BERT-enhanced Relational Sentence Ordering Network

Baiyun Cui[1], **Yingming Li**[1]*, **and Zhongfei Zhang**[2]
[1]College of Information Science and Electronic Engineering, Zhejiang University, China
[2]Computer Science Department, Binghamton University, Binghamton, NY, USA
baiyunc@yahoo.com, yingming@zju.edu.cn, zzhang@binghamton.edu

Abstract

In this paper, we introduce a novel BERT-enhanced Relational Sentence Ordering Network (referred to as BERSON) by leveraging BERT for capturing a better dependency relationship among sentences to enhance the coherence modeling for the entire paragraph. In particular, we develop a new Relational Pointer Decoder (referred as RPD) by incorporating the relative ordering information into the pointer network with a Deep Relational Module (referred as DRM), which utilizes BERT to exploit the deep semantic connection and relative ordering between sentences. This enables us to strengthen both local and global dependencies among sentences. Extensive evaluations are conducted on six public datasets. The experimental results demonstrate the effectiveness and promise of BERSON, showing a significant improvement over the state-of-the-art by a wide margin.

1 Introduction

Coherence modeling is one of the essential aspects of natural language processing (Xu et al., 2019; Mesgar et al., 2019; Moon et al., 2019; Farag and Yannakoudakis, 2019). A coherent text can facilitate understanding and avoid the confusion for reading comprehension. The Sentence Ordering task (Barzilay and Lapata, 2008) aims to reconstruct a coherent paragraph from an unordered set of sentences and has shown to be beneficial to improve the coherence in many NLP tasks including multi-document summarization (Barzilay and Elhadad, 2002; Nallapati et al., 2017), conversational analysis (Zeng et al., 2018), and text generation (Konstas and Lapata, 2013; Holtzman et al., 2018). Table 1 shows an example of this task.

An unordered set of sentences		Coherent paragraph	
1	Dan was walking during the night.	1	Dan was walking during the night.
3	They tried to steal his book bag.	2	A group of thieves surrounded him.
4	A bystander noticed them.	3	They tried to steal his book bag.
2	A group of thieves surrounded him.	4	A bystander noticed them.
5	But she continued to walk away.	5	But she continued to walk away.

Table 1: Illustration of the sentence ordering task. It aims to reorganize an unordered set of sentences into a coherent paragraph.

In recent years, several approaches based on ranking or sorting frameworks have been developed to deal with this task. RankTxNet (Kumar et al., 2020) computes a score for each sentence and sorts these scores with ranking based loss functions. Pairwise Model (Chen et al., 2016) adopts a pairwise ranking algorithm to learn the relative order of each sentence pair. B-TSort (Prabhumoye et al., 2020) predicts the constraint between two sentences and uses the topological sort technique to find the ordering.

On the other hand, to better capture the global coherence, pointer network (Vinyals et al., 2015) has been gradually used for the decoder of the ordering model. It is able to capture the paragraph-level contextual information for generating an ordered sequence with the highest coherence probability (Gong et al., 2016; Logeswaran et al., 2018; Cui et al., 2018; Yin et al., 2019). Further, HAN (Wang and Wan, 2019) and TGCM (Oh et al., 2019) introduce the attention mechanism (Vaswani et al., 2017), and FUDecoder (Yin et al., 2020) proposes pairwise ordering prediction modules to enhance the traditional pointer network.

Despite having achieved great successes, pairwise ranking and pointer network-based ordering approaches have a few problems. The former focuses on learning the local relationship between sentence pairs, but may have trouble in capturing the global interactions among all the sentences. The latter overlooks the importance of learning relative order between sentence pairs through the encoder-decoder, and lacks enough local interac-

*Corresponding author

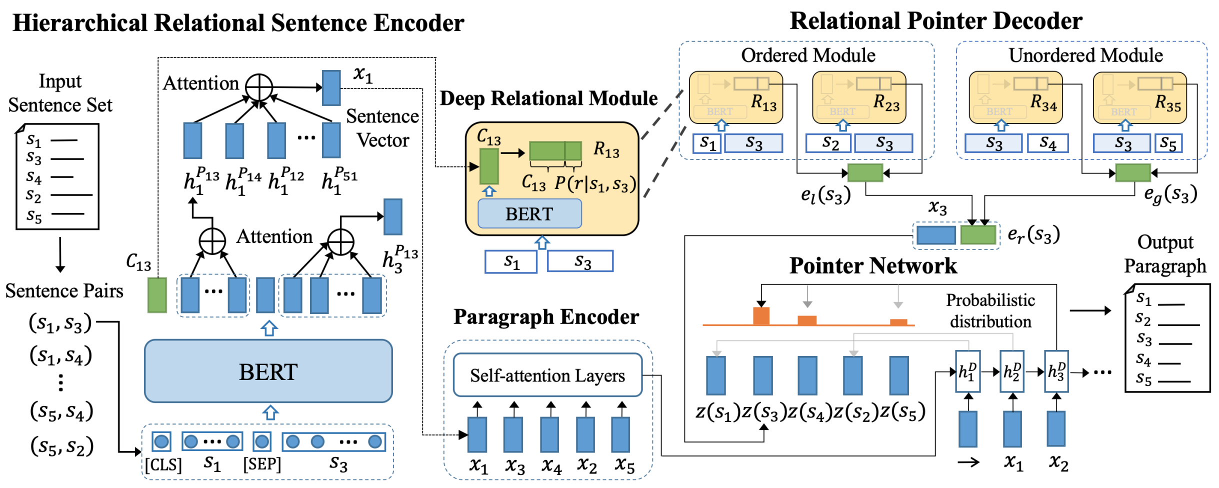

Figure 1: The architecture of the proposed BERSON. Given an unordered set of sentences, our BERT-based Hierarchical Relational Sentence Encoder first builds the high-level representation for each input sentence. Then, a self-attention based paragraph encoder is employed for paragraph encoding. Finally, the proposed Relational Pointer Decoder generates an ordered out sequence. For the sentence generation at the 3rd timestep in the decoder, s_1 and s_2 are the previous sorted sentences, and s_3, s_4, and s_5 are the unsorted ones. Here, we use the candidate sentence s_3 as an example to illustrate how to encode the relative ordering information for it in the pointer network based on Deep Relational Module. Please refer to Section 2.4 for more details of the decoder.

tions among sentences.

To address the above limitations, in this paper, we propose a novel BERT-enhanced Relational Sentence Ordering Network (referred to as BERSON) by integrating BERT (Devlin et al., 2019) with the pointer network to fully exploit the pairwise relationships between sentences for a better coherence modeling. Specifically, we first introduce a BERT-based Hierarchical Relational Sentence Encoder, which uses sentence pairs as the input to the model and learns the high-level representation for each sentence. Next, a Self-Attention based Paragraph Encoder is adopted for paragraph encoding.

Building upon the above pairwise sentence and paragraph encoding, a novel Relational Pointer Decoder (referred to as RPD) is developed by incorporating the informative relative ordering information into the pointer network with a Deep Relational Module (referred to as DRM). This module leverages the Next Sentence Prediction objective of BERT to learn the relative ordering between sentences and constructs a pairwise relationship representation for each sentence pair, which helps RPD not only exploit the global orientation information among unordered sentences but also consider the local coherence between the candidate sentence and the previously sorted ones. Thus, RPD is able to generate a more coherent order assignment for the input sentences. In addition, the pairwise ordering prediction loss is also added as the auxiliary objective to guide the coherence modeling in the training procedure. The overall architecture of our model is presented in Figure 1.

Extensive experiments are conducted on six public datasets in different domains to evaluate the performances of BERSON. The results show that BERSON significantly outperforms the existing approaches by a wide margin and achieves a state-of-the-art performance on all the datasets and under all the evaluation measurements.

2 Relational Sentence Ordering Network

In this section, we start by formulating the sentence ordering problem and then present the proposed model BERSON, which is composed of a BERT-based Hierarchical Relational Sentence Encoder, a Self-Attention based Paragraph Encoder, and a Relational Pointer Decoder enhanced by a new Deep Relational Module to model the text coherence in a more effective way.

2.1 Problem Definition

Given an out-of-order version set of N sentences $\mathbf{s} = [s_1, s_2, \cdots, s_N]$, and $s_i = [w_{i1}, w_{i2}, \cdots, w_{il_i}]$, where l_i is the number of words in sentence s_i. The model aims to recover the correct order $\mathbf{o} = [o_1, o_2, \cdots, o_N]$ for these sentences.

2.2 Hierarchical Relational Sentence Encoder

The sentence encoder is designed based on BERT with sentence pairs in the set as input, and further adopts two-level attention layers to encode the hierarchical semantic concepts and contextual information of the sentence.

Formally, for the given N sentences in the set, all the pair of sentences can be denoted as:

$$P = \{P_{ij} | i \in [1, N], j \in [1, N], i \neq j\} \quad (1)$$

where P_{ij} represents the sentence pair (s_i, s_j). The total number of sentence pairs is $|P| = A_N^2$. These sentence pairs are sent into BERT to not only learn the sentence representation but also capture the pairwise relationship between sentences.

As shown in the left part of Figure 1, given a sentence pair $P_{ij} = (s_i, s_j)$, the input sequence of this pair to the BERT model consists of a [CLS] token, the first sentence s_i in the pair, a separator token [SEP], and the second sentence s_j. The BERT model encodes the representation for this pair as:

$$\left\{C_{ij}, h_{i1}^{P_{ij}}, \cdots, h_{il_i}^{P_{ij}}, S_{ij}, h_{j1}^{P_{ij}}, \cdots, h_{jl_j}^{P_{ij}}\right\} \quad (2)$$

where C_{ij} and S_{ij} are the final hidden states of the [CLS] and [SEP] tokens, and $\left\{h_{i1}^{P_{ij}}, \cdots, h_{il_i}^{P_{ij}}\right\}$ and $\left\{h_{j1}^{P_{ij}}, \cdots, h_{jl_j}^{P_{ij}}\right\}$ are the output word representations of sentence s_i and s_j in this pair with the sequence length l_i and l_j respectively.

After the BERT encoder, we compose a fixed-dimensional representation for each sentence.

For sentence s_i in pair P_{ij}, the representations $\left\{h_{i1}^{P_{ij}}, \cdots, h_{il_i}^{P_{ij}}\right\}$ of each word are combined together with an attention mechanism to obtain its sentence representation $h_i^{P_{ij}}$:

$$u_{ik} = \tanh\left(W_w h_{ik}^{P_{ij}}\right), \alpha_{ik} = \frac{\exp(v_w u_{ik})}{\sum_{k=1}^{l_i} \exp(v_w u_{ik})}$$

$$h_i^{P_{ij}} = \sum\nolimits_{k=1}^{l_i} \alpha_{ik} h_{ik}^{P_{ij}} \quad (3)$$

where W_w and v_w are learnable parameters. Attention allows the model to concentrate on the informative words for coherence and helps build a better semantic representation. Similarly, we also compute the representation $h_j^{P_{ij}}$ for s_j in pair P_{ij}.

Further, all the sentence pairs related to sentence s_i can be described as: $P_i = \{P_{ij} | j \in [1, N], j \neq i\} \cup \{P_{ki} | k \in [1, N], k \neq i\}$. The number of pairs is $2N - 2$. The corresponding sentence representations of s_i obtained from these pairs are given as: $h_i = \left\{h_i^{P_{ij}} | j \in [1, N], j \neq i\right\} \cup \left\{h_i^{P_{ki}} | k \in [1, N], k \neq i\right\}$. We denote $h_i = \left\{h_i^1, \cdots, h_i^{2N-2}\right\}$ for simplification.

Since the sentence embeddings of s_i in different pairs capture different context features, to reward the most salient features that contribute highly to the overall contextual meaning of the sentence, a high-level attention mechanism is adopted to establish the final representation x_i for sentence s_i:

$$u_i^t = \tanh\left(W_s h_i^t\right), \alpha_i^t = \frac{\exp(v_s u_i^t)}{\sum_{t=1}^{2N-2} \exp(v_s u_i^t)}$$

$$x_i = \sum\nolimits_{t=1}^{2N-2} \alpha_i^t h_i^t \quad (4)$$

where W_s and v_s are also trainable weights. Essentially, as all the related sentence pairs are fairly considered, it is ensured that this representation being invariant to the input sentence order and being logically reliable to be used in our model.

2.3 Paragraph Encoder

After the sentence encoder, a self-attention based paragraph encoder is employed to capture the global dependency for all the sentences.

Specifically, the sentence representations obtained from the sentence encoder are packed together into a paragraph matrix $X = [x_1, \cdots, x_N]$ as $X^{(1)}$, which is then sent to L self-attention layers (Vaswani et al., 2017). For the l-th layer, the output matrix $X^{(l)}$ is computed as:

$$\widetilde{X}^{(l)} = \text{LN}(X^{(l-1)} + \text{MultiHead}(X^{(l-1)})) \quad (5)$$

$$X^{(l)} = \text{LN}(\widetilde{X}^{(l)} + \text{FFN}(\widetilde{X}^{(l)})) \quad (6)$$

where MultiHead$(\cdot)$ is multi-head attention function, FFN$(\cdot)$ denotes the fully-connected feed-forward network, and LN$(\cdot)$ is the layer normalization operation (Ba et al., 2016).

The final paragraph vector m is generated by averaging the output matrix $X^{(L)}$ from the last self-attention layer: $m = \frac{1}{N} \sum_{n=1}^{N} X_n^{(L)}$, where $X_n^{(L)}$ is the n-th row in $X^{(L)}$. This vector will then be used as the initial state of our decoder.

2.4 Relational Pointer Decoder

In this section, we propose a Relational Pointer Decoder (RPD), which utilizes the useful relative ordering information to enhance pointer network with a Deep Relational Module (DRM). In the following, we first describe the new module DRM and then incorporate it into the pointer network to strengthen the coherence modeling in the decoder.

2.4.1 Deep Relational Module

Our Deep Relational Module is based on BERT model, which aims to capture a better dependency relationship between sentences. The architecture of this module is shown in the middle part of Figure 1.

In particular, as illustrated in Section 2.2, given the sentence pair P_{ij}, the embedding of the [CLS] symbol from the top layer of BERT is denoted as C_{ij}. Owing to the Next Sentence Prediction pretraining objective of BERT, this vector C_{ij} is able to aggregate the semantic relations for the input sentence pair and is capable of identifying the relative order between two sentences. Therefore, we take full advantages of this vector to exploit the latent dependency for sentences.

Further, a probability distribution $P(r|s_i, s_j)$ is generated, $r \in \{\text{before}, \text{after}\}$, which measures the probability of s_i occuring before or after s_j:

$$P(r|s_i, s_j) = \text{softmax}(W_c C_{ij}) \tag{7}$$

where W_c denotes the learnable weights.

In order to obtain the richer pairwise relation information for the sentence pair, we combine the above semantic feature C_{ij} and the probability distribution together:

$$R_{ij} = \left[C_{ij}; P(r|s_i, s_j)\right] \tag{8}$$

This new vector R_{ij} is considered as the relational representation for this sentence pair (s_i, s_j), which is then leveraged to provide the relative order information for the pointer network. We compute such pairwise relational representation for all the sentence pairs in the paragraph, and utilize the subset of them at each step of the decoder.

Different from the previous method of using the learned sentence vectors to calculate the pairwise relationship between sentences (Yin et al., 2020), DRM employs the whole sequence of the sentence pair as the input to BERT. It allows us to directly relate words from different sentences together, which is more straightforward to exploit the intrinsic relations and coherence between sentences. Further, instead of relying on the modules trained from scratch to control the pairwise ordering predictions (Yin et al., 2020), DRM adopts BERT as the main building block to obtain a pairwise relationship representation for the sentence pair. Intuitively, being pre-trained on the large corpus in BERT, this representation encodes more reliable and accurate relative ordering information, and thus is more effective to help determine the pairwise ordering predictions in the decoder.

2.4.2 Integrating DRM with Pointer Network

As illustrated in the right part of Figure 1, Relational Pointer Decoder (RPD) incorporates Deep Relational Module into the pointer network to promote the coherence modeling among sentences.

Formally, the conditional coherence probability of a predicted order $\widehat{\mathbf{o}}$ for the given out-of-order sentence set $\mathbf{s}$ can be computed as:

$$P(\widehat{\mathbf{o}}|\mathbf{s}) = \prod\nolimits_{i=1}^{N} P(\widehat{o}_i|\widehat{\mathbf{o}}_{<i}, \mathbf{s}) \tag{9}$$

A higher probability indicates a more coherent sentences assignment. We employ an LSTM-based pointer network as the basis of our decoder, and the mathematical formulation for the i-th step in the decoder is:

$$h_i^D = \text{LSTM}(h_{i-1}^D, x_{\widehat{o}_{i-1}}) \tag{10}$$
$$P(\widehat{o}_i|\widehat{\mathbf{o}}_{<i}, \mathbf{s}) = \text{softmax}(g^T \tanh(W_q h_i^D + W_k Z_i))$$

where g, W_q, and W_k are all learnable parameters, h_i^D is the hidden state in the decoder with size d, $h_0^D = m$, and $x_{\widehat{o}_{i-1}}$ is the embedding of the previous predicted sentence $s_{\widehat{o}_{i-1}}$ at step $i-1$. The softmax function produces an output distribution over all unordered sentences (candidate sentences). The one that yields the highest probability from the distribution will be selected at position i.

The matrix Z_i encodes the relationship representation information of the candidate sentence with the other sentences in the set. For one candidate sentence, the other sentences can be divided into two groups: previously sorted subset and unsorted subset. The relative ordering information between the candidate sentence and the two group sentences are captured by the proposed DRM with its two versions: *Ordered Module* and *Unordered Module*, respectively. On the one hand, such modeling helps evaluate the local coherence between the previously sorted sentences and the candidate sentence for investigating the rationality of each candidate choice. On the other hand, the global relative orientation information of other unsorted sentences with respect to the candidate one also provides further clues for the current prediction. Thus, both the local dependency information and the global orientation are fully exploited in RPD.

For the Ordered Module, the pairwise relationships between the predicted sentence $s_{\widehat{o}_{i-1}}$ at step $i-1$ and the candidate sentence s_c can be effectively measured by our deep relational module with the following relational representation:

$$R_{\widehat{o}_{i-1}c} = \left[C_{\widehat{o}_{i-1}c}; P(r|s_{\widehat{o}_{i-1}}, s_c)\right] \tag{11}$$

which not only encodes the semantic relations between two sentences, but also includes the probability of whether sentence s_c truly appears after $s_{\widehat{o}_{i-1}}$ or not. In similar ways, the relational embedding generated with the previous ordered sentences can be described as $\left\{R_{\widehat{o}_1c}, \cdots, R_{\widehat{o}_{i-1}c}\right\}$. Then, we compose a high-level local coherence representation $e_l(s_c)$ for this candidate sentence by integrating these relational embeddings to summarize the overall local dependency for s_c.

For the Unordered Module, the relative orientation of another unordered sentence s_g with respect to the candidate sentence s_c can also be captured by our relational embedding as:

$$R_{cg} = \left[C_{cg}; P(r|s_c, s_g)\right] \tag{12}$$

Considering all the other unsorted sentences, a hierarchical global orientation representation $e_g(s_c)$ for s_c can be obtained, which is formulated as:

$$e_g(s_c) = \frac{1}{|S_c|}\sum\nolimits_{s_g \in S_c} R_{cg} \tag{13}$$

where S_c is the unordered sentence set except s_c.

Subsequently, to leverage the relative ordering information encoded by Ordered and Unordered Modules simultaneously, the representation $e_l(s_c)$ and $e_g(s_c)$ are integrated together, which allows us to build a more informative relational vector $e_r(s_c)$ for sentence s_c. Finally, a new representation for this candidate sentence s_c is obtained by combining its sentence embedding and relational vector $e_r(s_c)$ together:

$$z(s_c) = [x_c; e_r(s_c)] \tag{14}$$

Such representation is generated for all unsorted sentences, which are then packed into matrix Z_i for order predictions. During inference, we use beam search to select sentences sequentially.

2.5 Model Training

Assume that there are Q paragraphs in the training set $Q = \{(\mathbf{s}, \mathbf{o})\}$. Following the existing ordering networks (Gong et al., 2016; Oh et al., 2019), the model is trained to maximize the coherence probability by minimizing the loss function as follows:

Dataset	Length statistics		Data split			Vocabulary
	mean	max	train	valid	test	
NIPS abstract	6	15	2427	408	377	11505
AAN abstract	5	20	8569	962	2626	34485
NSF abstract	8.9	40	96070	10185	21580	334090
arXiv abstract	5.38	35	884912	110614	110615	64557
SIND	5	5	40155	4990	5055	30861
ROCStory	5	5	78529	9816	9817	33903

Table 2: Summary of datasets used in our experiments.

$$L_c = -\frac{1}{|Q|}\sum\nolimits_{(\mathbf{s},\mathbf{o})\in Q} \log P(\mathbf{o}|\mathbf{s};\theta) + \frac{\lambda}{2}\|\theta\|_2^2 \tag{15}$$

where θ denotes all the trainable parameters.

To further exploit the correct relative order information, we add the Pairwise Ordering Prediction Loss (Ploss) as an auxiliary objective L_p. It is defined as the cross-entropy loss function optimized by minimizing the negative log-likelihood of each pair's ground-truth relative ordering label $y_{ij} \in [0, 1]$, given the networks prediction $\widehat{y}_{ij}$:

$$\begin{aligned} L_p = -\frac{1}{|Q|}\frac{1}{|P|}\sum_{(\mathbf{s},\mathbf{o})\in Q}\sum_{P_{ij}\in P} & (-y_{ij}\log P(\widehat{y}_{ij}|P_{ij}) \\ & -(1-y_{ij})P(1-\widehat{y}_{ij}|P_{ij})) \end{aligned}$$

The final training objective of our model can be formulated as:

$$L = L_C + \alpha L_p \tag{16}$$

where α is the coefficient that makes a balance between the influences of the two loss functions.

3 Experiments

In this section, we empirically evaluate the effectiveness of BERSON in the sentence ordering task.

3.1 Datasets

The experiments are conducted on six public datasets in different domains:

NIPS abstract, AAN abstract, NSF abstract, arXiv abstract: These datasets contain abstracts of research papers. NIPS abstract[1] is from conference papers in NIPS, where papers in years 2005-2013/2014/2015 for training/validation/testing (Logeswaran et al., 2018). AAN abstract (Logeswaran et al., 2018) is collected from ACL Anthology Network corpus. ACL papers published up to year 2010 for training, year 2011 for validation and 2012-2013 for testing. NSF abstract (Logeswaran et al., 2018) is from NSF Research Award abstract dataset, where abstracts in years 1990-1999/2000/2001-2003 for

[1]https://github.com/DeepLearnXMU/NSEG

Dataset	learning rate	batch size	epochs	coefficient α
NIPS abstract	5e-5	8	20	0.2
AAN abstract	5e-5	32	10	0.4
NSF abstract	2e-5	16	10	0.4
arXiv abstract	2e-5	32	10	0.8
SIND	2e-5	8	10	1.0
ROCStory	2e-5	16	10	0.6

Table 3: Hyper-parameter settings of our model on each dataset.

training/validation/testing. ArXiv abstract (Gong et al., 2016; Chen et al., 2016) is from arXiv website[2]. The validation and test sets of this dataset are the first and last 10% abstracts from the shuffled data, and the remaining data are for training.

SIND, ROCStory: SIND is a visual storytelling dataset[3] (Huang et al., 2016), which is released as training/validation/testing following the 8:1:1 split. ROCStory is a commonsense story dataset[4] (Wang and Wan, 2019; Mostafazadeh et al., 2016). It is randomly split by 8:1:1 for the training/validation/test sets. Both of two datasets consist of 5 sentences in each story text.

Table 2 shows the details of all the datasets.

3.2 Evaluation Metrics

Following the existing work (Oh et al., 2019), we employ the three most commonly used metrics[5] in this task to assess the model performance:

Accuracy (Acc): This metric calculates the ratio of sentences whose absolute positions are correctly predicted (Logeswaran et al., 2018).

Perfect Match Ratio (PMR): It measures the percentage of the exactly matching orders across all the paragraphs: PMR=$\frac{1}{Q}\sum_{i=1}^{Q}\mathbb{1}(\widehat{\mathbf{o}}^i = \mathbf{o}^i)$, where $\widehat{\mathbf{o}}^i$ and $\mathbf{o}^i$ are the predicted and correct order of the i-th paragraph respectively (Chen et al., 2016).

Kendall's tau (τ): For a paragraph containing N sentences, τ is defined as: $\tau = 1 - 2\times$ (# *inversions*) $/\binom{N}{2}$, where # *inversions* denotes the number of pairs in the predicted sequence with the incorrect relative order (Lapata, 2003). The score ranges from -1 (the worst) to 1 (the best).

A higher score indicates a better performance for all the metrics.

3.3 Experimental Setup

We adopt the BERT$_{\text{BASE}}$ in the experiment and fine-tune it on each dataset. The paragraph encoder has 2 self-attention layers with 8 heads. The hidden size is 768 and beam size is 16. Adam is employed as the optimizer. To search for the optimal hyper-parameters, we adopt the grid search strategy for learning rate from {2e-5, 5e-5}, batch size from $\{8, 16, 32\}$, the number of epochs from $\{5, 10, 20\}$, and the coefficient α in the loss function from $\{0.2, 0.4, 0.6, 0.8, 1.0\}$. The model with the best performance on the validation set is selected for each setting. The recommended hyper-parameter configuration of the model on each dataset are presented in Table 3. To diminish the effects of randomness in training, the results of our model are averaged with 5 random initializations. For data preprocessing, we use the tokenizer[6] from BERT to preprocess the sentences. The experiments are conducted on GeForce GTX 1080Ti GPU with PyTorch framework .

3.4 Baselines

To demonstrate that BERSON truly improves the sentence ordering performance, we compare it with the state-of-the-art methods in this task, which can be categorized into two classes:

(1) Ranking or Sorting frameworks: Pairwise Model (Chen et al., 2016); RankTxNet (Kumar et al., 2020); B-TSort (Prabhumoye et al., 2020).

(2) Pointer network based models: HAN (Wang and Wan, 2019); LSTM+PtrNet (Gong et al., 2016); V-LSTM+PtrNet (Logeswaran et al., 2018); ATTOrderNet (Cui et al., 2018); SE-Graph (Yin et al., 2019); FUDecoder (Yin et al., 2020); TGCM (Oh et al., 2019).

In addition to the above existing approaches, we also investigate three variants of BERSON.

BertSenPD: This model replaces the ranking module in RankTxNet with the traditional pointer network decoder (PD). Please note that it uses the single sentence rather than sentence pair as the input to BERT to obtain the sentence vector.

BertPairPD, HRSEPD: These two models employ the sentence pair encoding strategy with BERT and utilize PD instead of our RPD as the decoder. HRSEPD adopts the proposed Hierarchical Relational Sentence Encoder (HRSE), while BertPairPD does not have two-level attention layers in the encoder. They aim to investigate the impact of both the Hierarchical Relational Sentence Encoder and the Relational Pointer Decoder.

[2]https://github.com/FudanNLP/NeuralSentenceOrdering

[3]http://visionandlanguage.net/VIST/dataset.html

[4]https://github.com/sodawater/SentenceOrdering

[5]Code for metrics: https://github.com/DeepLearnXMU/NSEG

[6]https://github.com/google-research/bert

Models	NIPS abstract			AAN abstract			NSF abstract			arXiv abstract			SIND			ROCStory		
	Acc	PMR	τ	Acc	PMR	τ	Acc	PMR	τ	Acc	PMR	τ	Acc	PMR	τ	Acc	PMR	τ
Pairwise Model	-	-	-	-	-	-	-	-	-	-	33.43	0.66	-	-	-	-	-	-
LSTM+PtrNet	50.87	-	0.67	58.20	-	0.69	32.45	-	0.52	-	40.44	0.72	-	12.34	0.48	-	-	-
V-LSTM+PtrNet	51.55	-	0.72	58.06	-	0.73	28.33	-	0.51	-	-	-	-	-	-	-	-	-
ATTOrderNet	56.09	-	0.72	63.24	-	0.73	37.72	-	0.55	-	42.19	0.73	-	14.01	0.49	-	-	-
HAN	-	-	-	-	-	-	-	-	-	-	44.55	0.75	-	15.01	0.50	-	39.62	0.73
SE-Graph	57.27	-	0.75	64.64	-	0.78	-	-	-	-	44.33	0.75	-	16.22	0.52	-	-	-
FUDecoder	-	-	-	-	-	-	-	-	-	-	46.58	0.77	-	17.37	0.53	-	46.00	0.77
TGCM	59.43	31.44	0.75	65.16	36.69	0.75	42.67	22.35	0.55	58.31	44.28	0.75	38.71	15.18	0.53	-	-	-
RankTxNet	-	24.13	0.75	-	39.18	0.77	-	9.78	0.58	-	43.44	0.77	-	15.48	0.57	-	38.02	0.76
B-TSort	61.48	32.59	0.81	69.22	50.76	0.83	35.21	10.44	0.66	-	-	-	52.23	20.32	0.60	-	-	-
BertSenPD	64.36	31.30	0.79	70.34	45.45	0.80	45.76	17.41	0.64	69.72	46.26	0.78	52.12	19.19	0.58	75.05	52.56	0.81
BertPairPD	67.65	32.89	0.81	73.99	50.53	0.83	46.88	18.76	0.65	71.03	48.98	0.79	54.36	23.86	0.60	78.01	60.39	0.83
HRSEPD	67.99	35.54	0.83	75.45	53.27	0.84	47.01	19.02	0.65	71.65	49.91	0.80	56.22	25.22	0.63	79.81	61.93	0.84
BERSON	**73.87**	**48.01**	**0.85**	**78.03**	**59.79**	**0.85**	**50.02**	**23.07**	**0.67**	**75.08**	**56.06**	**0.83**	**58.91**	**31.69**	**0.65**	**82.86**	**68.23**	**0.88**

Table 4: Comparison results for different models on sentence ordering task. The best and second-best results are in bold and underlined respectively.

3.5 Main Results

The experimental results[7] are reported in Table 4. As we see, BERSON achieves the state-of-the-art-performance on all the datasets and under all the evaluation metrics.

The results show that BERSON significantly outperforms all the existing methods by a large margin. BERSON shows remarkable improvements over the existing best systems of 12.39% and 16.77% accuracy score on NIPS and arXiv datasets, and with 15.42%, 11.37%, and even 22.23% gains in PMR score on NIPS, SIND, and ROCStory datasets respectively, which strongly demonstrates the effectiveness of our model[8].

Compared with the existing ranking approaches, our BertSenPD baseline performs much better than RankTxNet with stable improvements, which confirms the superiority of the traditional pointer network to the ranking module used in their model. This could be due to that RankTxNet only computes a score for each sentence in parallel, which overlooks the coherence of the whole predicted sequence and may have trouble in generating a more coherent order assignment. Besides, although B-TSort outperforms RankTxNet with clear improvements, it only considers the sentence-pair interactions and does not take the entire paragraph into account. Therefore, B-TSort is limited by the lack of a global structure and falls behind other baselines for Acc and PMR scores on the large dataset NSF abstract. In contrast, BERSON not only captures the local coherence between every two sentences but also obtains the paragraph-level contextual information for the global dependency, hence being more competitive in the sentence ordering.

In addition, among the pointer network based ordering models, FUDecoder exhibits a better performance. However, the ordering prediction modules of FUDecoder are built based on two non-linear layers trained from scratch and with the learned sentence vectors as the input, which is still difficult to fully explore the latent dependency among sentences. Once these modules are not sufficiently trained especially on small datasets, they may mislead the decoder with the wrong relative orientation information. Our BERSON overcomes the limitation of FUDecoder by utilizing BERT model as the main building block of our DRM to improve the pairwise ordering strategy. As shown in Table 4, BERSON achieves to outperform FUDecoder with significant improvements of about 14.32% and 22.23% PMR score on SIND and ROCStory respectively, which proves the promise of incorporating more reliable ordering module into the decoder to ensure the more accurate relative ordering information.

Moreover, for the variants of our model, BertPairPD and HRSEPD perform better than BertSenPD on all the datasets. This shows that with the sentence pair instead of single sentence as the input to BERT, the model directly builds the inter-

[7]The validation results, average runtime, and the number of parameters are reported in Appendix.

[8]Ordering prediction examples from BERSON and several baselines are also shown in Appendix.

Models	arXiv abstract			ROCStory		
	Acc	PMR	τ	Acc	PMR	τ
BertPairPD	71.03	48.98	0.79	78.01	60.39	0.83
HRSEPD	71.65	49.91	0.80	79.81	61.93	0.84
BERSON	75.08	56.06	0.83	82.86	68.23	0.88
- Ordered Module	73.62	54.51	0.82	81.14	66.61	0.86
- Unordered Module	72.57	50.49	0.80	80.35	62.47	0.85
- Ploss	73.99	54.89	0.82	81.63	66.92	0.87

Table 5: Ablation studies on arXiv and ROCStory datasets. We remove various modules and explore their influences to our model.

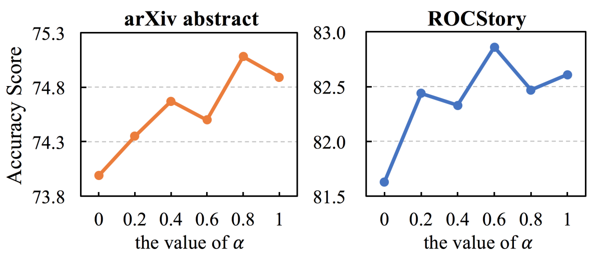

Figure 2: Results of varying the coefficient α for Pairwise Ordering Prediction Loss in our model.

actions between words from different sentences, which is capable of capturing the rich contextual information for each sentence and is more beneficial to modeling the relations among sentences. Besides, HRSEPD outperforms BertPairPD with stable improvements, which reflects the strength of our Hierarchical Relational Sentence Encoder. Furthermore, by adopting our Relational Pointer Decoder to replace the traditional pointer network, BERSON achieves further improvements across the datasets, which demonstrates the advantage of enhancing the pointer network with DRM to reach a superior performance.

3.6 Ablation Study

Further, to better understand the contributions of different components in our Relational Pointer Decoder, we conduct ablation study on arXiv and ROCStory datasets, which are both the largest datasets in the two domains for providing more reliable analysis. The results are reported in Table 5.

Effect of Ordered and Unordered Modules: It is observed that the removal of two modules hurts the model performance dramatically though they still outperform our baseline models BertPairPD and HRSEPD. Compared with Ordered Module, the lack of Unordered Module leads noticeable drops, which indicates that the relative orientations between unsorted sentences are more important for

Models	arXiv abstract		SIND	
	head	tail	head	tail
Pairwise Model	84.85	62.37	-	-
LSTM+PtrNet	90.47	66.49	74.66	53.30
ATTOrderNet	91.00	68.08	76.00	54.42
SE-Graph	92.28	70.45	78.12	56.68
FUDecoder	92.76	<u>71.49</u>	78.08	57.32
TGCM	92.46	69.45	78.98	56.24
RankTxNet	<u>92.97</u>	69.13	<u>80.32</u>	<u>59.68</u>
BertSenPD	93.38	72.57	81.46	61.02
BertPairPD	94.01	73.99	82.37	62.24
BERSON	**94.75**	**76.69**	**84.95**	**64.87**

Table 6: Accuracy of predicting the first and the last sentences on arXiv and SIND datasets.

order predictions. The superior performance of BERSON over these two variants shows the necessity of having both modules in RPD to leverage the global orientation and local dependency information simultaneously for a better coherence model.

Effect of Pairwise Ordering Prediction Loss: As shown in Table 5, removing the Pairwise Ordering Prediction Loss (Ploss) in the training procedure causes a performance degradation on both datasets. This proves the benefit of encouraging the accurate relative ordering information through the loss function. As the coefficient α in Equation 16 directly controls the impact of Ploss, we further study how the value of this coefficient affects the performance of BERSON. Figure 3 shows the results of accuracy score on arXiv and ROCStory datasets. It is shown that α = 0.8, 0.6 is superior to other settings for arXiv and ROCStory respectively. Thus, it is essential to have an appropriate value to balance the importance of Ploss and the original training objective for BERSON.

3.7 Analysis

In this section, we delve into further analysis to investigate the stability and adaptability of the proposed model.

3.7.1 Prediction of First and Last Sentences

Previous studies (Oh et al., 2019; Yin et al., 2019) have mentioned that both the first and last sentences play crucial roles in a paragraph due to their special positions. Thus, we also report the performances of our models in correctly predicting these two sentences on arXiv and SIND datasets. As summarized in Table 6, both of the two variants BertSenPD and BertPairPD outperform the existing state-of-the-arts. BERSON achieves fur-

Models	Win=1	Win=2	Win=3	Win=1	Win=2	Win=3
	NIPS abstract			SIND		
B-TSort	87.59	95.59	98.11	82.67	95.01	99.09
BERSON	**91.98**	**96.84**	**99.29**	**84.01**	**95.11**	**99.10**
	NSF abstract			AAN abstract		
B-TSort	61.41	75.52	83.87	90.56	96.78	98.71
BERSON	**70.08**	**80.29**	**86.46**	**93.31**	**97.57**	**99.09**

Table 7: Analysis of the displacement of sentences on four datasets. Win denotes the Window size.

Models	Acc	PMR	τ	LCS	Rouge-S
	NIPS abstract				
B-TSort	39.43	0.00	0.74	71.68	83.26
BertPairPD	43.75	6.67	0.75	72.59	87.44
BERSON	**50.89**	**13.33**	**0.76**	**74.96**	**87.94**
	AAN abstract				
B-TSort	36.86	0.00	0.69	72.01	78.52
BertPairPD	40.75	3.45	0.71	73.87	85.77
BERSON	**47.38**	**6.90**	**0.73**	**76.53**	**86.48**

Table 8: Performance for paragraphs containing more than 10 sentences in NIPS and AAN datasets.

ther boosts and reaches the best performances on both datasets. For identifying the last sentence, BERSON obtains significant improvements over RankTxNet of 7.56% and 5.19% gain on arXiv and SIND datasets respectively, which also indicates the benefits of the proposed model.

3.7.2 Sentence Displacement Analysis

Additionally, we analyze the displacement of sentences in the predicted orders by calculating the percentage of sentences whose predicted location is within one, two or three positions from their original location (Prabhumoye et al., 2020). The higher score is better, which denotes less displacement of sentences. As summarized in Table 7, BERSON also achieves a better performance than B-TSort across the datasets and on all the window sizes especially for the smaller ones. BERSON even reaches 99% percent when the window size is 3 on NIPS, AAN, and SIND datasets, which clearly demonstrates the promise of BERSON.

3.7.3 Performance on Longer Paragraphs

Following the prior approach (Prabhumoye et al., 2020), we also evaluate the model performance on paragraphs longer than 10 sentences, which are much challenging for the order prediction. In addition to the three metrics adopted in the previous sections, here we also utilize two other metrics: Longest Common Subsequence (LCS) and Rouge-S for a more comprehensive comparison[9]. Table 8 reports the results on NIPS and AAN datasets. BERSON significantly outperforms B-TSort with all the metrics, showing more than 10% gain in accuracy score on both datasets. Besides, the results of PMR score indicate that it is difficult for B-TSort to exactly match orders for all the sentences, while our BERSON consistently shows a good potential on these longer paragraphs, which proves the stronger ability of BERSON in modeling the long-range dependency across the sentences.

4 Conclusion

In this work, we develop a new BERT-enhanced Relational Sentence Ordering Network (BERSON) by integrating BERT with the pointer network for a better coherence modeling. In particular, a novel Relational Pointer Decoder is developed to incorporate the relative ordering information into the pointer network with a Deep Relational Module, which leverages BERT to fully exploit the pairwise relationships between sentences helping generate an ordered sequence. The experiments on six datasets demonstrate the superiority of BERSON to the baselines, which achieves the state-of-the-art performance across the datasets.

Acknowledgments

This work is supported in part by Science and Technology Innovation 2030-"New Generation Artificial Intelligenc" Major Project No.(2018AAA0100904), National Key R&D Program of China (No. 2018YFB1403600), NSFC (No. 61672456, 61702448, U19B2043), Artificial Intelligence Research Foundation of Baidu Inc., the funding from HIKVision and Horizon Robotics, and ZJU Converging Media Computing Lab. We thank all reviewers for their valuable comments.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

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

[9]The explanation of two metrics are illustrated in the Appendix.

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

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

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

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

Youmna Farag and Helen Yannakoudakis. 2019. Multi-task learning for coherence modeling. In *ACL*, pages 629–639.

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

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 Kenneth Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Aishwarya Agrawal, Jacob Devlin, Ross Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, et al. 2016. Visual storytelling. In *NAACL*, pages 1233–1239.

Ioannis Konstas and Mirella Lapata. 2013. Inducing document plans for concept-to-text generation. In *EMNLP*, pages 1503–1514.

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. Association for Computational Linguistics.

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

Mohsen Mesgar, Sebastian Bücker, and Iryna Gurevych. 2019. A neural model for dialogue coherence assessment. *arXiv preprint arXiv:1908.08486*.

Han Cheol Moon, Muhammad Tasnim Mohiuddin, Shafiq Joty, and Chi Xu. 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. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*, pages 3075–3081.

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, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 6000–6010.

Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. 2015. Order matters: Sequence to sequence for sets. *arXiv preprint arXiv:1511.06391*.

Tianming Wang and Xiaojun Wan. 2019. Hierarchical attention networks for sentence ordering. In *AAAI*, volume 33, pages 7184–7191.

Peng Xu, Hamidreza Saghir, Jin Sung Kang, Teng Long, Avishek Joey Bose, Yanshuai Cao, and Jackie Chi Kit Cheung. 2019. A cross-domain transferable neural coherence model. In *ACL*, pages 678–687.

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.

Xingshan Zeng, Jing Li, Lu Wang, Nicholas Beauchamp, Sarah Shugars, and Kam-Fai Wong. 2018. Microblog conversation recommendation via joint modeling of topics and discourse. In *NAACL*, pages 375–385.

A Appendix

A.1 Case Study

Table 9 reports the sentence ordering results for two examples produced by different models. For the baseline FUDecoder and B-TSort, we ran the public code provided by the authors to generate the order predictions. As we see, in the first example, BERSON achieves to exact match orders for all the sentences while all the baseline methods have some incorrect order predictions. For the second input paragraph, BERSON is able to correctly predict the order for the most of sentences which also shows a better performance than the competing models.

A.2 Two other metrics used in the Analysis

Longest Common Subsequence (LCS): It calculates the percentage of longest correct subsequence between the predicted order and the gold order (Gong et al., 2016). The consecutiveness is not necessary for it[10].

Rouge-S: This metric (Chen et al., 2016; Gong et al., 2016) measures the fraction of pairs of sentences whose predicted relative order is the same as the ground truth order[11]. It allows for any arbitrary gaps between two sentences as long as their relative order is correctly identified.

A higher score is better for both metrics.

A.3 Discussion of Topic Shift Problem

BERSON captures both global and local coherence among sentences, which is effective in reorganizing texts with multiple topics. In particular, the paragraph encoder is able to model the global topic information for all the sentences, which helps guide the order prediction process for the decoder. In addition, building upon the Next Sentence Prediction pre-training objective of BERT, the Deep Relational Module captures the local dependency relationship between each pair of sentences and identifies the tight semantic connections for sentence ordering, especially identifying the sentences containing topic shift clues of the whole text and acting as a link between the preceding and the following topics. Further, the Relational Pointer Decoder leverages both the topical context flows from the previously predicted sequences and from the unsorted sentences to generate an accurate order prediction for these topic-linking sentences and their neighboring ones. Therefore, BERSON is capable of generating a logically consistent output sequence for texts including texts with topic shift.

[10]Codes for metric: https://github.com/shrimai/Topological-Sort-for-Sentence-Ordering

[11]Code for metric: https://github.com/DeepLearnXMU/NSEG

Example 1	
(4) The reception began with the bride and groom dancing. (2) The bride and groom wrote their own wedding vows. (5) Then, family pictures were taken with the bride and groom. (1) We gathered at the church to celebrate the marriage. (3) They make a handsome couple.	
Ground Truth	(1) (2) (3) (4) (5)
FUDecoder	(1) (2) (5) (4) (3)
B-TSort	(2) (4) (3) (1) (5)
BertPairPD	(1) (3) (2) (4) (5)
BERSON	(1) (2) (3) (4) (5)
Example 2	
(4) The first one extracts relevant noun phrases as a heading. (6) Finally, the last one uses nominalization to propose titles. (3) Our application relies on three different titling methods. (1) This paper deals with an application of automatic titling. (5) And the second one selects words appearing in the text. (2) It aims to attribute a title for a given text. (7) Experiments show that our methods provide relevant titles.	
Ground Truth	(1) (2) (3) (4) (5) (6) (7)
FUDecoder	(2) (1) (3) (4) (6) (5) (7)
B-TSort	(1) (2) (4) (5) (6) (3) (7)
BertPairPD	(1) (2) (3) (4) (6) (7) (5)
BERSON	(1) (2) (3) (4) (5) (7) (6)

Table 9: Ordering prediction examples generated by different approaches. The prediction in blue indicates the wrong order assignment.

Dataset	BERSON		BertPairPD	
	Runtime	# Params	Runtime	# Params
NIPS abstract	58s	129M	47s	127M
AAN abstract	1min26s	129M	1min15s	127M
NSF abstract	57min59s	129M	48min40s	127M
arXiv abstract	2h44min23s	129M	2h31min10s	127M
SIND	3min33s	129M	2min58s	127M
ROCStory	6min31s	129M	5min28s	127M

Table 10: The runtime on the validation set and the total number of parameters for BERSON and baseline model BertPairPD.

A.4 Further Experimental Results

For the more detailed experimental results, Table 10 summarizes the runtime on the validation set and the number of parameters for BERSON and BertPairPD. The validation results of BERSON on all the datasets are reported in Table 11.

Dataset	Validation Results		
	Acc	PMR	τ
NIPS abstract	68.65	39.46	0.82
AAN abstract	76.75	58.84	0.85
NSF abstract	51.71	23.39	0.68
arXiv abstract	74.84	55.75	0.82
SIND	58.84	31.70	0.65
ROCStory	82.55	67.63	0.87

Table 11: The validation performance of BERSON.