Yale University

SMU SINGAPORE MANAGEMENT UNIVERSITY

School of Computing and Information Systems

Text-Attributed Graph Representation Learning: Methods, Applications, and Challenges Section 1 (30 min)

Delvin Ce Zhang[1], Menglin Yang[1], Rex Ying[1], and Hady W. Lauw[2]

[1]Yale University, [2]Singapore Management University

Textual Documents are Ubiquitous

- Ubiquity of unstructured textual documents
  - Unstructured, noisy, dynamic, multi-lingual, …

News articles

Academic papers

Today was a great day for your restaurant. There was a huge buffet with important guests. It seems that your team coped perfectly, after all, the customers left satisfied. Now it only remains to wash the dishes and prepare for the next day. Previously, the sheer amount of dirty dishes would have caused problems.

But not now, now you have Sun Eco extra power dishwasher tablets. They save time and money, because you only need one tablet (one pack contains 175 pieces). Moreover, do not worry about having to clean the machine after - as the tablet and it's plastic wrap dissolve completely during the washing process.

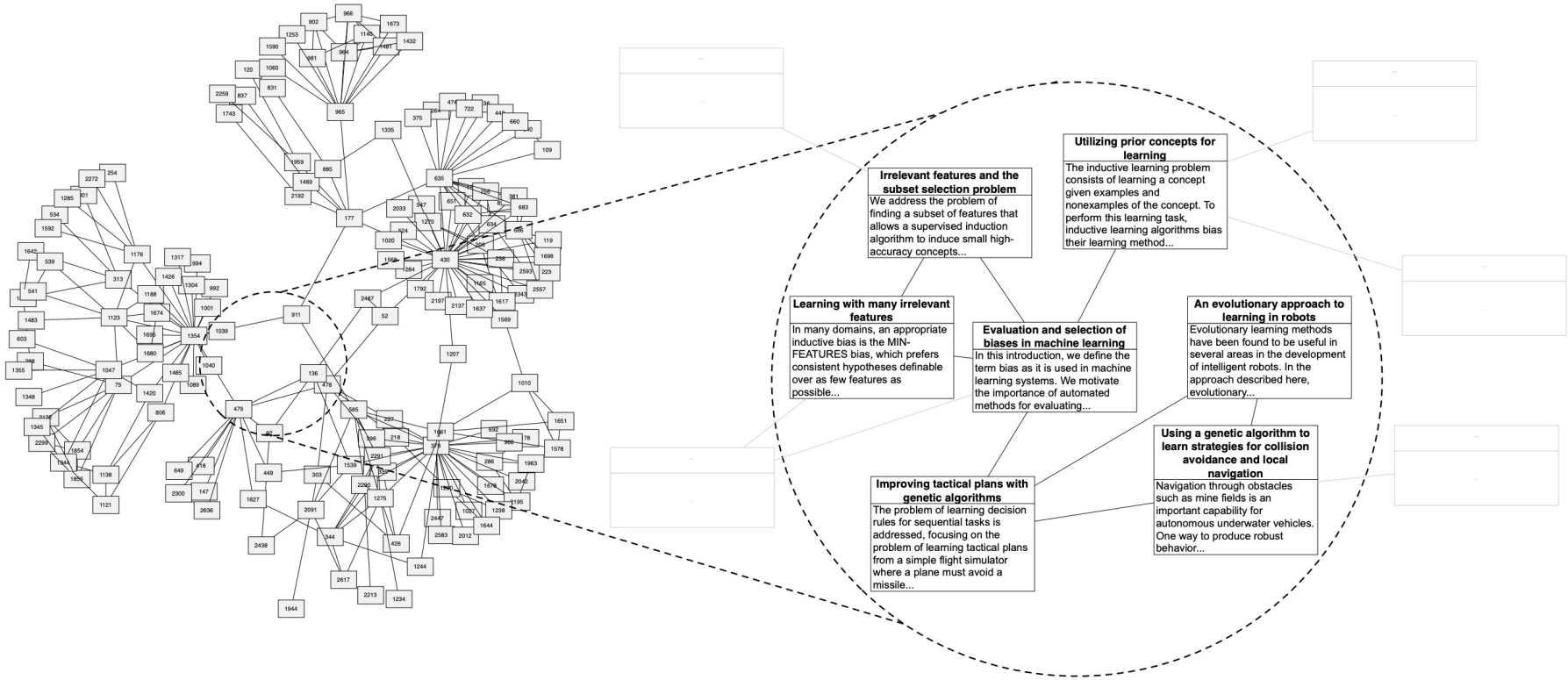
After use, all that remains is to admire the result: all the dirt on the dishes, including soot on the pans, oil, grease and food residues will simply disappear. You will be left with just sparklingly clean dishes without marks and stains. Even pre-rinsing is not required, which saves water and reduces company expenses. In addition, the composition does not include phosphates, so there are no harmful substances or tastes left on the dishes

Use a secret weapon against massive loads of dirty dishes!

Product descriptions

Text-Attributed Graph

- Documents are also connected in a graph structure – **Text-Attributed Graph**
  - Academic citation graph, news article hyperlink graph, product contextual graph…
  - Consisting of a corpus of documents $\mathcal{D}$, and graph structure $\mathcal{E}$.

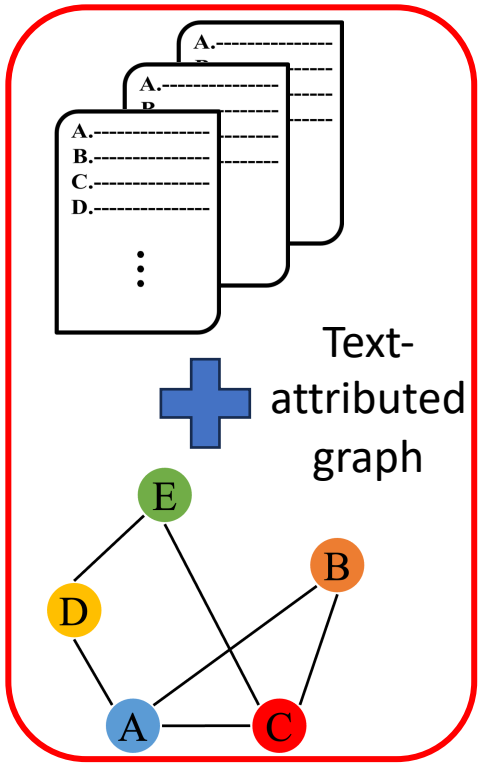


How to Process Text-Attributed Graph

- Graph Neural Networks (GNNs)
  - Pros: Capture both node attribute and graph structure;
  - Cons: Most models don’t specifically model textual attribute, thus document representations lack language representations or linguistic semantics.
- Pre-trained Language Models (PLMs)
  - Pros: Learn contextualized document representations;
  - Cons: Most models don’t capture graph structure across documents.
- Topic Models (TMs)
  - Pros: Infer topic representation for textual documents;
  - Cons: Most models don’t capture graph structure across documents.

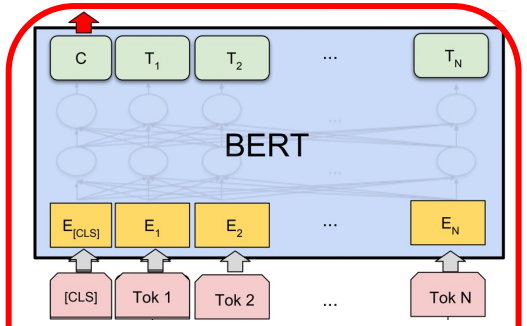
How to Process Text-Attributed Graph

- This tutorial – **Text-Attributed Graph Representation Learning**
  - Infer document representations that preserve **both i)** contextualized semantics contained in rich text documents, **and ii)** graph connectivity across documents.

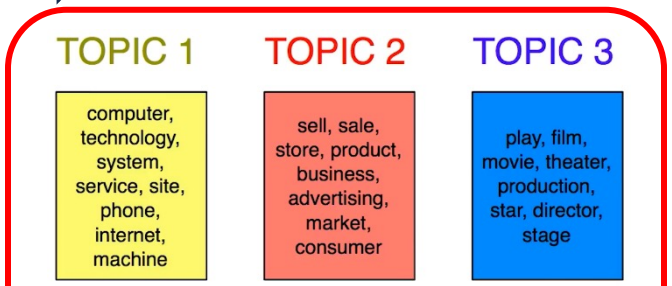


Text-attributed graph

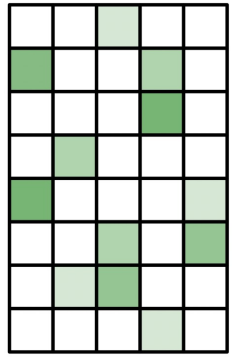
Section 1: Introduction



Section 2: GNN + LLM



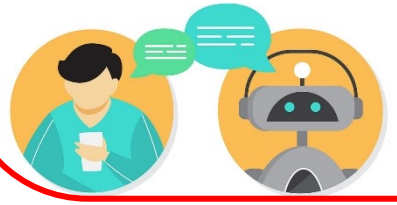
Section 3: GNN + Topic Model



Document embeddings

Text classification

Citation recommendation



Document retrieval

Section 4: Applications

Tutorial Outline

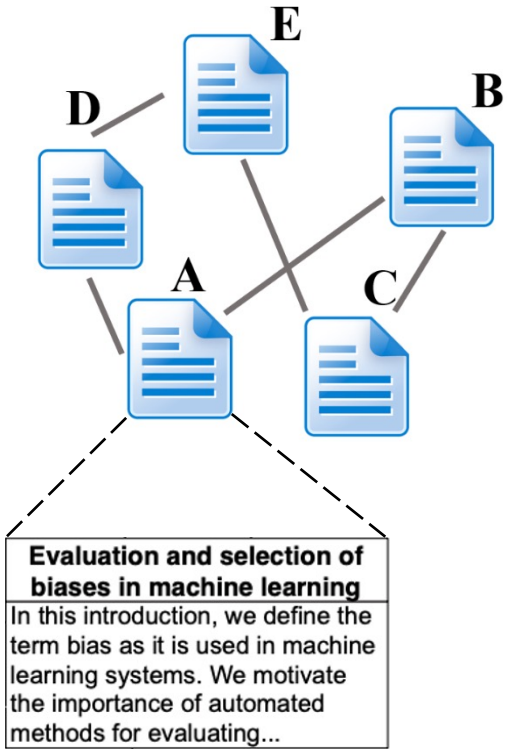
- Section 1: Text-Attributed Graph and Preliminaries (30 min)
- Section 2: PLM-based Text-Attributed Graph Models (45 min)
- Section 3: TM-based Text-Attributed Graph Models (45 min)
- Section 4: Applications, Challenges, and Future Directions (45 min)
- Summary and Q&A

Section 1: Text-Attributed Graph and Preliminaries

- Section 1.1: Formal Definition of Text-Attributed Graph (TAG)
- Section 1.2: Graph Neural Networks (GNNs)
- Section 1.3: Pre-trained Language Models (PLMs)
- Section 1.4: Topic Models (TMs)

Formal Definition of Text-Attributed Graph (TAG)

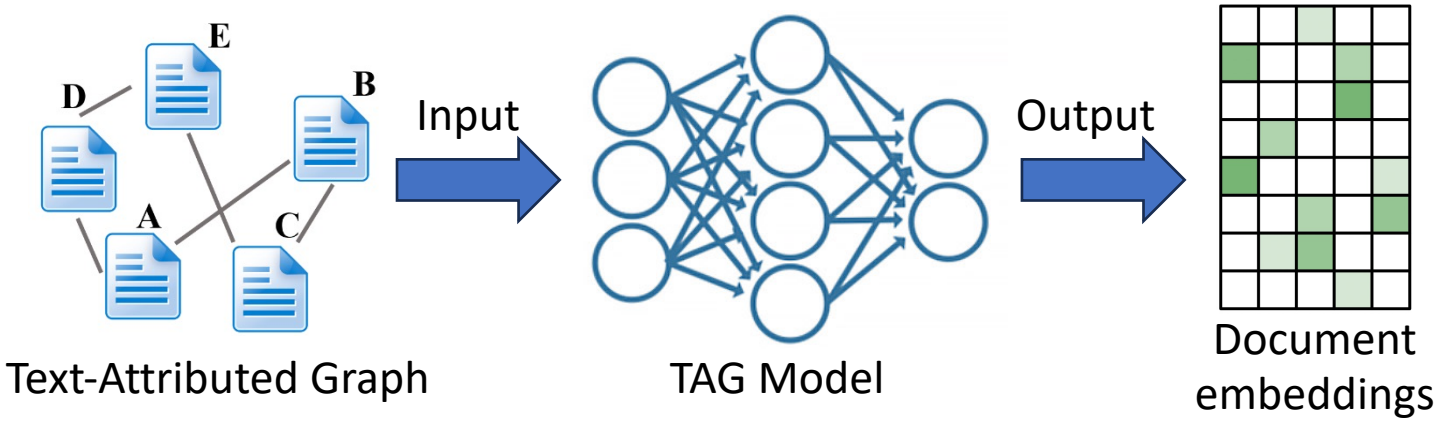
- **Text-Attributed Graph (TAG):**
- We are given a Text-Attributed Graph (TAG) $\mathcal{G} = \{\mathcal{D}, \mathcal{E}\}$.
- $\mathcal{D} = \{d_i\}_{i=1}^{N}$ is a set of N documents. Each document d_i contains a sequence of words $d_i = \{w_{i,n}\}_{n=1}^{N_{d_i}}$.
- $\mathcal{E} = \{e_{ij}\}$ is a set of graph links where $e_{ij} \in \mathcal{E}$ is there is a link between documents d_i and d_j. We model an undirected graph, $e_{ij} = e_{ji}$. Neighbor set $\mathcal{N}(i)$ contains documents directly linked to document d_i.



$e_{AB} = e_{BA}$ is a link;
$e_{AD} = e_{DA}$ is a link;
$\mathcal{N}(A) = \{B, D\}$.

Formal Definition of Text-Attributed Graph (TAG)

- **Text-Attributed Graph (TAG) Representation Learning:**
- Given a Text-Attributed Graph (TAG) $\mathcal{G} = \{\mathcal{D}, \mathcal{E}\}$ as **input**, we aim to design a model that **outputs** document representations $\boldsymbol{Z}_{\mathcal{D}} = \{\boldsymbol{z}_d\}_{d \in \mathcal{D}}$ that capture both textual semantics in $\mathcal{D}$ and graph connectivity in $\mathcal{E}$.
- Note that we do not simply consider $\mathcal{D}$ as general attribute, but instead, we specifically model language representations in text corpus $\mathcal{D}$.

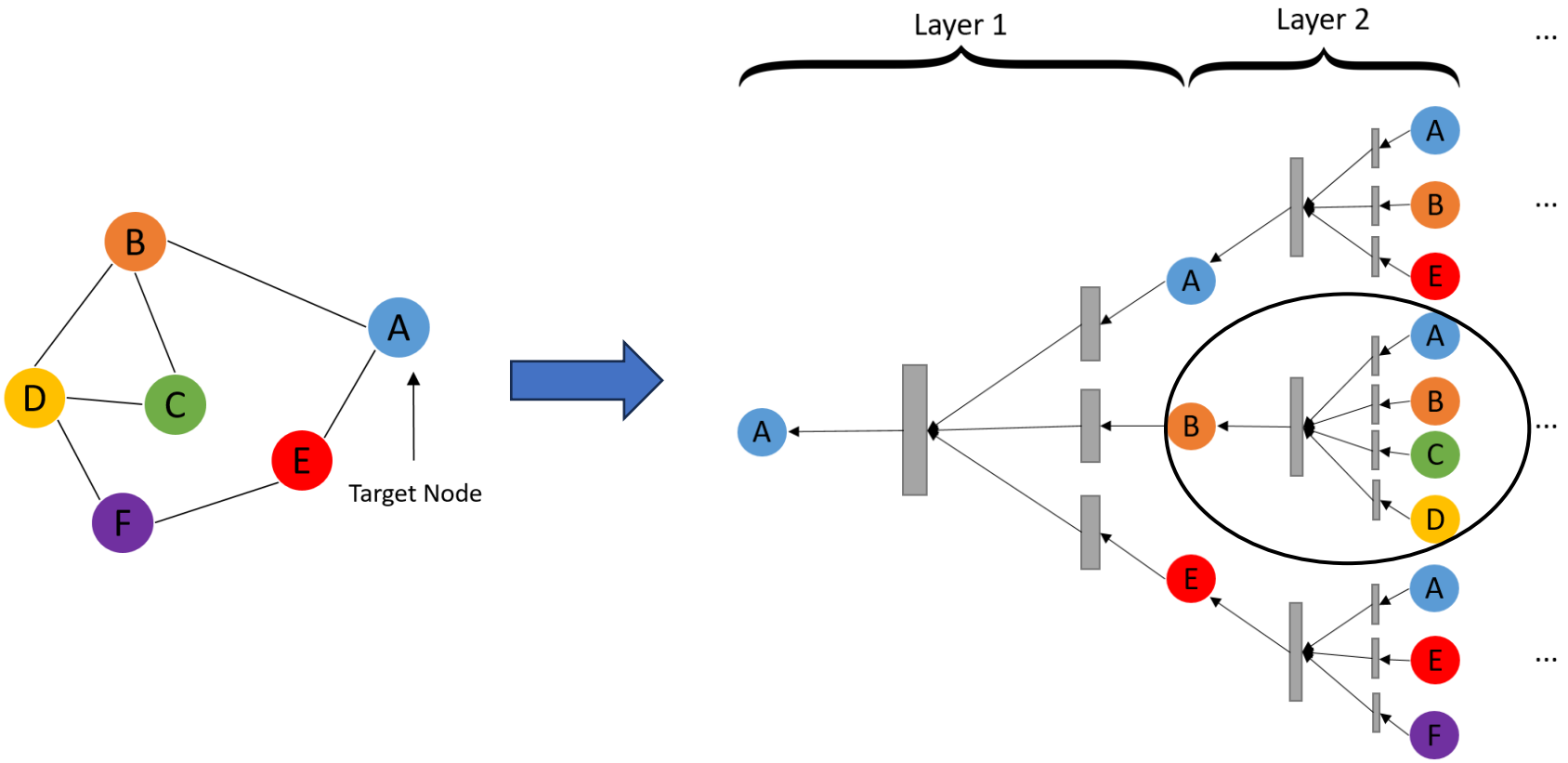


Section 1: Text-Attributed Graph and Preliminaries

- Section 1.1: Formal Definition of Text-Attributed Graph (TAG)
- Section 1.2: Graph Neural Networks (GNNs)
- Section 1.3: Pre-trained Language Models (PLMs)
- Section 1.4: Topic Models (TMs)

Graph Neural Networks (GNNs)

- Graph Neural Networks (GNNs) learn node representations that preserve both node attributed and graph structure. Below we illustrate GAT [1].

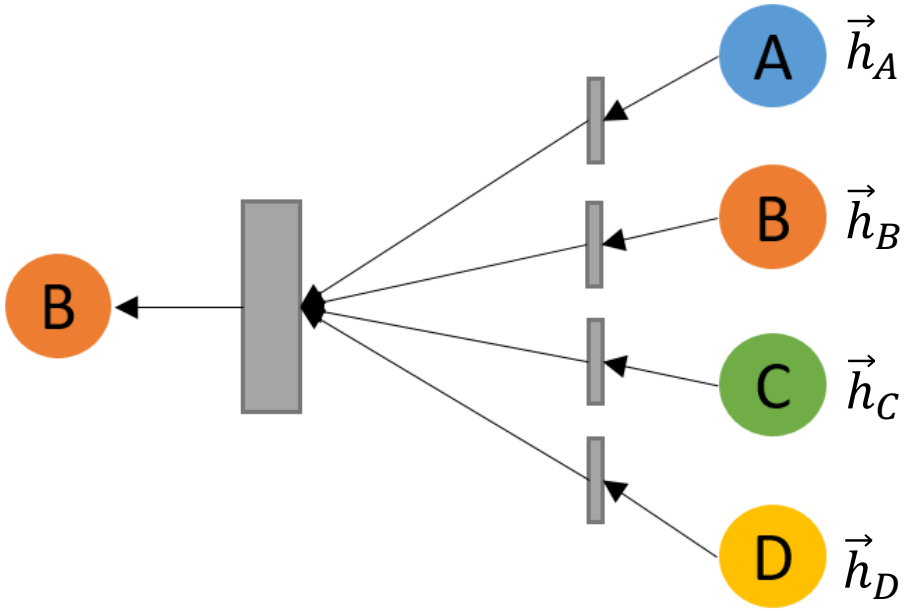


[1] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2018). Graph attention networks. In International Conference on Learning Representations, 2018.

Graph Neural Networks (GNNs)

- 1. Start with the original features, e.g., Bag-of-Words.

$$\mathbf{h} = \{\vec{h}_1, \vec{h}_2, \ldots, \vec{h}_N\}, \vec{h}_i \in \mathbb{R}^F$$



Graph Neural Networks (GNNs)

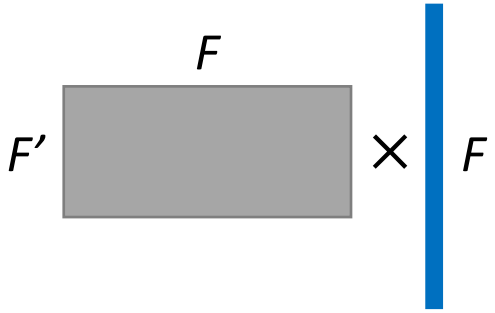
- 1. Start with the original features, e.g., Bag-of-Words.

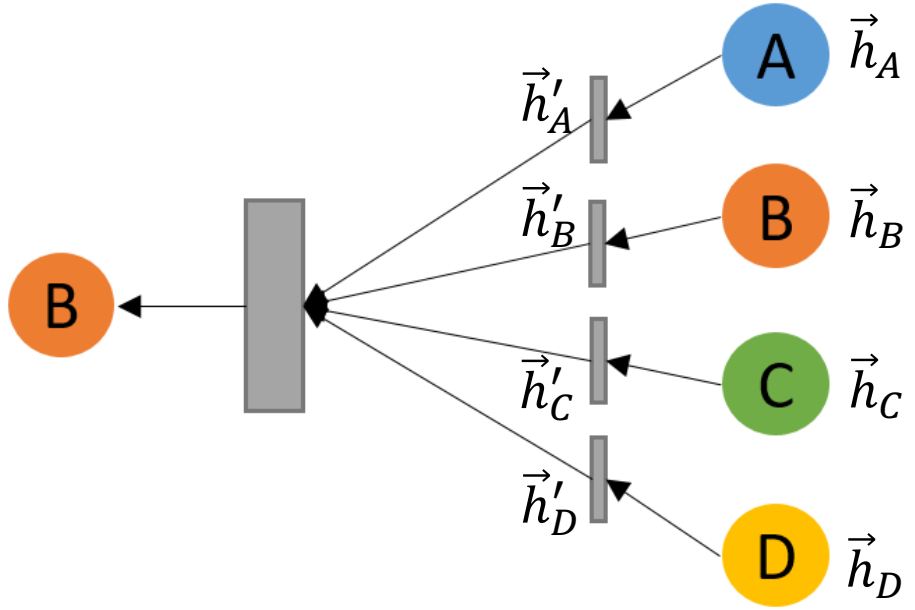
$$\mathbf{h} = \{\vec{h}_1, \vec{h}_2, \ldots, \vec{h}_N\}, \vec{h}_i \in \mathbb{R}^F$$

- 2. Linear transformation.

$$\vec{h}'_i = \mathbf{W}\vec{h}_i$$

$$\mathbf{h}' = \{\vec{h}'_1, \vec{h}'_2, \ldots, \vec{h}'_N\}, \vec{h}'_i \in \mathbb{R}^{F'}$$





Graph Neural Networks (GNNs)

- 1. Start with the original features, e.g., Bag-of-Words.

$$\mathbf{h} = \{\vec{h}_1, \vec{h}_2, \ldots, \vec{h}_N\}, \vec{h}_i \in \mathbb{R}^F$$

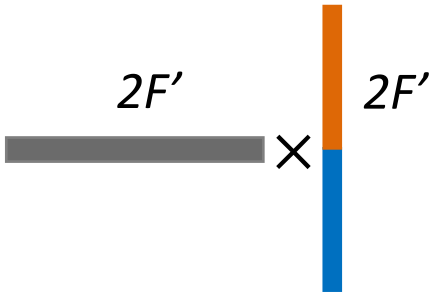
- 2. Linear transformation.

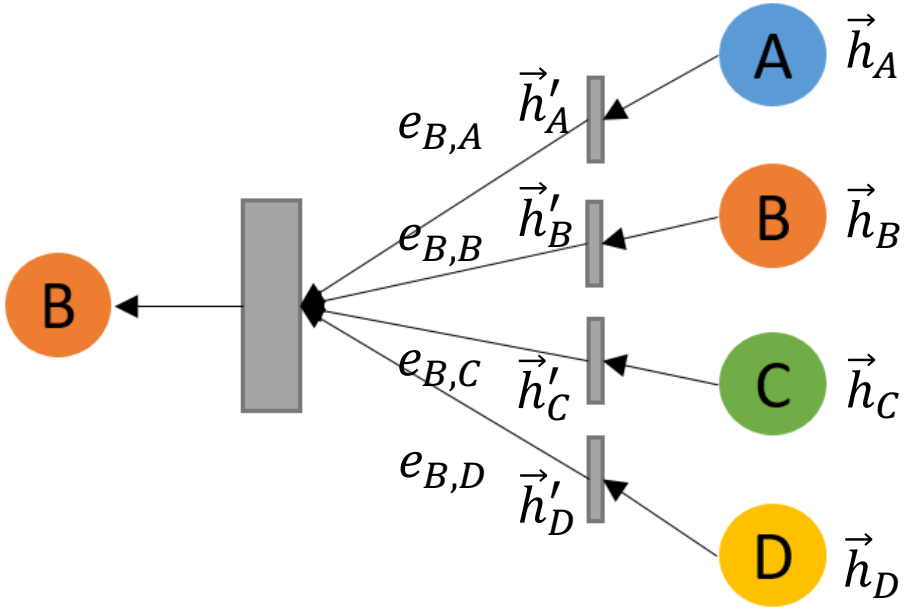
$$\vec{h}'_i = \mathbf{W}\vec{h}_i$$

$$\mathbf{h}' = \{\vec{h}'_1, \vec{h}'_2, \ldots, \vec{h}'_N\}, \vec{h}'_i \in \mathbb{R}^{F'}$$

- 3. Neighbor attention.

$$e_{B,j} = \text{LeakyReLU}(\vec{a}^T[\vec{h}'_B || \vec{h}'_j]), \text{ where } j = A, B, C, D$$





Graph Neural Networks (GNNs)

- 1. Start with the original features, e.g., Bag-of-Words.

$$\mathbf{h} = \{\vec{h}_1, \vec{h}_2, \ldots, \vec{h}_N\}, \vec{h}_i \in \mathbb{R}^F$$

- 2. Linear transformation.

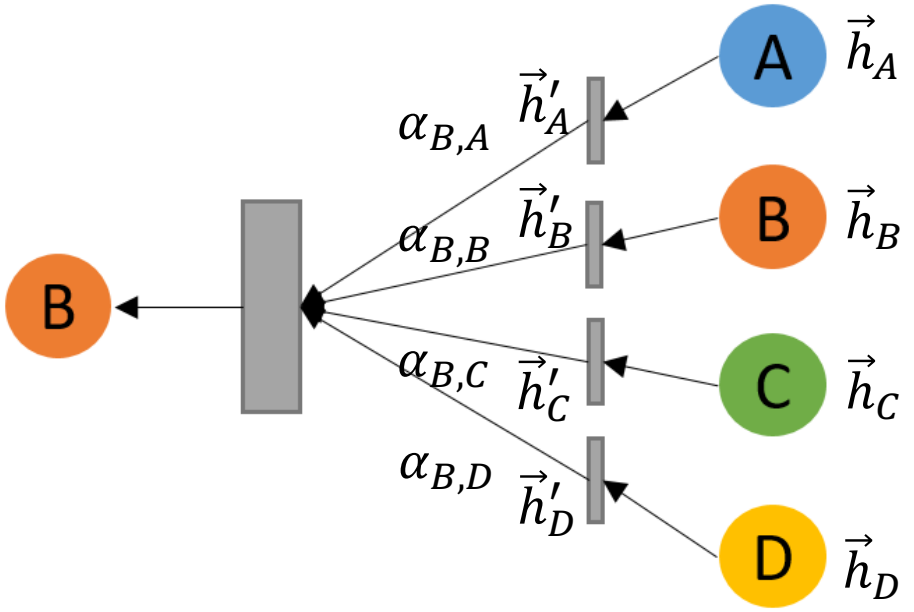
$$\vec{h}'_i = \mathbf{W}\vec{h}_i$$

$$\mathbf{h}' = \{\vec{h}'_1, \vec{h}'_2, \ldots, \vec{h}'_N\}, \vec{h}'_i \in \mathbb{R}^{F'}$$

- 3. Neighbor attention.

$$e_{B,j} = \text{LeakyReLU}(\vec{a}^T[\vec{h}'_B || \vec{h}'_j]), \text{ where } j = A, B, C, D$$

$$\alpha_{B,j} = \text{softmax}(e_{B,j}) = \frac{\exp(e_{B,j})}{\sum_j \exp(e_{B,j})}, \text{ where } j = A, B, C, D.$$



Graph Neural Networks (GNNs)

- 1. Start with the original features, e.g., Bag-of-Words.

$$\mathbf{h} = \{\vec{h}_1, \vec{h}_2, \ldots, \vec{h}_N\}, \vec{h}_i \in \mathbb{R}^F$$

- 2. Linear transformation.

$$\vec{h}'_i = \mathbf{W}\vec{h}_i$$

$$\mathbf{h}' = \{\vec{h}'_1, \vec{h}'_2, \ldots, \vec{h}'_N\}, \vec{h}'_i \in \mathbb{R}^{F'}$$

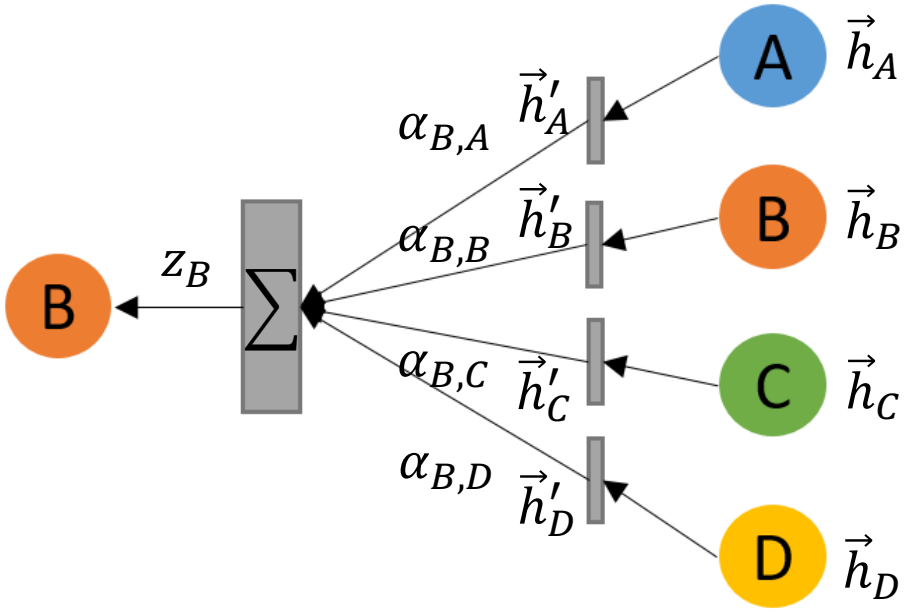
- 3. Neighbor attention.

$$e_{B,j} = \text{LeakyReLU}(\vec{a}^T[\vec{h}'_B || \vec{h}'_j]), \text{ where } j = A, B, C, D$$

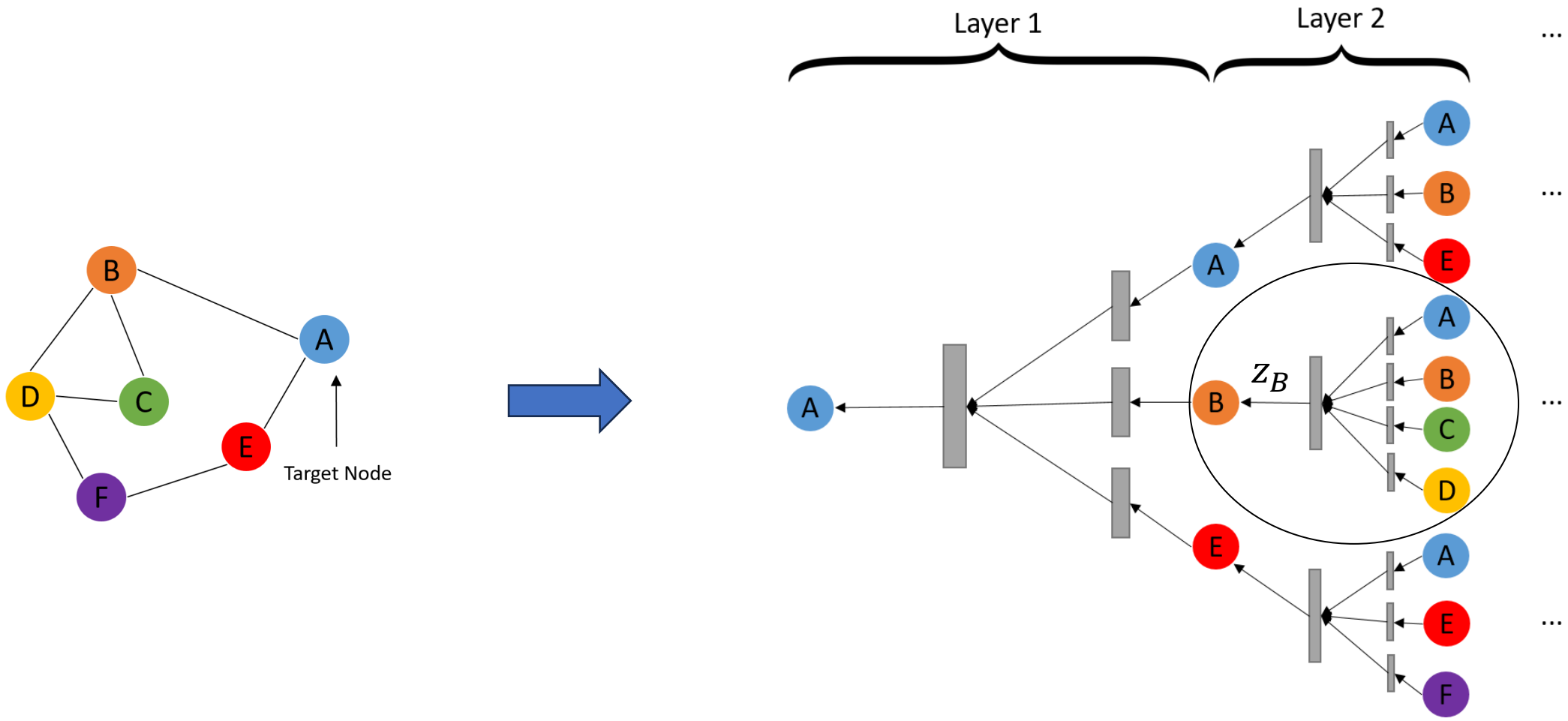
$$\alpha_{B,j} = \text{softmax}(e_{B,j}) = \frac{\exp(e_{B,j})}{\sum_j \exp(e_{B,j})}, \text{ where } j = A, B, C, D.$$

- 4. Neighbor aggregation.

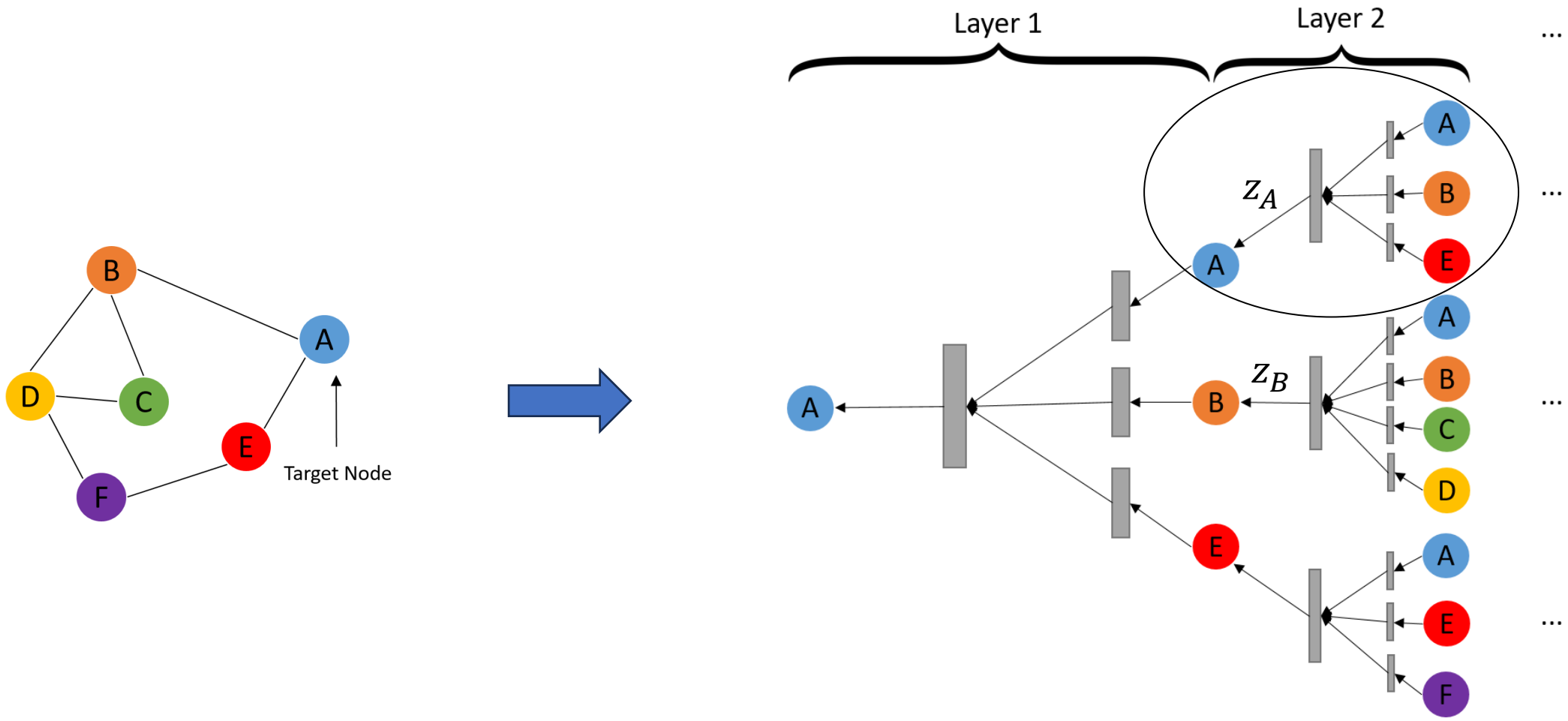
$$z_B = \sigma(\sum_j \alpha_{B,j}\vec{h}'_j), \text{ where } j = A, B, C, D.$$



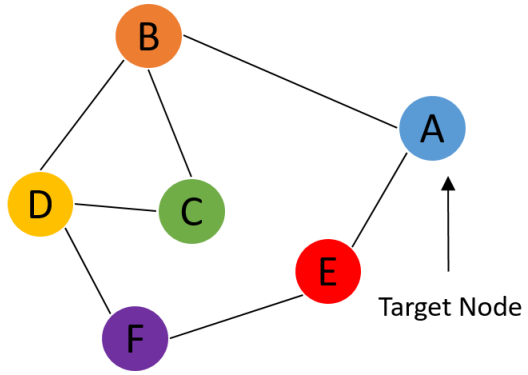
Graph Neural Networks (GNNs)

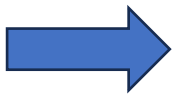


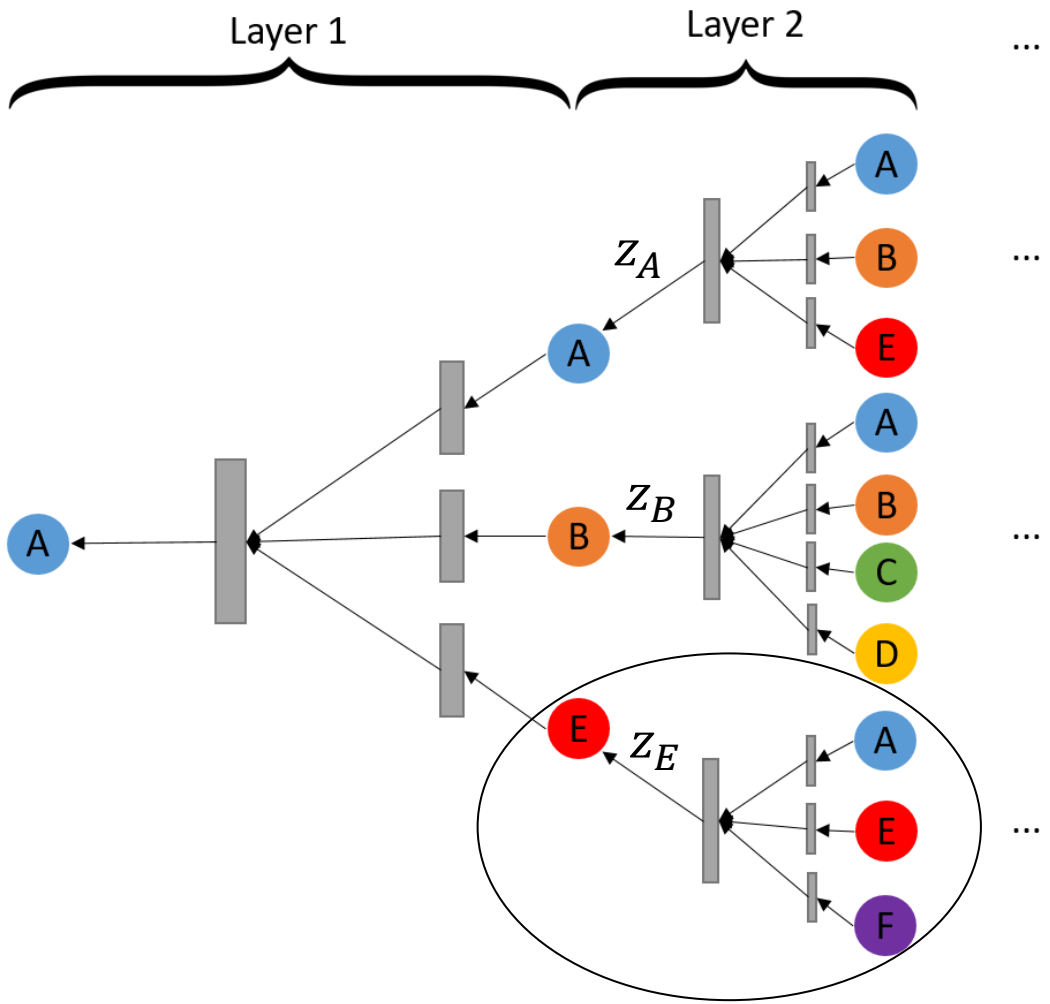
Graph Neural Networks (GNNs)



Graph Neural Networks (GNNs)

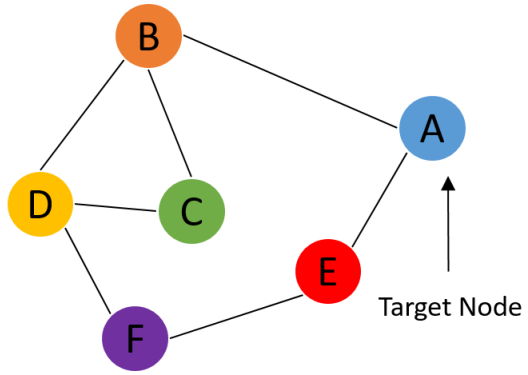


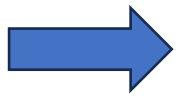


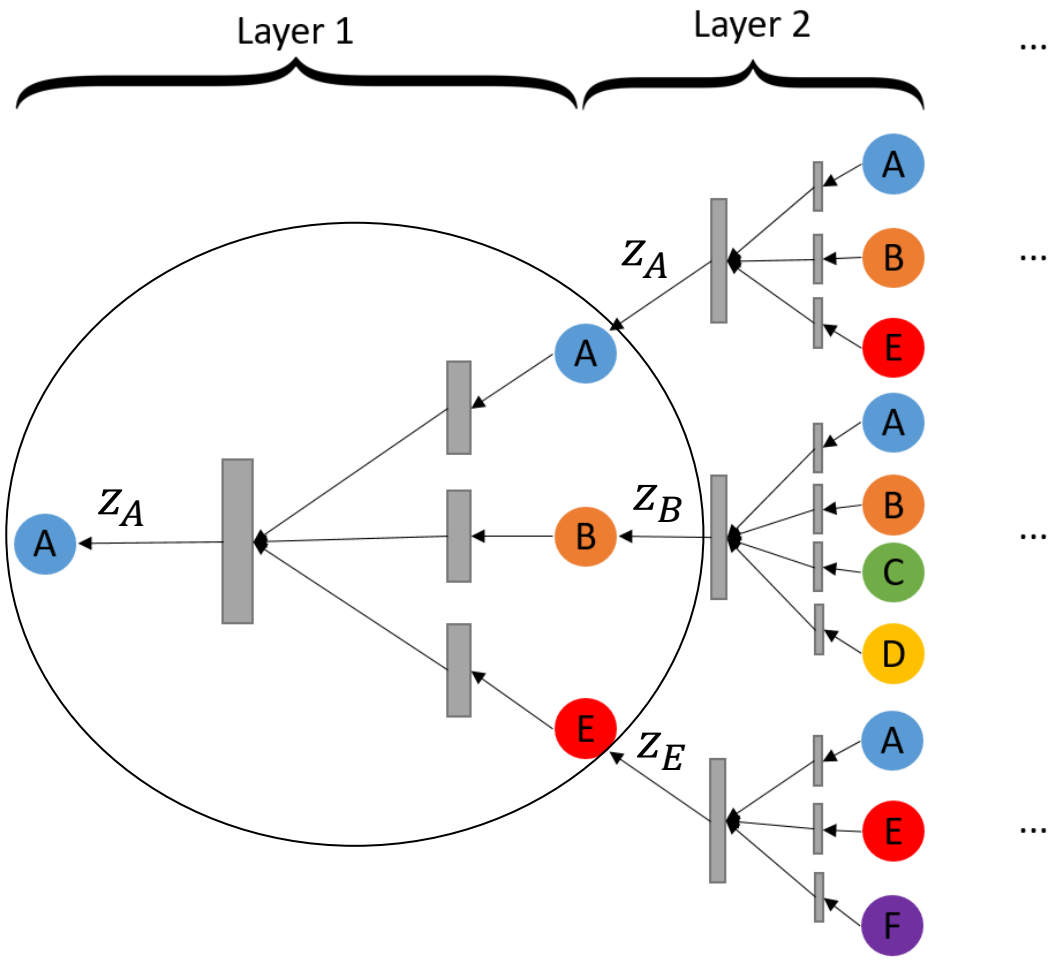


Graph Neural Networks (GNNs)

- 5. Cross-entropy loss for link prediction.







Graph Neural Networks (GNNs)

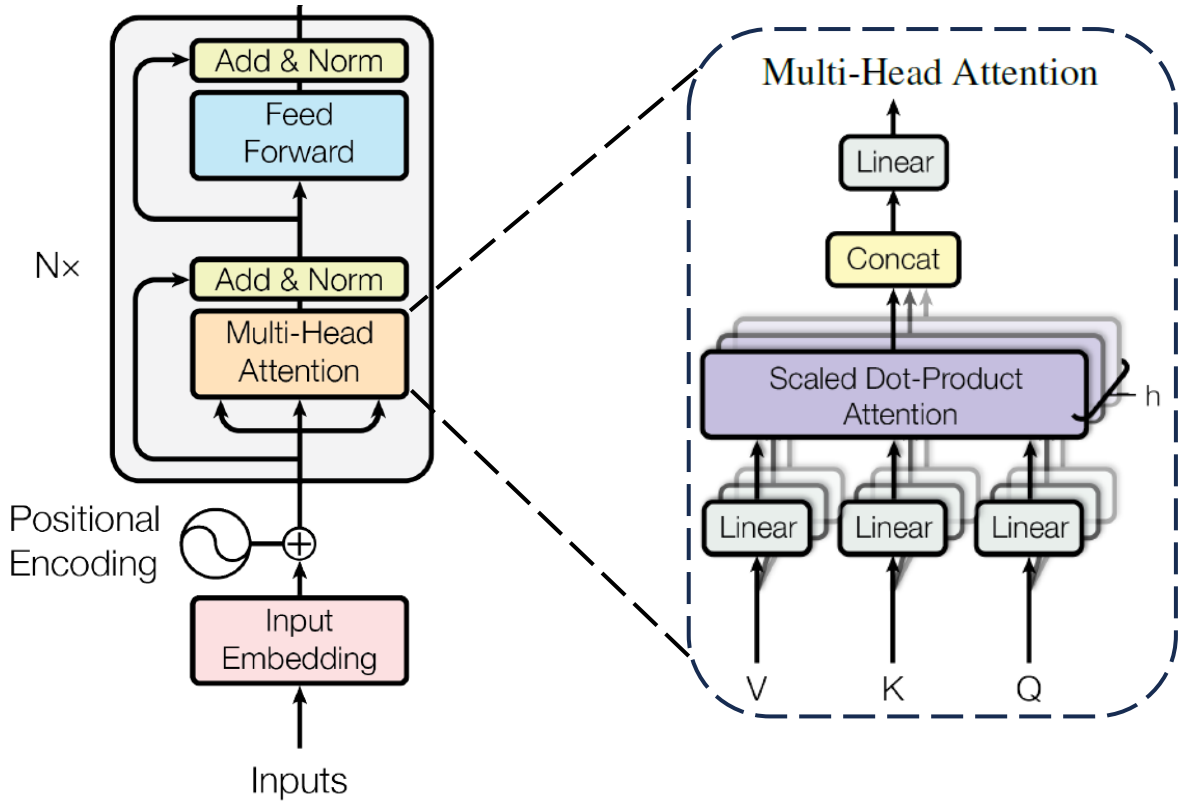
- Graph Neural Networks (GNNs) consider text in documents as general attribute and do not specifically deal with text data. Consequently, they can not capture language representations or linguistic semantics in text corpora

Section 1: Text-Attributed Graph and Preliminaries

- Section 1.1: Formal Definition of Text-Attributed Graph (TAG)
- Section 1.2: Graph Neural Networks (GNNs)
- Section 1.3: Pre-trained Language Models (PLMs)
- Section 1.4: Topic Models (TMs)

Pre-trained Language Models (PLMs)

- Pre-trained Language Models (PLMs) learn contextualized document representations that preserve textual semantics. Below we illustrate Transformer [1].



[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.

Pre-trained Language Models (PLMs)

- 1. Positional encoding.
  - Suppose the input text is “the web conference”.

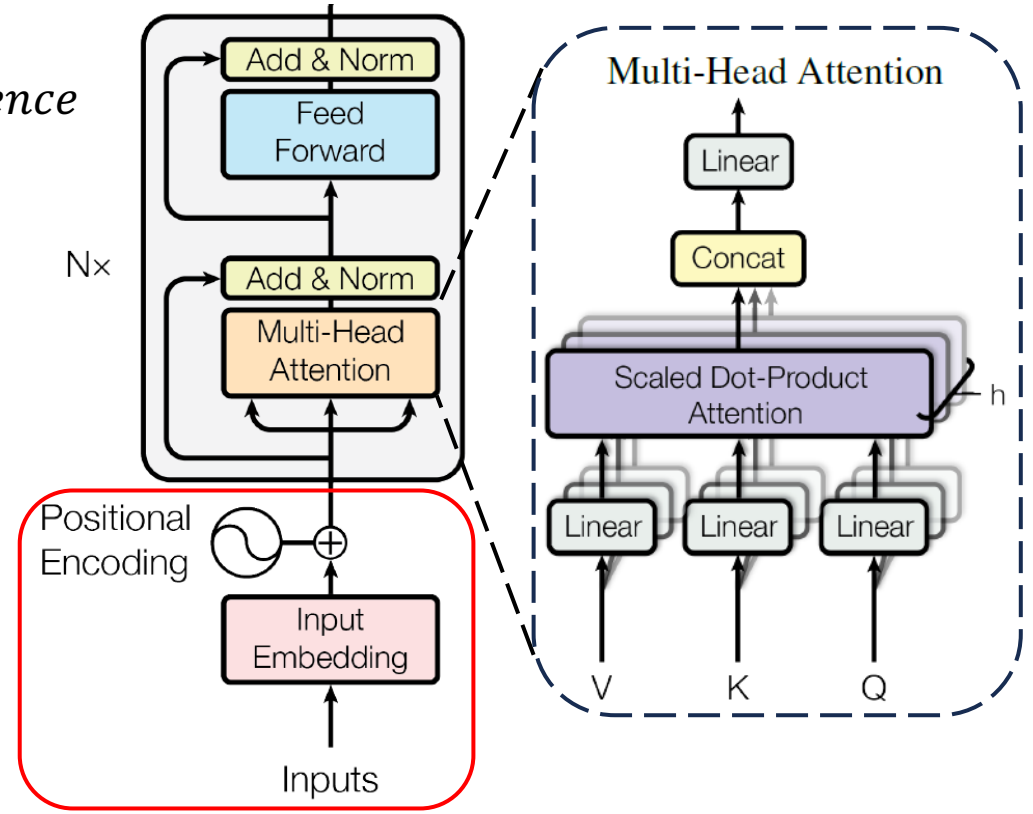
Input embeddings: $\boldsymbol{e}_i = \boldsymbol{w}_i + \boldsymbol{p}_i$, where $i = the, web, conference$

Word embedding Positional encoding:

$$p_{(pos,2i)} = \sin(pos/10000^{2i/\dim})$$

$$p_{(pos,2i+1)} = \cos(pos/10000^{2i/\dim})$$

$$\boldsymbol{p}_{the} = [p_{(0,0)}, p_{(0,1)}, \cdots, p_{(0,\dim-1)}]$$



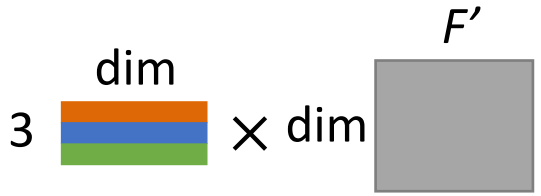
Pre-trained Language Models (PLMs)

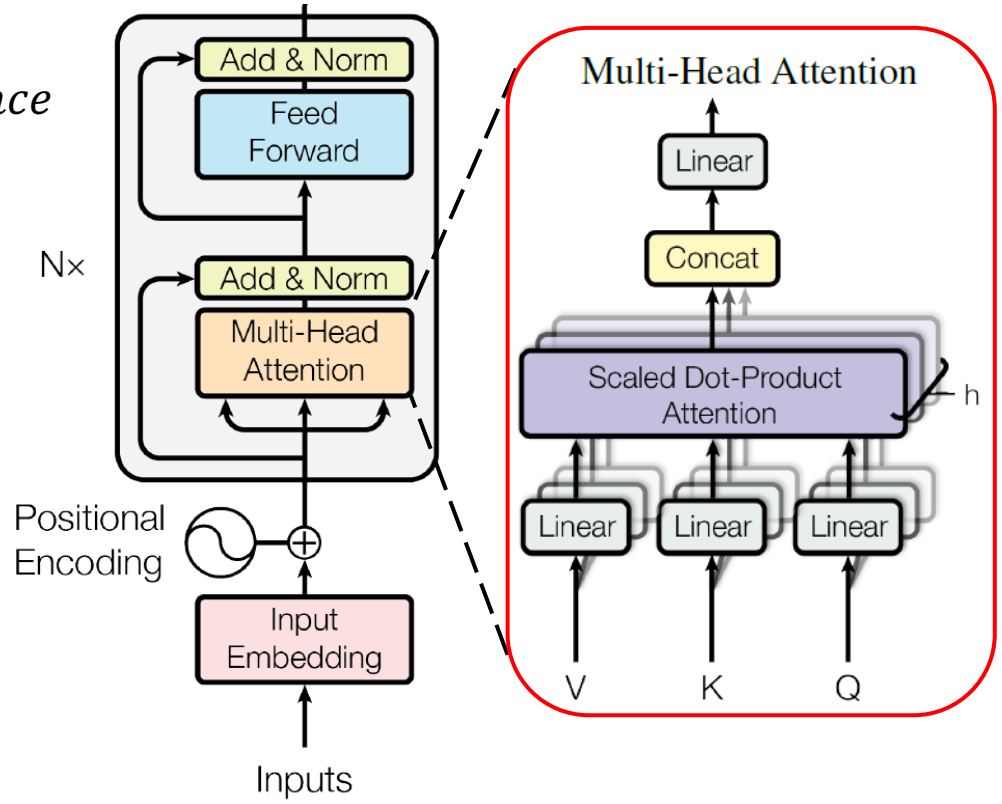
- 1. Positional encoding.
  - Suppose the input text is “the web conference”.

Input embeddings: $\boldsymbol{e}_i = \boldsymbol{w}_i + \boldsymbol{p}_i$, where $i = the, web, conference$

- 2. Multi-head attention.

$\boldsymbol{Q} = \boldsymbol{E}\boldsymbol{W}^Q, \boldsymbol{K} = \boldsymbol{E}\boldsymbol{W}^K, \boldsymbol{V} = \boldsymbol{E}\boldsymbol{W}^V$, where $\boldsymbol{E} = [\boldsymbol{e}_{the}, \boldsymbol{e}_{web}, \boldsymbol{e}_{conference}]$





Pre-trained Language Models (PLMs)

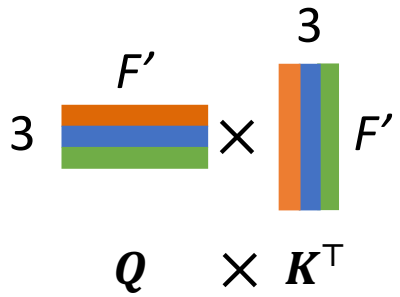
- 1. Positional encoding.
  - Suppose the input text is “the web conference”.

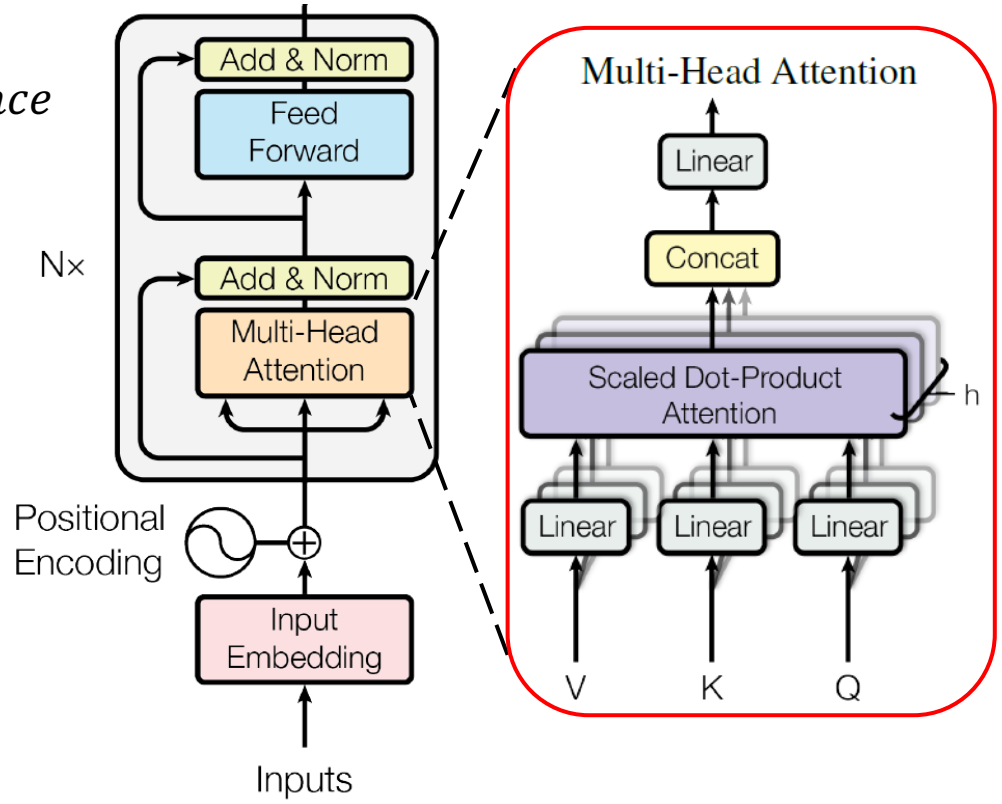
Input embeddings: $\boldsymbol{e}_i = \boldsymbol{w}_i + \boldsymbol{p}_i$, where $i = the, web, conference$

- 2. Multi-head attention.

$\boldsymbol{Q} = \boldsymbol{E}\boldsymbol{W}^Q, \boldsymbol{K} = \boldsymbol{E}\boldsymbol{W}^K, \boldsymbol{V} = \boldsymbol{E}\boldsymbol{W}^V$, where $\boldsymbol{E} = [\boldsymbol{e}_{the}, \boldsymbol{e}_{web}, \boldsymbol{e}_{conference}]$

$$\mathrm{Att}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \mathrm{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{\dim}}\right)\boldsymbol{V}$$





Pre-trained Language Models (PLMs)

- 1. Positional encoding.
  - Suppose the input text is “the web conference”.

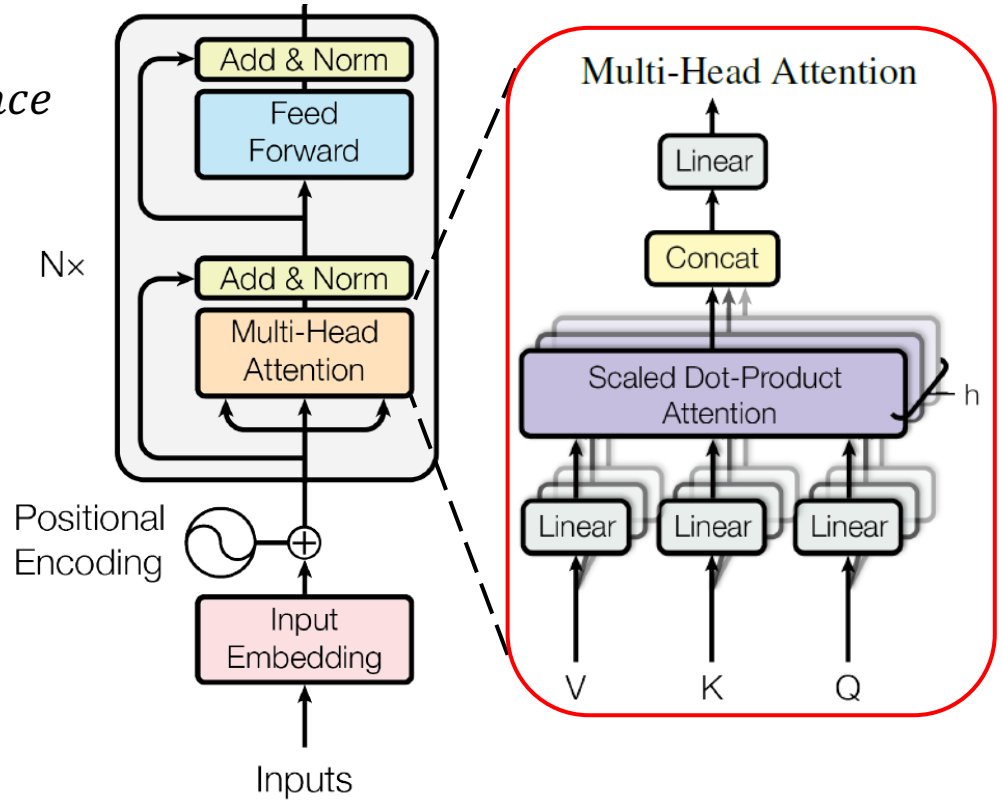
Input embeddings: $\boldsymbol{e}_i = \boldsymbol{w}_i + \boldsymbol{p}_i$, where $i = the, web, conference$

- 2. Multi-head attention.

$\boldsymbol{Q} = \boldsymbol{E}\boldsymbol{W}^Q, \boldsymbol{K} = \boldsymbol{E}\boldsymbol{W}^K, \boldsymbol{V} = \boldsymbol{E}\boldsymbol{W}^V$, where $\boldsymbol{E} = [\boldsymbol{e}_{the}, \boldsymbol{e}_{web}, \boldsymbol{e}_{conference}]$

$$\mathrm{Att}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \mathrm{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{\dim}}\right)\boldsymbol{V}$$

$$\mathrm{MultiHeadAtt}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = [\mathrm{Att}_1(\boldsymbol{Q}_1, \boldsymbol{K}_1, \boldsymbol{V}_1)|| \ldots ||\mathrm{Att}(\boldsymbol{Q}_H, \boldsymbol{K}_H, \boldsymbol{V}_H)]\boldsymbol{W}$$



Pre-trained Language Models (PLMs)

- 1. Positional encoding.
  - Suppose the input text is “the web conference”.

Input embeddings: $\boldsymbol{e}_i = \boldsymbol{w}_i + \boldsymbol{p}_i$, where $i = the, web, conference$

- 2. Multi-head attention.

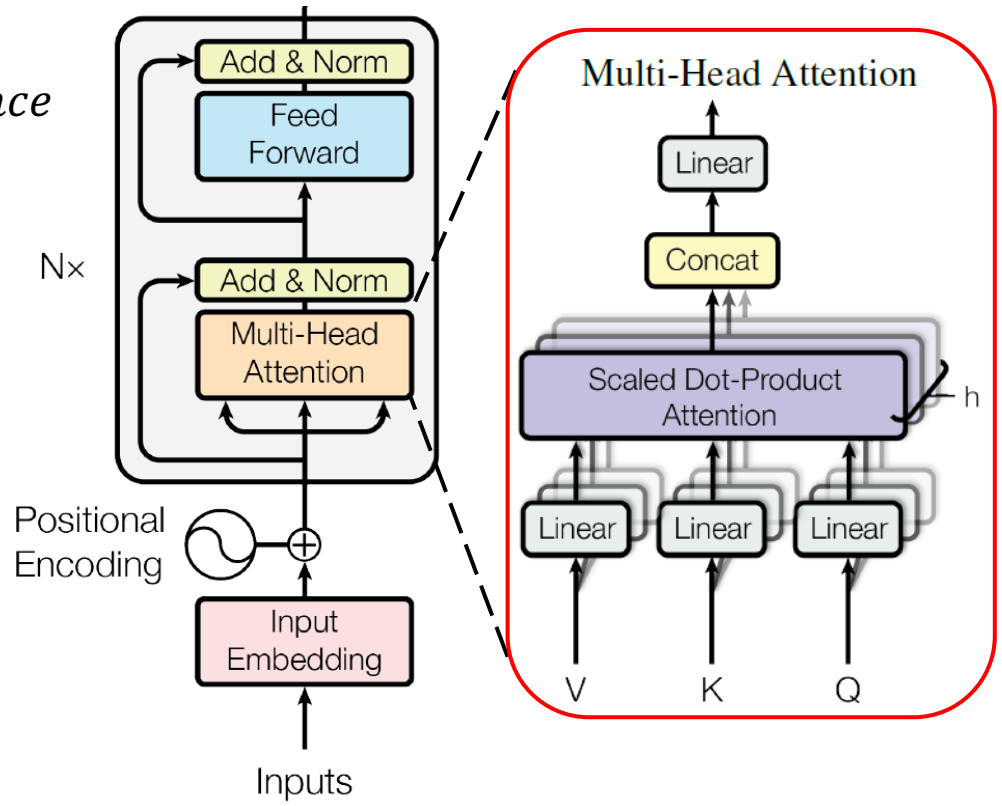
$\boldsymbol{Q} = \boldsymbol{E}\boldsymbol{W}^Q, \boldsymbol{K} = \boldsymbol{E}\boldsymbol{W}^K, \boldsymbol{V} = \boldsymbol{E}\boldsymbol{W}^V$, where $\boldsymbol{E} = [\boldsymbol{e}_{the}, \boldsymbol{e}_{web}, \boldsymbol{e}_{conference}]$

$$\mathrm{Att}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \mathrm{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{\dim}}\right)\boldsymbol{V}$$

$$\mathrm{MultiHeadAtt}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = [\mathrm{Att}_1(\boldsymbol{Q}_1, \boldsymbol{K}_1, \boldsymbol{V}_1)|| \ldots ||\mathrm{Att}(\boldsymbol{Q}_H, \boldsymbol{K}_H, \boldsymbol{V}_H)]\boldsymbol{W}$$

- 3. Addition and normalization.

$$\boldsymbol{E}' = \mathrm{LayerNorm}(\boldsymbol{E} + \mathrm{MultiHeadAtt}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}))$$



Pre-trained Language Models (PLMs)

- 1. Positional encoding.
  - Suppose the input text is “the web conference”.

Input embeddings: $\boldsymbol{e}_i = \boldsymbol{w}_i + \boldsymbol{p}_i$, where $i = the, web, conference$

- 2. Multi-head attention.

$\boldsymbol{Q} = \boldsymbol{E}\boldsymbol{W}^Q, \boldsymbol{K} = \boldsymbol{E}\boldsymbol{W}^K, \boldsymbol{V} = \boldsymbol{E}\boldsymbol{W}^V$, where $\boldsymbol{E} = [\boldsymbol{e}_{the}, \boldsymbol{e}_{web}, \boldsymbol{e}_{conference}]$

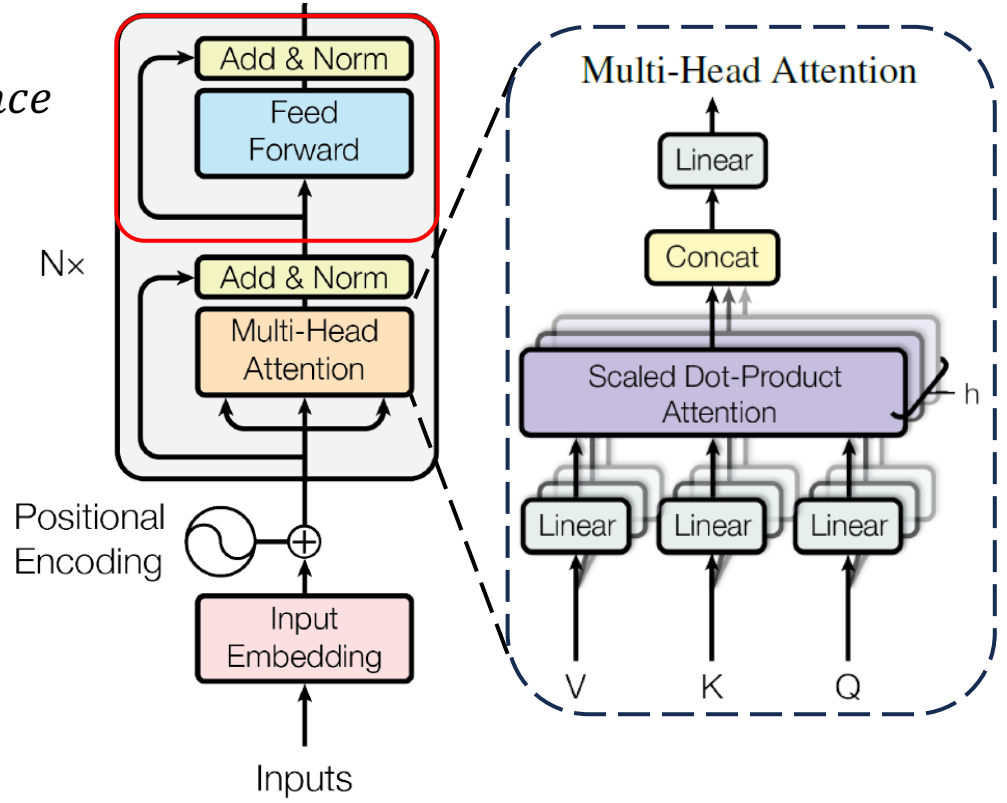
$$\mathrm{Att}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \mathrm{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{\dim}}\right)\boldsymbol{V}$$

$$\mathrm{MultiHeadAtt}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = [\mathrm{Att}_1(\boldsymbol{Q}_1, \boldsymbol{K}_1, \boldsymbol{V}_1)|| \ldots ||\mathrm{Att}(\boldsymbol{Q}_H, \boldsymbol{K}_H, \boldsymbol{V}_H)]\boldsymbol{W}$$

- 3. Addition and normalization.

$$\boldsymbol{E}' = \mathrm{LayerNorm}(\boldsymbol{E} + \mathrm{MultiHeadAtt}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}))$$

- 4. Feed forward. $\boldsymbol{E}'' = \mathrm{LayerNorm}(\boldsymbol{E}' + \mathrm{FFN}(\boldsymbol{E}'))$



Pre-trained Language Models (PLMs)

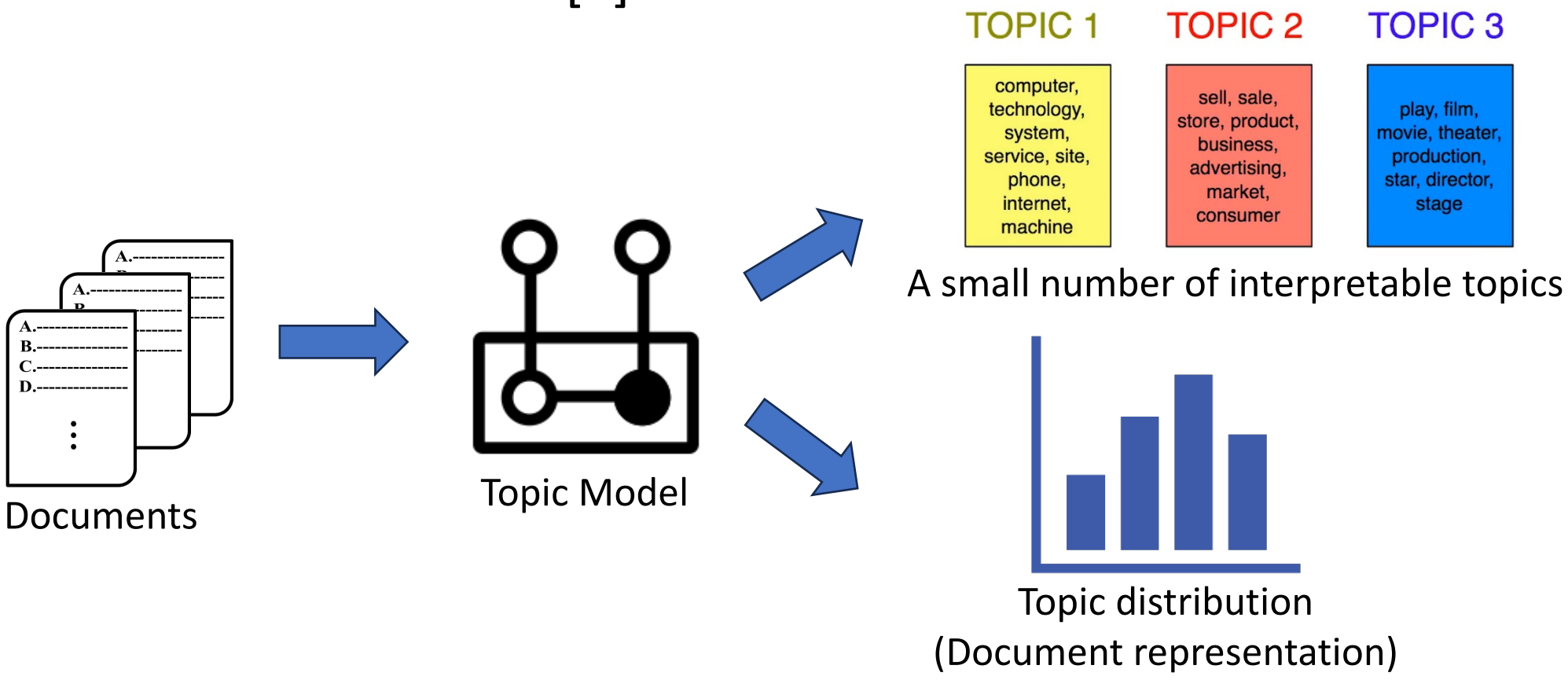
- Pre-trained Language Models (PLMs) mainly deal with text within each individual document only, and ignore the graph adjacency across documents, e.g., citations and hyperlinks.

Section 1: Text-Attributed Graph and Preliminaries

- Section 1.1: Formal Definition of Text-Attributed Graph (TAG)
- Section 1.2: Graph Neural Networks (GNNs)
- Section 1.3: Pre-trained Language Models (PLMs)
- Section 1.4: Topic Models (TMs)

Topic Models (TMs)

- Topic Models (TMs) learn topic distributions as document representations. Below we illustrate NVDM [1].

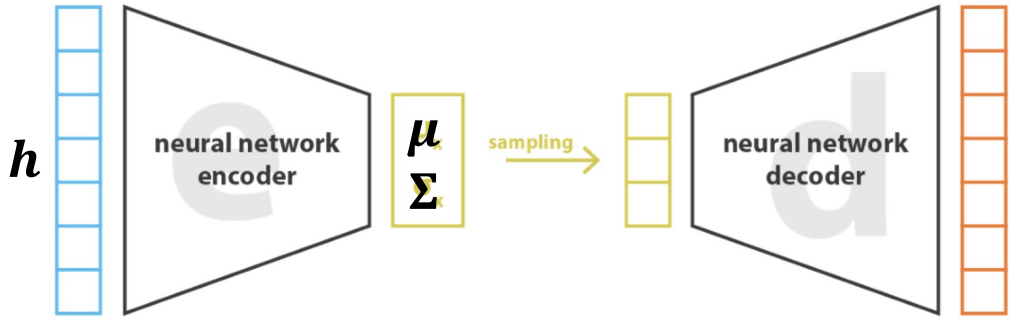


[1] Miao, Y., Yu, L., & Blunsom, P. (2016, June). Neural variational inference for text processing. In International conference on machine learning (pp. 1727-1736). PMLR.

Topic Models (TMs)

- 1. Variational encoding with $\boldsymbol{h}$, i.e., Bag-of-Words.

$$\boldsymbol{\mu} = f_{\mu}(f_{\mathrm{MLP}}(\boldsymbol{h})) \qquad \boldsymbol{\Sigma} = f_{\sigma}(f_{\mathrm{MLP}}(\boldsymbol{h}))$$



Topic Models (TMs)

- 1. Variational encoding with $\boldsymbol{h}$, i.e., Bag-of-Words.

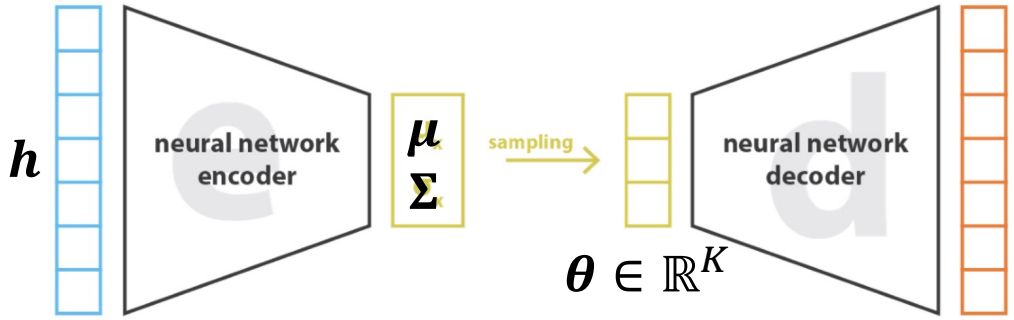
$$\boldsymbol{\mu} = f_\mu(f_{\text{MLP}}(\boldsymbol{h})) \qquad \boldsymbol{\Sigma} = f_\sigma(f_{\text{MLP}}(\boldsymbol{h}))$$

- 2. Reparameterization.

$$\boldsymbol{\theta} = \text{softmax}\left(\boldsymbol{\mu} + \boldsymbol{\Sigma}^{\frac{1}{2}}\boldsymbol{\epsilon}\right) \in \mathbb{R}^K, \quad \text{where} \quad \boldsymbol{\epsilon} \sim \text{Gaussian}(\boldsymbol{0}, \boldsymbol{I})$$

Topic distribution

K is number of topics



Topic Models (TMs)

- 1. Variational encoding with $\boldsymbol{h}$, i.e., Bag-of-Words.

$$\boldsymbol{\mu} = f_\mu(f_{\text{MLP}}(\boldsymbol{h})) \qquad \boldsymbol{\Sigma} = f_\sigma(f_{\text{MLP}}(\boldsymbol{h}))$$

- 2. Reparameterization.

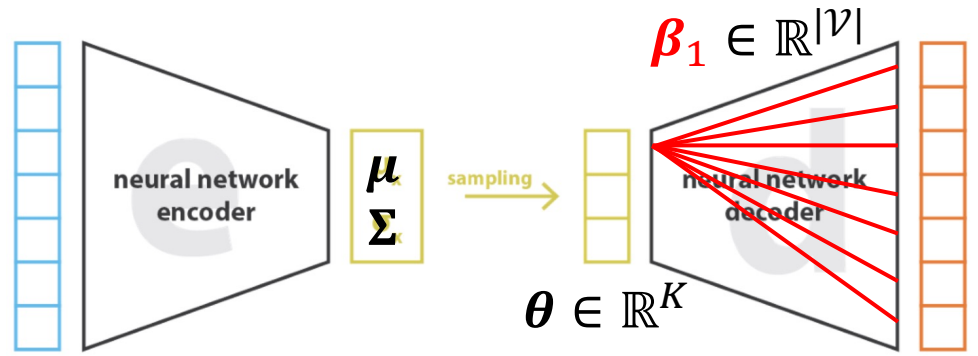
$$\boldsymbol{\theta} = \text{softmax}\left(\boldsymbol{\mu} + \boldsymbol{\Sigma}^{\frac{1}{2}}\boldsymbol{\epsilon}\right) \in \mathbb{R}^K, \quad \text{where} \quad \boldsymbol{\epsilon} \sim \text{Gaussian}(\boldsymbol{0}, \boldsymbol{I})$$

- 3. Topic modeling decoding.

$$\boldsymbol{\beta}_k = \text{softmax}(\boldsymbol{W}_k) \in \mathbb{R}^{|\mathcal{V}|}, \text{ where } \boldsymbol{W}_k \text{ is randomly initialized}$$

Topic-word distribution of one topic k

$\mathcal{V}$ is vocabulary



Topic Models (TMs)

- 1. Variational encoding with $\boldsymbol{h}$, i.e., Bag-of-Words.

  $\boldsymbol{\mu} = f_\mu(f_{\mathrm{MLP}}(\boldsymbol{h}))$ $\quad$ $\boldsymbol{\Sigma} = f_\sigma(f_{\mathrm{MLP}}(\boldsymbol{h}))$

- 2. Reparameterization.

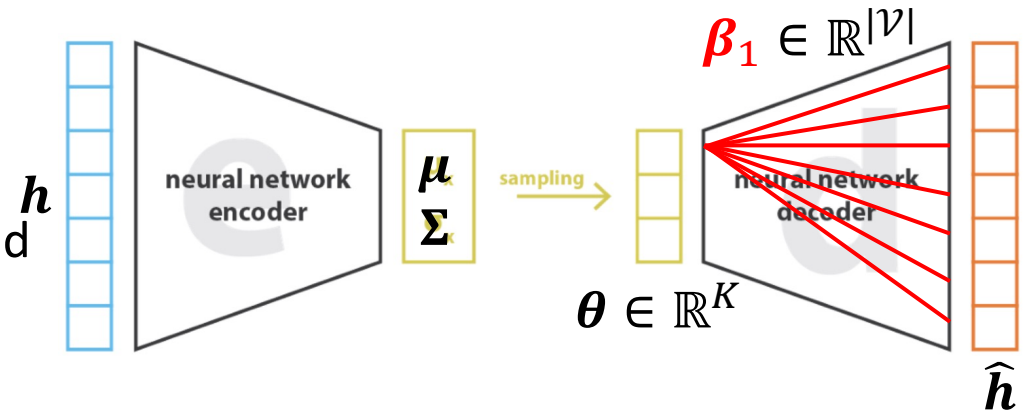
  $\boldsymbol{\theta} = \mathrm{softmax}\left(\boldsymbol{\mu} + \boldsymbol{\Sigma}^{\frac{1}{2}}\boldsymbol{\epsilon}\right) \in \mathbb{R}^K$, where $\boldsymbol{\epsilon} \sim \mathrm{Gaussian}(\boldsymbol{0}, \boldsymbol{I})$

- 3. Topic modeling decoding.

  $\boldsymbol{\beta}_k = \mathrm{softmax}(\boldsymbol{W}_k) \in \mathbb{R}^{|\mathcal{V}|}$, where $\boldsymbol{W}_k$ is randomly initialized

  $\boldsymbol{\beta} = [\boldsymbol{\beta}_1; \boldsymbol{\beta}_2; \ldots, \boldsymbol{\beta}_K] \in \mathbb{R}^{K \times |\mathcal{V}|}$ is topic-word distribution

  $\widehat{\boldsymbol{h}} = \boldsymbol{\theta} \times \boldsymbol{\beta}$



Topic Models (TMs)

- 1. Variational encoding with $\boldsymbol{h}$, i.e., Bag-of-Words.

  $\boldsymbol{\mu} = f_\mu(f_{\text{MLP}}(\boldsymbol{h}))$ $\quad$ $\boldsymbol{\Sigma} = f_\sigma(f_{\text{MLP}}(\boldsymbol{h}))$

- 2. Reparameterization.

  $\boldsymbol{\theta} = \text{softmax}\left(\boldsymbol{\mu} + \boldsymbol{\Sigma}^{\frac{1}{2}}\boldsymbol{\epsilon}\right) \in \mathbb{R}^K$, where $\boldsymbol{\epsilon} \sim \text{Gaussian}(\boldsymbol{0}, \boldsymbol{I})$

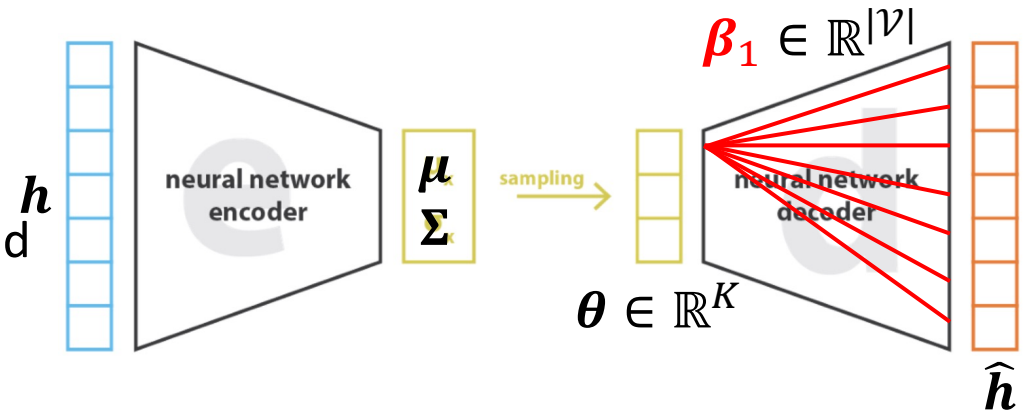
- 3. Topic modeling decoding.

  $\boldsymbol{\beta}_k = \text{softmax}(\boldsymbol{W}_k) \in \mathbb{R}^{|\mathcal{V}|}$, where $\boldsymbol{W}_k$ is randomly initialized

  $\boldsymbol{\beta} = [\boldsymbol{\beta}_1; \boldsymbol{\beta}_2; \dots, \boldsymbol{\beta}_K] \in \mathbb{R}^{K \times |\mathcal{V}|}$ is topic-word distribution

  $\widehat{\boldsymbol{h}} = \boldsymbol{\theta} \times \boldsymbol{\beta}$

- 4. Cross-entropy loss between $\boldsymbol{h}$ and $\widehat{\boldsymbol{h}}$.



Yale

Q & A

Yale University

SMU SINGAPORE MANAGEMENT UNIVERSITY

School of Computing and Information Systems

Text-Attributed Graph Representation Learning: Methods, Applications, and Challenges Section 2 (45 min)

Delvin Ce Zhang[1], Menglin Yang[1], Rex Ying[1], and Hady W. Lauw[2]

[1]Yale University, [2]Singapore Management University

Tutorial Outline

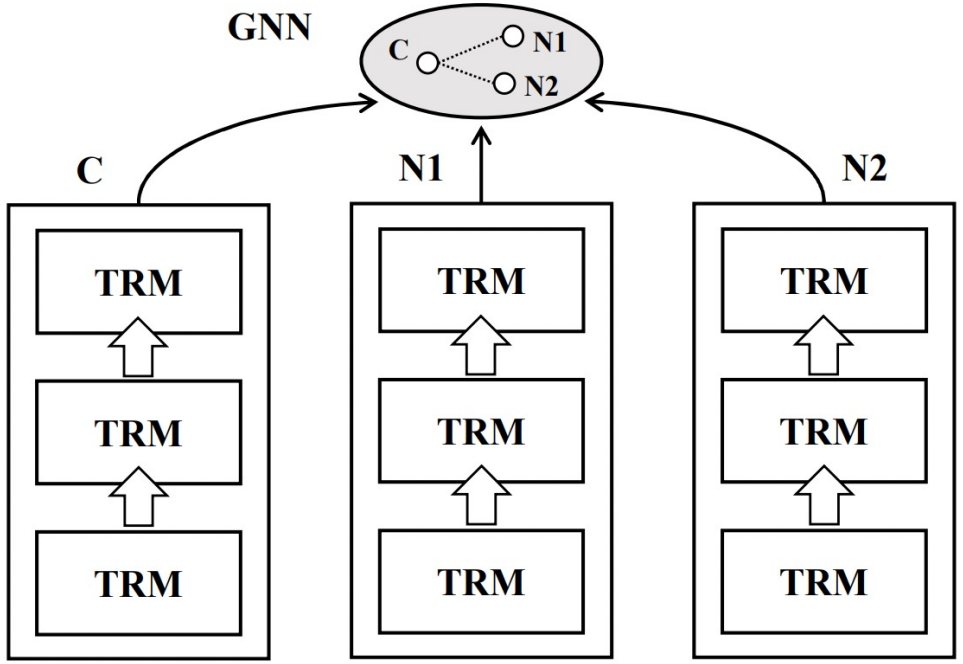
- Section 1: Text-Attributed Graph and Preliminaries (30 min)
- Section 2: PLM-based Text-Attributed Graph Models (45 min)
- Section 3: TM-based Text-Attributed Graph Models (45 min)
- Section 4: Applications, Challenges, and Future Directions (45 min)
- Summary and Q&A

Section 2: PLM-based Text-Attributed Graph Models

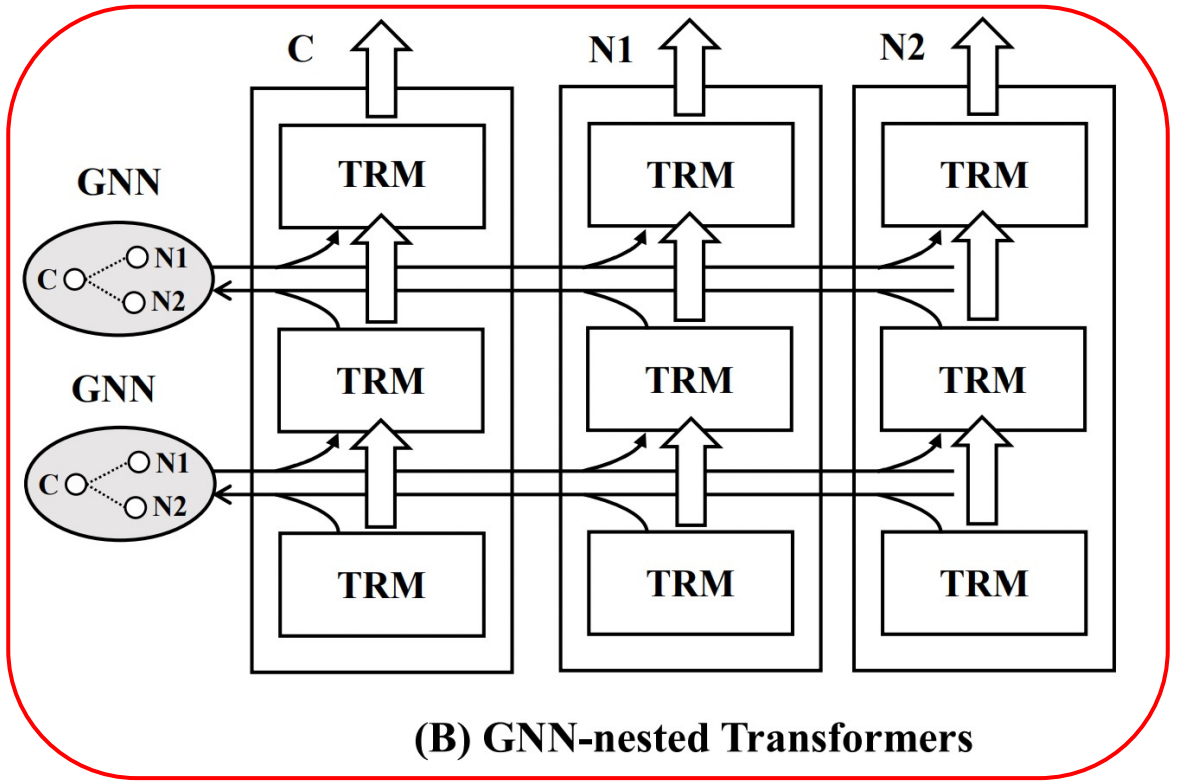
- Section 2.1: PLMs for Static Text-Attributed Graph
- Section 2.2: PLMs for Heterogeneous Text-Attributed Graph
- Section 2.3: Textual-edge Text-Attributed Graph
- Section 2.4: Pre-training Strategy

PLMs for Static Text-Attributed Graph

- **Static Text-Attributed Graph (TAG) :** We observe the whole TAG at once.
- Existing works are split into **cascaded architecture** and **nested architecture**.



(A) Cascaded Transformers-GNN



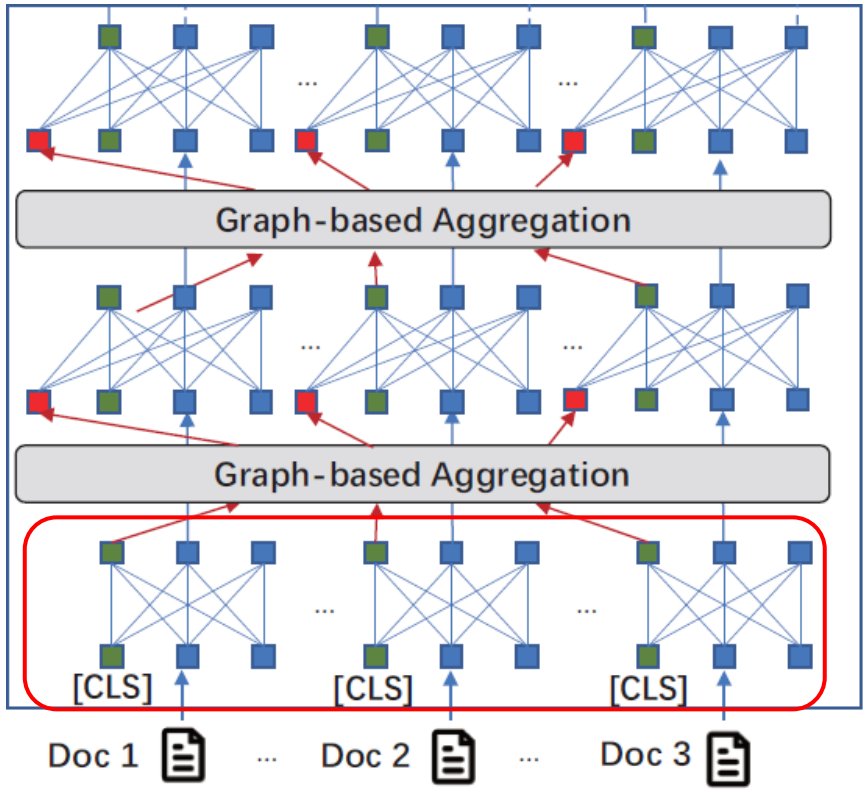
(B) GNN-nested Transformers

Focus of this section

PLMs for Static Text-Attributed Graph

- We present GraphFormer [1] for the static scenario. Below figure is from [2].
- 1. Feature initialization.

$$\boldsymbol{H}_i^1 = \text{Transformer}^{l=1}(\boldsymbol{d}_i), \quad \text{where} \quad \boldsymbol{H}_i^1 = \{\boldsymbol{h}_{i,CLS}^1, \boldsymbol{h}_{i,w_1}^1, \dots, \boldsymbol{h}_{i,w_d}^1\}$$



[1] Yang, J., Liu, Z., Xiao, S., Li, C., Lian, D., Agrawal, S., ... & Xie, X. (2021). Graphformers: Gnn-nested transformers for representation learning on textual graph. Advances in Neural Information Processing Systems, 34, 28798-28810.
[2] Jin, B., Zhang, W., Zhang, Y., Meng, Y., Zhang, X., Zhu, Q., & Han, J. (2023). PATTON: Language Model Pretraining on Text-Rich Networks. In 61st Annual Meeting of the Association for Computational Linguistics, ACL 2023 (pp. 7005-7020). Association for Computational Linguistics (ACL).

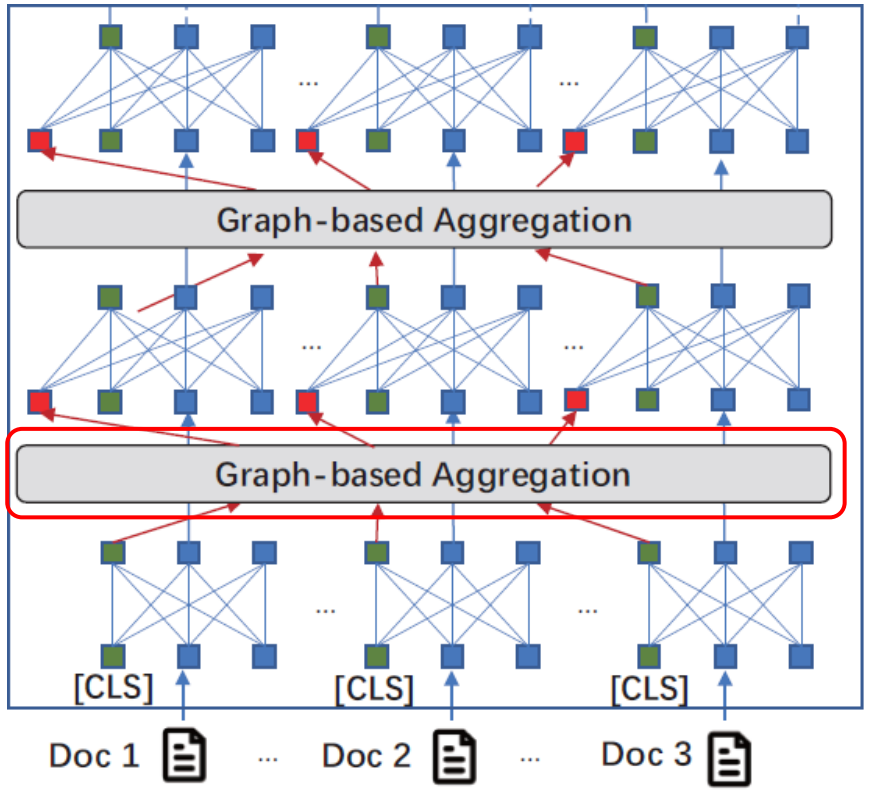
PLMs for Static Text-Attributed Graph

- We present GraphFormer [1] for the static scenario. Below figure is from [2].
- 1. Feature initialization.

$$\boldsymbol{H}_i^1 = \text{Transformer}^{l=1}(\boldsymbol{d}_i), \quad \text{where} \quad \boldsymbol{H}_i^1 = \{\boldsymbol{h}_{i,CLS}^1, \boldsymbol{h}_{i,w_1}^1, \dots, \boldsymbol{h}_{i,w_d}^1\}$$

- 2. Graph aggregation.

$$\boldsymbol{z}_i^1 = \text{GNN}\left(\boldsymbol{h}_{i,CLS}^1, \{\boldsymbol{h}_{j,CLS}^1\}_{j \in \mathcal{N}(i)}\right)$$



[1] Yang, J., Liu, Z., Xiao, S., Li, C., Lian, D., Agrawal, S., ... & Xie, X. (2021). Graphformers: Gnn-nested transformers for representation learning on textual graph. Advances in Neural Information Processing Systems, 34, 28798-28810.
[2] Jin, B., Zhang, W., Zhang, Y., Meng, Y., Zhang, X., Zhu, Q., & Han, J. (2023). PATTON: Language Model Pretraining on Text-Rich Networks. In 61st Annual Meeting of the Association for Computational Linguistics, ACL 2023 (pp. 7005-7020). Association for Computational Linguistics (ACL).

PLMs for Static Text-Attributed Graph

- We present GraphFormer [1] for the static scenario. Below figure is from [2].
- 1. Feature initialization.

$$\boldsymbol{H}_i^1 = \text{Transformer}^{l=1}(\boldsymbol{d}_i), \quad \text{where} \quad \boldsymbol{H}_i^1 = \{\boldsymbol{h}_{i,CLS}^1, \boldsymbol{h}_{i,w_1}^1, \dots, \boldsymbol{h}_{i,w_d}^1\}$$

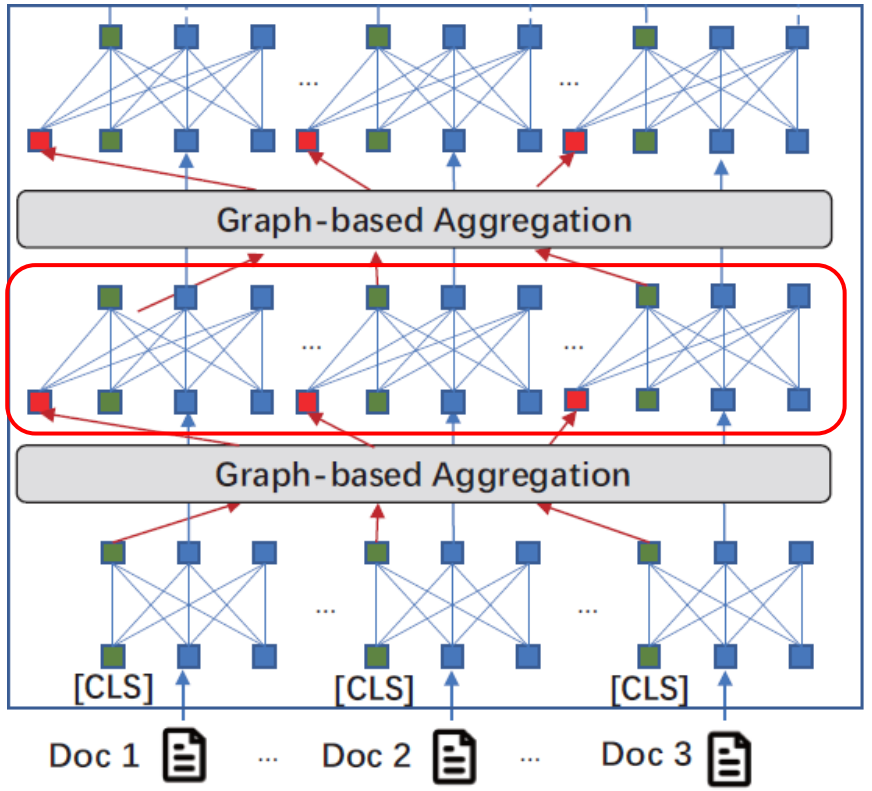
- 2. Graph aggregation.

$$\boldsymbol{z}_i^1 = \text{GNN}\left(\boldsymbol{h}_{i,CLS}^1, \{\boldsymbol{h}_{j,CLS}^1\}_{j \in \mathcal{N}(i)}\right)$$

- 3. Concatenate graph embedding with tokens.

$$\widehat{\boldsymbol{H}}_i^1 = \text{Concat}(\boldsymbol{z}_i^1, \boldsymbol{H}_i^1) \qquad \boldsymbol{H}_i^2 = \text{Transformer}^{l=2}(\widehat{\boldsymbol{H}}_i^1)$$

- Due to contextualization, $\boldsymbol{H}_i^2$ preserves graph.

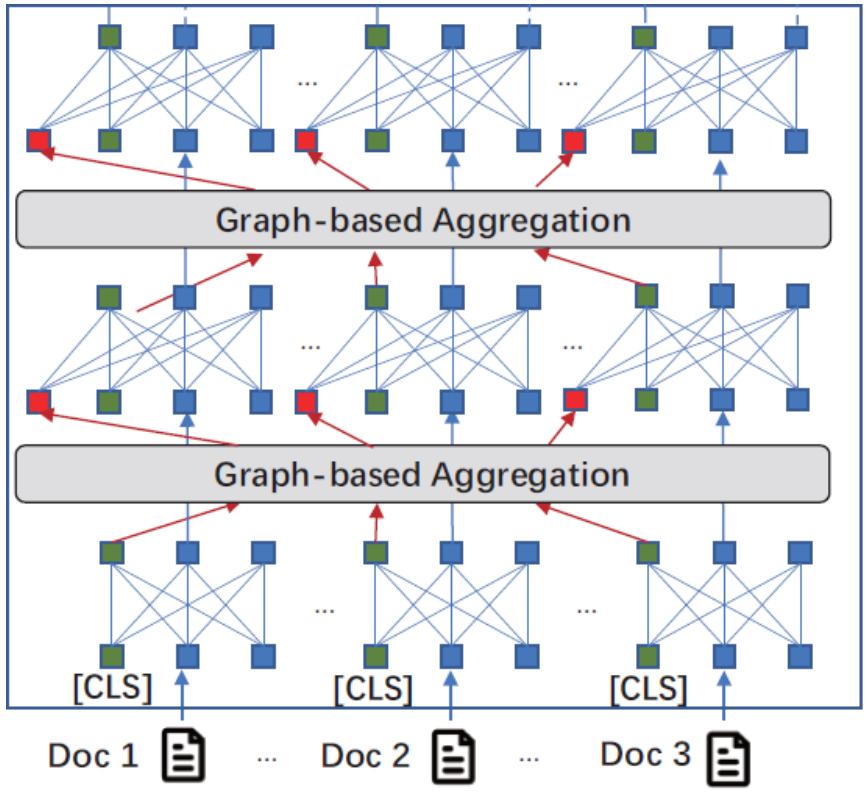


[1] Yang, J., Liu, Z., Xiao, S., Li, C., Lian, D., Agrawal, S., ... & Xie, X. (2021). Graphformers: Gnn-nested transformers for representation learning on textual graph. Advances in Neural Information Processing Systems, 34, 28798-28810.
[2] Jin, B., Zhang, W., Zhang, Y., Meng, Y., Zhang, X., Zhu, Q., & Han, J. (2023). PATTON: Language Model Pretraining on Text-Rich Networks. In 61st Annual Meeting of the Association for Computational Linguistics, ACL 2023 (pp. 7005-7020). Association for Computational Linguistics (ACL).

PLMs for Static Text-Attributed Graph

- 4. So far, we finish from layer 1 to 2. We repeat this process for L layers.

$$\boldsymbol{H}_i^L = \{\boldsymbol{h}_{i,CLS}^L, \boldsymbol{h}_{i,w_1}^L, \dots, \boldsymbol{h}_{i,w_d}^L\}$$



PLMs for Static Text-Attributed Graph

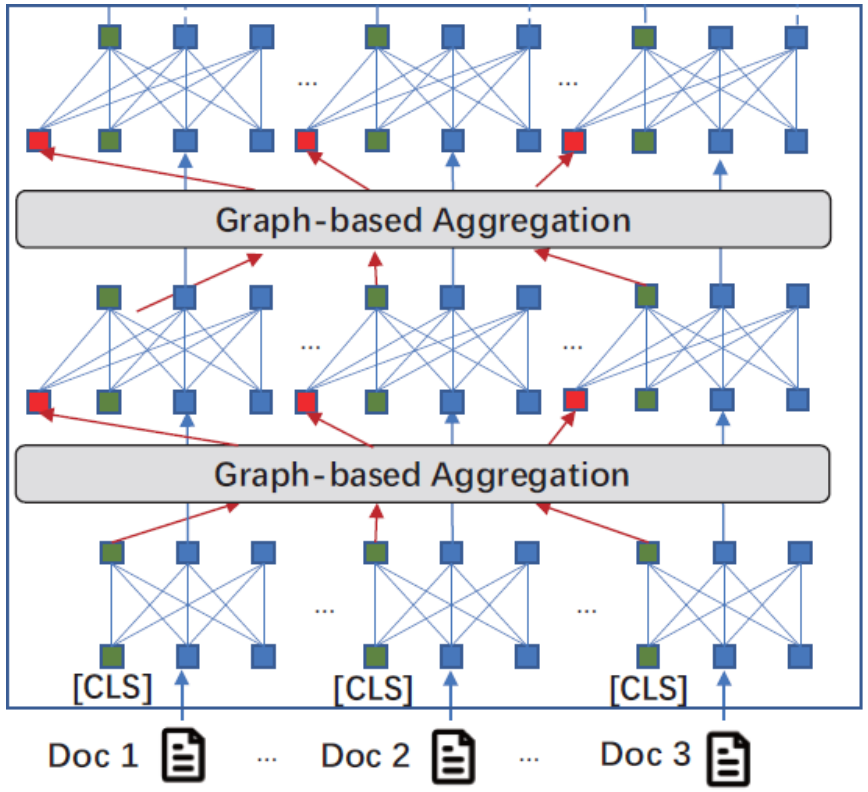
- 4. So far, we finish from layer 1 to 2. We repeat this process for L layers.

$$\boldsymbol{H}_i^L = \{\boldsymbol{h}_{i,CLS}^L, \boldsymbol{h}_{i,w_1}^L, \dots, \boldsymbol{h}_{i,w_d}^L\}$$

- 5. Contrastive loss.

$$\mathcal{L} = -\log \frac{\exp(< \boldsymbol{h}_{i,CLS}^L, \boldsymbol{h}_{j,CLS}^L >)}{\exp(< \boldsymbol{h}_{i,CLS}^L, \boldsymbol{h}_{j,CLS}^L >) + \sum_{b \in B \setminus \mathrm{j}} \exp(< \boldsymbol{h}_{i,CLS}^L, \boldsymbol{h}_{b,CLS}^L >)}$$

Documents in a minibatch B



PLMs for Static Text-Attributed Graph

- **Experiments**
- 1. Datasets

	Product	DBLP	Wiki
#Item	5,643,688	4,894,081	4,818,679
#N	4.71	9.31	8.86
#Train	22,146,934	3,009,506	7,145,834
#Valid	30,000	60,000	66,167
#Test	306,742	100,000	100,000

PLMs for Static Text-Attributed Graph

- **Experiments**
- 2. Link prediction
  - One query document is provided with 1 positive neighbor and 299 negative samples.

Models w/o graph

Models with cascaded architecture

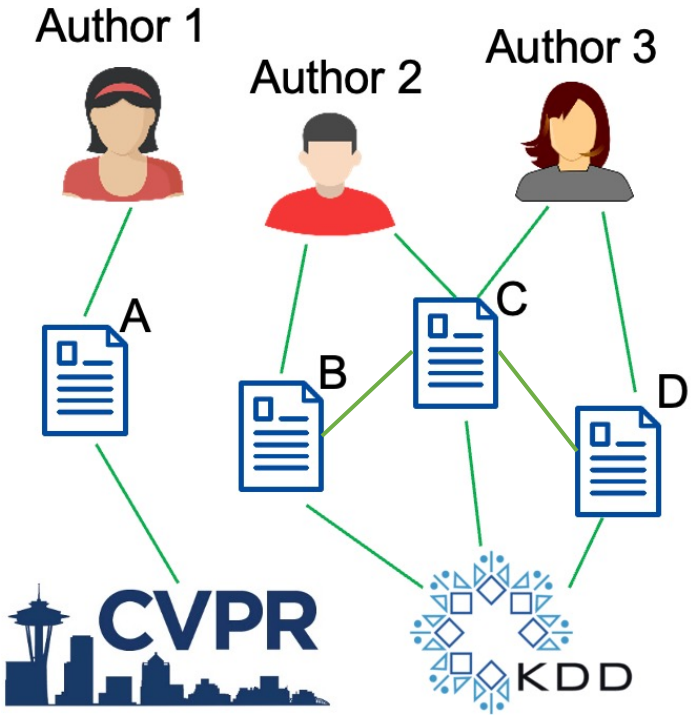
	Product			DBLP			Wiki		
Methods	P@1	NDCG	MRR	P@1	NDCG	MRR	P@1	NDCG	MRR
PLM	0.6563	0.7911	0.7344	0.5673	0.7484	0.6777	0.3466	0.5799	0.4712
TNVE	0.4618	0.6204	0.5364	0.2978	0.5295	0.4163	0.1786	0.4274	0.2933
IFTN	0.5233	0.6740	0.5982	0.3691	0.5798	0.4773	0.1838	0.4276	0.2945
PLM+GAT	0.7540	0.8637	0.8232	0.6633	0.8204	0.7667	0.3006	0.5430	0.4270
PLM+Max	0.7570	0.8678	0.8280	0.6934	0.8386	0.7900	0.3712	0.6071	0.5022
PLM+Mean	0.7550	0.8671	0.8271	0.6896	0.8359	0.7866	0.3664	0.6037	0.4980
PLM+Att	0.7513	0.8652	0.8246	0.6910	0.8366	0.7875	0.3709	0.6067	0.5018
GraphFormers	**0.7786**	**0.8793**	**0.8430**	**0.7267**	**0.8565**	**0.8133**	**0.3952**	**0.6230**	**0.5220**

Section 2: PLM-based Text-Attributed Graph Models

- Section 2.1: PLMs for Static Text-Attributed Graph
- Section 2.2: PLMs for Heterogeneous Text-Attributed Graph
- Section 2.3: Textual-edge Text-Attributed Graph
- Section 2.4: Pre-training Strategy

PLMs for Heterogeneous Text-Attributed Graph

- Documents are associated with meta-data, e.g., authors and venues.
  - Papers written by the same authors discuss similar research topics;
  - News articles edited by the same journalists report similar events.
- Documents + their meta-data → heterogeneous graph.
- Challenges:
  - Previous works do not model meta-data;
  - Some meta-data, such as authors, do not have texts.
- Here we specifically present Heterformer [1].



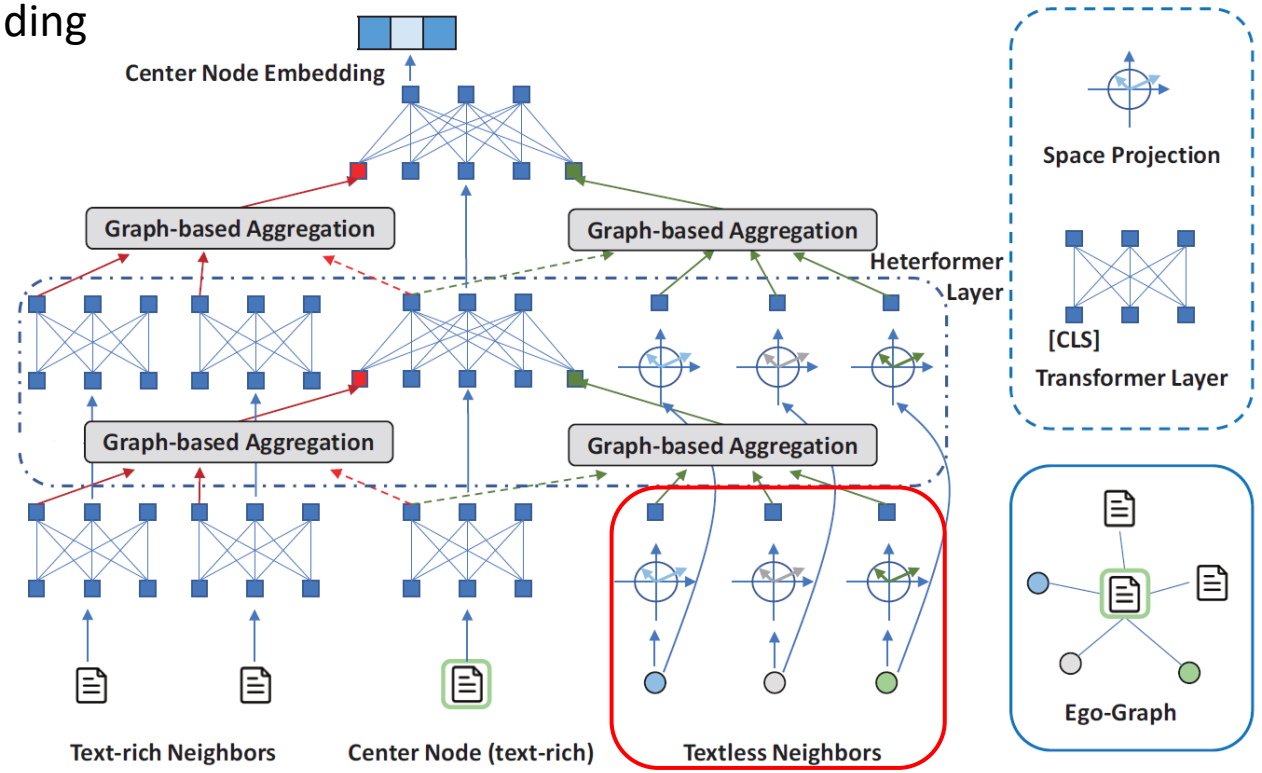
[1] Jin, B., Zhang, Y., Zhu, Q., & Han, J. (2023, August). Heterformer: Transformer-based deep node representation learning on heterogeneous text-rich networks. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (pp. 1020-1031).

PLMs for Heterogeneous Text-Attributed Graph

- 1. Textless node initialization.

$$\boldsymbol{h}_v = \boldsymbol{W}_{\phi_v}\boldsymbol{z}_v, \quad \text{where } \boldsymbol{z}_v \text{ is randomly initialized embedding}$$

ϕ_v is node type of v



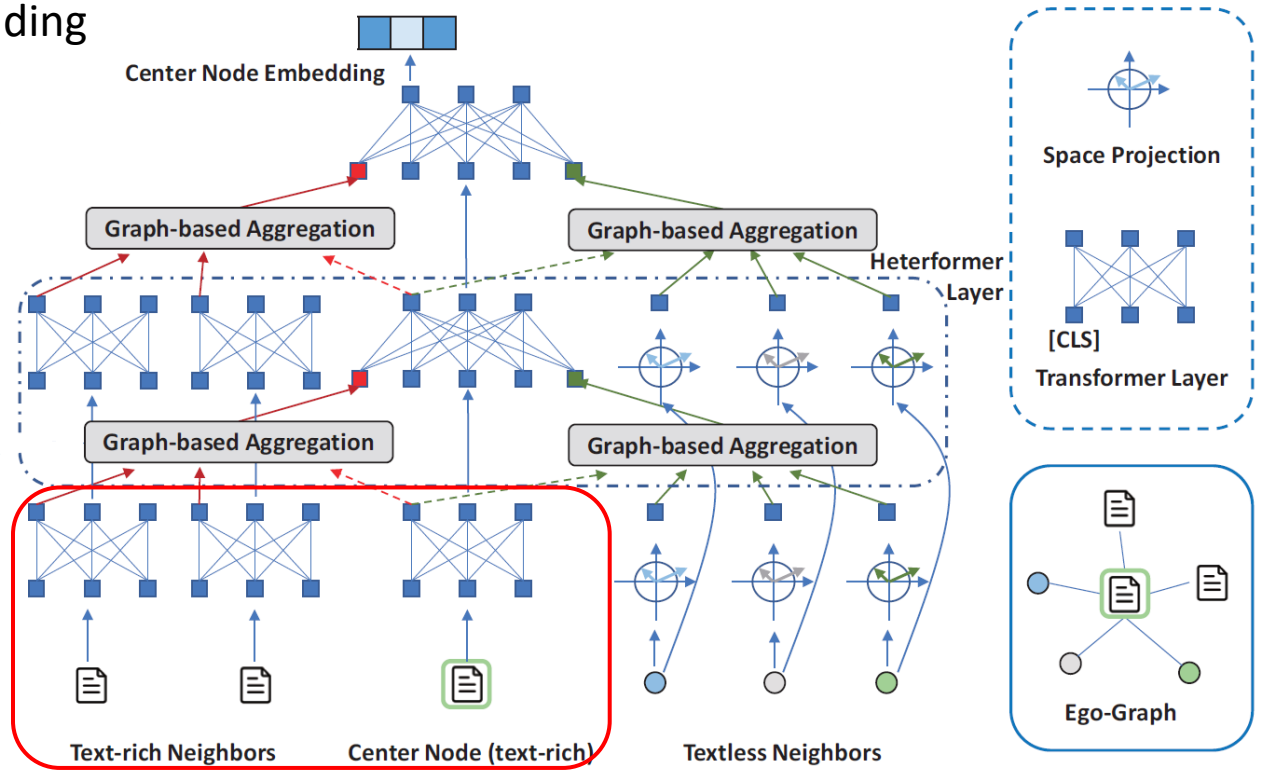
PLMs for Heterogeneous Text-Attributed Graph

- 1. Textless node initialization.

  $\boldsymbol{h}_v = \boldsymbol{W}_{\phi_v}\boldsymbol{z}_v$, where $\boldsymbol{z}_v$ is randomly initialized embedding

- 2. Text-rich node initialization.

  $\boldsymbol{H}_i^1 = \text{Transformer}^{l=1}(\boldsymbol{d}_i)$,
  where $\boldsymbol{H}_i^1 = \{\boldsymbol{h}_{i,CLS}^1, \boldsymbol{h}_{i,w_1}^1, \ldots, \boldsymbol{h}_{i,w_d}^1\}$



PLMs for Heterogeneous Text-Attributed Graph

- 1. Textless node initialization.

$$\boldsymbol{h}_v = \boldsymbol{W}_{\phi_v}\boldsymbol{z}_v, \quad \text{where } \boldsymbol{z}_v \text{ is randomly initialized embedding}$$

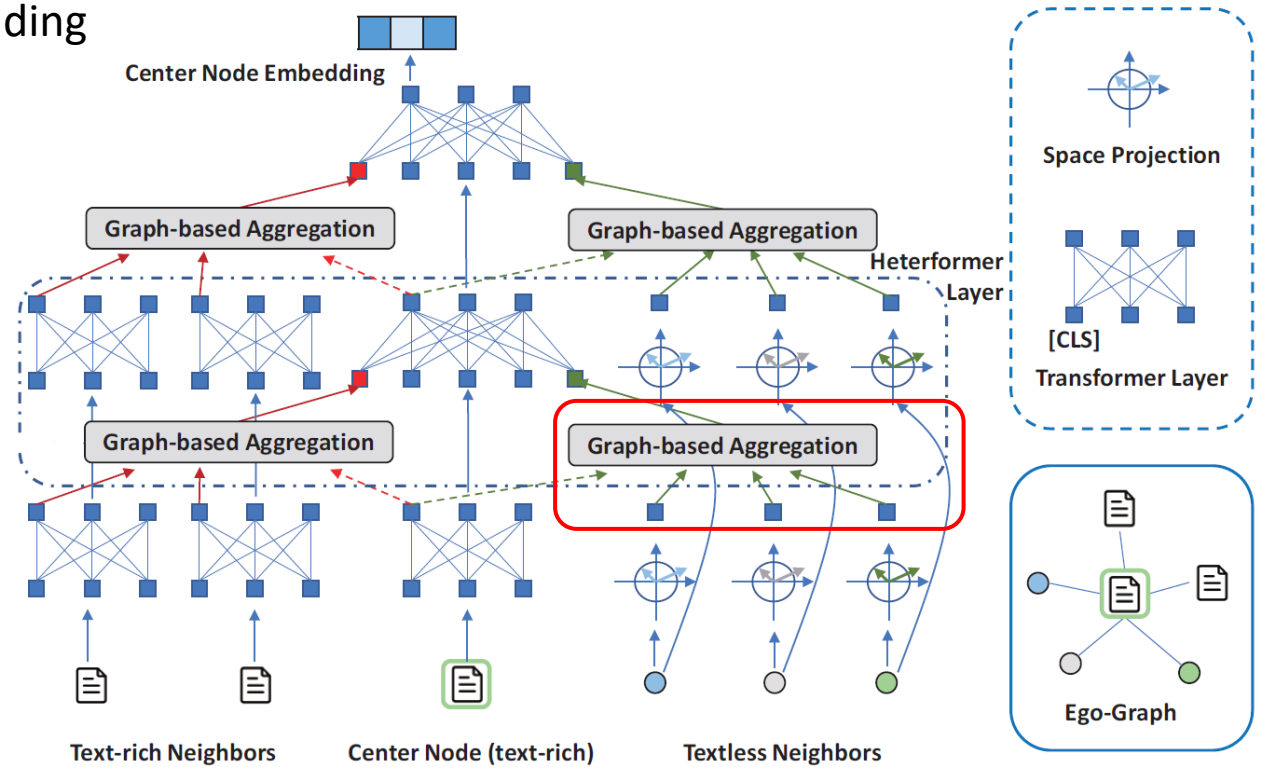
- 2. Text-rich node initialization.

$$\boldsymbol{H}_i^1 = \text{Transformer}^{l=1}(\boldsymbol{d}_i),$$
where $\boldsymbol{H}_i^1 = \{\boldsymbol{h}_{i,CLS}^1, \boldsymbol{h}_{i,w_1}^1, \ldots, \boldsymbol{h}_{i,w_d}^1\}$

- 3. Textless neighbor aggregation.

$$\boldsymbol{z}_{i,TL}^1 = \text{GNN}\left(\boldsymbol{h}_{i,CLS}^1, \{\boldsymbol{h}_v^1\}_{v\in\mathcal{N}_{TL}(i)}\right)$$

↓ Textless neighbor set



PLMs for Heterogeneous Text-Attributed Graph

- 1. Textless node initialization.

$$\boldsymbol{h}_v = \boldsymbol{W}_{\phi_v}\boldsymbol{z}_v, \quad \text{where } \boldsymbol{z}_v \text{ is randomly initialized embedding}$$

- 2. Text-rich node initialization.

$$\boldsymbol{H}_i^1 = \text{Transformer}^{l=1}(\boldsymbol{d}_i),$$
where $\boldsymbol{H}_i^1 = \{\boldsymbol{h}_{i,CLS}^1, \boldsymbol{h}_{i,w_1}^1, \ldots, \boldsymbol{h}_{i,w_d}^1\}$

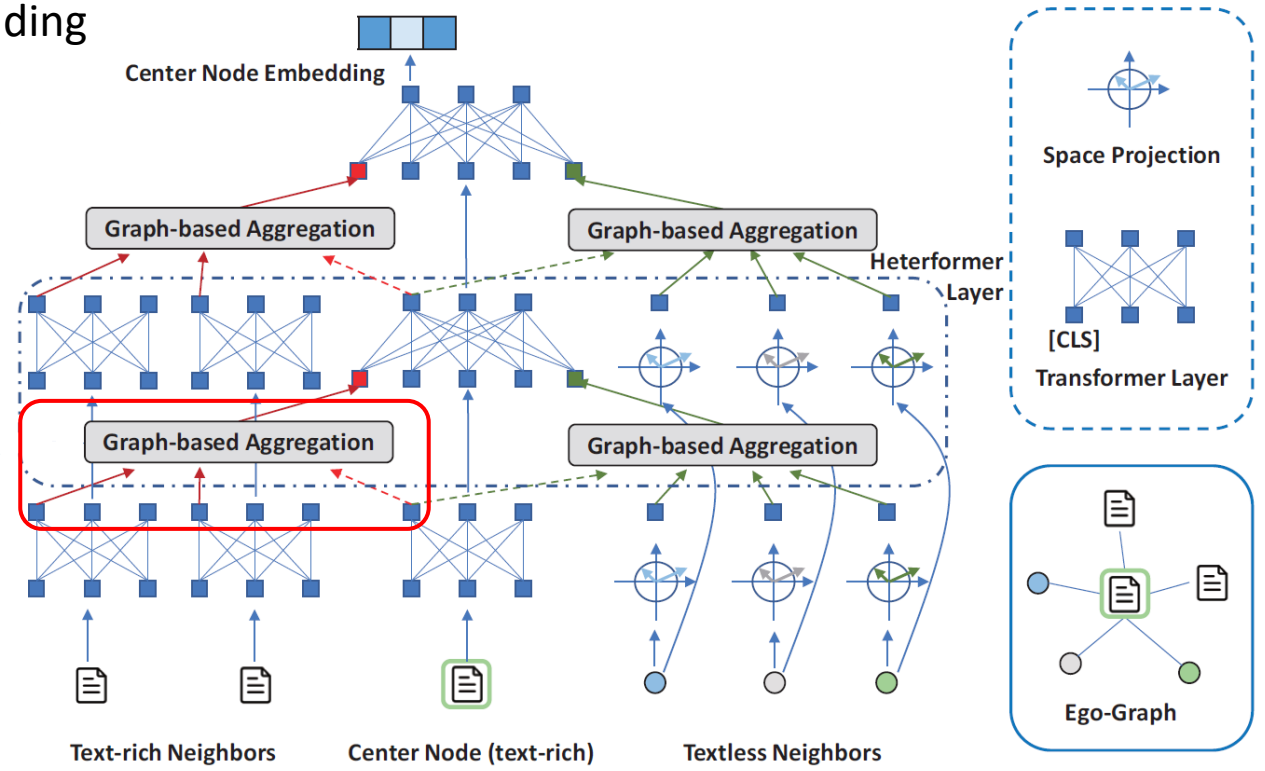
- 3. Textless neighbor aggregation.

$$\boldsymbol{z}_{i,TL}^1 = \text{GNN}\left(\boldsymbol{h}_{i,CLS}^1, \{\boldsymbol{h}_v^1\}_{v\in\mathcal{N}_{TL}(i)}\right)$$

- 4. Text-rich neighbor aggregation.

$$\boldsymbol{z}_{i,TR}^1 = \text{GNN}\left(\boldsymbol{h}_{i,CLS}^1, \{\boldsymbol{h}_{v,CLS}^1\}_{v\in\mathcal{N}_{TR}(i)}\right)$$

Text-rich neighbor set



PLMs for Heterogeneous Text-Attributed Graph

- 1. Textless node initialization.

$$\boldsymbol{h}_v = \boldsymbol{W}_{\phi_v}\boldsymbol{z}_v, \quad \text{where } \boldsymbol{z}_v \text{ is randomly initialized embedding}$$

- 2. Text-rich node initialization.

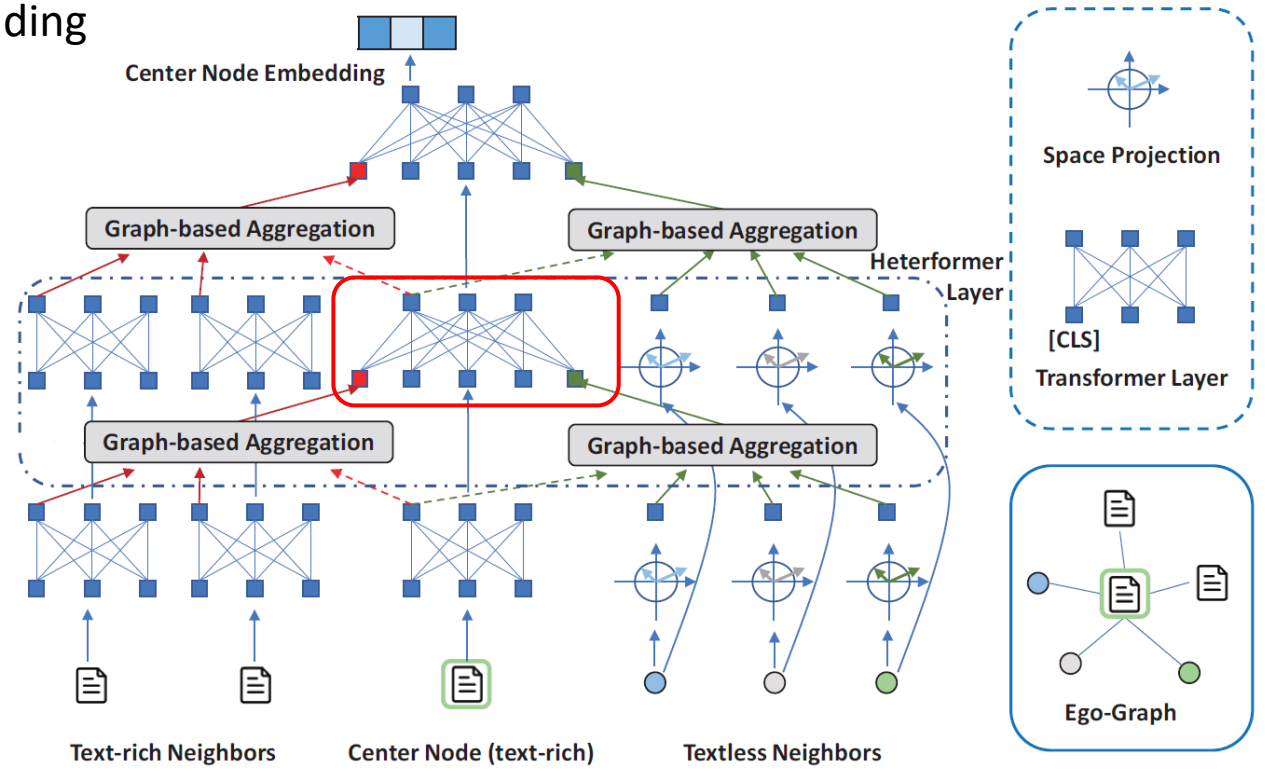
$$\boldsymbol{H}_i^1 = \text{Transformer}^{l=1}(\boldsymbol{d}_i),$$
where $\boldsymbol{H}_i^1 = \{\boldsymbol{h}_{i,CLS}^1, \boldsymbol{h}_{i,w_1}^1, \ldots, \boldsymbol{h}_{i,w_d}^1\}$

- 3. Textless neighbor aggregation.

$$\boldsymbol{z}_{i,TL}^1 = \text{GNN}\left(\boldsymbol{h}_{i,CLS}^1, \{\boldsymbol{h}_v^1\}_{v\in\mathcal{N}_{TL}(i)}\right)$$

- 4. Text-rich neighbor aggregation.

$$\boldsymbol{z}_{i,TR}^1 = \text{GNN}\left(\boldsymbol{h}_{i,CLS}^1, \{\boldsymbol{h}_{v,CLS}^1\}_{v\in\mathcal{N}_{TR}(i)}\right)$$



- 5. Concatenate graph with token embeddings.

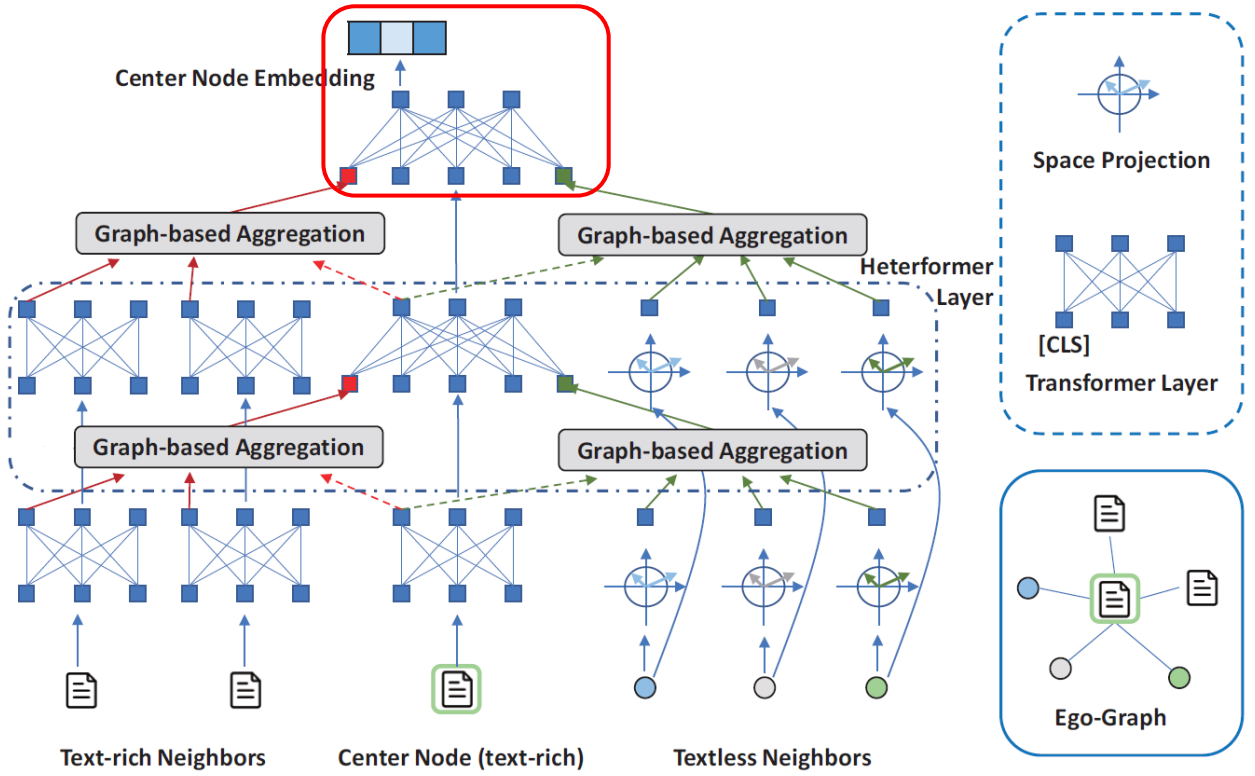
$$\widehat{\boldsymbol{H}}_i^1 = \text{Concat}(\boldsymbol{z}_{i,TR}^1, \boldsymbol{H}_i^1, \boldsymbol{z}_{i,RL}^1) \qquad \boldsymbol{H}_i^2 = \text{Transformer}^{l=2}(\widehat{\boldsymbol{H}}_i^1)$$

PLMs for Heterogeneous Text-Attributed Graph

- 6. So far, we finish from layer 1 to 2. We repeat this process for L layers.

$$\boldsymbol{H}_i^L = \{\boldsymbol{h}_{i,CLS}^L, \boldsymbol{h}_{i,w_1}^L, \ldots, \boldsymbol{h}_{i,w_d}^L\}$$

- 7. Contrastive link prediction loss.



PLMs for Heterogeneous Text-Attributed Graph

- **Experiments**
- 1. Datasets

Dataset	Node	Edge
DBLP	# paper*: 3,597,191 # venue: 28,638 # author: 2,717,797	# paper-paper: 36,787,329 # venue-paper: 3,633,613 # author-paper: 10,212,497
Twitter	# tweet*: 279,694 # POI*: 36,895 # hashtag: 72,297 # user: 76,398 # mention: 24,089	# tweet-POI: 279,694 # user-tweet: 195,785 # hashtag-tweet: 194,939 # mention-tweet: 50,901
Goodreads	# book*:1,097,438 # shelves: 6,632 # author: 205,891 # format: 768 # publisher: 62,934 # language code: 139	# book-book: 11,745,415 # shelves-book: 27,599,160 # author-book: 1,089,145 # format-book: 588,677 # publisher-book: 591,456 # language code-book: 485,733

PLMs for Heterogeneous Text-Attributed Graph

- **Experiments**
- 2. Link prediction

Models w/o graph

Models w/o meta-data

Models with meta-data

	Method	DBLP PREC	DBLP MRR	DBLP NDCG	Twitter PREC	Twitter MRR	Twitter NDCG	Goodreads PREC	Goodreads MRR	Goodreads NDCG
	MeanSAGE	0.70186	0.79643	0.84372	0.64890	0.74501	0.79911	0.63020	0.74086	0.80012
	BERT	0.75685	0.83400	0.87262	0.71792	0.78331	0.82653	0.55711	0.66675	0.73945
Homo GNN	BERT+MeanSAGE	0.81308	0.87789	0.90695	0.72007	0.78452	0.82754	0.73012	0.81668	0.85935
Homo GNN	BERT+MAXSAGE	0.81932	0.88250	0.91051	0.71978	0.78451	0.82762	0.72803	0.81639	0.85930
Homo GNN	BERT+GAT	0.81188	0.87705	0.90630	0.72308	0.78731	0.83000	0.73328	0.81700	0.85931
Homo GNN	GraphFormers	0.83242	0.89158	0.91753	0.72583	0.78906	0.83120	0.74436	0.82600	0.86649
Hetero GNN	BERT+RGCN	0.79786	0.86327	0.89448	0.71106	0.77643	0.82093	0.74883	0.83027	0.86986
Hetero GNN	BERT+HAN	0.81359	0.87823	0.90720	0.72365	0.78804	0.83056	0.73291	0.81741	0.85973
Hetero GNN	BERT+HGT	0.81704	0.88138	0.90978	0.71532	0.78003	0.82372	0.72237	0.81118	0.85519
Hetero GNN	BERT+SHGN	0.81485	0.87854	0.90736	0.72176	0.78664	0.82953	0.73623	0.81950	0.86134
Hetero GNN	GraphFormers++	0.82333	0.88557	0.91296	0.71587	0.77989	0.82359	0.75361	0.83284	0.87170
	Heterformer	**0.84736***	**0.90193***	**0.92547***	**0.72716***	**0.79076***	**0.83276***	**0.76328***	**0.84000***	**0.87731***

PLMs for Heterogeneous Text-Attributed Graph

- **Experiments**
- 3. Inductive text-rich node classification

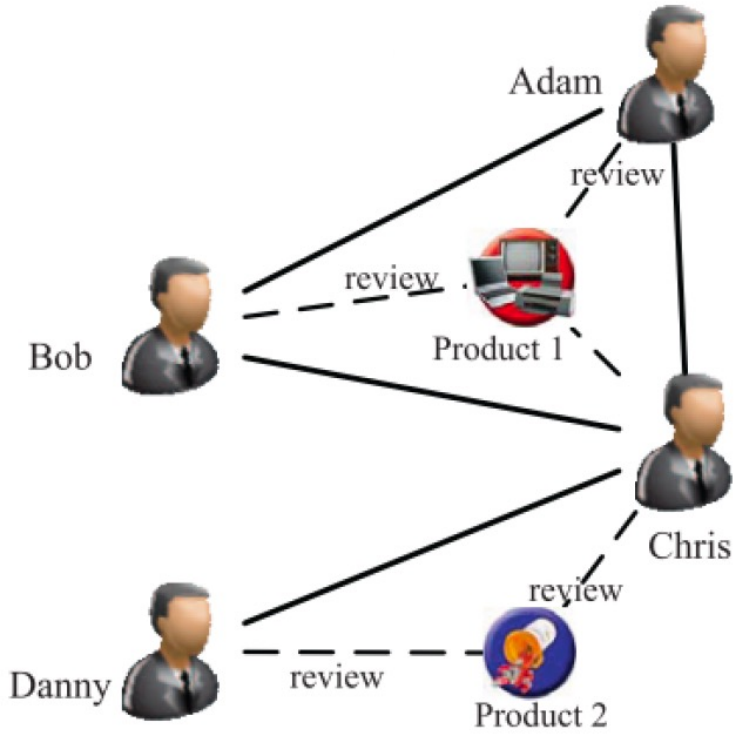
Method	DBLP		Goodreads	
	PREC	**NDCG**	**PREC**	**NDCG**
BERT	0.71782	0.83493	0.96373	0.91734
BERT+MaxSAGE	0.72987	0.84266	0.97746	0.93670
BERT+MeanSAGE	0.73058	0.84285	0.97697	0.93552
BERT+GAT	0.71328	0.83205	0.97345	0.93035
GraphFormers	0.73634	0.84652	0.97667	0.93512
BERT+HAN	0.71523	0.83371	0.97365	0.93057
BERT+HGT	0.75917	0.86049	0.97890	0.93869
BERT+SHGN	0.71512	0.83306	0.97365	0.93099
GraphFormers++	0.75096	0.85534	0.97984	0.94027
Heterformer	**0.76700***	**0.86526***	**0.98243***	**0.94513***

Section 2: PLM-based Text-Attributed Graph Models

- Section 2.1: PLMs for Static Text-Attributed Graph
- Section 2.2: PLMs for Heterogeneous Text-Attributed Graph
- Section 2.3: Textual-edge Text-Attributed Graph
- Section 2.4: Pre-training Strategy

Textual-edge Text-Attributed Graph

- Documents appear on edges, instead of nodes. We present Edgeformers [1].



(a) Online user-product review graph

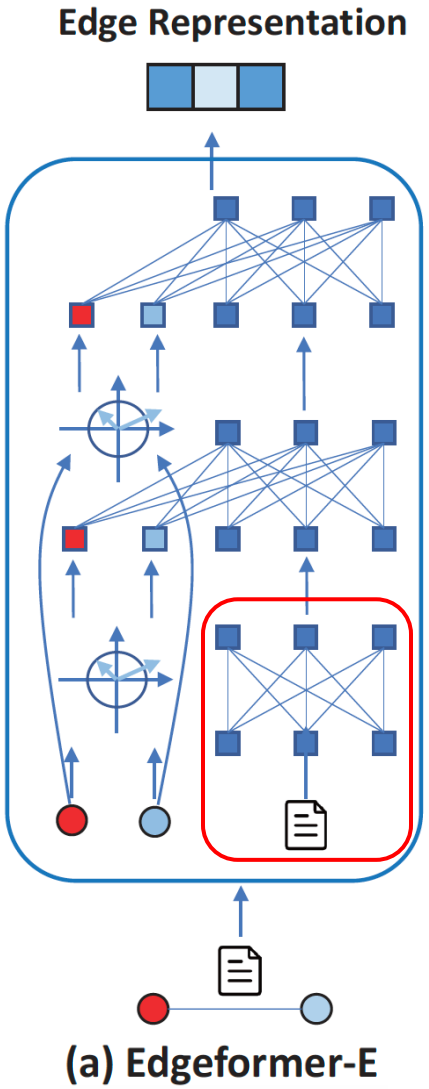
(b) Email communication graph

[1] Jin, B., Zhang, Y., Meng, Y., & Han, J. (2022, September). Edgeformers: Graph-Empowered Transformers for Representation Learning on Textual-Edge Networks. In *The Eleventh International Conference on Learning Representations*.

Textual-edge Text-Attributed Graph

- 1. Feature initialization.

$$\boldsymbol{H}_e^1 = \text{Transformer}^{l=1}(\boldsymbol{d}_e), \quad \text{where} \quad \boldsymbol{H}_e^1 = \{\boldsymbol{h}_{e,CLS}^1, \boldsymbol{h}_{e,w_1}^1, \dots, \boldsymbol{h}_{e,w_e}^1\}$$



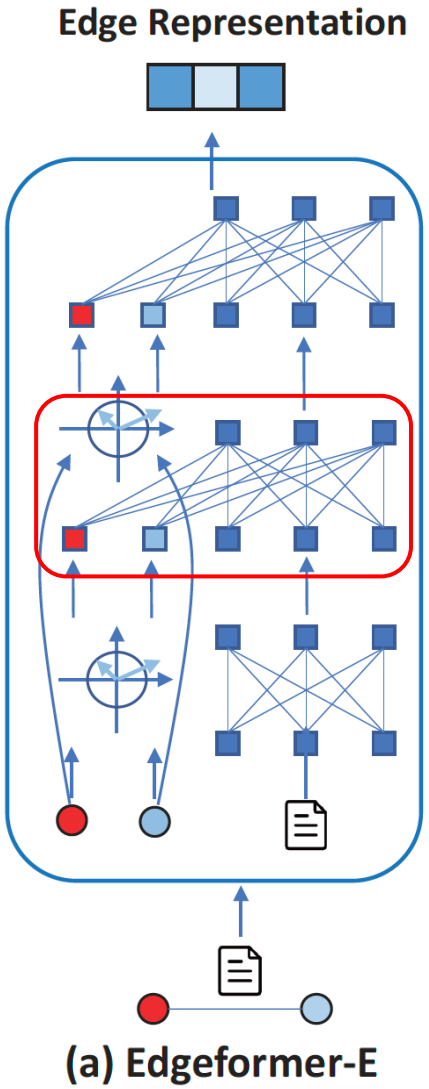
Textual-edge Text-Attributed Graph

- 1. Feature initialization.

$$\boldsymbol{H}_e^1 = \text{Transformer}^{l=1}(\boldsymbol{d}_e), \quad \text{where} \quad \boldsymbol{H}_e^1 = \{\boldsymbol{h}_{e,CLS}^1, \boldsymbol{h}_{e,w_1}^1, \dots, \boldsymbol{h}_{e,w_e}^1\}$$

- 2. Concatenate graph embeddings with tokens.

$$\widehat{\boldsymbol{H}}_e^1 = \text{Concat}(\boldsymbol{z}_i^1, \boldsymbol{z}_j^1, \boldsymbol{H}_e^1)$$



Textual-edge Text-Attributed Graph

- 1. Feature initialization.

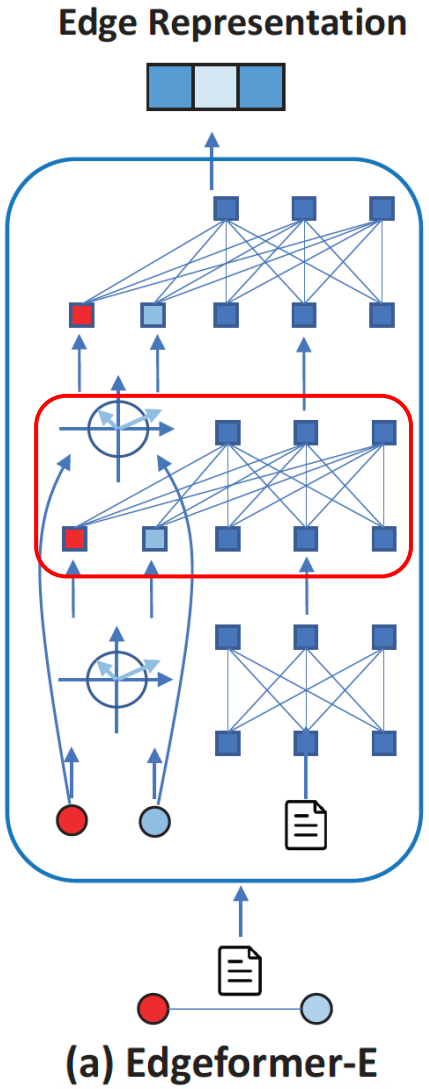
$$\boldsymbol{H}_e^1 = \text{Transformer}^{l=1}(\boldsymbol{d}_e), \quad \text{where} \quad \boldsymbol{H}_e^1 = \{\boldsymbol{h}_{e,CLS}^1, \boldsymbol{h}_{e,w_1}^1, \dots, \boldsymbol{h}_{e,w_e}^1\}$$

- 2. Concatenate graph embeddings with tokens.

$$\widehat{\boldsymbol{H}}_e^1 = \text{Concat}(\boldsymbol{z}_i^1, \boldsymbol{z}_j^1, \boldsymbol{H}_e^1)$$

- 3. Transformer layer.

$$\boldsymbol{H}_e^2 = \text{Transformer}^{l=2}(\widehat{\boldsymbol{H}}_e^1)$$



Textual-edge Text-Attributed Graph

- 1. Feature initialization.

$$\boldsymbol{H}_e^1 = \text{Transformer}^{l=1}(\boldsymbol{d}_e), \quad \text{where} \quad \boldsymbol{H}_e^1 = \{\boldsymbol{h}_{e,CLS}^1, \boldsymbol{h}_{e,w_1}^1, \dots, \boldsymbol{h}_{e,w_e}^1\}$$

- 2. Concatenate graph embeddings with tokens.

$$\widehat{\boldsymbol{H}}_e^1 = \text{Concat}(\boldsymbol{z}_i^1, \boldsymbol{z}_j^1, \boldsymbol{H}_e^1)$$

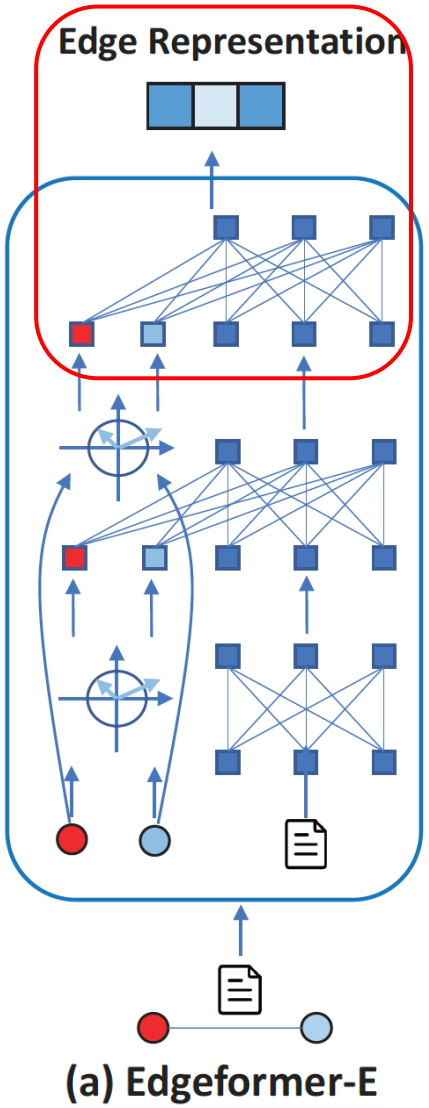
- 3. Transformer layer.

$$\boldsymbol{H}_e^2 = \text{Transformer}^{l=2}(\widehat{\boldsymbol{H}}_e^1)$$

- 4. Repeat steps 2 and 3 for L layers.

$$\boldsymbol{H}_e^L = \{\boldsymbol{h}_{e,CLS}^L, \boldsymbol{h}_{e,w_1}^L, \dots, \boldsymbol{h}_{e,w_e}^L\}$$

- 5. Edge classification with cross-entropy loss.



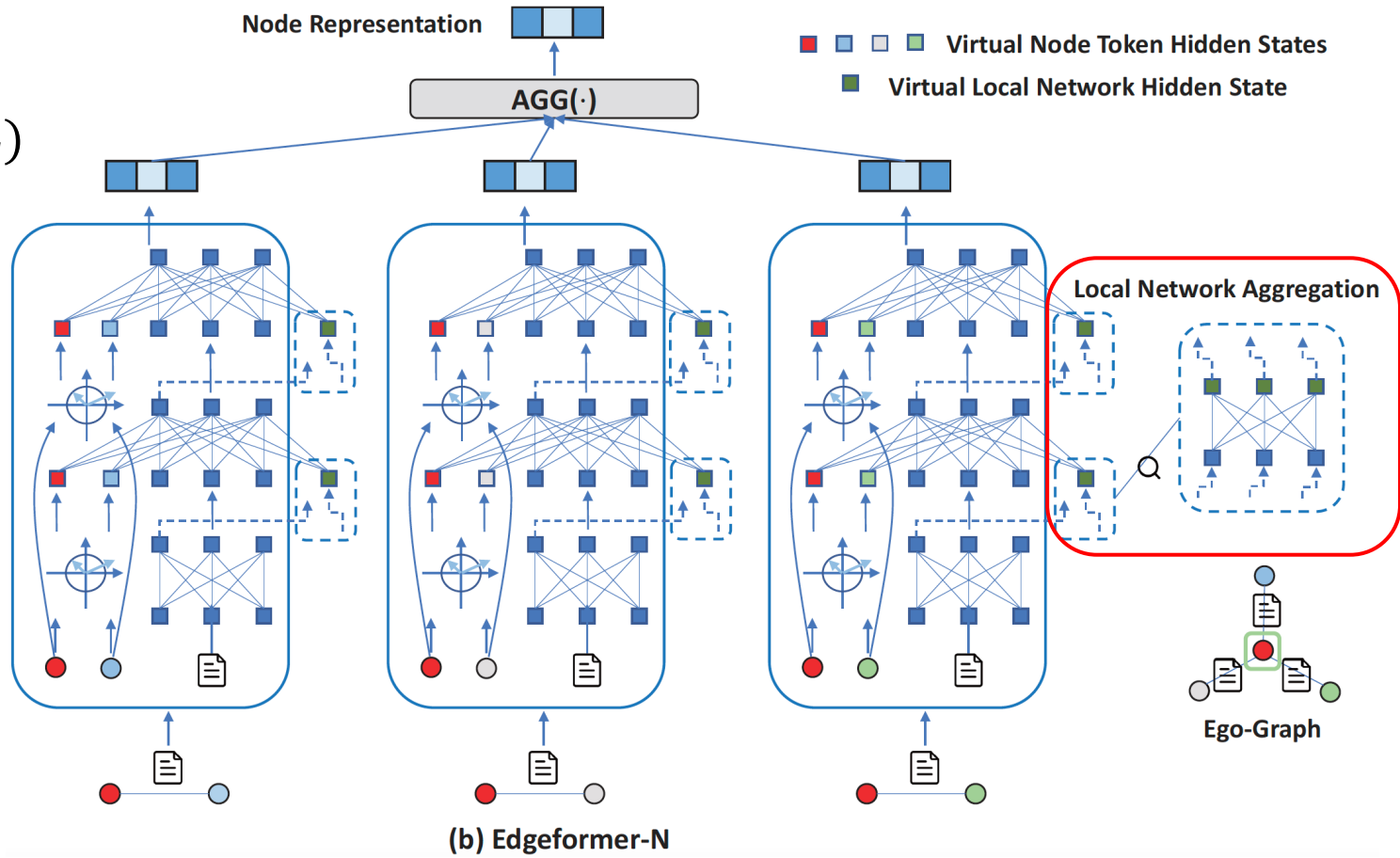
Textual-edge Text-Attributed Graph

- 1. Local edge aggregation.

$$\overline{\boldsymbol{H}}_e^l = \text{MultiHeadAtt}(\boldsymbol{h}_{e_1,CLS}^l, \boldsymbol{h}_{e_2,CLS}^l, \boldsymbol{h}_{e_3,CLS}^l)$$

$$\widehat{\boldsymbol{H}}_e^l = \text{Concat}(\boldsymbol{z}_i^l, \boldsymbol{z}_j^l, \overline{\boldsymbol{h}}_e^l, \boldsymbol{H}_e^l)$$

$$\boldsymbol{H}_e^{l+1} = \text{Transformer}^{l+1}(\widehat{\boldsymbol{H}}_e^l)$$



Textual-edge Text-Attributed Graph

- 1. Local edge aggregation.

$$\overline{\boldsymbol{H}}_e^l = \text{MultiHeadAtt}(\boldsymbol{h}_{e_1,CLS}^l, \boldsymbol{h}_{e_2,CLS}^l, \boldsymbol{h}_{e_3,CLS}^l)$$

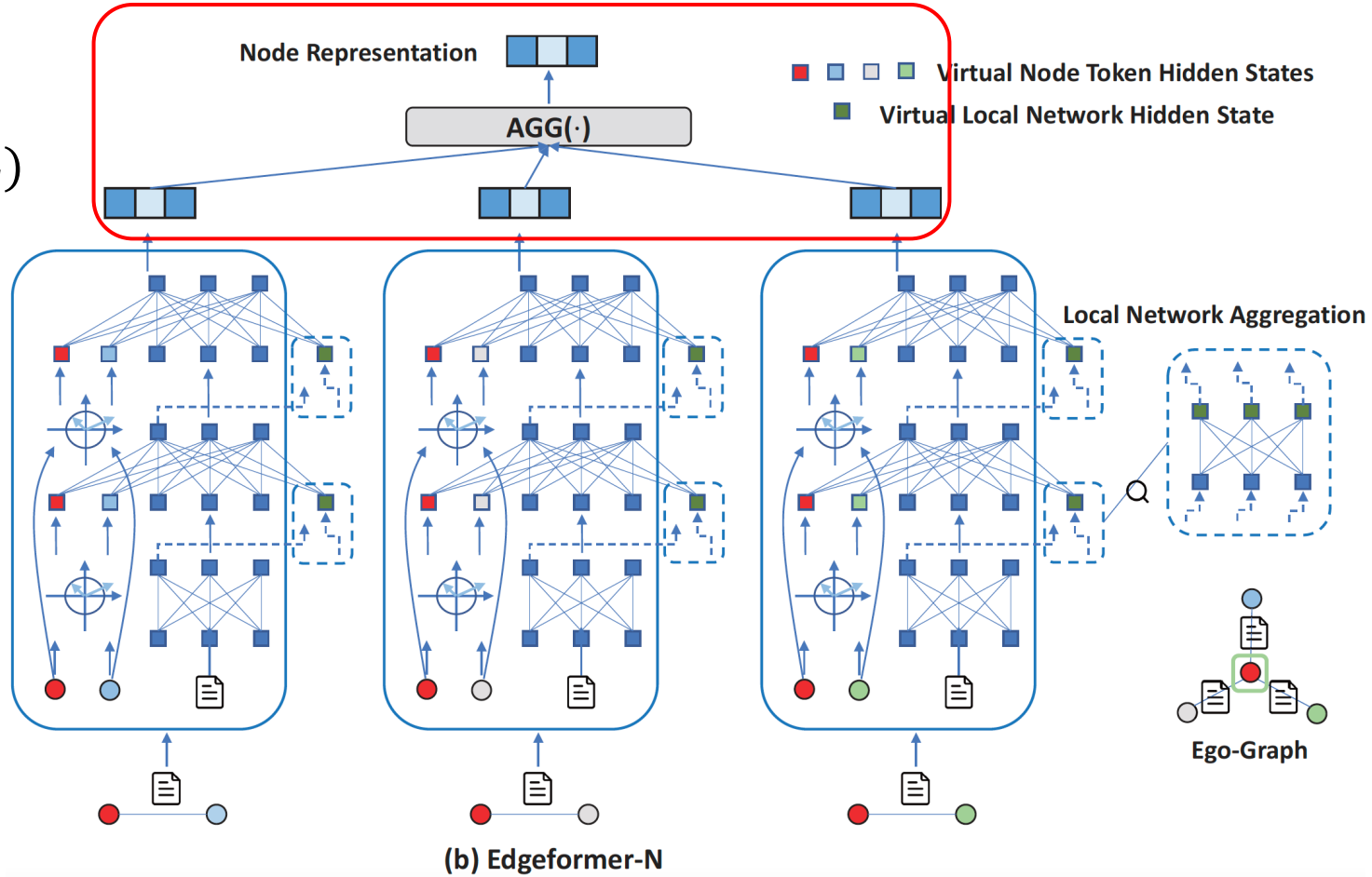
$$\widehat{\boldsymbol{H}}_e^l = \text{Concat}(\boldsymbol{z}_i^l, \boldsymbol{z}_j^l, \overline{\boldsymbol{h}}_e^l, \boldsymbol{H}_e^l)$$

$$\boldsymbol{H}_e^{l+1} = \text{Transformer}^{l+1}(\widehat{\boldsymbol{H}}_e^l)$$

- 2. Global edge aggregation.

$$\alpha_{e,v} = \text{softmax}(\boldsymbol{h}_{e,CLS}^L \boldsymbol{W} \boldsymbol{z}_v)$$

$$\boldsymbol{h}_v = \sum_{e \in \{e_1,e_2,e_3\}} \alpha_{e,v} \boldsymbol{h}_{e,CLS}^L$$



Textual-edge Text-Attributed Graph

- **Experiments**
- 1. Datasets

Dataset	# Node	# Edge
Amazon-Movie	173,986	1,697,533
Amazon-Apps	100,468	752,937
Goodreads-Crime	385,203	1,849,236
Goodreads-Children	192,036	734,640
StackOverflow	129,322	281,657

Textual-edge Text-Attributed Graph

- **Experiments**
- 2. Edge classification with Edgeformer-E

Cascaded method

	Amazon-Movie		Amazon-Apps		Goodreads-Crime		Goodreads-Children	
Model	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1
TF-IDF	50.01	64.22	48.30	62.88	43.07	51.72	39.42	49.90
TF-IDF+nodes	53.59	66.34	50.56	65.08	49.35	57.50	47.32	56.78
BERT	61.38	71.36	59.11	69.27	56.41	61.29	51.57	57.72
BERT+nodes	*63.00*	*72.45*	*59.72*	*70.82*	*58.64*	*65.02*	*54.42*	*60.46*
Edgeformer-E	**64.18**	**73.59**	**60.67**	**71.28**	**61.03**	**65.86**	**57.45**	**61.71**

Textual-edge Text-Attributed Graph

- **Experiments**
- 3. Link prediction with Edgeformer-N

Cascaded models

	Amazon-Movie		Amazon-Apps		Goodreads-Crime		Goodreads-Children		StackOverflow	
Model	MRR	NDCG	MRR	NDCG	MRR	NDCG	MRR	NDCG	MRR	NDCG
MF	0.2032	0.3546	0.1482	0.3052	0.1923	0.3443	0.1137	0.2716	0.1040	0.2642
MeanSAGE	0.2138	0.3657	0.1766	0.3343	0.1832	0.3368	0.1066	0.2647	0.1174	0.2768
MaxSAGE	0.2178	0.3694	0.1674	0.3258	0.1846	0.3387	0.1066	0.2647	0.1173	0.2769
GIN	0.2140	0.3648	0.1797	0.3362	0.1846	0.3374	0.1128	0.2700	0.1189	0.2778
CensNet	0.2048	0.3568	0.1894	0.3457	0.1880	0.3398	0.1157	0.2726	0.1235	0.2806
NENN	0.2565	0.4032	0.1996	0.3552	0.2173	0.3670	0.1297	0.2854	0.1257	0.2854
BERT	0.2391	0.3864	0.1790	0.3350	0.1986	0.3498	0.1274	0.2836	0.1666	0.3252
BERT+MaxSAGE	0.2780	0.4224	0.2055	0.3602	0.2193	0.3694	0.1312	0.2872	0.1681	0.3264
BERT+MeanSAGE	0.2491	0.3972	0.1983	0.3540	0.1952	0.3477	0.1223	0.2791	0.1678	0.3264
BERT+GIN	0.2573	0.4037	0.2000	0.3552	0.2007	0.3522	0.1238	0.2801	*0.1708*	*0.3279*
GraphFormers	0.2756	0.4198	0.2066	0.3607	0.2176	0.3684	0.1323	0.2887	0.1693	0.3278
BERT+CensNet	0.1919	0.3462	0.1544	0.3132	0.1437	0.3000	0.0847	0.2436	0.1173	0.2789
BERT+NENN	*0.2821*	*0.4256*	*0.2127*	*0.3666*	*0.2262*	*0.3756*	*0.1365*	*0.2925*	0.1619	0.3215
Edgeformer-N	**0.2919**	**0.4344**	**0.2239**	**0.3771**	**0.2395**	**0.3875**	**0.1446**	**0.3000**	**0.1754**	**0.3339**
+ Δ %	**3.5%**	**2.1%**	**5.3%**	**2.9%**	**5.9%**	**3.2%**	**5.9%**	**2.6%**	**2.7%**	**1.8%**

Section 2: PLM-based Text-Attributed Graph Models

- Section 2.1: PLMs for Static Text-Attributed Graph
- Section 2.2: PLMs for Heterogeneous Text-Attributed Graph
- Section 2.3: Textual-edge Text-Attributed Graph
- Section 2.4: Pre-training Strategy

Pre-training Strategy

- **Pre-training:**
- Given Text-Attributed Graph (TAG) $\mathcal{G} = \{\mathcal{D}, \mathcal{E}\}$, we train the model in a self-supervised method only using textual content $\mathcal{D}$ and graph structure $\mathcal{E}$.
- **Fine-tuning:**
- Given the pre-trained model, we further optimize its parameters with a specific downstream task, such as text classification.

- We present Patton [1] and Specter [2].

[1] Jin, B., Zhang, W., Zhang, Y., Meng, Y., Zhang, X., Zhu, Q., & Han, J. (2023). PATTON: Language Model Pretraining on Text-Rich Networks. In 61st Annual Meeting of the Association for Computational Linguistics, ACL 2023 (pp. 7005-7020). Association for Computational Linguistics (ACL).
[2] Cohan, A., Feldman, S., Beltagy, I., Downey, D., & Weld, D. S. (2020). SPECTER: Document-level representation learning using citation-informed transformers. In 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020 (pp. 2270-2282). Association for Computational Linguistics (ACL).

Pre-training Strategy: Patton

- Patton is a pre-training model for GraphFormer. Below is a quick review.
- 1. Feature initialization.

$$\boldsymbol{H}_i^1 = \text{Transformer}^{l=1}(\boldsymbol{d}_i), \quad \text{where} \quad \boldsymbol{H}_i^1 = \{\boldsymbol{h}_{i,CLS}^1, \boldsymbol{h}_{i,w_1}^1, \ldots, \boldsymbol{h}_{i,w_d}^1\}$$

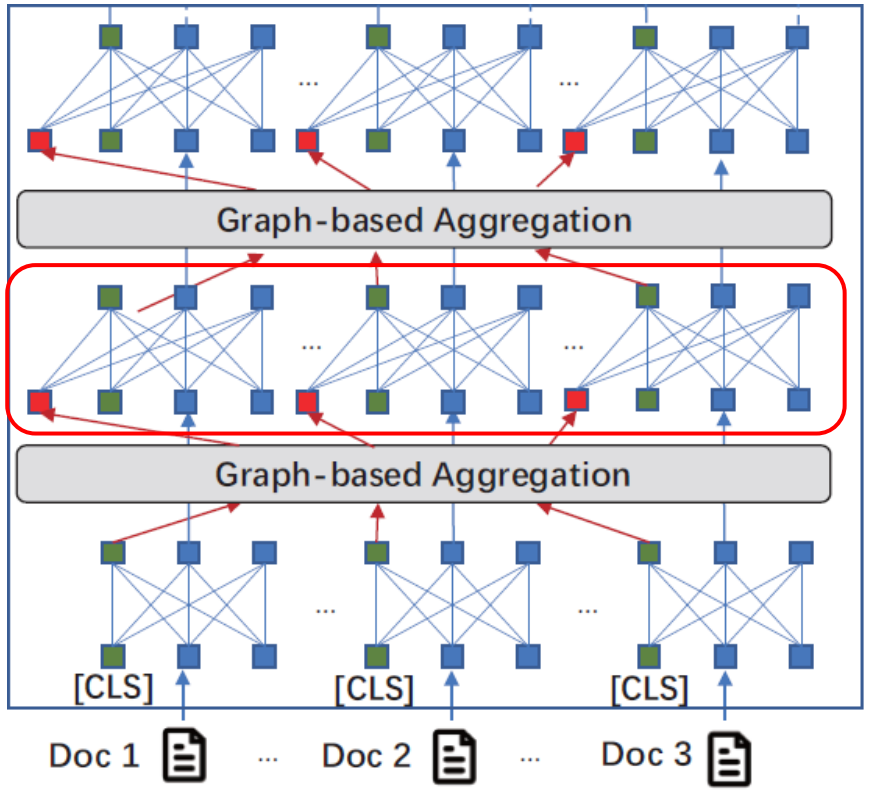
- 2. Graph aggregation.

$$\boldsymbol{z}_i^1 = \text{GNN}\left(\boldsymbol{h}_{i,CLS}^1, \{\boldsymbol{h}_{j,CLS}^1\}_{j \in \mathcal{N}(i)}\right)$$

- 3. Concatenate graph embedding with tokens.

$$\widehat{\boldsymbol{H}}_i^1 = \text{Concat}(\boldsymbol{z}_i^1, \boldsymbol{H}_i^1) \qquad \boldsymbol{H}_i^2 = \text{Transformer}^{l=2}(\widehat{\boldsymbol{H}}_i^1)$$

- Due to contextualization, $\boldsymbol{H}_i^2$ preserves graph.



Pre-training Strategy: Patton

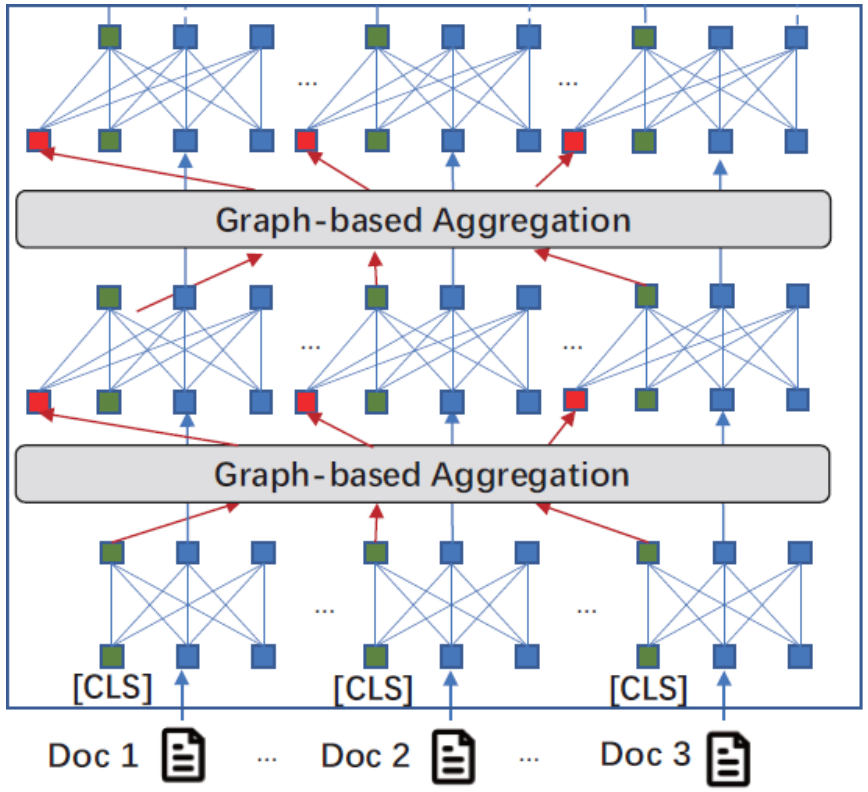
- After we obtain document representations, pre-training loss has two parts.

$$\boldsymbol{H}_i^L = \{\boldsymbol{h}_{i,CLS}^L, \boldsymbol{h}_{i,w_1}^L, \dots, \boldsymbol{h}_{i,w_d}^L\}$$

- Part 1: Masked Language Modeling (MLM).

$$p_{w_j} = \text{softmax}(\boldsymbol{W}\boldsymbol{h}_{i,w_j}^L) \qquad \mathcal{L}_{\text{MLM}} = -\sum_{w_j \in Masked} \log p_{w_j}$$

Parameter for MLM



Pre-training Strategy: Patton

- After we obtain document representations, pre-training loss has two parts.

$$\boldsymbol{H}_i^L = \{\boldsymbol{h}_{i,CLS}^L, \boldsymbol{h}_{i,w_1}^L, \dots, \boldsymbol{h}_{i,w_d}^L\}$$

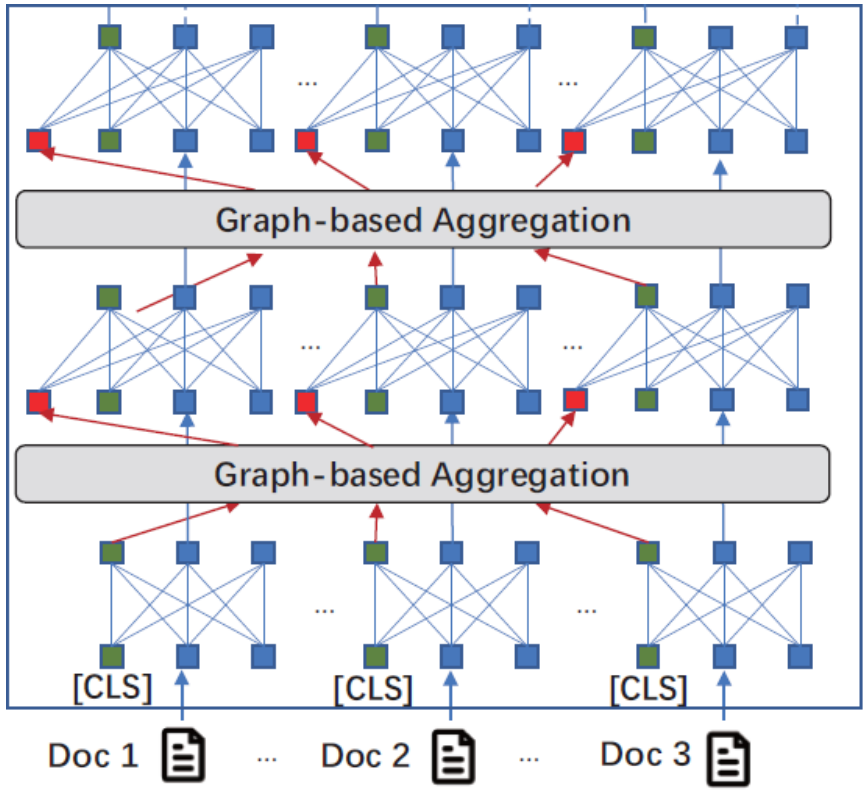
- Part 1: Masked Language Modeling (MLM).

$$p_{w_j} = \text{softmax}(\boldsymbol{W}\boldsymbol{h}_{i,w_j}^L) \qquad \mathcal{L}_{\text{MLM}} = -\sum_{w_j \in Masked} \log p_{w_j}$$

- Part 2: Masked Node Prediction (MNP).

$$\mathcal{L}_{MNP} = -\log \frac{\exp(< \boldsymbol{h}_{i,CLS}^L, \boldsymbol{h}_{j,CLS}^L >)}{\exp(< \boldsymbol{h}_{i,CLS}^L, \boldsymbol{h}_{j,CLS}^L >) + \sum_{b \in B \setminus \text{j}} \exp(< \boldsymbol{h}_{i,CLS}^L, \boldsymbol{h}_{b,CLS}^L >)}$$

Documents in a minibatch B



Pre-training Strategy: Patton

- After we obtain document representations, pre-training loss has two parts.

$$\boldsymbol{H}_i^L = \{\boldsymbol{h}_{i,CLS}^L, \boldsymbol{h}_{i,w_1}^L, \dots, \boldsymbol{h}_{i,w_d}^L\}$$

- Part 1: Masked Language Modeling (MLM).

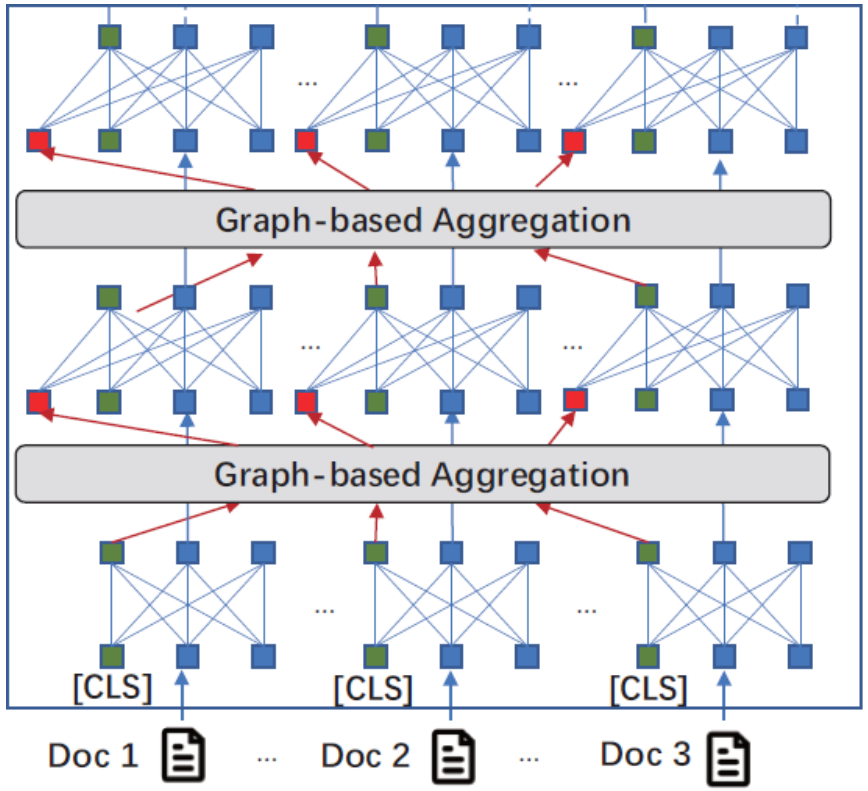
$$p_{w_j} = \text{softmax}(\boldsymbol{W}\boldsymbol{h}_{i,w_j}^L) \qquad \mathcal{L}_{\text{MLM}} = -\sum_{w_j \in Masked} \log p_{w_j}$$

- Part 2: Masked Node Prediction (MNP).

$$\mathcal{L}_{\text{MNP}} = -\log \frac{\exp(< \boldsymbol{h}_{i,CLS}^L, \boldsymbol{h}_{j,CLS}^L >)}{\exp(< \boldsymbol{h}_{i,CLS}^L, \boldsymbol{h}_{j,CLS}^L >) + \sum_{b \in B \setminus \text{j}} \exp(< \boldsymbol{h}_{i,CLS}^L, \boldsymbol{h}_{b,CLS}^L >)}$$

- Final pre-training loss.

$$\mathcal{L} = \mathcal{L}_{\text{MLM}} + \mathcal{L}_{\text{MNP}}$$

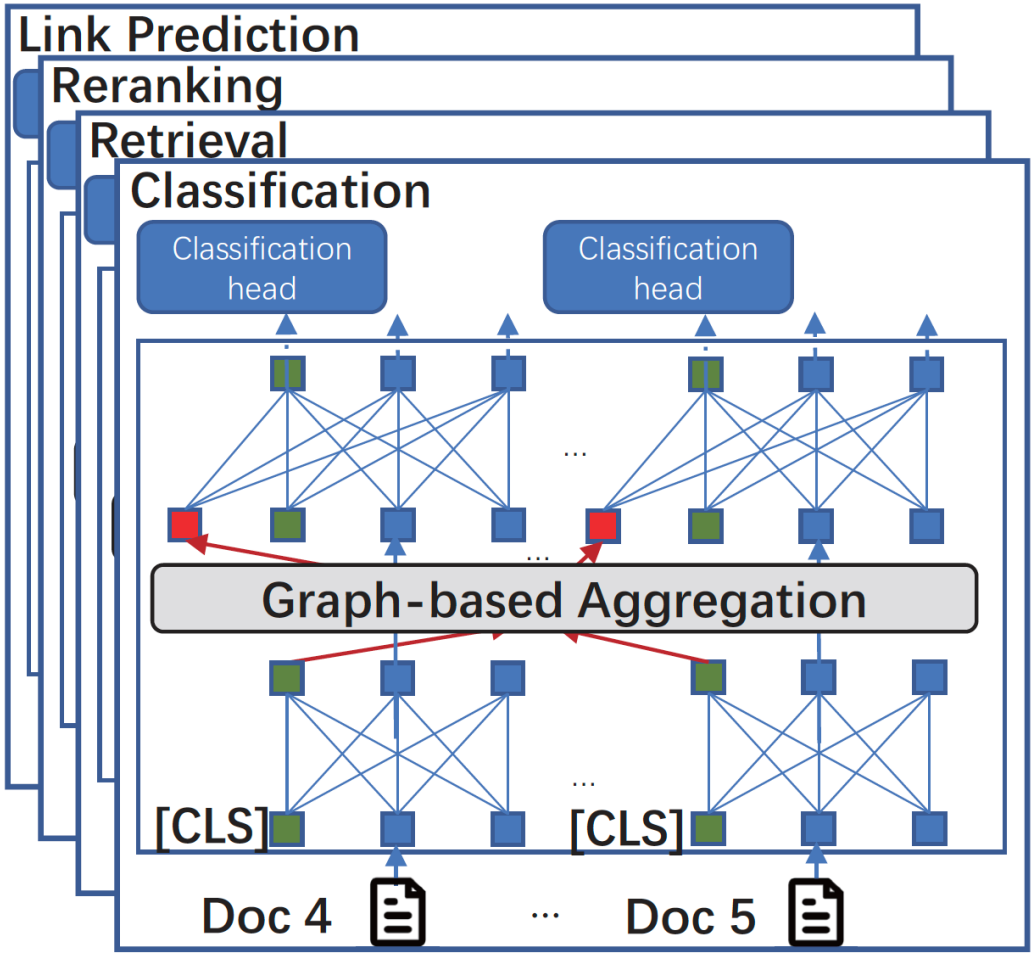


Pre-training Strategy: Patton

- 1. Fine-tune with text classification.

$$\hat{\boldsymbol{p}} = \text{softmax}\left(f_{\text{MLP}}\left(\boldsymbol{h}_{i,CLS}^{L}\right)\right)$$

$$\mathcal{L}_{\text{TextClf}} = \text{CrossEntropy}(\hat{\boldsymbol{p}}, \boldsymbol{p})$$



Pre-training Strategy: Patton

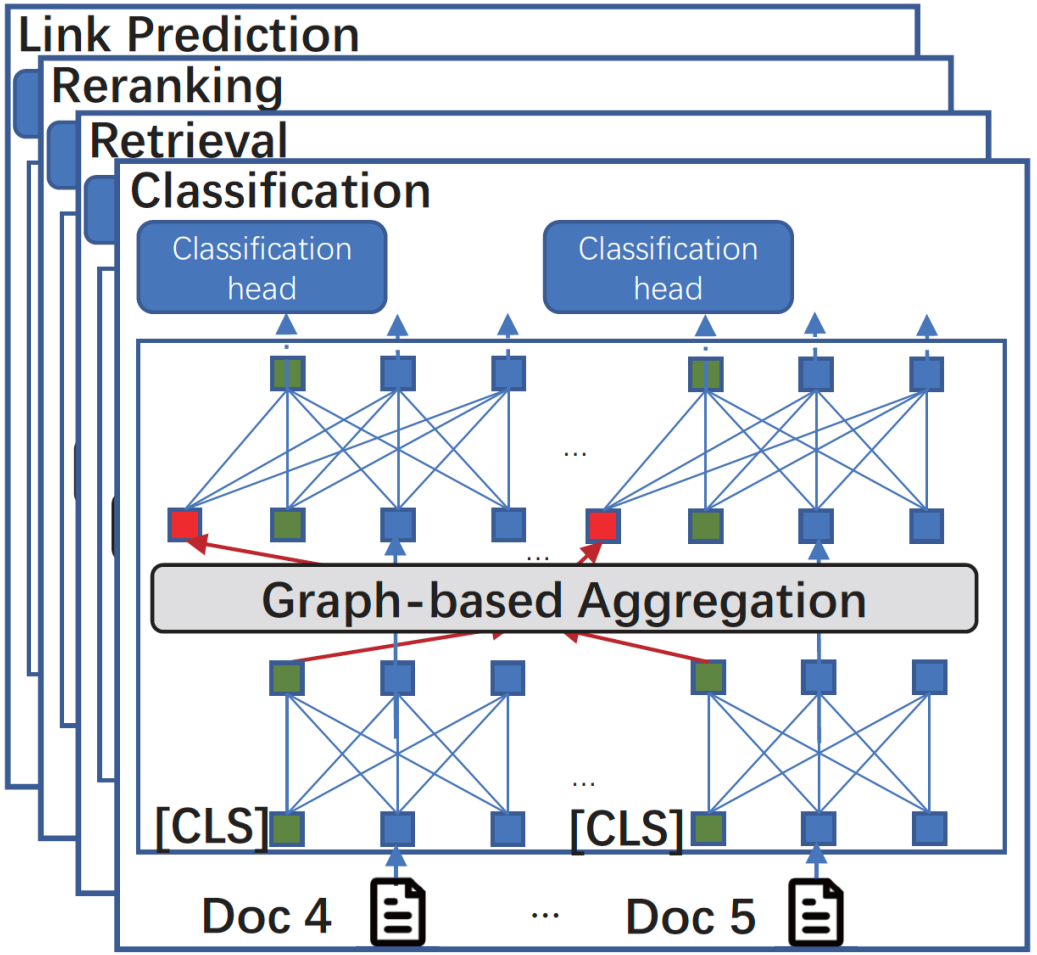
- 1. Fine-tune with text classification.

$$\hat{\boldsymbol{p}} = \mathrm{softmax}\left(f_{\mathrm{MLP}}\left(\boldsymbol{h}_{i,CLS}^{L}\right)\right)$$

$$\mathcal{L}_{\mathrm{TextClf}} = \mathrm{CrossEntropy}(\hat{\boldsymbol{p}}, \boldsymbol{p})$$

- 2. Fine-tune with link prediction.

$$\mathcal{L}_{\mathrm{MNP}} = -\log \frac{\exp(< \boldsymbol{h}_{i,CLS}^{L}, \boldsymbol{h}_{j,CLS}^{L} >)}{\exp(< \boldsymbol{h}_{i,CLS}^{L}, \boldsymbol{h}_{j,CLS}^{L} >) + \sum_{b \in B \setminus \mathrm{j}} \exp(< \boldsymbol{h}_{i,CLS}^{L}, \boldsymbol{h}_{b,CLS}^{L} >)}$$



Pre-training Strategy: Patton

- **Experiments**
- 1. Datasets

Dataset	#Nodes	#Edges	#Fine-Classes	#Coarse-Classes
Mathematics	490,551	2,150,584	14,271	18
Geology	431,834	1,753,762	7,883	17
Economics	178,670	1,042,253	5,205	40
Clothes	889,225	7,876,427	2,771	9
Sports	314,448	3,461,379	3,034	16

Pre-training Strategy: Patton

- **Experiments**
- 2. Document classification

w/o pretrain on TAG

pretrain only with MLM

Ablation

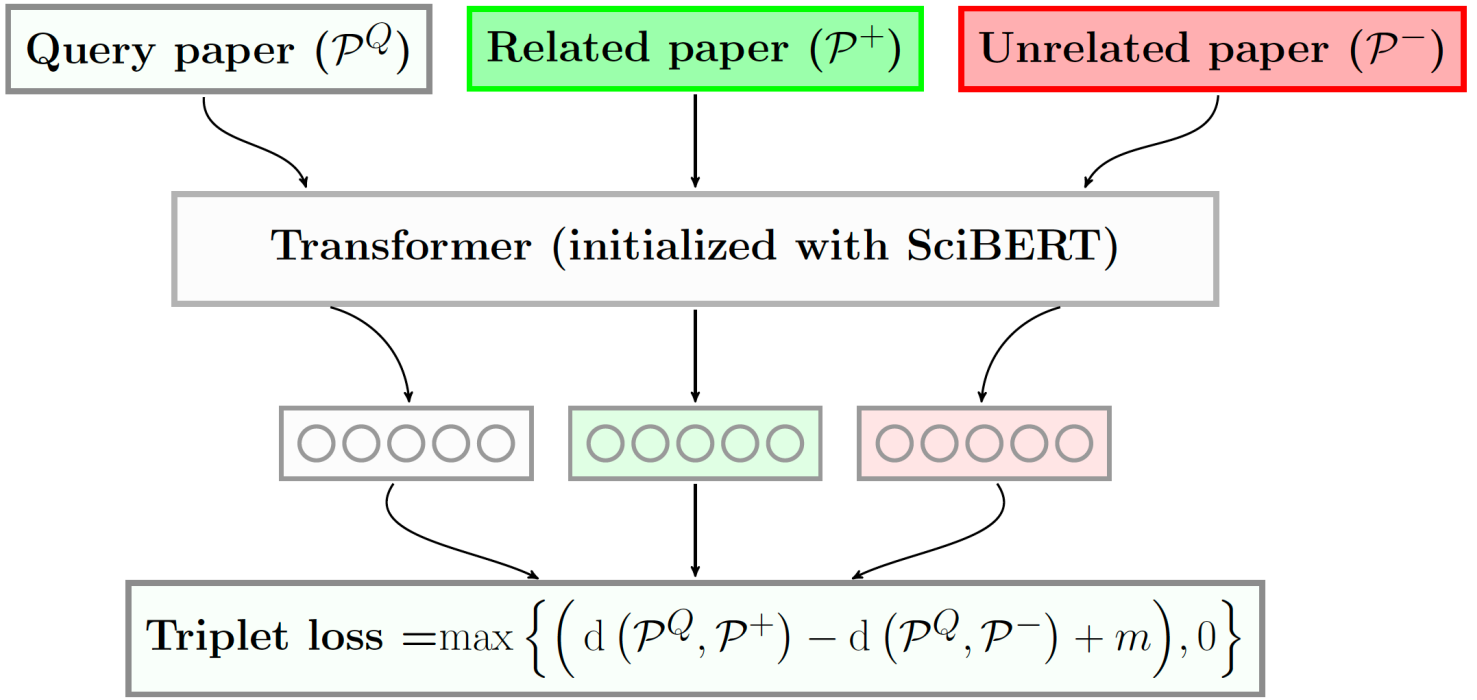
Method	Mathematics		Geology		Economics		Clothes		Sports	
	Macro-F1	**Micro-F1**	**Macro-F1**	**Micro-F1**	**Macro-F1**	**Micro-F1**	**Macro-F1**	**Micro-F1**	**Macro-F1**	**Micro-F1**
BERT	$18.14_{0.07}$	$22.04_{0.32}$	$21.97_{0.87}$	$29.63_{0.36}$	$14.17_{0.08}$	$19.77_{0.12}$	$45.10_{1.47}$	$68.54_{2.25}$	$31.88_{0.23}$	$34.58_{0.56}$
GraphFormers	$18.69_{0.52}$	$23.24_{0.46}$	$22.64_{0.92}$	$31.02_{1.16}$	$13.68_{1.03}$	$19.00_{1.44}$	$46.27_{1.92}$	$68.97_{2.46}$	$43.77_{0.63}$	$50.47_{0.78}$
SciBERT	$23.50_{0.64}$	$23.10_{2.23}$	$29.49_{1.25}$	$37.82_{1.89}$	$15.91_{0.48}$	$21.32_{0.66}$	-	-	-	-
SPECTER	$23.37_{0.07}$	$29.83_{0.96}$	$30.40_{0.48}$	$38.54_{0.77}$	$16.16_{0.17}$	$19.84_{0.47}$	-	-	-	-
SimCSE (unsup)	$20.12_{0.08}$	$26.11_{0.39}$	$38.78_{0.19}$	$38.55_{0.17}$	$14.54_{0.26}$	$19.07_{0.43}$	$42.70_{2.32}$	$58.72_{0.34}$	$41.91_{0.85}$	$59.19_{0.55}$
SimCSE (sup)	$20.39_{0.07}$	$25.56_{0.00}$	$25.66_{0.28}$	$33.89_{0.40}$	$15.03_{0.53}$	$18.64_{1.32}$	$52.82_{0.87}$	$75.54_{0.98}$	$46.69_{0.10}$	$59.19_{0.55}$
LinkBERT	$15.78_{0.91}$	$19.75_{1.19}$	$24.08_{0.58}$	$31.32_{0.04}$	$12.71_{0.12}$	$16.39_{0.22}$	$44.94_{2.52}$	$65.33_{4.34}$	$35.60_{0.33}$	$38.30_{0.09}$
BERT.MLM	$23.44_{0.39}$	$31.75_{0.58}$	$36.31_{0.36}$	$48.04_{0.69}$	$16.60_{0.21}$	$22.71_{1.16}$	$46.98_{0.84}$	$68.00_{0.84}$	$62.21_{0.13}$	$75.43_{0.74}$
SciBERT.MLM	$23.34_{0.42}$	$30.11_{0.97}$	$36.94_{0.28}$	$46.54_{0.40}$	$16.28_{0.38}$	$21.41_{0.81}$	-	-	-	-
SimCSE.in-domain	$25.15_{0.09}$	$29.85_{0.20}$	$38.91_{0.08}$	$48.93_{0.14}$	$18.08_{0.22}$	$23.79_{0.44}$	$57.03_{0.20}$	$80.16_{0.31}$	$65.57_{0.35}$	$75.22_{0.18}$
PATTON	$\mathbf{27.58}_{0.03}$	$\mathbf{32.82}_{0.01}$	$39.35_{0.06}$	$48.19_{0.15}$	$19.32_{0.05}$	$25.12_{0.05}$	$\mathbf{60.14}_{0.28}$	$\mathbf{84.88}_{0.09}$	$\mathbf{67.57}_{0.08}$	$\mathbf{78.60}_{0.15}$
SciPATTON	$27.35_{0.04}$	$31.70_{0.01}$	$\mathbf{39.65}_{0.10}$	$\mathbf{48.93}_{0.06}$	$\mathbf{19.91}_{0.08}$	$\mathbf{25.68}_{0.32}$	-	-	-	-
w/o NMLM	$25.91_{0.45}$	$27.79_{2.07}$	$38.78_{0.19}$	$48.48_{0.17}$	$18.86_{0.23}$	$24.25_{0.26}$	$56.68_{0.24}$	$80.27_{0.17}$	$65.83_{0.28}$	$76.24_{0.54}$
w/o MNP	$24.79_{0.65}$	$29.44_{1.50}$	$38.00_{0.73}$	$47.82_{1.06}$	$18.69_{0.59}$	$25.63_{1.44}$	$47.35_{1.20}$	$68.50_{2.60}$	$64.23_{1.53}$	$76.03_{1.67}$

Pre-training Strategy: Specter

- 1. After obtaining document representations, we have pre-training loss.

$$\mathcal{L} = \max\{\text{dist}(\boldsymbol{h}_{Q,CLS}, \boldsymbol{h}_{+,CLS}) - \text{dist}(\boldsymbol{h}_{Q,CLS}, \boldsymbol{h}_{-,CLS}) + m, 0\}$$

$$\text{dist}(\boldsymbol{h}_{Q,CLS}, \boldsymbol{h}_{+,CLS}) = \left|\left|\boldsymbol{h}_{Q,CLS} - \boldsymbol{h}_{+,CLS}\right|\right|_2$$



Section 2: PLM-based Text-Attributed Graph Models

- Section 2.1: PLMs for Static Text-Attributed Graph
- Section 2.2: PLMs for Heterogeneous Text-Attributed Graph
- Section 2.3: Textual-edge Text-Attributed Graph
- Section 2.4: Pre-training Strategy

Yale

Q & A

Yale University

SMU SINGAPORE MANAGEMENT UNIVERSITY

School of Computing and Information Systems

Text-Attributed Graph Representation Learning: Methods, Applications, and Challenges Section 3 (45 min)

Delvin Ce Zhang[1], Menglin Yang[1], Rex Ying[1], and Hady W. Lauw[2]

[1]Yale University, [2]Singapore Management University

Tutorial Outline

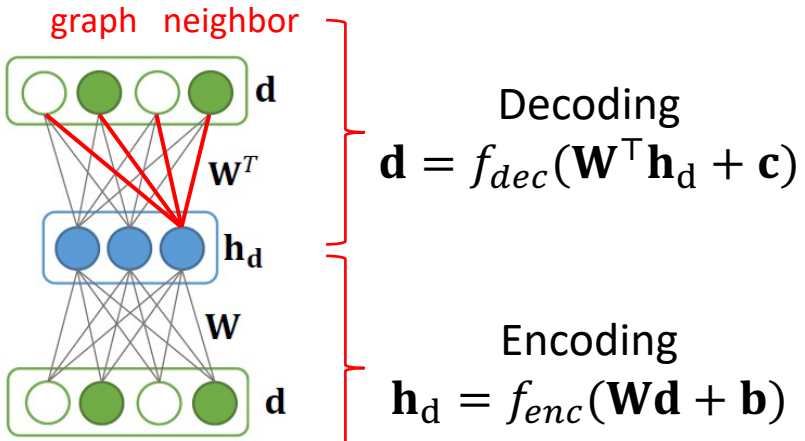
- Section 1: Text-Attributed Graph and Preliminaries (30 min)
- Section 2: PLM-based Text-Attributed Graph Models (45 min)
- Section 3: TM-based Text-Attributed Graph Models (45 min)
- Section 4: Applications, Challenges, and Future Directions (45 min)
- Summary and Q&A

Section 3: TM-based Text-Attributed Graph Models

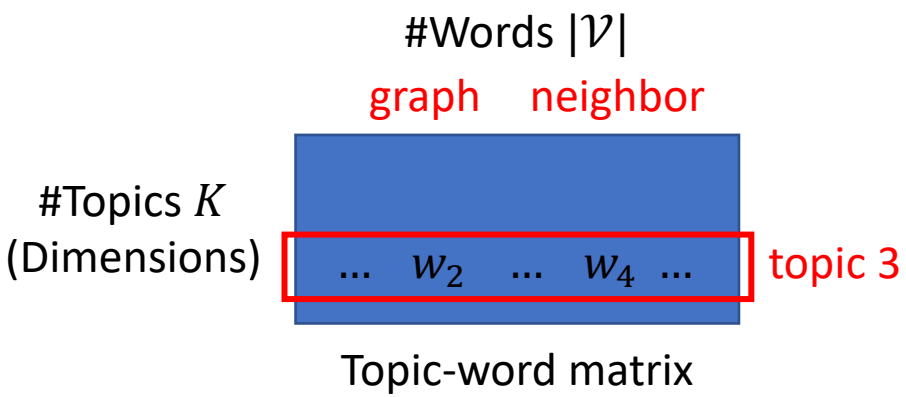
- Section 3.1: TMs for Static Text-Attributed Graph
- Section 3.2: TMs for Heterogeneous Text-Attributed Graph
- Section 3.3: Hierarchical Text-Attributed Graph

TMs for Static Text-Attributed Graph

- **Static Text-Attributed Graph (TAG) :** We observe the whole TAG at once.
- We present Adjacent-Encoder [1] and DBN [2].



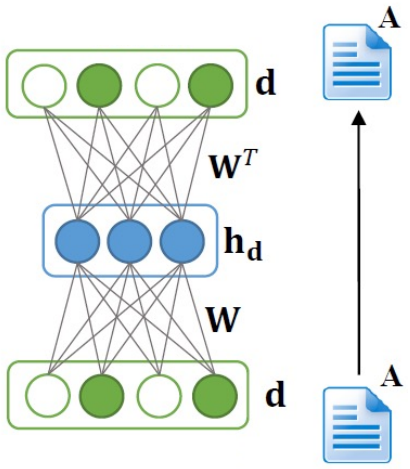
(a) Auto-Encoder



[1] Zhang, C., & Lauw, H. W. (2020, April). Topic modeling on document networks with adjacent-encoder. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 34, No. 04, pp. 6737-6745).
[2] Zhang, D. C., & Lauw, H. W. (2023). Topic Modeling on Document Networks with Dirichlet Optimal Transport Barycenter. IEEE Transactions on Knowledge and Data Engineering.

TMs for Static Text-Attributed Graph

- **Static Text-Attributed Graph (TAG) :** We observe the whole TAG at once.
- We present Adjacent-Encoder [1] and DBN [2].



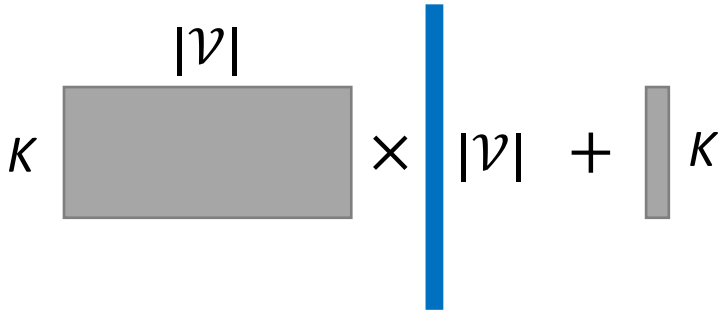
(a) Auto-Encoder

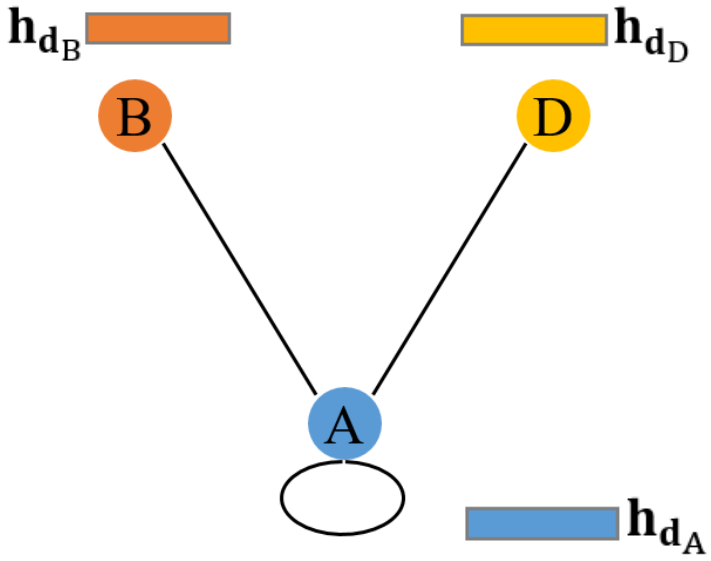
[1] Zhang, C., & Lauw, H. W. (2020, April). Topic modeling on document networks with adjacent-encoder. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 34, No. 04, pp. 6737-6745).
[2] Zhang, D. C., & Lauw, H. W. (2023). Topic Modeling on Document Networks with Dirichlet Optimal Transport Barycenter. IEEE Transactions on Knowledge and Data Engineering.

TMs for Static Text-Attributed Graph: Adjacent-Encoder

- 1. Encoding. Can be a MLP, GNN, or PLM.

$$\boldsymbol{h}_i = \tanh(\boldsymbol{W}\boldsymbol{d}_i + \boldsymbol{b})$$





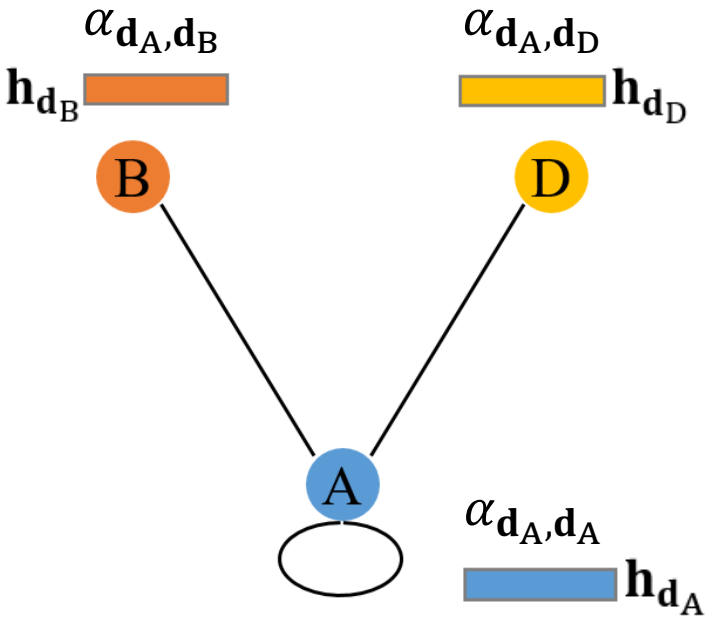
TMs for Static Text-Attributed Graph: Adjacent-Encoder

- 1. Encoding. Can be a MLP, GNN, or PLM.

$$\boldsymbol{h}_i = \tanh(\boldsymbol{W}\boldsymbol{d}_i + \boldsymbol{b})$$

- 2. Neighbor attention.

$$\hat{\alpha}_{ij} = \boldsymbol{h}_i^\top \boldsymbol{h}_j \qquad \alpha_{ij} = \frac{\exp(\hat{\alpha}_{ij})}{\sum_{j' \in \mathcal{N}(i)} \exp(\hat{\alpha}_{ij'})}$$



TMs for Static Text-Attributed Graph: Adjacent-Encoder

- 1. Encoding. Can be a MLP, GNN, or PLM.

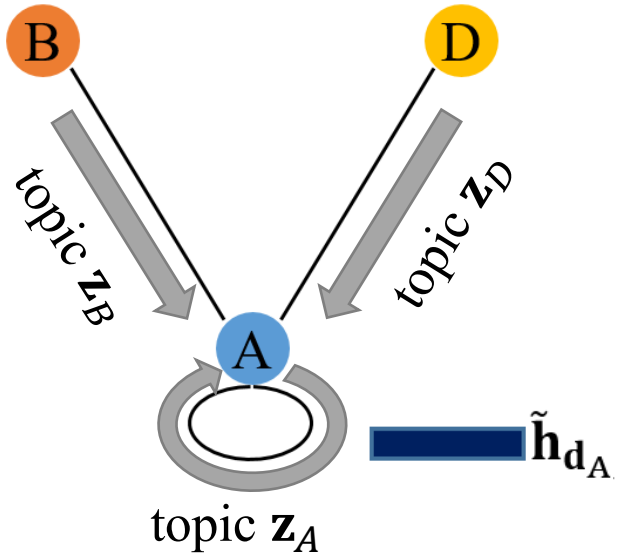
$$\boldsymbol{h}_i = \tanh(\boldsymbol{W}\boldsymbol{d}_i + \boldsymbol{b})$$

- 2. Neighbor attention.

$$\hat{\alpha}_{ij} = \boldsymbol{h}_i^\top \boldsymbol{h}_j \qquad \alpha_{ij} = \frac{\exp(\hat{\alpha}_{ij})}{\sum_{j' \in \mathcal{N}(i)} \exp(\hat{\alpha}_{ij'})}$$

- 3. Neighbor aggregation.

$$\tilde{\boldsymbol{h}}_i = \sum_{j \in \mathcal{N}(i)} \alpha_{ij} \boldsymbol{h}_j$$



TMs for Static Text-Attributed Graph: Adjacent-Encoder

- 1. Encoding. Can be a MLP, GNN, or PLM.

$$\boldsymbol{h}_i = \tanh(\boldsymbol{W}\boldsymbol{d}_i + \boldsymbol{b})$$

- 2. Neighbor attention.

$$\hat{\alpha}_{ij} = \boldsymbol{h}_i^\top \boldsymbol{h}_j \qquad \alpha_{ij} = \frac{\exp(\hat{\alpha}_{ij})}{\sum_{j' \in \mathcal{N}(i)} \exp(\hat{\alpha}_{ij'})}$$

- 3. Neighbor aggregation.

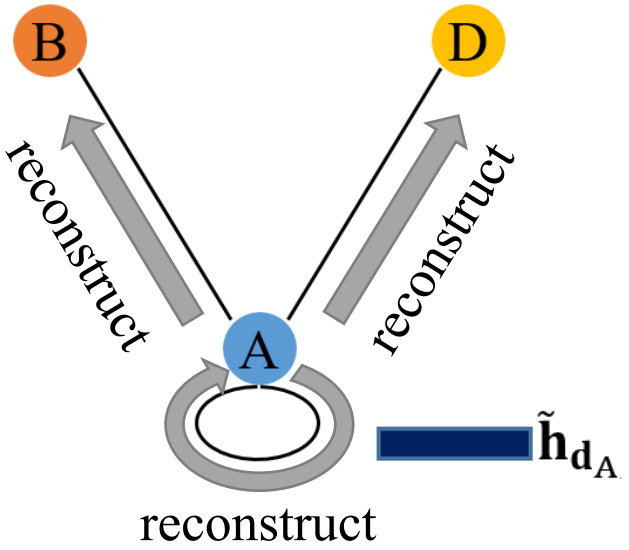
$$\tilde{\boldsymbol{h}}_i = \sum_{j \in \mathcal{N}(i)} \alpha_{ij} \boldsymbol{h}_j$$

- 4. Neighbor reconstruction.

$$\hat{\boldsymbol{d}}_i = \text{sigmoid}(\boldsymbol{W}^\top \tilde{\boldsymbol{h}}_i + \boldsymbol{c})$$

$$l(\mathbf{d}_j, \hat{\mathbf{d}}_i) = -\sum_{w \in \mathcal{V}} [d_{j,w} \log(\hat{d}_{i,w}) + (1 - d_{j,w}) \log(1 - \hat{d}_{i,w})]$$

↓ Neighbor document



TMs for Static Text-Attributed Graph: Adjacent-Encoder

- **Experiments**
- 1. Datasets

Name	#classes	#documents	#edges	vocabulary
DS	9	570	1336	3085
HA	6	223	515	2073
ML	7	1980	5748	4431
PL	9	1553	4851	4105

TMs for Static Text-Attributed Graph: Adjacent-Encoder

- **Experiments**
- 2. Document classification

Model	Document Classification			
	DS	HA	ML	PL
Adjacent-Encoder	**0.739**	**0.842**	**0.864**	**0.772**
AE	0.558	0.688	0.739	0.616
DAE	0.656	0.799	0.790	0.694
CAE	0.558	0.685	0.741	0.620
VAE	0.652	0.789	0.796	0.679
KSAE	0.537	0.672	0.710	0.581
KATE	0.628	0.808	0.762	0.651
ProdLDA	0.637	0.780	0.764	0.631
RTM	0.543	0.637	0.663	0.574
PLANE	0.690	0.799	0.750	0.648
NRTM	0.591	0.816	0.549	0.503
VGAE	0.671	0.827	0.807	0.718

Models w/o graph

Models with graph

TMs for Static Text-Attributed Graph: Adjacent-Encoder

- **Experiments**
- 3. Topic coherence

	Topic	Adjacent-Encoder
MDP	1	maze, markov, mdp, observable, minute, intractable, severe, markovian, pomdp, analog
KNN	2	move, 0-1, image, nearest, promoter, neighbor, grid, k-nearest, analogy, sketch
RL	3	reward, influence, recurrent, credit, exploratory, max, reinforcement, net, reactive, policy
Markov chain	4	inference, hmm, graphical, practitioner, translation, defense, methodological, causal, probable, assist
Neural network	5	pair, net, coordination, backpropagation, stronger, broad, network, classic, pendulum, multiclass

TMs for Static Text-Attributed Graph: Adjacent-Encoder

- **Experiments**
- 3. Topic coherence

$$\text{PMI} = \log \frac{p(w_i, w_j)}{p(w_i)p(w_j)}$$

Total number of co-occurrences

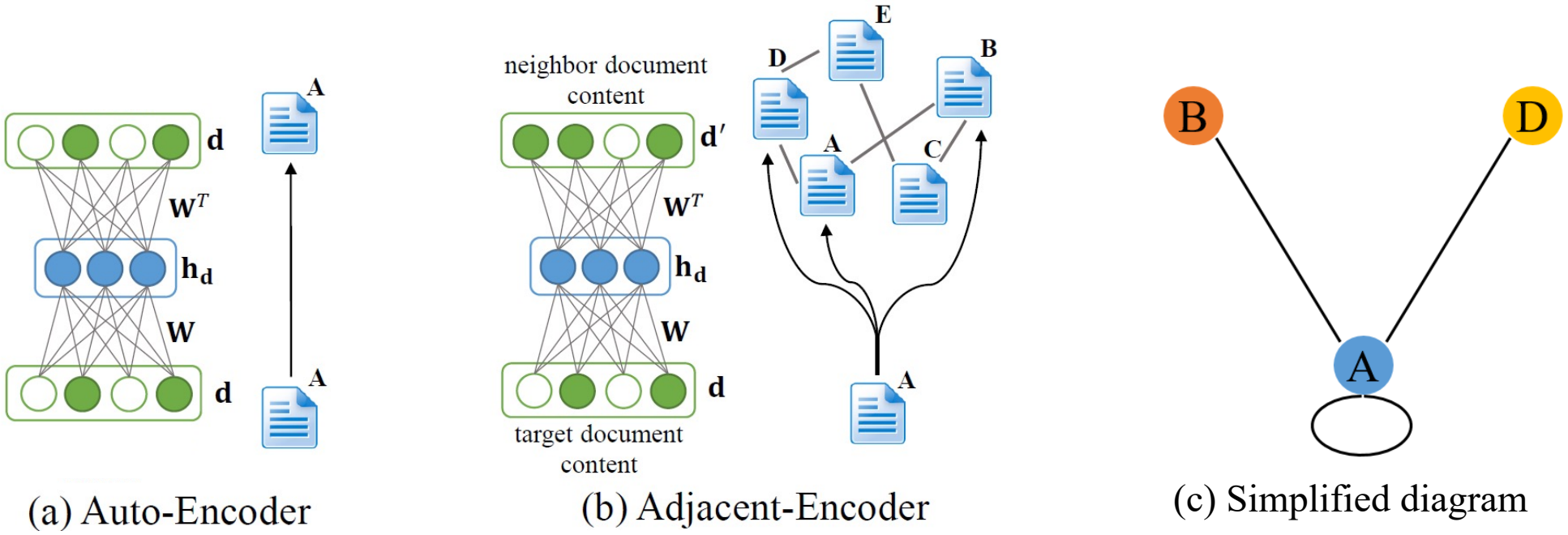
Number of occurrence of each word w_i

If two words w_i and w_j discuss the same topic, they should have high co-occurrence $p(w_i, w_j)$

Model	PMI			
	DS	HA	ML	PL
Adjacent-Encoder	**2.360**	**2.054**	**2.180**	**2.499**
AE	0.294	0.446	0.665	0.969
DAE	1.170	1.125	1.203	1.553
CAE	0.348	0.558	0.526	0.684
VAE	0.685	0.793	1.831	1.132
KSAE	0.547	0.285	0.770	0.759
KATE	1.312	1.755	1.619	2.003
ProdLDA	1.638	1.315	1.837	2.088
RTM	1.279	1.678	1.199	1.615
PLANE	1.585	1.847	1.756	2.099
NRTM	1.533	2.041	1.328	1.632

TMs for Static Text-Attributed Graph

- **Static Text-Attributed Graph (TAG) :** We observe the whole TAG at once.
- We present Adjacent-Encoder [1] and DBN [2].



[1] Zhang, C., & Lauw, H. W. (2020, April). Topic modeling on document networks with adjacent-encoder. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 34, No. 04, pp. 6737-6745).

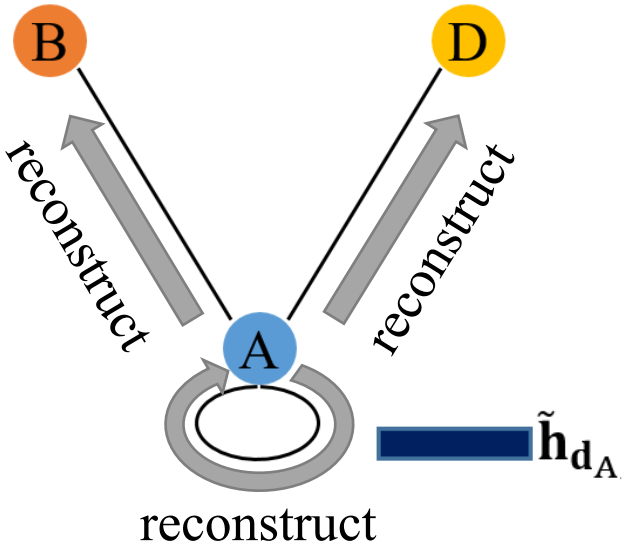
[2] Zhang, D. C., & Lauw, H. W. (2023). Topic Modeling on Document Networks with Dirichlet Optimal Transport Barycenter. IEEE Transactions on Knowledge and Data Engineering.

TMs for Static Text-Attributed Graph: DBN

- The difference between Adjacent-Encoder and DBN is their decoders.
- Adjacent-Encoder uses MLP, while DBN uses Optimal Transport (OT).

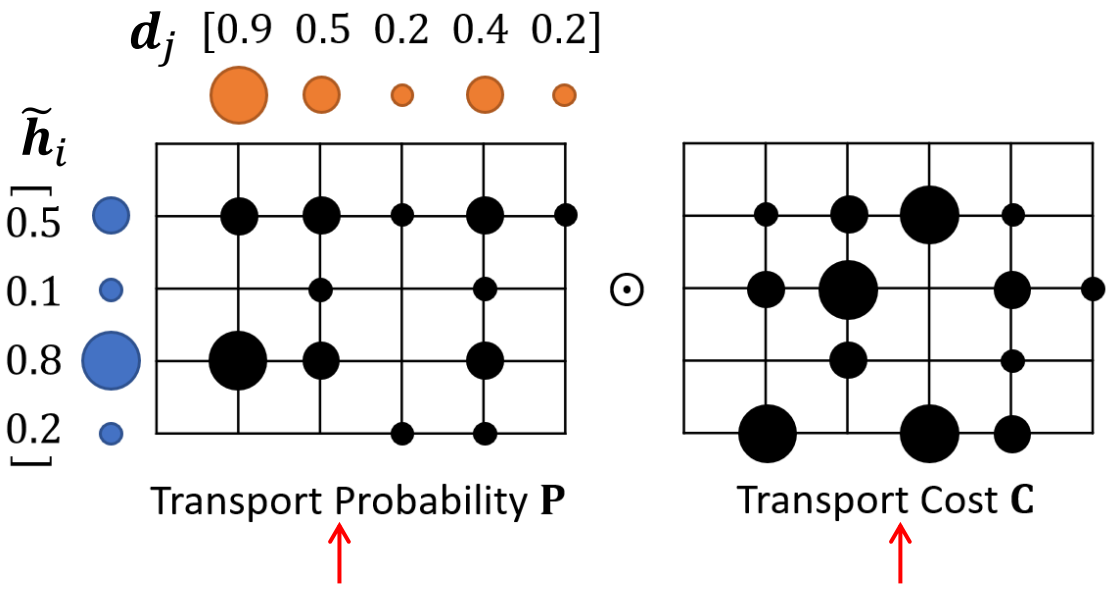
$$\min \sum_{j \in \mathcal{N}(i)} \alpha_{ij} d_{\mathrm{OT}}(\tilde{\boldsymbol{h}}_i, \boldsymbol{d}_j)$$

Optimal Transport distance



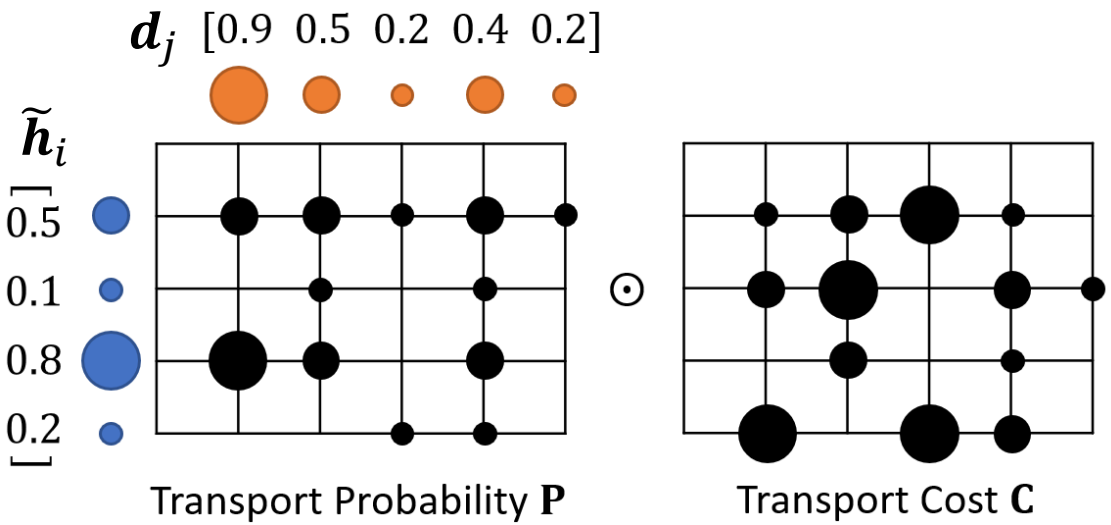
TMs for Static Text-Attributed Graph: DBN

- The difference between Adjacent-Encoder and DBN is their decoders.
- Adjacent-Encoder uses MLP, while DBN uses Optimal Transport (OT).
- OT measures the distance between two probability distributions.



TMs for Static Text-Attributed Graph: DBN

- The difference between Adjacent-Encoder and DBN is their decoders.
- Adjacent-Encoder uses MLP, while DBN uses Optimal Transport (OT).
- OT measures the distance between two probability distributions.



$$= \min_{\mathbf{P}} \sum_{i=1}^{K} \sum_{j=1}^{|\mathcal{V}|} p_{ij} c_{ij} = \mu(\tilde{\mathbf{z}}_D, \mathbf{d}_G)$$

Optimal Transport distance

$$c_{ij} = 1 - \cos(\mathbf{g}_i, \mathbf{e}_j)$$

Trainable topic embedding

Pre-trained word embedding

TMs for Static Text-Attributed Graph: Adjacent-Encoder

- **Experiments**
- 1. Datasets

Name	#Documents	#Links	Vocabulary	#Labels	#Words/doc
DS	1,703	3,234	3,134	9	58.0
ML	3,087	8,573	3,040	7	64.2
PL	2,597	7,754	3,106	9	64.0
Aminer	42,564	40,269	4,094	10	6.5
Web	445,657	565,502	10,015	N.A.	79.8

TMs for Static Text-Attributed Graph: Adjacent-Encoder

- **Experiments**
- 2. Document classification

Models w/o graph

Optimal transport is better

Models with graph

Model	Micro F1 score			
	DS	ML	PL	Aminer
ProdLDA	51.4±1.1	67.0±0.6	51.8±0.6	40.5±0.0
DVAE	54.7±2.2	68.9±0.6	55.7±1.3	66.1±0.5
ETM	42.2±2.4	53.9±1.8	45.0±2.1	53.2±0.7
WLDA	34.0±3.6	32.4±1.1	30.5±3.1	47.8±2.1
NSTM	42.5±1.8	46.3±1.8	42.0±1.6	48.4±0.3
GTM	39.1±1.6	45.9±1.4	40.8±1.6	65.2±0.2
RTM	50.1±2.3	63.7±0.9	52.3±0.7	53.0±0.7
NRTM	38.2±1.2	45.5±1.2	39.2±1.5	47.7±3.9
Adjacent-Encoder	58.8±1.2	72.8±0.6	60.0±1.7	59.5±0.2
LANTM	56.8±2.4	71.8±1.0	62.6±1.3	N.A.
GTNN	52.9±1.4	68.6±1.1	59.5±2.3	65.5±0.4
GRTM	56.5±1.9	51.7±2.3	45.2±1.0	70.7±0.2
GNCTM	63.7±1.4	77.8±1.1	65.5±3.8	69.3±1.2
NetDTM	62.3±1.0	74.8±1.0	63.0±1.1	68.1±0.1
VGATM	63.6±2.1	74.8±1.3	64.3±0.9	57.2±0.7
VGAE	39.8±2.0	56.6±1.7	47.6±3.4	64.7±0.5
PGCL	62.9±1.2	74.9±1.2	64.7±1.1	69.7±0.4
DBN	**66.2±1.4**	78.4±0.8	67.3±0.5	71.2±0.4

TMs for Static Text-Attributed Graph: Adjacent-Encoder

- **Experiments**
- 2. Topic coherence

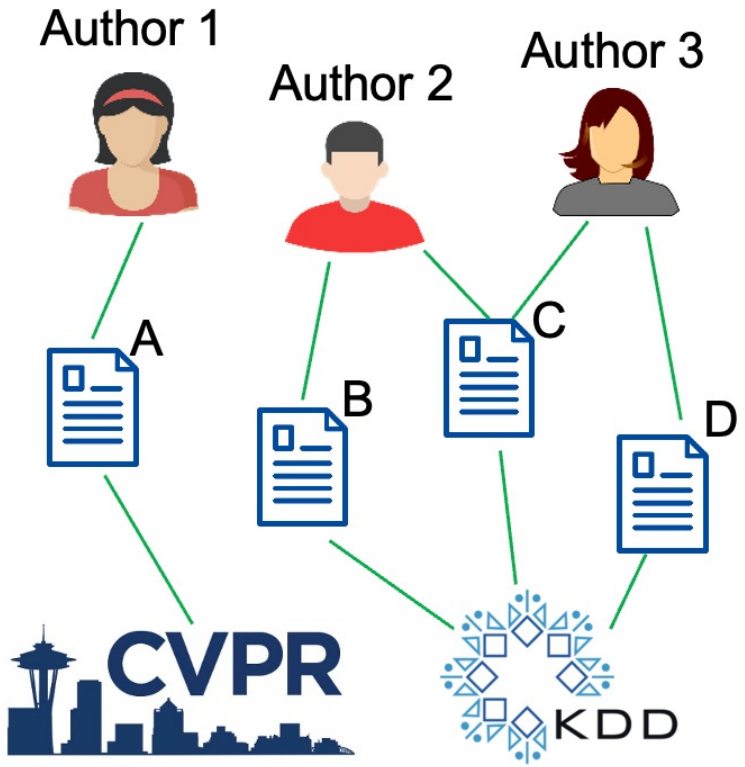
Model	Topic Coherence NPMI				
	DS	ML	PL	Aminer	Web
ProdLDA	10.5±0.3	10.9±0.7	12.1±0.7	8.9±0.0	21.2±0.2
DVAE	15.5±0.2	14.7±0.1	15.0±0.1	13.7±0.1	17.6±0.2
ETM	7.3±0.2	7.1±0.2	8.7±0.1	5.4±0.3	16.4±0.6
WLDA	8.7±0.3	9.8±0.3	11.7±0.4	14.0±1.6	23.9±0.8
NSTM	19.0±1.0	17.2±0.7	19.2±0.7	**24.0±0.3**	27.9±0.5
RTM	7.6±0.3	7.1±0.3	9.3±0.2	4.7±0.3	20.9±0.4
NRTM	8.2±0.5	9.4±0.1	10.9±0.5	6.1±0.8	26.1±0.3
Adjacent-Encoder	12.0±0.2	9.9±0.9	11.3±0.9	3.5±0.5	15.2±0.1
LANTM	6.4±0.5	5.4±0.3	7.2±0.8	N.A.	N.A.
GTNN	9.9±1.5	7.2±0.6	5.8±0.6	7.6±0.6	7.7±1.7
DBN	**22.7±0.4**	**21.5±0.6**	**20.9±0.5**	23.6±0.4	28.5±0.2

Section 3: TM-based Text-Attributed Graph Models

- Section 3.1: TMs for Static Text-Attributed Graph
- Section 3.2: TMs for Heterogeneous Text-Attributed Graph
- Section 3.3: Hierarchical Text-Attributed Graph

TMs for Heterogeneous Text-Attributed Graph

- Documents have authors and publication venues.
  - Papers written by the same authors discuss similar research topics;
  - News articles edited by the same journalists report similar events.



TMs for Heterogeneous Text-Attributed Graph

- Documents have authors and publication venues.
  - Papers written by the same authors discuss similar research topics;
  - News articles edited by the same journalists report similar events.
- **Challenges**
  - Existing works for text-attributed graph ignore authorship and venues.
- We present VGATM [1] in this section.

[1] Zhang, D. C., & Lauw, H. W. (2022, August). Variational graph author topic modeling. In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (pp. 2429-2438).

TMs for Heterogeneous Text-Attributed Graph

- Given a corpus $\mathcal{C} = \{\mathcal{D}, \mathcal{A}, \mathcal{V}\}$, we construct a graph.

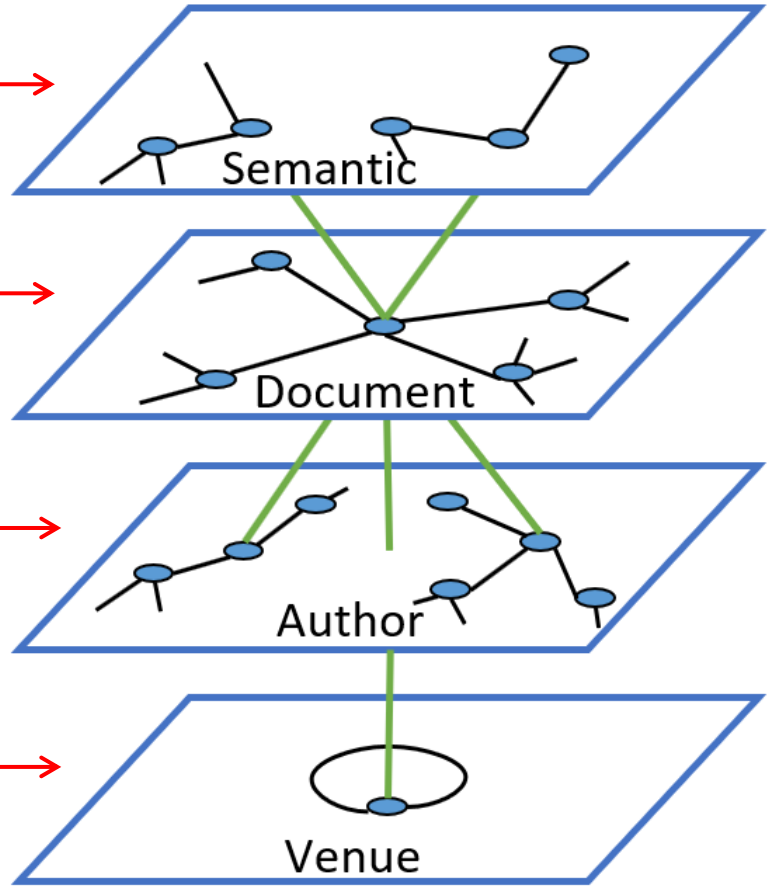
Documents Authors Venues

How to construct word-layer graph?

Paper citations, Webpage hyperlinks, …

Co-authorship links

Self-loop links



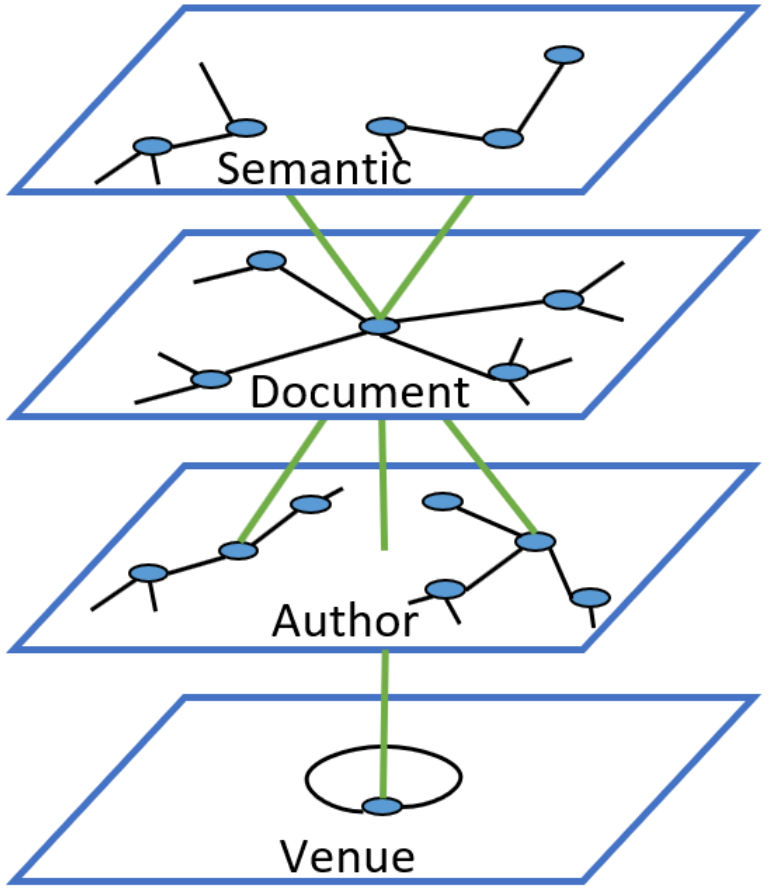
TMs for Heterogeneous Text-Attributed Graph

- Given a corpus $\mathcal{C} = \{\mathcal{D}, \mathcal{A}, \mathcal{V}\}$, we construct a graph.
- Semantic word layer

$$S_{sem} = \cos(g(w_i), g(w_j))$$

Pre-trained word embedding of word w_i

- For each w_i, its top-5 scores with w_j are neighbors.



TMs for Heterogeneous Text-Attributed Graph

- Given a corpus $\mathcal{C} = \{\mathcal{D}, \mathcal{A}, \mathcal{V}\}$, we construct a graph.
- Semantic word layer

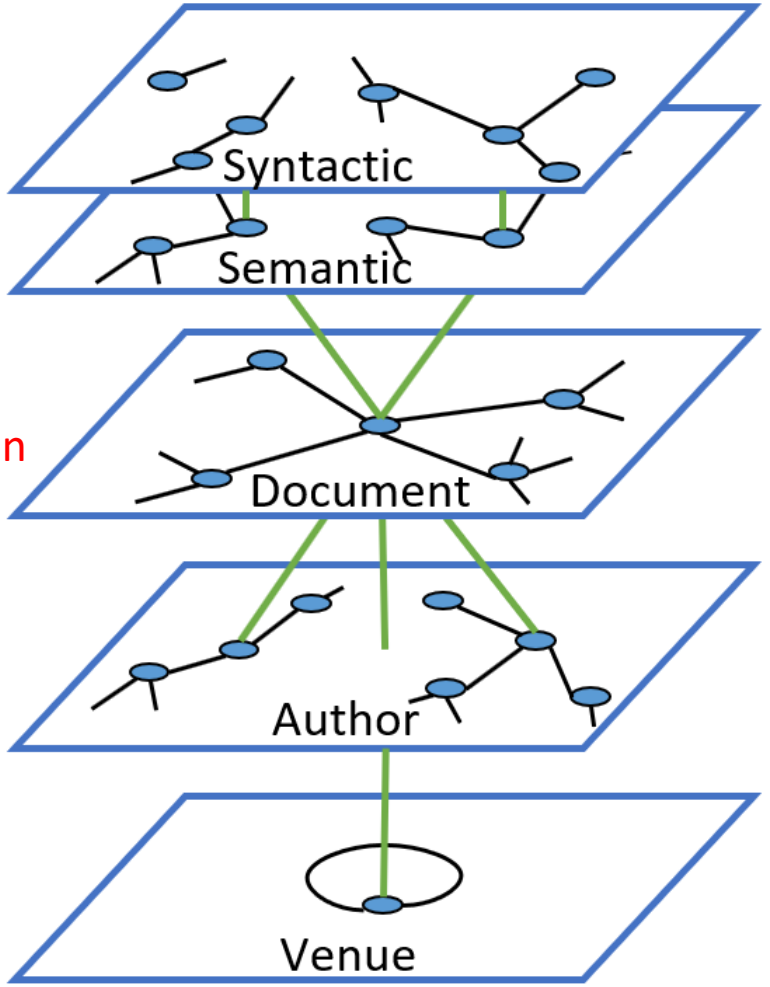
$$S_{sem} = \cos(g(w_i), g(w_j))$$

- Syntactic word layer

$$S_{syn} = \frac{N_{syn}(w_i, w_j)}{N_{co\text{-}occur}(w_i, w_j)}$$

Number of times that w_i and w_j have syntactic relation

Total number of co-occurrences



TMs for Heterogeneous Text-Attributed Graph

- Given a corpus $\mathcal{C} = \{\mathcal{D}, \mathcal{A}, \mathcal{V}\}$, we construct a graph.
- Semantic word layer

$$S_{sem} = \cos(g(w_i), g(w_j))$$

- Syntactic word layer

$$S_{syn} = \frac{N_{syn}(w_i, w_j)}{N_{co\text{-}occur}(w_i, w_j)}$$

- Contextual word layer

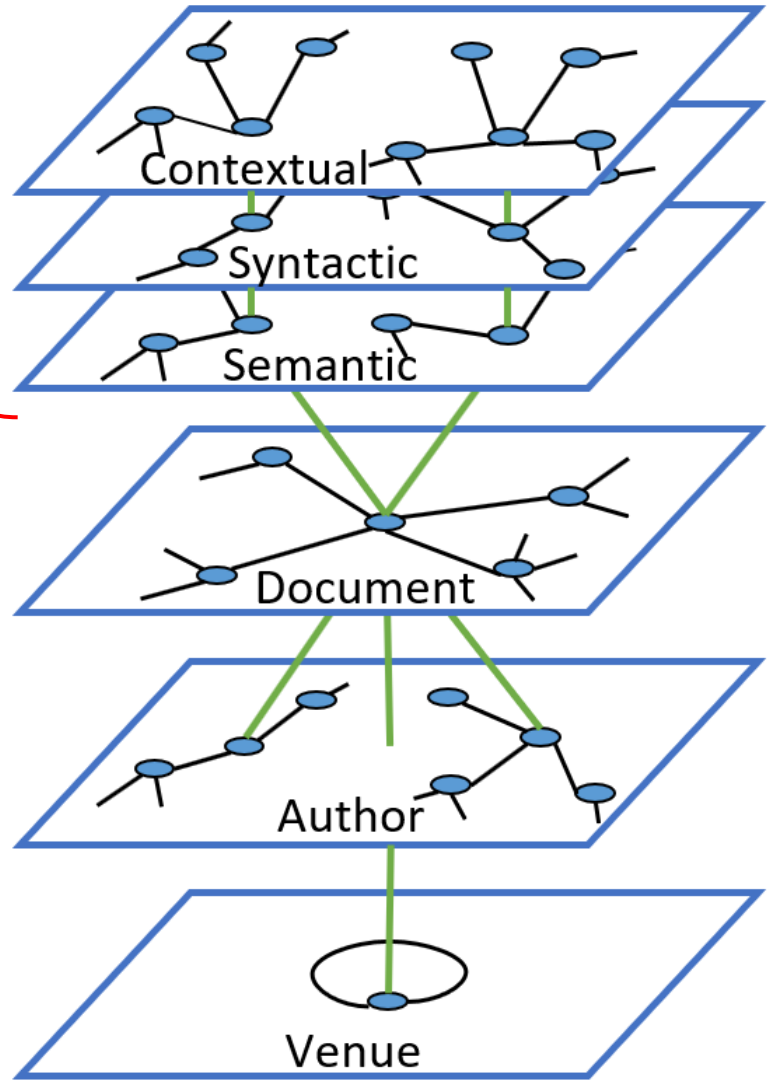
$$S_{ctx}(w_i, w_j) = \log \frac{p(w_i, w_j)}{p(w_i)p(w_j)}$$

Total number of co-occurrences

Number of occurrence of each word w_i

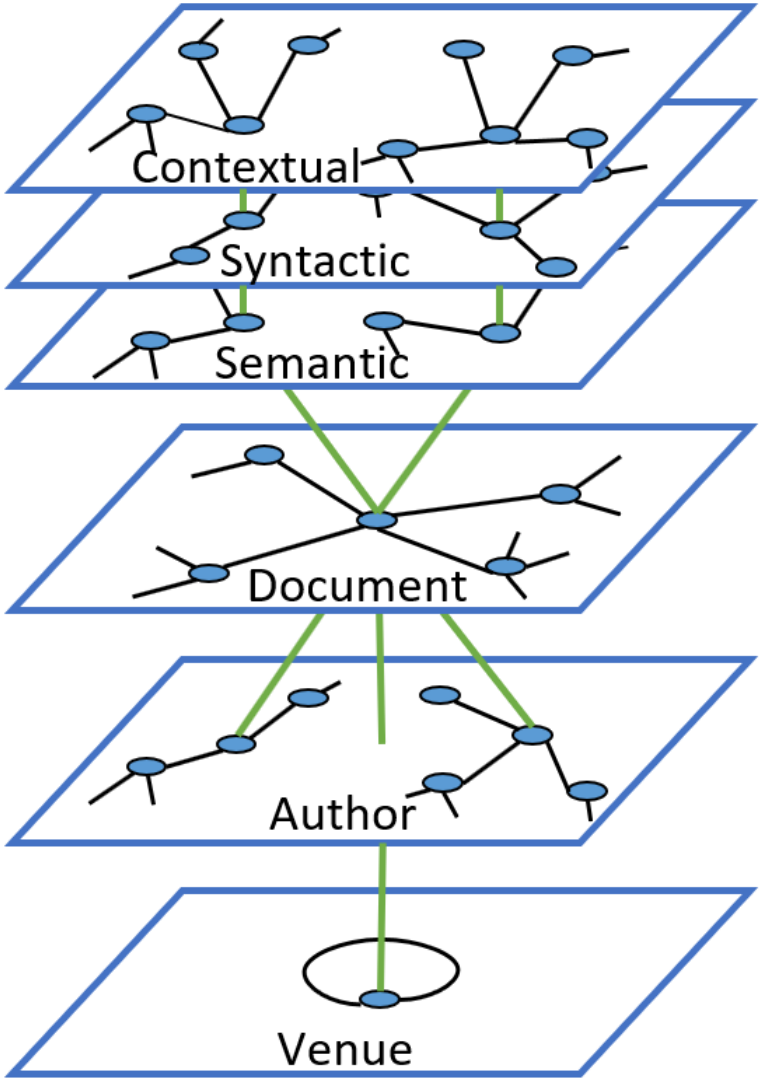
Text graph with 4 **layers**

3 **word sub-layers**



TMs for Heterogeneous Text-Attributed Graph

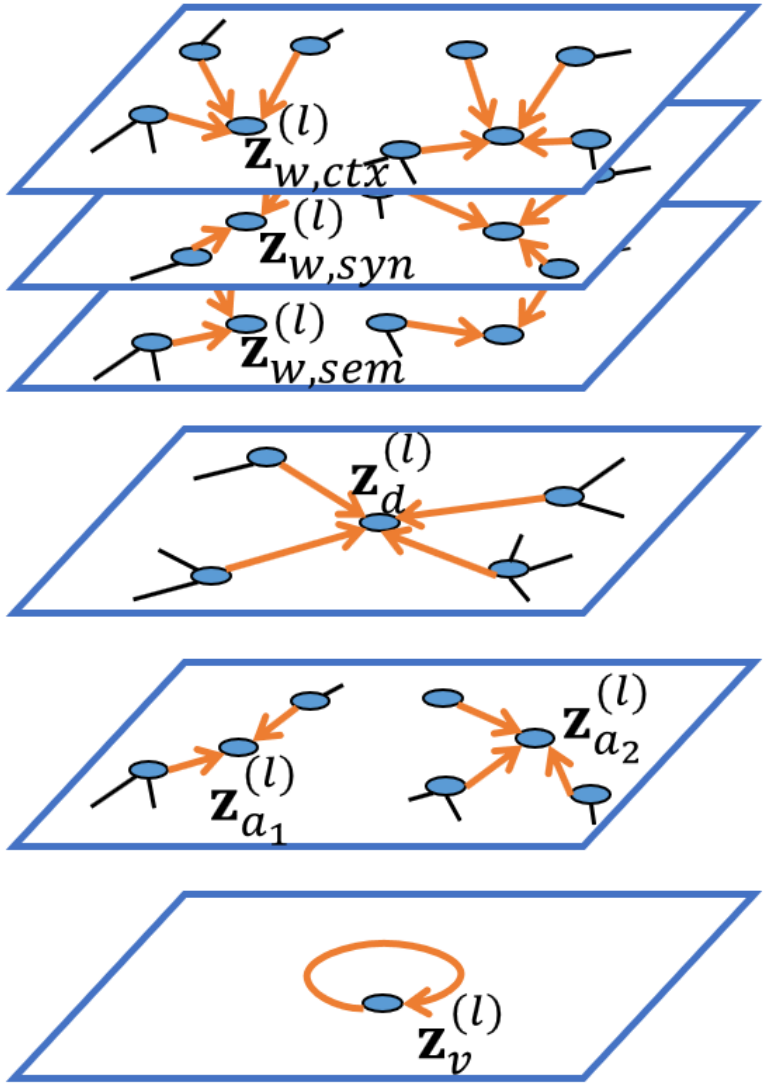
- 1. Graph Convolutional Encoder
- 2. Graph Decoder



TMs for Heterogeneous Text-Attributed Graph

- **Intra-layer embedding propagation**
- Linear transformation.

$$\tilde{\mathbf{z}}_i^{(l)} = \mathbf{W}_o^{(l)} \mathbf{z}_i^{(l-1)}$$



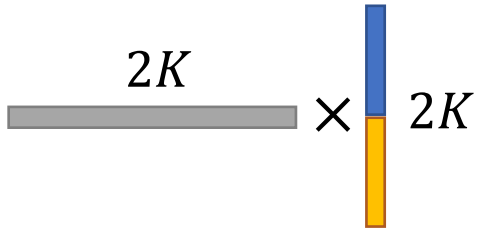
TMs for Heterogeneous Text-Attributed Graph

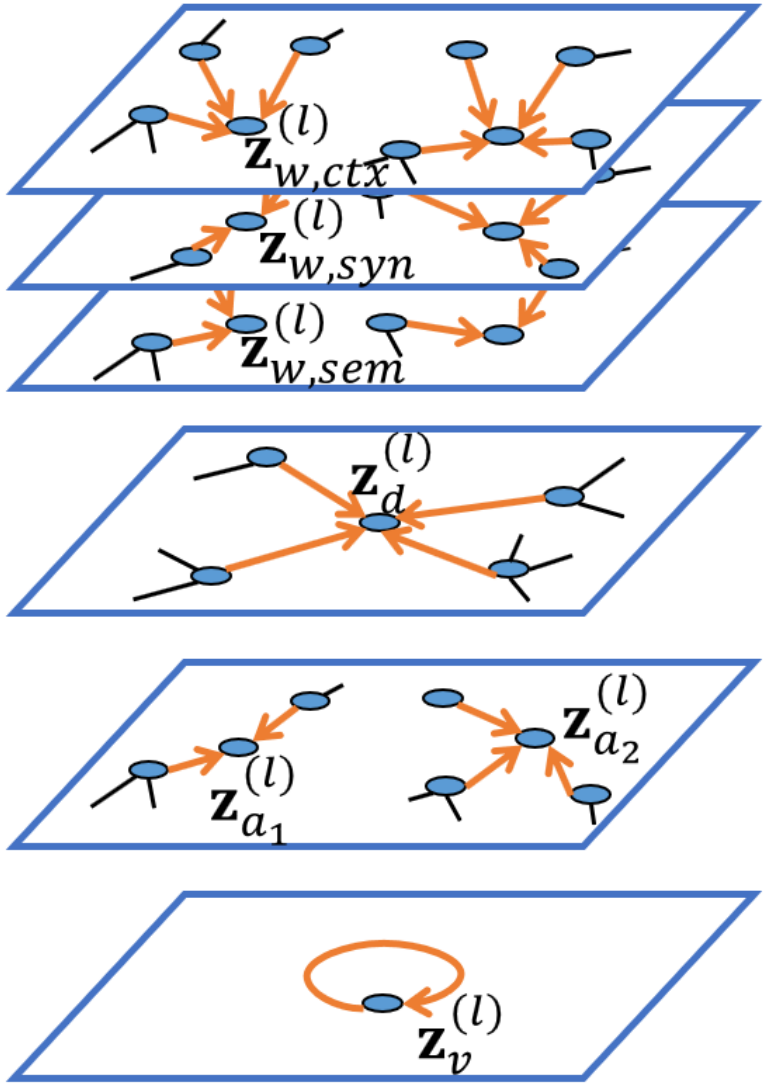
- **Intra-layer embedding propagation**
- Linear transformation.

$$\tilde{\mathbf{z}}_i^{(l)} = \mathbf{W}_o^{(l)} \mathbf{z}_i^{(l-1)}$$

- Intra-layer neighbor attention.

$$\alpha_{ij} = \text{softmax}\Big(\text{LeakyReLU}(\mathbf{b}_o^{(l)\top}[\tilde{\mathbf{z}}_i^{(l)} || \tilde{\mathbf{z}}_j^{(l)}])\Big), \quad j \in \mathcal{N}_o(i)$$





TMs for Heterogeneous Text-Attributed Graph

- **Intra-layer embedding propagation**
- Linear transformation.

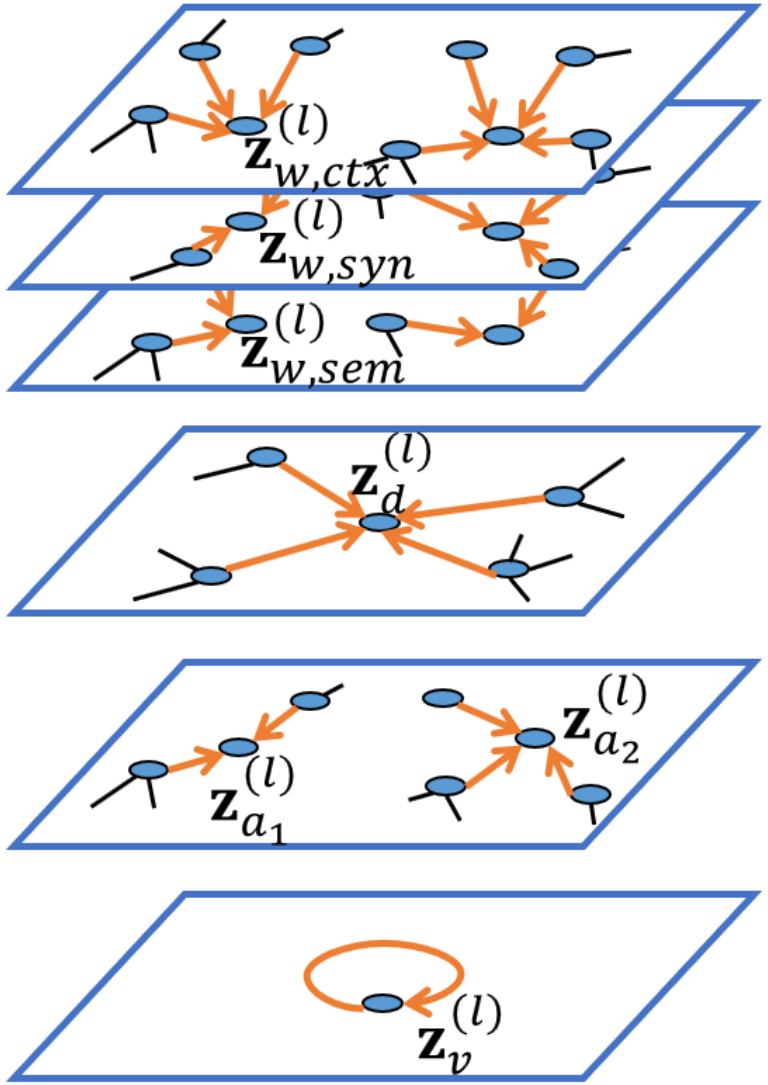
$$\tilde{\mathbf{z}}_i^{(l)} = \mathbf{W}_o^{(l)} \mathbf{z}_i^{(l-1)}$$

- Intra-layer neighbor attention.

$$\alpha_{ij} = \text{softmax}\Big(\text{LeakyReLU}(\mathbf{b}_o^{(l)\top}[\tilde{\mathbf{z}}_i^{(l)} || \tilde{\mathbf{z}}_j^{(l)}])\Big), \quad j \in \mathcal{N}_o(i)$$

- Intra-layer embedding propagation.

$$\mathbf{z}_i^{(l)} = \tanh\Big(\frac{1}{2}(\tilde{\mathbf{z}}_i^{(l)} + \sum_{j \in \mathcal{N}_o(i)} \alpha_{ij} \tilde{\mathbf{z}}_j^{(l)})\Big)$$



TMs for Heterogeneous Text-Attributed Graph

- **Intra-layer embedding propagation**
- Linear transformation.

$$\tilde{\mathbf{z}}_i^{(l)} = \mathbf{W}_o^{(l)} \mathbf{z}_i^{(l-1)}$$

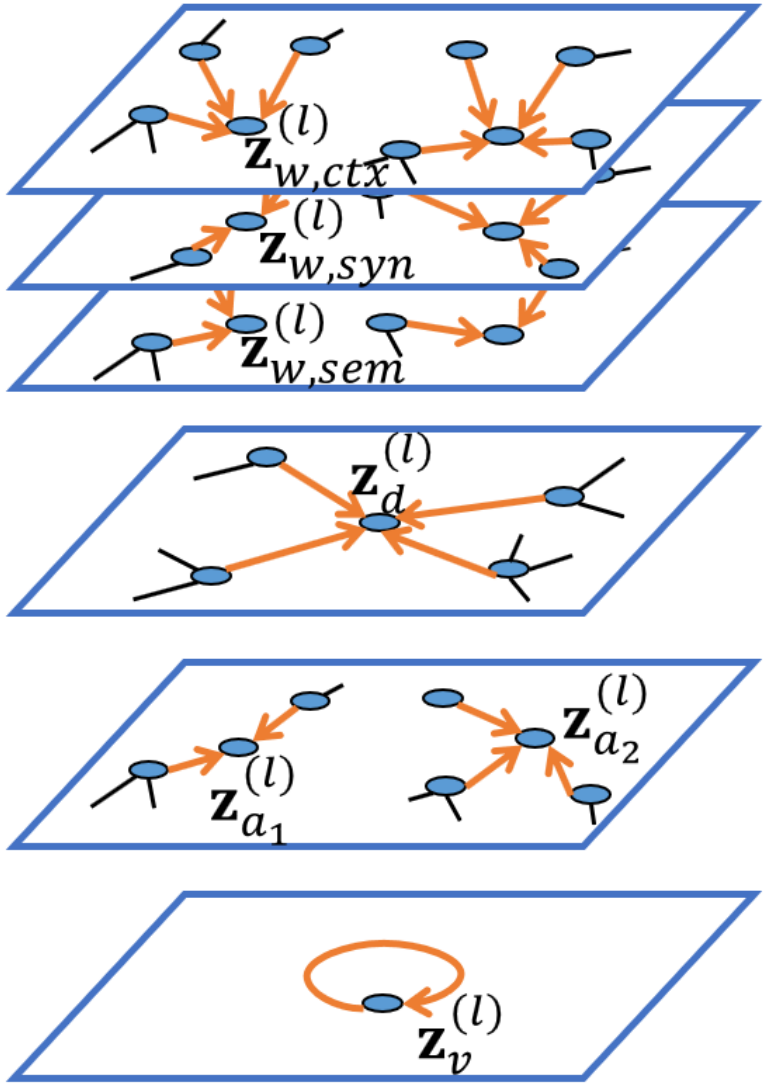
- Intra-layer neighbor attention.

$$\alpha_{ij} = \text{softmax}\Big(\text{LeakyReLU}(\mathbf{b}_o^{(l)\top}[\tilde{\mathbf{z}}_i^{(l)} || \tilde{\mathbf{z}}_j^{(l)}])\Big), \quad j \in \mathcal{N}_o(i)$$

- Intra-layer embedding propagation.

$$\mathbf{z}_i^{(l)} = \tanh\Big(\frac{1}{2}(\tilde{\mathbf{z}}_i^{(l)} + \sum_{j \in \mathcal{N}_o(i)} \alpha_{ij} \tilde{\mathbf{z}}_j^{(l)})\Big)$$

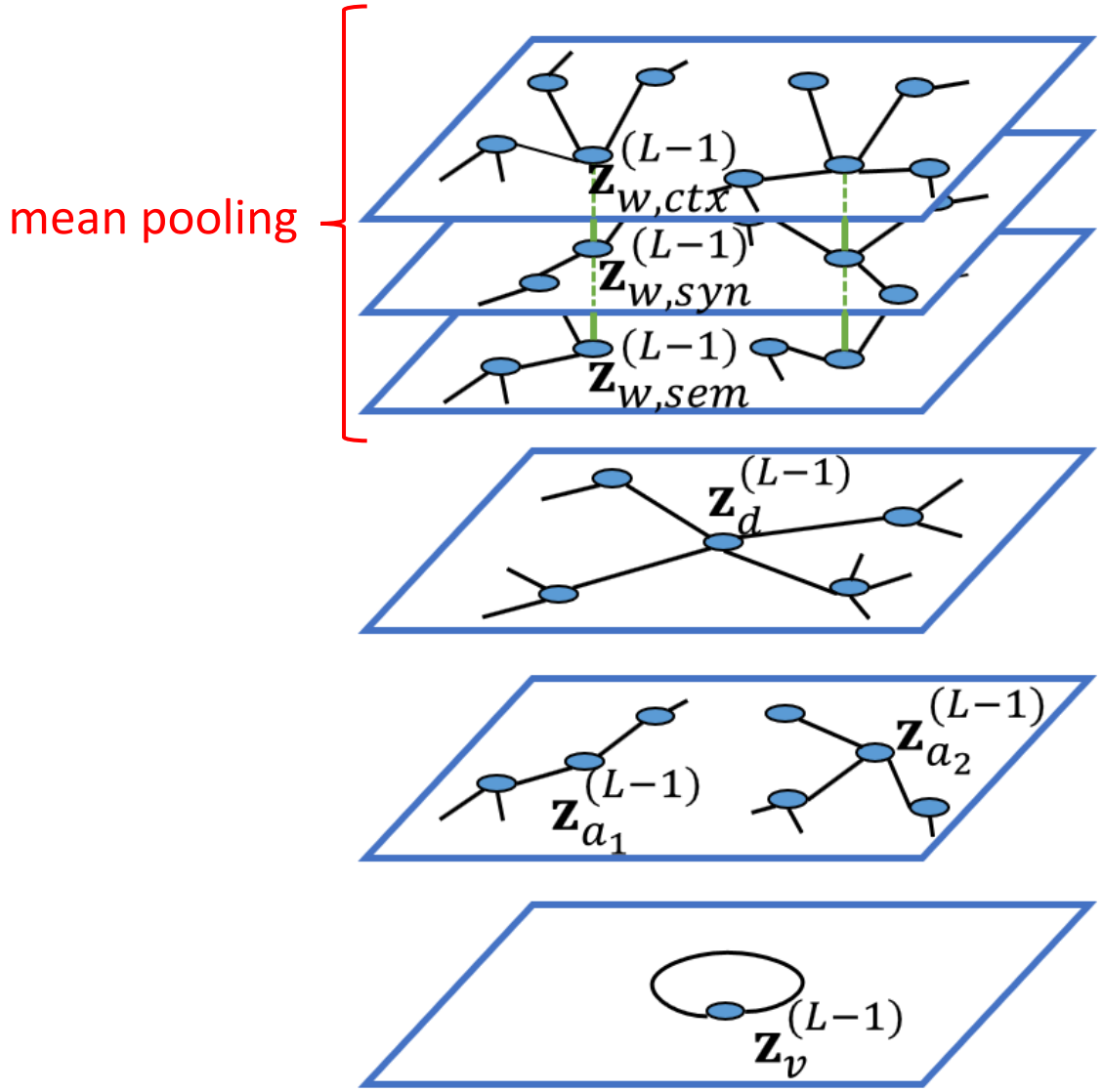
- Repeat above process for $(L-1)$ times.



TMs for Heterogeneous Text-Attributed Graph

- **Cross-layer embedding propagation**
- Cross-word-layer mean pooling.

$$\mathbf{z}_w^{(L-1)} = \text{mean}(\mathbf{z}_{w,ctx}^{(L-1)}, \mathbf{z}_{w,syn}^{(L-1)}, \mathbf{z}_{w,sem}^{(L-1)})$$



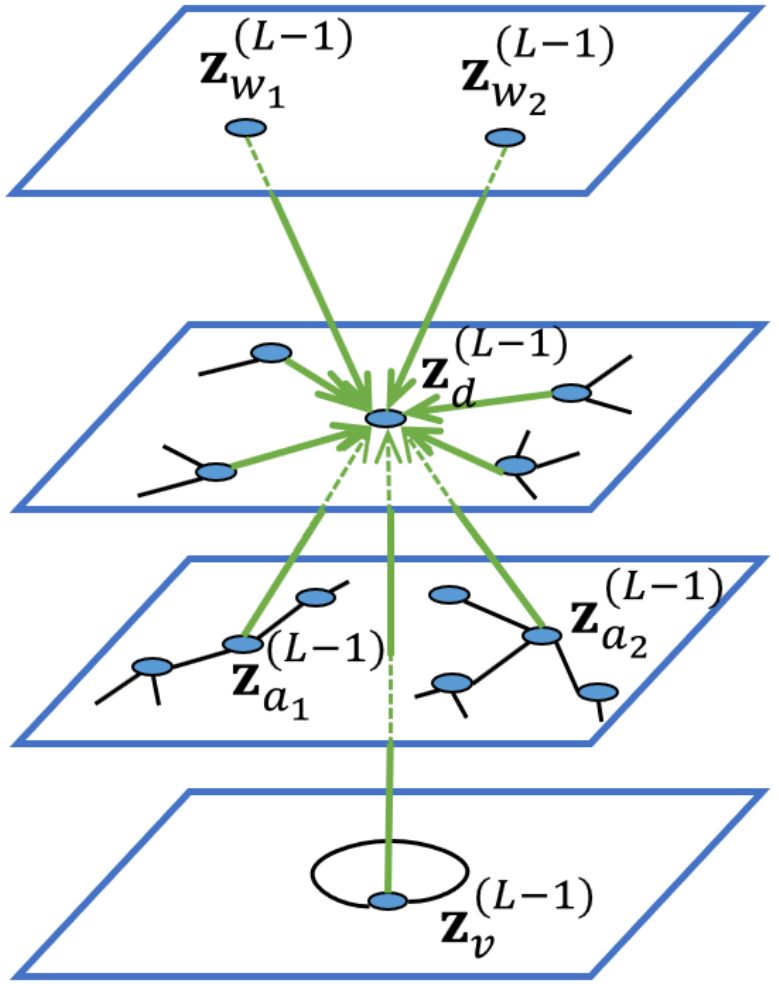
TMs for Heterogeneous Text-Attributed Graph

- **Cross-layer embedding propagation**
- Cross-word-layer mean pooling.

$$\mathbf{z}_w^{(L-1)} = \text{mean}(\mathbf{z}_{w,ctx}^{(L-1)}, \mathbf{z}_{w,syn}^{(L-1)}, \mathbf{z}_{w,sem}^{(L-1)})$$

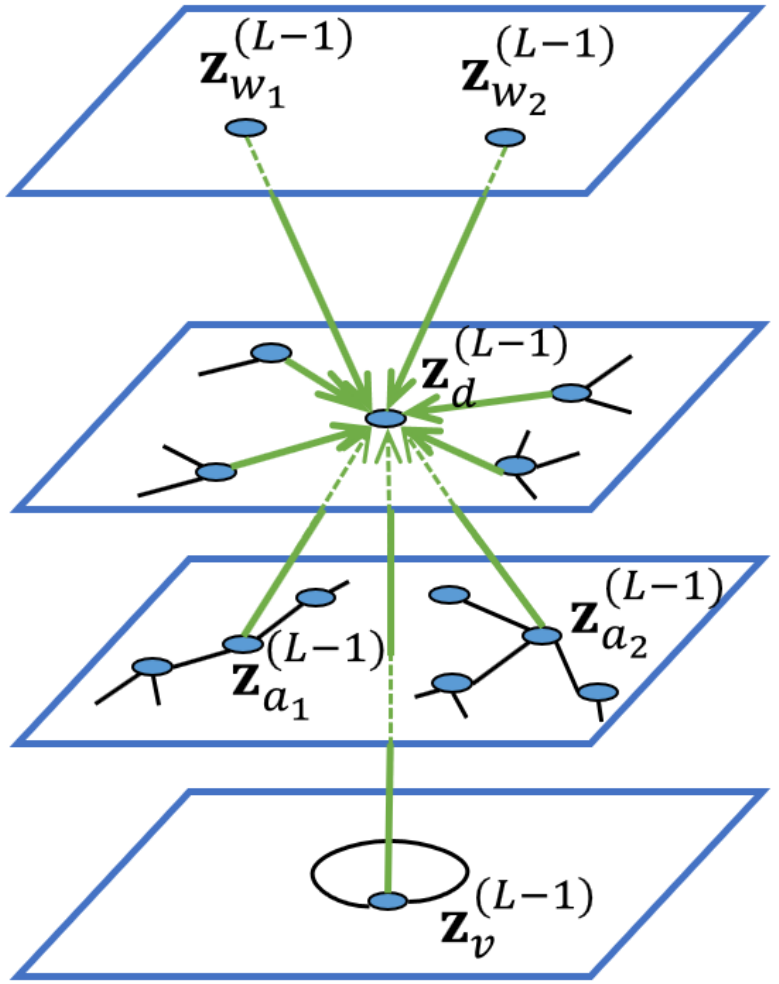
- Cross-layer embedding propagation.

$$\boldsymbol{\mu}_d = (1-\eta) \times \frac{1}{2}(\tilde{\mathbf{z}}_d^{(L)} + \tilde{\mathbf{h}}_d^{(L)}) + \eta \times \text{mean}(\tilde{\mathbf{h}}_w^{(L)}, \tilde{\mathbf{h}}_a^{(L)}, \tilde{\mathbf{h}}_v^{(L)})$$



TMs for Heterogeneous Text-Attributed Graph

- 1. Graph Convolutional Encoder
- 2. Graph Decoder



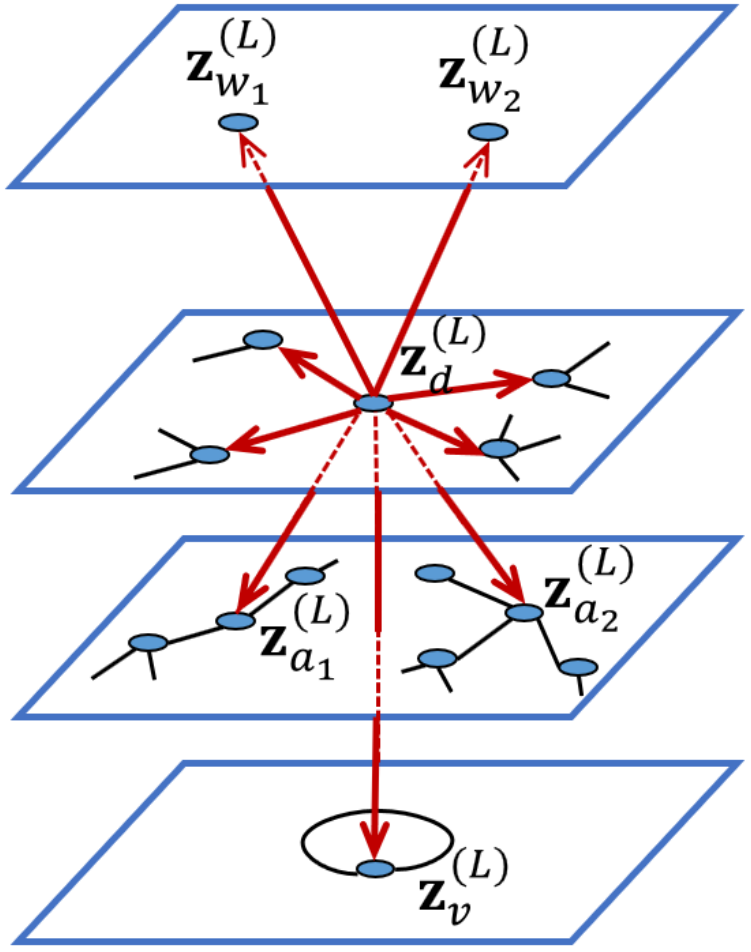
TMs for Heterogeneous Text-Attributed Graph

- Negative sampling decoder.

$$\sum_{w:d_w=1} [\log \phi(\mathbf{z}_d^\top \mathbf{z}_w) + \sum_{m=1}^{M} \mathbb{E}_{w' \sim p_n(w)} \log \phi(-\mathbf{z}_d^\top \mathbf{z}_{w'})]$$

inner product

negative sampling



TMs for Heterogeneous Text-Attributed Graph

- **Experiments**
- 1. Datasets

Name	#Documents	#Authors	#Venues	#Doc-Doc Edges	Vocabulary	#Labels
ML	2,947	2,814	N.A.	8,146	5,814	7
PL	2,449	2,778	N.A.	7,274	5,066	9
COVID	1,500	880	169	5,706	5,083	5
HEP-TH	20,151	10,432	343	234,193	5,001	N.A.
Aminer	114,741	143,534	50	265,345	10,018	10
Web	445,657	36,405	N.A.	565,502	10,015	N.A.

TMs for Heterogeneous Text-Attributed Graph

- **Experiments**
- 2. Document classification

Models w/o authors or venues

Models with authors or venues

Model	ML	PL	COVID	Aminer
ProdLDA	69.3±0.7	53.1±2.5	73.9±1.6	64.0±0.2
WLDA	31.8±3.5	33.2±1.8	65.6±2.5	65.5±0.2
NSTM	45.2±2.6	41.3±3.2	69.0±2.1	44.6±0.3
DVAE	70.8±1.3	58.7±1.5	77.8±2.1	67.4±0.3
ATM	52.0±1.3	43.8±3.0	72.4±1.7	64.1±0.8
RTM	61.6±2.4	53.3±1.4	70.5±3.2	58.8±0.5
Adjacent-Encoder	80.5±0.6	72.2±0.9	83.7±1.0	49.6±0.3
LANTM	73.5±1.6	61.8±0.9	78.2±1.6	N.A.
VGAE	67.7±1.9	55.0±2.3	56.4±4.6	63.6±0.6
VGATM	81.5±0.4	73.7±0.5	83.2±1.1	98.0±0.1

TMs for Heterogeneous Text-Attributed Graph

- **Experiments**
- 3. Topic coherence

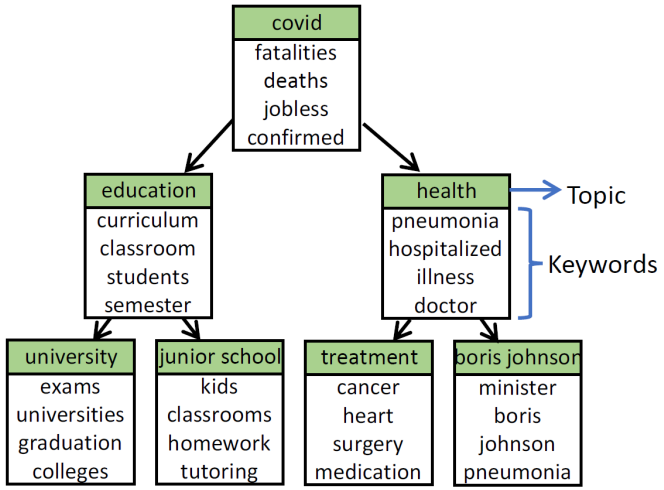
Model	Topic Coherence NPMI					
	ML	PL	COVID	HEP-TH	Aminer	Web
ProdLDA	10.0±0.7	9.4±0.5	12.0±0.7	10.3±0.6	9.3±0.5	21.2±0.2
WLDA	9.7±0.2	11.6±0.1	12.5±0.5	13.7±0.4	17.9±0.5	**23.9±0.8**
DVAE	14.7±0.0	15.2±0.1	15.8±0.1	14.8±0.1	15.5±0.1	17.6±0.2
ATM	10.2±0.4	12.0±0.5	9.8±0.2	10.2±0.3	15.0±0.2	23.2±0.7
RTM	7.3±0.2	8.9±0.5	16.2±0.5	6.6±0.3	10.8±0.3	20.9±0.4
Adjacent-Encoder	12.4±0.9	12.5±0.7	13.8±0.4	13.4±0.4	11.4±0.2	15.2±0.1
LANTM	9.9±1.2	9.8±0.7	8.6±0.3	10.4±1.5	N.A.	N.A.
TVGAE	3.3±0.5	3.8±0.5	5.2±0.5	N.A.	2.6±0.3	N.A.
VGATM	**13.6±1.1**	**20.5±1.0**	**19.4±1.8**	**19.0±0.0**	**21.7±1.1**	**23.7±1.7**

Section 3: TM-based Text-Attributed Graph Models

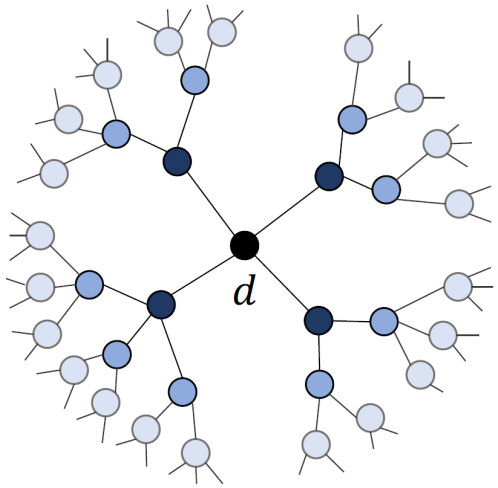
- Section 3.1: TMs for Static Text-Attributed Graph
- Section 3.2: TMs for Heterogeneous Text-Attributed Graph
- Section 3.3: Hierarchical Text-Attributed Graph

Hierarchical Text-Attributed Graph

- **Hierarchical topics $\mathcal{D}$**
  - Some articles report global COVID situation, while others focus on specific event.
  - → Text hierarchy $\mathcal{D}$
- **Hierarchical graph $\mathcal{E}$**
  - A breaking news article is traced by following articles reporting subsequent events.
  - → Graph hierarchy $\mathcal{E}$



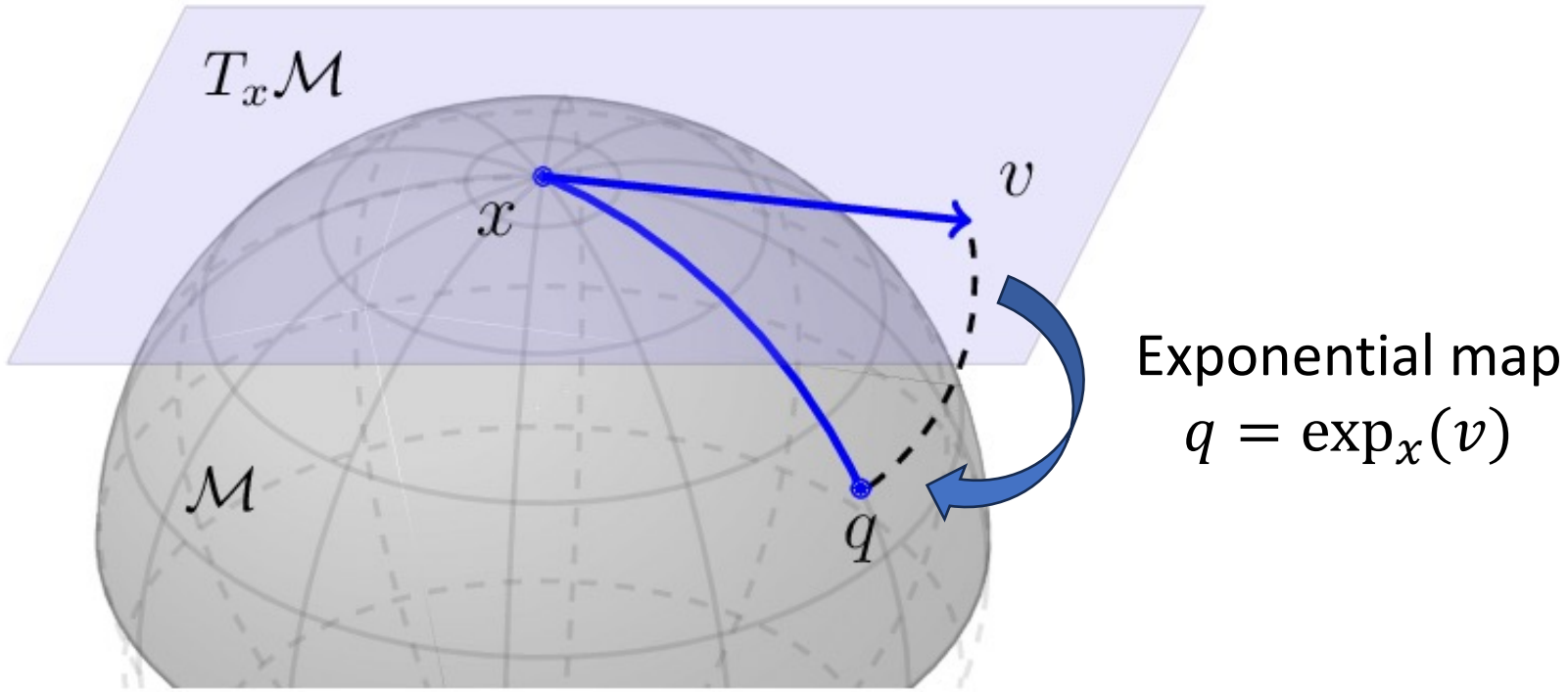
(a) Text hierarchy



(b) Graph hierarchy

Hierarchical Text-Attributed Graph

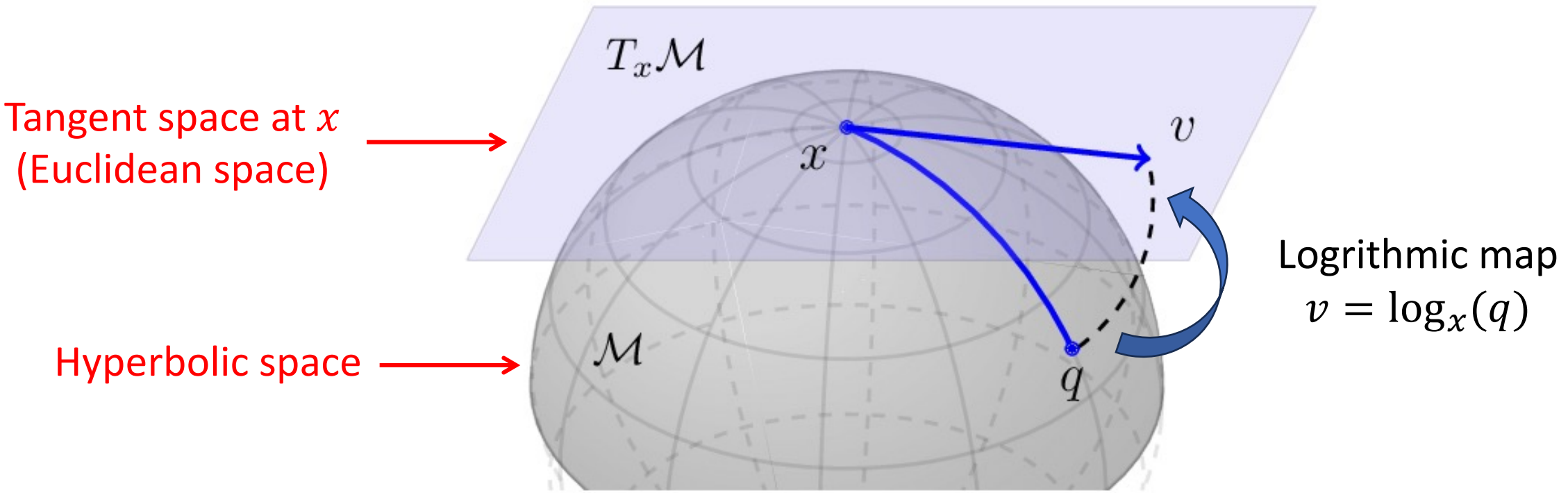
- We present HGTM [1] in this section, a hyperbolic GNN-based topic model.



[1] Zhang, D. C., Ying, R., & Lauw, H. W. (2023, August). Hyperbolic graph topic modeling network with continuously updated topic tree. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (pp. 3206-3216).

Hierarchical Text-Attributed Graph

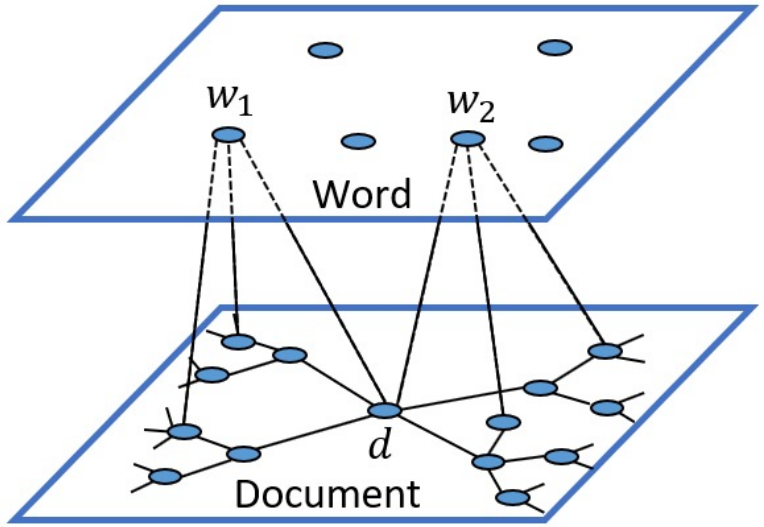
- We present HGTM [1] in this section, a hyperbolic GNN-based topic model.



[1] Zhang, D. C., Ying, R., & Lauw, H. W. (2023, August). Hyperbolic graph topic modeling network with continuously updated topic tree. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (pp. 3206-3216).

Hierarchical Text-Attributed Graph

- **Hyperbolic Graph Encoder (for graph hierarchy $\mathcal{E}$)**
- 1. Given a Text-Attributed Graph $\mathcal{G} = \{\mathcal{D}, \mathcal{E}\}$, we construct a two-layer graph for documents and words.



Hierarchical Text-Attributed Graph

- **Hyperbolic Graph Encoder (for graph hierarchy $\mathcal{E}$)**
- 2. **Intra-layer** encoding
  - A. Hyperbolic linear transformation

$$\tilde{\mathbf{z}}_d'^{(l)} = \exp_{\mathbf{0}}^{c}(\mathbf{W}^{(l)} \log_{\mathbf{0}}^{c}(\mathbf{z}_d^{(l-1)}))$$

  - B. Hyperbolic neighbor attention

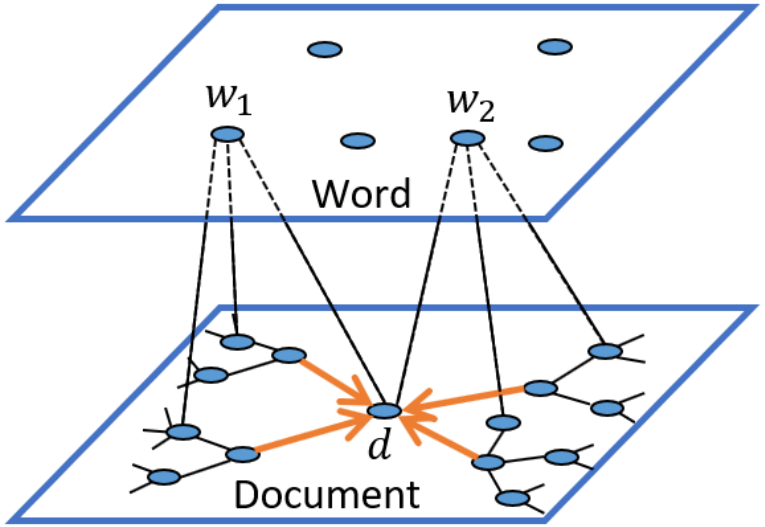
$$a_{ij} = \text{softmax}\Big(\sigma\big(\boldsymbol{\beta}^{(l)\top}[\log_{\mathbf{0}}^{c}(\tilde{\mathbf{z}}_{d_i}^{(l)}) \,||\, \log_{\mathbf{0}}^{c}(\tilde{\mathbf{z}}_{d_j}^{(l)})]\big)\Big) \text{ where } d_j \in \mathcal{N}(i)$$

  - C. Hyperbolic aggregation

$$\mathbf{z}_{d_i}^{(l)} = f_{\text{act}}^{c,c'}\left(\exp_{\mathbf{0}}^{c}\left(\frac{1}{2}\left(\log_{\mathbf{0}}^{c}(\tilde{\mathbf{z}}_{d_i}^{(l)}) + \sum_{d_j \in \mathcal{N}(i)} a_{ij} \log_{\mathbf{0}}^{c}(\tilde{\mathbf{z}}_{d_j}^{(l)})\right)\right)\right)$$

  - Summarizing above three steps, we have

$$\boldsymbol{z_{intra}} = HGNN\big(d, N_{intra}(d)\big)$$



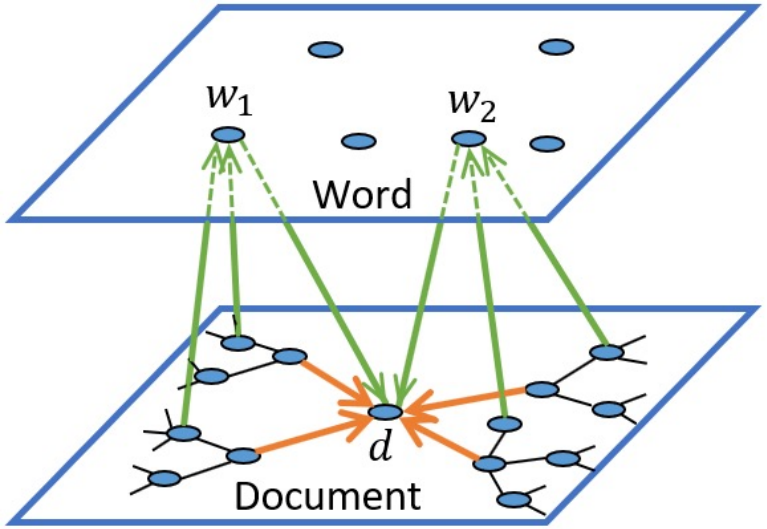
Hierarchical Text-Attributed Graph

- **Hyperbolic Graph Encoder (for graph hierarchy $\mathcal{E}$)**
- 3. **Cross-layer** encoding

$$z_{cross} = HGNN\left(d, N_{cross}(d)\right)$$

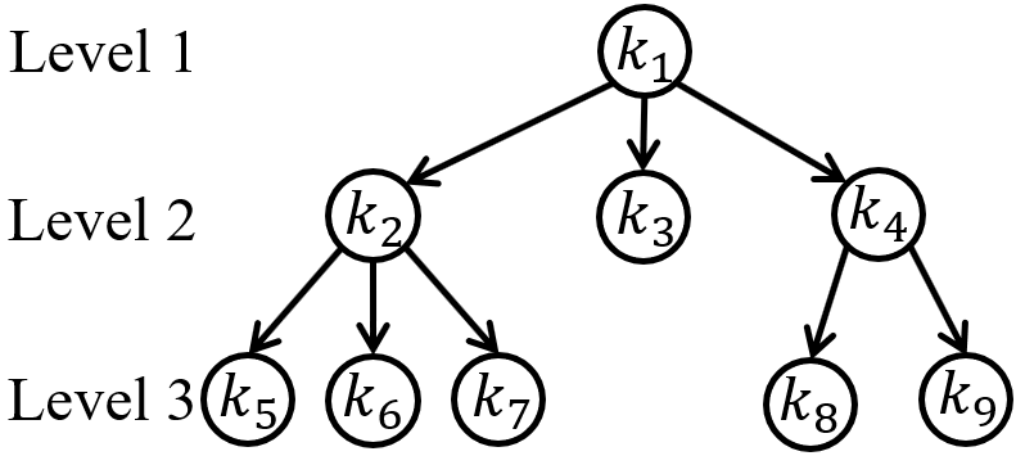
- 4. Hyperbolic mean pooling

$$z_d = \exp\left(\frac{1}{2} \times (\log(z_{intra}) + \log(z_{cross}))\right)$$



Hierarchical Text-Attributed Graph

- **Tree-Structured Decoder (for text hierarchy $\mathcal{D}$)**
- 1. Initialize a latent topic tree

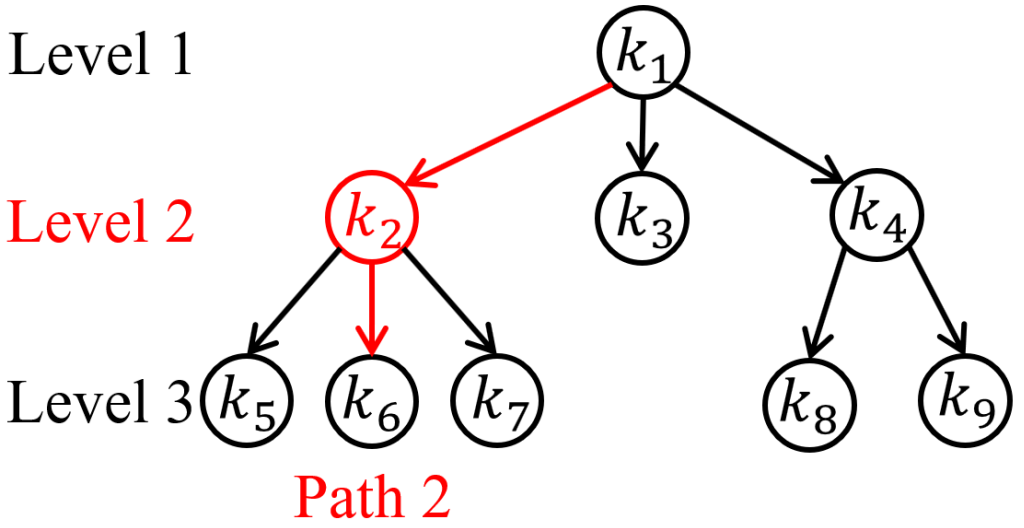


Hierarchical Text-Attributed Graph

- **Tree-Structured Decoder (for text hierarchy $\mathcal{D}$)**
- 1. Initialize a latent topic tree
- 2. For doc d, evaluate path distribution

$$p(k_1 \to k_2 \to k_6) = p(k_6|k_2)p(k_2|k_1)p(k_1)$$

$$p(k_2|k_1) = \frac{\left(1 + \text{dist}\left(z_d, t_{k_2}\right)\right)^{-1}}{\sum_{k'=k_2,k_3,k_4}\left(1 + \text{dist}(z_d, t_{k'})\right)^{-1}}$$



- 3. Evaluate level distribution

$$p(\text{level } s) = \frac{(1 + h(s)^2)^{-1}}{\sum_{s'=1,2,3}(1 + h(s')^2)^{-1}} \quad \text{where} \quad h(s)^2 = \min\left\{\text{dist}\left(z_d, t_{k_s}\right)^2 \middle| k_s \in \text{level } s\right\}$$

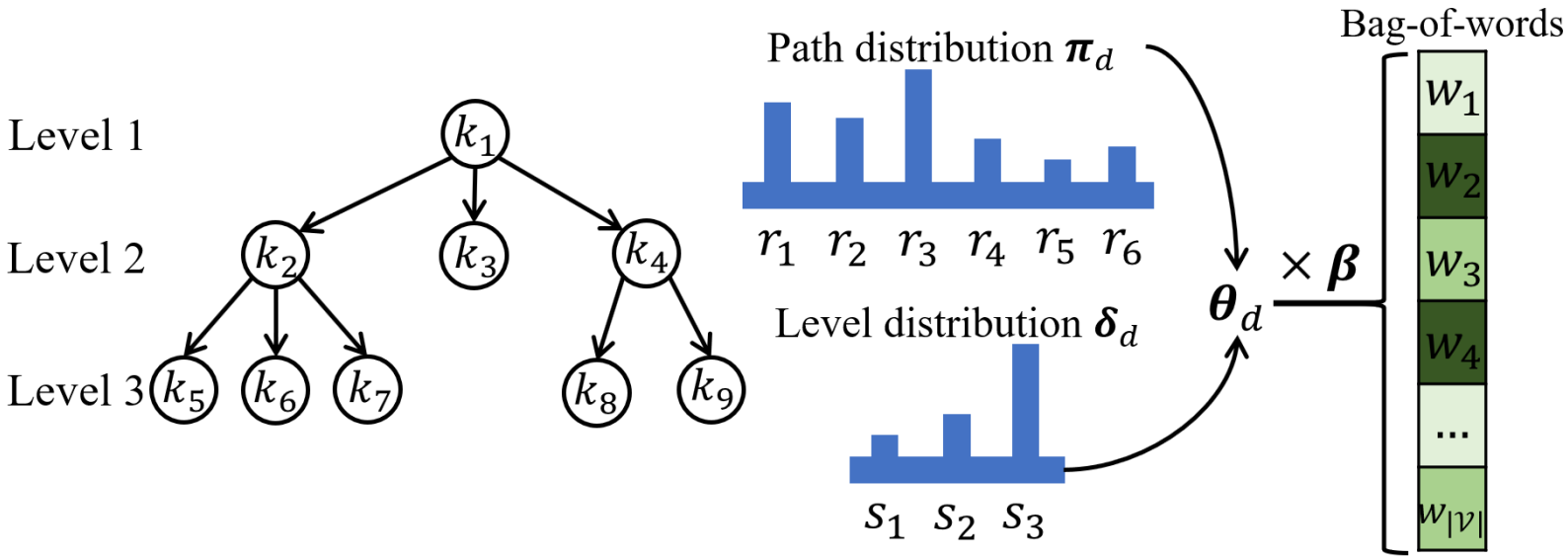
Hierarchical Text-Attributed Graph

- **Tree-Structured Decoder (for text hierarchy $\mathcal{D}$)**
- 4. Topic distribution

$$p(k_2) = p(s = 2) \times \big(p(\text{path } 1) + p(\text{path } 2) + p(\text{path } 3)\big)$$

$$\theta_d = [p(k_1), p(k_2), \dots, p(k_9)]$$

- 5. Decoding with cross-entropy loss $\hat{d} = \text{softmax}(\beta\theta_d)$



Hierarchical Text-Attributed Graph

- **Experiments**
- 1. Datasets

Name	#Documents	#Links	Vocabulary	#Labels
ML	3,087	8,573	2,885	7
PL	2,597	7,754	3,106	9
COVID	1,500	5,706	5,083	5
Aminer	114,741	265,345	10,018	10
Web	445,657	565,502	10,015	N.A.

Hierarchical Text-Attributed Graph

- **Experiments**
- 2. Document classification

Models with graph hierarchy but without text hierarchy

Models with text hierarchy but without graph hierarchy

Model	Micro F1 score			
	ML	PL	COVID	Aminer
HGCN	82.6±1.3	70.3±1.0	86.0±0.5	67.6±1.0
LGCN	73.0±2.2	62.8±2.6	60.5±5.0	N.A.
TextGCN	78.3±0.7	67.5±0.7	83.7±0.5	N.A.
HyperGAT	80.0±0.4	65.8±2.5	84.3±1.2	**74.2±1.5**
HINT	69.5±1.1	55.4±2.3	85.7±1.5	66.5±0.5
HGTM	**83.8±0.5**	**72.2±1.4**	**86.3±1.7**	70.0±0.3

Hierarchical Text-Attributed Graph

- **Experiments**
- 3. Topic coherence

Model	Topic Coherence NPMI (higher is better)				
	ML	PL	COVID	Aminer	Web
ProdLDA	10.9±0.7	12.1±0.7	12.0±0.7	9.3±0.5	21.2±0.2
ETM	7.1±0.2	8.7±0.1	8.2±0.7	5.4±0.3	16.4±0.6
NSTM	17.2±0.7	19.2±0.7	22.0±0.6	15.5±0.3	24.0±0.3
GATON	17.4±1.0	5.4±1.1	13.8±1.2	19.4±1.5	4.8±1.1
GraphBTM	5.1±0.5	7.0±0.4	10.4±0.1	8.2±0.3	N.A.
GNTM	12.1±0.3	15.4±0.7	13.8±0.8	17.3±0.4	23.8±0.3
nCRP	2.2±0.1	2.2±0.1	3.0±0.1	0.2±0.1	2.8±0.0
TSNTM	12.1±0.6	15.1±0.8	14.1±0.8	17.6±0.8	**26.6±2.3**
HTV	10.8±1.0	13.3±1.8	16.6±2.5	17.2±0.6	26.5±0.9
RTM	7.1±0.3	9.3±0.2	16.2±0.5	10.8±0.3	20.9±0.4
Adjacent-Encoder	9.9±0.9	11.3±0.9	13.8±0.4	11.4±0.2	15.2±0.1
LANTM	5.4±0.3	7.2±0.8	8.6±0.3	N.A.	N.A.
GTNN	7.2±0.6	5.8±0.6	13.5±2.7	12.6±0.5	7.9±1.6
HINT	6.6±2.2	8.6±2.4	11.6±3.0	12.1±3.3	N.A.
HGTM	**19.0±2.6**	**21.9±2.8**	**23.3±3.1**	**20.5±1.4**	25.0±1.7

Modeling hierarchy is useful

Section 3: TM-based Text-Attributed Graph Models

- Section 3.1: TMs for Static Text-Attributed Graph
- Section 3.2: TMs for Heterogeneous Text-Attributed Graph
- Section 3.3: Hierarchical Text-Attributed Graph

Yale

Q & A

Yale University

SMU SINGAPORE MANAGEMENT UNIVERSITY

School of Computing and Information Systems

Text-Attributed Graph Representation Learning: Methods, Applications, and Challenges Section 4 (45 min)

Delvin Ce Zhang[1], Menglin Yang[1], Rex Ying[1], and Hady W. Lauw[2]

[1]Yale University, [2]Singapore Management University

Tutorial Outline

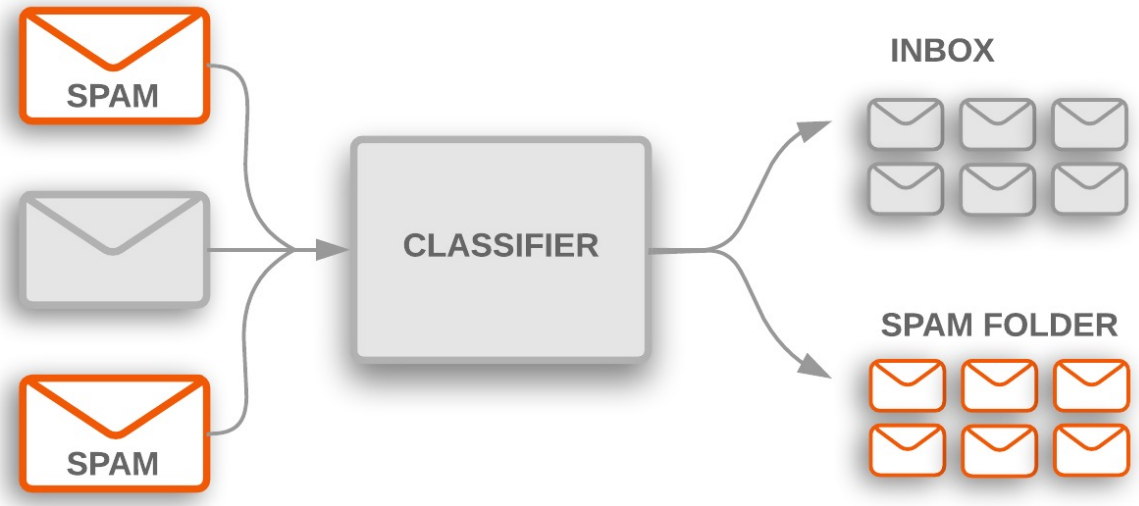
- Section 1: Text-Attributed Graph and Preliminaries (30 min)
- Section 2: PLM-based Text-Attributed Graph Models (45 min)
- Section 3: TM-based Text-Attributed Graph Models (45 min)
- Section 4: Applications, Challenges, and Future Directions (45 min)
- Summary and Q&A

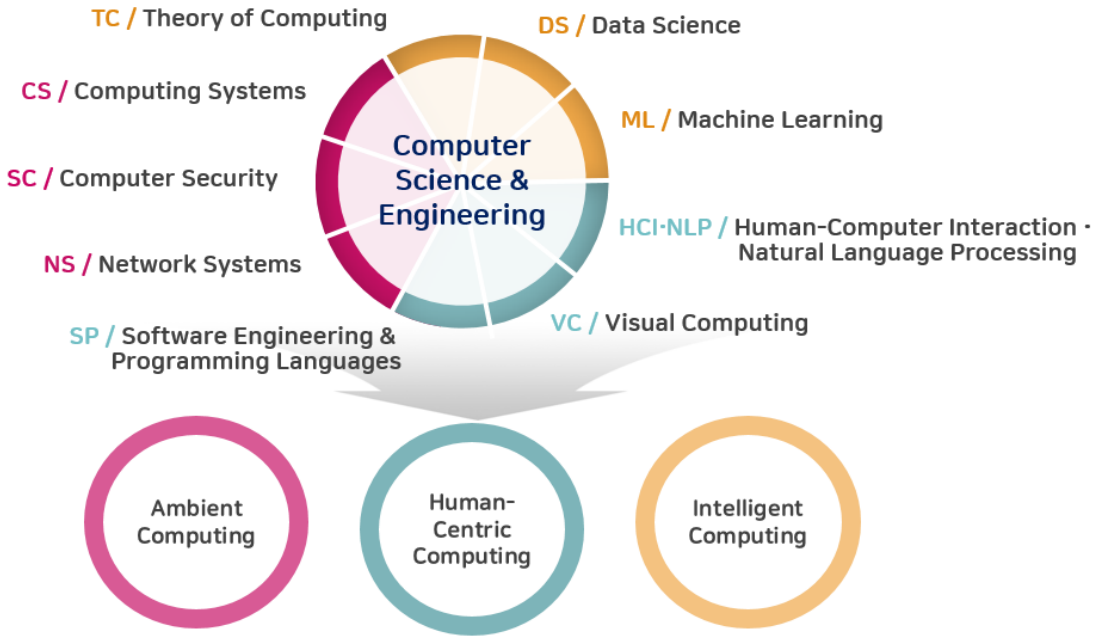
Section 4: Applications, Challenges, and Directions

- Section 4.1: Text Classification
- Section 4.2: Question Answering
- Section 4.3: Citation Recommendation
- Section 4.4: Challenges and Future Directions

Text Classification

- Text-Attributed Graph can be used for text classification where graph structure is auxiliary data to complement textual content.
- We present G2P2 [1] in this section.



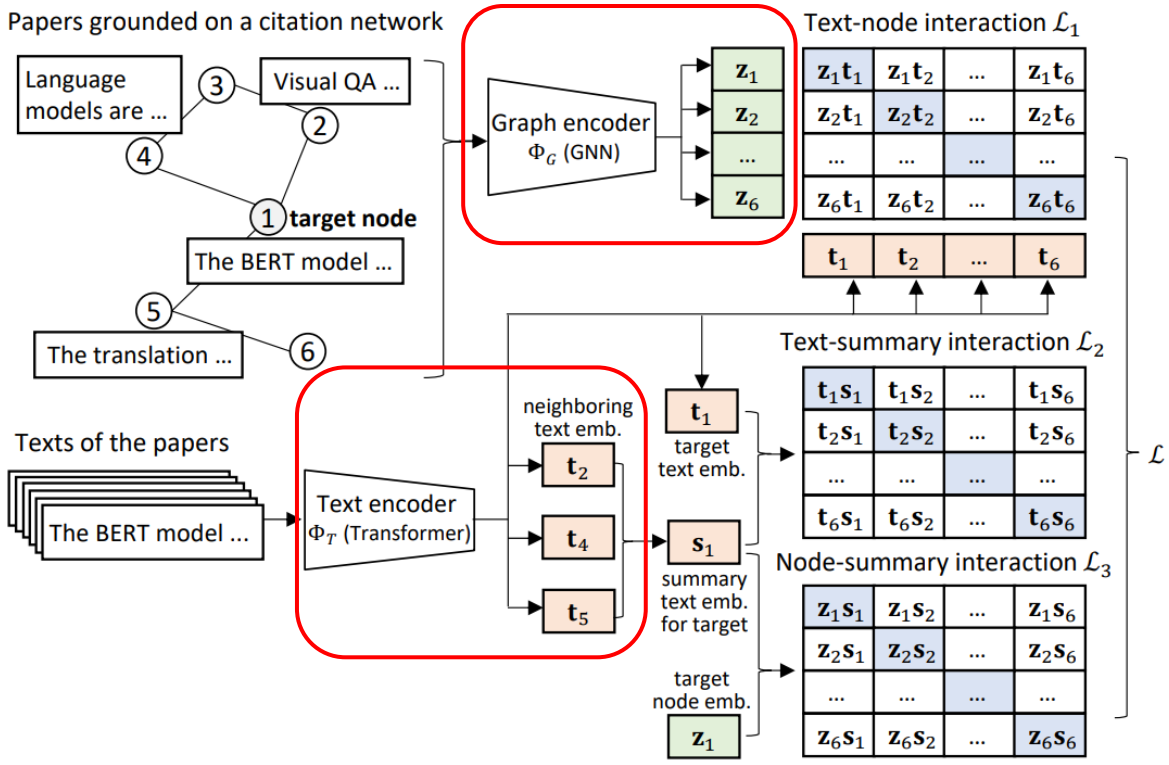


[1] Wen, Z., & Fang, Y. (2023, July). Augmenting low-resource text classification with graph-grounded pre-training and prompting. In Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 506-516).

Text Classification

- **Pre-training**
- 1. Text and graph encodings.

$$\boldsymbol{z}_i = \mathrm{GNN}(d_i, \mathcal{N}(i)) \qquad \boldsymbol{t}_i = \mathrm{Transformer}(d_i)$$



Text Classification

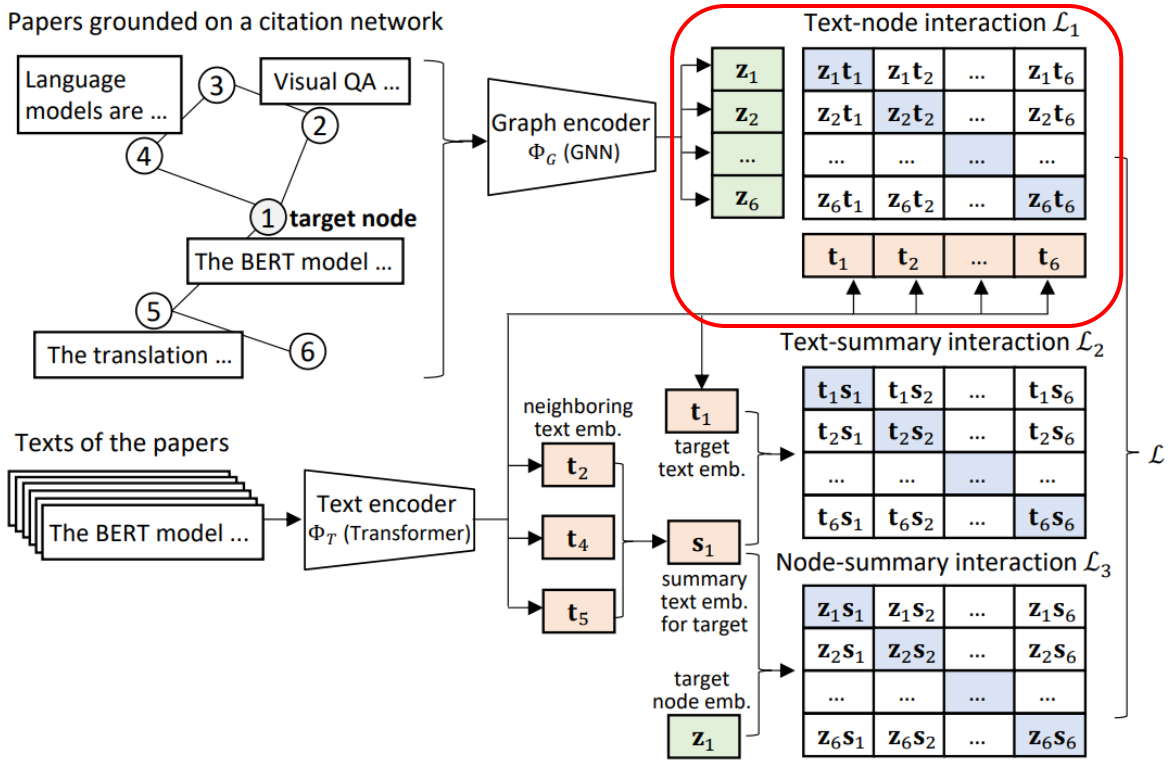
- **Pre-training**
- 1. Text and graph encodings.

$$\boldsymbol{z}_i = \mathrm{GNN}(d_i, \mathcal{N}(i)) \qquad \boldsymbol{t}_i = \mathrm{Transformer}(d_i)$$

- 2. Loss 1: text-node interaction.

$$\mathcal{L}_1 = \mathrm{CrossEntropy}(\exp(\tau) \times \cos(\boldsymbol{Z}, \boldsymbol{T}), \mathrm{Diag}(N))$$

Learnable scalar — Diagonal matrix



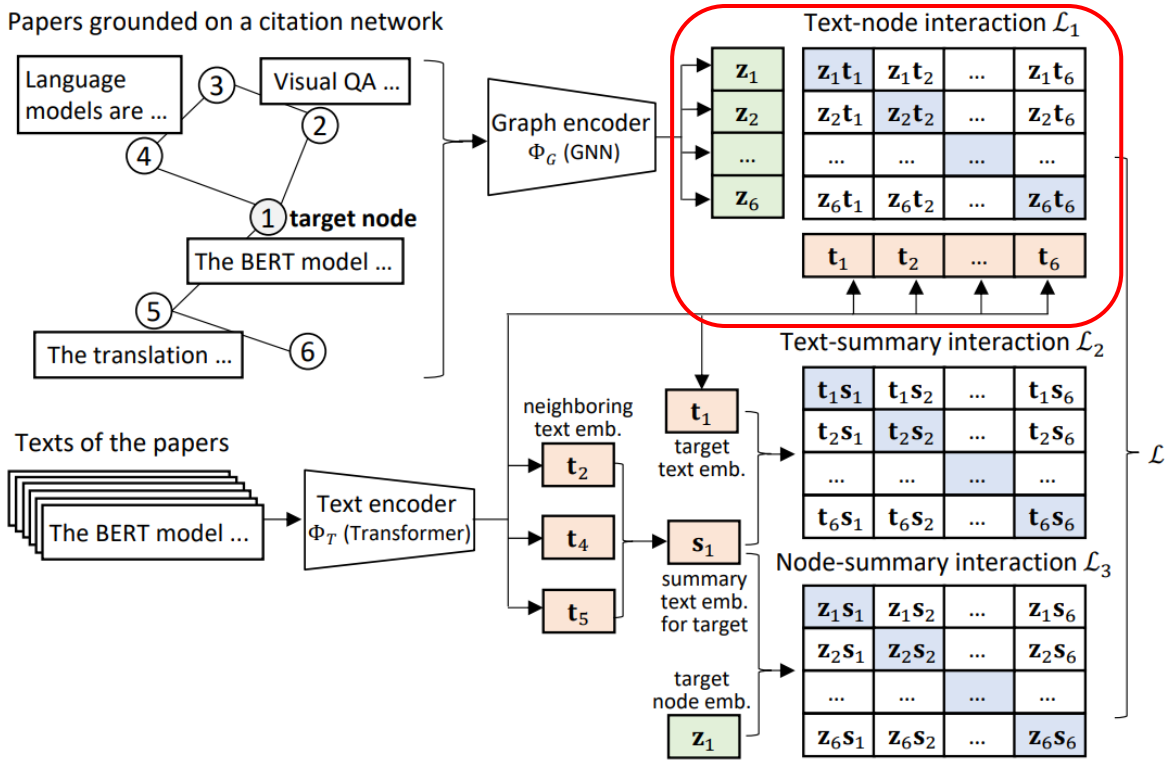
Text Classification

- **Pre-training**
- 1. Text and graph encodings.

$$\boldsymbol{z}_i = \mathrm{GNN}(d_i, \mathcal{N}(i)) \qquad \boldsymbol{t}_i = \mathrm{Transformer}(d_i)$$

- 2. Loss 1: text-node interaction.

$$\mathcal{L}_1 = \mathrm{CrossEntropy}(\exp(\tau) \times \cos(\boldsymbol{Z}, \boldsymbol{T}), \mathrm{Diag}(N)) + \mathrm{CrossEntropy}(\exp(\tau) \times \cos(\boldsymbol{Z}, \boldsymbol{T})^\top, \mathrm{Diag}(N))$$



Text Classification

- **Pre-training**
- 1. Text and graph encodings.

$$\boldsymbol{z}_i = \mathrm{GNN}(d_i, \mathcal{N}(i)) \qquad \boldsymbol{t}_i = \mathrm{Transformer}(d_i)$$

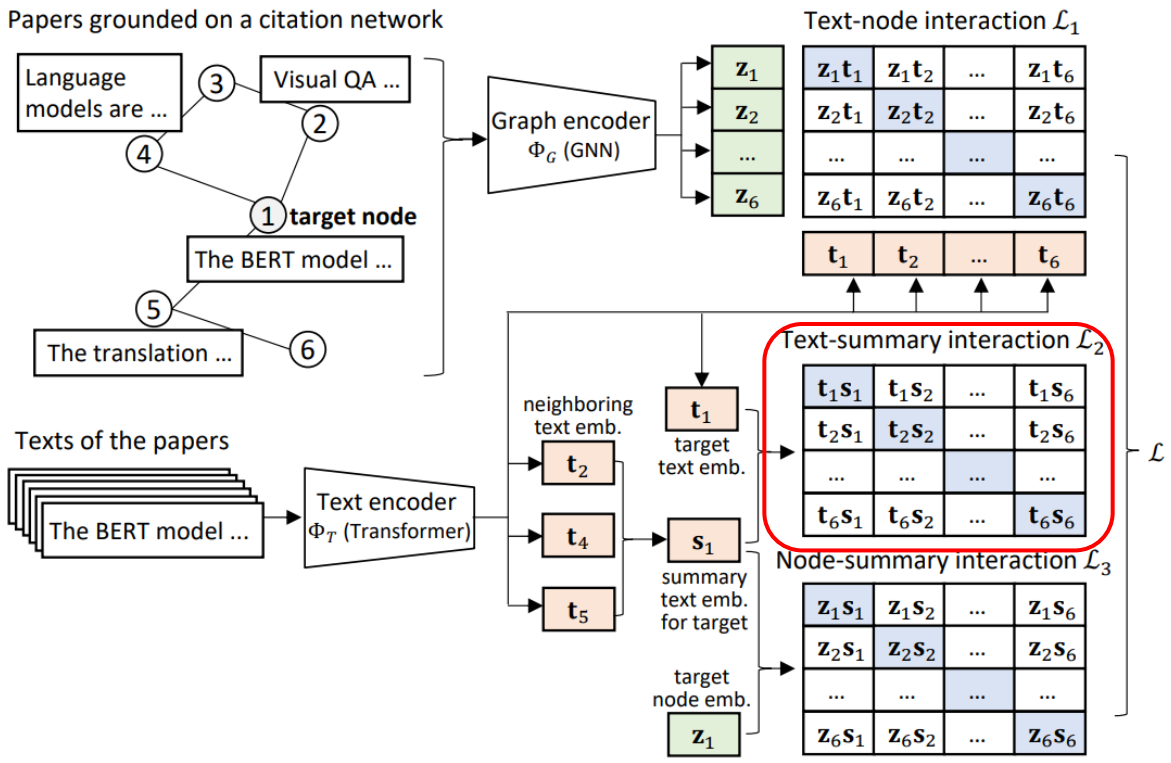
- 2. Loss 1: text-node interaction.

$$\mathcal{L}_1 = \mathrm{CrossEntropy}(\exp(\tau) \times \cos(\boldsymbol{Z}, \boldsymbol{T}), \mathrm{Diag}(N)) + \mathrm{CrossEntropy}(\exp(\tau) \times \cos(\boldsymbol{Z}, \boldsymbol{T})^\top, \mathrm{Diag}(N))$$

- 3. Loss 2: text-summary interaction.

$$\boldsymbol{s}_i = \mathrm{mean}(\boldsymbol{t}_i, \boldsymbol{t}_j | j \in \mathcal{N}(i))$$

$$\mathcal{L}_2 = \mathrm{CrossEntropy}(\exp(\tau) \times \cos(\boldsymbol{T}, \boldsymbol{S}), \mathrm{Diag}(N)) + \mathrm{CrossEntropy}(\exp(\tau) \times \cos(\boldsymbol{T}, \boldsymbol{S})^\top, \mathrm{Diag}(N))$$



Text Classification

- **Pre-training**
- 1. Text and graph encodings.

$$\boldsymbol{z}_i = \mathrm{GNN}(d_i, \mathcal{N}(i)) \qquad \boldsymbol{t}_i = \mathrm{Transformer}(d_i)$$

- 2. Loss 1: text-node interaction.

$$\mathcal{L}_1 = \mathrm{CrossEntropy}(\exp(\tau) \times \cos(\boldsymbol{Z}, \boldsymbol{T}), \mathrm{Diag}(N)) + \mathrm{CrossEntropy}(\exp(\tau) \times \cos(\boldsymbol{Z}, \boldsymbol{T})^\top, \mathrm{Diag}(N))$$

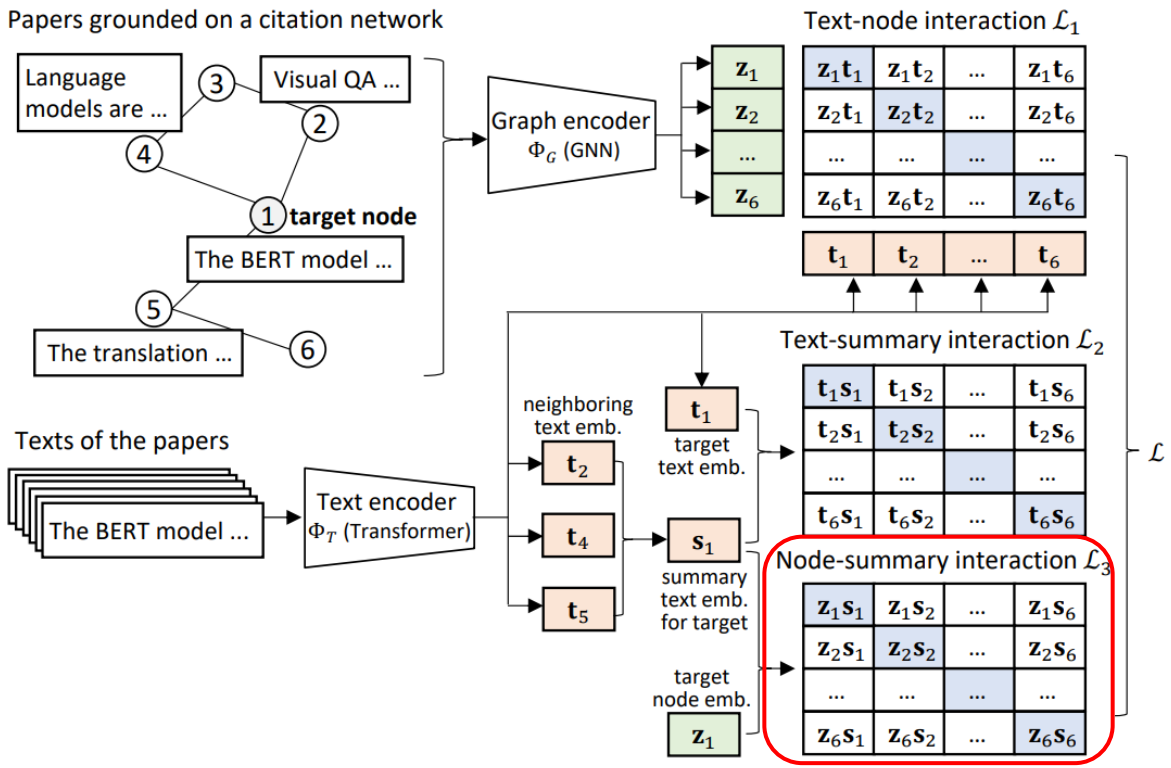
- 3. Loss 2: text-summary interaction.

$$\boldsymbol{s}_i = \mathrm{mean}(\boldsymbol{t}_i, \boldsymbol{t}_j | j \in \mathcal{N}(i))$$

$$\mathcal{L}_2 = \mathrm{CrossEntropy}(\exp(\tau) \times \cos(\boldsymbol{T}, \boldsymbol{S}), \mathrm{Diag}(N)) + \mathrm{CrossEntropy}(\exp(\tau) \times \cos(\boldsymbol{T}, \boldsymbol{S})^\top, \mathrm{Diag}(N))$$

- 4. Loss 3: Node-summary interaction.

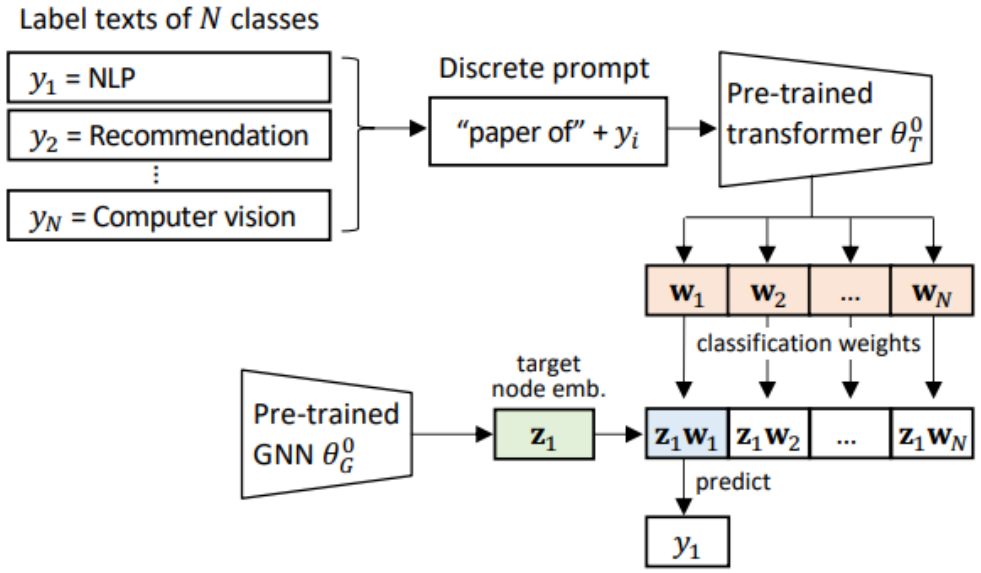
$$\mathcal{L}_3 = \mathrm{CrossEntropy}(\exp(\tau) \times \cos(\boldsymbol{Z}, \boldsymbol{S}), \mathrm{Diag}(N)) + \mathrm{CrossEntropy}(\exp(\tau) \times \cos(\boldsymbol{Z}, \boldsymbol{S})^\top, \mathrm{Diag}(N))$$



Text Classification

- **Zero-shot fine-tuning:**
- 1. Prompt.

$$p_y = \text{"paper of"} + [\text{CLASS}_y]$$



Text Classification

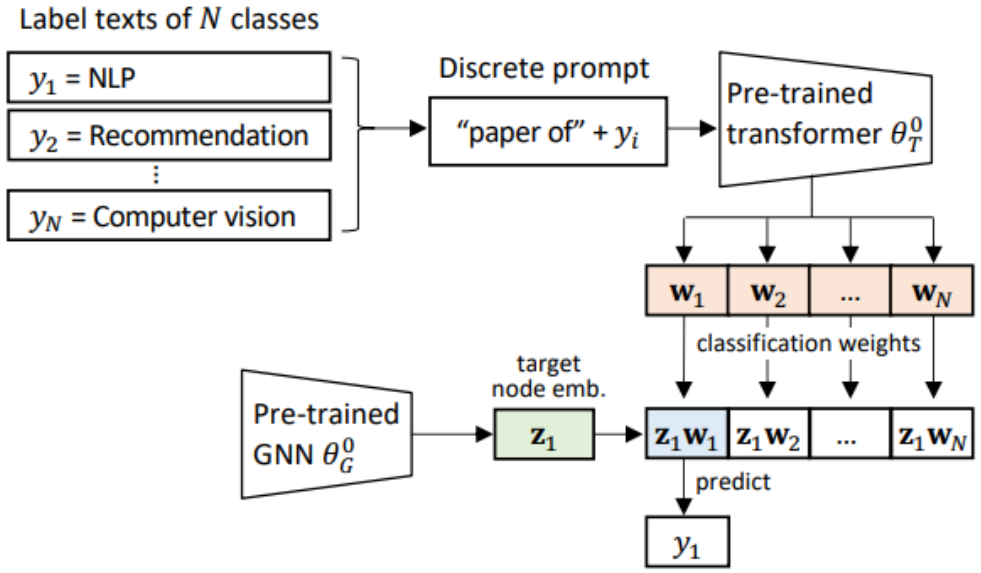
- **Zero-shot fine-tuning:**
- 1. Prompt.

$$p_y = \text{"paper of"} + [\text{CLASS}_y]$$

- 2. Obtain class embeddings.

$$\boldsymbol{w}_y = \text{Transformer}(p_c), \quad \text{where} \quad y = 1,2,\dots,N$$

Total number of classes



Text Classification

- **Zero-shot fine-tuning:**
- 1. Prompt.

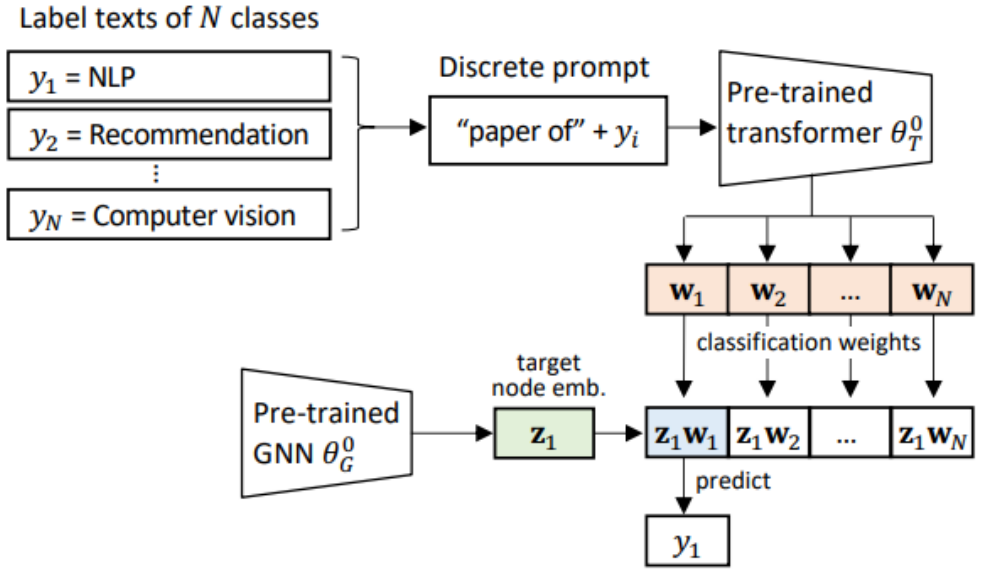
$$p_y = \text{"paper of"} + [\text{CLASS}_y]$$

- 2. Obtain class embeddings.

$$\boldsymbol{w}_y = \text{Transformer}(p_c), \quad \text{where} \quad y = 1,2,\dots,N$$

- 3. Evaluate class similarity.

$$p(y|\boldsymbol{z}_i) = \text{softmax}(\cos(\boldsymbol{z}_i, \boldsymbol{w}_y)), \quad \text{where} \quad y = 1,2,\dots,N$$

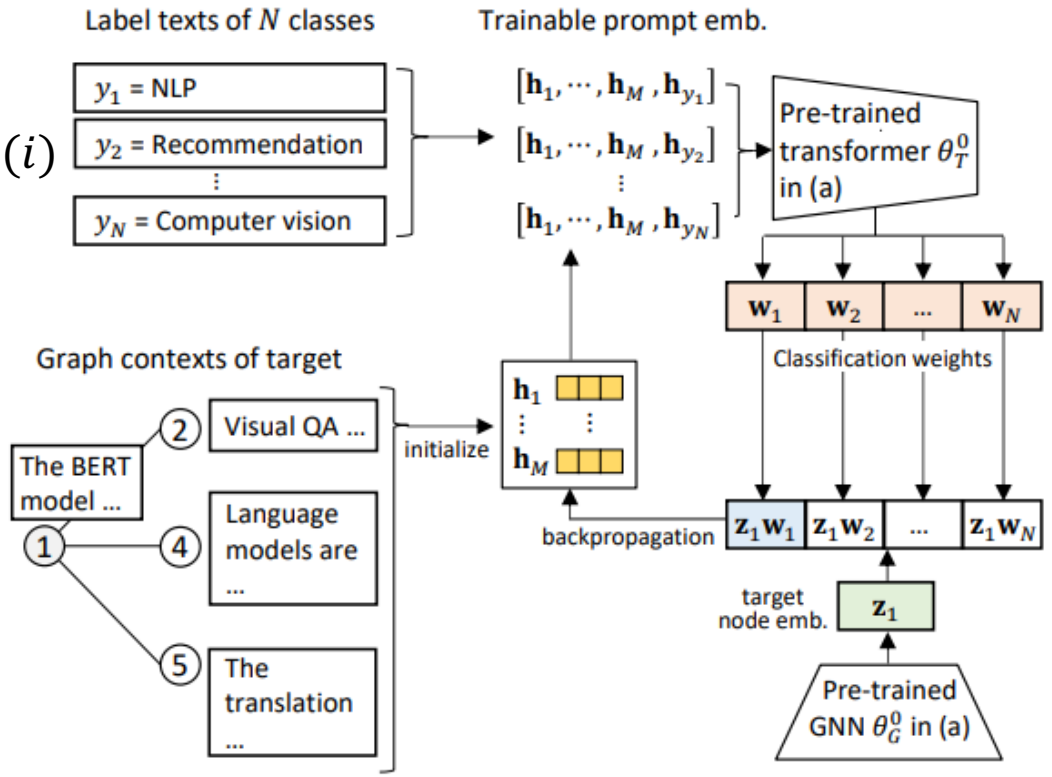


Text Classification

- **Few-shot fine-tuning:**
- 1. Prompt.

  $d_i = [\boldsymbol{h}_{i,1}, \boldsymbol{h}_{i,2}, \ldots, \boldsymbol{h}_{i,M}]$ $d_j = [\boldsymbol{h}_{j,1}, \boldsymbol{h}_{j,2}, \ldots, \boldsymbol{h}_{j,M}]$, where $j \in \mathcal{N}(i)$

- For a document d_i, obtain its first M words.

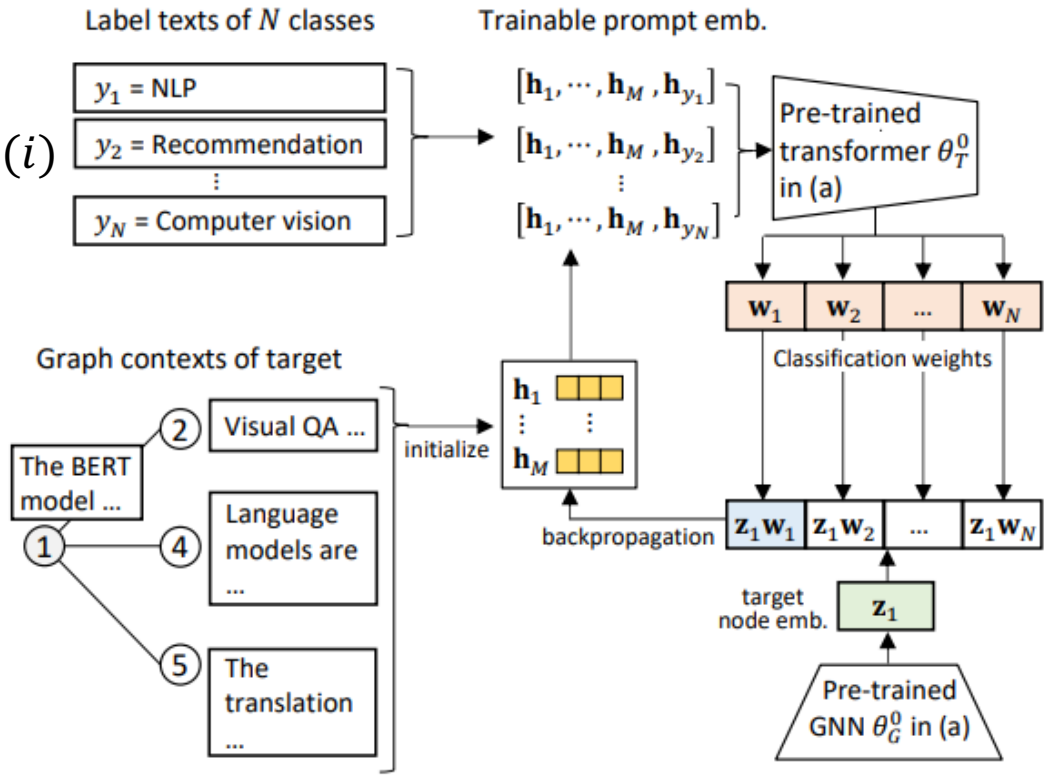


Text Classification

- **Few-shot fine-tuning:**
- 1. Prompt.

  $d_i = [\boldsymbol{h}_{i,1}, \boldsymbol{h}_{i,2}, \ldots, \boldsymbol{h}_{i,M}]$ $\quad d_j = [\boldsymbol{h}_{j,1}, \boldsymbol{h}_{j,2}, \ldots, \boldsymbol{h}_{j,M}]$, where $j \in \mathcal{N}(i)$

  $\boldsymbol{h}_m = \text{mean}(\boldsymbol{h}_{i,m}, \boldsymbol{h}_{j,m} | j \in \mathcal{N}(i))$
- For a document d_i, obtain its first M words.



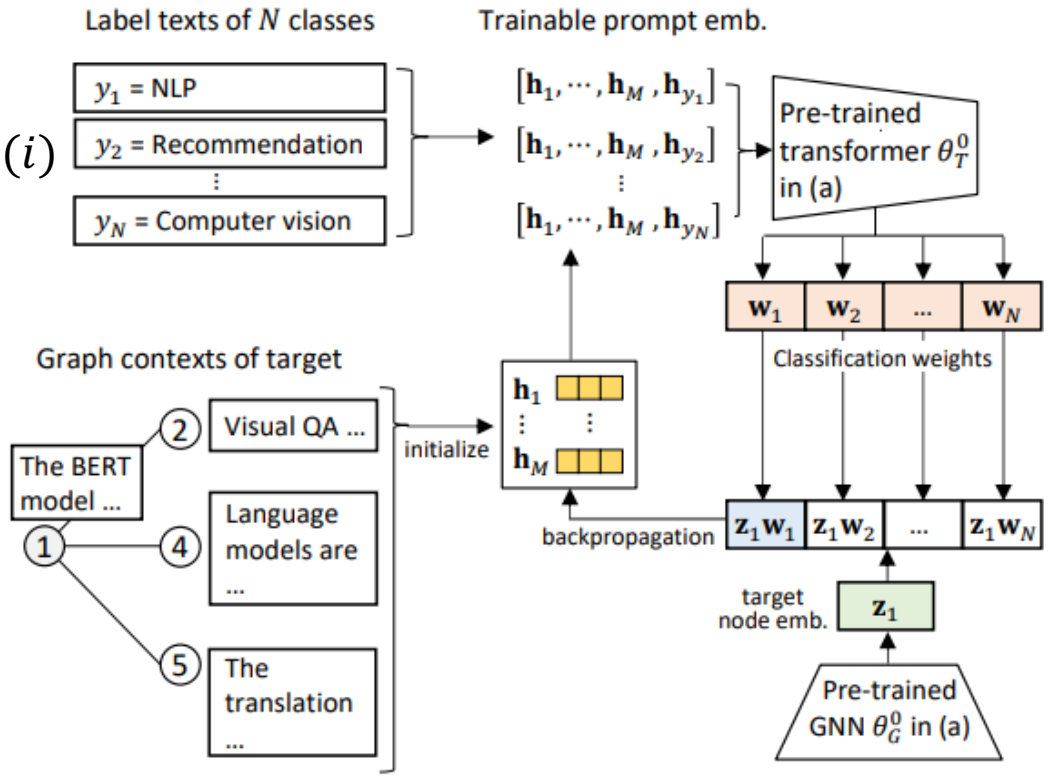
Text Classification

- **Few-shot fine-tuning:**
- 1. Prompt.

  $d_i = [\boldsymbol{h}_{i,1}, \boldsymbol{h}_{i,2}, \ldots, \boldsymbol{h}_{i,M}]$ $d_j = [\boldsymbol{h}_{j,1}, \boldsymbol{h}_{j,2}, \ldots, \boldsymbol{h}_{j,M}]$, where $j \in \mathcal{N}(i)$

  $\boldsymbol{h}_m = \text{mean}(\boldsymbol{h}_{i,m}, \boldsymbol{h}_{j,m} | j \in \mathcal{N}(i))$
- For a document d_i, obtain its first M words.
- 2. Obtain class embeddings.

$\boldsymbol{w}_y = \text{Transformer}([\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_M, \boldsymbol{h}_{CLASS_y}])$, where $y = 1,2,\ldots,Y$



Text Classification

- **Few-shot fine-tuning:**
- 1. Prompt.

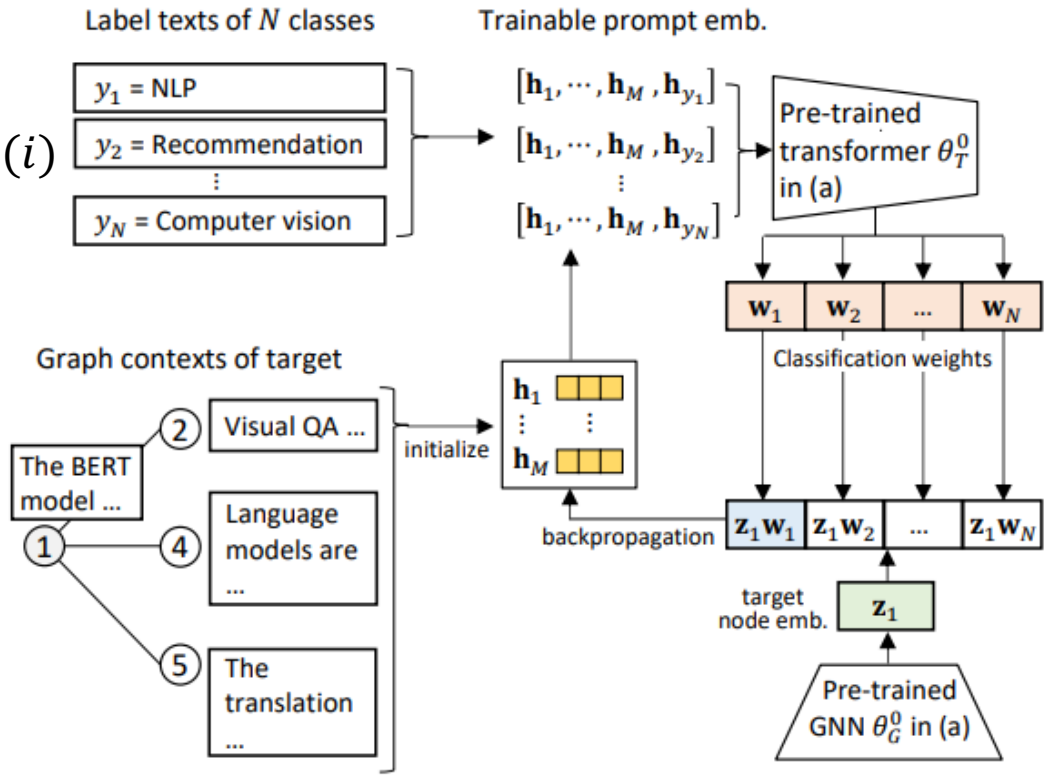
  $d_i = [\boldsymbol{h}_{i,1}, \boldsymbol{h}_{i,2}, \dots, \boldsymbol{h}_{i,M}]$ $d_j = [\boldsymbol{h}_{j,1}, \boldsymbol{h}_{j,2}, \dots, \boldsymbol{h}_{j,M}]$, where $j \in \mathcal{N}(i)$

  $\boldsymbol{h}_m = \text{mean}(\boldsymbol{h}_{i,m}, \boldsymbol{h}_{j,m} | j \in \mathcal{N}(i))$
- For a document d_i, obtain its first M words.
- 2. Obtain class embeddings.

$\boldsymbol{w}_y = \text{Transformer}([\boldsymbol{h}_1, \boldsymbol{h}_2, \dots, \boldsymbol{h}_M, \boldsymbol{h}_{CLASS_y}])$, where $y = 1,2, \dots, Y$

- 3. Evaluate class similarity.

$p(y|\boldsymbol{z}_i) = \text{softmax}(\cos(\boldsymbol{z}_i, \boldsymbol{w}_y))$, where $y = 1,2, \dots, Y$



Text Classification

- **Experiments**
- 1. Datasets

Dataset	Cora	Art	Industrial	M.I.
# Documents	25,120	1,615,902	1,260,053	905,453
# Links	182,280	4,898,218	3,101,670	2,692,734
# Avg. doc length	141.26	54.23	52.15	84.66
# Avg. node deg	7.26	3.03	2.46	2.97
# Total classes	70	3,347	2,462	1,191

Text Classification

- **Experiments**
- 2. Zero-shot text classification

Models w/o graph structure

	Cora	Art	Industrial	M.I.
RoBERTa	30.46±2.01	42.80±0.94	42.89±0.97	36.40±1.20
RoBERTa*	39.58±1.26	34.77±0.65	37.78±0.32	32.17±0.68
RoBERTa*+d	45.53±1.33	36.11±0.66	39.40±1.22	37.65±0.33
BERT	23.58±1.88	35.88±1.44	37.32±0.85	37.42±0.80
BERT*	23.38±1.96	54.27±1.85	56.02±1.22	50.19±0.72
BERT*+d	26.65±1.71	56.61±1.76	55.93±0.96	52.13±0.88
G2P2	63.52±2.89	76.52±0.59	76.66±0.31	74.60±0.62
G2P2+d	**65.28*** ±3.12	**76.99*** ±0.60	**77.43*** ±0.27	**75.86*** ±0.69
(improv.)	(+45.38%)	(+36.00%)	(+38.22%)	(+45.52%)

Text Classification

- **Experiments**
- 3. Five-shot text classification

	Cora		Art		Industrial		M.I.	
	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
GCN	41.15±2.41	34.50±2.23	22.47±1.78	15.45±1.14	21.08±0.45	15.23±0.29	22.54±0.82	16.26±0.72
$SAGE_{sup}$	41.42±2.90	35.14±2.14	22.60±0.56	16.01±0.28	20.74±0.91	15.31±0.37	22.14±0.80	16.69±0.62
TextGCN	59.78±1.88	55.85±1.50	43.47±1.02	32.20±1.30	53.60±0.70	45.97±0.49	46.26±0.91	38.75±0.78
GPT-GNN	76.72±2.02	72.23±1.17	65.15±1.37	52.79±0.83	62.13±0.65	54.47±0.67	67.97±2.49	59.89±2.51
DGI	78.42±1.39	74.58±1.24	65.41±0.86	53.57±0.75	52.29±0.66	45.26±0.51	68.06±0.73	60.64±0.61
$SAGE_{self}$	77.59±1.71	73.47±1.53	76.13±0.94	65.25±0.31	71.87±0.61	65.09±0.47	77.70±0.48	70.87±0.59
BERT	37.86±5.31	32.78±5.01	46.39±1.05	37.07± 0.68	54.00±0.20	47.57±0.50	50.14±0.68	42.96±1.02
BERT*	27.22±1.22	23.34±1.11	45.31±0.96	36.28±0.71	49.60±0.27	43.36±0.27	40.19±0.74	33.69±0.72
RoBERTa	62.10±2.77	57.21±2.51	72.95±1.75	62.25±1.33	76.35±0.65	70.49±0.59	70.67±0.87	63.50±1.11
RoBERTa*	67.42±4.35	62.72±3.02	74.47±1.00	63.35±1.09	77.08±1.02	71.44±0.87	74.61±1.08	67.78±0.95
P-Tuning v2	71.00±2.03	66.76±1.95	76.86±0.59	66.89±1.14	79.65±0.38	74.33±0.37	72.08±0.51	65.44±0.63
G2P2-p	79.16±1.23	74.99±1.35	79.59±0.31	68.26±0.43	80.86±0.40	74.44±0.29	81.26±0.36	74.82±0.45
G2P2	**80.08*** ±1.33	**75.91*** ±1.39	**81.03*** ±0.43	**69.86*** ±0.67	**82.46*** ±0.29	**76.36*** ±0.25	**82.77*** ±0.32	**76.48*** ±0.52
(improv.)	(+2.12%)	(+1.78%)	(+5.43%)	(+4.44%)	(+3.53%)	(+2.7%)	(+6.53%)	(+7.92%)

Section 4: Applications, Challenges, and Directions

- Section 4.1: Text Classification
- Section 4.2: Question Answering
- Section 4.3: Citation Recommendation
- Section 4.4: Challenges and Future Directions

Question Answering

- Text-Attributed Graph, such as hyperlinked Wikipedia page, provides high-order knowledge to answer the question.
- We present LinkBERT [1] in this section.

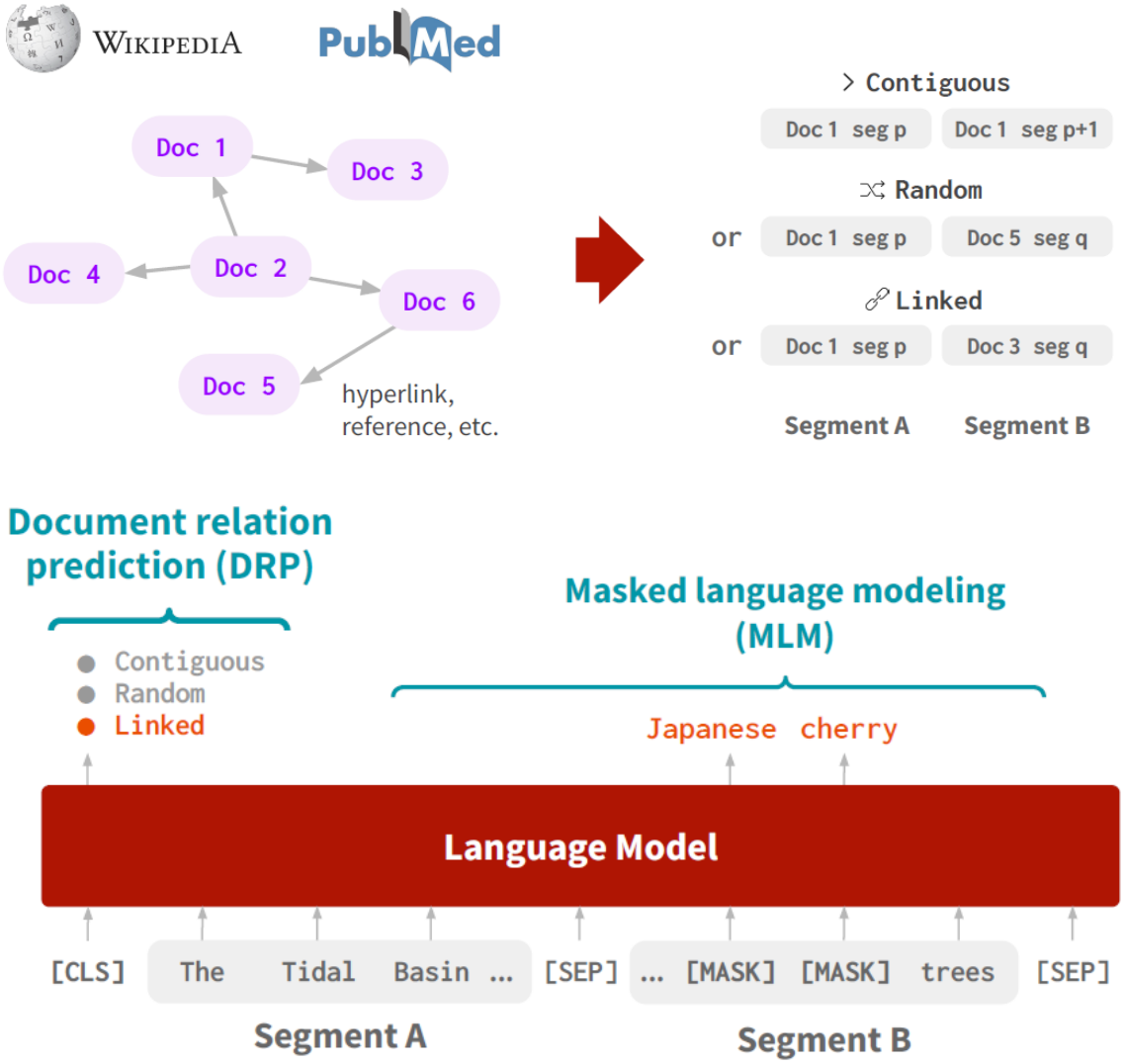
The linked document reveals that **the National Cherry Blossom Festival** celebrates **Japanese** cherry trees.

[1] Yasunaga, M., Leskovec, J., & Liang, P. (2022, May). LinkBERT: Pretraining Language Models with Document Links. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 8003-8016).

Question Answering

- 1. Linked documents.

$$\boldsymbol{H} = \mathrm{BERT}([\mathrm{CLS}]d_1[\mathrm{SEP}]d_3[\mathrm{SEP}]),\ \text{where}\ d_3 \in \mathcal{N}(d_1)$$



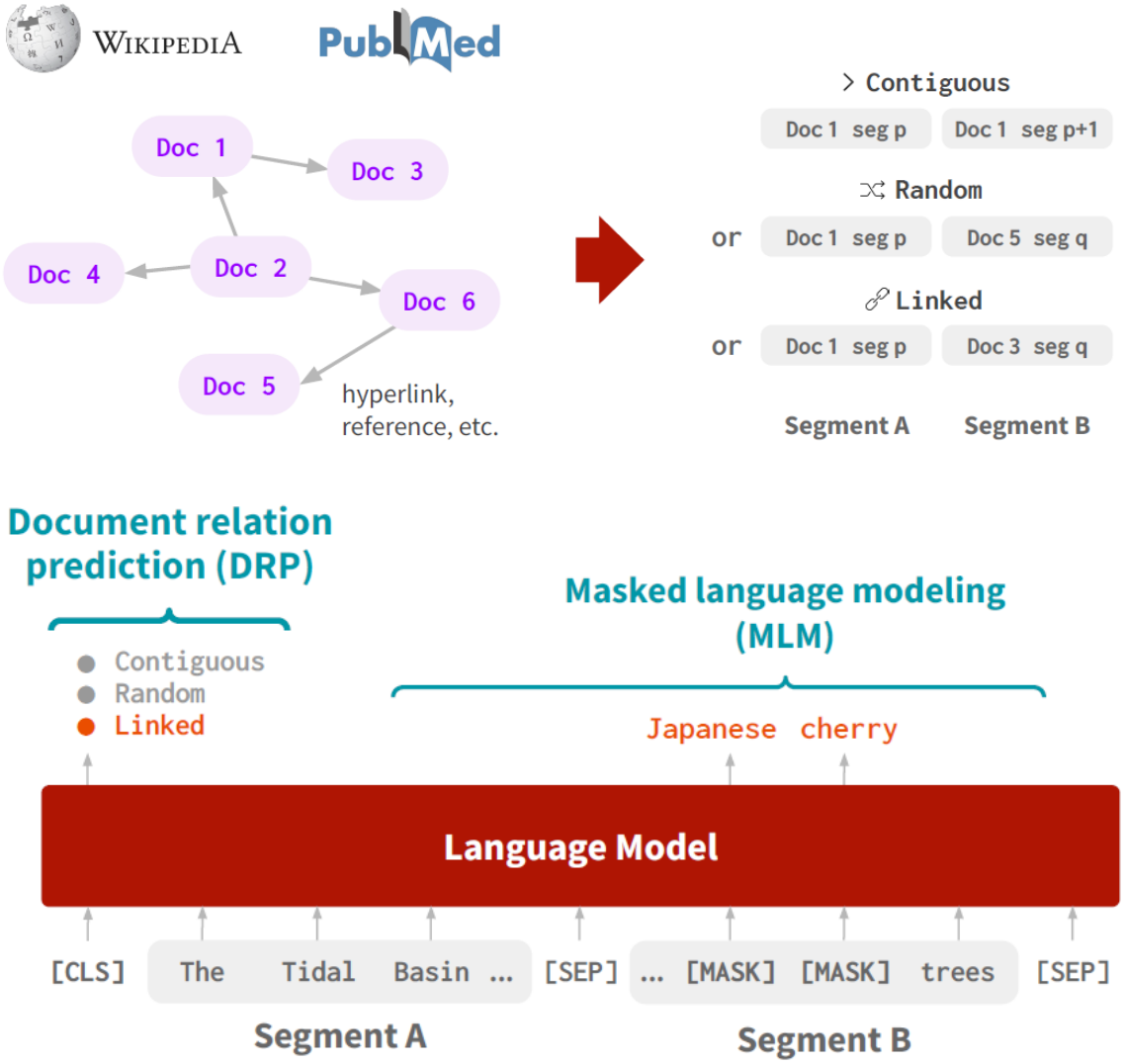
Question Answering

- 1. Linked documents.

$$\boldsymbol{H} = \mathrm{BERT}([\mathrm{CLS}]d_1[\mathrm{SEP}]d_3[\mathrm{SEP}]), \text{ where } d_3 \in \mathcal{N}(d_1)$$

- 2. Random documents.

$$\boldsymbol{H} = \mathrm{BERT}([\mathrm{CLS}]d_1[\mathrm{SEP}]d_5[\mathrm{SEP}]), \text{ where } d_5 \notin \mathcal{N}(d_1)$$



Question Answering

- 1. Linked documents.

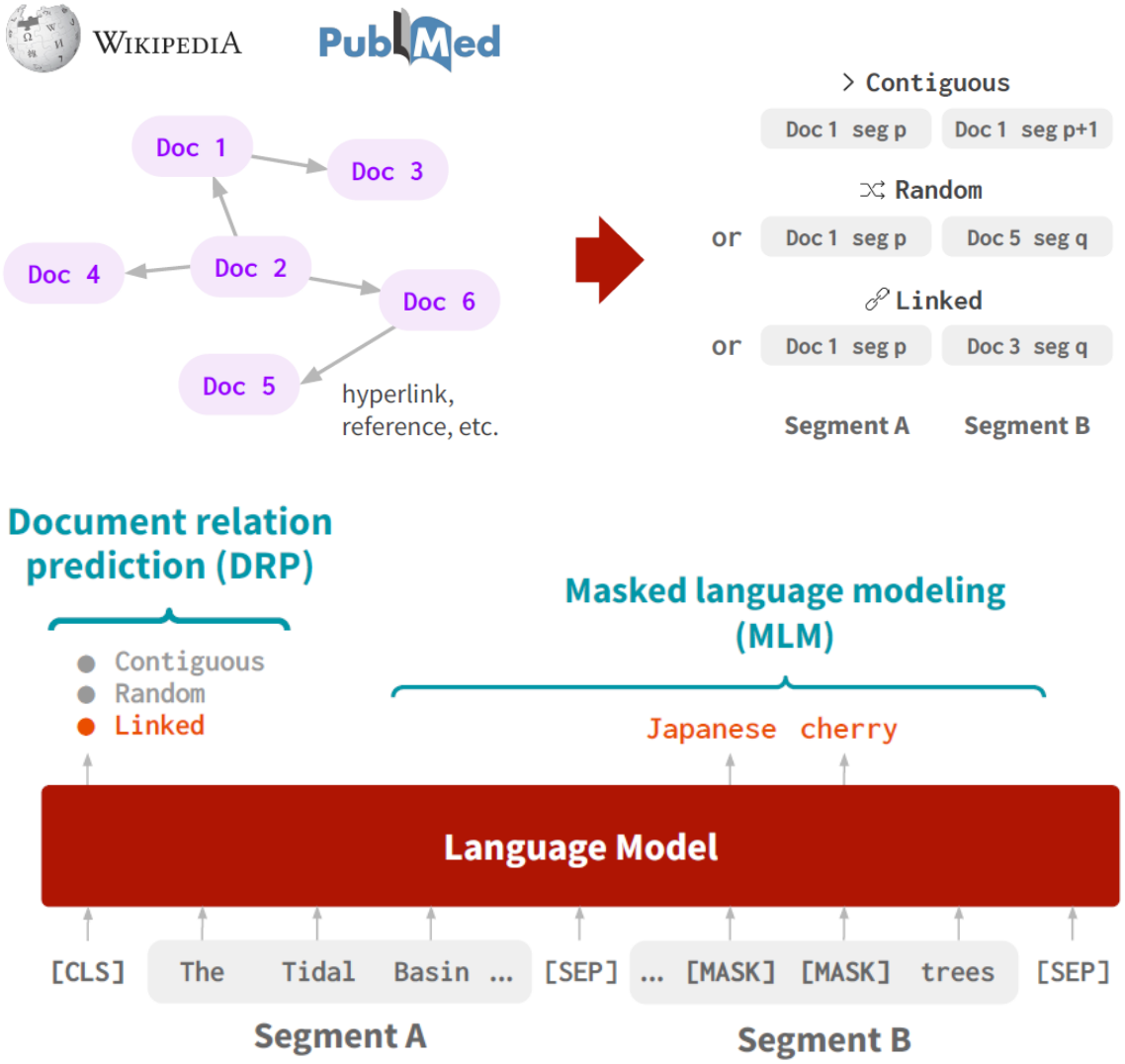
$$\boldsymbol{H} = \mathrm{BERT}([\mathrm{CLS}]d_1[\mathrm{SEP}]d_3[\mathrm{SEP}]), \text{ where } d_3 \in \mathcal{N}(d_1)$$

- 2. Random documents.

$$\boldsymbol{H} = \mathrm{BERT}([\mathrm{CLS}]d_1[\mathrm{SEP}]d_5[\mathrm{SEP}]), \text{ where } d_5 \notin \mathcal{N}(d_1)$$

- 3. Contiguous documents.

$$\boldsymbol{H} = \mathrm{BERT}([\mathrm{CLS}]d_1[\mathrm{SEP}]d_1'[\mathrm{SEP}]), \text{ where } d_1' \subset d_1$$



Question Answering

- 1. Linked documents.

$$\boldsymbol{H} = \mathrm{BERT}([\mathrm{CLS}]d_1[\mathrm{SEP}]d_3[\mathrm{SEP}]), \text{ where } d_3 \in \mathcal{N}(d_1)$$

- 2. Random documents.

$$\boldsymbol{H} = \mathrm{BERT}([\mathrm{CLS}]d_1[\mathrm{SEP}]d_5[\mathrm{SEP}]), \text{ where } d_5 \notin \mathcal{N}(d_1)$$

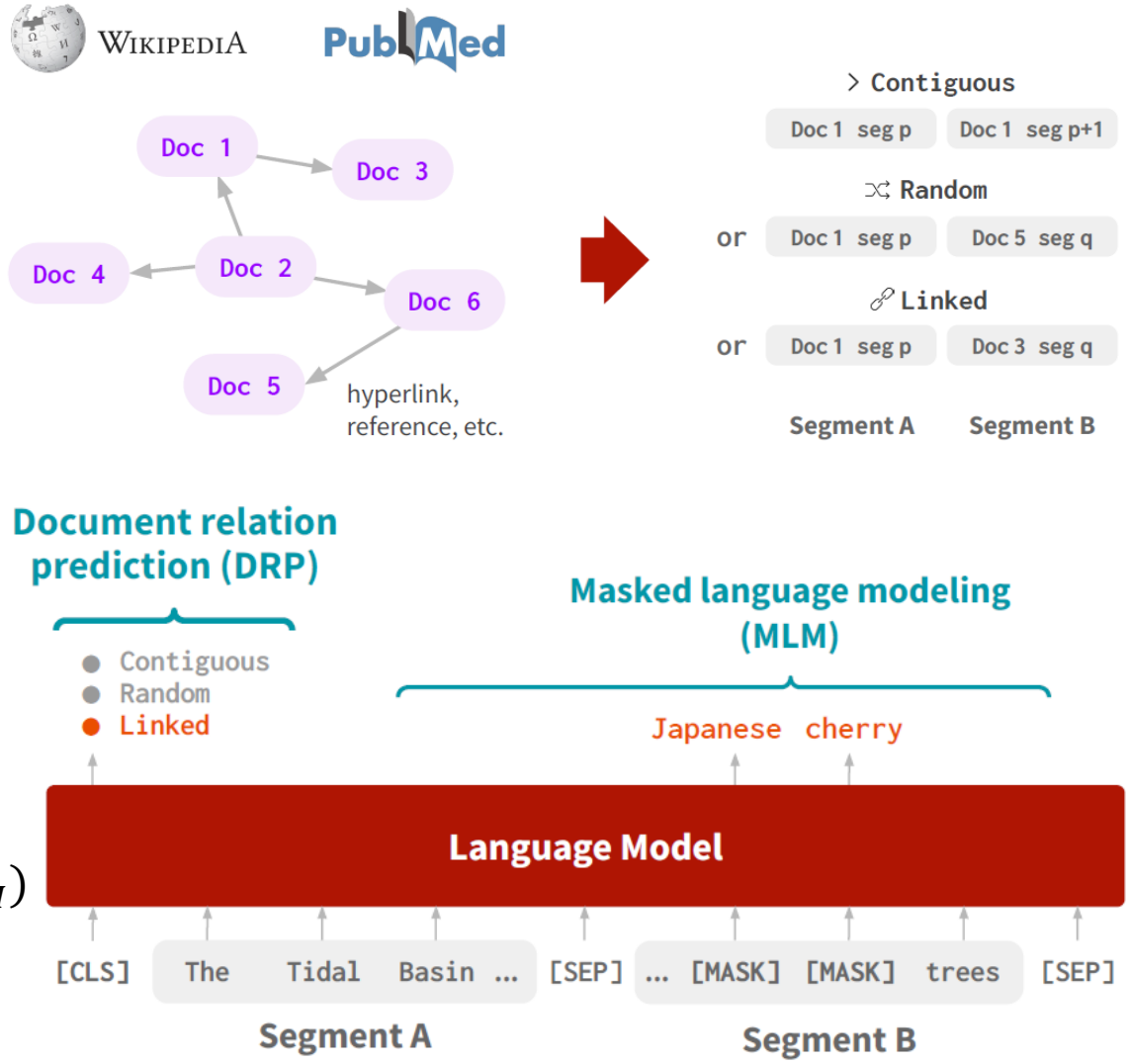
- 3. Contiguous documents.

$$\boldsymbol{H} = \mathrm{BERT}([\mathrm{CLS}]d_1[\mathrm{SEP}]d_1'[\mathrm{SEP}]), \text{ where } d_1' \subset d_1$$

- 4. Pre-training MLM loss.

$$\hat{\boldsymbol{p}}_{MLM} = \mathrm{softmax}(\boldsymbol{W}\boldsymbol{h}_{w_j}) \quad \mathcal{L}_{MLM} = \mathrm{CrosEntropy}(\hat{\boldsymbol{p}}_{MLM}, \boldsymbol{p}_{MLM})$$

Parameter for MLM



Question Answering

- 1. Linked documents.

$$\boldsymbol{H} = \mathrm{BERT}([\mathrm{CLS}]d_1[\mathrm{SEP}]d_3[\mathrm{SEP}]), \text{ where } d_3 \in \mathcal{N}(d_1)$$

- 2. Random documents.

$$\boldsymbol{H} = \mathrm{BERT}([\mathrm{CLS}]d_1[\mathrm{SEP}]d_5[\mathrm{SEP}]), \text{ where } d_5 \notin \mathcal{N}(d_1)$$

- 3. Contiguous documents.

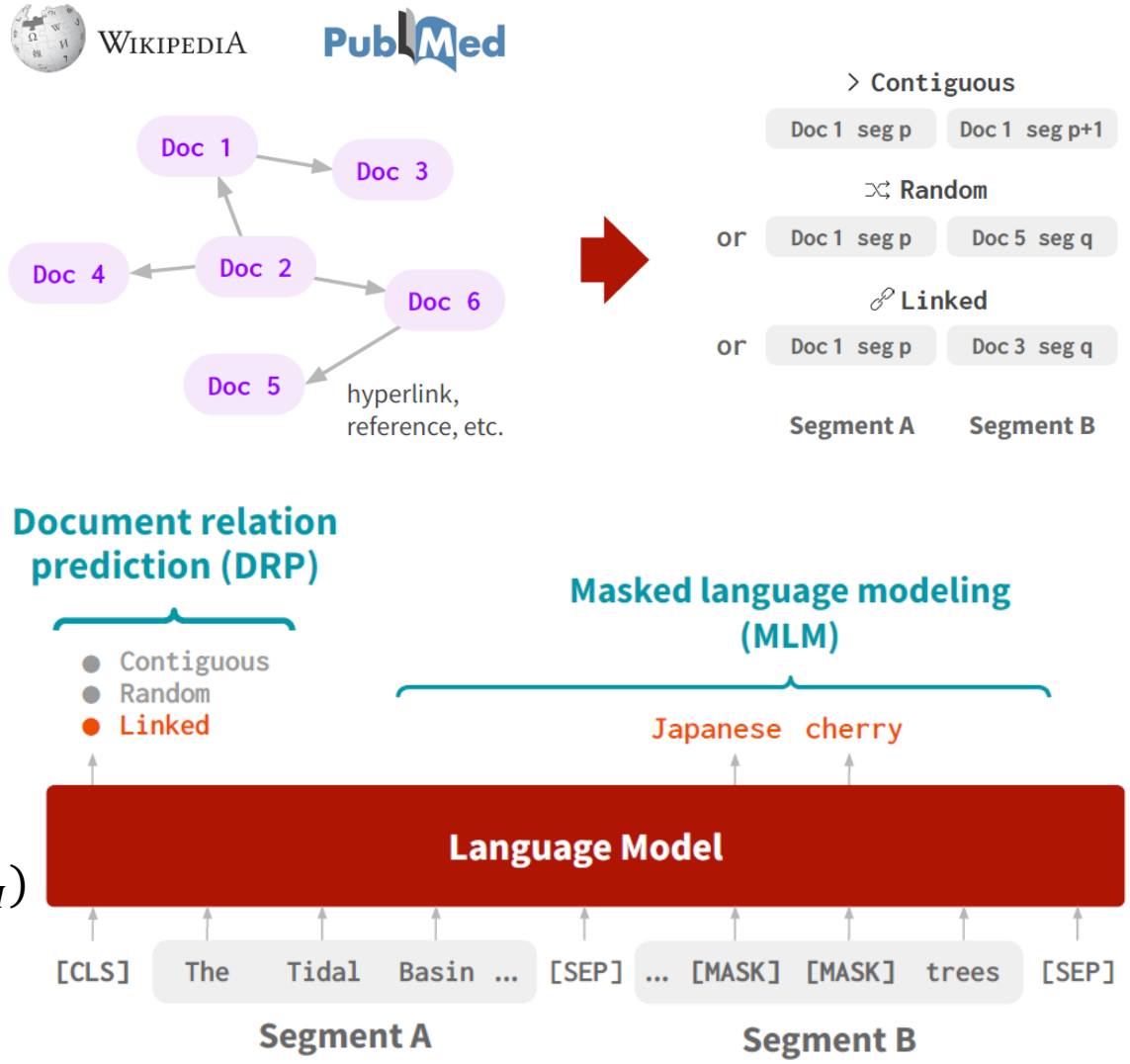
$$\boldsymbol{H} = \mathrm{BERT}([\mathrm{CLS}]d_1[\mathrm{SEP}]d_1'[\mathrm{SEP}]), \text{ where } d_1' \subset d_1$$

- 4. Pre-training MLM loss.

$$\hat{\boldsymbol{p}}_{MLM} = \mathrm{softmax}(\boldsymbol{W}\boldsymbol{h}_{w_j}) \quad \mathcal{L}_{MLM} = \mathrm{CrosEntropy}(\hat{\boldsymbol{p}}_{MLM}, \boldsymbol{p}_{MLM})$$

- 5. Pre-training relation prediction loss.

$$\hat{\boldsymbol{p}}_r = \mathrm{softmax}(\boldsymbol{W}'\boldsymbol{h}_{CLS}) \quad \mathcal{L}_{DRP} = \mathrm{CrossEntropy}(\hat{\boldsymbol{p}}_r, \boldsymbol{p}_r)$$



Question Answering

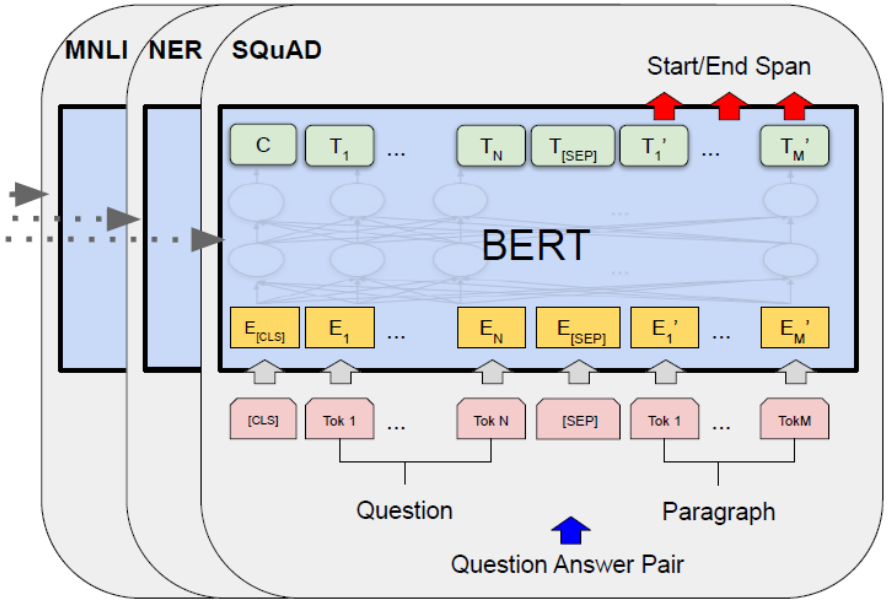
- **Experiments**
- 1. Datasets

Dataset	Train	Dev
SQuAD	86,588	10,507
NewsQA	74,160	4,212
TriviaQA♠	61,688	7,785
SearchQA♠	117,384	16,980
HotpotQA	72,928	5,904
Natural Questions	104,071	12,836

- LinkBERT, pre-trained on Wikipedia hyperlink graph.
- BioLinkBERT, pre-trained on PubMed citation graph.

Question Answering

- **Experiments**
- 2. Extractive question answering
- Given a document (or set of documents) and a question as input, the task is to identify an answer span from the document.



Question Answering

- **Experiments**
- 2. Extractive question answering
- Given a document (or set of documents) and a question as input, the task is to identify an answer span from the document.

	HotpotQA	TriviaQA	SearchQA	NaturalQ	NewsQA	SQuAD	Avg.
$\text{BERT}_{\texttt{tiny}}$	49.8	43.4	50.2	58.9	41.3	56.6	50.0
$\text{LinkBERT}_{\texttt{tiny}}$	**54.6**	**50.0**	**58.6**	**60.3**	**42.8**	**58.0**	**54.1**
$\text{BERT}_{\texttt{base}}$	76.0	70.3	74.2	76.5	65.7	88.7	75.2
$\text{LinkBERT}_{\texttt{base}}$	**78.2**	**73.9**	**76.8**	**78.3**	**69.3**	**90.1**	**77.8**
$\text{BERT}_{\texttt{large}}$	78.1	73.7	78.3	79.0	70.9	91.1	78.5
$\text{LinkBERT}_{\texttt{large}}$	**80.8**	**78.2**	**80.5**	**81.0**	**72.6**	**92.7**	**81.0**

Question Answering

- **Experiments**
- 2. Biomedical domain

	PubMed-BERT$_{base}$	BioLink-BERT$_{base}$	BioLink-BERT$_{large}$
Question answering			
PubMedQA (Jin et al., 2019)	55.84	**70.20**	**72.18**
BioASQ (Nentidis et al., 2019)	87.56	**91.43**	**94.82**

Section 4: Applications, Challenges, and Directions

- Section 4.1: Text Classification
- Section 4.2: Question Answering
- Section 4.3: Citation Recommendation
- Section 4.4: Challenges and Future Directions

Citation Recommendation

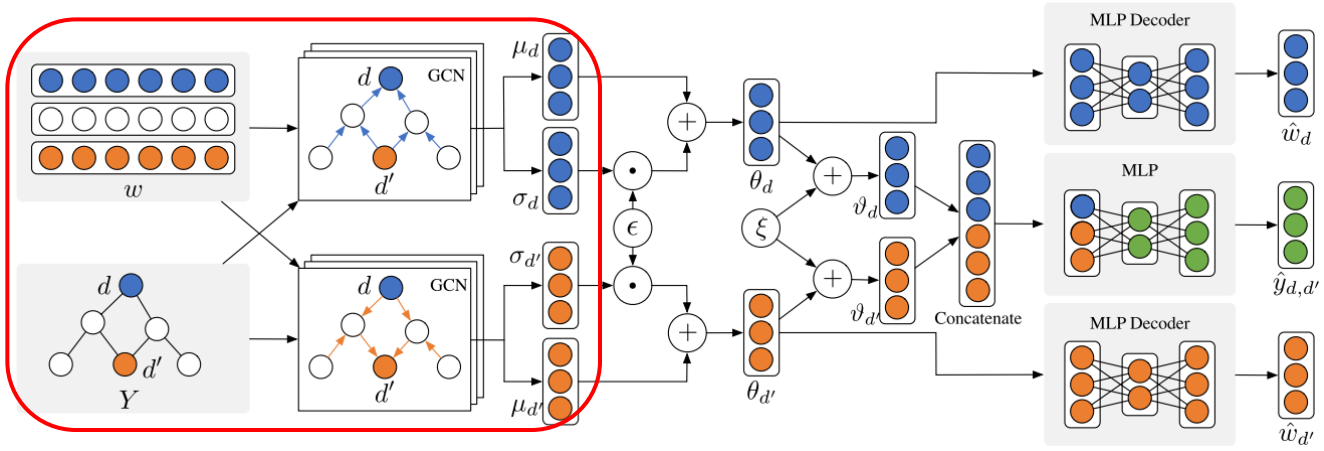
- Academic papers with citations constitute a Text-Attributed Graph.
- We can model both modalities for citation recommendation.
- We present GNCTM [1] in this section.

[1] Xie, Q., Zhu, Y., Huang, J., Du, P., & Nie, J. Y. (2021). Graph neural collaborative topic model for citation recommendation. ACM Transactions on Information Systems (TOIS), 40(3), 1-30.

Citation Recommendation

- 1. GNN encoding.

$$\boldsymbol{\mu}_i = \mathrm{GNN}_{\mu}(d_i, d_j | j \in \mathcal{N}(i)) \qquad \boldsymbol{\Sigma}_i = \mathrm{GNN}_{\Sigma}(d_i, d_j | j \in \mathcal{N}(i))$$



Citation Recommendation

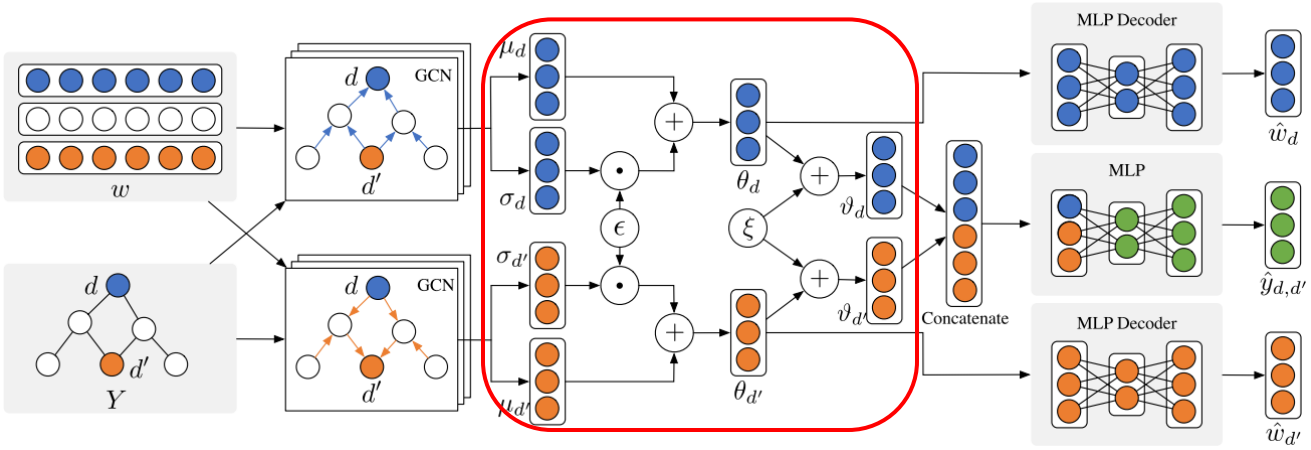
- 1. GNN encoding.

$$\boldsymbol{\mu}_i = \mathrm{GNN}_{\mu}(d_i, d_j | j \in \mathcal{N}(i)) \qquad \boldsymbol{\Sigma}_i = \mathrm{GNN}_{\Sigma}(d_i, d_j | j \in \mathcal{N}(i))$$

- 2. Reparameterization.

$$\boldsymbol{\theta}_i = \boldsymbol{\mu}_i + \boldsymbol{\Sigma}_i^{1/2}\boldsymbol{\epsilon}, \quad \text{where} \quad \boldsymbol{\epsilon} \sim \mathrm{Gaussian}(\mathbf{0}, \boldsymbol{I})$$

$$\boldsymbol{\vartheta}_i = \boldsymbol{\theta}_i + \boldsymbol{\xi}, \quad \text{where} \quad \boldsymbol{\xi} \sim \mathrm{Gaussian}(\mathbf{0}, \lambda \boldsymbol{I}) \longrightarrow \text{The purpose is to further distinguish similar documents}$$



Citation Recommendation

- 1. GNN encoding.

$$\boldsymbol{\mu}_i = \mathrm{GNN}_{\mu}(d_i, d_j | j \in \mathcal{N}(i)) \qquad \boldsymbol{\Sigma}_i = \mathrm{GNN}_{\Sigma}(d_i, d_j | j \in \mathcal{N}(i))$$

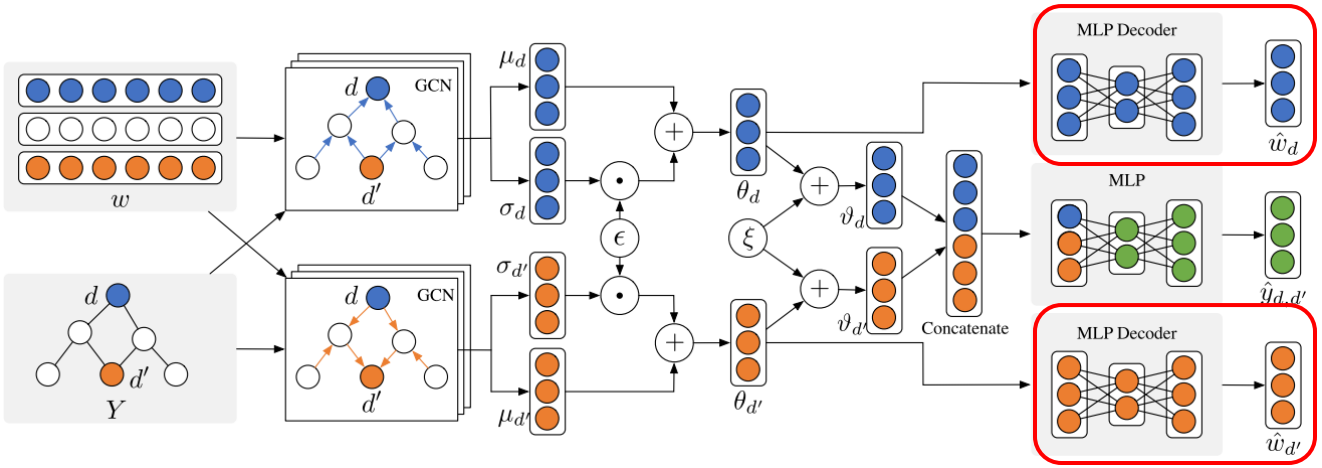
- 2. Reparameterization.

$$\boldsymbol{\theta}_i = \boldsymbol{\mu}_i + \boldsymbol{\Sigma}_i^{1/2}\boldsymbol{\epsilon}, \text{ where } \boldsymbol{\epsilon} \sim \mathrm{Gaussian}(\mathbf{0}, \boldsymbol{I})$$

$$\boldsymbol{\vartheta}_i = \boldsymbol{\theta}_i + \boldsymbol{\xi}, \text{ where } \boldsymbol{\xi} \sim \mathrm{Gaussian}(\mathbf{0}, \lambda \boldsymbol{I})$$

- 3. Text decoder.

$$\widehat{\boldsymbol{d}}_i = f_{MLP}(\boldsymbol{\vartheta}_i) \qquad \mathcal{L}_{text} = \mathrm{CrossEntropy}(\widehat{\boldsymbol{d}}_i, \boldsymbol{d}_i)$$



Citation Recommendation

- 1. GNN encoding.

$$\boldsymbol{\mu}_i = \mathrm{GNN}_\mu(d_i, d_j | j \in \mathcal{N}(i)) \qquad \boldsymbol{\Sigma}_i = \mathrm{GNN}_\Sigma(d_i, d_j | j \in \mathcal{N}(i))$$

- 2. Reparameterization.

$$\boldsymbol{\theta}_i = \boldsymbol{\mu}_i + \boldsymbol{\Sigma}_i^{1/2}\boldsymbol{\epsilon}, \text{ where } \boldsymbol{\epsilon} \sim \mathrm{Gaussian}(\mathbf{0}, \boldsymbol{I})$$

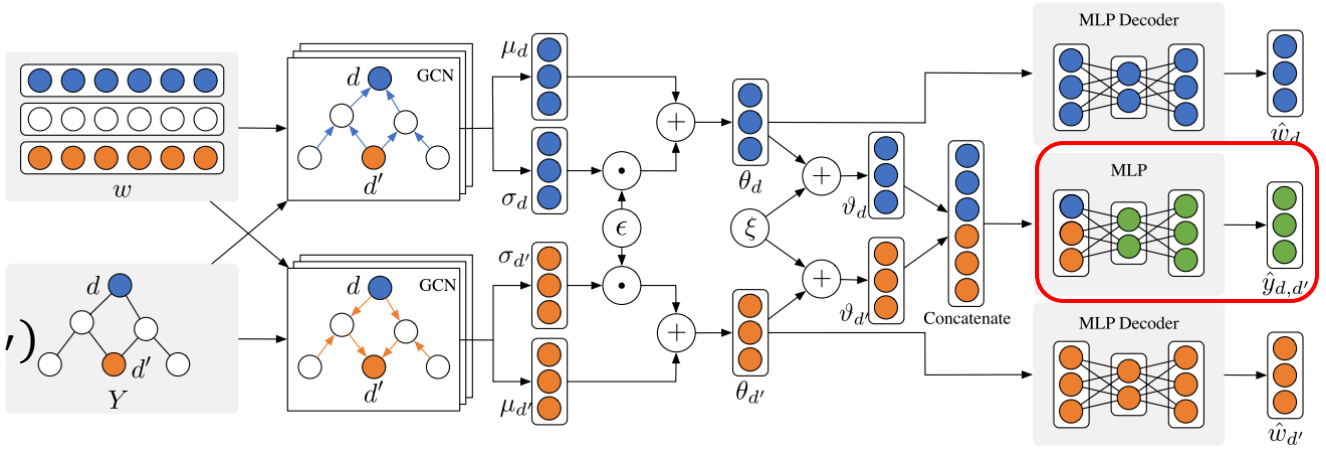
$$\boldsymbol{\vartheta}_i = \boldsymbol{\theta}_i + \boldsymbol{\xi}, \text{ where } \boldsymbol{\xi} \sim \mathrm{Gaussian}(\mathbf{0}, \lambda \boldsymbol{I})$$

- 3. Text decoder.

$$\hat{\boldsymbol{d}}_i = f_{MLP}(\boldsymbol{\vartheta}_i) \qquad \mathcal{L}_{text} = \mathrm{CrossEntropy}(\hat{\boldsymbol{d}}_i, \boldsymbol{d}_i)$$

- 4. Citation decoder.

$$\hat{y}_{i,i'} = f'_{MLP}(\boldsymbol{\vartheta}_i || \boldsymbol{\vartheta}_{i'}) \quad \mathcal{L}_{graph} = \mathrm{CrossEntropy}(\hat{y}_{i,i'}, y_{i,i'})$$



Citation Recommendation

- **Experiments**
- 1. Datasets

Datasets	Documents	Vocabulary size	Citations
Citeulike-a	16,980	8,000	44,709
Cora	13,147	17,509	57,018
MAG-50	126,666	8,000	2,452,340

Citation Recommendation

- **Experiments**
- 2. Citation recommendation

Models w/o high-order graph

Models w/o text decoding

Models w/o high-order encoding

HR@K (K=)	1	2	3	4	5	6	7	8	9	10
RTM	0.158	0.268	0.345	0.414	0.471	0.522	0.572	0.615	0.642	0.673
ProdLDA+MF	0.326	0.491	0.586	0.642	0.674	0.705	0.718	0.743	0.751	0.764
NMF	0.419	0.576	0.657	0.702	0.734	0.767	0.788	0.808	0.820	0.833
NCF	0.493	0.624	0.687	0.724	0.745	0.757	0.777	0.792	0.801	0.808
RDL	0.601	0.704	0.741	0.759	0.777	0.789	0.785	0.778	0.780	0.792
VGAE	0.428	0.606	0.716	0.787	0.831	0.865	0.891	0.913	0.925	0.938
NGCF	0.549	0.719	0.794	0.834	0.859	0.881	0.885	0.897	0.905	0.913
LightGCN	0.646	0.776	0.829	0.867	0.879	0.891	0.901	0.905	0.910	0.912
NRTM	0.613	0.776	0.854	0.897	0.919	0.940	0.951	0.957	0.968	0.966
AdjEnc	0.721	0.844	0.889	0.917	0.932	0.944	0.952	0.957	0.965	0.966
GNCTM	**0.742**	**0.873**	**0.909**	**0.936**	**0.952**	**0.964**	**0.973**	**0.978**	**0.981**	**0.986**

Section 4: Applications, Challenges, and Directions

- Section 4.1: Text Classification
- Section 4.2: Question Answering
- Section 4.3: Citation Recommendation
- Section 4.4: Challenges and Future Directions

Challenges and Future Directions

- **Explainability**
- We are curious about why a model makes certain predictions and how to explain its behaviors.
- GNNExplainer [1] pioneers this research by explaining which graph substructures are informative for predictions, but lacks textual semantics.
- A future research is to design a model that jointly incorporates both GNNs and PLMs/TMs, and provides explanations on text-attributed graph, e.g., which text spans are important for predictions.

[1] Ying, Z., Bourgeois, D., You, J., Zitnik, M., & Leskovec, J. (2019). Gnnexplainer: Generating explanations for graph neural networks. Advances in neural information processing systems, 32.

Challenges and Future Directions

- **Hierarchical Pre-training**
- Existing text-attributed graph pretraining treats all the documents equally. However, in many cases documents present a hierarchical instead of a flat structure.
- For example, survey papers summarize a broad area and regular papers deal with specific problems.
- Modeling such document hierarchy can better preserve textual semantics.

Section 4: Applications, Challenges, and Directions

- Section 4.1: Text Classification
- Section 4.2: Question Answering
- Section 4.3: Citation Recommendation
- Section 4.4: Challenges and Future Directions

Yale

Q & A