Improving Graph-based Sentence Ordering with Iteratively Predicted Pairwise Orderings

Shaopeng Lai[1], **Ante Wang**[1], **Fandong Meng**[2], **Jie Zhou**[2], **Yubin Ge**[3], **Jiali Zeng**[4], **Junfeng Yao**[1], **Degen Huang**[5] and **Jinsong Su**[1,6*]

[1]Xiamen University, China [2]Pattern Recognition Center, WeChat AI, Tencent Inc, China
[3]University of Illinois at Urbana-Champaign, USA [4]Tencent Cloud Xiaowei, China
[5]Dalian University of Technology, China [6]Pengcheng Laboratory, China
splai@stu.xmu.edu.cn, fandongmeng@tencent.com, jssu@xmu.edu.cn

Abstract

Dominant sentence ordering models can be classified into pairwise ordering models and set-to-sequence models. However, there is little attempt to combine these two types of models, which inituitively possess complementary advantages. In this paper, we propose a novel sentence ordering framework which introduces two classifiers to make better use of pairwise orderings for graph-based sentence ordering (Yin et al., 2019, 2021). Specially, given an initial sentence-entity graph, we first introduce a graph-based classifier to predict pairwise orderings between linked sentences. Then, in an iterative manner, based on the graph updated by previously predicted high-confident pairwise orderings, another classifier is used to predict the remaining uncertain pairwise orderings. At last, we adapt a GRN-based sentence ordering model (Yin et al., 2019, 2021) on the basis of final graph. Experiments on five commonly-used datasets demonstrate the effectiveness and generality of our model. Particularly, when equipped with BERT (Devlin et al., 2019) and FHDecoder (Yin et al., 2020), our model achieves state-of-the-art performance. Our code is available at https://github.com/DeepLearnXMU/IRSEG.

1 Introduction

With the rapid development and increasing applications of natural language processing (NLP), modeling text coherence has become a significant task, since it can provide beneficial information for understanding, evaluating and generating multi-sentence texts. As an important subtask, sentence ordering aims at recovering unordered sentences back to naturally coherent paragraphs. It is required to deal with logic and syntactic consistency, and has increasingly attracted attention due to its wide applications on several tasks such as text generation (Konstas and Lapata, 2012; Holtzman et al., 2018) and extractive summarization (Barzilay et al., 2002; Nallapati et al., 2012).

Recently, inspired by the great success of deep learning in other NLP tasks, researchers have resorted to neural sentence ordering models, which can be classified into: *pairwise ordering models* (Chen et al., 2016; Agrawal et al., 2016; Li and Jurafsky, 2017; Moon et al., 2019; Kumar et al., 2020; Prabhumoye et al., 2020; Zhu et al., 2021) and *set-to-sequence models* (Gong et al., 2016; Nguyen and Joty, 2017; Logeswaran et al., 2018; Mohiuddin et al., 2018; Cui et al., 2018; Yin et al., 2019; Oh et al., 2019; Yin et al., 2020; Cui et al., 2020; Yin et al., 2021). Generally, the former predicts the relative orderings between pairwise sentences, which are then leveraged to produce the final ordered sentence sequence. Its advantage lies in the lightweight pairwise ordering predictions, since the predictions only depend on the semantic representations of involved sentences. By contrast, the latter is mainly based on an encoder-decoder framework, where an encoder is first used to learn contexualized sentence representations by considering other sentences, and then a decoder, such as pointer network (Vinyals et al., 2015a), outputs ordered sentences.

Overall, these two kinds of models have their own strengths, which are complementary to each other. To combine their advantages, Yin et al. (2020) propose FHDecoder that is equipped with three pairwise ordering prediction modules to enhance the pointer network decoder. Along this line, Cui et al. (2020) introduce BERT to exploit the deep semantic connection and relative orderings between sentences and achieve SOTA performance when equipped with FHDecoder. However, there still exist two drawbacks: 1) their pairwise ordering predictions only depend on involved sentence pairs, without considering other sentences in the same set; 2) their one-pass pairwise ordering predictions are relatively rough, ignoring distinct difficulties in

* Corresponding author

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 2407–2417
November 7–11, 2021.

predicting different sentence pairs. Therefore, we believe that the potential of pairwise orderings in neural sentence ordering models has not been fully exploited.

In this paper, we propose a novel iterative pairwise ordering prediction framework which introduces two classifiers to make better use of pairwise orderings for graph-based sentence ordering (Yin et al., 2019, 2021). As an extension of *Sentence-Enity Graph Recurrent Network* (SE-GRN) (Yin et al., 2019, 2021), our framework enriches the graph representation with iteratively predicted orderings between pairwise sentences, which further benefits the subsequent generation of ordered sentences. The basic intuitions behind our work are two-fold. First, learning contextual sentence representations is helpful to predict pairwise orderings. Second, difficulties of predicting ordering vary with respect to different sentence pairs. Thus, it is more reasonable to first predict the orderings of pairwise sentences easily to be predicted, and then leverage these predicted orderings to refine the predictions for other pairwise sentences.

Concretely, we propose two graph-based classifiers to iteratively conduct ordering predictions for pairwise sentences. The first classifier takes the sentence-entity graph (SE-Graph) (Yin et al., 2019, 2021) as input and yields relative orderings of linked sentences via corresponding probabilities. Next, in an iterative manner, the second classifier enriches the previous graph representation by converting high-value probabilities into the weights of the corresponding edges, and then reconduct graph encoding to predict orderings for the other pairwise sentences. Based on the final weighted graph representation, we adapt SE-GRN to construct a graph-based sentence ordering model, of which the decoder is also a pointer network.

To the best of our knowledge, our work is the first to exploit pairwise orderings to enhance the graph encoding for graph-based set-to-squence sentence ordering. To investigate the effectiveness of our framework, we conduct extensive experiments on several commonly-used datasets. Experimental results and in-depth analyses show that our model enhanced with some proposed technologies (Devlin et al., 2019; Yin et al., 2020) achieves the state-of-the-art performance.

2 Related Work

Early studies mainly focused on exploring human-designed features for sentence ordering (Lapata, 2003; Barzilay and Lee, 2004; Barzilay and Lapata, 2005, 2008; Elsner and Charniak, 2011; Guinaudeau and Strube, 2013). Recently, neural network based sentence ordering models have become dominant , consisting of the following two kinds of models:

1) **Pairwise models**. Generally, they first predict the pairwise orderings between sentences and then use them to produce the final sentence order via ranking algorithms (Chen et al., 2016; Agrawal et al., 2016; Li and Jurafsky, 2017; Kumar et al., 2020; Prabhumoye et al., 2020; Zhu et al., 2021). For example, Chen et al. (2016) first framed sentence ordering as a ranking task conditioned on pairwise scores. Agrawal et al. (2016) conducted the same experiments as (Chen et al., 2016) in the task of image caption storytelling. Similarly, Li and Jurafsky (2017) investigated the effectiveness of discriminative and generative models on ordering pairs of sentences in small domains. Moon et al. (2019) proposed a unified model that incorporates sentence grammar, pairwise coherence and global coherence into a common neural framework. Recently, Prabhumoye et al. (2020) and Zhu et al. (2021) employed ranking techniques to find the right order of the sentences under the constraint of the predicted pairwise sentence ordering;

2) **Set-to-sequence Models**. Basically, these models are based on an encoder-decoder framework, where the encoder is used to obtain sentence representations and then the decoder produces ordered sentences progressively. Among them, both Gong et al. (2016) and Logeswaran et al. (2018) explored RNN based encoder, while both Nguyen and Joty (2017) and Mohiuddin et al. (2018) employed neural entity grid models as encoders. Typically, Cui et al. (2018) proposed ATTOrderNet that uses self-attention mechanism to learn sentence representations. Inspired by the successful applications of graph neural network (GNN) in many NLP tasks (Song et al., 2018; Xue et al., 2019; Song et al., 2019, 2020), Yin et al. (2019, 2021) represented input sentences with a unified SE-Graph and then applied GRN to learn sentence representations. Very recently, we notice that Chowdhury et al. (2021) proposes a BART-based sentence ordering model. Please note that our porposed framework is compatible with BART (Lewis et al., 2020). For

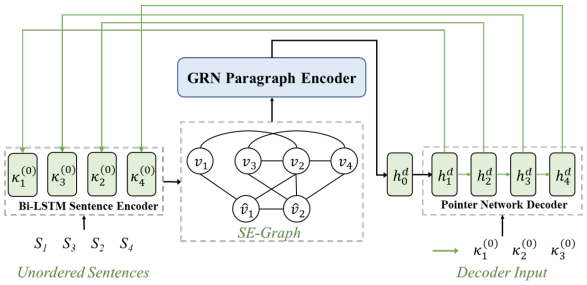

Figure 1: The architecture of SE-GRN model (Yin et al., 2019, 2021).

example, we can easily adapt the BART encoder as our sentence encoder.

With similar motivation with ours, that is, to combine advantages of above-mentioned two kinds of models, Yin et al. (2020) introduced three pairwise ordering predicting modules (FHDecoder) to enhance the pointer network decoder of ATTOrderNet. Recently, Cui et al. (2020) proposed BERSON that is also equipped with FHDecoder and utilizes BERT to exploit the deep semantic connection and relative ordering between sentences.

However, significantly different from them, we borrow the idea from the mask-predict framework (Gu et al., 2018; Ghazvininejad et al., 2019; Deng et al., 2020) to progressively incorporate pairwise ordering information into SE-Graph, which is the basis of our graph-based sentence ordering model. To the best of our knowledge, our work is the first attempt to explore iteratively refined GNN for sentence ordering.

3 Background

In this section, we give a brief introduction to the SE-GRN (Yin et al., 2019, 2021), which is selected as our baseline due to its competitive performance. As shown in Figure 1, SE-GRN is composed of a Bi-LSTM sentence encoder, GRN (Zhang et al., 2018) paragraph encoder, and a pointer network (Vinyals et al., 2015b) decoder.

3.1 Sentence-Entity Graph

The SE-GRN takes I sentences $\boldsymbol{s} = [s_{o_1}, \ldots, s_{o_I}]$ as input and tries to predict their correct order $\boldsymbol{o}^* = [o_1^*, \ldots, o_I^*]$. At first, each sentence s_{o_i} is fed into a Bi-LSTM sentence encoder, where the concatenation of the last hidden states in two directions is used as the context-aware sentence representation $\boldsymbol{\kappa}_{o_i}^{(0)}$. As illustrated in the middle of Figure 1, each input sentence set is represented as an undirected sentence-entity graph $G = (\boldsymbol{V}, \boldsymbol{E})$, where $\boldsymbol{V} = \{v_i\}_{i=1}^{I} \cup \{\hat{v}_j\}_{j=1}^{J}$ and $\boldsymbol{E} = \{e_{i,i'}\}_{i=1,i'=1}^{I,I} \cup \{\bar{e}_{i,j}\}_{i=1,j=1}^{I,J} \cup \{\hat{e}_{j,j'}\}_{j=1,j'=1}^{J,J}$ represent the nodes and edges respectively. Here, nodes include *sentence nodes* (such as v_i) and *entity nodes* (such as $\hat{v}_j$), and each edge is 1) *sentence-sentence edge* (*ss-edge*, such as $e_{i,i'}$) linking two sentences having the same entity; or 2) *sentence-entity edge* (*se-edge*, such as $\bar{e}_{i,j}$) connecting an entity to a sentence that contains it. Each se-edge is assigned with a label including *subject*, *object* or *other*, based on the syntactic role of its involved entity; or 3) *entity-entity edge* (*ee-edge*, such as $\hat{e}_{j,j'}$) connecting two semantic related entities. Besides, a virtual global node connecting to all nodes is introduced to capture global information effectively.

3.2 Paragraph Encoding with GRN

Node representations of each sentence and each entity are first initialized with the concatenation of bidirectional last states of the Bi-LSTM sentence encoder and the corresponding GloVe word embedding, respectively. Then, a GRN is adapted to encode the above sentence-entity graph, where node states are updated iteratively. During the process of updating hidden states, the messages for each node are aggregated from its adjacent nodes. Specifically, the sentence-level message $\boldsymbol{m}_i^{(l)}$ and entity-level message $\tilde{\boldsymbol{m}}_i^{(l)}$ for a sentence s_i are defined as follows:

$$\begin{aligned} \boldsymbol{m}_i^{(l)} &= \sum_{v_{i'} \in N_i} w(\boldsymbol{\kappa}_i^{(l-1)}, \boldsymbol{\kappa}_{i'}^{(l-1)}) \boldsymbol{\kappa}_{i'}^{(l-1)}, \\ \hat{\boldsymbol{m}}_i^{(l)} &= \sum_{v_j \in \hat{N}_i} \bar{w}(\boldsymbol{\kappa}_i^{(l-1)}, \boldsymbol{\epsilon}_j^{(l-1)}, \boldsymbol{r}_{ij}) \boldsymbol{\epsilon}_j^{(l-1)}, \end{aligned} \tag{1}$$

where $\boldsymbol{\kappa}_{i'}^{(l-1)}$ and $\boldsymbol{\epsilon}_j^{(l-1)}$ stand for the neighboring sentence and entity representations of the i-th sentence node v_i at the $(l-1)$-th layer, N_i and $\hat{N}_i$ denote the sets of neighboring sentences and entities of v_i, and both $w(*)$ and $\bar{w}(*)$ are gating functions with single-layer networks, involving associated node states and edge label r_{ij} (if any).

Afterwards, $\boldsymbol{\kappa}_i^{(l-1)}$ is updated by concatenating its original representation $\boldsymbol{\kappa}_i^{(0)}$, the messages from neighbours ($\boldsymbol{m}_i^{(l)}$ and $\hat{\boldsymbol{m}}_i^{(l)}$) and the global state $\boldsymbol{g}^{(l-1)}$ via GRU:

$$\begin{aligned} \boldsymbol{\xi}_i^{(l)} &= [\boldsymbol{\kappa}_i^{(0)}; \boldsymbol{m}_i^{(l)}; \hat{\boldsymbol{m}}_i^{(l)}; \boldsymbol{g}^{(l-1)}], \\ \boldsymbol{\kappa}_i^{(l)} &= \text{GRU}(\boldsymbol{\xi}_i^{(l)}, \boldsymbol{\kappa}_{i'}^{(l-1)}). \end{aligned} \tag{2}$$

Similar to updating sentence nodes, each entity state $\boldsymbol{\epsilon}_j^{(l-1)}$ is updated based on its word embedding

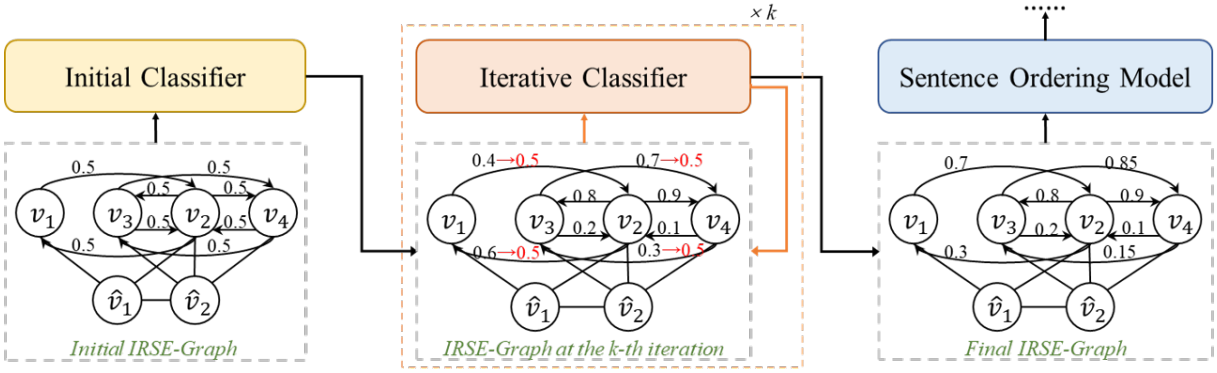

Figure 2: The architecture of our model during inference. IRSE-Graph is a weighted graph representation, of which weights of ss-edges are iteratively refined by iterative classifier. Note that we construct the sentence ordering model based on the final IRSE-Graph.

$\boldsymbol{emb}_j$, hidden states of its connected sentence nodes (such as $\boldsymbol{\kappa}_i^{(l\text{-}1)}$), and $\boldsymbol{g}^{(l\text{-}1)}$:

$$\begin{aligned}
\boldsymbol{m}_j^{(l)} &= \sum_{v_i \in N_j} \bar{w}(\boldsymbol{\epsilon}_j^{(l-1)}, \boldsymbol{\kappa}_i^{(l-1)}, \boldsymbol{r}_{ij}) \boldsymbol{\kappa}_i^{(l-1)}, \\
\hat{\boldsymbol{m}}_j^{(l)} &= \sum_{v_{j'} \in \hat{N}_j} \tilde{w}(\boldsymbol{\epsilon}_j^{(l-1)}, \boldsymbol{\epsilon}_{j'}^{(l-1)}) \boldsymbol{\epsilon}_j^{(l-1)}, \\
\boldsymbol{\xi}_j^{(l)} &= [\boldsymbol{emb}_j; \boldsymbol{m}_j^{(l)}; \hat{\boldsymbol{m}}_j^{(l)}; \boldsymbol{g}^{(l-1)}], \\
\boldsymbol{\epsilon}_j^{(l)} &= \mathrm{GRU}(\boldsymbol{\xi}_j^{(l)}, \boldsymbol{\epsilon}_j^{(l-1)}).
\end{aligned} \tag{3}$$

Finally, the messages from both sentence and entity states are used to update global state $\boldsymbol{g}^{(l\text{-}1)}$ via

$$\boldsymbol{g}^{(l)} = \mathrm{GRU}(\frac{1}{|\boldsymbol{V}|} \sum_{v_i \in \boldsymbol{V}} \boldsymbol{\kappa}_i^{(l-1)}, \frac{1}{|\hat{\boldsymbol{V}}|} \sum_{\hat{v}_j \in \hat{\boldsymbol{V}}} \boldsymbol{\epsilon}_j^{(l-1)}, \boldsymbol{g}^{(l-1)}). \tag{4}$$

The above updating process is iterated for L times. Usually, the top hidden states are considered as fine-grained graph representations, which will provide dynamical context for the decoder via attention mechanism.

3.3 Decoding with Pointer Network

Given the learned hidden states $\{\boldsymbol{\kappa}_i^{(L)}\}$ and $\boldsymbol{g}^{(L)}$, the prediction procedure for order $\boldsymbol{o}'$ can be formalized as follows:

$$\begin{aligned}
P(\boldsymbol{o}'|\boldsymbol{K}^{(L)}) &= \prod_{t=1}^{\boldsymbol{I}} P(o'_t|\boldsymbol{o}'_{<t}, \boldsymbol{K}^{(L)}_{o'_{t-1}}), \\
P(o'_t|\boldsymbol{o}'_{<t}, \boldsymbol{K}^{(L)}_{o'_{t-1}}) &= \mathrm{softmax}(\boldsymbol{q}^{\mathrm{T}} \tanh(\boldsymbol{W}\boldsymbol{h}_t^d + \boldsymbol{U}\boldsymbol{K}^{(L)}_{o'_{t-1}})), \\
\boldsymbol{h}_t^d &= \mathrm{LSTM}(\boldsymbol{h}_{t-1}^d, \boldsymbol{\kappa}^{(0)}_{o'_{t-1}}).
\end{aligned} \tag{5}$$

Here, $\boldsymbol{q}$, $\boldsymbol{W}$ and $\boldsymbol{U}$ are learnable parameters, $\boldsymbol{K}^{(L)}_{o'_{t-1}}$ and $\boldsymbol{h}_t^d$ denote the sentence representations with predicted order $\left[\boldsymbol{\kappa}^{(L)}_{o'_1}, \ldots, \boldsymbol{\kappa}^{(L)}_{o'_{t-1}}\right]$ and the decoder hidden state at the t-th time step, which is initialized by $\boldsymbol{g}^{(L)}$ as t=0, respectively.

4 Our Framework

In this section, we give a detailed description to our framework. As shown in Figure 2, we first introduce two graph-based classifiers to construct an iteratively refined sentence-entity graph (IRSE-Graph). It is a weighted version of SE-Graph, where pairwise ordering inforamtion is iteratively incorporated to update ss-edge weights. Then, we adapt the conventional GRN to establish a neural sentence ordering model based on the final IRSE-Graph.

4.1 The Definition of IRSE-Graph

As an extension of SE-Graph, IRSE-Graph can be denoted as G=($\boldsymbol{V}$,$\boldsymbol{E}$,$\boldsymbol{W}$), where $\boldsymbol{V}$ and $\boldsymbol{E}$ share the same definitions with those of SE-Graph. Particularly, in IRSE-Graph, each ss-edge $e_{i,i'}$ is a directed one with a weight $w_{i,i'} \in \boldsymbol{W}$ indicating the probability of sentence s_i occurring before sentence $s_{i'}$. Meanwhile, there must exist a corresponding ss-edge $e_{i',i}$ with the weight $w_{i',i}$=1−$w_{i,i'}$ denoting the probability of s_i appearing after $s_{i'}$. For example, in Figure 2, for two linked sentence nodes v_1 and v_2, there exist two ss-edges $e_{1,2}$ and $e_{2,1}$ with weights $w_{1,2}$ and $w_{2,1}$ respectively, both of which are iteratively updated during constructing IRSE-Graph.

4.2 Constructing IRSE-Graph

Inspired by Gui et al. (2020), we successively introduce two classifiers — *initial classifier* and *iterative classifier* to construct IRSE-Graph. Both classifiers are constructed using slightly adapted GRN and utilized to deal with different scenarios, respectively. In this way, we can fully exploit the

potential of iterative classifier to predict better pairwise orderings. We will give a detail introduction to the slightly adapted GRN in Section §4.3.

To better understand the procedure of constructing IRSE-Graph, we provide the details in Algorithm 1. During this procedure, pairwise orderings are iteratively predicted and gradually incorporated to refine IRSE-Graph. Here we introduce a set $VP^{(k)}$ to collect sentence node pairs with uncertain pairwise orderings at the k-th iteration.

First, we bulid an initial classifier based on the initial IRSE-Graph, where the learned sentence representations are used to predict pairwise orderings between any two linked sentences only once (**Lines 2-6**). Note that in the initial IRSE-Graph, all weights of ss-edges are set to 0.5. In this case, IRSE-Graph degrades to the conventional SE-Graph. Concretely, for any two linked sentence nodes v_i and $v_{i'}$, we concatenate their vector representations $\boldsymbol{\kappa}_i$ and $\boldsymbol{\kappa}_{i'}$ as $[\boldsymbol{\kappa}_i; \boldsymbol{\kappa}_{i'}]$ and $[\boldsymbol{\kappa}_{i'}; \boldsymbol{\kappa}_i]$, which are fed into an MLP classifier to obtain two probabilities. Then, we normalize and convert these two probabilities into ss-edge weights $w_{i,i'}$ and $w_{i',i}$. If both $w_{i,i'}$ and $w_{i',i}$ are within a prefixed interval $[\delta_{min}, \delta_{max}]$, we consider $(v_i, v_{i'})$ as a sentence node pair with uncertain pairwise ordering and add it into $VP^{(0)}$. Moreover, we replace both $w_{i,i'}$ and $w_{i',i}$ with 0.5, indicating that they will be repredicted in the next iteration.

In the following, we also construct an iterative classifier based on IRSE-Graph. However, in an easy-to-hard manner, we use iterative classifier to perform pairwise ordering predictions, where the ss-edge weights of IRSE-Graph are continously updated with previously-predicted pairwise orderings with high confidence (**Lines 13-26**). By doing so, graph representations can be continously refined for better subsequent predictions. More specifically, the k-th iteration of this classifier mainly involve three steps:

In **Step 1**, based on the current IRSE-Graph, we employ the adapted GRN to conduct graph encoding to learn sentence representations (**Line 15**).

In **Step 2**, on the top of learned sentence representations, we stack an MLP classifier to predict pairwise orderings for sentence node pairs in $VP^{(k)}$ (**Lines 16-19**). Likewise, we collect sentence node pairs with uncertain pairwise orderings to form $VP^{(k+1)}$, and reassign their corresponding ss-edge weights as 0.5, so as to avoid the negative effect of these uncertain ss-edge weights during the next iteration (**Lines 20-23**).

In **Step 3**, if $VP^{(k+1)}$ is equal to $VP^{(k)}$ or empty, we believe the learning of IRSE-Graph G has converged and thus return it (**Lines 26-27**).

Although both of our classifiers are constructed using IRSE-Graph, their training procedures are slightly different. As for initial classifier, we directly train it on the initial IRSE-Graph without any pairwise ordering information (all ss-edge weights are set to 0.5). By contrast, we train iterative classifier on IRSE-Graph with partial pairwise orderings. To enable iterative classifier generalizable to any IRSE-Graph with partial predicted pairwise orderings, we first set all ss-edge weights to 1 or 0 according to their ground-truth pairwise orderings, and then train the classifier to correctly predict pari-

Algorithm 1 The procedure of constructing IRSE-Graph

Input: the initial IRSE-Graph: $G=(\boldsymbol{V}, \boldsymbol{E}, \boldsymbol{W})$ with all $w_{i,i'}=0$; two thresholds: $\delta_{min}, \delta_{max}$
Output: the final IRSE-Graph: $G = (\boldsymbol{V}, \boldsymbol{E}, \boldsymbol{W})$

1: $VP^{(0)} \leftarrow \emptyset$
2: $\{\boldsymbol{\kappa}_i\}_{i=1}^{I} \leftarrow \text{GRN}(G)$
3: **for** any linked sentence node pair $(v_i, v_{i'})$ **&&** $i<i'$ **do**
4: $\quad w_{i,i'} \leftarrow \text{InitialClassifer}([\boldsymbol{\kappa}_i; \boldsymbol{\kappa}_{i'}])$
5: $\quad w_{i',i} \leftarrow \text{InitialClassifer}([\boldsymbol{\kappa}_{i'}; \boldsymbol{\kappa}_i])$
6: $\quad w_{i,i'}, w_{i',i} \leftarrow \text{Normalize}(w_{i,i'}, w_{i',i})$
7: $\quad$ **if** $\delta_{min} \leq w_{i,i'} \leq \delta_{max}$ **then**
8: $\quad\quad VP^{(0)} \leftarrow VP^{(0)} \cup \{(v_i, v_{i'})\}$
9: $\quad\quad w_{i,i'} \leftarrow 0.5, w_{i',i} \leftarrow 0.5$
10: $\quad$ **end if**
11: **end for**
12: $k \leftarrow 0$
13: **repeat**
14: $\quad VP^{(k+1)} \leftarrow \emptyset$
15: $\quad \{\boldsymbol{\kappa}_i\}_{i=1}^{I} \leftarrow \text{GRN}(G)$
16: $\quad$ **for** $(v_i, v_{i'}) \in VP^{(k)}$ **do**
17: $\quad\quad w_{i,i'} \leftarrow \text{IterativeClassifer}([\boldsymbol{\kappa}_i; \boldsymbol{\kappa}_{i'}])$
18: $\quad\quad w_{i',i} \leftarrow \text{IterativeClassifer}([\boldsymbol{\kappa}_{i'}; \boldsymbol{\kappa}_i])$
19: $\quad\quad w_{i,i'}, w_{i',i} \leftarrow \text{Normalize}(w_{i,i'}, w_{i',i})$
20: $\quad\quad$ **if** $\delta_{min} \leq w_{i,i'} \leq \delta_{max}$ **then**
21: $\quad\quad\quad VP^{(k+1)} \leftarrow VP^{(k+1)} \cup \{(v_i, v_{i'})\}$
22: $\quad\quad\quad w_{i,i'} \leftarrow 0.5, w_{i',i} \leftarrow 0.5$
23: $\quad\quad$ **end if**
24: $\quad$ **end for**
25: $\quad k \leftarrow k+1$
26: **until** $VP^{(k+1)} == VP^{(k)}$ || $VP^{(k)} == \emptyset$
27: **return** G

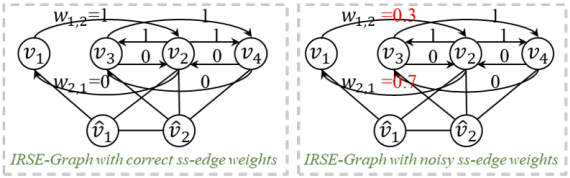

Figure 3: Introducing noisy ss-edge weights into IRSE-Graph.

wise orderings for other pairs. Concretely, if $\boldsymbol{s}_i$ appears before $\boldsymbol{s}_{i'}$, we set $w_{i,i'}$=1 and $w_{i',i}$=0, vice versa. For example, in the left part of Figure 3, the ground-truth sentence sequence is s_1,s_2,s_3,s_4, and thus we assign the ss-edge weights of linked sentence node pairs (v_1, v_2), (v_3, v_2), (v_3, v_4), (v_2, v_4) as follows: $w_{1,2}$=1, $w_{2,3}$=1, $w_{2,4}$=1, $w_{3,4}$=1, and $w_{2,1}$=0, $w_{3,2}$=0, $w_{4,2}$=0, $w_{4,3}$=0.

Moreover, to enhance the robustness of the iterative classifier, we randomly select a certain ratio η of sentence pairs and assign their ss-edges with incorrect weights. Let us revisit Figure 3, for the randomly selected sentence node pair (v_1, v_2), we assign ss-edges weights $w_{1,2}$ and $w_{2,1}$ with randomly generated noisy values 0.3 and 0.7 respectively. In this way, we expect that iterative classifier can conduct correct predictions even given incorrect previously-predicted pairwise orderings.

4.3 IRSE-Graph Sentence Ordering Model

Finally, following the conventional SE-GRN (Yin et al., 2019, 2021), we construct a graph-based sentence ordering model. Note that the above two classifiers and our sentence ordering model are all based on IRSE-Graph rather than the conventional SE-Graph, which makes the standard GRN unable to be applied directly. To deal with this issue, we slightly adapt GRN to utilize pairwise ordering information for graph encoding. Specifically, we adapt Equation 1 to incorporate ss-edge weights into the message aggregation of sentence-level nodes:

$$\begin{aligned} \boldsymbol{m}_i^{(l)} &= \sum_{v_{i'} \in N_i} w_{i,i'} \cdot w(\boldsymbol{\kappa}_i^{(l\text{-}1)}, \boldsymbol{\kappa}_{i'}^{(l\text{-}1)}) \boldsymbol{\kappa}_{i'}^{(l\text{-}1)}, \\ w(\boldsymbol{\kappa}_i^{(l\text{-}1)}, \boldsymbol{\kappa}_{i'}^{(l\text{-}1)}) &= \sigma(W_g[\boldsymbol{\kappa}_i^{(l\text{-}1)}; \boldsymbol{\kappa}_{i'}^{(l\text{-}1)}]). \end{aligned} \tag{6}$$

Here σ denotes sigmoid function and W_g is learnable parameter matrix. Equation 6 expresses that the sentence-level aggregation should consider not only the semantic representations of the two involved sentences, but also the relative ordering between them. In addition, other Equations are the same as those of conventional GRN, which have been described in Section §3.2.

5 Experiment

5.1 Setup

Datasets. Following previous work (Yin et al., 2020; Cui et al., 2018; Yin et al., 2021), we carry out experiments on five benchmark datasets:

- **SIND, ROCStory**. SIND (Huang et al., 2016) is a visual storytelling dataset and ROCStory (Mostafazadeh et al., 2016) is about commonsense stories. Both two datasets are composed of 5-sentence stories and randomly split by 8:1:1 for the training/validation/test sets.
- **NIPS Abstract, AAN Abstract, arXiv Abstract**. These three datasets consist of abstracts from research papers, which are collected from NIPS, ACL anthology and arXiv, respectively (Radev et al., 2016; Chen et al., 2016). The partitions for training/validation/test of each dataset are as follows: NIPS Abstract: 2,427/408/377, AAN Abstract: 8,569/962/2,626, arXiv Abstract: 884,912/110,614/110,615 for the training/validation/test sets.

Settings. For fair comparison, we use the same settings as our most related baseline SE-GRN (Yin et al., 2021) for our model and its variants. Specifically, we apply 100-dimensional GloVe word embeddings, and set the sizes of Bi-LSTM hidden states, sentence node states, and entity node states as 512, 512 and 150, respectively. The recurrent step of GRN is 3. We empirically set thresholds δ_{min} and δ_{max} as 0.2 and 0.8, and set η as 20%, 15%, 25%, 15%, 15% according to accuracies of initial classifier on validation sets. Besides, we individually set the coefficient λ (See Equation 18 in (Yin et al., 2020)) as 0.5, 0.5, 0.2, 0.4, 0.5 on the five datasets. We adopt Adadelta (Zeiler, 2012) with $\epsilon = 10^{-6}$, $\rho = 0.95$ and initial learning rate 1.0 as the optimizer. We employ L2 weight decay with coefficient 10^{-5}, batch size of 16 and dropout rate of 0.5.

When constructing our model based on BERT, we use the same settings as (Cui et al., 2020). Concretely, we set sizes of hidden states and node states to 768, the learning rate of BERT as 3e-3, the batch size as 16, 32, 128, 128, 64 for the five datasets.

Baselines. To demonstrate the effectiveness of our model (IRSE-GRN), we compare it with SE-GRN (Yin et al., 2021). Besides, we report the performance of following sentence ordering models: 1) **Pairwise models**: Pairwise Model (Chen et al., 2016), RankTxNet (Kumar et al., 2020), and BTSort (Prabhumoye et al., 2020), ConsGraph (Zhu et al., 2021); 2) **Set-to-sequence models**: HAN (Wang and Wan, 2019), LSTM+PtrNet (Gong et al., 2016), V-LSTM+PtrNet (Logeswaran et al., 2018), ATTOrderNet (Cui et al., 2018), TGCM (Oh et al., 2019), SE-GRN (Yin et al., 2019), SE-GRN (Yin et al., 2021), ATTOrderNet+FHDecoder (Yin et al.,

Model	NIPS Abstract			AAN Abstract			SIND			ROCStory			arXiv Abstract		
	Acc	τ	PMR	Acc	τ	PMR	Acc	τ	PMR	Acc	τ	PMR	Acc	τ	PMR
Pairwise Model (Chen et al., 2016)†	-	-	-	-	-	-	-	-	-	-	-	-	-	66.00	33.43
LSTM+PtrNet (Gong et al., 2016)†	50.87	67.00	-	58.20	69.00	-	-	48.42	12.34	-	-	-	-	71.58	40.44
V-LSTM+PtrNet (Logeswaran et al., 2018)†	51.55	72.00	-	58.07	73.00	-	-	-	-	-	-	-	-	-	-
ATTOrderNet (Cui et al., 2018)†	56.09	72.00	-	63.24	73.00	-	-	49.00	14.01	-	-	-	-	73.00	42.19
HAN (Wang and Wan, 2019)†	-	-	-	-	-	-	-	50.00	15.01	-	73.00	39.62	-	75.00	44.55
SE-GRN (Yin et al., 2019)†	57.27	75.00	-	64.64	78.00	-	-	52.00	16.22	-	-	-	-	75.00	44.33
SE-GRN (Yin et al., 2021)	58.25	76.49	25.73	65.06	78.60	44.87	49.58	53.16	17.17	68.96	75.46	42.67	59.07	75.74	44.72
ATTOrderNet+FHDecoder (Yin et al., 2020)†	-	-	-	-	-	-	-	53.19	17.37	-	76.81	46.00	-	76.54	46.58
TGCM (Oh et al., 2019)†	59.43	75.00	31.44	65.16	75.00	36.69	38.71	15.18	53.00	-	-	-	58.31	75.00	44.28
RankTxNet (Kumar et al., 2020)†	-	75.00	24.13	-	77.00	39.18	-	57.00	15.48	-	76.00	38.02	-	77.00	43.44
B-TSort (Prabhumoye et al., 2020)†	61.48	81.00	32.59	69.22	83.00	50.76	52.23	60.00	20.32	-	-	-	-	-	-
ConsGraph (Zhu et al., 2021)†	-	80.29	32.84	-	82.36	49.81	-	58.56	19.07	-	81.22	49.52	-	-	-
BERSON (Cui et al., 2020)†	73.87	85.00	48.01	78.03	85.00	59.79	58.91	65.00	**31.69**	82.86	88.00	68.23	75.08	83.00	56.06
IRSE-GRN	63.14	80.45	32.63	68.51	82.09	49.56	51.01	54.97	18.77	71.28	77.43	46.38	70.15	84.22	56.85
IRSE-GRN+FHDecoder	73.62	87.45	50.19	77.34	87.87	62.24	54.98	61.87	22.77	77.70	84.20	57.11	74.45	88.57	60.30
IRSE-GRN+BERT+FHDecoder	**78.00**	**90.35**	**58.81**	**82.07**	**91.11**	**68.93**	**59.08**	**66.14**	28.79	**83.77**	**89.09**	**69.06**	**78.64**	**90.30**	**66.59**

Table 1: Main results on the sentence ordering task, where † indicates previously reported scores. Please note that RankTxNet, B-TSort and ConsGraph are pairwise models based on BERT, and the previous SOTA BERSON is also based on BERT and equipped with FHDecoder.

2020) and BERSON (Cui et al., 2020).

Furthermore, to examine the compatibility of other technologies with our model, we report the performance of IRSE-GRN equipped with some effective components: 1) **IRSE-GRN+FHDecoder**. In this variant, we equip our model with FHDecoder (Yin et al., 2020), where pairwise ordering information is incorporated; 2) **IRSE-GRN+BERT+FHDecoder**. In addition to FHDecoder, we construct the sentence encoder based on BERT, where the mean-pooling outputs of all learned word representations are used to initialize sentence nodes.

Evaluation Metrics. Following previous work (Oh et al., 2019; Cui et al., 2020; Prabhumoye et al., 2020; Zhu et al., 2021; Yin et al., 2021), we use the following three metrics: 1) **Kendall's Tau (τ)**: Formally, this metric is calculated as 1-$2\times$(*number of inversions*)/$\binom{I}{2}$, where I denotes the sequence length and *number of inversions* is the number of pairs in the predicted sequence with incorrect relative order (Lapata, 2003); 2) **Perfect Match Ratio (PMR)**: This metric calculates the ratio of samples where the entire sequence is correctly predicted (Chen et al., 2016); 3) **Accuracy (Acc)**: This metric measures the percentage of sentences, whose absolute positions are correctly predicted (Logeswaran et al., 2018).

5.2 Pairwise Ordering

Since pairwise ordering plays a crucial role in our proposed framework, we first compare the performance of different classifiers on various datasets. Table 2 shows the experimental results. Obviously, the utilization of iterative classifier further benefits the predictions of pairwise orderings.

Dataset	Initial Classifier	Initial + Iterative Classifiers
NIPS Abstract	80.46%	**86.32%**
AAN Abstract	84.53%	**86.74%**
SIND	77.72%	**83.55%**
ROCstory	87.59%	**92.23%**
arXiv Abstract	84.09%	**86.82%**

Table 2: The accuracies of our two classifiers on five test datasets.

5.3 Main Results

Table 1 reports the overall experimental results of sentence ordering. When incorporating BERT and FHDecoder into IRSE-GRN, our model achieves SOTA performance on most of datasets. Besides, we arrive at the following conclusions:

First, IRSE-GRN significantly surpasses SE-GRN on all datasets (bootstrapping test, $p<0.01$), indicating that iteratively refining graph representations indeed benefit the ordering of input sentences.

Second, IRSE-GRN+FHDecoder exhibits better performance than IRSE-GRN and all non-BERT baselines, which are shown above the upper dotted line of Table 1, across datasets in different domains. Therefore, we confirm that our framework is orthogonal to the current approach exploiting pairwise ordering information for decoder.

Third, when constructing our model based on BERT, IRSE-GRN+BERT+FHDecoder also outperforms all BERT-based baselines, such as ConsGraph, BERSON, achieving SOTA performance. It can be known that our proposed framework is also effective when combining with pretrained language model.

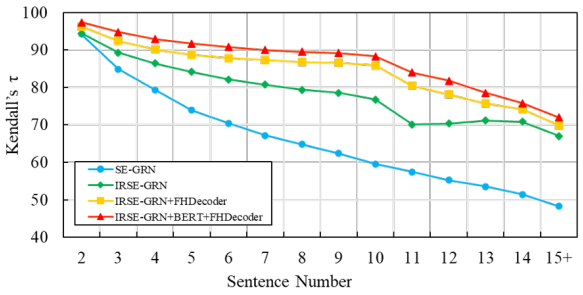

Figure 4: The Kendall's τ of different models with respect to different sentence numbers on the arXiv abstract test set.

Finally, we note that IRSE-GRN+BERT+FHDecoder gains relatively marginal improvement on SIND and ROCStory, and performs worse than BERSON in PMR on SIND. We speculate that there exist less ss-edges on these two datasets, resulting in that our proposed framework can not achieve its full potential. Specifically, average edge numbers of SIND and ROCStory are 2.85 and 5.66 respectively, far fewer than 16.60, 10.86 and 16.73 on NIPS Abstract, ANN Abstract and arXiv Abstract.

Besides, since it is a challenge to order longer paragraphs, we investigate the Kendall's τ of our models and SE-GRN with respect to different sentence numbers, as shown in Figure 4. Overall, all models degrade with the increase of sentence number. However, our model and its two enhanced versions always exhibit better performance than SE-GRN.

5.4 Predictions of the First and Last Sentences

As mentioned in previous studies (Gong et al., 2016; Chen et al., 2016; Cui et al., 2018; Oh et al., 2019), the first and last sentences are very important in a paragraph. Following these studies, we compare models by conducting experiments to predict the first and last sentences.

As displayed in Table 3, IRSE-GRN surpasses all non-BERT baselines, and IRSE-GRN+BERT+FHDecoder wins against BERTSON. These results are consistent with those reported in Table 1, further demonstrating the effectiveness of our model.

5.5 Ablation Study

We conduct several experiments to investigate the impacts of our proposed components on ROCstory dataset and arXiv dataset which are the two largest datasets. All results are provided in Table 4, where we draw the following conclusions:

Model	SIND		arXiv Abstract	
	head	tail	head	tail
Pairwise Model (Chen et al., 2016)†	-	-	84.85	62.37
LSTM+PtrNet (Gong et al., 2016)†	74.66	53.30	90.47	66.49
ATTOrderNet (Cui et al., 2018)†	76.00	54.42	91.00	68.08
SE-GRN (Yin et al., 2019)†	78.12	56.68	92.28	70.45
SE-GRN (Yin et al., 2021)	79.01	57.27	92.23	70.46
ATTOrderNet+FHDecoder (Yin et al., 2020)†	78.08	57.32	92.76	71.49
TGCM (Oh et al., 2019)†	78.98	56.24	92.46	69.45
RankTxNet (Kumar et al., 2020)†	80.32	59.68	92.97	69.13
B-Tsort (Prabhumoye et al., 2020)†	78.06	58.36	-	-
ConsGraph (Zhu et al., 2021)†	79.80	60.44	-	-
BERSON (Cui et al., 2020)†	84.95	64.87	94.75	76.69
IRSE-GRN	78.62	59.11	94.46	80.97
IRSE-GRN+FHDecoder	82.87	64.15	96.09	85.04
IRSE-GRN+BERT+FHDecoder	**86.21**	**67.14**	**98.23**	**88.33**

Table 3: The ratios of correctly predicting first and last sentences on arXiv Abstract and SIND. † indicates previously reported scores.

Model	ROCStory			arXiv Abstract		
	Pairwise	τ	PMR	Pairwise	τ	PMR
IRSE-GRN	**92.23**	**77.43**	**46.38**	**86.82**	**84.22**	**56.85**
w/o initial classifier	88.96	77.31	45.06	77.90	81.03	51.65
iterative number k=1	91.73	77.21	46.13	86.22	83.89	56.03
w/o iterative classifier	87.59	75.98	44.14	84.09	83.24	55.05
w/o noise	90.42	77.06	46.02	80.45	82.23	53.10

Table 4: Ablation study on the impacts of our proposed components on ROCStory dataset and arXiv abstract dataset.

First, using only iterative classifier, IRSE-GRN(w/o initial classifier) performs worse than IRSE-GRN. This result proves that iterative classifier fails to predict well from scratch and the pairwise ordering predicted by initial classifier is beneficial to construct a well-formed graph representation for iterative classifier.

Second, when the iteration number k is set as 1, the performance of IRSE-GRN decreases. Moreover, if we remove iterative classifier, the performance of IRSE-GRN becomes even worse. Therefore, we confirm that the iterative predictions of pairwise ordering indeed benefit the learning of graph representations.

Finally, the result in the last line indicates that removing noisy weights leads to a significant performance drop. It suggests that the utilization of noisy weights is useful for the training of iterative classifier, which makes our model more robust.

5.6 Summary Coherence Evaluation

Following previous studies (Barzilay and Lapata, 2005; Nayeem and Chali, 2017), we further in-

Dataset	SE-GRN		IRSE-GRN		IRSE-GRN+FHDecoder		IRSE-GRN+BERT+FHDecoder	
	Runtime	#Params	Runtime	#Params	Runtime	#Params	Runtime	#Params
NIPS abstract	6s	23.9M	6.2s	24.0M	18s	25.0M	29s	128.0M
AAN abstract	31s	23.9M	32.5s	24.0M	1min8s	25.0M	1min20s	128.0M
SIND	1min6s	23.9M	1min9s	24.0M	2min3s	25.0M	2min16s	128.0M
ROCStory	2min	23.9M	2min5s	24.0M	4min2s	25.0M	4min42s	128.0M
arXiv abstract	25min	23.9M	27min57s	24.0M	46min	25.0M	56min	128.0M

Table 6: The runtime on the validation sets and the numbers of parameters for our enhanced models and baseline.

Model	Coherence	
SE-GRN (Yin et al., 2021)	46.71	59.47
IRSE-GRN	47.48	60.01
IRSE-GRN+FHDecoder	49.84	61.81
IRSE-GRN+BERT+FHDecoder	51.01	62.87

Table 5: Coherence probabilities of summaries reordered by different models using weights of 0.8 (left) and 0.5 (right).

spect the validity of our proposed framework via multi-document summarization. Concretely, we train different neural sentence ordering models on a large-scale summarization corpus (Fabbri et al., 2019), and then individually use them to reorder the small-scale summarization data of DUC2004 (Task2). Finally, we use coherence probability proposed by (Nayeem and Chali, 2017) to evaluate the coherence of summaries. In this group of experiments, we conduct experiments using different weights: 0.5 and 0.8, as implemented in (Nayeem and Chali, 2017) and (Yin et al., 2020) respectively.

The results are reported in Table 5. We can observe that the summaries reordered by IRSE-GRN and its variants achieve higher coherence probabilities than baseline, verifying the effectiveness of our proposed framework in the downstream task.

5.7 Further Experiment Results

To provide more experimental results, we summarize the runtime on the validation sets and the numbers of parameters for our enhanced models and baseline SE-GRN in Table 6.

6 Conclusion

In this work, we propose a novel sentence ordering framework that makes better use of pairwise orderings for graph-based sentence ordering. Specifically, we introduce two classifiers to iteratively predict pairwise orderings, which are gradually incorporated into the graph as edge weights. Then, based on this refined graph, we construct a graph-based sentence ordering model. Experiments on five datasets demonstrate not only the superiority of our model over baselines, but also the compatibility to other modules utilizing pairwise ordering information. Moreover, when equipped with BERT and FHDecoder, our enhanced model achieves SOTA performance across datasets.

In the future, we plan to explore more effective GNN for sentence ordering. In particular, we will improve our model by iteratively merging nodes to refine the graph representation.

Acknowledgment

The project was supported by National Key Research and Development Program of China (No. 2020AAA0108004), National Natural Science Foundation of China (No. 61672440), Natural Science Foundation of Fujian Province of China (No. 2020J06001), and Youth Innovation Fund of Xiamen (No. 3502Z20206059). We also thank the reviewers for their insightful comments.

References

Harsh Agrawal, Arjun Chandrasekaran, Dhruv Batra, Devi Parikh, and Mohit Bansal. 2016. Sort story: Sorting jumbled images and captions into stories. In *EMNLP*, pages 925–931.

Regina Barzilay, Noemie Elhadad, and Kathleen R. McKeown. 2002. Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, 17:35–55.

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *ACL*, pages 141–148.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *NAACL*, pages 113–120.

Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2016. Neural sentence ordering. *arXiv:1607.06952*.

Somnath Basu Roy Chowdhury, Faeze Brahman, and Snigdha Chaturvedi. 2021. Reformulating sentence ordering as conditional text generation. *arXiv:2104.07064*.

Baiyun Cui, Yingming Li, Ming Chen, and Zhongfei Zhang. 2018. Deep attentive sentence ordering network. In *EMNLP*, pages 4340–4349.

Baiyun Cui, Yingming Li, and Zhongfei Zhang. 2020. Bert-enhanced relational sentence ordering network. In *EMNLP*, pages 6310–6320.

Chaorui Deng, Ning Ding, Mingkui Tan, and Qi Wu. 2020. Length-controllable image captioning. In *ECCV*, pages 712–729.

J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186.

Micha Elsner and Eugene Charniak. 2011. Extending the entity grid with entity-specific features. In *ACL*, pages 125–129.

Alexander R. Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir R. Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *ACL*, pages 1074–1084.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *EMNLP-IJCNLP*, pages 6111–6120.

Jingjing Gong, Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2016. End-to-end neural sentence ordering using pointer network. *arXiv:1611.04953*.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *ICLR*.

Tao Gui, Jiacheng Ye, Qi Zhang, Zhengyan Li, Zichu Fei, Yeyun Gong, and Xuanjing Huang. 2020. Uncertainty-aware label refinement for sequence labeling. In *EMNLP*, pages 2316–2326.

Camille Guinaudeau and Michael Strube. 2013. Graph-based local coherence modeling. In *ACL*, pages 93–103.

Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. 2018. Learning to write with cooperative discriminators. In *ACL*, pages 1638–1649.

Ting-Hao Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Aishwarya Agrawal, Jacob Devlin, Ross Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, C. Lawrence Zitnick, Devi Parikh, Lucy Vanderwende, Michel Galley, and Margaret Mitchell. 2016. Visual storytelling. In *NAACL*, pages 1233–1239.

Ioannis Konstas and Mirella Lapata. 2012. Unsupervised concept-to-text generation with hypergraphs. In *NAACL*, pages 752–761.

Pawan Kumar, Dhanajit Brahma, Harish Karnick, and Piyush Rai. 2020. Deep attentive ranking networks for learning to order sentences. In *AAAI*, pages 8115–8122.

Mirella Lapata. 2003. Probabilistic text structuring: Experiments with sentence ordering. In *ACL*, pages 545–552.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*, pages 7871–7880.

Jiwei Li and Dan Jurafsky. 2017. Neural net models of open-domain discourse coherence. In *EMNLP*, pages 198–209.

Lajanugen Logeswaran, Honglak Lee, and Dragomir R. Radev. 2018. Sentence ordering and coherence modeling using recurrent neural networks. In *AAAI*, pages 5285–5292.

Muhammad Tasnim Mohiuddin, Shafiq R. Joty, and Dat Tien Nguyen. 2018. Coherence modeling of asynchronous conversations: A neural entity grid approach. In *ACL*, pages 558–568.

Han Cheol Moon, Tasnim Mohiuddin, Shafiq R. Joty, and Xu Chi. 2019. A unified neural coherence model. In *EMNLP-IJCNLP*, pages 2262–2272.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *NAACL*, pages 839–849.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2012. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*, pages 3075–3081.

Mir Tafseer Nayeem and Yllias Chali. 2017. Extract with order for coherent multi-document summarization. In *ACL TextGraphs Workshop*, pages 51–56.

Dat Tien Nguyen and Shafiq Rayhan Joty. 2017. A neural local coherence model. In *ACL*, pages 1320–1330.

Byungkook Oh, Seungmin Seo, Cheolheon Shin, Eunju Jo, and Kyong-Ho Lee. 2019. Topic-guided coherence modeling for sentence ordering by preserving global and local information. In *EMNLP-IJCNLP*, pages 2273–2283.

Shrimai Prabhumoye, Ruslan Salakhutdinov, and Alan W. Black. 2020. Topological sort for sentence ordering. In *ACL*, pages 2783–2792.

Dragomir R. Radev, Mark Thomas Joseph, Bryan R. Gibson, and Pradeep Muthukrishnan. 2016. A bibliometric and network analysis of the field of computational linguistics. *Journal of the Association for Information Science and Technology*, 67(3):683–706.

Linfeng Song, Daniel Gildea, Yue Zhang, Zhiguo Wang, and Jinsong Su. 2019. Semantic neural machine translation using AMR. *Transactions of the Association for Computational Linguistics*, 7:19–31.

Linfeng Song, Ante Wang, Jinsong Su, Yue Zhang, Kun Xu, Yubin Ge, and Dong Yu. 2020. Structural information preserving for graph-to-text generation. In *ACL*, pages 7987–7998.

Linfeng Song, Zhiguo Wang, Mo Yu, Yue Zhang, Radu Florian, and Daniel Gildea. 2018. Exploring graph-structured passage representation for multi-hop reading comprehension with graph neural networks. *arXiv:1809.02040*.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015a. Pointer networks. In *NIPS*, page 3104–3112.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015b. Pointer networks. In *NIPS*, pages 2692–2700.

Tianming Wang and Xiaojun Wan. 2019. Hierarchical attention networks for sentence ordering. In *AAAI*, pages 7184–7191.

Mengge Xue, Weiming Cai, Jinsong Su, Linfeng Song, Yubin Ge, Yubao Liu, and Bin Wang. 2019. Neural collective entity linking based on recurrent random walk network learning. In *IJCAI*, pages 5327–5333.

Yongjing Yin, Shaopeng Lai, Linfeng Song, Chulun Zhou, Xianpei Han, Junfeng Yao, and Jinsong Su. 2021. An external knowledge enhanced graph-based neural network for sentence ordering. *Journal of Artificial Intelligence Research*, 70:545–566.

Yongjing Yin, Fandong Meng, Jinsong Su, Yubin Ge, Linfeng Song, Jie Zhou, and Jiebo Luo. 2020. Enhancing pointer network for sentence ordering with pairwise ordering predictions. In *AAAI*, pages 9482–9489.

Yongjing Yin, Linfeng Song, Jinsong Su, Jiali Zeng, Chulun Zhou, and Jiebo Luo. 2019. Graph-based neural sentence ordering. In *IJCAI*, pages 5387–5393.

Matthew D. Zeiler. 2012. ADADELTA: an adaptive learning rate method. *arXiv:1212.5701*.

Yue Zhang, Qi Liu, and Linfeng Song. 2018. Sentence-state lstm for text representation. In *ACL*, pages 317–327.

Yutao Zhu, Kun Zhou, Jian-Yun Nie, Shengchao Liu, and Zhicheng Dou. 2021. Neural sentence ordering based on constraint graphs. In *AAAI*, pages 14656–14664.