Using Conditional Sentence Representation in Pointer Networks for Sentence Ordering

Farhood Farahnak Leila Kosseim
Computational Linguistics at Concordia (CLaC) Laboratory
Department of Computer Science and Software Engineering
Concordia University, Montréal, Québec, Canada
Email: firstname.lastname@concordia.ca

***Abstract*—Sentence ordering aims to arrange sentences in a coherent manner and hence has important applications in Natural Language Generation. Recently, several approaches have used Pointer Networks for this task. Such networks arrange a list of sentences based on fixed sentence representations, where these representations are independent of the sentence's position in the text and its relation to the previously selected sentences. In this work, we propose a conditional sentence representation, which incorporates the information of the previously selected sentences into the candidate sentence representations. By using such information, the Pointer Network is able to better capture dependencies among sentences. Experiments indicate that our proposed model achieves state-of-the-art performance on most sentence ordering benchmarks and achieves a significant improvement over state-of-the-art performance on short stories datasets.**

I. Introduction

Achieving textual coherence is a key goal of natural language generation. In order to be coherent, a text needs to have a clear discourse structure that makes it easy for the reader to understand the relations between its elements and their communicative goals. Several work has addressed the modeling of text coherence in natural language processing (e.g. [1], [2], [3]). A particular sub-task of the generation of coherent text is sentence ordering, whose aim is to arrange a given set of sentences into its most coherent order. Sentence ordering has attracted much attention in recent years (e.g. [4], [5], [6]) and is useful for several downstream applications such as multi-document text summarization (e.g. [7], [8]) or concept-to-text generation(e.g. [9]).

Recently, several methods have proposed the use of pointer networks [10] for sentence ordering (e.g. [5], [6]). One key advantage of these models is that they are able to predict the order of the set of sentences directly, as opposed to computing and evaluating all possible sentence orderings, hence significantly reducing their run-time requirements while achieving competitive performance.

In a coherent discourse, sentences are connected to one another not only based on their main content but also on more subtle aspects that are highlighted when the sentence is placed in a specific context. Although sentence representations that are independent of their position and their surrounding sentences can capture the main content of each sentence, secondary aspects which can further link consecutive sentences logically are not well captured. In the task of sentence ordering, we do not have access to the correct positional information of sentences beforehand, hence in conventional pointer-based models, the sentence representations are the same regardless of the time step and their position. We argue that, at each decoding step, the decoder should consider different aspects of the sentences depending on the previously selected sentences in order to better capture the discourse relations between consecutive sentences. We believe that it is worthy of exploring a sentence representation that not only is a function of the content of the sentence but also is a function of the candidate position of the sentence and the surrounding sentences in their correct order for sentence ordering.

In order to better model discourse coherence, we have developed a conditional sentence representation that captures local dependencies among consecutive sentences. These representations are obtained based on the content of the sentence conditioned on the information of previously selected sentences. At each decoding step, the pointer is looking for different information based on previously selected sentences. Therefore, at each decoding step, the model creates a new conditional representation for the candidate sentences. Considering these representations alongside conventional static representations allows the model to better capture logical and structural dependencies for sentence ordering.

In this paper, we propose to use a conditional sentence representation in pointer networks for the task of sentence ordering. Our model consists of an encoder and a decoder. The encoder is a hierarchical attention mechanism stacked on top of a pre-trained BERT model [11]. The pointer decoder uses the proposed dynamic sentence representations to predict the correct order of sentences in a paragraph. We calculate these conditional representations considering both the content of the sentence and the previously selected sentences in the text. Experiments show that our model achieves state-of-the-art performance on several sentence ordering benchmarks. It achieves significantly better performance on stories datasets where the local relation of consecutive sentences plays a pivotal role in the task.

In the following sections, we first present previous work in the area, then in Section III we introduce our model in detail. Section IV describes the results of our models on different datasets along with analysis of the results. Finally, Section V

F. Farahnak and L. Kosseim, "Using Conditional Sentence Representation in Pointer Networks for Sentence Ordering," *2021 IEEE 15th International Conference on Semantic Computing (ICSC)*, 2021, pp. 288-295, doi: 10.1109/ICSC50631.2021.00056.

discusses conclusion and future work.

II. Related Work

Traditionally, sentence ordering approaches have relied on handcrafted linguistic features to model textual coherence. The Probabilistic Model of [12] used linguistic features to represent sentences in a vector space, then calculated the probability of transition between adjacent sentence vectors. The Content Model of [13] modeled topic transition using Hidden Markov Model (HMM); while the Entity Grid approach of [3] modeled entities transition between adjacent sentences to capture local coherence. These models constituted successful non-neural approaches; however, they have been shown to be highly domain specific, hence difficult to port across domains.

To address the portability issue, several data-driven approaches have been proposed such as window network [14] and pairwise ranking [4]. These models used neural networks to represent sentences in a vector space, then assigned the probability of being coherent using another neural network. Although these methods are able to discriminate coherent texts from non-coherent texts, they are not efficient enough to address the task of sentence ordering, as they require the evaluation of each possible sentence ordering.

More recently, sentence ordering models have been developed using an encoder-decoder model taking advantage of pointer networks [10]. In these models, a sentence encoder initially provides a dense representation of each sentence regardless of its context in the paragraph. Then, a paragraph encoder creates a context vector to represent the paragraph as a whole. Finally, the decoder, implemented as a pointer network, decides which sentence should be selected next given the paragraph representation as well as the previously selected sentences. A variety of architectures have been explored to use as the encoders and decoders. For example, [15] and [5] used LSTMs [16] for both the sentence and paragraph encoders as well as for the decoder. On the other hand, given that there is no information regarding the correct order of input sentences and RNN-based models are sensitive to the order of input sequences, [6] proposed the use of transformer [17] as the paragraph encoder. Hierarchical Attention Networks [18] used an LSTM network for encoding words and transformer encoders for encoding sentences and paragraphs. They also utilized a transformer decoder instead of an LSTM for the pointer decoder.

Some studies tried to take advantage of linguistic features to augment pointer network based models. In particular, Graph-based models [19] use a graph encoder to encode a paragraph, where the graph is derived from the linguistic relations between sentences. On the other hand, Topic-guided models [20] augment sentence representations with their topics and let the pointer network choose the next sentence based on both the content of the sentences as well as their topics.

Apart from pointer based models, Ranking networks [21] take advantage of the pre-trained BERT model and transformer encoders for encoding sentences and paragraphs. However, as a decoder, they use a feed-forward network to predict a relative score of each sentence as an indicator for their position in the paragraph.

Our proposed model differs from the previous approaches of [5], [6], [18], [19] and [20], as these calculate sentence representations prior to the decoding step, which leads to representations that are independent from the relative position of the candidate sentence. In contrast, our conditional sentence representation is a function of both the content of the candidate sentence as well as the previously selected sentences. In this way, we dynamically calculate a new set of sentence representations at each decoding step where these representations capture the relation between the candidate sentences and the previously selected sentences. Moreover, our model does not use any linguistic features which makes it easy to apply to new datasets or domains.

III. Model Description

Our model uses a hierarchical encoder and a decoder module. The encoder module consists of three sub-modules: word encoder, sentence encoder, and paragraph encoder which generate embedding representations corresponding to each word, sentence and paragraph respectively. The decoder, a pointer-based module, receives the output of the word encoder (word embeddings), the sentence encoder (paragraph-aware sentence embeddings), and the paragraph encoder (paragraph embedding) as its input, then, at each decoding step, finds the most probable sentence that should appear next.

In this section we will first define the task, then describe each sub-module in detail.

A. Task Description

The goal of sentence ordering is to find the most coherent order of a given set S of n sentences that belong to a paragraph P, where $S = \{s_1, s_2, \cdots, s_n\}$, each sentence $s_i = \{w_{i_1}, w_{i_2}, \ldots, w_{i_{L_i}}\}$, L_i is the length of the i^{th} sentence and w_{i_j} is the j^{th} word in s_i. The task is to train a probabilistic model to compute the probability of an ordering $o = \{s_{o_1}, s_{o_2}, \ldots, s_{o_n}\}$:

$$P(o|S) = \prod_{j=1}^{n} P(s_{o_j}|s_{o_{j-1}}, \ldots, s_{o_1}) \tag{1}$$

Given the correct order $o^* = \{s_{o_1^*}, s_{o_2^*}, \ldots, s_{o_n^*}\}$, the objective is to maximize the probability $P(o^*|S)$ among all possible permutations:

$$P(o^*|S) \geq P(o_i|S), \forall o_i \in \psi \tag{2}$$

where ψ is the set of all possible permutations of the sentences s_i in S.

B. Word Encoder

In our experiments, we use the pre-trained BERT language model [11] as the word encoder module. However, any language model that provides contextual word representations can be used as the word encoder module. The BERT model which uses a transformer encoder [17], encodes each word w_{ij} in sentence s_i and generates a contextual embeddings $e_{w_{ij}}$ that

contains information of the word itself as well as its context (i.e. the sentence). In the following sections, we will describe the main characteristics of the transformer encoder used in our model.

C. Sentence Encoder

Our sentence encoder consists of two sub-modules: a) a multi-head attention that provides a summary of the word embeddings of each sentence, b) a transformer encoder [17] that captures the logical relations among sentences.

First, the sentence encoder module receives all the word embeddings $e_{w_{ij}}$ for each sentence s_i and generates the sentence embedding e_{s_i} using a multi-head attention [17] module followed by a layer norm (LN) (see Eq. 3).

$$\begin{aligned} &Attn(Q,K,V) = softmax\left(\frac{QK^\top}{\sqrt{d/H}}\right)V \\ &h_j = Attn(QW_j^Q, KW_j^K, VW_j^V) \\ &MultiHead(Q,K,V) = [h_1;h_2;\ldots;h_H]W^O \\ &MH = MultiHead(Q,K,V) \\ &e_{s_i} = LN(MH + FFN(MH)) \end{aligned} \tag{3}$$

where W_j^Q,W_j^K, and W_j^V are the weights for the query, key and value of j^{th} attention head (h_j), $MultiHead(Q,K,V)$ is the concatenation of all heads multiplied by the output weights W^O, MH is the output of the multi-head attention, and FFN is a position-wise feed-forward layer. Here we considered word representations $e_{w_{ij}}$ as keys (K) and values (V). We used a fixed query Q (a vector of 1s) and let the model learn which words in the sentence it should give more attention to, in order to get a better sentence representation.

Sentence embeddings e_{s_i} have the semantic representation of the sentences s_i. In order to have a high level representation and capture the logical relations among sentences, following [6], we apply a transformer encoder module on the sentence embeddings (e_{s_i}).

Our transformer encoder consists of l identical layers of self attention where the queries, keys and values of the first layer, are all derived from the sentence embeddings (e_{s_i}). Then, the output of each layer (E_{out}^j) feeds as input (E_{in}^{j+1}) to the next layer (see Eq. 4). The output of the last layer contains contextual information of each sentence with respect to the other sentences in the paragraph; hence, they capture the logical dependencies between all sentences. We call these representations: *paragraph-aware sentence embeddings* $e_{s_i}^P$.

$$\begin{aligned} &E_{in}^1 = \{e_{s_1}, e_{s_2}, \ldots, e_{s_n}\} \\ &E_{in}^j = E_{out}^{j-1} \\ &MH = MultiHead(E_{in}^j, E_{in}^j, E_{in}^j) \\ &E_{out}^j = LN(MH + FFN(MH)) \\ &e_{s_i}^P = E_{out}^l \end{aligned} \tag{4}$$

D. Paragraph Encoder

In order to have a paragraph representation (e_P) of the paragraph P, similarly to the previous step (the sentence encoder module), we use a multi-head attention (see Eq. 5) where the query is a fixed vector of 1s and the keys and values are the paragraph-aware sentence embeddings ($e_{s_i}^P$).

$$\begin{aligned} &MH = MultiHead(Q = [1\ldots1], K = e_{s_i}^P, V = e_{s_i}^P) \\ &e_P = LN(MH + FFN(MH)) \end{aligned} \tag{5}$$

E. Decoder

Our decoder is based on pointer networks [10] which consists of an LSTM network augmented with an attention mechanism. At each decoding step, the attention mechanism (pointer) chooses which sentence should be selected among the remaining sentences. The selected sentence passes to the output and its corresponding representation is fed to the LSTM. For the pointer, we use a feed forward attention mechanism using Eq. 6.

$$\begin{aligned} &q = ReLU(W^q(query)) \\ &k = ReLU(W^k(states)) \\ &scores = W^e(Tanh(q;k)) \\ &pointers = Softmax(scores) \end{aligned} \tag{6}$$

where $query$ is the hidden state of the LSTM and $states$ are the sentence representations.

In previous work (e.g. [5], [6]), a fixed set of sentence representations (e.g. $e_{s_i}^P$) is used for all the decoding steps. We argue that pointer networks should consider different aspects of the sentences based on previously selected sentences. Hence, we need to dynamically create a new set of sentence representations ($E_{s_i}^t$) at each decoding step t. In our proposed architecture, the pointer attends over these new representations and then we feed the corresponding paragraph-aware sentence representation ($e_{s_i}^P$) to the LSTM.

Figure 1 shows the general architecture of our decoder. As illustrated, the decoder receives paragraph embeddings (e_P) from the paragraph encoder module, paragraph-aware sentence embeddings ($e_{s_i}^P$) from the sentence encoder module, and word embeddings ($e_{w_{ij}}$) from the word encoder module. The paragraph embedding (e_P) is fed as the initial hidden state of the LSTM. As the start sentence, a vector of 0s is fed to the LSTM as its input.

In order to create representations that can capture the meaning of a sentences and at the same time capture its relation to the previous sentences, we propose a conditional module. Our conditional module is a multi-head attention (see Eq. 7) where similarly to the sentence encoder, keys (K) and values (V) are the word representations $e_{w_{ij}}$; however, we feed the hidden state (h^{t-1}) of the LSTM network as the query (Q). Since the query Q has the information of previously selected sentences, the attention heads focus more on the information that can connect the sentence to the previously selected sentences. Considering that we use the

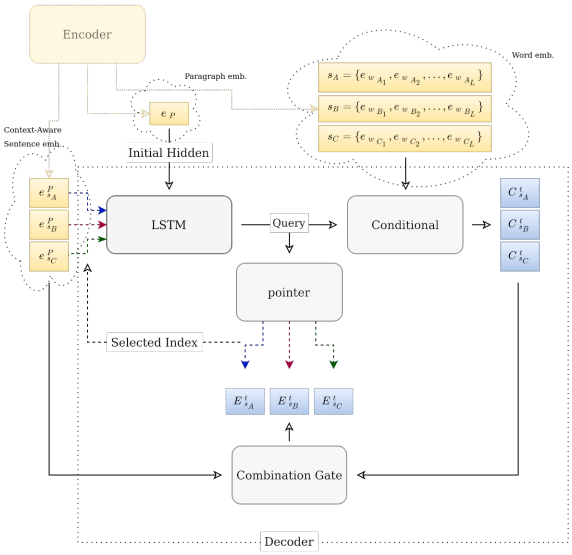

Fig. 1. General architecture of the pointer decoder

same query for both the conditional module and the pointer, the representations achieved by the conditional module are aware of what information the pointer is looking for, hence, the conditional module highlights those information in the sentence representations.

$$MH = MultiHead(Q = h^{t-1}, K = e_{w_{ij}}, V = e_{w_{ij}})$$
$$C^t_{s_i} = LN(MH + FFN(MH)) \quad (7)$$

We can either pass the conditional representations ($C^t_{s_i}$) directly to the pointer or we can pass both the conditional representations ($C^t_{s_i}$) and the paragraph-aware representations ($e^P_{s_i}$) and let the model learn how to use these two representations.

In the case where only the conditional representations are used to choose the next sentence, these representations are passed directly to the pointer mechanism (see Eq. 8). We call this model *Conditional-Only* in Section IV.

$$E^t_{s_i} = C^t_{s_i} \quad (8)$$

In the case where these two representations ($C^t_{s_i}$ and $e^P_{s_i}$) are combined, two types of combination are used (see Fig 1): via an additive gate and via concatenation. Combining these two representations allows the model to consider both the local dependencies among consecutive sentences ($C^t_{s_i}$) and the global logical relations among all sentences ($e^P_{s_i}$) to choose the next sentence.

a) Additive gate: We calculate the weighted sum of $C^t_{s_i}$ and $e^P_{s_i}$ using a gate g (see Eq. 9). To compute the gate g, first, we feed the concatenation of the two vectors $C^t_{s_i}$ and $e^P_{s_i}$ to a dense layer then a sigmoid function calculates the value of g between 0 and 1. The gate g determines the impact of each of $C^t_{s_i}$ and $e^P_{s_i}$ in the final representation.

$$g = \sigma(W_g[C^t_{s_i}; e^P_{s_i}])$$
$$E^t_{s_i} = LN(g \times C^t_{s_i} + (1 - g) \times e^P_{s_i}) \quad (9)$$

b) Concatenation: Another strategy is to simply map the concatenated vector to $E^t_{s_i}$ using a matrix W_s (see Eq. 10).

$$E^t_{s_i} = W_s[C^t_{s_i}; e^P_{s_i}] \quad (10)$$

Dataset	train	valid	test
NIPS	2427	408	377
NSF	87365	8468	21031
SIND	40155	4990	5055
ROC	78529	9816	9817

TABLE I
NUMBER OF SENTENCES IN EACH DATASET

IV. EXPERIMENTS AND RESULTS

A. Datasets

We evaluated our proposed model on two different domains: scientific abstracts and short stories because they have different characteristics. Scientific abstracts are usually well-written and have a clear and stereotypical discourse structure which make them suitable candidates for the task of sentence ordering. On the other hand, short stories usually do not follow a specific discourse pattern which requires the model to capture the intrinsic meaning of the story.

1) Scientific Abstracts: Two datasets of scientific abstracts were used: NIPS [19], and NSF [22]. Table I shows statistics of these datasets.

a) NIPS: Following previous work [5], we considered abstract of papers published in NIPS from 2005 to 2013 for training, and 2014 and 2015 for validation and test respectively. We used the parsed dataset provided by [19] in their Github repository without any modification. It contains abstracts ranging from 2 to 15 sentences in length.

b) NSF: We also considered the NSF Research Award Abstracts dataset from the UCI Machine Learning Repository [22] for our experiments. We used abstracts from 1990 to 1999 for training, 2000 for validation and 2001 to 2003 for testing. Due to lack of computational resources, for training and validation, we removed abstracts longer than 15 sentences; however we used abstracts with a maximum of 40 sentences for testing to have a fair comparison with previous work [5].

2) Short Stories: For short stories, we used two datasets: SIND [23] and ROC [24] (see Table I).

a) SIND: The SIND (Sequential Image Narrative) dataset [23], also known as the VIST (Visual Story Telling) dataset, contains sequential images and their corresponding description and stories. We used the story texts for the task of sentence ordering. Each story has 5 sentences. We used the same training, validation and test sets as provided.

Hyper-Parameter	NIPS	NSF	SIND	ROC
# layers	2	4	2	4
Model dim	256	128	64	128
LSTM dim	128	256	128	512

TABLE II
HYPER-PARAMETERS FOR EACH MODEL. # LAYERS INDICATES THE NUMBER OF ATTENTIONS LAYERS. MODEL DIM REFERS TO THE EMBEDDING SIZE OF THE MODEL. LSTM DIM IS THE HIDDEN SIZE OF LSTM IN THE POINTER NETWORK.

b) ROC: The ROCStory dataset (common sense story dataset) [24] contains stories of 5 sentences. Following previous work [18], [21], we randomly split the dataset by $8:1:1$ to get the training, validation and testing datasets. Compared to the SIND dataset, the stories contain more explicit signals of discourse coherence since no images are provided as supplementary information.

B. Experiments

a) Training setup: We implemented our model using the PyTorch [25] library and trained it using the AdamW [26] optimizer with the initial learning rate of $1e^{-4}$ for all modules, except for the word encoder. We did not train the word encoder module for the first two epochs, then, we started training it with a learning rate of $1e^{-5}$. The learning rate was divided by 2 if there was no improvement on the validation set. The training was stopped if no improvement was observed on the validation set in a period of 3 epochs. At the end, the model with the highest performance on the validation set was chosen for testing. The weights of the word encoder module were initialized with the pre-trained BERT base model from HuggingFace [27] with 12 encoder layers. The following hyper-parameters were chosen based on the validation sets without extensive hyper-parameter optimization: we used 4 heads of attention with a dropout rate of 0.2 for all multi-head attention modules. We also considered dropout rates of 0.5 and 0.3 for the word encoder and pointer respectively. The rest of the hyper-parameters, which were dataset-dependent are shown in Table II.

b) Models: We experimented with three models that use our proposed conditional sentence representation with three different strategies described in Section III-E. All these models use a similar encoder and feed the same sentence representation to the LSTM; the only difference between these three models is in the way they use the conditional representations (i.e. the combination of $C^t_{s_i}$ and $e^P_{s_i}$):

1) *Add-Conditional* uses the additive gate in the decoder to combine both representations ($C^t_{s_i}$) and ($e^P_{s_i}$) using ($g \times C^t_{s_i} + (1-g) \times e^P_{s_i}$), where g is learned by the model.
2) *Cat-Conditional* uses a decoder that concatenates both representations ($[C^t_{s_i}; e^P_{s_i}]$).
3) *Conditional-Only* only passes the conditional representations ($C^t_{s_i}$). This model is equivalent to *Add-Conditional* with $g = 1$.

In order to better understand how the conditional sentence representations impact the performance, we also considered a baseline model without using the conditional sentence representation:

4) *Baseline* only passes the paragraph-aware representations ($e^P_{s_i}$). This model is equivalent to *Add-Conditional* with $g = 0$.

The performance of these four models on the benchmark datasets of Section IV-A are shown in Table III, along with the performance of state-of-the-art models as reported in their paper. Recall from Section II that LSTM+PtrNet [5] used LSTMs for both the sentence and paragraph encoders with pointer networks. ATTOrderNet [6] substituted the LSTM for a paragraph encoder with a transformer mechanism. Hierarchical attention [18] used transformer for both the paragraph encoder and the pointer. Ranking Model [21] used a regression approach to predict a relative score for each sentences as an indicator for their positions. SE-Graph [19] augmented the pointer network with a graph neural network to encode sentences. Topic-Guided [20] added sentences representation with their topics to have more information for each sentences.

c) Evaluation Metrics: Accuracy (Acc) or position-wise accuracy measures the absolute position of each sentence. It penalizes shifts even with a correct relative order.

Perfect Match Rate (PMR) is a hard metric that counts the number of exact matching paragraphs over the total number of paragraphs. It is hard because if a single sentence is misordered, it considers the entire paragraph as wrong.

Kendall's Tau (τ) is a rank correlation metric which measures how the predicted order is similar to the given order. The formula of Kendall's Tau is given in Eq. 11 where the number of inversions is the number of pairs with incorrect relative order and n is the length of the paragraph. Its range is between -1 to 1, from worst to best.

$$\tau = 1 - \frac{2(\#inversion)}{n(n-1)/2} \tag{11}$$

C. Results and Analysis

Table III reports on the experimental results on all datasets. As the table shows, all models with the conditional sentence representations ($C^t_{s_i}$) achieved a better performance than the *Baseline* model.

As expected, models that used our proposed representations achieved a significantly better performance on the short stories datasets (SIND and ROC) compared to *Baseline* (the model using paragraph-aware sentence representations $e^P_{s_i}$ only). In these short stories, local dependencies and the relation between consecutive sentences have a significant impact on the order of sentences. Hence, we can inferred from these results that the conditional sentence representations ($C^t_{s_i}$) were able to capture these local dependencies successfully.

On the NIPS dataset, the *Cat-Conditional* model achieved better results than the *Baseline* model, but this time, the improvement is not statistically significant. On the other hand,

Model	Scientific Abstracts						Short Stories			
	NIPS			NSF			SIND		ROC	
	Acc	PMR	τ	Acc	PMR	τ	PMR	τ	PMR	τ
LSTM+PtrNet	51.55	–	0.72	28.33	–	0.51	–	–	–	–
ATTOrderNet	56.09	–	0.72	37.72	–	0.55	14.01	0.49	–	–
Hierarchical	–	–	–	–	–	–	15.01	0.5021	39.62	0.7322
Ranking	–	24.13	0.7462	–	9.78	0.5798	15.48	0.5652	38.02	0.7602
SE-Graph	57.27	–	0.75	–	–	–	16.22	0.52	–	–
Topic-Guided	59.43	31.44	0.75	**42.67**	**22.35**	0.55	15.18	0.53	–	–
Baseline	64.71	29.17	0.7962	38.66	10.68	0.5818	15.66	0.5607	36.79	0.7554
Add-Conditional	64.97	30.76	0.7950	41.86	13.55	**0.6060**	**17.46**	0.5640	39.53	0.7652
Cat-Conditional	**65.73**	**31.83**	**0.7990**	41.46	13.19	0.6029	17.01	0.5592	39.46	0.7660
Conditional-Only	63.42	30.23	0.7843	41.56	13.44	0.6006	16.99	**0.5672**	**42.01**	**0.7726**

TABLE III
EXPERIMENTAL RESULTS OF DIFFERENT METHODS ON STANDARD SENTENCE ORDERING DATASETS

Conditional-Only achieved the lowest performance among the models with conditional representations. This seems to agree with the fact that in these abstracts, sentences follow a clear discourse structure and local dependencies among consecutive sentences have a lower impact on the correct order of sentences compared to other datasets such as short stories.

Results with the NSF dataset indicate that all models that used the conditional sentence representations achieved similar performances ($\tau \approx 0.60$); while the model without this representation suffers from a significant drop in performance ($\tau \approx 0.58$). This behavior does not occur with the NIPS dataset, where the performance with and without the conditional representation does not show such a marked difference. This is rather surprising as both datasets are scientific abstracts and hence should share a similar discourse structure. We suspect that this behaviour stems from the strict writing guidelines imposed by NSF, which aim to simplify the language, the grammar and the structure of the abstracts, that the conditional representation picks up on. However, more analysis is required to investigate this.

Compared with previous work [28], [18], [21], the models with conditional sentence representations ($C^t_{s_i}$) achieved better results on most metrics on all datasets. Results show that the *Baseline* model (which does not use $C^t_{s_i}$) achieved similar or lower results than previous work on most datasets. This clearly shows that our highest performance is not related to the settings we used for our training, but rather the use of the conditional representations ($C^t_{s_i}$) alone. For the NIPS dataset, we cannot easily compare our results with previous work since the number of samples in our test sets is different from the one reported in previous work [5]. In addition, since the NIPS dataset is much smaller than the other datasets (see Table I), using a pretrained model such as BERT has a significant impact on its performance. However, the Ranking model of [21] also used the same pretrained model yet its performance lies in the same range as other work [20], [18].

In order to better understand the impact of the conditional representations, we analyzed the focus of the attention heads. Figure 2 illustrates the values of all the attention heads of the sentence encoder and the conditional modules of the *Add-Conditional* model trained on the ROC dataset alongside with their corresponding value of g gate. In Figure 2, the conditional modules are shown on the left hand side; while the heads of the sentence encoder are shown on the right hand side. Recall that the output of the sentence encoder is independent of the position of the sentences, while this is not the case for the conditional module. Figure 2 shows the heads of the conditional module only for the correct position of each sentence. As Figure 2 shows, the sentence encoder module tends to attend to all tokens in the sentence to capture the meaning of the sentence; however, the conditional module focuses on specific tokens based on the relation of the sentence to the previous sentences. For example, in the first sentence *"Judy wanted to make some cookies."*, since there is no previously selected sentence, more attention is given to the subject of the sentence (*Judy*); however, on the third sentence, the attention focuses on the specific words *without* and *it*, where *it* refers to *butter* in the previous sentence, another head attends more on *decided* and *to* which connects the sentence to the previous one.

We further analyzed the value of the g gate to see how the two sentence representations contribute in the final representation. Table IV shows the average value of g for sentences in their correct position. As Table IV indicates, for the first sentences, where there is no previously selected sentences therefore, the paragraph-aware representations ($e^P_{s_i}$) contribute more to the final representations ($E^t_{s_i}$). However, for the rest of the sentences, g is closer to 0.5 leading to a more equal contribution of the conditional representation ($C^t_{s_i}$) and the paragraph-aware representations ($e^P_{s_i}$). Again, this agrees with our intuition that the conditional representation tends to capture local dependencies between consecutive sentences and the gate g puts more weight on the conditional representation when information from previously selected sentences is available.

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed a model based on pointer networks for the task of sentence ordering. The model uses a hierarchical encoder to encode words, sentences and the paragraph as a whole. Then the decoder uses all these embeddings to predict the next sentence in the paragraph. Our decoder considers a novel conditional sentence representation,

which combines both the content of the sentence and the previously selected sentences in the text. Experiments with standard benchmark datasets show that the proposed model achieves state-of-the-art results using most of the evaluation metrics. The increase in performance is more significant in short stories datasets, where local dependencies in consecutive sentences play an important role in the discourse structure of the paragraph.

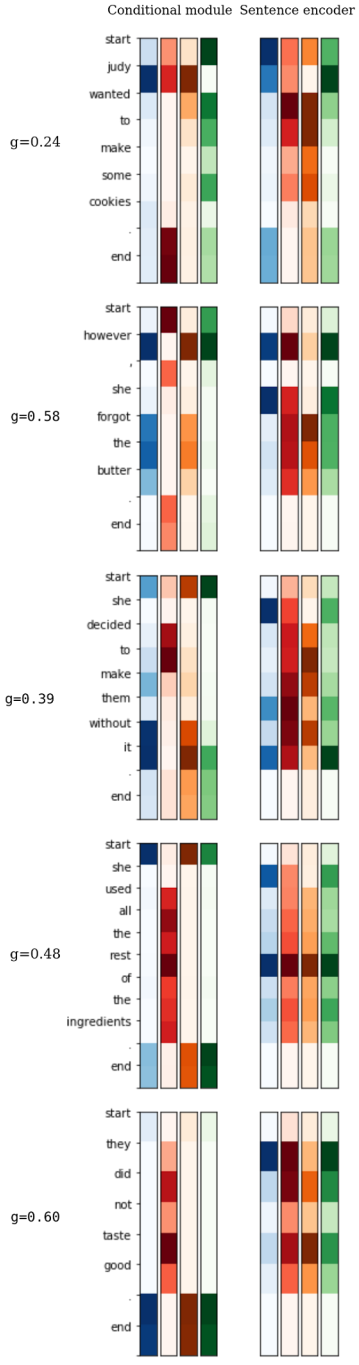

Fig. 2. Visualisation of the attention heads for the *Add-Conditional* model trained on the ROC dataset with their corresponding values of the g gate. The heads of the conditional module are on the left and the heads of the sentence encoder module are on the right.

Sentence	g
1	0.3 ± 0.1
2	0.5 ± 0.1
3	0.6 ± 0.1
4	0.6 ± 0.1
5	0.6 ± 0.1

TABLE IV
THE VALUE OF THE GATE g IN THE *Add-Conditional* MODEL TRAINED ON THE ROC DATASET. RECALL FROM EQ. 9 THAT A HIGHER VALUE OF g INDICATES MORE CONTRIBUTION OF THE CONDITIONAL REPRESENTATION ($C^t_{s_i}$).

As future work, we would like to investigate the use of our proposed model on different domains and textual genres. In addition, our conditional sentence representation only considers the previously selected sentences, however, considering the next sentences to generate embedding representations is also worth investigating. Finally, the scientific abstracts used in this study required significant preprocessing and selection procedures, that led to different numbers of samples of different quality. Designing a standard and well structured dataset for this task is a crucial need.

ACKNOWLEDGMENT

This work was financially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC). The authors would like to thank the anonymous reviewers for their feedback on a previous version of this paper.

REFERENCES

[1] W. Mann and S. Thompson, "Rhetorical structure theory: Towards a functional theory of text organization," *Text*, vol. 8, no. 3, p. 243–281, 1988.

[2] J. Hobbs, *Literature and Cognition*, ser. CSLI Lecture Notes. Center for the Study of Language (CSLI), 1990, no. 21.

[3] R. Barzilay and M. Lapata, "Modeling local coherence: An entity-based approach," *Computational Linguistics*, vol. 34, no. 1, pp. 1–34, 2008.

[4] X. Chen, X. Qiu, and X. Huang, "Neural sentence ordering," *CoRR*, vol. abs/1607.06952, 2016.

[5] L. Logeswaran, H. Lee, and D. R. Radev, "Sentence ordering and coherence modeling using recurrent neural networks," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, 2018, pp. 5285–5292.

[6] B. Cui, Y. Li, M. Chen, and Z. Zhang, "Deep attentive sentence ordering network," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, Brussels, Belgium, Oct.-Nov. 2018, pp. 4340–4349.

[7] R. Barzilay, N. Elhadad, and K. R. McKeown, "Inferring strategies for sentence ordering in multidocument news summarization," *Journal of Artificial Intelligence Research*, vol. 17, no. 1, pp. 35–55, Aug. 2002.

[8] R. Nallapati, F. Zhai, and B. Zhou, "Summarunner: A recurrent neural network based sequence model for extractive summarization of documents," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI'17)*, San Francisco, California, USA, 2017, pp. 3075–3081.

[9] I. Konstas and M. Lapata, "Concept-to-text generation via discriminative reranking," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, Jeju Island, Korea, Jul. 2012, pp. 369–378.

[10] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks," in *Advances in Neural Information Processing Systems 28*, ser. (NeurIPS 2015), C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Montreal, Canada: Curran Associates, Inc., May 2015, pp. 2692–2700.

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[12] M. Lapata, "Probabilistic text structuring: Experiments with sentence ordering," in *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL 2003)*, Sapporo, Japan, Jul. 2003, pp. 545–552.

[13] R. Barzilay and L. Lee, "Catching the drift: Probabilistic content models, with applications to generation and summarization," in *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: (HLT-NAACL 2004)*, Boston, Massachusetts, USA, May 2 - May 7 2004, pp. 113–120.

[14] J. Li and E. Hovy, "A model of coherence based on distributed sentence representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, Doha, Qatar, Oct. 2014, pp. 2039–2048.

[15] J. Gong, X. Chen, X. Qiu, and X. Huang, "End-to-end neural sentence ordering using pointer network," *CoRR*, vol. abs/1611.04953, 2016.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30*, ser. (NeurIPS 2017), I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Long Beach CA: Curran Associates, Inc., Dec 2017, pp. 5998–6008.

[18] T. Wang and X. Wan, "Hierarchical attention networks for sentence ordering," in *Proceedings of the AAAI Conference on Artificial Intelligence*, ser. (AAAI 2019), vol. 33, Honolulu, Hawaii, USA, 07 2019, pp. 7184–7191.

[19] Y. Yin, L. Song, J. Su, J. Zeng, C. Zhou, and J. Luo, "Graph-based neural sentence ordering," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, (IJCAI-19)*, Macao, China, 2019, pp. 5387–5393.

[20] B. Oh, S. Seo, C. Shin, E. Jo, and K.-H. Lee, "Topic-guided coherence modeling for sentence ordering by preserving global and local information," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, Nov. 2019, pp. 2273–2283.

[21] P. Kumar, D. Brahma, H. Karnick, and P. Rai, "Deep attentive ranking networks for learning to order sentences," *arXiv preprint arXiv:2001.00056*, 2019.

[22] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: https://archive.ics.uci.edu/

[23] T.-H. K. Huang, F. Ferraro, N. Mostafazadeh, I. Misra, J. Devlin, A. Agrawal, R. Girshick, X. He, P. Kohli, D. Batra *et al.*, "Visual storytelling," in *15th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2016)*, 2016.

[24] N. Mostafazadeh, N. Chambers, X. He, D. Parikh, D. Batra, L. Vanderwende, P. Kohli, and J. Allen, "A corpus and cloze evaluation for deeper understanding of commonsense stories," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, Jun. 2016, pp. 839–849.

[25] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlche Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035.

[26] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Seventh International Conference on Learning Representations (ICLR 2019)*, New Orleans, Louisiana, May 2019.

[27] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and J. Brew, "Huggingface's transformers: State-of-the-art natural language processing," *ArXiv*, vol. abs/1910.03771, 2019.

[28] B. Cui, Y. Li, Y. Zhang, and Z. Zhang, "Text coherence analysis based on deep neural network," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, ser. CIKM '17. New York, NY, USA: ACM, 2017, pp. 2027–2030.