

Graph Transformer Networks

(YUN ET AL., 2019)

Ana Carolina Erthal e Felipe Lamarca

Escola de Matemática Aplicada
Fundação Getulio Vargas

ESTRUTURA DA APRESENTAÇÃO

1	Introdução	2
1.1	Estado da arte	2
1.2	Soluções anteriores	3
1.3	Graph Transformer Networks	5
2	Metodologia	6
2.1	Definições	6
2.2	Graph Convolutional Network	8
2.3	Geração de Meta-Paths	9
3	Graph Transformer Network	11
4	Experimentos	13
5	Referências bibliográficas	15

INTRODUÇÃO

ESTADO DA ARTE

Para prever arestas e classificar nós e grafos, normalmente usa-se Graph Neural Networks (GNNs).

Esse modelo, no entanto, assume que os **grafos são homogêneos**, o que não é verdade em muitos casos.

Grafos de citações são heterogêneos

- ▶ Nós representam Autores, Papers, Conferências
- ▶ Arestas representam relações do tipo Autor-Paper, Paper-Conferência

Uma abordagem naive é ignorar esses tipos e tratar todos os grafos como homogêneos, o que é uma abordagem sub-ótima na medida em que implica perda de informação.

INTRODUÇÃO

SOLUÇÕES ANTERIORES

Para contornar problema, é comum...

1. Pré-processar o grafo para transformá-lo em homogêneo
2. Aplicar uma GNN em seguida

Pré-processamento através de meta-paths

- ▶ Relações compostas entre diferentes tipos de nó
- ▶ Selecionados manualmente caso a caso
- ▶ Escolhas ruins de meta-paths podem levar a resultados ruins

INTRODUÇÃO

SOLUÇÕES ANTERIORES

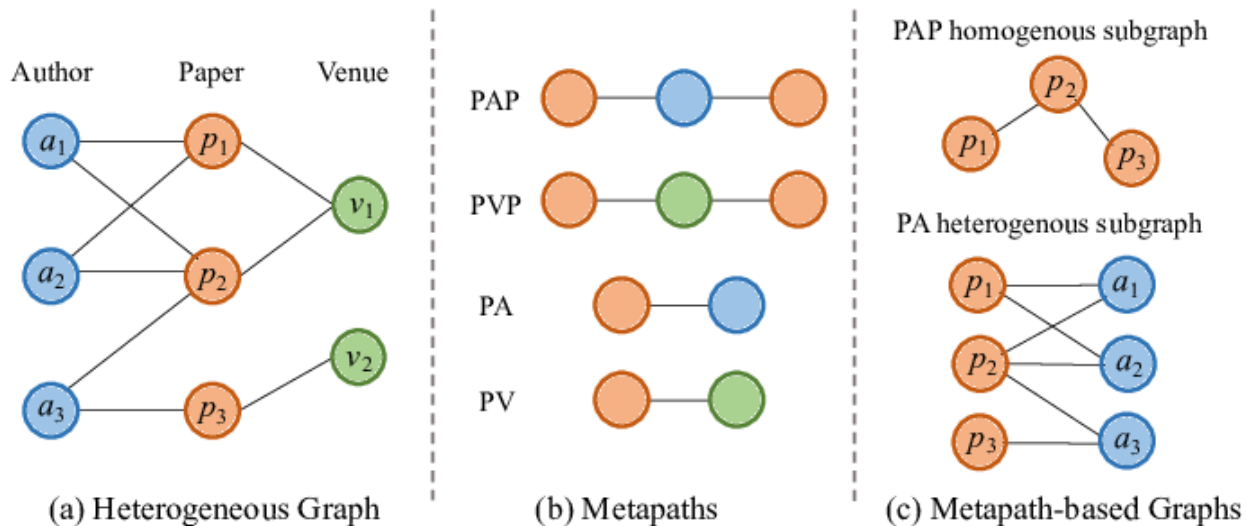


Figure. (CAI et al., 2021)

INTRODUÇÃO

GRAPH TRANSFORMER NETWORKS

A solução trazida pela **GTN** é um framework end-to-end que:

1. Aprende a transformar grafos heterogêneos em homogêneos, identificando meta-paths úteis de acordo com a tarefa em questão
2. Aprende *node representations* no grafo transformado através de uma GNN para poder realizar a tarefa.

METODOLOGIA

DEFINIÇÕES

Definition 2.1 (Grafo heterogêneo)

$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{T}^v, \mathcal{T}^e)$ é um grafo direcionado onde cada nó $v \in \mathcal{V}$ e cada aresta $e \in \mathcal{E}$ estão associadas às suas respectivas funções de mapeamento de tipo $f_v : \mathcal{V} \rightarrow \mathcal{T}^v$ e $f_e : \mathcal{E} \rightarrow \mathcal{T}^e$.

Quando $|\mathcal{T}^e| = 1$ e $|\mathcal{T}^v| = 1$, trata-se de um grafo homogêneo. Trataremos do caso $|\mathcal{T}^e| > 1$.

Seja N o número de nós do grafo. O grafo heterogêneo pode ser representado por um conjunto de matrizes de adjacência $\{A_k\}_{k=1}^K$, onde $K = |\mathcal{T}^e|$, e $A_k \in \mathbf{R}^{N \times N}$ é a matriz de adjacência onde $A_k[i, j]$ é não-nulo quando há uma aresta do tipo k entre j e i . De maneira mais concisa, o grafo heterogêneo pode ser escrito como o tensor $\mathbb{A} \in \mathbf{R}^{N \times N \times K}$.

METODOLOGIA

DEFINIÇÕES

Definition 2.2 (Meta-path)

Meta-path (p): caminho em um grafo heterogêneo, dado por $v_1 \xrightarrow{t_1} v_2 \xrightarrow{t_2} \dots \xrightarrow{t_l} v_{l+1}$, $t_l \in \mathcal{T}^e$.

Para uma sequência de tipos de aresta (t_1, t_2, \dots, t_l) , a matriz de adjacência do meta-path definido por essa sequência é dada por

$$A_p = A_{t_l} \dots A_{t_2} A_{t_1}$$

- ▶ Por exemplo, o meta-path APC (Autor-Paper-Conferência) é do tipo $A \xrightarrow{AP} P \xrightarrow{PC} C$, e a matriz de adjacência do meta-path $A_{APC} = A_{PC} A_{AP}$

METODOLOGIA

GRAPH CONVOLUTIONAL NETWORK

Uma **GCN** é utilizada para aprender *node representations* após já termos as matrizes de adjacência dos meta-paths aprendidos para o grafo \mathcal{G} . O *forward propagation* da nossa GCN será:

$$H^{(l+1)} = \sigma(\tilde{D}^{\frac{1}{2}} \tilde{A} \tilde{D}^{\frac{1}{2}} H^{(l)} W^{(l)}),$$

onde $\tilde{A} = A + I$, adicionando *self-connections*, $\tilde{D} = \sum_i \tilde{A}_{ij}$ e W é uma matriz de pesos treinável.

- ▶ Na prática, a camada de convolução é dada pela composição de uma parte fixa e uma transformação linear dos nós.
- ▶ Além disso, como veremos a seguir, A representa múltiplos meta-paths, então nos beneficiamos de várias estruturas de grafo.
- ▶ Usamos grafos direcionados, então adaptamos para $H^{(l+1)} = \sigma(\tilde{D}^{-1} \tilde{A} H^{(l)} W^{(l)})$

METODOLOGIA

GERAÇÃO DE META-PATHS

- ▶ A Graph Transformer Layer escolhe duas estruturas de grafo a partir de \mathbb{A} aplicando a *softmax* nos pesos iniciais.

Cada layer da GTN aprende a escolher matrizes de adjacência (tipos de arestas) através de uma convolução 1×1 com os pesos da softmax:

$$\begin{aligned} Q &= F(\mathbb{A}; \phi^{(k)}) \\ &= \text{conv}_{1 \times 1}(\mathbb{A}; \text{softmax}(\phi^{(k)})) \\ &= \sum_{t=1}^{|\mathcal{T}_e|} \alpha_t^{(k)} A_t, \end{aligned}$$

onde $\phi^{(k)} \in \mathbf{R}^{1 \times 1 \times |\mathcal{T}_e|}$ é o parâmetro da convolução e $\alpha^{(k)} = \text{softmax}(\phi^{(k)})$. Na prática, uma combinação linear das matrizes de adjacência.

- ▶ Multiplica as duas matrizes para obter uma nova estrutura de grafo.

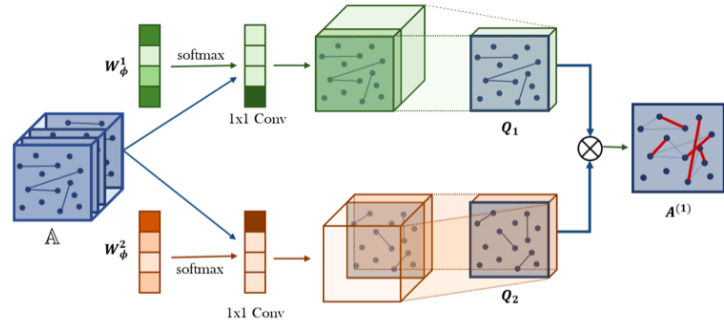


Figure. Graph Transformer Layer, (YUN et al., 2019)

METODOLOGIA

GERAÇÃO DE META-PATHS

A matriz de adjacência de meta-paths de tamanho l é dada por:

$$A_p = \left(\sum_{t_1 \in \mathcal{T}^e} \alpha_{t_1}^{(1)} A_{t_1} \right) \left(\sum_{t_2 \in \mathcal{T}^e} \alpha_{t_2}^{(2)} A_{t_2} \right) \dots \left(\sum_{t_l \in \mathcal{T}^e} \alpha_{t_l}^{(l)} A_{t_l} \right)$$

Assim, com l layers conseguimos aprender meta-paths de tamanho l .

- Uma questão é que adicionando uma layer, aumentamos o tamanho do meta-path, e às vezes é bom ter meta-paths curtos. Por isso, incluímos $A_0 = I$ em \mathbb{A} .

GRAPH TRANSFORMER NETWORK

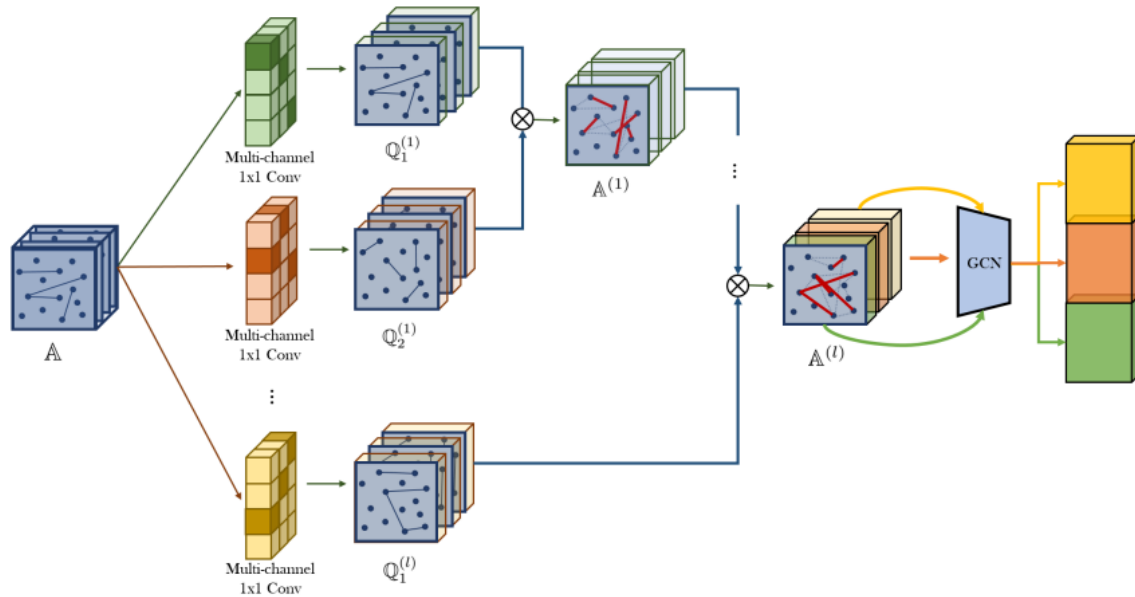


Figure. Graph Transformer Network, (YUN et al., 2019)

GRAPH TRANSFORMER NETWORK

Para considerarmos vários meta-paths simultaneamente, utilizamos convoluções 1×1 com output de tamanho C , sendo C o número desejado de meta-paths.

- Isso é útil porque queremos aprender *node representations* a partir de várias estruturas de grafos, para que haja mais "tentativas" de adequação à tarefa.

Sendo assim, após l GT layers, aplicamos uma GCN a cada canal matriz \mathbb{A} de meta-paths, e concatenamos os *node representations* obtidos:

$$Z = \left\| \right\|_{i=1}^C \sigma(\tilde{D}_i^{-1} \tilde{A}_i^{(l)} XW),$$

onde W é uma matriz de pesos compartilhada entre os canais e X é a matriz de features para cada nó.

Tendo *node representations*, podemos incluir camadas densas para realizar classificações.

EXPERIMENTOS

	DeepWalk	metapath2vec	GCN	GAT	HAN	GTN _{-I}	GTN (proposed)
DBLP	63.18	85.53	87.30	93.71	92.83	93.91	94.18
ACM	67.42	87.61	91.60	92.33	90.96	91.13	92.68
IMDB	32.08	35.21	56.89	58.14	56.77	52.33	60.92

Figure. Resultados na tarefa de classificação de nós (F1-score), (YUN et al., 2019)

EXPERIMENTOS

Dataset	Predefined Meta-path	Meta-path learnt by GTNs	
		Top 3 (between target nodes)	Top 3 (all)
DBLP	APCPA, APA	APCPA, APAPA, APA	CPCPA, APCPA, CP
ACM	PAP, PSP	PAP, PSP	APAP, APA, SPAP
IMDB	MAM, MDM	MDM, MAM, MDMDM	DM, AM, MDM

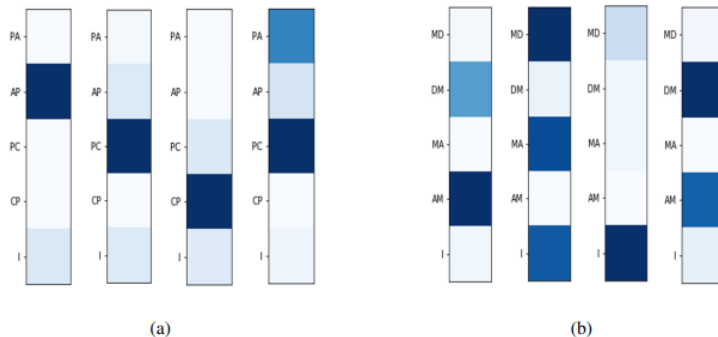




Figure. Meta-paths e atenção obtidos pelo GTN em comparação a meta-paths pré-definidos (YUN et al., 2019)

REFERÊNCIAS BIBLIOGRÁFICAS

 CAI, X. et al. Hmsg: Heterogeneous graph neural network based on metapath subgraph learning. 09 2021.

 YUN, S. et al. Graph transformer networks. *CoRR*, abs/1911.06455, 2019. Disponível em: <<http://arxiv.org/abs/1911.06455>>.

 YUN, S. et al. Graph transformer networks: Learning meta-path graphs to improve gnns. *Neural Networks*, v. 153, p. 104–119, 2022. ISSN 0893-6080. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0893608022002003>>.