Neural Sentence Ordering Based on Constraint Graphs

Yutao Zhu[♠], Kun Zhou[♡], Jian-Yun Nie[♠], Shengchao Liu[♠♣], Zhicheng Dou[◇]
[♠] Université de Montréal, Montréal, Québec, Canada
[♡] School of Information, Renmin University of China, Beijing, China
[♣] Mila - Québec Artificial Intelligence Institute, Montréal, Canada
[◇] Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
yutao.zhu@umontreal.ca, francis_kun_zhou@163.com, nie@iro.umontreal.ca,
liusheng@mila.quebec, dou@ruc.edu.cn

Abstract

Sentence ordering aims at arranging a list of sentences in the correct order. Based on the observation that sentence order at different distances may rely on different types of information, we devise a new approach based on multi-granular orders between sentences. These orders form multiple constraint graphs, which are then encoded by Graph Isomorphism Networks and fused into sentence representations. Finally, sentence order is determined using the order-enhanced sentence representations. Our experiments on five benchmark datasets show that our method outperforms all the existing baselines significantly, achieving a new state-of-the-art performance. The results demonstrate the advantage of considering multiple types of order information and using graph neural networks to integrate sentence content and order information for the task. Our code is available at https://github.com/DaoD/ConstraintGraph4NSO.

Introduction

Text coherence is an essential problem in natural language processing (NLP). Coherent texts with well-organized logical structures are much easier for people to read and understand. As one subtask of coherence modeling, sentence ordering (Barzilay and Lapata 2008) aims at learning to reconstruct a coherent paragraph from an unordered set of sentences. This task underlies many downstream applications such as determining the ordering of: concepts in concept-to-text generation (Konstas and Lapata 2012, 2013), answer spans in retrieval-based question answering (Yu et al. 2018), information from various document in extractive multi-document summarization (Barzilay, Elhadad, and McKeown 2002; Nallapati, Zhai, and Zhou 2017), and events in story generation (Fan, Lewis, and Dauphin 2019; Zhu et al. 2020). An example of this task is shown in Table 1.

Early studies on sentence ordering generally use hand-crafted linguistic features to model the document structure (Lapata 2003; Barzilay and Lee 2004; Barzilay and Lapata 2008). However, these manual features are strongly domain-dependent and their definition requires heavy domain expertise. Therefore, it is difficult to apply these methods to different domains. To avoid the burden of hand-crafted features, numerous neural approaches have been proposed recently, which can be roughly categorized into two groups: The first group tends to solve this problem by predicting the pairwise sentence order with a classifier then inferring the global order (Chen, Qiu, and Huang 2016; Li and Jurafsky 2017; Prabhumoye, Salakhutdinov, and Black 2020). An obvious missing part in these approaches is the global context information, which is complementary to the local order information. Another group predicts the sentence order sequentially based on contextual sentence representations. For example, typical methods are based on pointer networks (Gong et al. 2016; Logeswaran, Lee, and Radev 2018; Cui et al. 2018; Yin et al. 2020; Kumar et al. 2020), in which sentences and paragraphs are represented by encoders and the order is predicted by a decoder sequentially. However, local coherence information is not taken into account. Indeed, both local order information and global context information are useful in this task. For the example in Table 1, it is preferable to put sentence s_4 after s_3 as the pronoun "they" in s_4 may refer to the noun "kids" in s_3. However, the word "they" also appears in s_2. Without the contextual information in s_1, it would be hard to decide the order between s_2 and s_3. On the other hand, given the information in s_1 and s_2, both s_3 and s_4 can be contextually coherent candidates for the next sentence. It is still hard to decide which should come first after s_2 without knowing the relative order $s_3 \prec s_4$.

Order	Unordered Sentences
(2)	When they arrived they saw some airplanes in the back of a truck.
(3)	The **kids** had a hard time deciding what to ride first.
(1)	The family got together to go to the fair.
(5)	***Finally*** they played the dart game.
(4)	Then **they played** some games to win prizes.

Table 1: An example of unordered sentences in a paragraph and their correct order is on the left.

Therefore, we argue that both local information and global context are crucial to sentence ordering. This raises two problems: 1) how to **capture** both the local information and the global context; 2) how to **utilize** them effectively for sentence ordering.

For the first problem, we observe that the order preference between sentences can be hinted by different types of information. For example in Table 1, when we compare sentences s_3 and s_4, the words "kids" and "they played" indicate that s_4 may immediately follow s_3. While comparing sentences s_5 and s_3, no obvious information can tell that one sentence immediately should follow another. However, the word "finally" in s_5 implies that it should appear at some position after s_3. In these examples, the clues we use to place sentences at successive positions or at some distance are different. In general, per writing rules, immediately successive sentences may present some strong intrinsic order information (whether it is causal, about time series, or others), while the same type of information may not be observed between sentences separated at a larger distance (*e.g.*, three sentences apart). In the latter case, hints on more global order information can help. For example, "finally" suggests a position towards the end, the sentence containing "go to fair" should be placed somewhere before "played some games" because of their semantic contents. These examples illustrate the need to consider useful order information at different distances or granularities. Based on these observations, we propose to model sentence order at different distances or granularities, each based on its own features.

For the second problem, we propose to model sentence ordering using graph representation. Graph representation is appealing in that different types of information can be naturally described with it. In our task, it is natural to treat each sentence in a paragraph as a node and represent their relation (the relative order) by an edge between them. With the help of graph neural networks (GNNs), the representation of each sentence can aggregate the information from its neighbors, leading to a representation that integrates both local order information and global context information. More importantly, as we want to take into account sentence relations at different granularities, GNN provides an appropriate way to fuse such information by computing sentence representations over multiple graphs. We expect that such an enhanced representation can help better predict sentence order.

More specifically, we design two phases in our framework. In the first phase, we learn multiple classifiers to judge the relative order between two sentences, each within a given distance (granularity level). In the second phase, multiple graphs (called *constraint graphs*) are built based on the order preferences. Then, we represent sentences as vectors by an encoder and employ GNNs to update these representations based on the graphs. Finally, the sentence representations are fused to predict their order.

Our main contributions are three-fold:

(1) We propose to capture sentence order information at different granularities, allowing to cover a wide spectrum of useful information;

(2) We propose a graph representation for sentence orders and design a novel GNN-based method to fuse local and global information;

(3) We conduct extensive experiments on five benchmark datasets and our model achieves better performance than the existing state-of-the-art methods. This clearly shows the superior capability of our model for sentence ordering by leveraging different types of useful information. Our ablation analysis also shows the impacts of different modules in our framework.

Related Work

Traditional methods for sentence ordering often rely on handcrafted linguistic features and domain knowledge. For example, Lapata (2003) computed transition probabilities between sentences and ordered them by a greedy algorithm. Barzilay and Lee (2004) proposed a content model which represents topics as states in a Hidden Markov Model (HMM). Then Barzilay and Lapata (2008) took into account entities and computed entity transition probabilities among sentences. Recently, neural network based methods have shown great capability in sentence ordering. We review two groups of neural approaches most relevant to ours:

Pairwise model. Pairwise models first predict pairwise sentence order, based on which the entire order is inferred. Chen, Qiu, and Huang (2016) investigated various methods to judge the order of a sentence pair, and the final ranking score of a sentence is obtained by summing up all its scores in sentence pairs. Prabhumoye, Salakhutdinov, and Black (2020) proposed to use a topological sort method to infer the entire order based on the pairwise order. While our framework also considers the relative order between each pair of sentences, it also uses a learning method based on graphs rather than a sort algorithm to infer the entire order. Furthermore, we consider sentence order within multiple distances, leading to a multi-granular view of sentence order. As we will see in our experiments, these extensions significantly improve the results of sentence ordering.

Sequence generation model. This family of models aims to compute better representations for sentences, that encode some order information. Typical methods are based on pointer network (Vinyals, Fortunato, and Jaitly 2015), which is a sequence-to-sequence model using attention as a pointer to select successively a member of the input sequence as the output. Gong et al. (2016) first applied such a model to sentence ordering task, where the encoder represents all sentences into vectors and the decoder predicts results by iteratively selecting one sentence from the input sequence. Later on, many extensions have been proposed. For example, Cui et al. (2018) refined the encoder by the self-attention mechanism, while Yin et al. (2019) modeled the co-occurrences between entities in the sentences by an entity transition graph. More recently, researchers found that using feed-forward neural network as decoder and training the whole model by a listwise loss can further improve the performance (Kumar et al. 2020). Besides, adding supplementary loss functions during the training process is also helpful (Yin et al. 2020). Sentence representations in these approaches are improved, since they can incorporate more contextual information. The relations between sentences are implicitly modeled by self-attention mechanism in these methods. Compared to these methods, our framework explicitly learns and captures the relation at multiple granularities, thus sentence representation can be further improved.

To our best knowledge, This is the first investigation that utilizes graphs to represent the relations between sentences in sentence ordering problem[1]. Our study will show that graph is a powerful and adequate formalism to represent order-related information for the task.

Methodology

Task Description

The sentence ordering task aims at ordering a set of sentences as a coherent text (paragraph). Formally, a set of n sentences with the order $\mathbf{o} = [o_1, \cdots, o_n]$ can be denoted as $\mathbf{p} = [s_{o_1}, \cdots, s_{o_n}]$. The goal is to find the correct order $\mathbf{o}^* = [o_1^*, \cdots, o_n^*]$, with which the whole paragraph have the highest coherence probability:

$$P(\mathbf{o}^*|\mathbf{p}) > P(\mathbf{o}|\mathbf{p}), \forall \mathbf{o} \in \Psi, \tag{1}$$

where $\mathbf{o}$ indicates any order of the sentences, and Ψ denotes the set of all possible orders. For example, the order of sentences in Table 1 is $[2, 3, 1, 5, 4]$, while the correct order $\mathbf{o}^*$ is $[1, 2, 3, 4, 5]$. Following the existing work (Chen, Qiu, and Huang 2016; Kumar et al. 2020; Prabhumoye, Salakhutdinov, and Black 2020), this task is framed as a ranking problem, where the model predicts a score for each sentence and the global order is determined by sorting all scores.

Model Overview

As shown in Figure 1, our framework contains two phases to capture and to use order information. In the first phase, different from existing methods (Chen, Qiu, and Huang 2016; Prabhumoye, Salakhutdinov, and Black 2020), we propose to learn multiple classifiers to predict the relative order at *different distances* between two sentences. Such relative orderings can describe sentence relations in various granularities, thus can provide more comprehensive information for the overall order prediction. This process indeed simulates a common strategy of our human beings to order sentences, *i.e.*, analyzing the relative order between sentence pairs before inferring the entire order. In the second phase, we focus on inferring the overall order based on the previously learned relations. Concretely, we build multiple graphs based on the obtained relations, and employ a GNN on each graph to incorporate the relative ordering information into the sentence representations. Finally, the representations from multiple GNNs are fused together to calculate a final score for each sentence.

Phase 1: Relative Order Prediction

Given n sentences $[s_{o_1}, \cdots, s_{o_n}]$ in any order $[o_1, \cdots, o_n]$, we can create from it a set of constraints ($\mathcal{S}_n^d$) at distance d. The set $\mathcal{S}_n^d$ describes the relative order between a pair of sentences within distance d, which can be described as:

$$\mathcal{S}_n^d = \{s_i \prec s_j \mid 0 < j - i \leq d, 1 \leq i, j \leq n\}. \tag{2}$$

For example, assuming the actual order of four sentences is $s_1 \prec s_2 \prec s_3 \prec s_4$, if $d = 2$, we have five constraints $\{s_1 \prec s_2, s_1 \prec s_3, s_2 \prec s_3, s_2 \prec s_4, s_3 \prec s_4\}$.

[1]Related work about graph representations and GNNs is provided in Appendix.

To obtain $\mathcal{S}_n^d$, we build a classifier to predict if the relative order between any two sentences s_i and s_j belongs to this set. To this end, we fine-tune the Bidirectional Encoder Representations from Transformers (BERT) pre-trained language model (Devlin et al. 2019) on each dataset with a multi-layer perceptron (MLP). The input is the sequence of tokens of sentence s_i, followed by a separator token "[SEP]", followed by the sequence of tokens of sentence s_j. Then the pooled representation of all the time steps is fed into the MLP and output a probability of $s_i \prec s_j \in \mathcal{S}_n^d$, which is denoted as $p(s_i, s_j)$. Formally, given $s_i = [w_1^i, w_2^i, \cdots, w_u^i]$ and $s_j = [w_1^j, w_2^j, \cdots, w_v^j]$ containing u and v words respectively, $p(s_i, s_j)$ is computed as:

$$\mathbf{t}_{ij} = \text{BERT}([w_1^i, \cdots, w_u^i, [\text{SEP}], w_1^j, \cdots, w_v^j]), \tag{3}$$

$$q_{ij} = \text{Sigmoid}(\text{MLP}(\mathbf{t}_{ij})), \tag{4}$$

$$p(s_i, s_j) = \begin{cases} q_{ij}, & \text{if } q_{ij} > 0.5, \\ 0, & \text{else,} \end{cases} \tag{5}$$

where $q_{ij} \in \mathbb{R}$ represents the probability of $s_i \prec s_j \in \mathcal{S}_n^d$. The classifier is trained by the binary cross entropy loss with the ground-truth label y:

$$\mathcal{L}_1(\theta_1) = -y \log(q_{ij}) - (1 - y) \log(1 - q_{ij}). \tag{6}$$

In our model, we use several distances. Therefore, the above process is repeated several times, each for a given distance, leading to multiple sets of constraints. Notice also that the predicted result for s_i and s_j is not symmetrical because we cannot infer $s_j \prec s_i \in \mathcal{S}_n^d$ from $s_i \prec s_j \notin \mathcal{S}_n^d$. Therefore, $p(s_i, s_j)$ and $p(s_j, s_i)$ should be predicted separately.

Phase 2: Sentence Ordering

The relative orderings are exploited in the second phase to infer the order of all sentences. As suggested by the existing work (Chen, Qiu, and Huang 2016; Kumar et al. 2020; Prabhumoye, Salakhutdinov, and Black 2020), the ordering process can be treated as a ranking problem, where the order can be obtained by sorting the scores of sentences. Thus the question is transformed to building better representations for sentences to incorporate the order information.

As shown in Figure 1, it is straightforward to represent the determined pairwise order as a graph, in which a sentence corresponds to a node, while the relation between sentences forms an edge. The advantage of using graphs is that the pairwise relations become explicit, thus provide together a global view of the order information. With the help of GNNs, the connection information (*i.e.*, sentence relation) contained in the edge can be incorporated into the sentence representations.

More specifically, we build a graph from $\mathcal{S}_n^d$ as follows. First, each sentence s_i is represented by BERT and treated as a node in the graph:

$$\mathbf{s}_i = \text{BERT}(s_i), i = 1, 2, \cdots, n, \tag{7}$$

where $\mathbf{s}_i \in \mathbb{R}^{768}$ is the representation of the i^{th} sentence. Different from the first phase, the input here is a single sentence. Then, an edge $s_i \rightarrow s_j$ with weight $p(s_i, s_j)$ is added

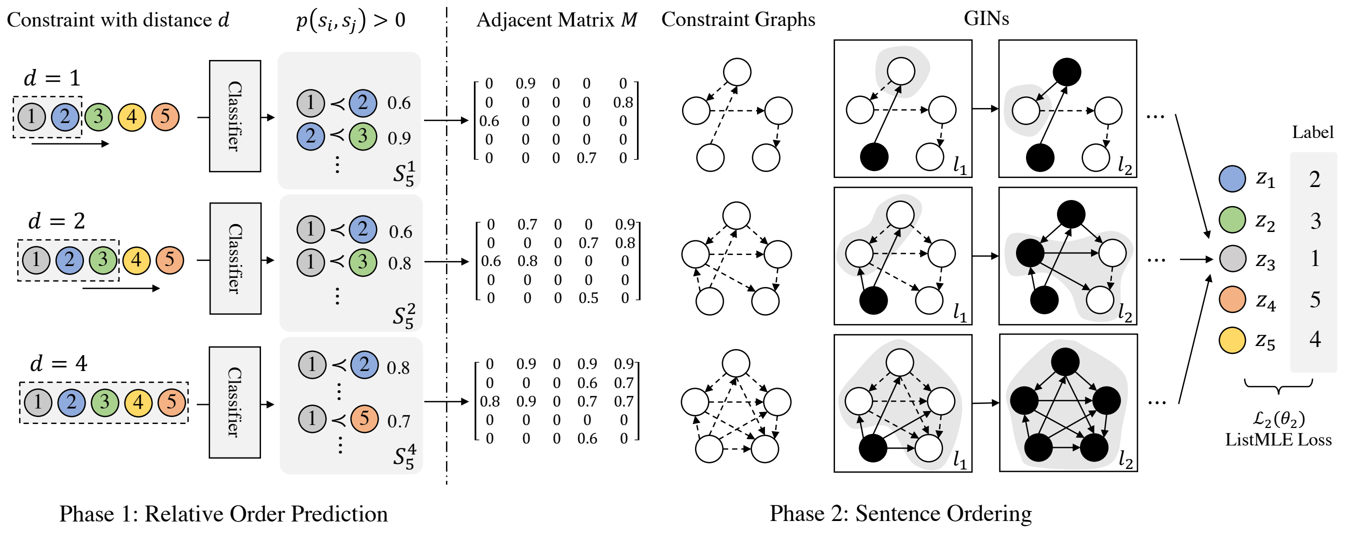

Figure 1: Model architecture. The left part shows the process of the first phase in our method. The example here involves three constraint sets. The second phase is shown on the right side where each constraint set is represented as a constraint graph and integrated into sentence representation by GINs. All sentence representations are fused together to predict the final score of sentences. The second phase of the model is optimized with ListMLE loss.

to the graph if $p(s_i, s_j) > 0$. This process forms an adjacent matrix $\mathbf{M}$ for the graph:

$$\mathbf{M}_{i,j} = p(s_i, s_j). \tag{8}$$

We call such a graph *Constraint Graph*.

We then apply Graph Isomorphism Networks (GINs) to update the representation of the node by recursively aggregating and transforming representation vectors of its neighboring nodes. We choose GIN in this work because it achieves state-of-the-art performance on many graph representation tasks (Xu et al. 2019). Alternatively, it could be replaced by any other GNN model. The node representation is initialized by the BERT representation of sentence (*i.e.*, $\mathbf{h}_i^0 = \mathbf{s}_i$) and updated in a multi-layer GIN through computation with the adjacent matrix $\mathbf{M}$. The representation of the i^{th} node in the l^{th} layer is computed as follows:

$$\mathbf{h}_i^l = \text{MLP}^{(l)}\left((1+\epsilon^{(l)}) \cdot \mathbf{h}_i^{l-1} + \sum_{j \in \mathcal{N}_i} \mathbf{h}_j^{l-1}\right), \tag{9}$$

where $\mathbf{h}_i^l \in \mathbb{R}^m$, $l \in \{1, 2, \cdots, L\}$, ϵ is a hyperparameter which controls the weights of centered node when doing aggregation in GNN, and $\mathcal{N}_i$ is the set of neighbor nodes of i^{th} node. Then, the node representations from all layers are concatenated and fused together as:

$$\mathbf{h}_i = \text{MLP}([\mathbf{h}_i^0; \mathbf{h}_i^1; \cdots; \mathbf{h}_i^L]), \tag{10}$$

where $[;]$ is concatenation operation. It is worth noting that the number of layers L should be carefully tuned according to the distance d used in the first phase. We will discuss about this in more detail later.

The above process is applied to the multiple constraint graphs of different distances. Assuming k constraint graphs, then k GINs with different numbers of layers $(L_1, \cdots, L_k)$ compute their own representations for each node. The representations from different GINs will then be fused into an augmented representation $\tilde{\mathbf{h}}_i$:

$$\tilde{\mathbf{h}}_i = \text{MLP}([\mathbf{h}_i^{(1)}; \cdots; \mathbf{h}_i^{(k)}]), \tag{11}$$

where $\tilde{\mathbf{h}}_i \in \mathbb{R}^m$ and $\mathbf{h}_i^{(k)}$ is the sentence representation in the k^{th} GIN.

Finally, we compute a score for each node (*i.e.*, sentence) through an MLP:

$$z_i = \text{MLP}(\text{ReLU}(\tilde{\mathbf{h}}_i)), \tag{12}$$

which determines the order of sentences in the paragraph.

We apply ListMLE (Xia et al. 2008) as the objective function, which is a surrogate loss to the perfect order 0-1 based loss function. Given a corpus with N paragraphs, the i^{th} paragraph with n_i unordered sentences is denoted by $\mathbf{p}_i = [s_1, \cdots, s_{n_i}]$. Assume the correct order of $\mathbf{p}_i$ is $\mathbf{o}_i^* = [o_1^*, \cdots, o_{n_i}^*]$, then ListMLE is computed as:

$$\mathcal{L}_2(\theta_2) = -\sum_{i=1}^{N} \log f(\mathbf{o}_i^* | \mathbf{p}_i), \tag{13}$$

$$f(\mathbf{o}_i^* | \mathbf{p}_i) = \prod_{j=1}^{n_i} \frac{\exp(z_{o_j^*})}{\sum_{k=j}^{n_i} \exp(z_{o_k^*})}. \tag{14}$$

With ListMLE, the model will assign the highest score to the first sentence and the lowest score to the last one.

Experiment

Datasets

Following previous work (Cui et al. 2018; Kumar et al. 2020), we conduct experiments on five public datasets. The detailed statistics of these datasets are shown in Table 2.

Datasets	Min.	Max.	Avg.	Train	Val.	Test
NeurIPS abstracts	1	15	6	2,448	409	402
AAN abstracts	1	20	5	8,569	962	2,626
NSF abstracts	2	40	8.9	96,017	10,185	21,573
SIND captions	5	5	5	40,155	4,990	5,055
ROCStory	5	5	5	78,529	9,816	9,816

Table 2: The statistics of all datasets.

- **NeurIPS/AAN/NSF abstracts** (Logeswaran, Lee, and Radev 2018). These datasets consist of abstracts from NeurIPS, ACL and NSF research award papers. The data are split into training, validation, and test set according to the publication year.
- **SIND captions** (Huang et al. 2016). This is a visual story dataset. Each story contains five sentences.
- **ROCStory** (Mostafazadeh et al. 2016). It is a commonsense story dataset. Each story comprises five sentences. Following (Wang and Wan 2019), we make an 8:1:1 random split on the dataset to get the training, validation, and test set.

Baseline Models

We compare our model with the following methods:
Traditional methods: Entity Grid (Barzilay and Lapata 2008); Window Network (Li and Hovy 2014). These methods use hand-crafted features or neural networks to capture the coherence of text.
Pairwise models: Seq2seq (Li and Jurafsky 2017); B-TSort (Prabhumoye, Salakhutdinov, and Black 2020)[2]. These methods predict the relative order between sentence pairs, then infer the entire order.
Sequence generation models: CNN/LSTM+PtrNet (Gong et al. 2016); Variant-LSTM+PtrNet (Logeswaran, Lee, and Radev 2018); ATTOrderNet (Cui et al. 2018); HierarchicalATTNet (Wang and Wan 2019); SE-Graph (Yin et al. 2019); ATTOrderNet+TwoLoss (Yin et al. 2020); RankTxNet+ListMLE (Kumar et al. 2020). These architectures adopt CNN/RNN based approaches to obtain the representation for the input sentences and employ the pointer network as the decoder to predict order. The last four methods are extended models based on ATTOrderNet.

Evaluation Metrics

We use Kendall's τ and Perfect Match Ratio (PMR) as metrics, both being commonly used in previous work (Gong et al. 2016; Cui et al. 2018; Logeswaran, Lee, and Radev 2018; Yin et al. 2020; Kumar et al. 2020).
Kendall's Tau (τ): it is one of the most frequently used metrics for the automatic evaluation of text coherence (Lapata 2006, 2003; Logeswaran, Lee, and Radev 2018). It quantifies the distance between the predicted order and the correct order in terms of the number of the inversions. $\tau = 1 - 2I/\binom{n}{2}$, where I is the number of pairs in the predicted order with incorrect relative order, and n is number of sentences in the paragraph. This value ranges from -1 (the worst) to 1 (the best).
PMR: it calculates the percentage of samples for which the entire order of the sequence is correctly predicted (Chen, Qiu, and Huang 2016). PMR $= \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}\{\hat{\mathbf{o}}_i = \mathbf{o}_i^*\}$, where $\hat{\mathbf{o}}_i$ and $\mathbf{o}_i^*$ are the predicted and correct orders for the i^{th} paragraph. N is the number of samples in the dataset.

Implementation Details

All models are implemented with PyTorch (Paszke et al. 2019) and trained on a TITAN V GPU. In the first phase, we employ BERT uncased model (Wolf et al. 2019) with an MLP to predict constraints. The batch size and learning rate is set as 50 and 5e-5 respectively for all datasets. In the second phase, sentences are also represented by an uncased BERT model and a dropout layer with rate 0.1 is applied over the representations. Three GINs[3] with the number of layers $L = \{2, 3, 5\}$ are used for the three corresponding graphs ($k = 3$) obtained in the first phase (denoted as g_1, g_2 and g_3 respectively). The hidden size of all GIN layers is 512 ($m = 512$). A ReLU activation function is added between each layer. ϵ is tuned in $\{0, 0.1, 0.5\}$ and set as 0 according to experimental results on validation set. Batch normalization is applied to avoid overfitting. The batch size is set as 128 for all datasets while the learning rate is set as 1e-4, 5e-4, 6e-4, 4e-4 and 3e-4 for NeurIPS, AAN, NSF, SIND and ROCStory dataset. The maximum number of words in sentences is set as 50 on all datasets, which means sentences containing more than 50 words are truncated while those having less than 50 words are padded. All paragraphs in the datasets are randomly shuffled. To make a fair comparison, all models are tested on the same test set without any other preprocessing operations. The models in both two phases are optimized with AdamW (Loshchilov and Hutter 2019).

Relationship between distance d and the number of layers L

In our experiments, we observe that the hyperparameter L in the second phase should be selected according to d in the first phase. The larger the distance d is, the fewer the GIN layers are required to obtain good results. This trend can be explained by the coverage of the entire set of sentences in a paragraph as follows.

As illustrated in Figure 1, assuming that there are five sentences to be ordered, we can see when $d = 1$ (the top one), the classifier in the first phase only predict the relative order between two successive sentences. Therefore, in the corresponding constraint graph, the model need to stack four layers (hops) to connect the first sentence to the last sentence. Similarly, if $d = 2$, the model need two layers to build the

[2]Note that the results of B-TSort are slightly worse than those reported in the original paper, because the provided source code does not shuffle the sentence order on test set when applying topological sort algorithm, which artificially improves the results.

[3]We use the implementation at https://github.com/chao1224/BioChemGNN_Dense.

	NeurIPS		AAN		NSF		SIND		ROCStory	
	τ	PMR	τ	PMR	τ	PMR	τ	PMR	τ	PMR
Entity Grid$^{\triangle}$	0.09	-	0.10	-	-	-	-	-	-	-
Window Network$^{\triangle}$	0.59	-	0.65	-	0.28	-	-	-	-	-
Seq2seq$^{\triangle}$	0.27	-	0.40	-	0.10	-	0.19	12.50	0.34	17.93
CNN + PtrNet$^{\heartsuit}$	0.6976†	19.36†	0.6700†	28.75†	0.4460†	5.95†	0.4197†	9.50†	0.6538†	27.06†
LSTM + PtrNet$^{\heartsuit}$	0.7373†	20.95†	0.7394†	38.30†	0.5460†	10.68†	0.4833†	12.96†	0.6787†	28.24†
Variant-LSTM + PtrNet$^{\heartsuit}$	0.7258†	22.02†	0.7521†	40.67†	0.5544†	10.97†	0.4878†	13.57†	0.6852†	30.28†
ATTOrderNet$^{\heartsuit}$	0.7466†	21.22†	0.7493†	40.71†	0.5494†	10.48†	0.4823†	12.27†	0.7011†	34.32†
HierarchicalATTNet$^{\diamondsuit}$	0.7008†	19.63†	0.6956†	30.29†	0.5073†	8.12†	0.4814†	11.01†	0.6873†	31.73†
SE-Graph$^{\diamondsuit}$	0.7370†	24.63‡	0.7616†	41.63†	0.5602†	10.94†	0.4804†	12.58†	0.6852†	31.36†
ATTOrderNet + TwoLoss$^{\diamondsuit}$	0.7357†	23.63‡	0.7531†	41.59†	0.4918†	9.39†	0.4952†	14.09†	0.7302†	40.24†
RankTxNet + ListMLE$^{\heartsuit}$	0.7316†	20.40†	0.7579†	36.89†	0.4899†	6.81†	0.5560†	13.93†	0.7215†	28.30†
B-TSort$^{\diamondsuit}$	0.7884	30.59	0.8064‡	48.08	0.4813†	7.88†	0.5632†	17.35‡	0.7941†	48.06‡
Our	**0.8029**	**32.84**	**0.8236**	**49.81**	**0.6082**	**13.67**	**0.5856**	**19.07**	**0.8122**	**49.52**

Table 3: Results on five benchmark datasets. $\triangle$ indicates previously reported scores. Models with $\diamondsuit$ are implemented with the provided source code while those with $\heartsuit$ are implemented by ourselves. The numbers here are our runs of the model. Hence, they are slightly different from the numbers reported in the original paper. † and ‡ denote significant improvements with our method in t-test with $p < 0.01$ and $p < 0.05$ respectively.

connection; and if $d = 4$, any two sentences in the graph are connected, thus only one layer is enough.

Therefore, to connect any sentence pair within a set of n_i sentences for paragraph i, we have the empirical formula:

$$L_i \geq \lceil (n_i - 1)/d_i \rceil, \tag{15}$$

where $\lceil \cdot \rceil$ is the ceiling function. In practice, we have to fix d in order to train classifiers, and we can only implement models with a fixed L. Therefore, in our experiments, we use GNNs with $L = \{2, 3, 5\}$ for all datasets and compute the corresponding distance d as:

$$d_i = \lceil \big(\max(\{n_i\}_{i=1}^{N}) - 1 \big) / (L_i - 1) \rceil. \tag{16}$$

As a result, the distance d is set as {14, 7, 4}, {19, 10, 5}, {39, 20, 10}, {4, 2, 1}, and {4, 2, 1} for NeurIPS, AAN, NSF, SIND, and ROCStory dataset respectively.

To validate the Equation (15), we design an experiment that use the ground-truth constraint graphs as input to test how many layers are necessary to obtain perfect results. The results on SIND validation sets ($n = 5$ for all samples) confirm our empirical analysis. The details of this experiment is presented in Appendix.

Sentence Ordering Results

Table 3 shows the sentence ordering results on all datasets[4]. Our method significantly outperforms all baseline models on both evaluation metrics.

The improvement is generalized across different topics and sizes of data. On research paper abstracts datasets, our model outperforms the previous best baseline method by about 1.5% τ score and 1.7% PMR score on NeurIPS and AAN datasets, while the improvement is more than 2.5% on larger datasets, *i.e.*, NSF. As for two story datasets, our method outperforms the previous state-of-the-art model by around 1.8% τ score and 1.3% PMR score. Interestingly, our method achieves 49.52% PMR score on ROCStory, meaning that about half of the stories in the test set can be ordered completely right. The performance clearly demonstrates the effectiveness and wide applicability of our proposed method.

B-TSort is a recently proposed pairwise method, which predicts the entire order merely based on the relative order of two sentences at only one distance (as $d = n - 1$ in our framework). Compared to it, our method can achieve better performance because multiple sentence relations are considered. Besides, in the second phase, we use neural networks rather than a sorting algorithm (*e.g.*, a topological sort) to infer the entire order, which can effectively fuse the order information from multiple constraint graphs.

According to the results, being equipped with BERT may bring improvements (*e.g.*, RankTxNet+ListMLE on SIND and ROCStory dataset). However, compared with the last two baselines that also use BERT as encoder, our framework still yields better results. This indicates that the better performance we obtained is not merely due to BERT embeddings but also to the way we create representations on top of it.

Ablation Study

We investigate the impact of different modules in our model by an ablation study. The experiments are performed on AAN abstracts and SIND captions dataset. From the results shown in Table 4, we can observe:

(1) The effect of each graph varies on different datasets. Compared with using the graph g_1 (*i.e.*, the graph with largest distance), our model with g_3 (smaller distance) performs better on AAN but worse on SIND. However, using

[4]The average accuracy of each classifier in the first phase on all datasets is around 80%, and the detailed results are reported in Appendix.

	AAN		SIND	
	τ	PMR	τ	PMR
Our ($g_1 + g_2 + g_3$)	0.8236	49.81	0.5856	19.07
Only g_1	0.8078	47.79	0.5708	17.23
Only g_2	0.8170	48.82	0.5377	17.17
Only g_3	0.8116	48.06	0.5276	18.32
$g_1 + g_2$	0.8217	49.31	0.5744	17.74
$g_1 + g_3$	0.8196	49.54	0.5837	18.34
$g_2 + g_3$	0.8208	49.58	0.5539	**19.56**
with Fine-tuning BERT	**0.8245**	49.54	**0.5867**	18.56
with RoBERTa	0.8240	49.54	0.5761	17.80
with ALBERT	0.8228	**49.85**	0.5820	17.27
with GCN	0.8206	49.24	0.5818	17.71

Table 4: Ablation results on AAN and SIND dataset.

more graphs always lead to better results. This empirically validates our assumption that multiple graphs can indeed capture various types of information between sentences that are useful for sentence ordering.

(2) In our model, the BERT encoder used in the second phase is not fine-tuned during the training process. We also test the performance with fine-tuning BERT. The result shows that fine-tuning BERT can bring further improvements in terms of τ. Nevertheless, the number of parameters becomes significantly larger than without fine-tuning (117M vs. 7M), and the training time is much longer, *e.g.*, about 2.41 times more (917s vs. 381s per epoch) on SIND dataset. This suggests that fine-tuning BERT is a good strategy only if we have the necessary computation power.

(3) As many variants or extensions of BERT have been proposed recently, we also test our method with RoBERTa (Liu et al. 2019) and ALBERT (Lan et al. 2020). However, according to our experiments, these two models cannot perform well in the first phase. The potential reason is that they both remove the Next Sentence Prediction objective in the pre-training stage, which is very important for the constraint prediction of our framework. When applying them in the second phase, the final results show some slight improvements on AAN dataset. Through this experiment, we see that our method can work with other advanced pre-trained language models.

(4) We mentioned that GIN could be replaced by any GNN model. In this experiment, we replace it by graph convolutional networks (GCNs) (Kipf and Welling 2017) which are also widely used in many tasks. The results with GCNs are slightly worse compared to the model with GINs, but are still competitive. This shows that other GNN models could also be used on our graphs.

Further Analysis

As discussed by (Gong et al. 2016; Chen, Qiu, and Huang 2016; Cui et al. 2018; Kumar et al. 2020), the first and last sentence should be paid more attention to due to their crucial positions in a paragraph. We provide the accuracy for these sentences on AAN and SIND datasets in Table 5. Our method gives clearly better results than all baselines. In particular, on SIND dataset, our method achieves the absolute improvement of 2.08% on the last sentence prediction compared with the previous best method.

	AAN		SIND	
	First	Last	First	Last
Our	**91.47**	**80.35**	**79.80**	**60.44**
CNN+PtrNet	77.60	67.24	68.23	47.63
LSTM+PtrNet	84.63	72.63	75.01	53.79
Variant-LSTM+PtrNet	85.63	73.28	75.05	53.87
ATTOrderNet	84.82	73.09	74.52	52.46
HierarchicalATTNet	80.35	68.71	75.07	52.53
SE-Graph	87.00	73.59	74.58	53.73
ATTOrderNet + TwoLoss	86.01	73.93	75.33	53.99
RankTxNet + ListMLE	88.11	74.27	79.51	57.25
B-Tsort	89.53	79.78	78.06	58.36

Table 5: Results for the first and last sentence prediction on AAN and SIND dataset.

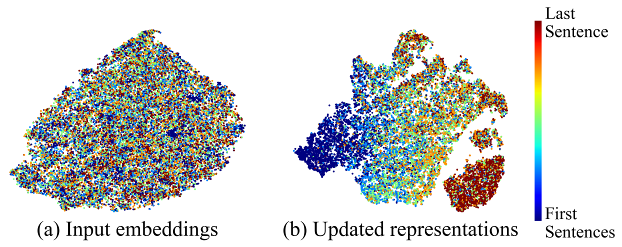

Figure 2: t-SNE embeddings of sentence representations for SIND dataset on sentence ordering task. Colors correspond to the position of the sentences in the original paragraph.

Following previous work (Logeswaran, Lee, and Radev 2018; Kumar et al. 2020), we use t-SNE embeddings to visualize the effect of training on the sentence representations for SIND dataset in Figure 2. We can clearly see that the updated representations contain more order information than the original BERT embeddings. This is another demonstration that our approach can effectively capture order information into sentence representations.

Conclusion

In this paper, we proposed a novel model for sentence ordering which takes into account relative order information at multiple granularities. This is based on the intuition that the order at different distances may rely on different types of information, and all such order information is useful for sentence ordering. We first classified sentence order within different distances and formed multiple constraint graphs from it. Then we employed GINs to incorporate the order information into sentence representations, which are finally used to predict the sentence order. Our model achieved state-of-the-art performance on five benchmark datasets. This study paves the way for a new research direction on sentence ordering by leveraging different types of information in the form of constraint graphs.

Acknowledgments

We thank Pan Du for the insightful discussions and the anonymous reviewers for their feedback. This work was supported by National Natural Science Foundation of China No. 61872370 and No. 61832017, Beijing Outstanding Young Scientist Program NO. BJJWZYJH012019100020098, and Shandong Provincial Natural Science Foundation under Grant ZR2019ZD06.

References

Barzilay, R.; Elhadad, N.; and McKeown, K. R. 2002. Inferring Strategies for Sentence Ordering in Multidocument News Summarization. *J. Artif. Intell. Res.* 17: 35–55.

Barzilay, R.; and Lapata, M. 2008. Modeling Local Coherence: An Entity-Based Approach. *Comput. Linguistics* 34(1): 1–34.

Barzilay, R.; and Lee, L. 2004. Catching the Drift: Probabilistic Content Models, with Applications to Generation and Summarization. In Hirschberg, J.; Dumais, S. T.; Marcu, D.; and Roukos, S., eds., *HLT-NAACL 2004, Boston, Massachusetts, USA, May 2-7, 2004*, 113–120. The Association for Computational Linguistics.

Chen, X.; Qiu, X.; and Huang, X. 2016. Neural Sentence Ordering. *CoRR* abs/1607.06952.

Cui, B.; Li, Y.; Chen, M.; and Zhang, Z. 2018. Deep Attentive Sentence Ordering Network. In Riloff, E.; Chiang, D.; Hockenmaier, J.; and Tsujii, J., eds., *EMNLP 2018, Brussels, Belgium, October 31 - November 4, 2018*, 4340–4349. Association for Computational Linguistics.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Burstein, J.; Doran, C.; and Solorio, T., eds., *NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, 4171–4186. Association for Computational Linguistics.

Dwivedi, V. P.; Joshi, C. K.; Laurent, T.; Bengio, Y.; and Bresson, X. 2020. Benchmarking Graph Neural Networks. *CoRR* abs/2003.00982.

Fan, A.; Lewis, M.; and Dauphin, Y. N. 2019. Strategies for Structuring Story Generation. In Korhonen, A.; Traum, D. R.; and Màrquez, L., eds., *ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, 2650–2660. Association for Computational Linguistics.

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural Message Passing for Quantum Chemistry. In Precup, D.; and Teh, Y. W., eds., *ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, 1263–1272. PMLR.

Gong, J.; Chen, X.; Qiu, X.; and Huang, X. 2016. End-to-End Neural Sentence Ordering Using Pointer Network. *CoRR* abs/1611.04953.

Huang, T. K.; Ferraro, F.; Mostafazadeh, N.; Misra, I.; Agrawal, A.; Devlin, J.; Girshick, R. B.; He, X.; Kohli, P.; Batra, D.; Zitnick, C. L.; Parikh, D.; Vanderwende, L.; Galley, M.; and Mitchell, M. 2016. Visual Storytelling. In Knight, K.; Nenkova, A.; and Rambow, O., eds., *NAACL HLT 2016, San Diego California, USA, June 12-17, 2016*, 1233–1239. The Association for Computational Linguistics.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Konstas, I.; and Lapata, M. 2012. Concept-to-text Generation via Discriminative Reranking. In *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, July 8-14, 2012, Jeju Island, Korea - Volume 1: Long Papers*, 369–378. The Association for Computer Linguistics.

Konstas, I.; and Lapata, M. 2013. Inducing Document Plans for Concept-to-Text Generation. In *EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, 1503–1514. ACL.

Kumar, P.; Brahma, D.; Karnick, H.; and Rai, P. 2020. Deep Attentive Ranking Networks for Learning to Order Sentences. In *AAAI 2020, IAAI 2020, EAAI 2020, New York, NY, USA, February 7-12, 2020*, 8115–8122. AAAI Press.

Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; and Soricut, R. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Lapata, M. 2003. Probabilistic Text Structuring: Experiments with Sentence Ordering. In Hinrichs, E. W.; and Roth, D., eds., *ACL 2003, 7-12 July 2003, Sapporo Convention Center, Sapporo, Japan*, 545–552. ACL.

Lapata, M. 2006. Automatic Evaluation of Information Ordering: Kendall's Tau. *Comput. Linguistics* 32(4): 471–484.

Li, J.; and Hovy, E. H. 2014. A Model of Coherence Based on Distributed Sentence Representation. In Moschitti, A.; Pang, B.; and Daelemans, W., eds., *EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, 2039–2048. ACL.

Li, J.; and Jurafsky, D. 2017. Neural Net Models of Open-domain Discourse Coherence. In Palmer, M.; Hwa, R.; and Riedel, S., eds., *EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, 198–209. Association for Computational Linguistics.

Liu, S.; Alnammi, M.; Ericksen, S. S.; Voter, A. F.; Ananiev, G. E.; Keck, J. L.; Hoffmann, F. M.; Wildman, S. A.; and Gitter, A. 2018. Practical model selection for prospective virtual screening. *Journal of chemical information and modeling* 59(1): 282–293.

Liu, S.; Demirel, M. F.; and Liang, Y. 2019. N-Gram Graph: Simple Unsupervised Representation for Graphs, with Applications to Molecules. In Wallach, H. M.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E. B.; and Garnett, R., eds., *NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, 8464–8476.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR* abs/1907.11692.

Logeswaran, L.; Lee, H.; and Radev, D. R. 2018. Sentence Ordering and Coherence Modeling using Recurrent Neural Networks. In McIlraith, S. A.; and Weinberger, K. Q., eds., *AAAI 2018, IAAI 2018, EAAI 2018, New Orleans, Louisiana, USA, February 2-7, 2018*, 5285–5292. AAAI Press.

Loshchilov, I.; and Hutter, F. 2019. Decoupled Weight Decay Regularization. In *ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Mostafazadeh, N.; Chambers, N.; He, X.; Parikh, D.; Batra, D.; Vanderwende, L.; Kohli, P.; and Allen, J. F. 2016. A Corpus and Evaluation Framework for Deeper Understanding of Commonsense Stories. *CoRR* abs/1604.01696.

Nallapati, R.; Zhai, F.; and Zhou, B. 2017. SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents. In Singh, S. P.; and Markovitch, S., eds., *AAAI 2017, February 4-9, 2017, San Francisco, California, USA*, 3075–3081. AAAI Press.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Köpf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Wallach, H. M.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E. B.; and Garnett, R., eds., *NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, 8024–8035.

Prabhumoye, S.; Salakhutdinov, R.; and Black, A. W. 2020. Topological Sort for Sentence Ordering. In Jurafsky, D.; Chai, J.; Schluter, N.; and Tetreault, J. R., eds., *ACL 2020, Online, July 5-10, 2020*, 2783–2792. Association for Computational Linguistics.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2017. Graph Attention Networks. *CoRR* abs/1710.10903.

Vinyals, O.; Fortunato, M.; and Jaitly, N. 2015. Pointer Networks. In Cortes, C.; Lawrence, N. D.; Lee, D. D.; Sugiyama, M.; and Garnett, R., eds., *NeurIPS 2015, December 7-12, 2015, Montreal, Quebec, Canada*, 2692–2700.

Wang, T.; and Wan, X. 2019. Hierarchical Attention Networks for Sentence Ordering. In *AAAI 2019, IAAI 2019, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, 7184–7191. AAAI Press.

Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; and Brew, J. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *CoRR* abs/1910.03771.

Wu, Z.; Ramsundar, B.; Feinberg, E. N.; Gomes, J.; Geniesse, C.; Pappu, A. S.; Leswing, K.; and Pande, V. S. 2018. MoleculeNet: A Benchmark for Molecular Machine Learning. *Chemical science* 9(2): 513–530.

Xia, F.; Liu, T.; Wang, J.; Zhang, W.; and Li, H. 2008. Listwise approach to learning to rank: theory and algorithm. In Cohen, W. W.; McCallum, A.; and Roweis, S. T., eds., *ICML 2008, Helsinki, Finland, June 5-9, 2008*, volume 307 of *ACM International Conference Proceeding Series*, 1192–1199. ACM.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019. How Powerful are Graph Neural Networks? In *ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Yin, Y.; Meng, F.; Su, J.; Ge, Y.; Song, L.; Zhou, J.; and Luo, J. 2020. Enhancing Pointer Network for Sentence Ordering with Pairwise Ordering Predictions. In *AAAI 2020, IAAI 2020, EAAI 2020, New York, NY, USA, February 7-12, 2020*, 9482–9489. AAAI Press.

Yin, Y.; Song, L.; Su, J.; Zeng, J.; Zhou, C.; and Luo, J. 2019. Graph-based Neural Sentence Ordering. In Kraus, S., ed., *IJCAI 2019, Macao, China, August 10-16, 2019*, 5387–5393. ijcai.org.

Yu, A. W.; Dohan, D.; Luong, M.; Zhao, R.; Chen, K.; Norouzi, M.; and Le, Q. V. 2018. QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. In *ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Zhu, Y.; Song, R.; Dou, Z.; Nie, J.; and Zhou, J. 2020. ScriptWriter: Narrative-Guided Script Generation. In Jurafsky, D.; Chai, J.; Schluter, N.; and Tetreault, J. R., eds., *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, 8647–8657. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.acl-main.765/.

Dataset	d	L	Acc. of 0	Acc. of 1	Overall Acc.
NeurIPS	14	2	0.8068	0.7599	0.7834
	7	3	0.8358	0.5143	0.7233
	4	5	0.6711	0.6791	0.6726
AAN	19	2	0.8822	0.8761	0.8791
	10	3	0.8772	0.8850	0.8811
	5	5	0.8434	0.8625	0.8523
NSF	39	2	0.7625	0.7133	0.7379
	20	3	0.7564	0.7634	0.7599
	10	5	0.7188	0.7243	0.7211
SIND	4	2	0.8068	0.7599	0.7834
	2	3	0.8358	0.5143	0.7233
	1	5	0.6248	0.7439	0.6486
ROCStory	4	2	0.9008	0.9048	0.9028
	2	3	0.8542	0.7693	0.8245
	1	5	0.9304	0.5647	0.8572

Table 6: Accuracy of constraints prediction on all datasets. We report the accuracy of label zero (Acc. of 0), the accuracy of label one (Acc. of 1) and the overall accuracy respectively.

Appendix

Related work about graph representation and graph neural network

Graph representation is appealing in the sense that many things can be naturally described with it: as long as the feature has intra-relations, *e.g.*, pixels in images have spatial connection, words in the sentences have temporal correlation, atoms in molecules are connected following the graph topology, and entities in the knowledge graph form a graph in a straightforward way.

The graph neural networks are proposed accordingly for better representation learning, and the core idea is that, for each node, we use message passing operation to exchange information within the k-hop neighborhood around it. Successful applications include image classification (Dwivedi et al. 2020), social network analysis (Kipf and Welling 2017; Velickovic et al. 2017), and molecule property prediction (Wu et al. 2018; Liu et al. 2018; Liu, Demirel, and Liang 2019; Gilmer et al. 2017; Xu et al. 2019).

Accuracy of our method in the first phase

In the first phase, our method learns a classifier to predict the constraint set with given distance. We report the accuracy of each classifier on the test set in Table 6. As the labels are imbalanced (samples with label zero are more than those with label one), we report the accuracy of each label and the overall accuracy respectively. We can observe that the accuracy is highly related to the dataset and influences the performance in the second phase directly. For example, the accuracy on AAN and ROCStory is higher than that on other datasets. Correspondingly, the performance in the second phase (as shown in Table 3) on these two datasets is also better. Moreover, although the accuracy on predicting some of the constraints is not very high, such as $d = 4$ on NeurIPS and $d = 1$ on SIND, the corresponding graph can

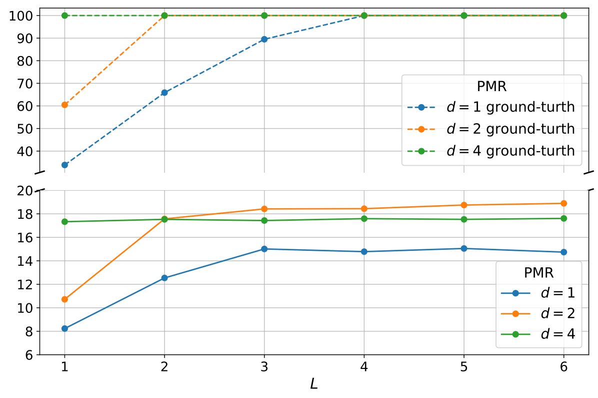

Figure 3: The PMR results on SIND datasets obtained by our method with different distances and different number of layers. The two parts use respectively ground-truth and predicted sentence orders.

still contribute to the final performance on sentence ordering. We believe that any prediction better than random could be useful. Therefore, using multiple graphs is an effective way to improve the sentence ordering performance.

Validation for relationship between distance d and the number of layers L

To validate the Equation (15), we design an experiment that uses the ground-truth constraint graphs as input to test how many layers are necessary to obtain perfect results. The results on SIND validation sets ($n = 5$ for all samples) confirm our empirical analysis. We observe that the results are consistent with our empirical formula that $d = 1$ requires at least four GIN layers ($L \geq 4$) to achieve perfect results, while $d = 4$ requires only one layer ($L \geq 1$). Therefore, on this dataset, the necessary setting of L is $\{1, 2, 4\}$. As a reference, the results with predicted constraints are illustrated in the lower part of the figure. According to the results, we find that adding one more layer may slightly improve the performance, thus we empercally set L as $\{2, 3, 5\}$ on all datasets.