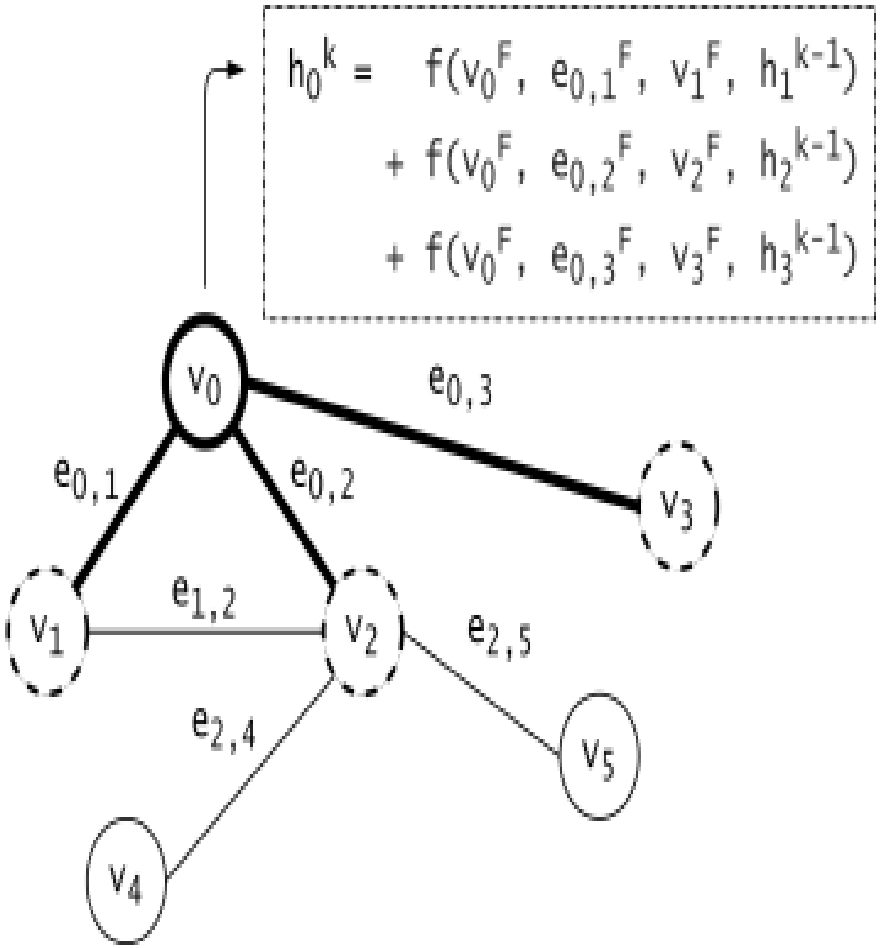
- Considere (M,d) um espaço métrico completo e uma contração f: M →M
- Então f possui um único ponto fixo

RGNN



- Cada vértice e aresta tem alguns recursos associados.
- Além disso, cada vértice vi é inicializado com algum h_i^0

RGNN



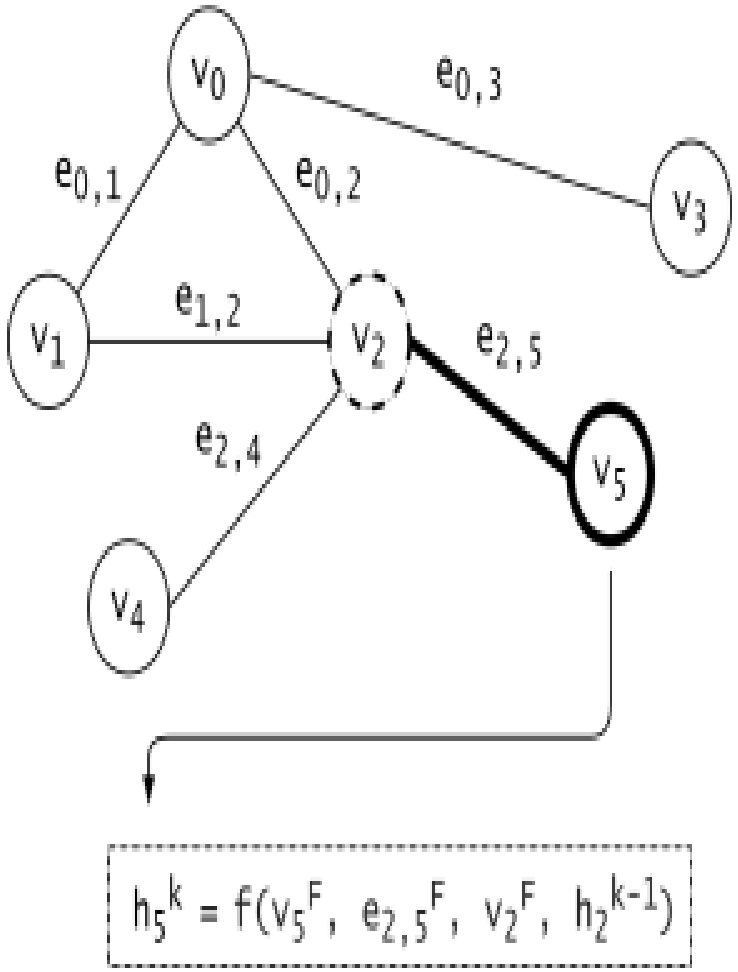
Calculamos o k^{th} estado para cada vizinho no gráfico.

Qualquer cálculo do k^{th} estado é dependente apenas dos recursos do gráfico de entrada e do $(k-1)^{th}$ estados.

Começamos calculando $\mathbf{h_0^k}$ para a vizinhança centrada em torno de v_0

v_0 **e suas conexões diretas** (reforçadas) para seus vértices vizinhos (pontilhados) são ilustrados aqui

RGNN



O mesmo cálculo é realizado em cada **vizinhança** no gráfo, até que a vizinhança do vértice final seja alcançada.

Feito isso, a representação da $\boldsymbol{k^{th}}$ **camada do gráfico** foi calculada.

o o processo é então repetido para o $\mathbf{(k+1)^{th}}$ **camada ,a** $\mathbf{(k+2)^{th}}$ **camada, e assim por diante, normalmente até que a convergência seja alcançado**.

RGNN

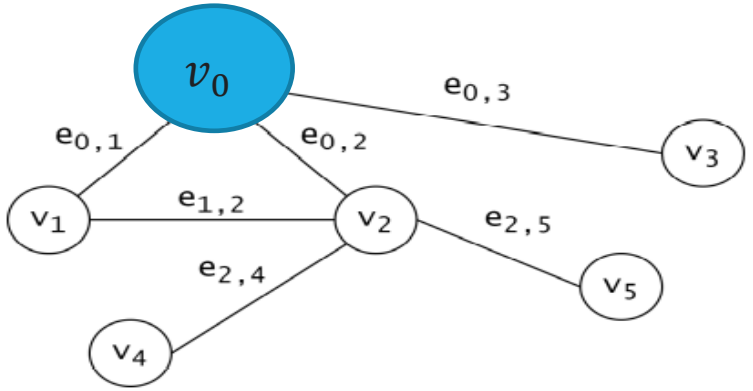
- O objetivo de **aplicações repetidas da função de transição** é, portanto, criar **embeddings** que podem ser usados para tarefas

RGNN

- **Saída**.
- A **função de saída** é responsável por obter o **estado oculto convergente de um grafo G (V, E)** e criando uma saída significativa.
- **A aplicação repetida da função de transição f às características de G (V, E)** garante que cada estado final h_i^{kmax} é simplesmente uma codificação de alguma região do gráfico.
- **O tamanho desta região é dependente da condição de parada** (convergência, etapas máximas de tempo, etc.), mas muitas vezes A **'passagem de mensagens'** garante que o estado final de cada vértice oculto tenha 'visto' o gráfico inteiro.
- Esses **codificações ricas normalmente têm dimensionalidade mais baixa do que os recursos de entrada do gráfico** e podem ser alimentados para camadas totalmente conectadas para fins de classificação, regressão e assim por diante

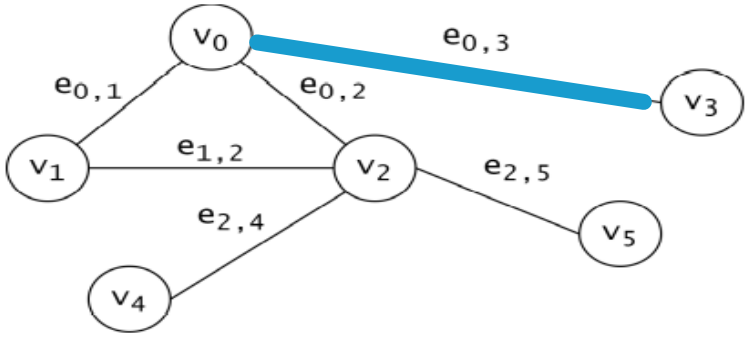
RGNN

- **Saída**.
- A questão agora é o que definimos como uma "**saída significativa**"? Isso depende muito do estrutura de tarefas:
- **• Estruturas de nível de vértice.**
  - Um valor de saída único é necessário para cada vértice, e a função de saída g leva as características de um vértice e o estado final como entradas:
  - $O_i = g\left(v_i^F\ , h_i^{kmax}\right)$
  - a função de saída g pode ser aplicada a todos os vértices do gráfico.



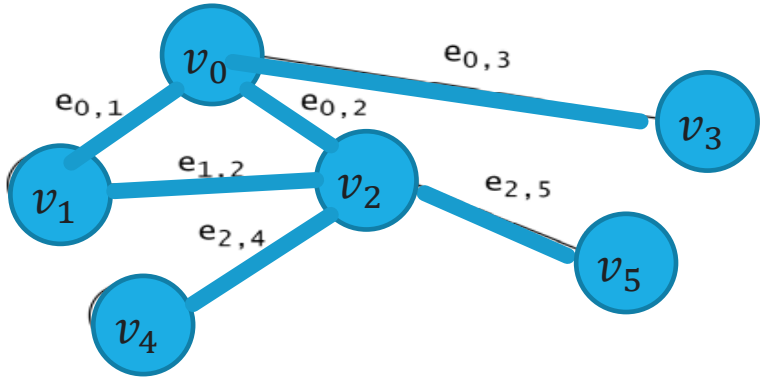
RGNN

- **Saída**.
- **Estruturas de nível de borda.**
  - Um valor de saída exclusivo é necessário para cada borda, e a saída função g **leva os recursos da aresta**, e os recursos do vértice e o estado final dos vértices que definem a aresta:
  - $O_{ij} = g\left(e_{i,j}^{F}, v_i^{F}, h_i^{kmax}, v_j^{F}, h_j^{kmax}\right)$
  - Da mesma forma, g pode ser aplicado a cada borda no gráfico



RGNN

- **Saída.**
- **Estruturas em nível de gráfico.**
  - Uma saída única é necessária em todo o grafo (ou seja, classificação do gráfico). A função de saída g irá considerar todas as informações relevantes calculadas até agora.
  - Esse irá incluir o estado final em torno de cada vértice e pode, opcionalmente, incluir o vértice inicial e recursos de borda:
  - $O \quad = g\left(h_o^{kmax}, h_1^{kmax} \ldots\ldots, h_{N-1}^{kmax}, v_0^F, v_1^F \ldots, v_{N-1}^F, e_{0,0}^F, e_{0,1}^F, e_{0,N-1,\ldots\ldots}^F, e_{N-1,N-1}^F\right)$



GRAFOS CONVOLUCIONAIS DE REDES NEURAIS
CGNNS

- CONVOLUÇAO ESPACIAL
- PROCESAMENTO DE SINAL DE GRAFICO
  - A matemática
- CONVOULÇÃO ESPECTRAL

CGNNS

- Os CGNNs foram amplamente inspirados pelo sucesso de **CNNs** em tarefas baseadas em **imagens em visão computacional - aplicando operações convolucionais,**
- CGNNs são capaz de realizar com eficiência o **reconhecimento de padrões** dentro do domínio do grafo.
- Apesar de processo de **transição** durante o passe para frente de um CGNN é alterado para incluir a **convolucao**, o processo principal de alto nível permanece o mesmo que nos RGNNs

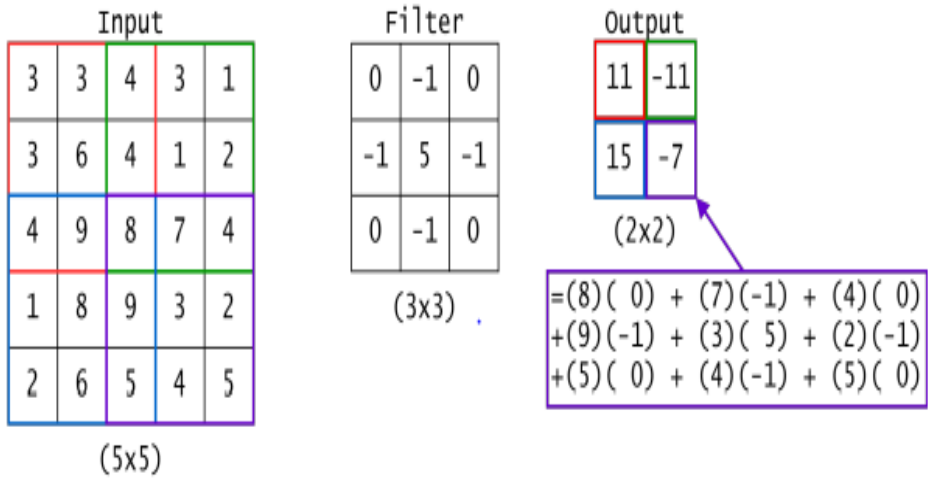
CGNNS

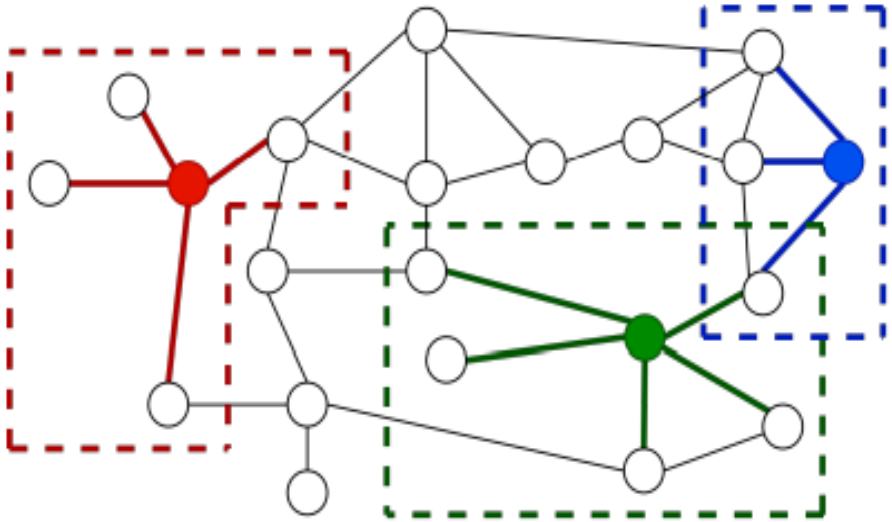
- **convolução espacial**
- Quando pensamos em convolucão, muitas vezes pensamos na operação de **convolucão** usada nas **CNNs**
- Isso esta de acordo com uma definição geral de convolucão: “Uma saída derivada de **duas entradas fornecidas por integração (ou soma)**, que expressa como a forma de uma e modificada pela outra.
- Como conciliamos **convolucões em entradas não estruturados como gráficos?**

CGNNS

- **convolução espacial**
- as **operações convolucionais** podem ser aplicadas a
  - **funções contínuas** (por exemplo, gravações de áudio e outros sinais),
- Durante a convolucao, uma entrada´e normalmente interpretada como **um filtro (ou kernel)** sendo aplicado `a outra entrada
- **Filtros ou kernels** específicos podem ser utilizados para realizar tarefas específicas:
  - no caso de gravações de áudio, filtros podem ser usados para **filtrar sinais de baixa frequência e,**
  - no caso de imagens, certos filtros podem ser usados para aumentar o contraste**, aumentar a nitidez ou desfocar as imagens.**
  - Relacionado as CNNs, os filtros convolucionais aprendidos realizam uma espécie de **extração de recursos** aprendidos.

CGNNS



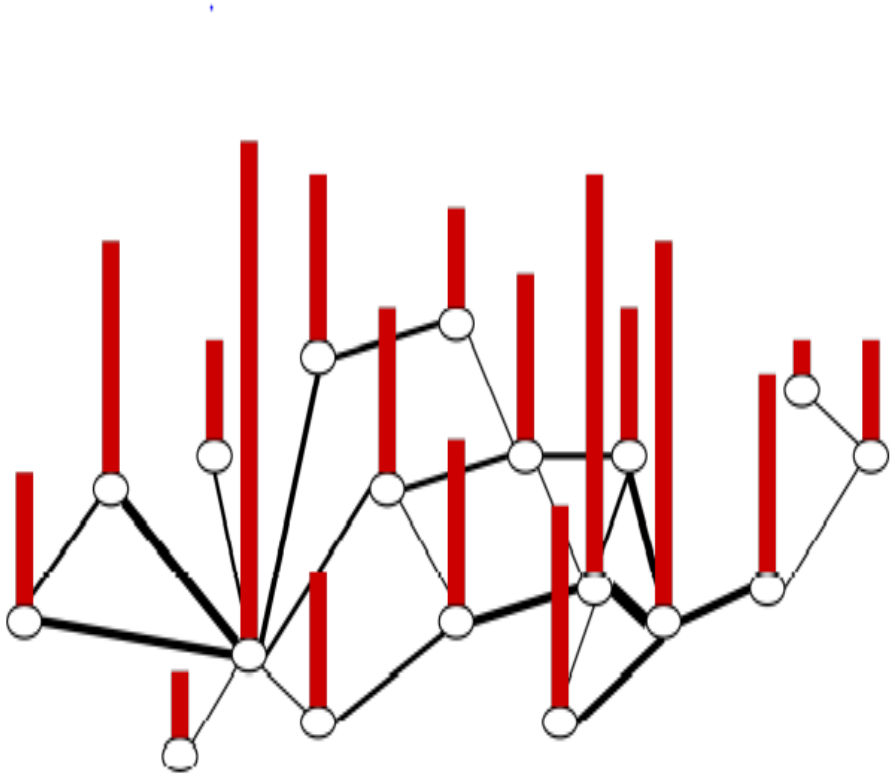


- **Três vizinhanças** em um determinado gráfico (**designadas por caixas pontilhadas**), com cada uma definida por um **vértice central (designado por um círculo de cor correspondente).**
- Na convolução espacial, **uma vizinhança é selecionados**, os valores dos **recursos dos** vértices incluídos são agregados e, em seguida, este valor agregado é usado para atualizar a incorporação de vértice **ao vértice central**.
- Este processo é **repetido** para todos as **vizinhanças** do grafico
- Esses **embeddings** podem então ser usados como "valores" de vértice na próxima camada de convolução espacial
- A principal **diferença e que os RGNNs continuam a iteração ate um ponto estável** e usam um função de transição ao faze-lo. Já, os **CGNNs iteram para um numero fixo de camadas**

CGNNS

- **Processamento de sinal de grafo**
- Um **sinal** é definido em um grafo se **mapeia cada vértice em um determinado grafo para um valor real**
- Formalmente, uma função ́e considerada um sinal se f: V→R, para todo V pertence G.
- Para simplificar, um **sinal simplesmente como um tipo de característica que cada vértice possui** – isso poderia ser facilmente apresentado como um vetor cujo elemento representa o valor do sinal em v

PROCESSAMENTO DE SINAL DE GRÁFICO



- Grafo de G (V, E), com os correspondentes valores de sinal de gráfo como barras vermelhas subindo perpendicularmente ao referido plano.
- Os **vértices V representam cidades,** e as **arestas E representam se duas cidades têm uma trajetória de vôo** entre elas.
- Este gráfico também tem uma matriz de peso W significa o número médio de voos por semana ao longo de uma dada trajetória de voo que são realmente realizados.
- **O gráfico de sinal representa a quantidade de pessoas com vírus contagioso em cada cidade, já que são 18 vértices, o vetor de sinal gráfo tem tamanho 18**

CGNNS

- **Processamento de sinal de gráfico**
  - **A matemática (A modelagem)**
  - Muitas das **ferramentas** simples e fundamentais para a compreensão e analisar um sinal não esta bem definido para grafos que são representados no **espaço do vértice.**
    - Para superar os desafios no processamento sinais em grafos, a **inclusão de analise espectral - semelhante `a usada na analise de sinais discretos -** foi fundamental no surgimento da **computação localizada** de informações de grafo.
- Para processar sinais de grafo, desta forma, somos obrigados a transformar **o grafo do espaço do vértice para espaço de frequência**

GNNS

- **Processamento sinal de gráfico**
- **As técnicas de Fourier** são necessárias para realizar **essa transformação de espaço de vértice para espaços de frequência**
- Classicamente, a **transformada de Fourier**́e a expansão de uma determinada função em termos das **autofunções do operador de Laplace**
- No domínio do grafo, usamos o grafo da transformada Fourier, que e a expansão de um determinado sinal em termos das autofunções do grafo Laplaciano
- E neste **espaço de frequência** que será realizada a **convolução.**

GNNS

- **Processamento de sinal de grafo**
- **A matemática**
- O **grafo Laplaciano L´e uma matriz simétrica** real, e como tal, podemos realizar **autocomposição sobre ela** e, portanto, extrair seu conjunto de autovalores e autovetores
- Existem muitas maneiras de calcular o autos sistema de uma matriz, como por meio de um valor singular **decomposição (SVD).**

$$\mathbf{L} = \mathbf{D} - \mathbf{A} = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} - \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 3 & -1 & -1 & -1 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 2 & 1 & 0 \\ -1 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix}$$

GNNS

- **Processamento de sinal de grafo**
- **A matemática**
- Pela propriedade fundamental dos autovetores, **a matriz Laplaciana** pode ser fatorada como três matrizes tais que
- $L = \mathbf{U}\,\Lambda\,\mathbf{U}^{-1}$
- Aqui, U´e uma matriz com autovetores como colunas, ordenados por autovalores Λ e´ e uma matriz diagonal com os ditos autovalores em sua diagonal principal.
- **os autovetores de L são exatamente as exponenciais da transformada discreta de Fourier**.

GNNS

- **Processamento de sinal de grafo**
- **A matemática**
- Uma vez que esta decomposição automática foi realizada e o sistema automático foi calculado, podemos expressar livremente um determinado sinal de grafo discreto em termos dos autovetores do grafo Laplaciano, produzindo assim o grafo transformada de Fourier .
- Na primeira equação :**Do espaço do vértice para o espaço de frequencia**
- Na segunda equação : **Do espaço de frequencia para espaço do vértice**

- $\hat{f}(k) = \sum_{k=1}^{N} f(i)\, u_k^*(i) \qquad ou \quad \hat{f} = U^{-1} f$

- $f(i) = \sum_{k=1}^{N} \hat{f}(k) \quad U_k(i) \qquad ou \quad f = \mathrm{U}\, \hat{f}$

GNNS (NÃO)

- **Processamento de sinal de grafo**
- **A matemática**
- Novamente, os autovetores aparecem como colunas na matriz U, que´e gerada no calculo do auto-sistema do grafico Matriz Laplaciana L
- De acordo com **o teorema da convolucao, a multiplicacao no domınio da frequencia corresponde à convolucao no domınio do tempo .** Esta propriedade tambem´e valida para grafos: **multiplicacao no domınio do vertice´e equivalente `a convolucao no domınio da frequencia.**
- Esta e uma propriedade importante, pois não podemos definir o operador convolucional clássico diretamente no domínio do vértice

GNNS

- **Processamento de sinal de grafo**
- A matemática
- No entanto, podemos definir a convolucao no domínio da frequência de acordo com

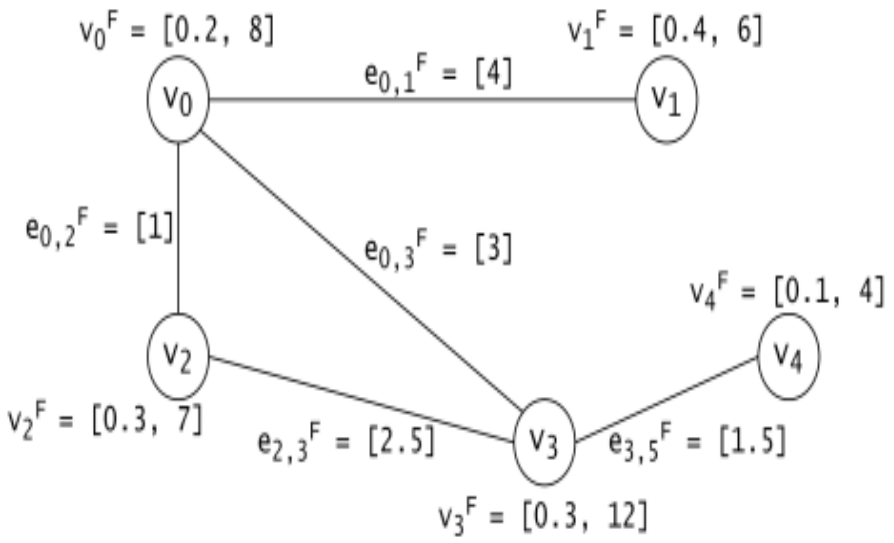
$$(f * g)(i) = \sum_{k=1}^{N} \hat{f}(k)\, \hat{g}(k)\, u_k(i)$$

- Na pratica, a **funcao g** que aparece e tipicamente um filtro aprendido
- Com uma compreensão de como a convolucão pode ser realizada em grafos no domínio espectral, podemos agora mergulhe na mecânica da convolucão espectral

GNNS

- **Convolucao Espectral**
- Antes de realizarmos uma´única camada de convolucão espectral, devemos determinar o grafo Auto-sistema de Laplaciano pois isso nos permite realizar **transformacões entre espaço vertice e espaco de frequencia**.
- $$(\mathbf{U}^{-1}\mathbf{f})$$
- Com o sistema autonomo , podemos agora convolver um único sinal de grafo com um filtro aprendido multiplicando-o por **ɵ**
- $$\boldsymbol{\theta}\,(\mathbf{U}^{-1}\mathbf{f})$$
- Para repetir este processo recursivamente, precisamos retornar o resultado convolvido no espaço do vertice , entao a operacão se torna
- $$\mathbf{U}(\boldsymbol{\theta}(U^{-1}\mathbf{f})).$$

GNNS



- Um exemplo concreto de como o gráfico Laplaciano é formado para um gráfico, e como podemos representar pesos de borda e sinais de gráfico.
- Neste exemplo, existem dois sinais gráficos;
  - o primeiro sinal gráfico $f1$ = [0,2; 0,4; 0,3; 0,3; 0,1],
  - equivalentemente, é o conjunto ordenado dos
  - primeiros elementos de cada vetor de características dos vértices V_i^F .
  - Da mesma forma, o segundo sinal de gráfico é
- $f2$ = [8, 6, 7, 12, 4].

$$\mathbf{L} = \mathbf{D} - \mathbf{A} = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} - \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 3 & -1 & -1 & -1 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 2 & 1 & 0 \\ -1 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix} = \mathbf{U}\Lambda\mathbf{U}^T$$

- O **gráfico Laplaciano** neste exemplo é simplesmente formado por L = D - A, e é real e simétrico,
- portanto, pode ser decomposto para determinar sua auto-sistema

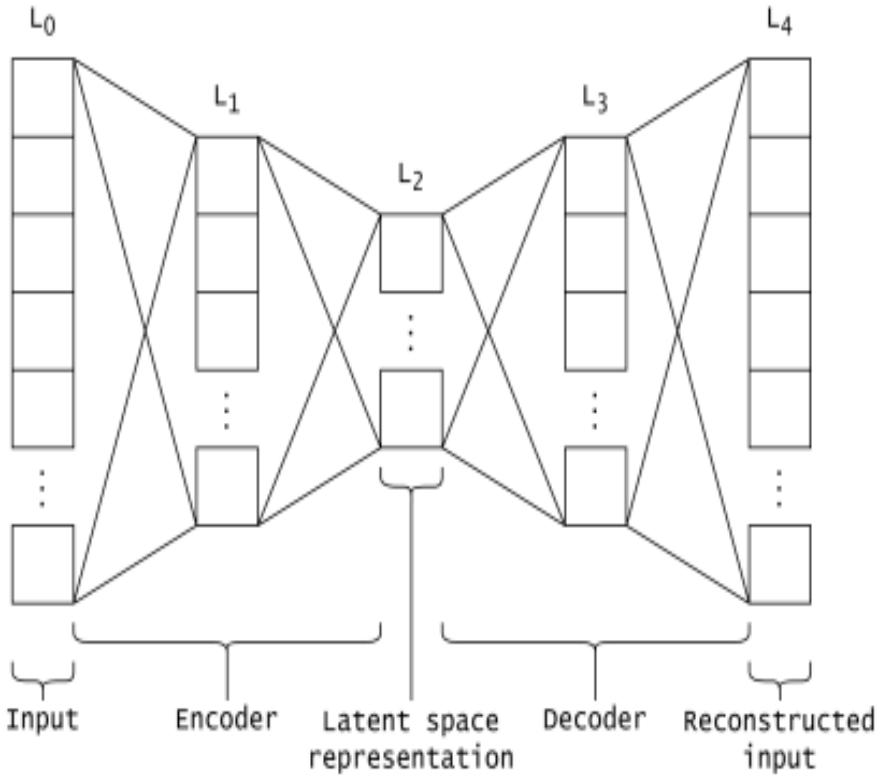
AUTOENCODERS (GAES)

- **Autoencoders Tradicionais (Aes)**
- **Autoencoders Variacionais**
- **Aplicação a grafos**
  - **A variável de destino**
  - **A função de perda**
  - **Codificadores.**
  - **Decodificadores.**

GAES

- os autoencoders (AEs) fazem uma **transição suave para o domínio do grafo** porque eles vem ”**preembalados”** com o conceito de **embeddings.**
- São uma técnica comum de aprendizagem não supervisionada, com aplicações bem-sucedidas em uma variedade de tarefas incluindo
  - remoção de ruído de imagem
  - redução de dimensionalidade
  - motores de recomendação
- Eles funcionam em duas etapas,
  - **primeiro codificando** os dados de entrada em um **espaço latente**, reduzindo assim sua **dimensionalidade** e,
  - em seguida, **decodificar** essa **representação compactada** para **reconstruir o original dados de entrada**

GAES



- A arquitetura para um AE padrão muito simples.
- Os AEs tomam uma entrada original, alteram a dimensionalidade através de múltiplas camadas totalmente conectadas e, assim, converter a referida entrada em um **vetor espacial latente**
- (este processo forma o codificador).
- A partir daí, o AE tenta **reconstruir** a entrada original (este processo forma o **decodificador**).
- Ao **minimizar a perda de reconstrução**, representações de espaço latente eficientes podem ser aprendido.
-

GAES

- **Autoencoders Tradicionais**
- AE é treinado para minimizar a perda de reconstrução, que é calculada usando apenas os dados de entrada e, portanto, podem ser treinados de maneira não supervisionada.
  - Em sua forma mais simples, tal uma perda´e medida como o erro quadrático médio entre a instancia de entrada e o reconstruído entrada ˆ,de acordo com a Equação

$$\text{Loss}_{AE} = \|X - \hat{X}\|^2$$

- A representação **espaço latente** de uma rede **codificador-decodificador** podem ser referidas como a incorporação das entradas, e eles **são análogos aos embeddings** de vértices
- Assim , uma **representação latente**´e simplesmente um **vetor de recurso aprendido**

GAES

- **Autoencoders Variacionais**
- Em vez de representar entradas com pontos´únicos no espaço latente, autoencoders variacionais **(VAEs) aprendem a codificar entradas como distribuições de probabilidade no espaço latente.**
- Ao contrario de AEs - onde a perda´e simples, o erro quadrático médio entre a entrada e o reconstruído entrada - uma perda VAEs tem dois termos.

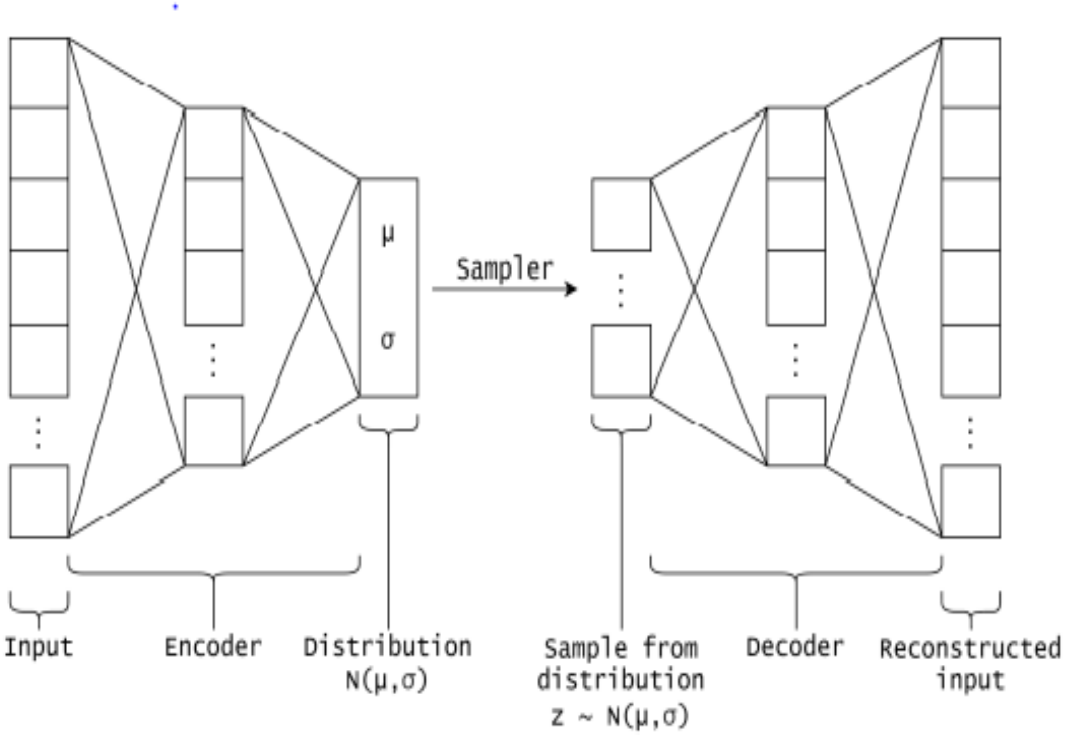
$$\text{Loss}_{VAE} = \|X - \hat{X}\|^2 + \text{KL}(N(\mu, \sigma), N(0, 1))$$

GAES

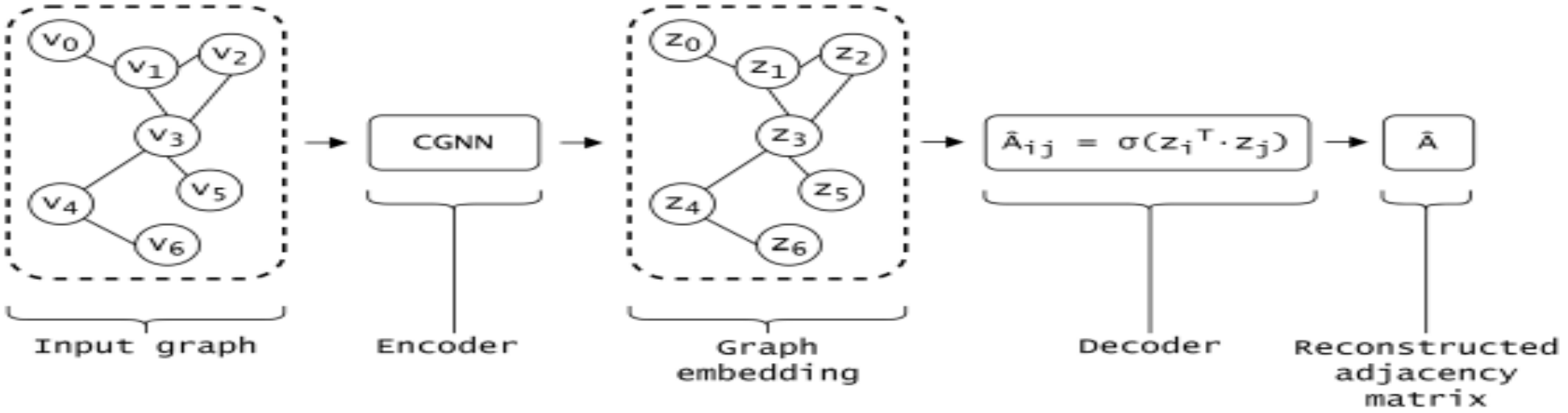
- **Autoencoders Variacionais**
- O primeiro termo´e o termo de reconstrucao, este termo ainda aparece como seu obviamente ainda´e o nosso desejo de reconstruir com precisao a entrada fornecida.
- O segundo termo **regulariza as distribuicoes do espaco latente** garantindo que eles nao divergem significativamente das distribuicao normais padrao (denotadas como (0, 1)), usando divergencia de Kulback-Leibler

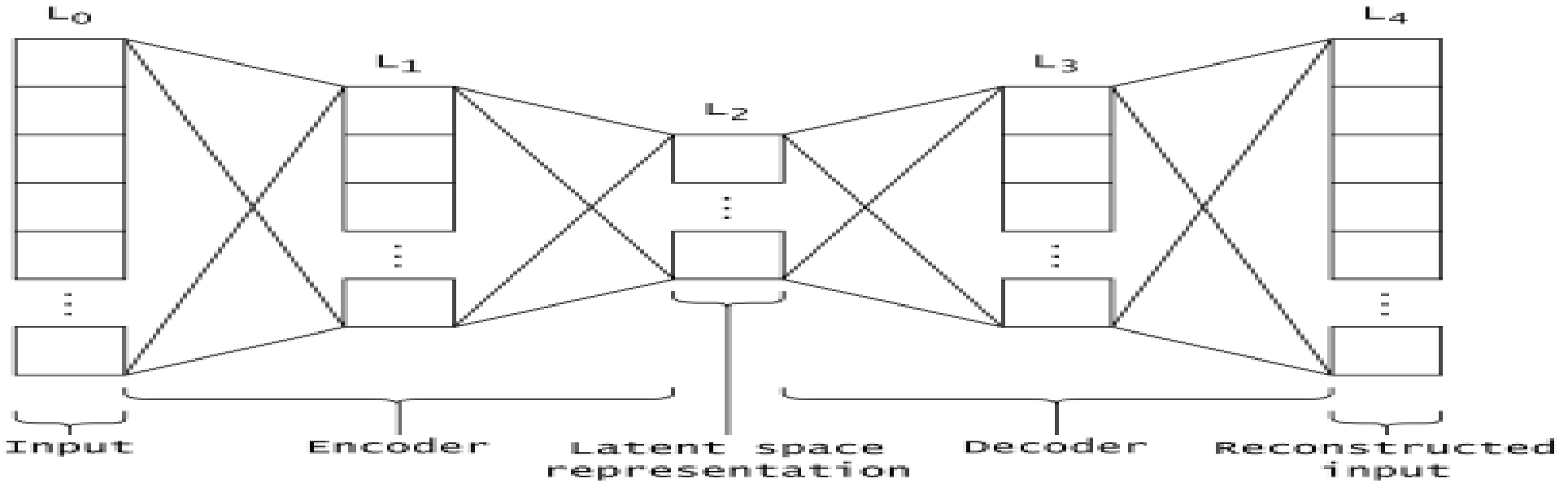
$$\text{Loss}_{VAE} = \|X - \hat{X}\|^2 + \text{KL}(N(\mu, \sigma), N(0, 1))$$

GAES



- A porção do codificador deste VAE leva uma entrada, e codifica-o por meio de várias camadas totalmente conectadas em uma **distribuição no espaço latente**.
- Esta distribuição é então amostrado para produzir um vetor de espaço latente
- O decodificador parte deste VAE pega esse vetor espacial latente e o decodifica em uma entrada reconstruída como em um AE.


v0
v1
v2
v3
v4
v5
v6
CGNN
z0
z1
z2
z3
z4
z5
z6
Âij = σ(ziT·zj)
Â
Input graph
Encoder
Graph embedding
Decoder
Reconstructed adjacency matrix


L0
L1
L2
L3
L4
Input
Encoder
Latent space representation
Decoder
Reconstructed input

GAES

- **Aplicação a grafos**
- Os AEs se traduzem bem no domínio do grafo porque são projetados em torno do conceito de **aprendizagem embeddings de entrada**.
  - **A variável de destino.**
  - **Em AEs**, a **entrada alvo** tem uma **estrutura bem definida** (por exemplo, **um vetor** de comprimento conhecido, ou uma **imagem de tamanho conhecido)**, e, portanto, a qualidade de sua **reconstrução e facilmente medido usando o erro quadratico médio**. (AE)
  - **Para medir** a adequação **de uma reconstrução em um GAE, é necessário produzir uma saída significativamente estruturada que nos permite identificar semelhanças entre a reconstrução e o grafo de entrada**

GAES

- **Aplicação a gráficos**
  - **A função de perda**.
  - Em VGAEs, a perda e semelhante a perda VAE padrao; Divergência KL´e ainda usado para medir a similaridade entre as distribuicoes preditas e verdadeiraś e usada para **medir a diferenca entre a matriz de adjacencia prevista e a verdadeira matriz de adjacencia**
  - Alternativa métodos incluem o uso de distancia no espaço Wassertein (issó e conhecido como a métrica Wassertein, e quantifica o custo associado a tornar uma distribuição de probabilidade igual a outra.
  -

GAES

- **Aplicação a gráficos**
  - **Codificadores.**
  - **Porque estes métodos operam apenas na matriz de adjacência, informações sobre o grafo inteiro e o bairros locais foram perdidos.**
  - Trabalhos mais recentes atenuam isso usando um codificador que agrega informações da vizinhança local de um vértice para aprender representações vetoriais latentes
  - Apesar disso, os GAEs típicos usam codificadores mais complexos - principalmente CGNNs - a fim de captura relações não lineares nos dados de entrada e bairros locais maiores

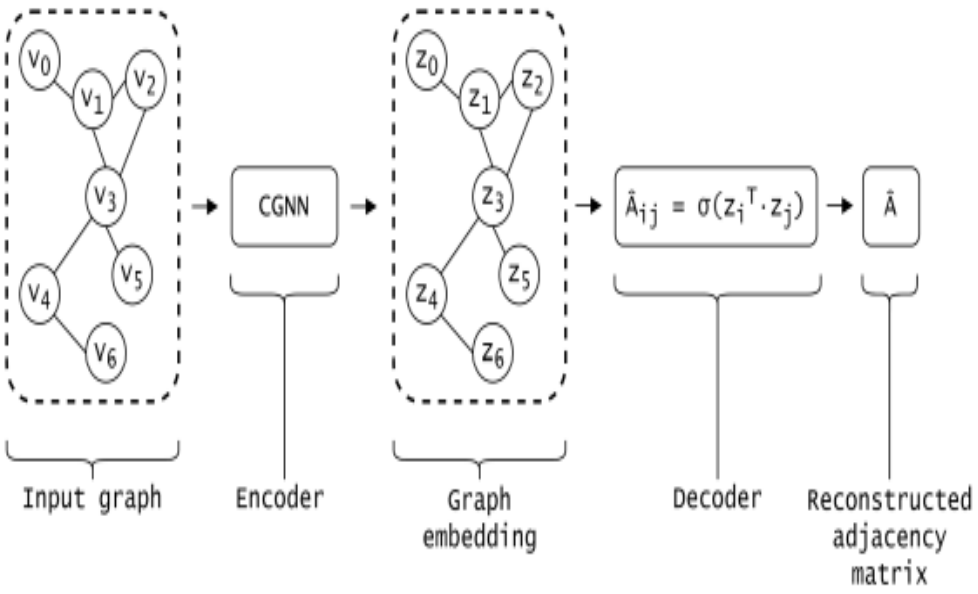
GAES

- **Aplicação a gráficos**
  - **Decodificadores.**
  - GNNs padrão, independentemente de suas funções de agregação normalmente gera uma **matriz contendo a incorporação de cada vértice.**
  - **GAEs aplicam um decodificador a esta representação codificado de cada vértice para reconstruir o grafo original. Por esse motivo, o decodificador dos GAEs é comumente referido como modelo generativo.**
  - Em muitas variações de GAEs , **o decodificador é um produto interno simples das variáveis latentes.** Como um **produto interno de dois vetores** equivale ao calculo de sua similaridade **de cosseno, quanto maior o resultado deste produto, mais provavelmente esses vértices estão conectados.**
  - Com esta previsão de similaridade de vértice, a adjacência do grafo matriz pode ser prevista apenas a partir dos embeddings de todos os vértices no grafo

GAES

- **Aplicação a grafos**
  - **Decodificadores.**
  - Calculando todos **os pares aproximações de distancias,** produzimos uma **aproximação da matriz de adjacência .**
  - Tal como acontece com AEs e VAEs, GAEs podem, portanto, ser treinados de uma **forma não supervisionada**, uma vez que um valor de perda pode ser calculado usando apenas instancias não rotuladas

$$\hat{A}_{ij} = \sigma\left(z_i^T z_j\right)$$



- Em vez de ingerir uma matriz ou vetor, o codificador ingere dados estruturados de gráfico G.
- Neste caso, um CGNN cumpre a função do codificador por criar recursos de vértice discriminativos / embeddings de vértice para cada vértice.
- **Isso não altera a estrutura de G. Tal como acontece com AEs e VAEs, o papel do decodificador é pegar esses embeddings e criar uma estrutura de dados que pode ser comparado à entrada.**
- No caso de GAEs, a matriz de adjacência de G é reconstruída.
- **Esta matriz de adjacência reconstruída $A\hat{}$ é comparada com a matriz de adjacência original A para criar um termo de perda, que é então usado para retropropagar erros em todo o GAE e permitir o treinamento**

RNNS

- F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. 2009. The Graph Neural Network Model. IEEE Transactions on Neural Networks 20, 1 (Jan 2009), 61–80. https://doi.org/10.1109/TNN.2008.2005605

CGNNS

- David I. Shuman, Sunil K. Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. 2012. Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Data Domains.
- David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. 2013. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. IEEE signal processing magazine 30, 3 (2013), 83–98.
- Tomas Simon, Hanbyul Joo, Iain A. Matthews, and Yaser Sheikh. 2017. Hand Keypoint Detection in Single Images using Multiview Bootstrapping
- Ljubisa Stankovic, Danilo P Mandic, Milos Dakovic, Ilia Kisil, Ervin Sejdic, and Anthony G Constantinides. 2019. Understanding the basis of graph signal processing via an intuitive example-driven approach [lecture notes]. IEEE Signal Processing Magazine

AUTOENCODERS

- Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and Composing Robust Features with Denoising Autoencoders. In Proceedings of the 25th International Conference on Machine Learning (ICML '08). Association for Computing Machinery, New York, NY, USA, 1096–1103.
- G. E. Hinton and R. R. Salakhutdinov. 2006. Reducing the Dimensionality of Data with Neural Networks. Science
- Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. AutoRec: Autoencoders Meet Collaborative Filtering. In Proceedings of the 24th International Conference on World Wide Web (WWW '15 Companion). Association for Computing Machinery, New York, NY, USA, 111–112. https://doi.org/10.1145/2740908.2742726

BIBLIOGRAFIA

- ANDERSON, R.; MAY, R. Infectious diseases of humans:dynamics and control. Oxford University. Citado na página 21. BARABASI. Emergence of scaling in random network. science, n. 286, p. 509–512, 1999. Citado 2 vezes nas páginas 21 e 24.
- BERNOULLI, D. Essai d'une nouvelle analyse de la mortalité causée par la petite vérole et de advantages de l'inoculation pour la prévenir. Mémorires de Mathématiques et de Psique, 1760. Citado na página 33.
- ERDOS; RENYI. On randon graphs. Publicationes Mathematicae, n. 6, p. 290–297, 1959. Citado 2 vezes nas páginas 24 e 27.
- EULER. Solutio problemat is ad geometriam situs pertinentis. Academiae Scientiarum Imperialis, n. 8, p. 128–140, 1741. Citado na página 23. FILHO, A.; M, R. Introdução á epidemiologia. 4a ed Guanabara Koogam, 2006. Citado na página 33. GILBERT, . Randon graphs. The Annals of Mathematical Statistics, n. 30, p. 1141–1144, 1959. Citado na página 27. NEWMAN, M. The structure and function of complex networks. Siam Review, n. 45, p. 11–19, 2003. Citado na página 23. PARETO, . Cours d'economie politique. Librairie Droz, p. 299–345, 1964. Citado na página 29. PRICE, D. S. Networks of scientific papers. Sciencie, n. 149, p. 510–515, 1965. Citado na página 38.
- ROSEN. Matematica discreta e suas aplicaçoes. 2009. Citado na página 24.
- SANTORRAS, P.; VESPIGNANI. Epidemic spreading in scalefree. Physical Review Letters, n. 86, p. 3200–3203, 2001. Citado na página 21.
- VERHULST, P. F. Notice sur la loi que la population poursuit dans son accroissement. Correspondance mathématique et physique, n. 10, p. 113–121, 1838. Citado na página 36. W, K.;
- KENDRICK, M. A contribution to the mathematical theory of epidemics. Proccedings of the Royal Society of London Series A Mathematical and Physical Sciences, n. 115, p. 700–721, 1927. Citado na página 33