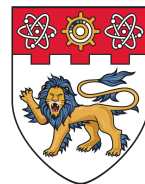


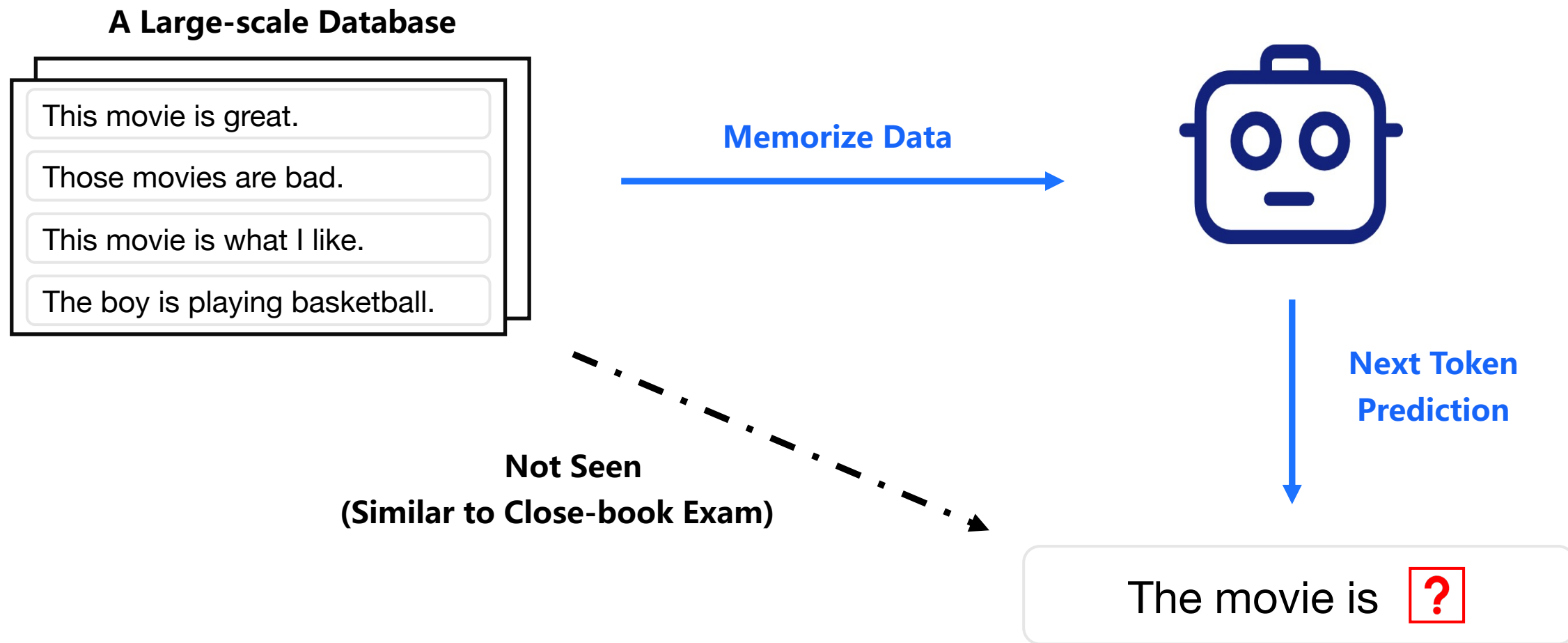
GNN-LM: Language Modeling Based on Global Contexts via GNN

Yuxian Meng, **Shi Zong**, Xiaoya Li, Xiaofei Sun, Tianwei Zhang, Fei Wu, Jiwei Li



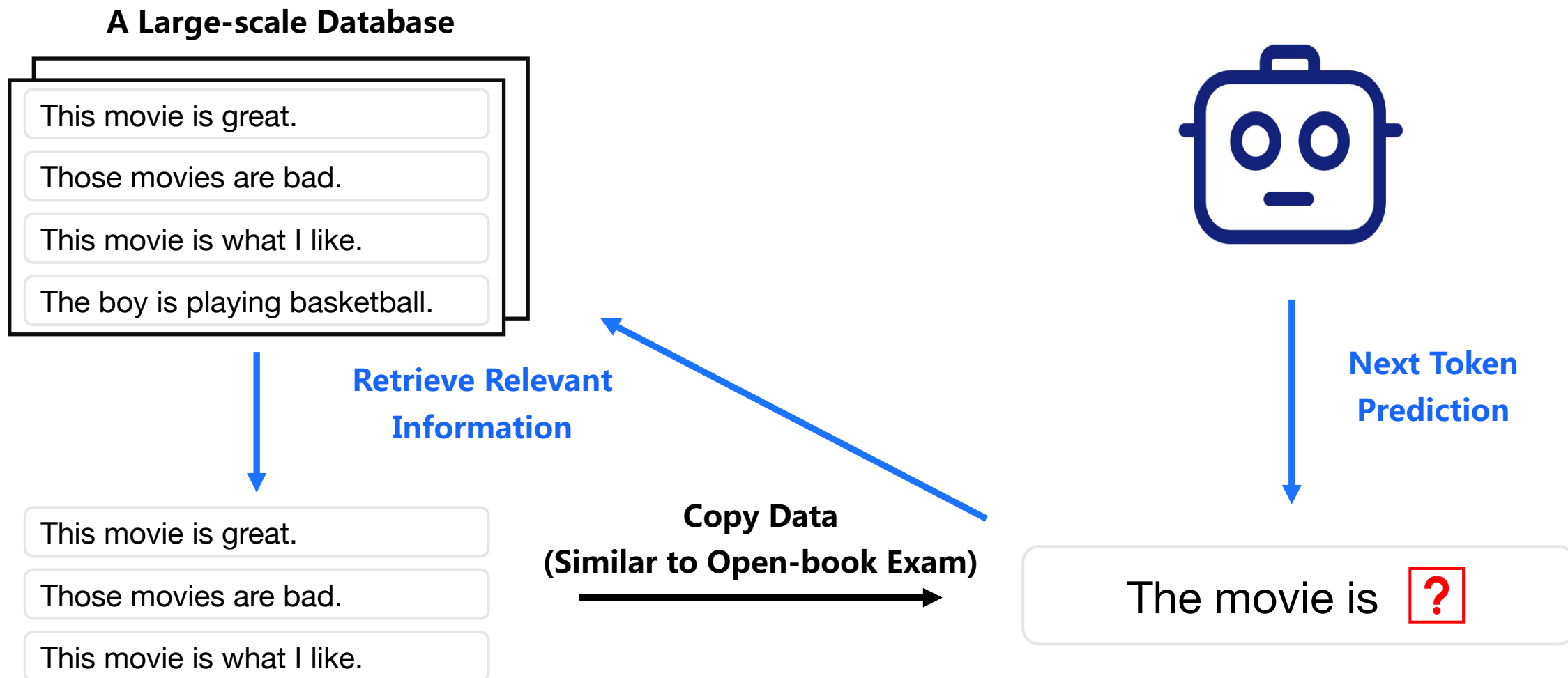
Memorization

Close-book Exam Strategy



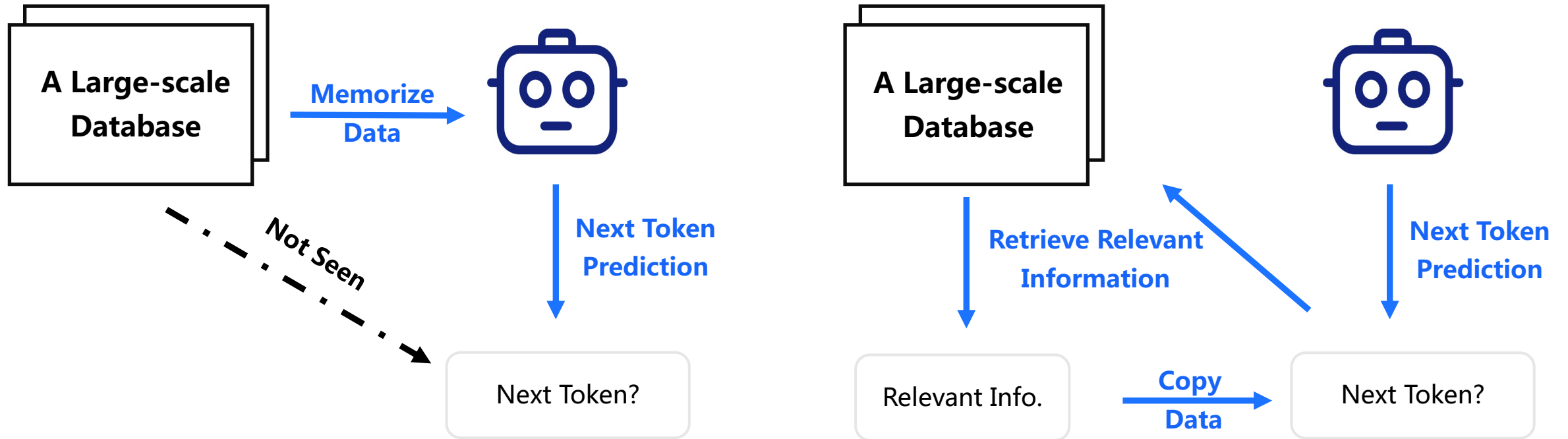
Copy

Open-book Exam Strategy



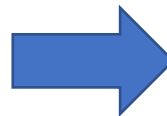
Copy, not Memorization!

Open-book Exam is Easier than Close-book



Memorization-based Method: Models have to memorize data and knowledge

Harder

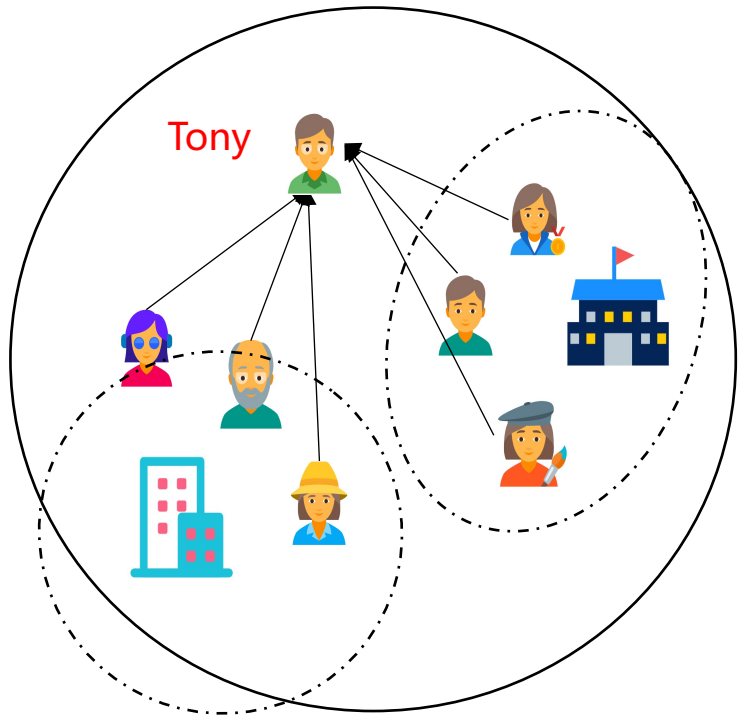


Open-book Exam Method: Models only need to retrieve and search for relevant information

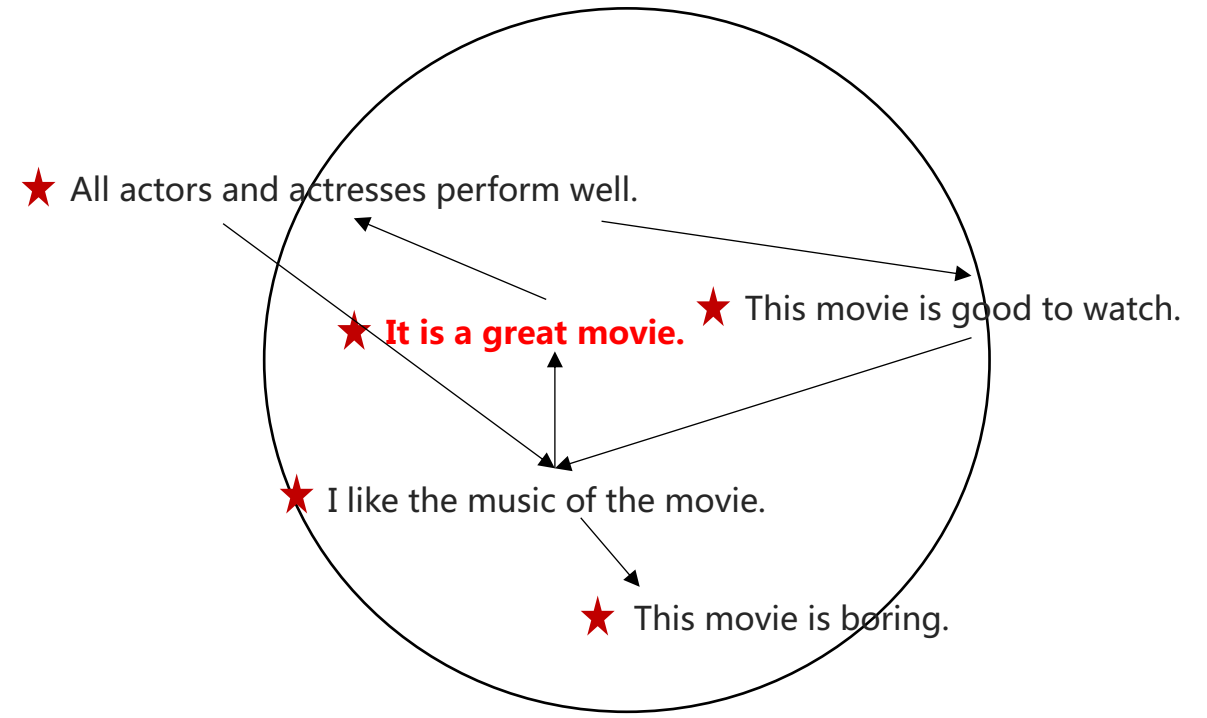
Easier

How to Retrieve Relevant Information?

A Graph Neural Network Strategy



We can understand people better by their social networks



We can obtain better sentence representations by text semantic networks

Data is connected by networks: we can search for relevant information via graph

Main Idea

Combining Copy Mechanism and Graph Together

- Open-book examination is easier compared to closed-book examination
- Incorporating information from neighbors of a sentence will improve its semantic representation

Main Idea

Combining Copy Mechanism and Graph Together

- Open-book examination is easier compared to closed-book examination
- Incorporating information from neighbors of a sentence will improve its semantic representation

○ Training Corpus

○ Global Context Graph

★ Training Sample

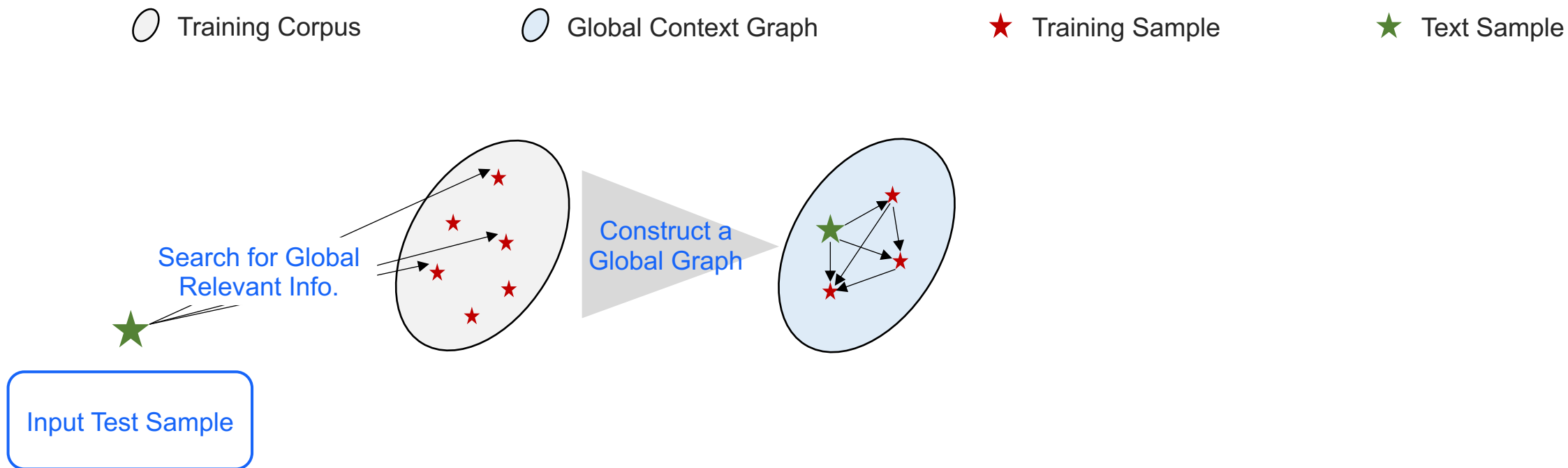
★ Text Sample



Main Idea

Combining Copy Mechanism and Graph Together

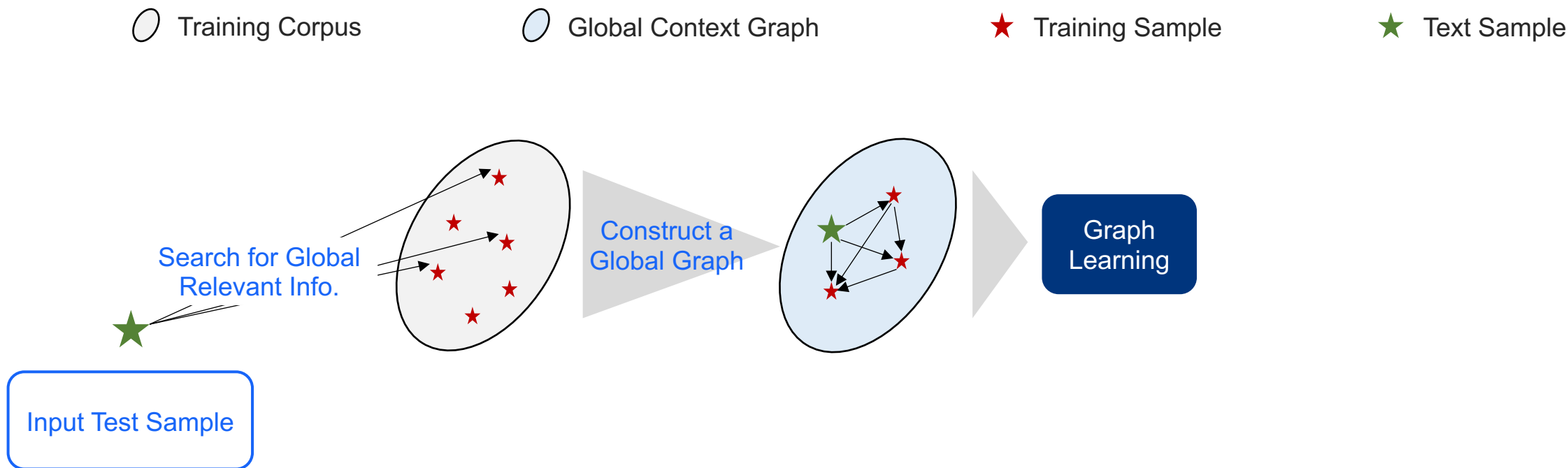
- Open-book examination is easier compared to closed-book examination
- Incorporating information from neighbors of a sentence will improve its semantic representation



Main Idea

Combining Copy Mechanism and Graph Together

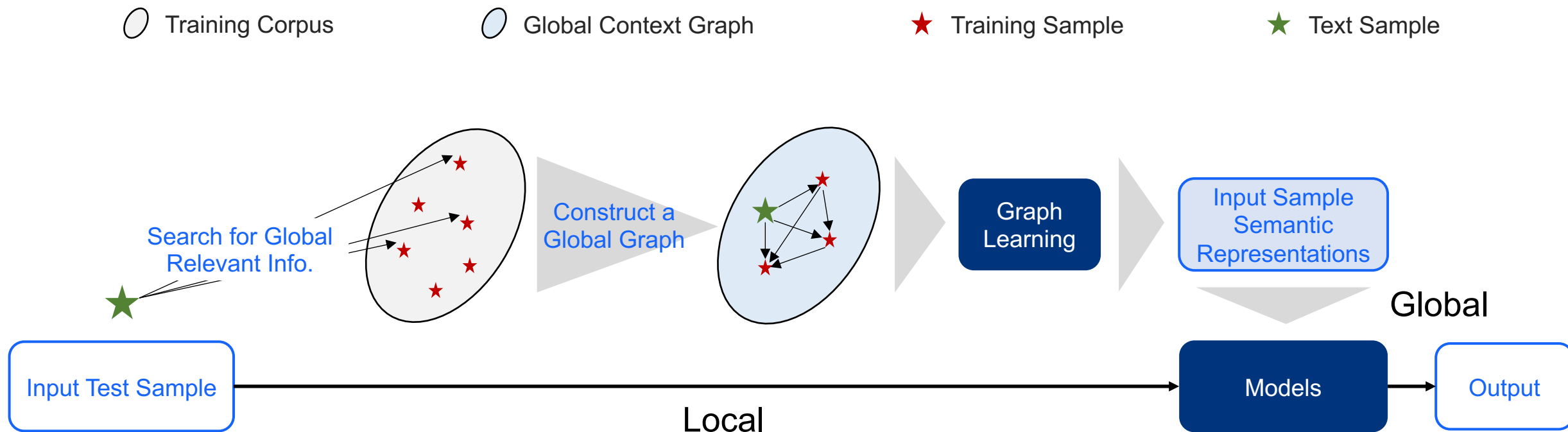
- Open-book examination is easier compared to closed-book examination
- Incorporating information from neighbors of a sentence will improve its semantic representation



Main Idea

Combining Copy Mechanism and Graph Together

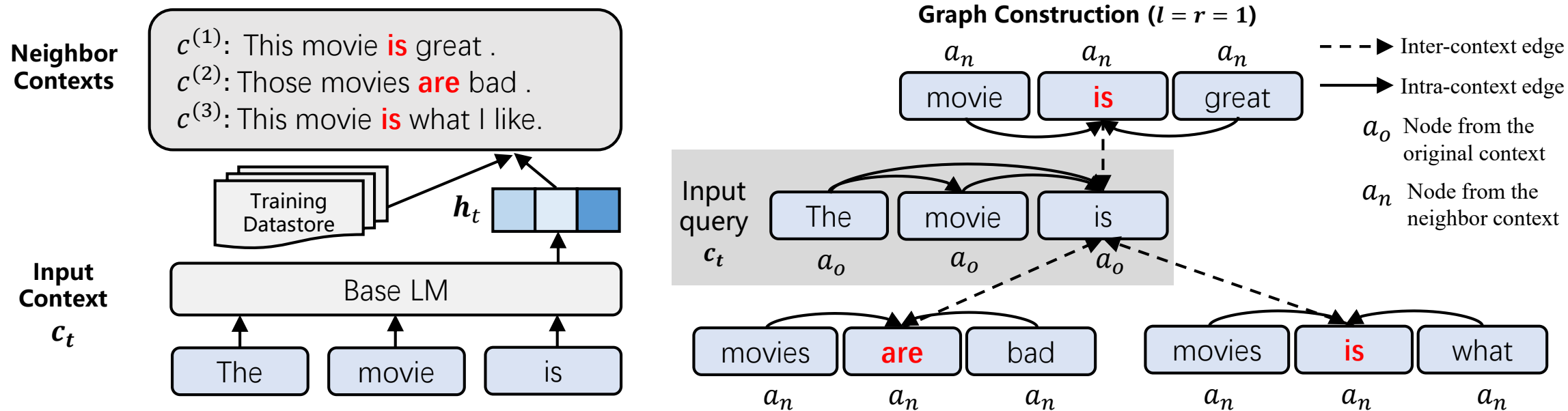
- Open-book examination is easier compared to closed-book examination
- Incorporating information from neighbors of a sentence will improve its semantic representation



GNN-LM

Overall Pipeline

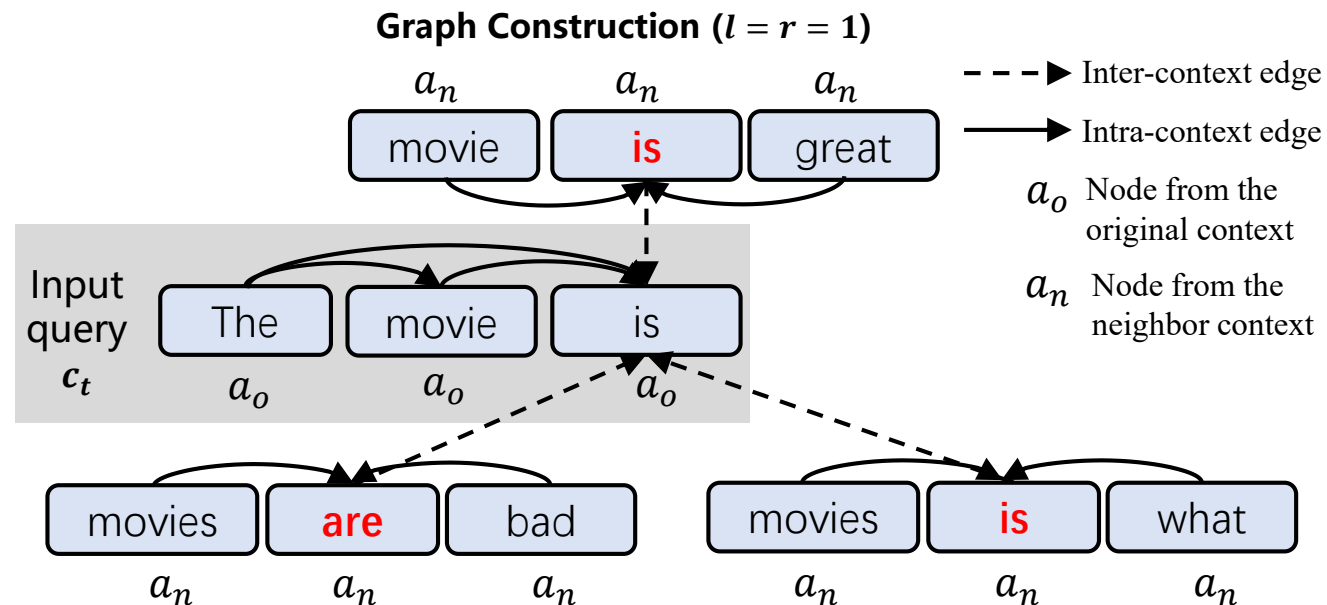
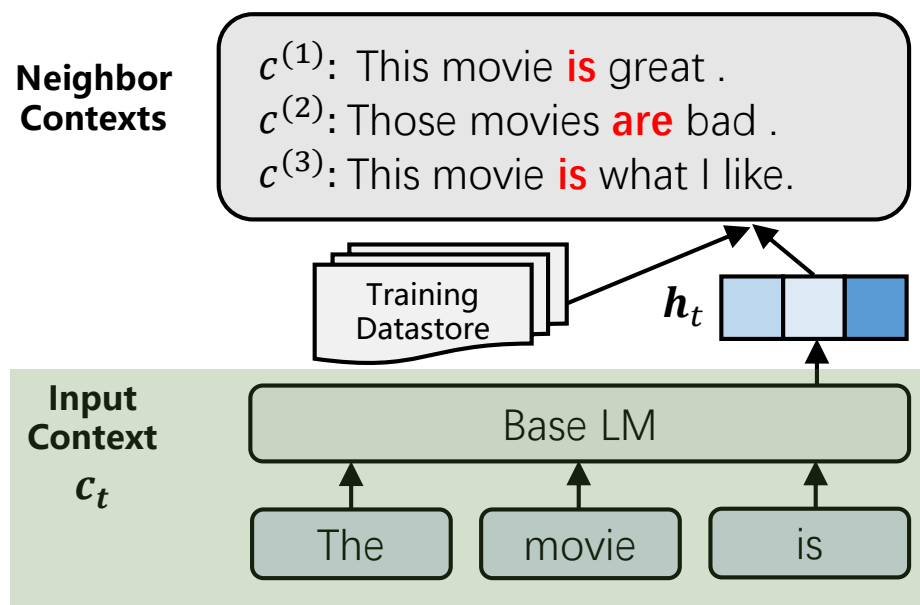
- Input context encoding
- Nearest neighbor contexts retrieval and graph construction
- Message passing on the built graph
- Token prediction for the next step



GNN-LM

Overall Pipeline

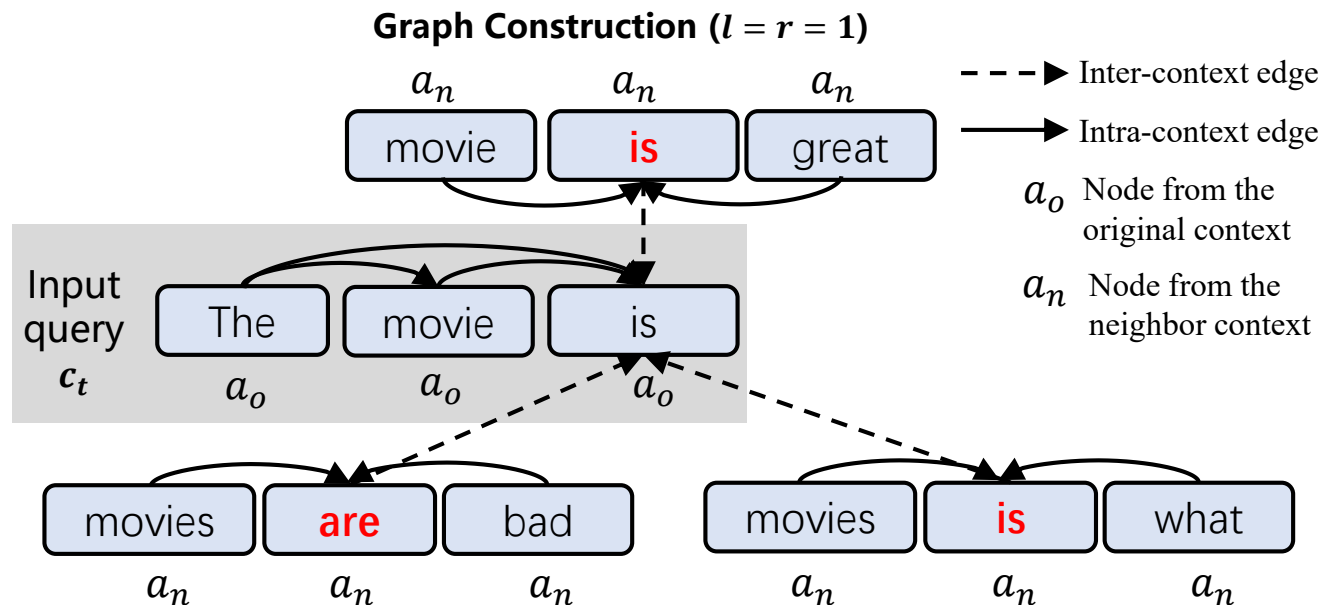
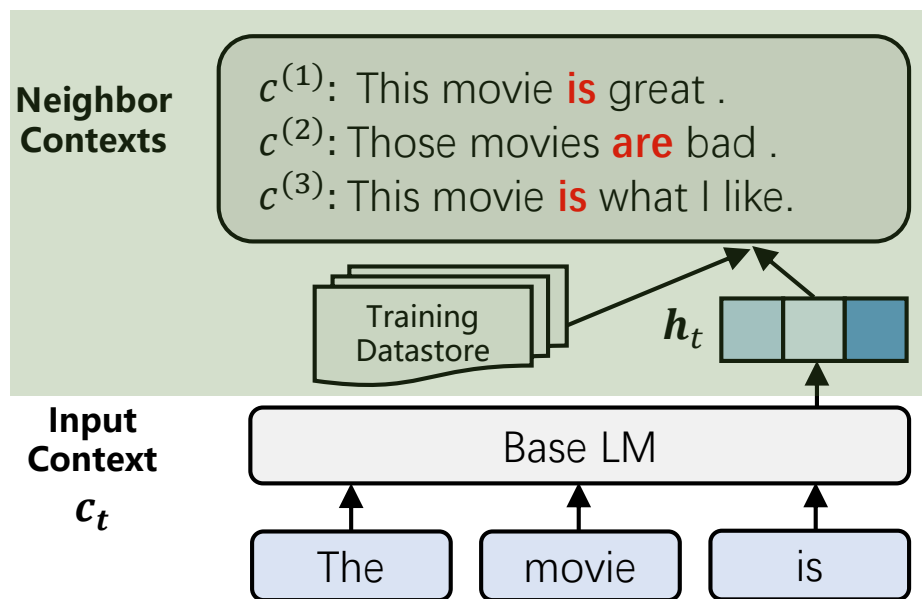
- Input context encoding
- Nearest neighbor contexts retrieval and graph construction
- Message passing on the built graph
- Token prediction for the next step



GNN-LM

Overall Pipeline

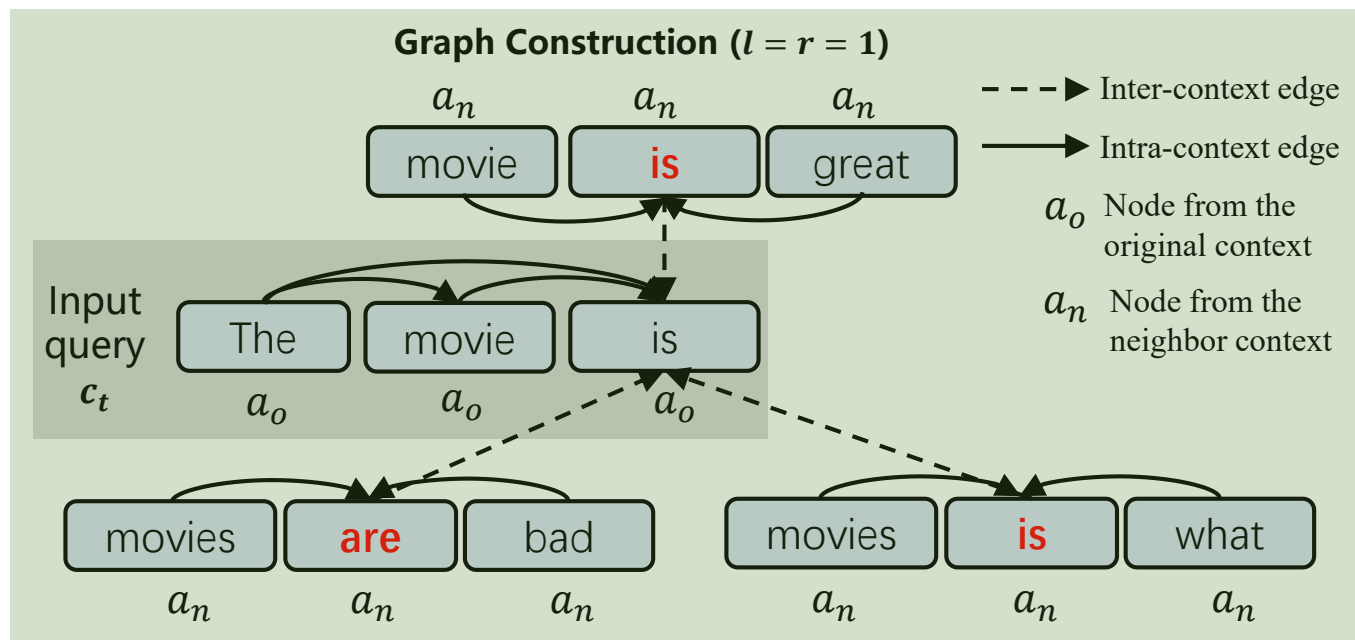
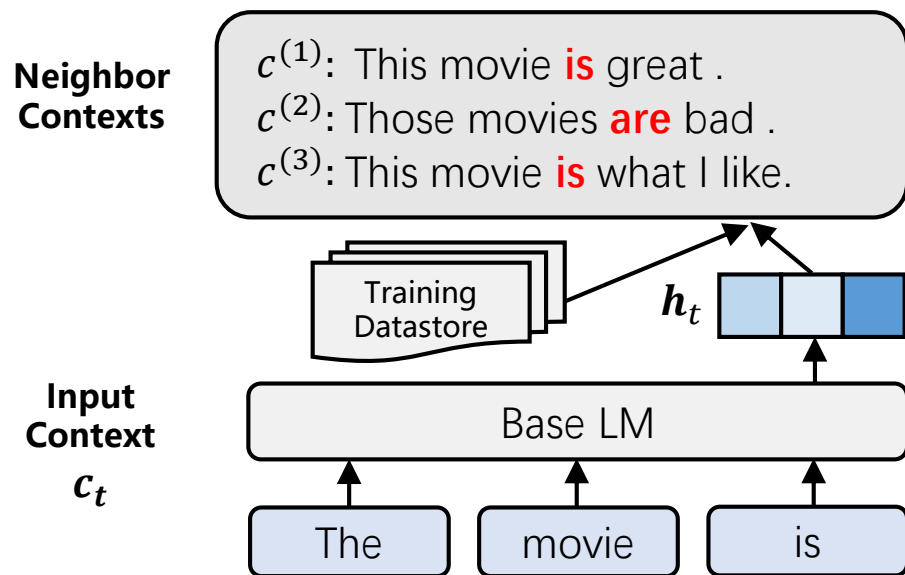
- Input context encoding
- Nearest neighbor contexts retrieval and graph construction
- Message passing on the built graph
- Token prediction for the next step



GNN-LM

Overall Pipeline

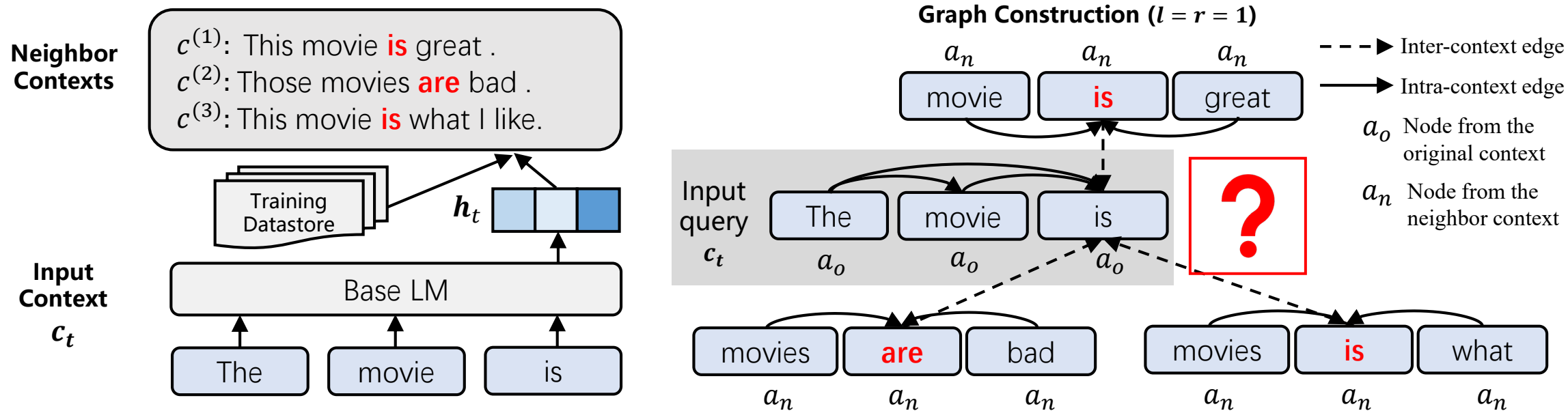
- Input context encoding
- Nearest neighbor contexts retrieval and graph construction
- Message passing on the built graph
- Token prediction for the next step



GNN-LM

Overall Pipeline

- Input context encoding
- Nearest neighbor contexts retrieval and graph construction
- Message passing on the built graph
- Token prediction for the next step

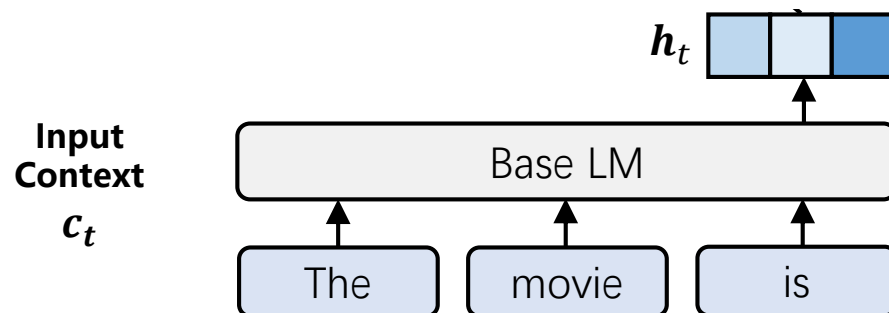


GNN-LM

Input Context Encoding

- A vanilla language model $f(\cdot)$ is used to encode a given input sequence $\mathbf{c}_t = (w_1, w_2, \dots, w_{t-1})$

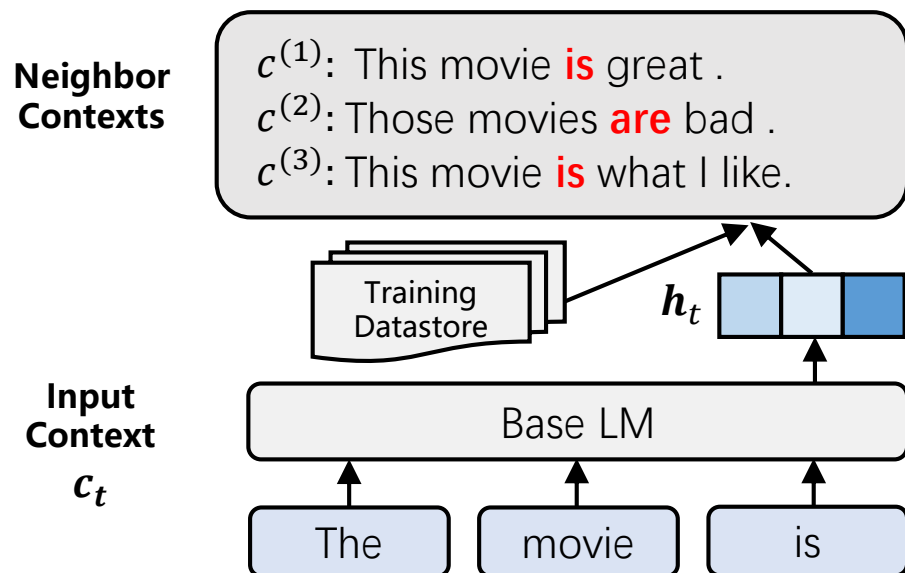
$$\mathbf{h}_t = f(\mathbf{c}_t) \in \mathbb{R}^d$$



GNN-LM

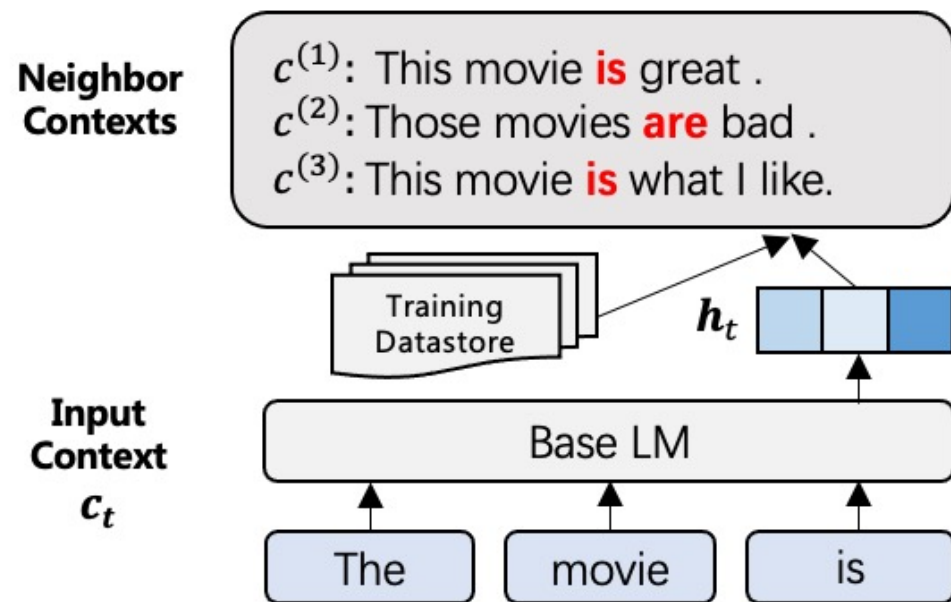
Near Neighbor Contexts Retrieval

- For an input context c_t , we retrieve k nearest neighbors $\mathcal{N}(c_t) = \{c_{t_1}^{(1)}, \dots, c_{t_k}^{(k)}\}$
 - h_t is used to retrieve top K tokens $\{w_j^{(i)}\}$
 - $w_j^{(i)}$ is expanded to $c_j^{(i)}$ by adding both left l and right r contexts: $c_j^{(i)} = \{w_{j+p}^{(i)}\}_{p=-l}^r$

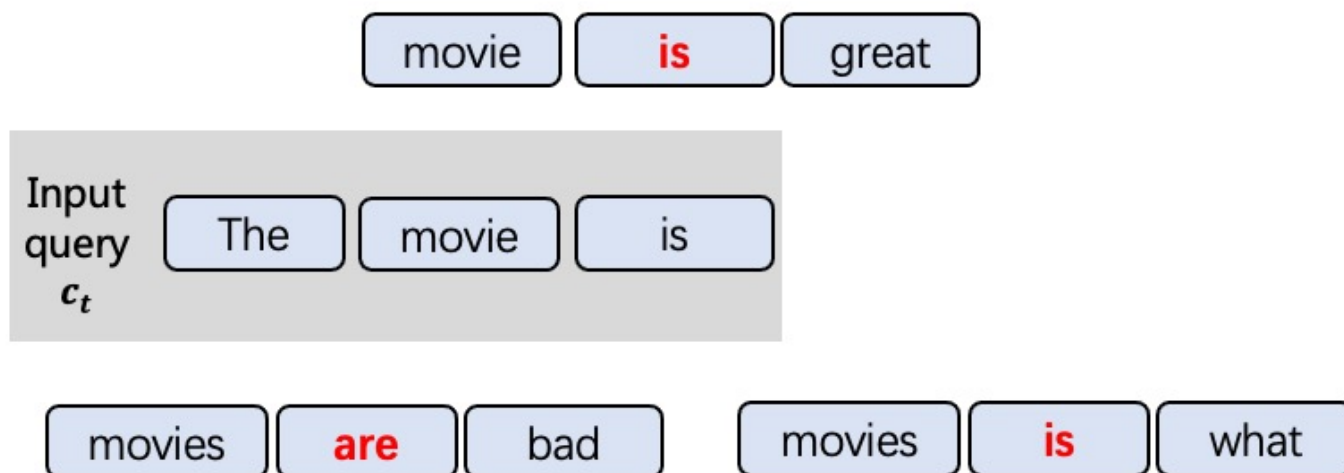


GNN-LM

Graph Construction

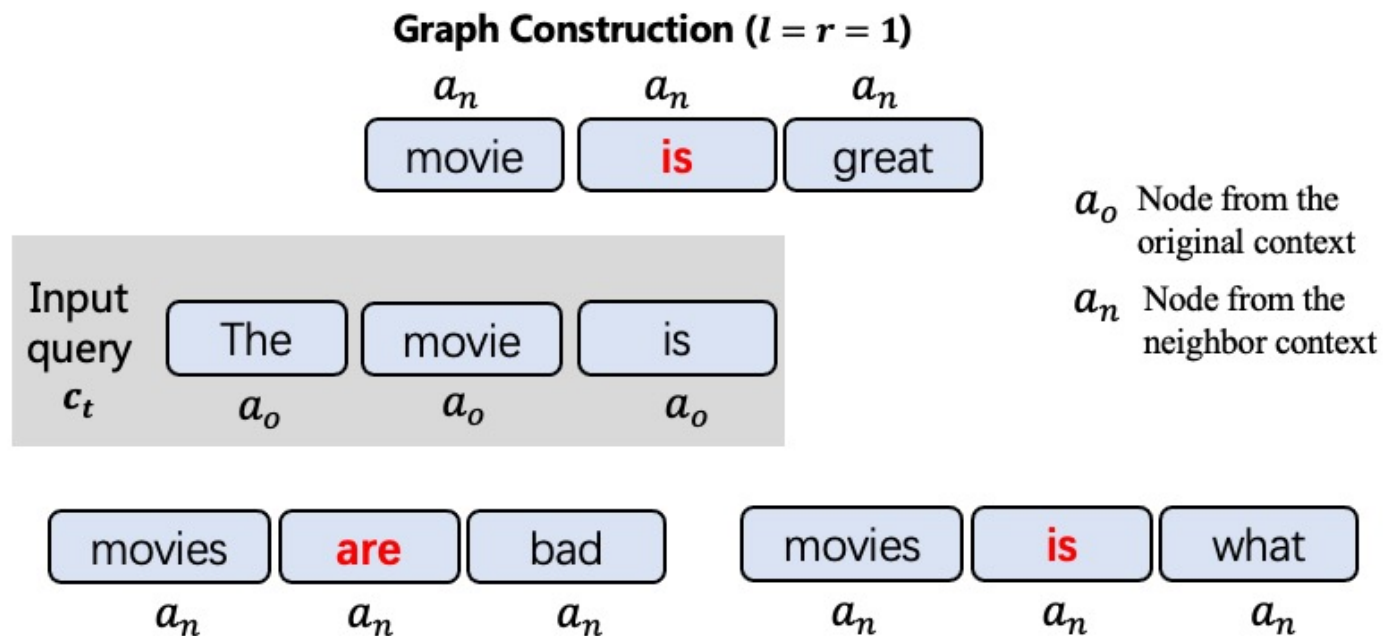
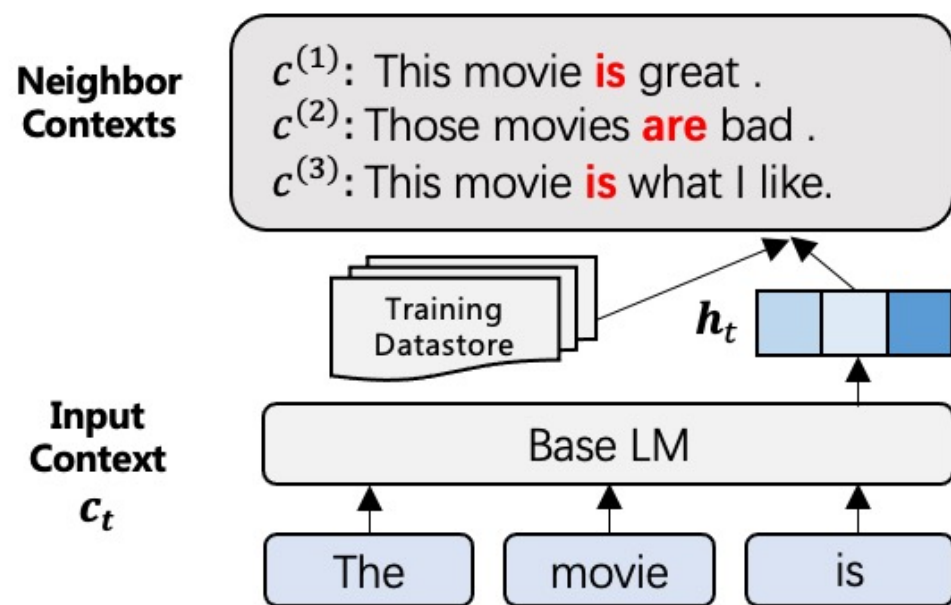


Graph Construction ($l = r = 1$)



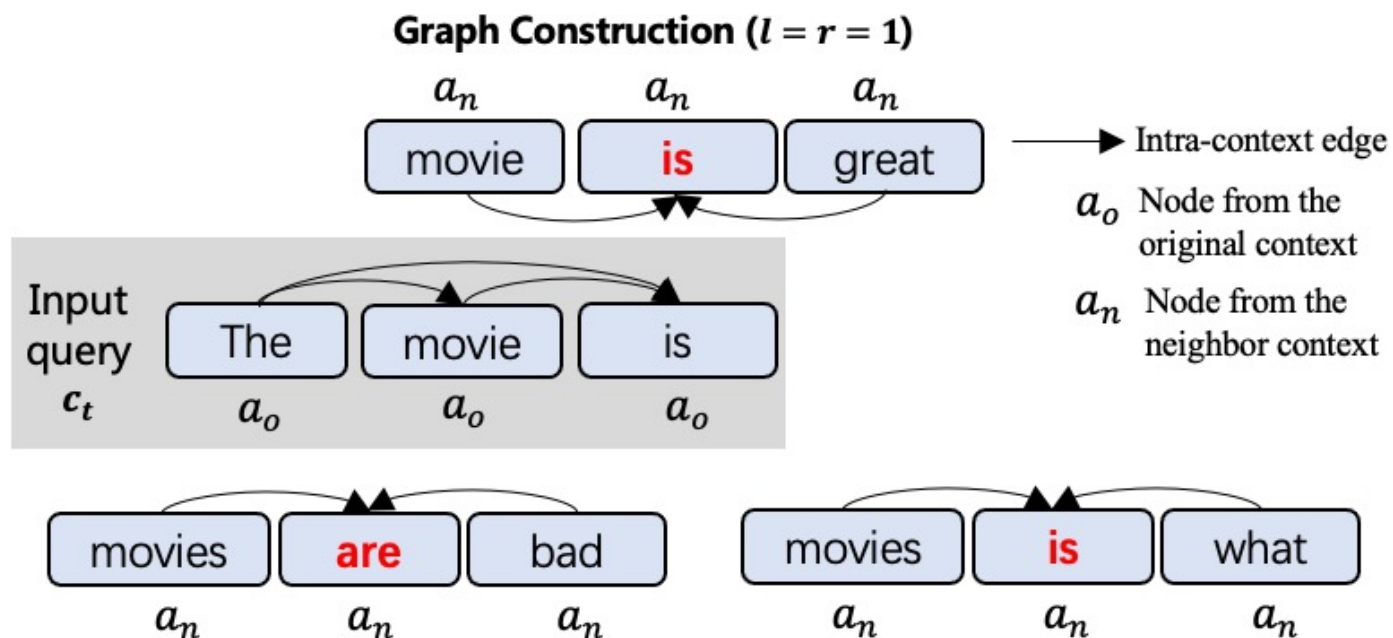
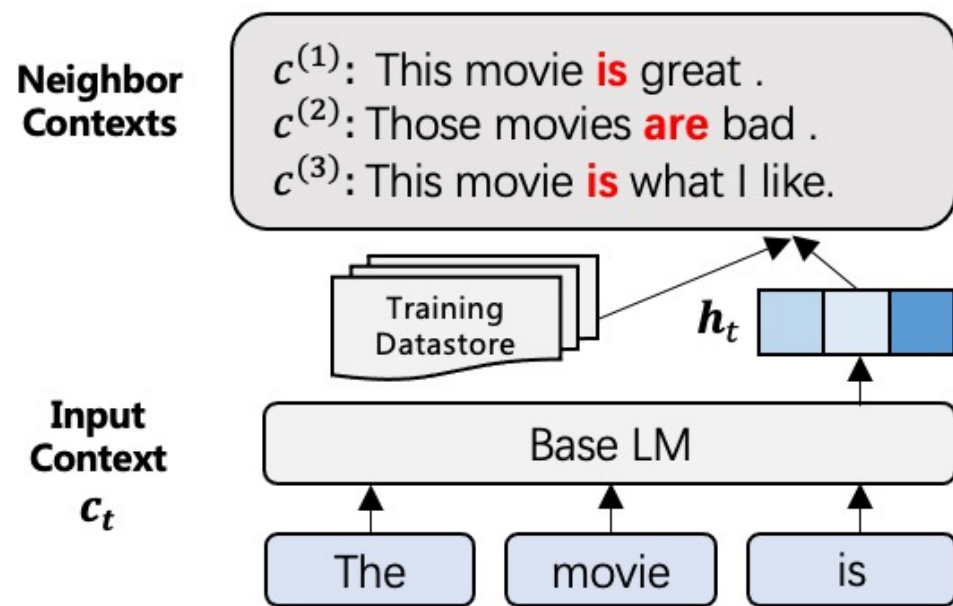
GNN-LM

Graph Construction



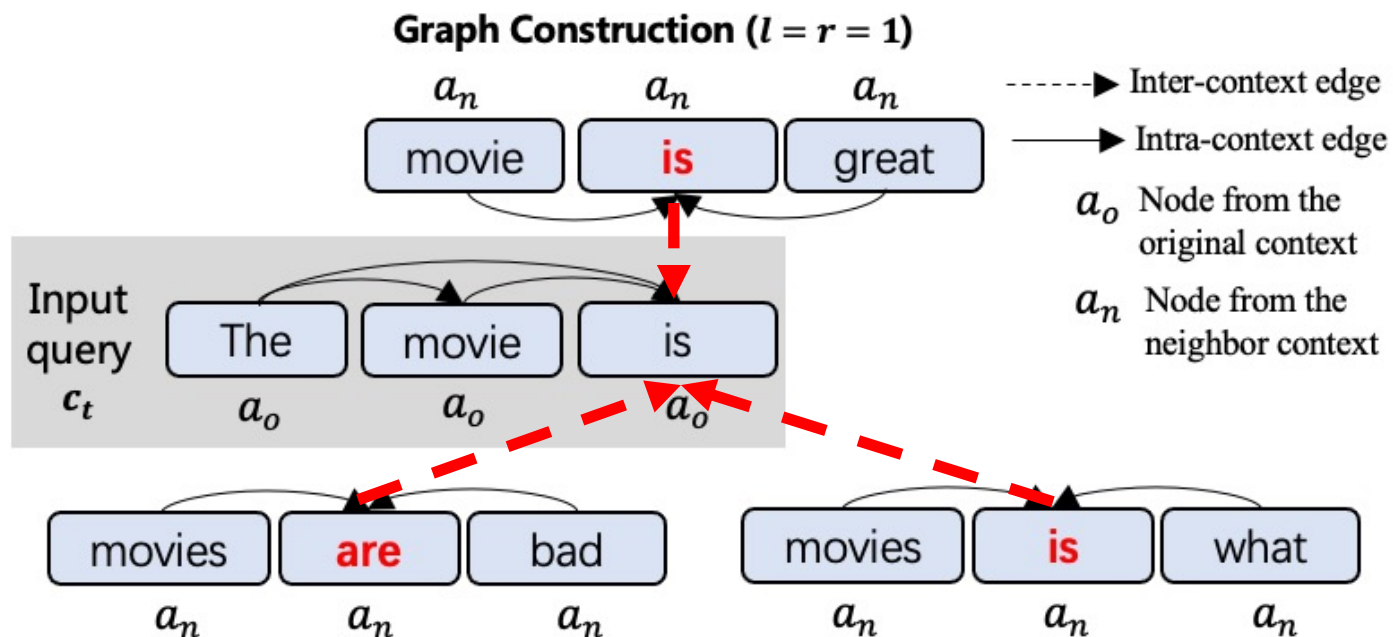
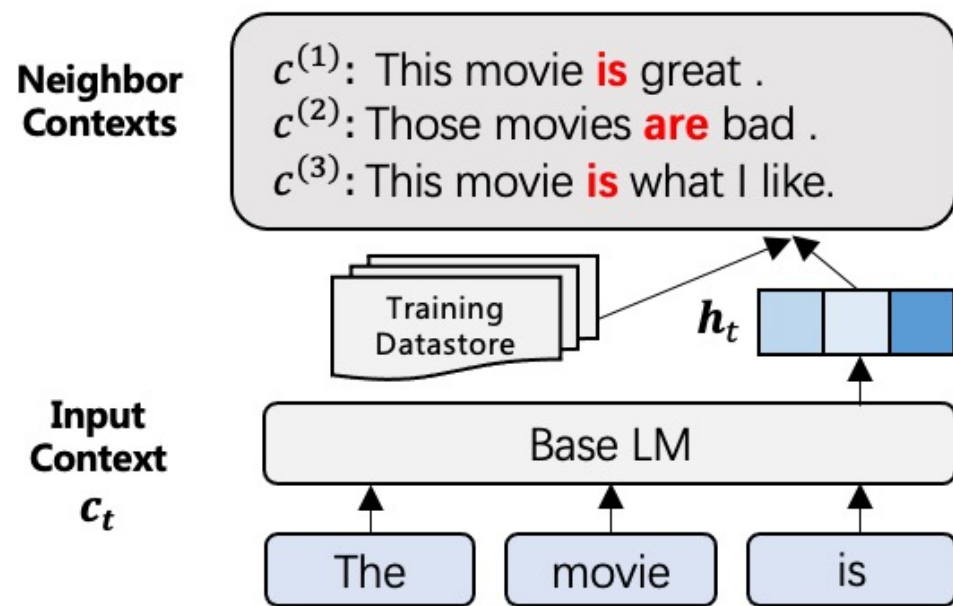
GNN-LM

Graph Construction



GNN-LM

Graph Construction



GNN-LM

GNN on the Constructed Graph

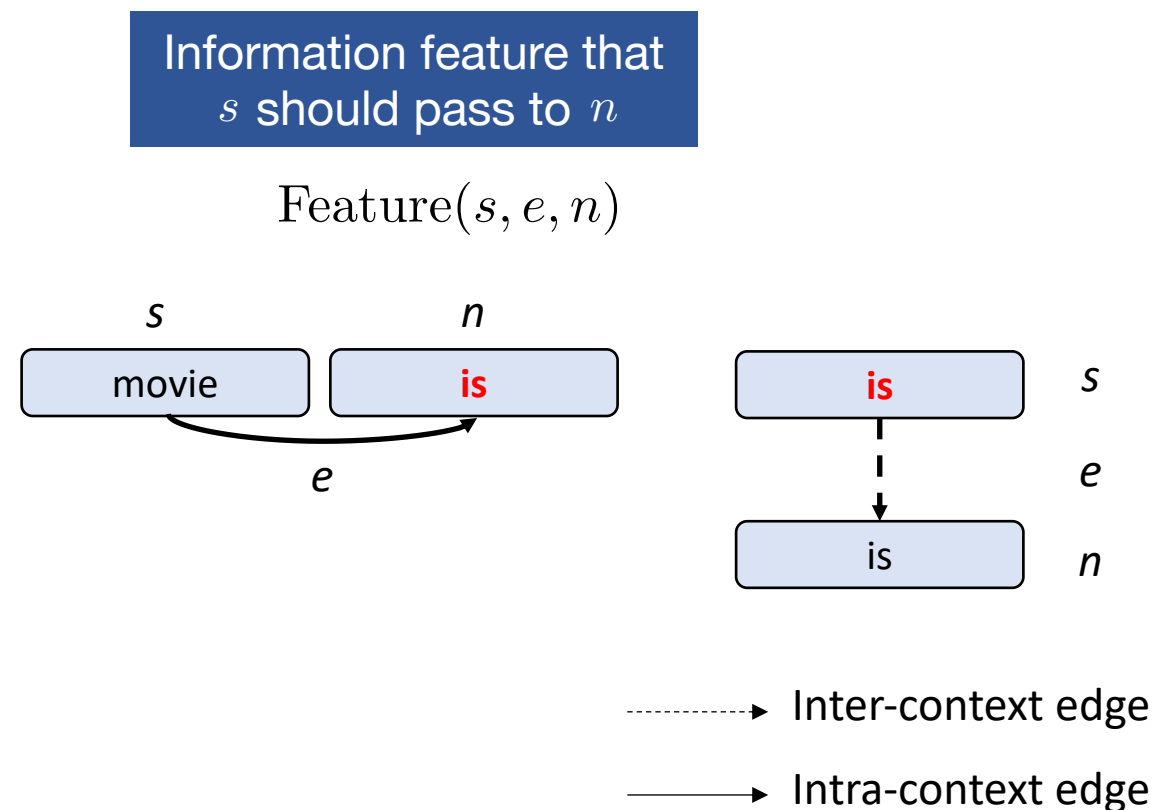
- A self-attention augmented GNN is applied on the heterogeneous graph
 - The l -th layer representation of node n is computed by the following

$$h_n^{[l]} = \boxed{\text{Updates from GNN}} + h_n^{[l-1]}$$

GNN-LM

GNN on the Constructed Graph

- A self-attention augmented GNN is applied on the heterogeneous graph
 - The l -th layer representation of node n is computed by:



GNN-LM

GNN on the Constructed Graph

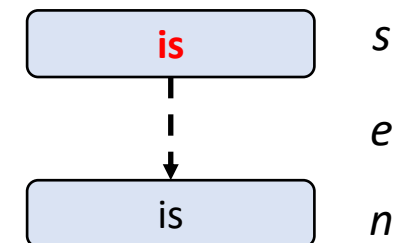
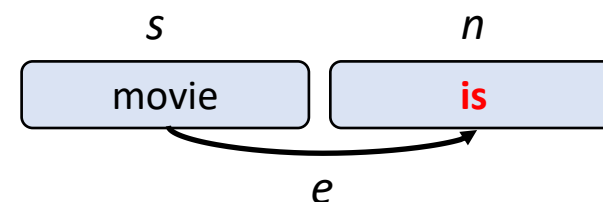
- A self-attention augmented GNN is applied on the heterogeneous graph
 - The l -th layer representation of node n is computed by:

Importance of the source node s
on target node n with relationship e

$\text{Attention}(s, e, n)$

Information feature that
 s should pass to n

$\text{Feature}(s, e, n)$



-----> Inter-context edge
-----> Intra-context edge

GNN-LM

GNN on the Constructed Graph

- A self-attention augmented GNN is applied on the heterogeneous graph
 - The l -th layer representation of node n is computed by:

Importance of the source node s
on target node n with relationship e

Information feature that
 s should pass to n

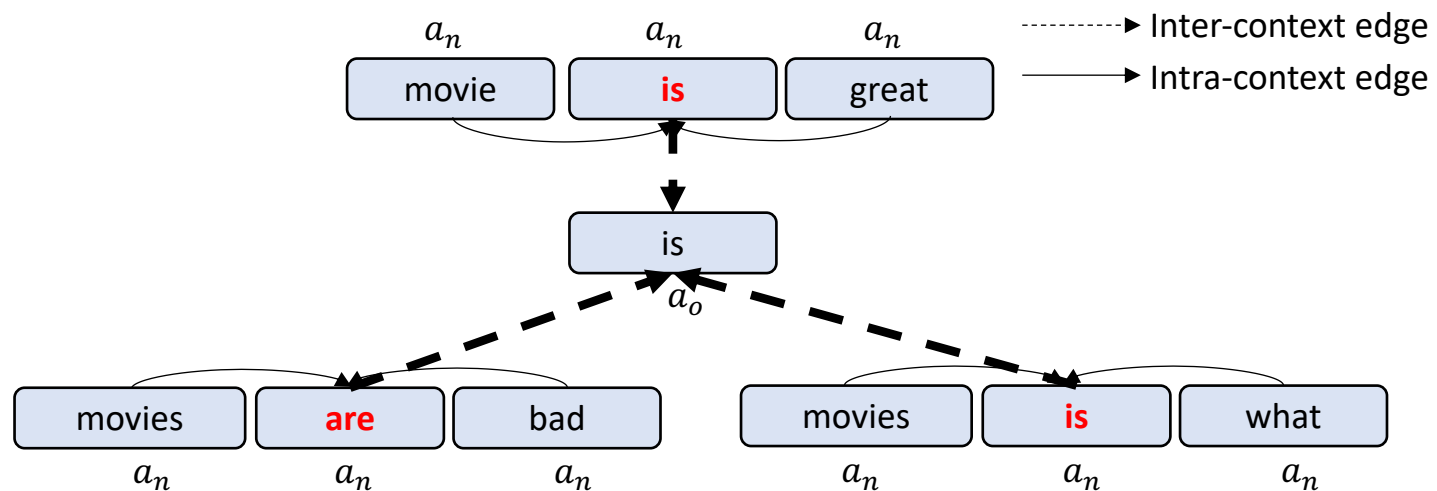
Aggregate(
 $\forall s \in \mathcal{N}(n)$

Attention(s, e, n)

.

Feature(s, e, n))

Aggregation between neighbors



GNN-LM

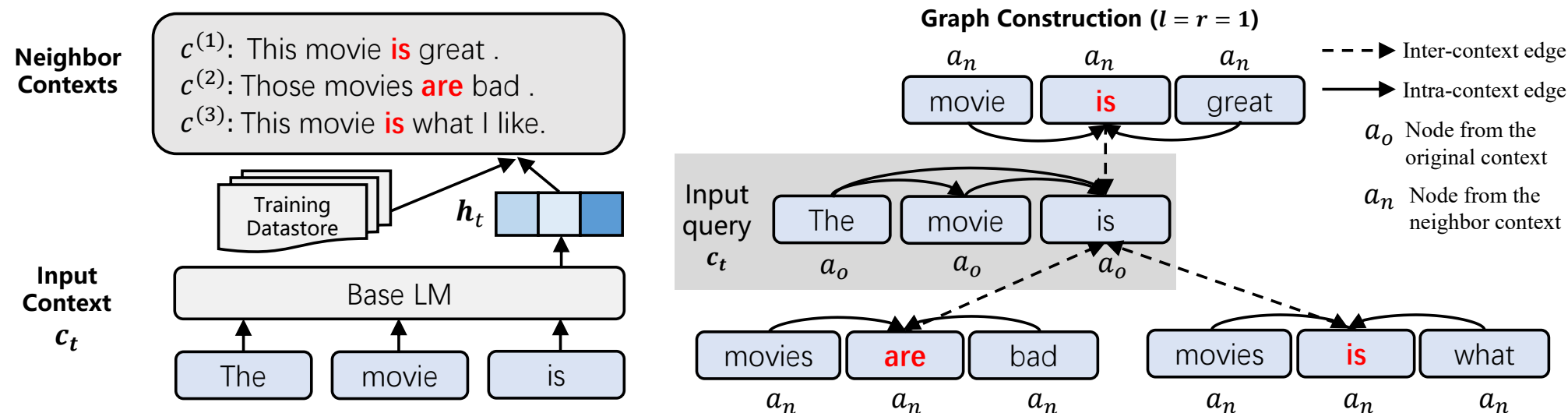
GNN on the Constructed Graph

- A self-attention augmented GNN is applied on the heterogeneous graph
 - The l -th layer representation of node n is computed by:

$$\mathbf{h}_n^{[l]} = \text{Aggregate}_{\forall s \in \mathcal{N}(n)} \left(\begin{array}{cc} \text{Importance of the source node } s & \text{Information feature that} \\ \text{on target node } n \text{ with relationship } e & s \text{ should pass to } n \end{array} \cdot \text{Feature}(s, e, n) \right) + \mathbf{h}_n^{[l-1]}$$

Aggregation between neighbors

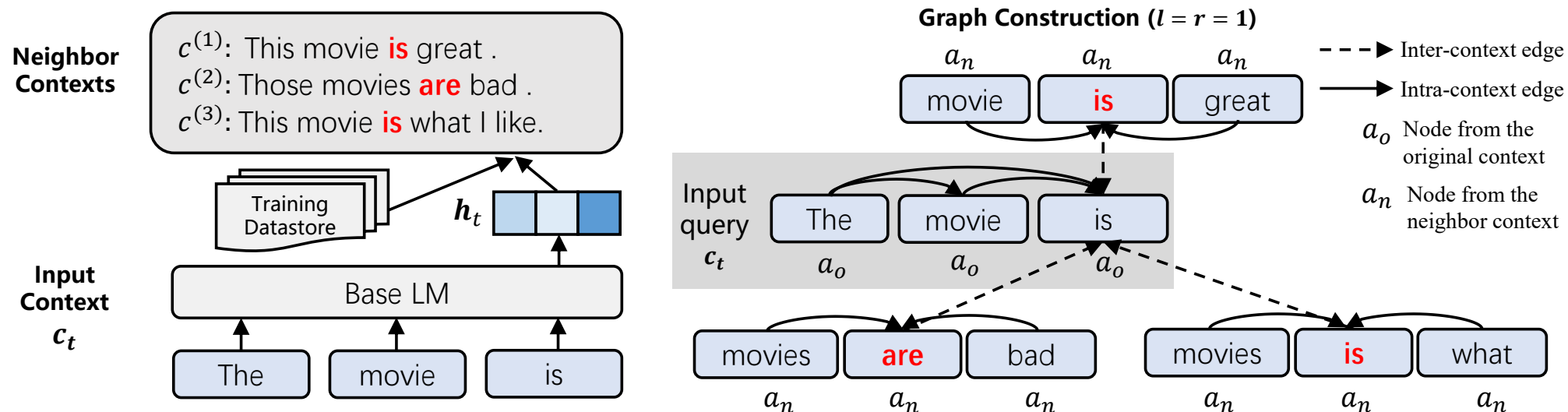
Next Token Prediction



- Additional information from reference tokens are incorporated via GNN

$$p_{\text{LM}}(w_t | c_t) = \text{softmax}(\mathbf{W} \mathbf{h}_t)$$

Next Token Prediction



- Additional information from reference tokens are incorporated via GNN

$$p_{\text{LM}}(w_t | c_t) = \text{softmax}(\mathbf{W}h_t)$$

- Probability for the next token is estimated by the linear interpolation of $p_{\text{LM}}(w_t | c_t)$ and $p_{\text{kNN}}(w_t | c_t)$

$$p(w_t | c_t) = \lambda p_{\text{kNN}}(w_t | c_t) + (1 - \lambda) p_{\text{LM}}(w_t | c_t)$$

kNN-LM (Khandelwal et al., 2019)

GNN-LM

Experimental Settings

- Datasets
 - WikiText-103 (Merity *et al.*, 2016)
 - One Billion Word (Chelba *et al.*, 2013)
 - enWik8 (Mahoney, 2011)
- Model configurations
 - 3-layer self-attention augmented GNN is added on the pretrained base LM
 - $k = 1,024$ nearest neighbors are retrieved for each source token, and the top 128 neighbors are used in graph

Experimental Results

WikiText-103 Dataset


Model	# Param	Test ppl (↓)
Hebbian + Cache (Rae et al., 2018)	151M	29.9
Transformer-XL (Dai et al., 2019)	257M	18.3
Transformer-XL + Dynamic Eval (Krause et al., 2019)	257M	16.4
Compressive Transformer (Rae et al., 2019)	-	17.1
KNN-LM + Cache (Khandelwal et al., 2019)	257M	15.8
Sandwich Transformer (Press et al., 2020a)	247M	18.0
Shortformer (Press et al., 2020b)	247M	18.2
SegaTransformer-XL (Bai et al., 2021)	257M	17.1
Routing Transformer (Roy et al., 2021)	-	15.8
base LM (Baevski & Auli, 2018)	247M	18.7
+GNN	274M	16.8
+GNN+ <i>k</i> NN	274M	14.8

- GNN-LM reduces the base LM perplexity from 18.7 to 16.8
- Combining GNN and *k*NN leads to a new **SOTA** result of **14.8**

Experimental Results

One Billion Word Dataset

Model	# Param	Test ppl (↓)
LSTM+CNN (Jozefowicz et al., 2016)	1.04B	30.0
High-Budget MoE (Shazeer et al., 2016)	5B	28.0
DynamicConv (Wu et al., 2018)	0.34B	26.7
Mesh-Tensorflow (Shazeer et al., 2018)	4.9B	24.0
Evolved Transformer (Shazeer et al., 2018)	-	28.6
Transformer-XL (Dai et al., 2019)	0.8B	21.8
Adaptive inputs (base) (Baevski & Auli, 2018)	0.36B	25.2
Adaptive inputs (large) (Baevski & Auli, 2018)	0.46B	23.9
base LM (Baevski & Auli, 2018)	1.03B	23.0
+ k NN	1.02B	22.8
+GNN	1.05B	22.7
+GNN+ k NN	1.05B	22.5




GNN-LM helps base LM reduce **0.5** perplexity with
only **27M** additional parameters

Experimental Results

One Billion Word Dataset

Model	# Param	Test ppl (↓)
LSTM+CNN (Jozefowicz et al., 2016)	1.04B	30.0
High-Budget MoE (Shazeer et al., 2016)	5B	28.0
DynamicConv (Wu et al., 2018)	0.34B	26.7
Mesh-Tensorflow (Shazeer et al., 2018)	4.9B	24.0
Evolved Transformer (Shazeer et al., 2018)	-	28.6
Transformer-XL (Dai et al., 2019)	0.8B	21.8
Adaptive inputs (base) (Baevski & Auli, 2018)	0.36B	25.2
Adaptive inputs (large) (Baevski & Auli, 2018)	0.46B	23.9
base LM (Baevski & Auli, 2018)	1.03B	23.0
+ k NN	1.02B	22.8
+GNN	1.05B	22.7
+GNN+ k NN	1.05B	22.5



GNN-LM helps base LM reduce **0.5** perplexity with only **27M** additional parameters

Experimental Results

Enwik8 Dataset

Model	# Param	BPC (↓)
64L Transformer (Al-Rfou et al., 2019)	235M	1.06
18L Transformer-XL (Dai et al., 2019)	88M	1.03
24L Transformer-XL (Dai et al., 2019)	277M	0.99
24L Transformer-XL + Dynamic Eval (Krause et al., 2019)	277M	0.94
Longformer (Beltagy et al., 2020)	102M	0.99
Adaptive Transformer (Sukhbaatar et al., 2019)	209M	0.98
Compressive Transformer (Rae et al., 2019)	277M	0.97
Sandwich Transformer (Press et al., 2020a)	209M	0.97
12L Transformer-XL (Dai et al., 2019)	41M	1.06
+ k NN	41M	1.04
+GNN	48M	1.04
+GNN+ k NN	48M	1.03

GNN- k NN-LM outperforms base LM by 0.03 Bit per Character (BPC), achieving 1.03 BPC with only **48M** parameters

Analysis

Neighbor Quality

- Given a sample $\mathbf{c}_t = (w_1, w_2, \dots, w_{t-1})$ and its k NN $\mathcal{N}(\mathbf{c}_t) = \{\mathbf{c}_{t_1}^{(1)}, \dots, \mathbf{c}_{t_k}^{(k)}\}$, we define the quality of k NN-retrieval as:

$$R(\mathbf{c}_t) = \sum_{i=1}^k \mathbb{1} \left[w_t = w_{t_i}^{(i)} \right]$$

kNN recall range	[0, 4)	[4, 27)	[27, 137)	[137, 463)	[463, 1024]
base LM	-7.14	-3.84	-2.21	-1.19	-0.30
+GNN+ k NN	-7.15	-3.46	-1.71	-0.80	-0.21
absolute improvement	-0.01	0.38	0.50	0.39	0.09
relative improvement	-0.0%	10%	23%	33%	32%

GNN- k NN-LM gains more relative improvements to base LM when the quality of k NN retrieval reaches a relatively high level

Examples

golden token to predict

extracted tokens

Input: In 2000 Boulter had a guest @-@ **starring**

Extracted 1: In 2009 , Beghe had a guest @-@ starring role on the television show Californication .

Extracted 2: had previously worked on Hack , for a guest @-@ starring episode arc on the show .

Extracted 3: and because of Patrick Stewart 's hilarious guest @-@ starring role as " Number One . "

Input: Tourism is a vital industry in Manila , **and**

Extracted 1: a large audience in Mogadishu , and was widely sold prior to the civil war .

Extracted 2: industry is well established , with Mumbai Port being one of the oldest and most

Extracted 3: transportation has become a large business in Newark , accounting for more than 17

Extracted contexts have a strong connection in semantics to the input,
thus leveraging the neighboring information will benefit model predictions!

Take Away

- We propose GNN-LM, a new paradigm for language modeling
 - Enable a LM model to reference similar contexts from the entire training corpus as cues for prediction
- Our method outperforms strong baselines in benchmark datasets
 - Achieve the state-of-the-art results on WikiText-103 with a test perplexity of 14.8

Code is publicly available at:

<https://github.com/ShannonAI/GNN-LM>