Modelando Problemas Relacionais com Graph Neural Networks em Pytorch

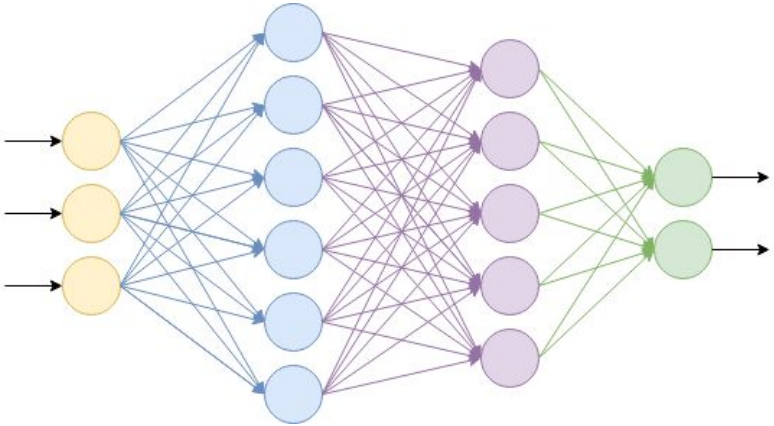
Marcelo Prates
Matheus Gonzaga

Outline

1. Deep Learning Recap
2. Problemas Relacionais
3. Graph Neural Networks
4. CNNs Recap
5. CNN → GNN
6. GNN Definition
7. Code Snippets
8. GNN Applications

Deep Learning Recap

- Machine learning a partir de redes neurais “profundas”
- Redes neurais são *antigas* (**1940s**)
- **Backpropagation** (Rumelhart, Hinton & Williams, 1986)
- **Big Data** (p.ex. ImageNet) / **Big Compute** (p.ex. GPUs)



Deep Learning Recap

- Classificação de imagens - CNNs / ResNets
- Detecção de objetos - R-CNNs, YOLO
- Segmentação de imagens - UNets
- Geração de imagens - VAEs, GANs
- Controle e Planejamento em games (Atari, Go, DOTA2, Starcraft) - RL
- Controle e Planejamento em robótica - RL
- Análise de texto - RNNs / Transformers
- Síntese de texto - RNNs / Transformers
- Predição de séries temporais - RNNs / LSTMs / GRUs / Transformers

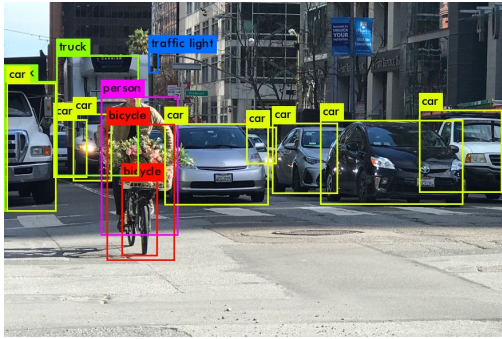
SYSTEM PROMPT (HUMAN-WRITTEN)

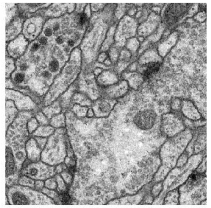
Recycling is good for the world.

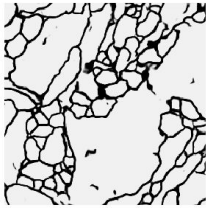
NO! YOU COULD NOT BE MORE WRONG!!

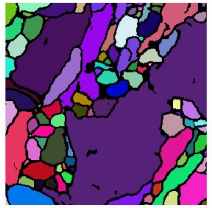
MODEL COMPLETION (MACHINE-WRITTEN, 25 TRIES)

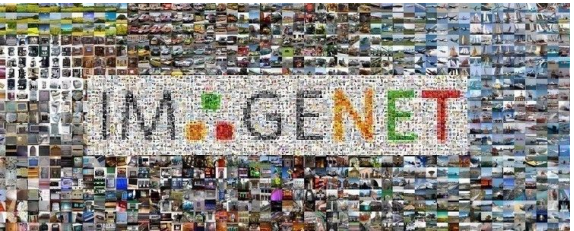
Recycling is NOT good for the world. It is bad for the environment, it is bad for our health, and it is bad for our economy. I'm not kidding. Recycling is not good for the environment. It is destructive to the earth and it is a major contributor to global warming. Recycling is not good for our health. It contributes to obesity and diseases like heart





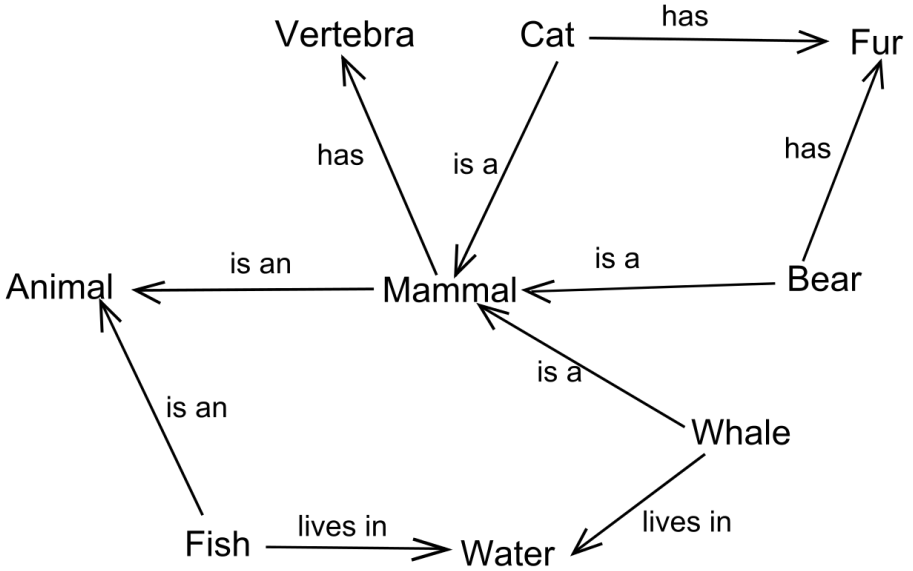






Inteligência Artificial ***Ontem***

- *“Reasoning as Search”*
  - Satisfação de restrições
  - Otimização Combinatória
- Discreta
- Simbólica
- Relacional



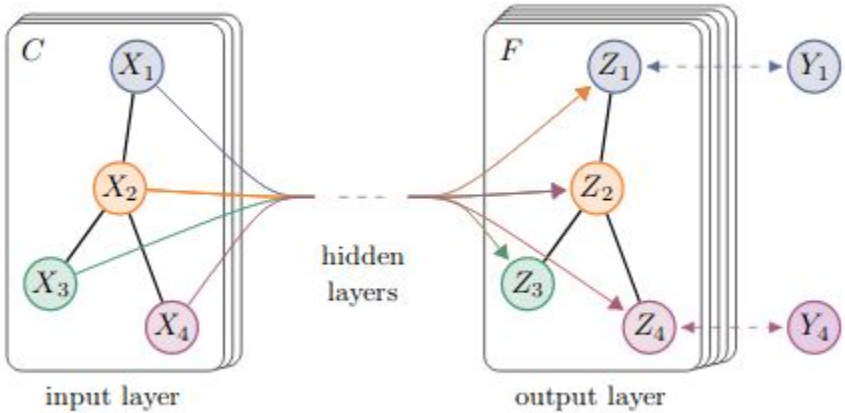
Problemas Relacionais Hoje

- Física (sistemas de partículas)
- Química (moléculas)
- Bioinformática (DNA)
- Matemática/Lógica (Expressões)
- Satisfação de Restrições / Otimização Combinatória
- Redes sociais

Graph Neural Networks

A new model for learning in graph domains
M Gori, G Monfardini, F Scarselli - ... on **Neural Networks**, 2005., 2005 - ieeexplore.ieee.org
In several applications **the** information is naturally represented by **graphs**. Traditional approaches cope with graphical data structures using **a** preprocessing phase which transforms **the graphs** into **a** set of flat vectors. However, in this way, important topological ...
☆ 99 Cited by 255 Related articles All 3 versions

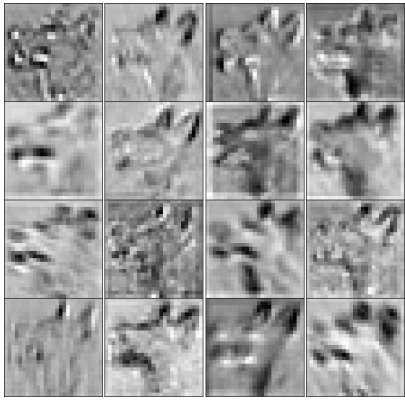
The graph neural network model
F Scarselli, M Gori, AC Tsoi... - ... on **Neural Networks**, 2008 - ieeexplore.ieee.org
Many underlying relationships among data in several areas of science and engineering, eg, computer vision, molecular chemistry, molecular biology, pattern recognition, and data mining, can be represented in terms of graphs. In this paper, we propose a new neural ...
☆ 99 Cited by 777 Related articles All 14 versions



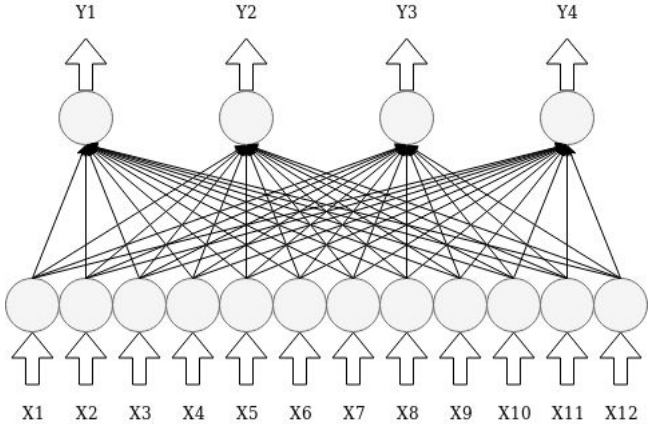
(a) Graph Convolutional Network

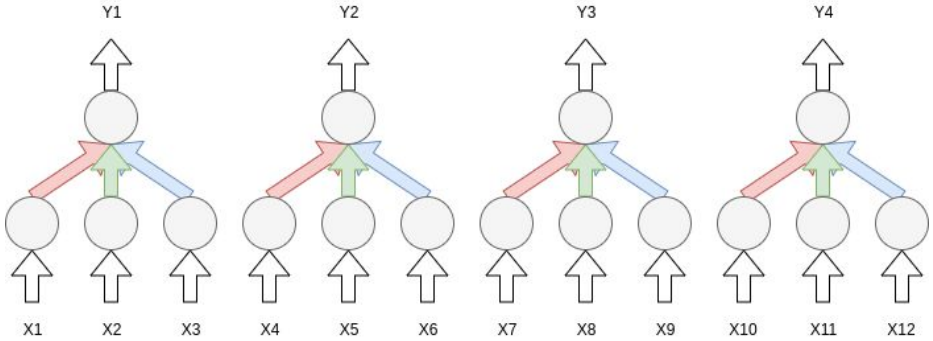
(b) Hidden layer activations

CNNs Recap



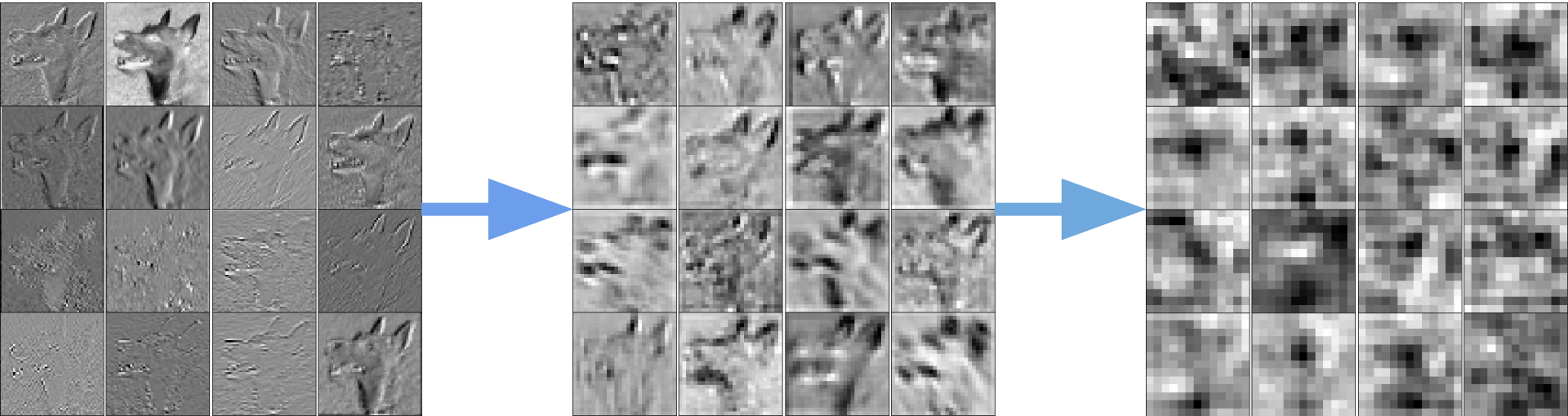
- Compartilhamento de Parâmetros
  - **Espaço de parâmetros reduzido**
  - **Equivariância a Translação**
  - **Inputs de tamanho variável**





CNNs Recap

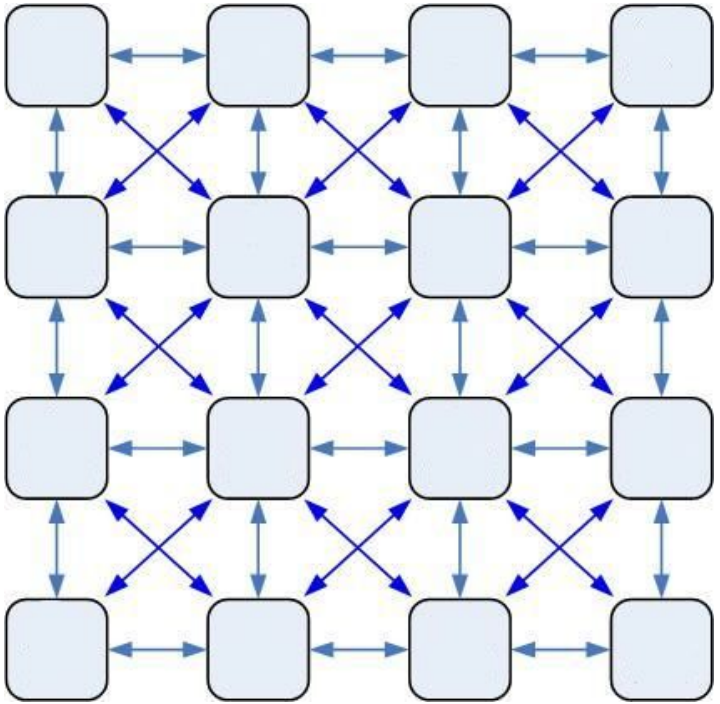
- CNNs aprendem a fazer feature engineering por conta própria
- Mais camadas → features de nível mais alto



CNNs Recap

$$X'_{ab} \leftarrow \frac{1}{9} \sum_{i=a-1}^{a+1} \sum_{b-1}^{b+1} X_{ij}$$

- Filtros convolucionais implementam transformações *locais* (vizinhança 9-conectada)
- Imagens são como grafos com topologia em grid



CNNs Recap

$$X'_{ab} \leftarrow \sum_{i=a-\frac{n-1}{2}}^{a+\frac{n-1}{2}} \sum_{b-\frac{n-1}{2}}^{b+\frac{n-1}{2}} W_{ij} X_{ij}$$

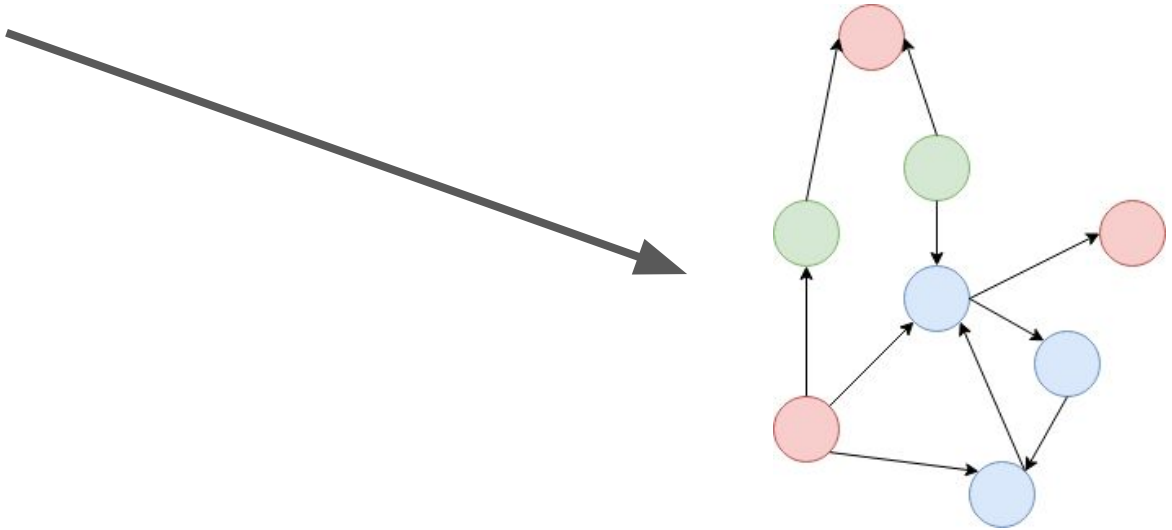
- De maneira geral
- Em uma CNN, os parâmetros são os **pesos**

$$W = \begin{pmatrix} W_{1,1} & \cdots & W_{1,n} \\ \vdots & \ddots & \vdots \\ W_{n,1} & \cdots & W_{n,n} \end{pmatrix}$$

CNNs → GNNs

$$X'_{ab} \leftarrow \frac{1}{9} \sum_{i=a-1}^{a+1} \sum_{b-1}^{b+1} X_{ij}$$

- A princípio podemos aplicar transformações análogas em outras topologias



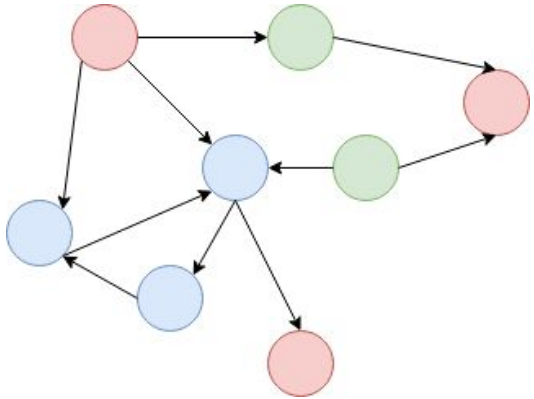
CNNs → GNNs

- Objetivo: substituir **imagens** por **grafos** como input
- Numa CNN, cada pixel é representado por um vetor n-dimensional (3D = RGB no input)
- Numa GNN, cada **nodo** do grafo de entrada é representado por um vetor n-dim

A new model for learning in graph domains
M Gori, G Monfardini, F Scarselli - ... on **Neural Networks**, 2005., 2005 - ieeexplore.ieee.org
In several applications **the** information is naturally represented by **graphs**. Traditional approaches cope with graphical data structures using **a** preprocessing phase which transforms **the graphs** into **a** set of flat vectors. However, in this way, important topological ...
☆ 99 Cited by 255 Related articles All 3 versions

The graph neural network model
F Scarselli, M Gori, AC Tsoi... - ... on **Neural Networks**, 2008 - ieeexplore.ieee.org
Many underlying relationships among data in several areas of science and engineering, eg, computer vision, molecular chemistry, molecular biology, pattern recognition, and data mining, can be represented in terms of graphs. In this paper, we propose a new neural ...
☆ 99 Cited by 777 Related articles All 14 versions

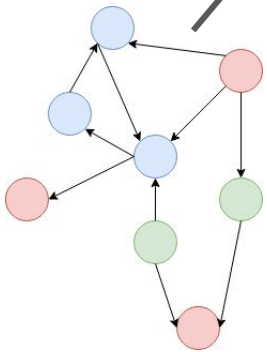
Semi-supervised classification with graph convolutional networks
TN Kipf, M Welling - arXiv preprint arXiv:1609.02907, 2016 - arxiv.org
We present a scalable approach for semi-supervised learning on **graph**-structured data that is based on an efficient variant of **convolutional** neural **networks** which operate directly on **graphs**. We motivate the choice of our **convolutional** architecture via a localized first-order ...
☆ 99 Cited by 2114 Related articles All 11 versions »



Graph Convolution Layer

- Idêntica à convolução tradicional, exceto por:
  - Vizinhança não é necessariamente em grid
  - Não temos mais um peso específico para cada vizinho
  - Ao invés disso: aplicamos uma transformação a cada feature

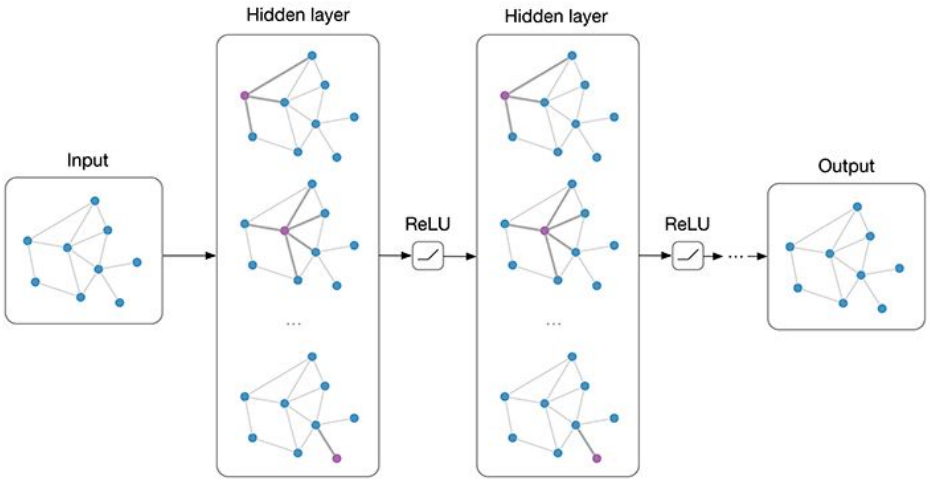
$$X_i' \leftarrow \sum_{j \in \mathcal{N}(i) \cup \{i\}} \Theta \times X_j$$



```python
import torch

# Nº of nodes
N = 8

# Adjacency matrix (NxN)
A = torch.tensor([
    [0,1,0,1,0,0,0,0],
    [0,0,1,0,0,0,0,0],
    [0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,1],
    [0,0,0,1,0,0,0,0],
    [0,0,0,1,0,0,0,0],
    [0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0],
]).float()

# Add self-edges
A += torch.eye(N)

# Dimensionality of feature vectors
d = 16
# Initial feature vectors
x = torch.randn(N, d)

# Parameters
θ = torch.nn.Linear(d, d, bias=False)

# GC layer
x = torch.mm(A,θ(x))
```

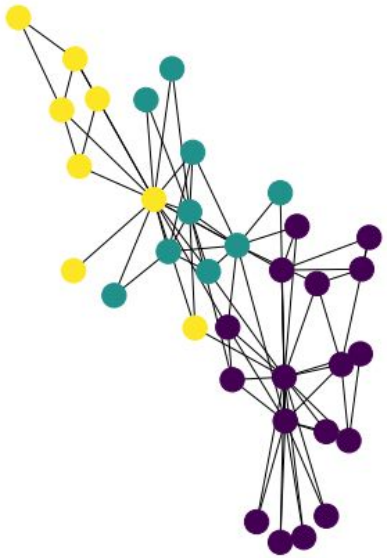
Graph Convolution Layer

- Podemos empilhar múltiplas camadas de GC



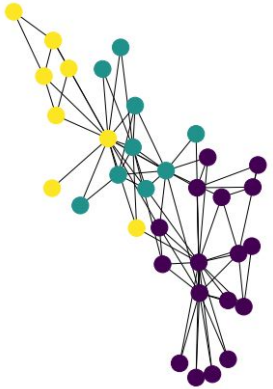

```python
import torch

# Nº of nodes
N = 8

# Adjacency matrix (NxN)
A = torch.tensor([
    [0,1,0,1,0,0,0,0],
    [0,0,1,0,0,0,0,0],
    [0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,1],
    [0,0,0,1,0,0,0,0],
    [0,0,0,1,0,0,0,0],
    [0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0],
]).float()

# Add self-edges
A += torch.eye(N)

# Dimensionality of feature vectors
d = 16
# Initial feature vectors
x = torch.randn(N, d)

# Parameters
01 = torch.nn.Linear(d, d, bias=False)
02 = torch.nn.Linear(d, d, bias=False)

# GC layers
x = torch.relu(torch.mm(A,01(x)))
x = torch.relu(torch.mm(A,02(x)))
```

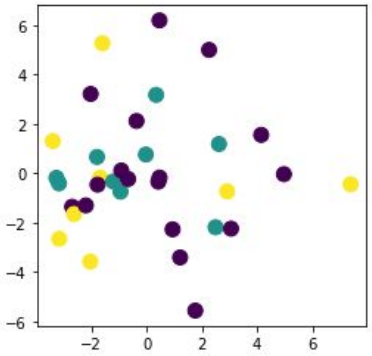
Graph Convolution Layer



```python
import torch
import numpy as np
import networkx as nx
from matplotlib import pyplot as plt
from sklearn.decomposition import PCA
from networkx.algorithms.community import greedy_modularity_communities

# Get Zachary's Karate Club graph
G = nx.karate_club_graph()
A = torch.tensor(nx.to_numpy_matrix(G)).float()

# Split graph into communities
communities = greedy_modularity_communities(G)
labels = np.zeros(len(G))
for i in range(len(communities)):
    labels[list(communities[i])] = i+1

# Draw graph
nx.draw(G, node_color=labels)
plt.show()
```

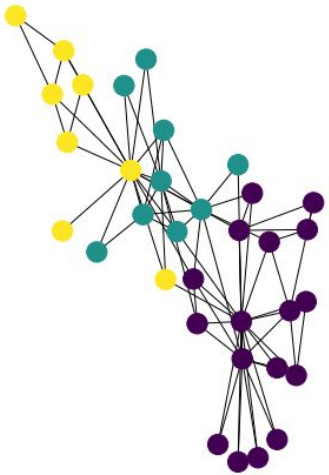
Graph Convolution Layer

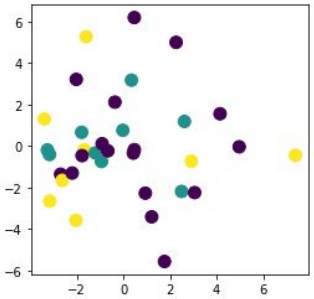


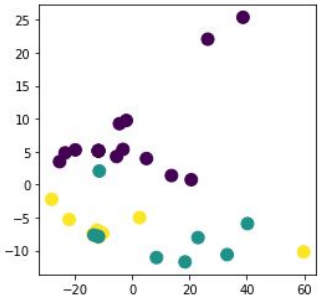



```python
# Util to project and plot feature vectors
def plot_feature_vectors(x):
    pca = PCA(n_components=2)
    x_proj = pca.fit_transform(x.detach().numpy())
    plt.figure(figsize=(4, 4))
    plt.scatter(x_proj[:,0], x_proj[:,1], c=labels, s=100)
    plt.show()

# Dimensionality of feature vectors
d = 100
# Initial feature vectors
x = torch.randn(len(G), d)

# Parameters
θ = [torch.nn.Linear(d, d, bias=False) for i in range(3)]

# Plot before GCs
plot_feature_vectors(x)
```

Graph Convolution Layer








```python
# Plot before GCs
plot_feature_vectors(x)

# Update feature vectors with repeated GCs
for i in range(3):
    x = torch.relu(torch.mm(A,θ[i](x)))

# Plot after GCs
plot_feature_vectors(x)
```

Message Passing Neural Nets

$$X_i^{(t+1)} \leftarrow \gamma^{(t)}\left(X_i^{(t)}, \square_{j \in \mathcal{N}(i)} \phi^{(t)}\left(X_i^{(t)}, X_j^{(t)}, e_{ij}\right)\right)$$

Função de *update*

Função de agregação

Função de “mensagem”

Features da aresta

Message Passing Neural Nets

```python
# edge features dimensionality
d_e = 3
# edge features
e = {
    (0,1): torch.tensor([[0,1,0]]).float(),
    (0,3): torch.tensor([[0,1,0]]).float(),
    (1,2): torch.tensor([[0,0,1]]).float(),
    (3,7): torch.tensor([[1,0,0]]).float(),
    (4,3): torch.tensor([[0,1,0]]).float(),
    (5,3): torch.tensor([[0,0,1]]).float()
}

# node embeddings dimensionality
d_n = 16
# Initial node embeddings
x = torch.randn(N, d_n)
```

```python
# Message function
φ = torch.nn.Sequential(
    torch.nn.Linear(2*d_n + d_e, 2*d_n + d_e),
    torch.nn.ReLU(),
    torch.nn.Linear(2*d_n + d_e, 2*d_n + d_e),
    torch.nn.ReLU(),
    torch.nn.Linear(2*d_n + d_e, d_n),
    torch.nn.ReLU()
)

# Update function
γ = torch.nn.LSTM(d_n, d_n)

# Aggregation function (in this case, addition)
def □(x):
    return x.sum(dim=0)
```

```python
# Run one MPNN step
def MPNN_step(x, h, c):
    for i in range(N):
        # Get neighbor indices
        js = [j for j in range(N) if A[i, j] == 1]

        # Compute message tensors
        if len(js) > 0:
            msgs = □(φ(torch.cat(
                [
                    x[i,:].unsqueeze(0).repeat(len(js),1),
                    x[js,:],
                    torch.cat([e[(i,j)] for j in js])
                ],
                dim = 1
            ))).unsqueeze(0).unsqueeze(0)
        else:
            msgs = torch.zeros(1, 1, d_n)

        # Update embedding and hidden states
        x[i:i+1, :], (h[i:i+1, :], c[i:i+1, :]) = γ(msgs, (h[i:i+1,:], c[i:i+1,:]))

    return x, (h, c)

# Run T MPNN steps
T = 20
h = torch.randn(N, 1, d_n)
c = torch.randn(N, 1, d_n)
for t in range(T):
    x, (h, c) = MPNN_step(x, h, c)
```

Aplicações de GNNs

Aprendizado Semi-Supervisionado

- Classificação de nodos de um grafo
- **Poucos labels** são conhecidos
- Como embeddings acumulam informação local, é possível treinar com menos labels

Semi-supervised classification with **graph convolutional networks**

TN Kipf, M Welling - arXiv preprint arXiv:1609.02907, 2016 - arxiv.org

We present a scalable approach for semi-supervised learning on **graph**-structured data that is based on an efficient variant of **convolutional** neural **networks** which operate directly on **graphs**. We motivate the choice of our **convolutional** architecture via a localized first-order ...

☆ 99 Cited by 2114 Related articles All 11 versions »»

Table 2: Summary of results in terms of classification accuracy (in percent).

Method	Citeseer	Cora	Pubmed	NELL
ManiReg [3]	60.1	59.5	70.7	21.8
SemiEmb [28]	59.6	59.0	71.1	26.7
LP [32]	45.3	68.0	63.0	26.5
DeepWalk [22]	43.2	67.2	65.3	58.1
ICA [18]	69.1	75.1	73.9	23.1
Planetoid* [29]	64.7 (26s)	75.7 (13s)	77.2 (25s)	61.9 (185s)
GCN (this paper)	**70.3** (7s)	**81.5** (4s)	**79.0** (38s)	**66.0** (48s)
GCN (rand. splits)	67.9 ± 0.5	80.1 ± 0.5	78.9 ± 0.7	58.4 ± 1.7

Dataset	Type	Nodes	Edges	Classes	Features	Label rate
Citeseer	Citation network	3,327	4,732	6	3,703	0.036
Cora	Citation network	2,708	5,429	7	1,433	0.052
Pubmed	Citation network	19,717	44,338	3	500	0.003
NELL	Knowledge graph	65,755	266,144	210	5,414	0.001

Few-Shot Learning

- Problema: aprender a partir de **poucos labels**
- Uma CNN produz feature maps para cada imagem de exemplo
- GNN recebe grafo totalmente conectado de feature maps
- Feature maps refinados ao longo de várias iterações
- MLP calcula similaridade entre feature maps
- Inferência: nearest neighbor

Few-shot learning with graph neural networks

V Garcia, J Bruna - arXiv preprint arXiv:1711.04043, 2017 - arxiv.org

We propose to study the problem of few-shot learning with the prism of inference on a partially observed graphical model, constructed from a collection of input images whose label can be either observed or not. By assimilating generic message-passing inference algorithms with their neural-network counterparts, we define a graph neural network architecture that generalizes several of the recently proposed few-shot learning models. Besides providing improved numerical performance, our framework is easily extended to ...

☆ 99 Cited by 128 Related articles All 2 versions »

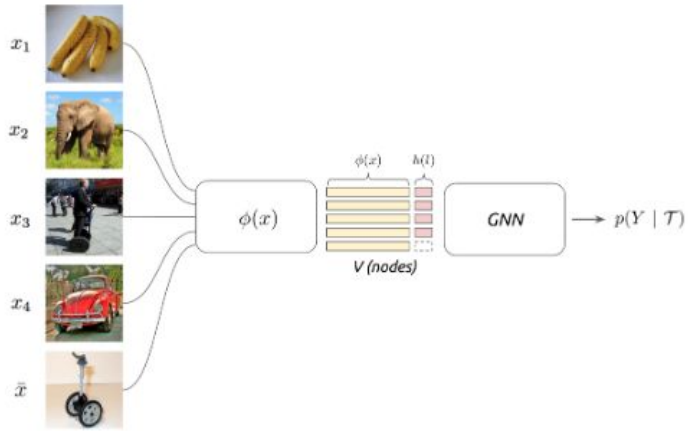


Figure 1: Visual representation of One-Shot Learning setting.

Relational Reasoning

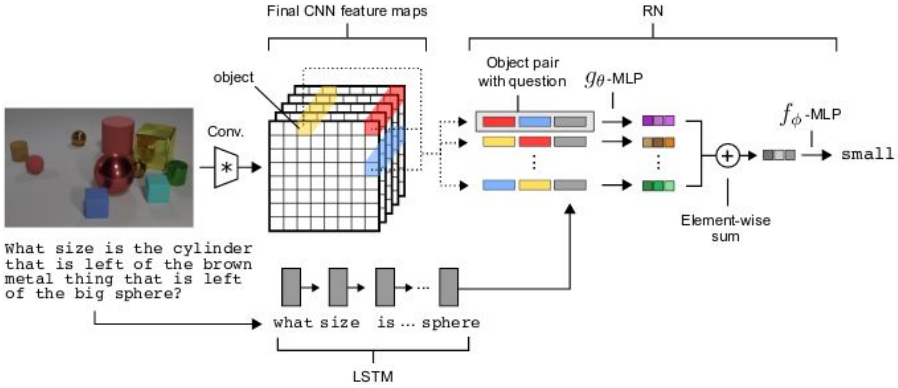
- Questão relacional sobre objetos numa imagem
- **LSTM** processa pergunta em LN e produz um *embedding de questão*
- **CNN** processa imagem e produz **d x d** feature maps
- Cada célula da matriz **d x d** é tratada como um objeto
- **GNN** (condicionada no *embedding de questão*) gera um embedding por par
- **MLP** recebe soma dos embeddings de pares e prediz

A simple neural network module for relational reasoning

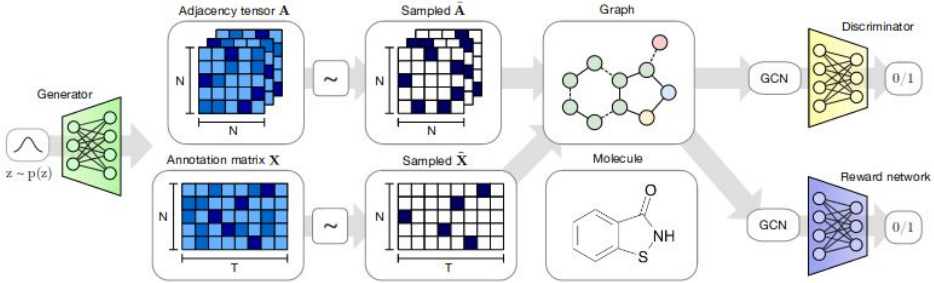
A Santoro, D Raposo, DG Barrett... - Advances in neural ..., 2017 - papers.nips.cc

Relational reasoning is a central component of generally intelligent behavior, but has proven difficult for neural networks to learn. In this paper we describe how to use Relation Networks (RNs) as a simple plug-and-play module to solve problems that fundamentally hinge on relational reasoning. We tested RN-augmented networks on three tasks: visual question answering using a challenging dataset called CLEVR, on which we achieve state-of-the-art, super-human performance; text-based question answering using the bAbI suite of ...

☆ Cited by 494 Related articles All 6 versions



Modelos Generativos de Grafos



MolGAN: An implicit generative model for small molecular graphs

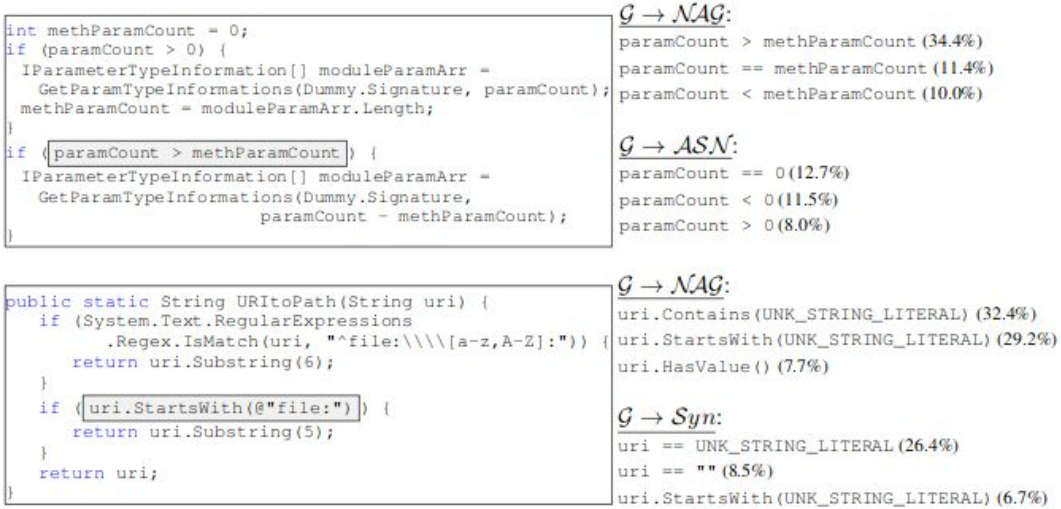
N De Cao, T Kipf - arXiv preprint arXiv:1805.11973, 2018 - arxiv.org

Deep generative models for graph-structured data offer a new angle on the problem of chemical synthesis: by optimizing differentiable models that directly generate molecular graphs, it is possible to side-step expensive search procedures in the discrete and vast space of chemical structures. We introduce MolGAN, an implicit, likelihood-free generative model for small molecular graphs that circumvents the need for expensive graph matching procedures or node ordering heuristics of previous likelihood-based methods. Our method …

☆ 99 Cited by 84 Related articles All 3 versions »

- GAN onde discriminante é uma GCN
- Discriminante separa grafos reais de artificiais
- Geradora produz grafos realísticos para enganar a discriminante

Modelos Generativos de Grafos



- Objetivo: hole completion
- Encoder (**GNN**) - Decoder
- Encoder recebe AST de código numa linguagem de programação
- Decoder produz uma AST para completar a lacuna

Generative code modeling with graphs

M Brockschmidt, M Allamanis, AL Gaunt… - arXiv preprint arXiv …, 2018 - arxiv.org

Generative models for source code are an interesting structured prediction problem, requiring to reason about both hard syntactic and semantic constraints as well as about natural, likely programs. We present a novel model for this problem that uses a graph to represent the intermediate state of the generated output. The generative procedure interleaves grammar-driven expansion steps with graph augmentation and neural message passing steps. An experimental evaluation shows that our new model can generate …

☆ 99 Cited by 15 Related articles All 4 versions »

Simulação Física

- GNN refina embeddings de partículas e prediz as derivadas (deltas) da posição e do momentum de cada uma
- Usados para montar equação diferencial
- Equação resolvida com ODE diferenciável para obter função de evolução do sistema

Hamiltonian Graph Networks with ODE Integrators

A Sanchez-Gonzalez, V Bapst, K Cranmer... - arXiv preprint arXiv ..., 2019 - arxiv.org

We introduce an approach for imposing physically informed inductive biases in learned simulation models. We combine graph networks with a differentiable ordinary differential equation integrator as a mechanism for predicting future states, and a Hamiltonian as an internal representation. We find that our approach outperforms baselines without these biases in terms of predictive accuracy, energy accuracy, and zero-shot generalization to time-step sizes and integrator orders not experienced during training. This advances the state-of ...

☆ 99 Cited by 1 All 4 versions »

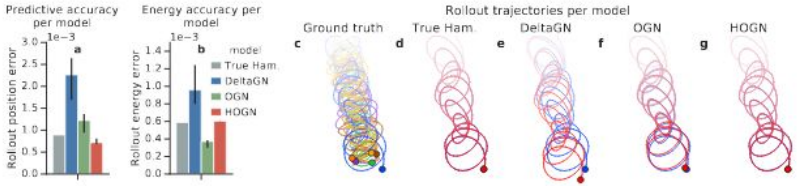
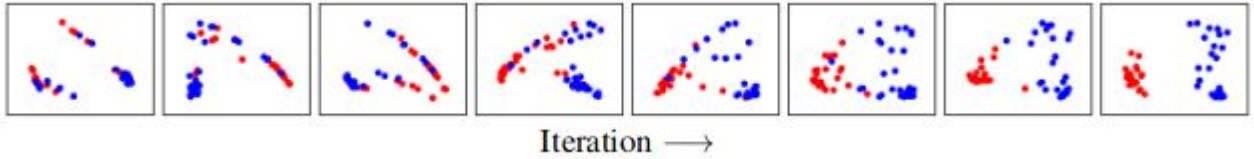


Figure 2: **(a)** Predictive accuracy (on 20-step trajectory) across models using RK4 for the ODE based models. The HOGN is most accurate. **(b)** Energy accuracy across the same models. **(c-g)** Last 300 steps of a 500-step trajectory of a 6-particle system, where dots indicate the final position of the particles (colors fade into the past). **(c)** Ground truth trajectory for all particles. **(d-g)** Trajectory for one of the particles (blue) superimposed by the trajectory obtained (red) when integrating the True Hamiltonian, or **(e-g)** using the best seed of the different learned models, at a time step of 0.1. While errors of all models are very small at the beginning of the trajectory, the HOGN is the only learned model still indistinguishable from ground truth at the end of the long 500-step trajectory (video link).

Satisfação de Restrições em problemas NP-Completos

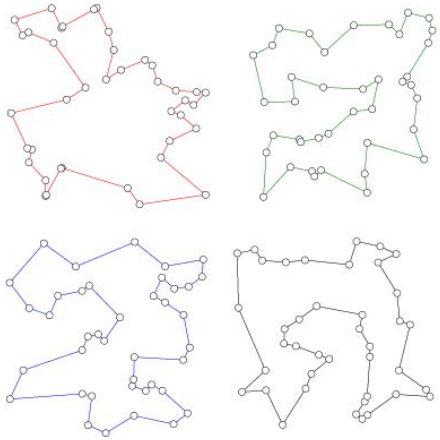


- Input: expressão booleana na CNF
  - F(x) = (¬**x1** ∨ ¬**x2** ∨ **x5**) ∧ (**x2** ∨ **x3** ∨ **x4**)
- Modelada como grafo entre literais (**x1**, ¬**x1**, **x2**, ¬**x2**, ...) e cláusulas
- **GNN** recebe grafo e refina embeddings de literal
- **MLP** calcula *"voto"* para cada literal
- Predição: voto médio

Learning a **SAT solver** from single-bit supervision
D Selsam, M Lamm, B Bünz, P Liang... - arXiv preprint arXiv ..., 2018 - arxiv.org
We present NeuroSAT, a message passing neural network that learns to **solve SAT** problems after only being trained as a classifier to predict satisfiability. Although it is not competitive with state-of-the-art **SAT** solvers, NeuroSAT can **solve** problems that are ...
☆ Cited by 64 Related articles All 6 versions »

Problemas NP-Completos em Grafos

- Input: grafo euclidiano
- Variante de decisão do TSP
- **GNN** refina embeddings de nodo e de aresta
- **MLP** calcula *"votos"* para os embeddings de aresta
- Predição: voto médio

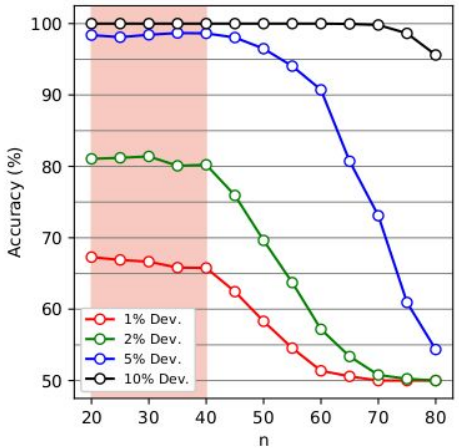


Learning to **solve np-complete** problems: A graph neural network for decision tsp

M Prates, PHC Avelar, H Lemos, LC Lamb... - Proceedings of the AAAI ..., 2019 - aaai.org

Abstract Graph Neural Networks (GNN) are a promising technique for bridging differential programming and combinatorial domains. GNNs employ trainable modules which can be assembled in different configurations that reflect the relational structure of each problem ...

☆ 99 Cited by 12 Related articles All 3 versions »



Otimização Combinatória

- **GNN** recebe grafo e refina embeddings de nodo
- **MLP** calcula probabilidade de cada nodo pertencer ao Maximum Independent Set
- Busca em árvore usa probabilidades para otimizar solução

Combinatorial optimization with graph convolutional networks and guided tree search

Z Li, Q Chen, V Koltun - Advances in Neural Information Processing ..., 2018 - papers.nips.cc

We present a learning-based approach to computing solutions for certain NP-hard problems. Our approach combines deep learning techniques with useful algorithmic elements from classic heuristics. The central component is a graph convolutional network that is trained to estimate the likelihood, for each vertex in a graph, of whether this vertex is part of the optimal solution. The network is designed and trained to synthesize a diverse set of solutions, which enables rapid exploration of the solution space via tree search. The ...

☆ 99 Cited by 33 Related articles All 5 versions »

Method	Solved	MIS	Time (s)
Classic	0.0%	403.98	0.31
Classic+GR+LS	7.9%	424.82	0.45
S2V-DQN	0.0%	413.77	2.26
S2V-DQN+GR+LS	8.9%	424.98	2.41
Gurobi	98.5%	426.86	175.83
Z3	**100.0%**	–	**0.01**
ReduMIS	**100.0%**	**426.90**	47.79
Ours	**100.0%**	**426.90**	11.47

Table 1: Results on the SATLIB test set. Fraction of solved SAT instances, average independent set size, and runtime.

Method	Solved	MIS	Time (s)
Classic	0.0%	453.25	0.30
Classic+GR+LS	75.0%	491.05	0.45
S2V-DQN	0.0%	462.05	2.19
S2V-DQN+GR+LS	80.0%	491.50	2.37
Gurobi	80.0%	–	141.66
Z3	**100.0%**	–	**0.01**
ReduMIS	**100.0%**	**492.85**	21.90
Ours	**100.0%**	**492.85**	12.20

Table 2: Results on the SAT Competition 2017. Fraction of solved SAT instances, average independent set size, and runtime.

DL em Redes Sociais

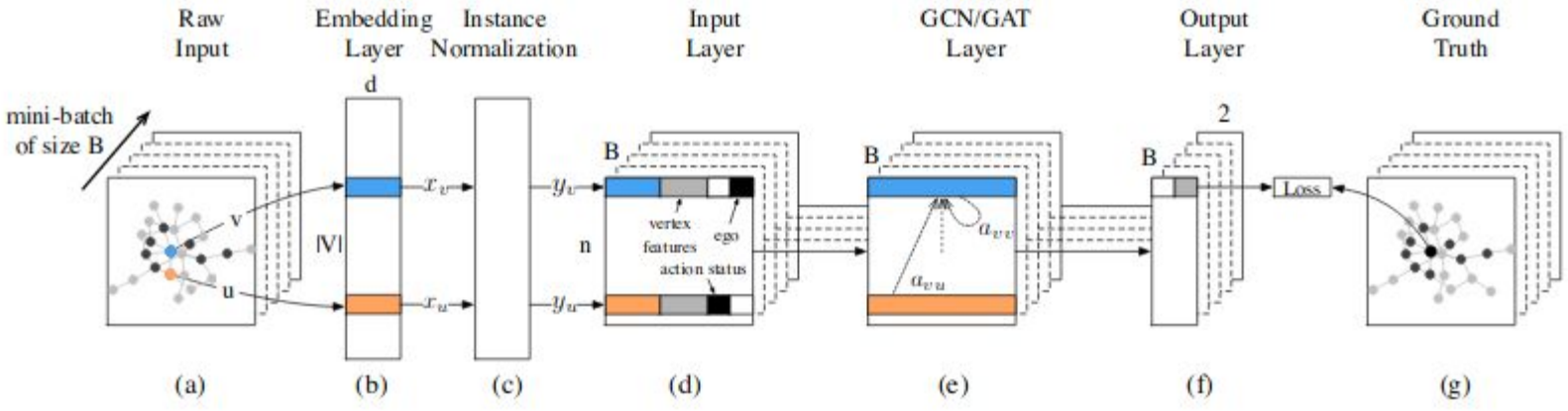
- GNNs podem ser usadas para predizer propriedades de agentes (p.ex. Influência) em redes sociais

Deepinf: Social influence prediction with deep learning

J Qiu, J Tang, H Ma, Y Dong, K Wang... - Proceedings of the 24th ..., 2018 - dl.acm.org

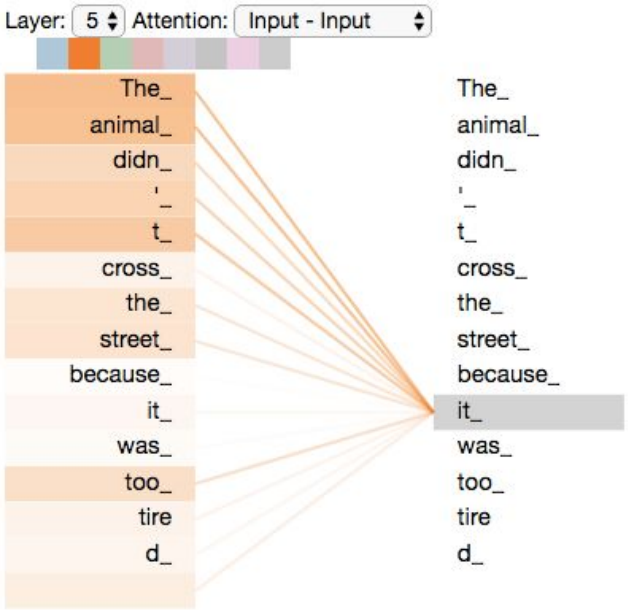
Social and information networking activities such as on Facebook, Twitter, WeChat, and Weibo have become an indispensable part of our everyday life, where we can easily access friends' behaviors and are in turn influenced by them. Consequently, an effective social influence prediction for each user is critical for a variety of applications such as online recommendation and advertising. Conventional social influence prediction approaches typically design various hand-crafted rules to extract user-and network-specific features ...

☆ 99 Cited by 25 Related articles All 6 versions



Notas Finais

- DL em grafos abre possibilidades
  - Redes sociais
  - Moléculas
  - Expressões simbólicas
  - Raciocínio relacional
  - Código
  - Otimização combinatória
  - Satisfação de restrições
  - Semi-supervised learning
  - Few-shot learning
  - Física
- Self-attention ≅ GNN



Attention is all you need

A Vaswani, N Shazeer, N Parmar… - Advances in neural …, 2017 - papers.nips.cc

The dominant sequence transduction models are based on complex recurrent orconvolutional neural networks in an encoder and decoder configuration. The best performing such models also connect the encoder and decoder through an attentionm …

☆ 99 Cited by 4694 Related articles All 12 versions »

Perguntas?

Marcelo Prates
marceloorp@gmail.com
https://www.linkedin.com/in/marceloprates/

Matheus Gonzaga
msgonzaga101@gmail.com
https://www.linkedin.com/in/msgonzaga/