Preprint from https://ostendorff.org/pub/

Rémi Calizzano, Malte Ostendorff, Georg Rehm. "Ordering Sentences and Paragraphs with Pre-trained Encoder-Decoder Transformers and Pointer Ensembles" in *Proceedings of the 21st ACM Symposium on Document Engineering*, 2021.

Ordering Sentences and Paragraphs with Pre-trained Encoder-Decoder Transformers and Pointer Ensembles

Rémi Calizzano
DFKI GmbH
Berlin, Germany
remi.calizzano@dfki.de

Malte Ostendorff
DFKI GmbH
Berlin, Germany
malte.ostendorff@dfki.de

Georg Rehm
DFKI GmbH
Berlin, Germany
georg.rehm@dfki.de

ABSTRACT

Passage ordering aims to maximise discourse coherence in document generation or document modification tasks such as summarisation or storytelling. This paper extends the passage ordering task from sentences to paragraphs, i. e., passages with multiple sentences. Increasing the passage length increases the task's difficulty. To account for this, we propose the combination of a pre-trained encoder-decoder Transformer model, namely BART, with variations of pointer networks. We empirically evaluate the proposed models for sentence and paragraph ordering. Our best model outperforms previous state of the art methods by 0.057 Kendall's Tau on one of three sentence ordering benchmarks (arXiv, VIST, ROCStory). For paragraph ordering, we construct two novel datasets from Wikipedia and CNN-DailyMail on which we achieve 0.67 and 0.47 Kendall's Tau. The best model variation utilises multiple pointer networks in an ensemble-like fashion. We hypothesise that the use of multiple pointers better reflects the multitude of possible orders of paragraphs in more complex texts. Our code, data, and models are publicly available[1].

CCS CONCEPTS

• **Information systems** → **Language models**; **Document structure**; **Content analysis and feature selection**.

KEYWORDS

ordering, pointer networks, discourse coherence, summarisation, transformers

ACM Reference Format:
Rémi Calizzano, Malte Ostendorff, and Georg Rehm. 2021. Ordering Sentences and Paragraphs with Pre-trained Encoder-Decoder Transformers and Pointer Ensembles. In *ACM Symposium on Document Engineering 2021 (DocEng '21), August 24–27, 2021, Limerick, Ireland.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3469096.3469874

1 INTRODUCTION

Under the umbrella of a project on document and content curation technologies [23], we work on methods of automatically selecting relevant text segments from multiple documents in order to reassemble them in meaningful, coherent ways, i. e., generating coherent stories from heterogeneous sources. To achieve this, after extracting text segments that relate to the same subject, we need to order them in such a way as to maximize discourse coherence. Achieving this overall task and vision would result in the creation of a system able to automate semantic storytelling [24]. The (re)ordering task, also known as passage ordering, is about finding the right or best order of a given set of text passages, e. g., sentences, so that their discourse coherence is maximised. The ability to structure and organise passages in a meaningful way is closely connected to discourse coherence modeling and a fundamental property of language models [17].

Beyond our use case, passage ordering models are used in text generation tasks such as summarisation for execution and evaluation [1, 16]. Recent ordering approaches [6, 17, 31] utilise a model architecture composed of two major components: an encoder-decoder model represents the passages and a pointer network selects the correct next passage given passage representations. These models are commonly evaluated on datasets, in which the passages meant to be ordered are sentences.

In order to match our use case, which is about ordering text segments of several sentences, i. e., paragraphs, we extend the ordering task by introducing two new paragraph ordering datasets. The first dataset, based on news articles from CNN-DailyMail [27], contains on average more than 14 paragraphs to order. The second dataset is constructed using Wikipedia and contains long paragraphs of 62 words on average. The dataset based on CNN-DailyMail is characterized in particular by a greater number of passages than the previous datasets while the one based on Wikipedia is differentiated by longer passages.

The two new datasets increase the task's difficulty compared to previous datasets since the greater passage length requires better modeling of long-range semantic relations within the text. Therefore, we propose a two-way enhancement of the encoder-decoder-pointer approach. First, we use a state of the art pre-trained Transformer model, BART [15], to replace the previously used encoder-decoder models. This requires an adaptation of BART to encode and decode text not only at the word but also at the passage level. Second, we test and evaluate BART in combination with the pointer network introduced by Wang and Wan [31] and two variations of this pointer network, one with more learning capacity and one using ensemble learning. In addition to our two new datasets, we empirically evaluate the three proposed models on three common sentence ordering datasets (arXiv, VIST, ROCStory). In summary, our main contributions are:

- Creation of two new datasets that extend the passage ordering task from sentences to paragraphs, which are more suited for the creation of "synthesised" documents
- Implementation and evaluation of pre-trained encoder-decoder Transformers with different pointer networks.
- New state of the art results that considerably outperform previous results by 13.51% on the sentence ordering datasets arXiv, VIST, and ROCStory

[1] https://github.com/airKlizz/passage-ordering

The rest of this article is structured as follows. Section 2 presents work related to passage ordering but also encoder-decoder models, pointer networks, and ensemble modeling. Section 3 explains the methodology used with detailed explanations of the three implemented models. Section 4 presents the experiences made with a presentation of each dataset, including those that we have created, and a presentation of metrics and baselines. Section 5 presents the results of the experiments. We discuss the results in Section 6 and provide concluding remarks in Section 7.

2 RELATED WORK

2.1 Encoder-decoder models

Encoder-decoders (also known as sequence-to-sequence models) are used in many other NLP tasks (e. g., translation or summarisation) including the passage ordering task. Encoder-decoders generate the output sequence step by step from an input sequence. The input sequence is used as the encoder's input and the decoder input is its output from the previous step. The RNN (Recurrent Neural Network) encoder-decoder model introduced by Sutskever et al. [28], improved the results in NLP tasks such as translation. Bengio and LeCun [3] improved RNN-based models by introducing attention mechanisms in the decoder part, which addressed the issue that the performance of simple RNN models decreases as the input size increases. Pushing the use of attention mechanisms and intending to facilitate parallelisation of the model, Vaswani et al. [29] introduced the Transformer architecture, an encoder-decoder architecture that uses attention without any recurrent elements. Devlin et al. [7] introduced a Transformer encoder-only model taking advantage of pre-training. The same idea has been applied to encoder-decoder Transformer models by Lewis et al. [15] with the publication of BART, and by Raffel et al. [22] with T5. These two models established the state of the art when they were published.

2.2 Pointer networks

Plain vanilla encoder-decoder models cannot compute the conditional probability of an output sequence when the number of possible classes depends on the input. However, this is needed for the ordering task as the number of classes is the number of passages to be ordered, i. e., it depends on the input. Vinyals et al. [30] introduced a novel architecture named pointer networks to address this issue. Pointer networks are composed of an encoder-decoder model with an attention layer that computes conditional probabilities depending on the encoder and decoder outputs. The computed conditional probabilities correspond to positions in the input sequence. Pointer networks are therefore suitable for the passage ordering task. Wang and Wan [31] improved the pointer network by using a scaled dot-product based attention.

2.3 Passage ordering

Barzilay et al. [1] introduced sentence ordering, attempting to increase the coherence of summaries in multi-document summarisation. Lapata [13] generalised the idea by defining the sentence ordering task as critical for natural language generation applications and proposed a method for computing the probability of each possible order. Barzilay and Lapata [2] proposed a method based on entities. Recently, ordering has gained more attention from the research community again due to several new methods, which can be divided into two categories. Pairwise ordering models order sentences in a pair by pair way [4, 16], while full ordering models order all sentences directly [5, 8, 17, 31]. Pairwise ordering models reduce the task to a classification problem where the two labels are the two possible orders. However, full ordering models, generally encoder-decoder models with a pointer network [8], proved their superiority over pairwise ordering models (see, e. g., Wang and Wan [31]), which is why an encoder-decoder model with a pointer network is the most frequently used architecture for this task. Gong et al. [8] proposed a model using an LSTM encoder-decoder with a pointer network that points to the next sentence using an attention mechanism. An LSTM encoder-decoder is a RNN encoder-decoder using the Long Short-Term Memory architecture introduced by Hochreiter and Schmidhuber [9]. Cui et al. [5] upgraded the model by adding attention to the encoder part. Following the same idea, Wang and Wan [31] replaced the LSTM encoder-decoder with a self-attention encoder-decoder. The resulting model uses LSTMs with attention for sentence representations, a multi-head self-attention encoder-decoder at the sentence level, and a pointer network using scaled dot-product attention [29]. A slightly different approach is used by Cui et al. [6]. The authors develop a new relational pointer decoder by incorporating the relative ordering information provided by BERT [7] into the pointer network. The results presented by Cui et al. [6] represent the state of the art in three major sentence ordering benchmarks [4, 10, 18].

2.4 Ensemble modeling

Ensemble modeling utilises multiple classifiers and combines their outputs into one to maximise overall classification precision. Rokach [26] presents an overview of existing ensemble methods, including the weighted sum method, which combines the outputs using combination weights. This method is especially suitable for the passage ordering task because the number of classes is the number of passages and, therefore, variable. Among others, ensemble modeling is useful for decision making [19], sentiment analysis [11], hate speech detection [25] and other tasks.

Sentence ordering models mainly use an encoder-decoder with a pointer network architecture [5, 8, 17, 31]. This architecture allows the model to order all the sentences directly and has demonstrated its efficiency compared to the pairwise approach for the ordering task [31]. The latest ordering models [6, 7] also take advantage of the latest Transformer architecture introduced by Vaswani et al. [29].

3 METHODOLOGY

3.1 Task Definition

Given $S = \{s_1, s_2, ..., s_N\}$, a sequence of N text segments (passages; sentences or paragraphs), the passage ordering task consists of finding the gold order $O = \{o_1, o_2, ..., o_N\}$ that maximises discourse coherence. The resulting coherent text is composed of the passages $\{s_{o_1}, s_{o_2}, ..., s_{o_N}\}$ in exactly that order, see Figure 1 for an example.

p_1	Jennifer felt bittersweet about it.
p_2	She went into class the next day, weary as can be.
p_3	Her teacher stated that the test is postponed for next week.
p_4	Jennifer has a big exam tomorrow.
p_5	She got so stressed, she pulled an all-nighter.
	Gold order $O = \{4, 5, 2, 3, 1\}$

Figure 1: Example from the ROCStory dataset. The gold order O maximises the coherence of the given sentences p_{1-5}.

The passage ordering model is trained to select the n^{th} passage from S regarding the previously ordered passages $\{s_{o_1}, s_{o_2}, ..., s_{o_{n-1}}\}$. It is therefore trained to maximise the probability:

$$\sum_{n=1}^{N} \log \mathcal{P}(s_{o_n} | s_{o_1}, s_{o_2}, \ldots, s_{o_{n-1}}) \tag{1}$$

3.2 Overview

Our approach adapts the pre-trained encoder-decoder Transformer BART [15] for the passage ordering task by adding a pointer network on top. Figure 2 shows the overall architecture. BART's encoder creates a representation of each passage (S) to order. Then, the decoder creates a representation of the already ordered passages ($\{s_{o_1}, s_{o_2}, \ldots, s_{o_{n-1}}\}$). Finally, the pointer network attends to the encoder representations depending on the last decoder representation. The attention weights correspond to the probability of each passage to be the next one.

3.3 BART

BART is a pre-trained sequence-to-sequence model. BART's encoder is a bidirectional Transformer similar to BERT [7] while the decoder is an auto-regressive Transformer (left to right) as in GPT [21]. BART is pre-trained on a combination of two tasks: text filling and sentence permutation. The sentence permutation task is similar to passage ordering, which is why we chose BART over other pre-trained encoder-decoder models such as T5 [22]. We use the `BASE` version of BART which is composed of six encoder and six decoder layers, with a hidden size of 768. The last hidden state of each input token is, therefore, a vector $\in \mathbb{R}^{768}$.

3.3.1 Encoder. We concatenate the passages S in a continuous text that we tokenise and use as the input of the encoder. Each passage is surrounded by special tokens that represent the beginning <s> and the end of a passage </s>. The *end* token </s> is used by BART to represent the sequence in sequence classification tasks. We follow the same approach and use the last encoder state of the *end* token as the encoder passage representation. We also add another *end* token at the very end of the text to mark the final passage to order. The pointer network has to attend to the encoder representation of this token when all passages are ordered. To summarise, the BART encoder input (X) is the tokenisation of:

<s>s_1</s>...<s>s_N</s><s></s>

We use the last encoder hidden state of the *end* tokens as the representation of the passages, i. e., we do not use the last encoder hidden of the other tokens to compute the representation of the passages. Simplifying, the equation for the encoder is:

$$E = \text{BART}_{\text{encoder}}(X) \tag{2}$$

where $E \in \mathbb{R}^{(N+1)\times 768}$, and e_i is the representation of the i^{th} passage and e_{N+1} is the representation of the last passage to order.

3.3.2 Decoder. BART's decoder takes as input the already ordered passages processed in a similar way as for the encoder as input (Z_n), as well as the encoder output E. Z_n corresponds to the tokenisation of:

<s></s><s>s_{o_1}</s>...<s>$s_{o_{n-1}}$</s>

when $n-1$ passages are already ordered. The first *end* token represents the beginning of the ordered passages and should point to the first passage s_{o_1}. The following *end* tokens should point to the next passages until all passages are ordered. The decoder equation is:

$$d_n = \text{BART}_{\text{decoder}}(E, Z_n) \tag{3}$$

where $d_n \in \mathbb{R}^{1\times 768}$ is the decoder output when the model computes the probability of passages to be the n^{th}.

3.4 Pointer network

The pointer network has to compute the probability of passages to be the n^{th} using the encoder output E and the decoder output d_n. The idea is to use an attention mechanism where d_n attends to each encoder passage representation E_i. The resulting attention weights (P_n) are used as the probabilities to be the next passage. We test three versions of the pointer network.

3.4.1 Version 1: Simple Pointer. This pointer network, used by Wang and Wan [31], computes a scaled dot product attention, where d_n is the query and encoder passage representations E_i are the keys:

$$Q = d_n W_Q \tag{4}$$

$$K = E W_K \tag{5}$$

$$P_n = \text{softmax}(\frac{QK^T}{\sqrt{dim}}) \tag{6}$$

where $W_Q \in \mathbb{R}^{dim\times dim}$ and $W_K \in \mathbb{R}^{dim\times dim}$ and dim is the dimension of BART outputs (768). $P_n \in \mathbb{R}^{N+1}$ is the probability distribution of the n^{th} passage. Precisely, $P_{n,i}$ is the probability of s_i to be the correct passage at the position n (s_{o_n}).

3.4.2 Version 2: Deep Pointer. The learning capacity of the Simple Pointer is limited to the matrices W^Q and W^K. To ensure that this learning capacity was not a limiting factor, we replace these matrices that correspond to single perceptrons with multi-layer perceptrons. We introduce l as the number of layers used.

3.4.3 Version 3: Ensemble Pointer. Many examples of passage ordering have more than one correct ordering, i. e., often there is more than one correct order, even though only one of these is part of the gold standard. To address this observation, we create an Ensemble Pointer composed of multiple Simple Pointer networks called heads. The different heads use the same inputs but have different

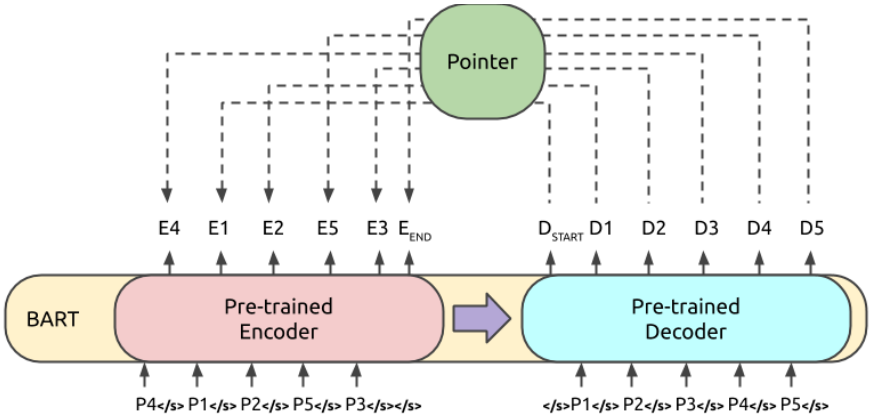

Figure 2: Overview of our model. BART computes encoder and decoder representations of the passages to order and those already ordered. The pointer network uses these to select the next passage using attention between the decoder outputs and encoder outputs.

trainable parameters. Therefore, we assume that each head represents a possible order and that by combining those heads we obtain an order which best reflects the ground truth. The combination consists of a weighted sum of the output of each head where the weights are trainable parameters. This technique takes advantage of the ensemble modeling where the motivation is to reduce the generalisation error of the prediction [12]. It corresponds to the following equation:

$$Q_j = d_n W_j^Q \tag{7}$$

$$K_j = E W_j^K \tag{8}$$

$$P_n^j = \frac{Q_j K_j^T}{\sqrt{dim}} \tag{9}$$

$$P_n = \sum_{j=0}^{h} W_j^O P_{n,i}^j \text{ , for every } i \text{ of } P_n^j. \tag{10}$$

where h is the number of heads and j represents the j^{th} head. Trainable parameters are $W_j^Q \in \mathbb{R}^{dim \times \frac{dim}{h}}$, $W_j^K \in \mathbb{R}^{dim \times \frac{dim}{h}}$ and $W^O \in \mathbb{R}^h$.

4 EXPERIMENTS

We train and evaluate the three models presented on three existing sentence ordering and two new datasets. We compare the results with the results of seven other methods on the three common ordering datasets and with the results of our baseline on the two datasets we created.

4.1 Datasets

We evaluate our models on three datasets for sentence ordering: arXiv [4], VIST [10], and ROCStory [18]. In addition, we create two datasets based on Wikipedia and CNN-DailyMail [27]. Table 1 provides an overview.

The arXiv dataset is composed of abstracts of scientific papers. The VIST dataset was compiled to experiment with visual storytelling tasks. Each entry consists of five images and five sentences that create a story; we only use the sentences. The ROCStory dataset consists of everyday life stories with five sentences each.

We also evaluate two novel datasets. First, we created the Wikipedia dataset to evaluate our model for ordering passages composed of multiple sentences, i. e., paragraphs. This dataset is based on the introductions of English Wikipedia articles; the paragraphs in the introductions are the segments to be ordered. Second, we use the news articles from CNN-DailyMail following the same methodology as with Wikipedia. The CNN-DailMail dataset is composed of 14.5 passages on average (Table 1), which is significantly more than the other datasets. For these two datasets, we randomly shuffle the ordered passages to create the passages to order.

We created these two datasets to evaluate passage ordering as part of the storytelling task since they are more suitable for this downstream task: the new datasets contain more passages or longer passages and, therefore, pose greater challenges to the ordering task. We test how the performance of our ordering models is affected by the challenges of the new datasets.

In addition to the difference in number of passages and passage length, the datasets differ with regard to their text genre which can also affect model performance. The arXiv and Wikipedia datasets are based on abstracts and introductions, respectively, which are typically well-structured. This structure reduces the task's difficulty, especially when passages contain temporal words. The stories from ROCStory also follow a well-defined structure compared to those from VIST and CNN-DailyMail. Indeed, VIST stories were originally linked to images and so the text alone often does not contain all needed information. The CNN-DailyMail news articles are less formalised than abstracts.

4.2 Metrics

We use two metrics to evaluate the models. Perfect Match Ratio (PMR) calculates the accuracy of exact match between gold order

Table 1: Comparison of the datasets. All examples from VIST and ROCStory contain five passages (short sentences). Examples from arXiv are also composed of around five passages on average but they contain twice as many words. CNN-Dailymail has many more passages to order than the other datasets. Wikipedia contains long passages composed of several sentences.

	Number of examples			Number of passages			Number of words per passage		
Dataset	train	validation	test	mean	30^{th} pctl	70^{th} pctl	mean	30^{th} pctl	70^{th} pctl
arXiv	884,912	110,614	110,615	5.38	4	6	27.9	20	32
VIST	40,155	4,990	5,055	5.0	5	5	11.4	8	13
ROCStory	78,525	9,816	9,816	5.0	5	5	10.0	8	12
Wikipedia	85,982	10,747	10,748	6.29	5	8	62.22	31	77
CNN-DailyMail	91,238	5,213	4,449	14.5	12	17	31.6	22	37

and predicted order. Kendall's Tau (τ) measures the correlation of an order with a score from -1 to 1, where 1 is the best score [14]. According to Lapata [14], this metric correlates reliably with human ratings and reading times. It is calculated as follows:

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

where n is the number of passages to order and i is the number of inversions, i. e., the minimum number of adjacent transpositions needed to obtain the gold order from the prediction.

4.3 Baselines

We compare our models with four other methods on arXiv, VIST, and ROCStory. We did not reimplement these methods but take their scores from Wang and Wan [31]. To evaluate our models on Wikipedia and CNN-DailyMail, we implemented a fifth baseline called LSTM+Attention.

LSTM+Pairwise Chen et al. [4] implemented a pairwise ordering model using LSTMs.

LSTM+Pointer Gong et al. [8] proposed an encoder-decoder using LSTMs with a pointer network following the original implementation by Vinyals et al. [30].

LSTM+Set2Seq Logeswaran et al. [17] use a model similar to LSTM+Pointer with attention in the encoder part of the model.

HierAttention The model presented by Wang and Wan [31] uses LSTMs with attention for sentence representations, a multi-head self-attention encoder-decoder at the sentence level and a pointer network using scaled dot-product attention.

B-TSort The model presented by Prabhumoye et al. [20] uses BERT to classify the relative order between two sentences and then a topological sorting algorithm to obtain the final order.

BERSON Cui et al. [6] uses BERT to build a high-level representation for each input sentence, a self-attention based paragraph encoder for encode the all input text, and a relational pointer decoder to order input sentences.

LSTM+Attention We implement a baseline inspired by Wang and Wan [31] that uses LSTMs for sentence representations, a multi-head self-attention encoder-decoder and the Simple Pointer presented in Section 3.4.1.

4.4 Implementation

We implement and train our models using Huggingface Transformers [32]. We use the BASE version of BART that allows up to 1024 input tokens. We use the Adam Optimizer with fixed weight decay and an initial learning rate of 1^{-5}. We train our models with a dropout of 0.1 on fully connected layers of BART and without dropout for the pointer networks. We set the number of neuron layers of the Deep Pointer l to 4 and the number of pointer heads of the Ensemble Pointer h to 12. We train the models until the evaluation loss stops decreasing which was 6 epochs for VIST, 4 for arXiv, ROCStory and CNN-DailyMail, and 3 epochs for the Wikipedia dataset. For inference, we do not use beam search because its accuracy improvements do not justify its increased run time.

For the baseline implementation, we use PyTorch and the same configuration presented in Wang and Wan [31]. We train the baseline for 10 epochs on each dataset. Our source code and trained models is publicly available[1].

5 RESULTS

5.1 BART for passage ordering

We compare our three models (BART with the three different pointer networks) with seven other models including our baseline. The results on arXiv, VIST, and ROCStory are shown in Table 2.

Our models outperform the baselines on arXiv, BERSON and B-TSort obtain better results on VIST, BERTSON outperforms our models on ROCStory. BART + Ensemble Pointer is better than the previous state of the art (BERSON) by 0.0570 Kendall's Tau and by 0.0678 PMR on arXiv. However, BERSON is better than our best model by 0.1070 Kendall's Tau and by 0.1578 PMR on VIST, and by 0.0784 Kendall's Tau and by 0.1821 PMR on ROCStory.

In addition to the comparison with the best baseline, we investigate the effect of BART as encoder-decoder for passage ordering. Our experiments show that BART + Simple Pointer is better than HierAttention which uses the same architecture as BART + Simple Pointer with the only change being the usage of LSTMs with attention model as encoder-decoder. Therefore, by simply replacing the LSTMs with attention encoder-decoder with BART, we observe a significant improvement.

Regarding the experiments on our two new datasets, we compare our models with the baseline we implemented. Table 3 shows the results. Our models considerably outperform the baseline we implemented.

Table 2: Results of the seven baselines and our three models on the arXiv, VIST and ROCStory datasets. BART + Ensemble Pointer is the best model on the arXiv dataset while BERSON is better on the two 5 sentences datasets. The best and second-best results are in bold and underlined respectively.

Methods	arXiv		VIST		ROCStory	
	τ	PMR	τ	PMR	τ	PMR
random	0	0.0827	0	0.0083	0	0.0083
LSTM + Pairwise [4]	0.6594	0.3343	–	–	–	–
LSTM + Pointer [8]	0.7158	0.4044	0.4842	0.1234	–	–
LSTM + Set2Seq [17]	0.7281	0.4157	0.4919	0.1380	0.7112	0.3581
HierAttention [31]	0.7536	0.4455	0.5021	0.1501	0.7322	0.3962
B-TSort [20]	–	–	<u>0.6000</u>	<u>0.2032</u>	–	–
BERSON [6]	<u>0.8300</u>	<u>0.5606</u>	**0.6500**	**0.3169**	**0.8800**	**0.6823**
LSTM + Attention	0.6979	0.3793	0.4590	0.1163	0.6893	0.3044
BART + Simple Pointer	0.8834	0.6274	0.4744	0.1422	0.8071	0.4896
BART + Deep Pointer	0.8810	0.6145	0.4838	0.1021	0.8007	0.4785
BART + Ensemble Pointer	**0.8870**	**0.6284**	0.5430	0.1591	<u>0.8016</u>	<u>0.5002</u>

Table 3: Results of the baseline we implemented and our three models on the Wikipedia and CNN-DailyMail datasets. Our Simple Pointer and Ensemble Pointer models obtain the best results, followed by BART + Deep Pointer and our baseline.

Methods	Wikipedia		CNN-DM	
	τ	PMR	τ	PMR
random	0	0.0308	0	0.0008
LSTM + Attention	0.5668	0.2101	0.2960	0.0049
BART + Simple Ptr.	0.6635	0.2969	0.4765	**0.0171**
BART + Deep Ptr.	0.5679	0.2449	0.4126	0.0067
BART + Ensemble Ptr.	**0.6715**	**0.3010**	**0.4768**	0.0160

5.2 Pointer networks

In addition to using BART, we also compare the different pointer networks on the five datasets (Table 2 and Table 3). Two major observations can be made. First, Deep Pointer yields the lowest scores compared to Simple Pointer and Ensemble Pointer on the Wikipedia and CNN-DailyMail datasets. However, Deep Pointer obtains results equivalent to the other two pointer networks on arXiv and ROCStory and to Simple Pointer on VIST. Second, Simple Pointer and Ensemble Pointer perform similarly well (difference of 1.6% on average) on all datasets. Only VIST is an exception, where Ensemble Pointer outperforms Simple Pointer by 0.0686 Kendall's Tau (+14%) and by 0.0169 PMR (+12%). To conclude, despite the fact that the three pointer network versions have similar results, Ensemble Pointer is performing best on average, followed by Simple Pointer and Deep Pointer.

To analyse each head of the pointer network, we simulate the results of each individual head as if it were a Simple Pointer. We conduct this experiment on all datasets and observe similar results. Each head alone yields lower scores than Simple Pointer. Moreover, all heads obtain results different on a majority of samples in the test set. Indeed, if we manually select the head with the best Kendall's Tau for each example, we obtain better results. For instance, doing this on the CNN-DailyMail dataset gives a Kendall's Tau of 0.6501 and a PMR of 0.0364 that is an increase of 0.1733 with respect to the Kendall's Tau and of 0.0204 with respect to the PMR compared to Ensemble Pointer. On arXiv, the result is a Kendall's Tau of 0.9582 and a PMR of 0.7835 that is an increase of 0.0712 with respect to the Kendall's Tau and of 0.1551 with respect to the PMR compared to Ensemble Pointer.

6 DISCUSSION

The most recent pre-trained Transformers are a recent breakthrough that have demonstrated their potential [7, 15, 22] by learning some linguistic concepts before enabling fine-tuning on a specific task. As passage ordering aims to maximise the linguistic concept of discourse coherence, the passage ordering models that make use of the pre-trained Transformers (BERSON, B-TSort, ours) are logically those obtaining the best results. In the particular case of BART, the sentence permutation pre-training objective seemed a valuable contribution to the passage ordering task. The comparison of the empirical results of HierAttention and our models (Section 5.1) confirms the positive effect of the pre-training of BART.

BERSON [6] represents a passage according to its relation with each of the other passages one by one using relative passages order information, and uses BERT to provide this relative passages order information. In contrast to that, our models represent a passage based on the whole context directly using BART's encoder. We observe in Section 5.1 that BERSON is performing better than our BART + Pointer models on the VIST and the ROCStory datasets, despite our models are better on arXiv. VIST and ROCStory both have 5 passages to order compared to 5.38 for arXiv, and the passages are on average 2.6x shorter in VIST and ROCStory than in arXiv. We hypothesise that the difference of shape between the datasets can

Simple Pointer > Ensemble Pointer		Ensemble Pointer > Simple Pointer	
p_1	Aunt Harriot had a little trouble deciding what kind of wine she wanted tonight.	p_1	Bill wonders who has the ball.
p_2	The restaurant we chose had amazing food and everyone loved the presentation.	p_2	Tim's waiting for the baseball to come his way.
p_3	Gemma really adored the restaurants decorations and was always gazing at them.	p_3	The family is having a party in the park.
p_4	The family sits together for dinner on the first night of the annual reunion.	p_4	Everyone is too tired to pack up and go home.
p_5	Bob had the whole family cracking up with his jokes.	p_5	Cindy is ready for action.
	$O = \{4, 2, 3, 1, 5\}$ $P_{\text{simple}} = \{4, 2, 1, 3, 5\}$ $P_{\text{ensemble}} = \{4, 3, 1, 5, 2\}$		$O = \{3, 1, 5, 2, 4\}$ $P_{\text{simple}} = \{2, 3, 5, 1, 4\}$ $P_{\text{ensemble}} = \{3, 1, 5, 2, 4\}$

Figure 3: Two examples of the VIST dataset with the gold order (O) and predictions by BART + Simple Pointer (P_{simple}) and by BART + Ensemble Pointer (P_{ensemble}). We picked one example for which Simple Pointer performs better than Ensemble Pointer and one where it is the other way around.

explain the difference in the results. Indeed, our models utilize the whole context to represent the passages and this can be especially beneficial for capturing complex semantic dependencies that can be found in longer texts like in arXiv. On the other hand, BERSON, by ordering the passages between them, works best when ordering rules are made at passage level, i. e. passages can be ordered in pairs without taking into account the full meaning of the input. We assume that this advantage of our models to use the whole input context, is particularly relevant for the two new datasets we created. Indeed, the Wikipedia dataset have long passages (62.22 words on average against 10 for ROCStory) and the CNN-DailyMail dataset have 14.5 passages to order on average. A comparison of our models with BERSON on the Wikipedia and the CNN-DailyMail datasets might help support our hypothesis, unfortunately we could not obtain a copy of the code associated with Cui et al. [6].

The results presented in Section 5.2 demonstrate that Ensemble Pointer works in the desired ensemble-like fashion where the heads are the classifiers and the weighted sum is the way to combine these classifiers. Indeed, each head (i. e., classifier) is not as good individually as their combination, which corresponds to the definition of an ensemble method. We also found that if the heads were perfectly combined (the best head is chosen for each example), the performance of the resulting model could be improved further. Thus, we hypothesize that each head learned a different way of ordering passages and that the combination of the different ordering ways corresponds to the average way of ordering that best matches the correct order. This hypothesis could also explain the fact that Ensemble Pointer is better on VIST but not on the other datasets. As presented in Section 4.1, the examples from VIST were originally associated with images and can be, when isolated from their images, ambiguous, which makes multiple orders possible. The Ensemble Pointer, by exploring multiple order possibilities through its heads, may produce better results than Simple Pointer on ambiguous datasets such as VIST. To illustrate the ambiguity, we show two examples from VIST in Figure 3 with predictions of our BART + Simple/Ensemble Pointer models. On these two specific examples, multiple orders make sense, which explains that Simple Pointer and Ensemble Pointer predict different orders. On the other datasets, the advantage of exploring multiple ordering ways offered by the ensemble method is compensated by easier training when there is only one pointer head (e. g., Simple Pointer), which makes Simple Pointer more efficient than each head from Ensemble Pointer individually. Finally, Deep Pointer performs worse than the other two pointer networks in most datasets. This demonstrates that adding more trainable parameters when creating the key and query of the pointer network does not automatically increase model performance. On the contrary, adding parameters makes training more difficult, which explains the worse results of Deep Pointer.

7 CONCLUSION

In this paper, we present new passage ordering models making use of the most recent advances in language modeling by using BART as the encoder-decoder model. On top of BART, we compare three pointer networks.

We conduct experiments on three existing datasets and on two new datasets that we created specifically to explore additional ordering scenarios. These new datasets, which contain either more or longer passages to order, have been designed to fit the actual prototype application in our project use case, i. e., ordering text segments related to the same topic and taken from different document sources, which can vary substantially in number as well as in length. The results show that BART is performing better than previous models with an improvement of 0.057 τ compared to the previous state of the art on the arXiv dataset. However, BERSON achieves 0.0927 better τ on average compared to our best model on the VIST and the ROCStory datasets, which can be explained by the limited nature of these two datasets, which contain only five passages to order.

Furthermore, our experiments on the three pointer network variants show that Ensemble Pointer outperforms the previous Simple Pointer for some datasets. In addition, our investigation on Ensemble Pointer shows that, with a better combination of the heads, the resulting model can largely outperform the models we

presented in this paper. These results are promising. We will explore them further in terms of future work.

ACKNOWLEDGMENTS

This research is funded by the German Federal Ministry of Education and Research (BMBF) through the "Unternehmen Region", instrument "Wachstumskern" QURATOR (grant no. 03WKDA1A). We would like to thank the anonymous reviewers for comments on an earlier version of this manuscript.

REFERENCES

[1] Regina Barzilay, Noemie Elhadad, and Kathleen R. McKeown. 2001. Sentence Ordering in Multidocument Summarization. In *Proceedings of the First International Conference on Human Language Technology Research* (San Diego) *(HLT '01)*. Association for Computational Linguistics, USA, 1–7. https://doi.org/10.3115/1072133.1072217

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

[3] Yoshua Bengio and Yann LeCun (Eds.). 2015. *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. https://iclr.cc/archive/www/doku.php%3Fid=iclr2015:accepted-main.html

[4] Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2016. Neural Sentence Ordering. *arXiv* (2016), arXiv–1607.

[5] Baiyun Cui, Yingming Li, Ming Chen, and Zhongfei Zhang. 2018. Deep Attentive Sentence Ordering Network. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 4340–4349. https://doi.org/10.18653/v1/D18-1465

[6] Baiyun Cui, Yingming Li, and Zhongfei Zhang. 2020. BERT-enhanced Relational Sentence Ordering Network. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online, 6310–6320. https://doi.org/10.18653/v1/2020.emnlp-main.511

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[8] Jingjing Gong, Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2016. End-to-End Neural Sentence Ordering Using Pointer Network. *arXiv* (2016), arXiv–1611.

[9] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[10] Ting-Hao Kenneth 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 *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, San Diego, California, 1233–1239. https://doi.org/10.18653/v1/N16-1147

[11] Monisha Kanakaraj and Ram Mohana Reddy Guddeti. 2015. NLP based sentiment analysis on Twitter data using ensemble classifiers. In *2015 3rd International Conference on Signal Processing, Communication and Networking (ICSCN)*. IEEE, Chennai, India, 1–5. https://doi.org/10.1109/ICSCN.2015.7219856

[12] Vijay Kotu and Bala Deshpande. 2015. Chapter 2: Data Mining Process. *Predictive Analytics and Data Mining. Elsevier* (2015), 26.

[13] Mirella Lapata. 2003. Probabilistic Text Structuring: Experiments with Sentence Ordering. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1* (Sapporo, Japan) *(ACL '03)*. Association for Computational Linguistics, USA, 545–552. https://doi.org/10.3115/1075096.1075165

[14] Mirella Lapata. 2006. Automatic Evaluation of Information Ordering: Kendall's Tau. *Computational Linguistics* 32, 4 (2006), 471–484.

[15] 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 *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, 7871–7880. https://doi.org/10.18653/v1/2020.acl-main.703

[16] Jiwei Li and Dan Jurafsky. 2017. Neural Net Models of Open-domain Discourse Coherence. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, 198–209. https://doi.org/10.18653/v1/D17-1019

[17] Lajanugen Logeswaran, Honglak Lee, and Dragomir Radev. 2018. Sentence Ordering and Coherence Modeling using Recurrent Neural Networks. *Proceedings of the AAAI Conference on Artificial Intelligence* 32, 1 (Apr. 2018). https://ojs.aaai.org/index.php/AAAI/article/view/11997

[18] 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 *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, San Diego, California, 839–849. https://doi.org/10.18653/v1/N16-1098

[19] Robi Polikar. 2006. Ensemble based systems in decision making. *IEEE Circuits and systems magazine* 6, 3 (2006), 21–45.

[20] Shrimai Prabhumoye, Ruslan Salakhutdinov, and Alan W Black. 2020. Topological Sort for Sentence Ordering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, 2783–2792. https://doi.org/10.18653/v1/2020.acl-main.248

[21] A. Radford. 2018. Improving Language Understanding by Generative Pre-Training.

[22] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research* 21, 140 (2020), 1–67. http://jmlr.org/papers/v21/20-074.html

[23] Georg Rehm, Peter Bourgonje, Stefanie Hegele, Florian Kintzel, Julian Moreno Schneider, Malte Ostendorff, Karolina Zaczynska, Armin Berger, Stefan Grill, Sören Räuchle, Jens Rauenbusch, Lisa Rutenburg, Andre Schmidt, Mikka Wild, Henry Hoffmann, Julian Fink, Sarah Schulz, Jurica Seva, Joachim Quantz, Joachim Böttger, Josefine Matthey, Rolf Fricke, Jan Thomsen, Adrian Paschke, Jamal Al Qundus, Thomas Hoppe, Naouel Karam, Frauke Weichhardt, Christian Fillies, Clemens Neudecker, Mike Gerber, Kai Labusch, Vahid Rezanezhad, Robin Schaefer, David Zellhöfer, Daniel Siewert, Patrick Bunk, Lydia Pintscher, Elena Aleynikova, and Franziska Heine. 2020. QURATOR: Innovative Technologies for Content and Data Curation. In *Proceedings of QURATOR 2020 – The conference for intelligent content solutions. Conference on Digital Curation Technologies (QURATOR-2020), January 20-21, Berlin, Germany*, Adrian Paschke, Clemens Neudecker, Georg Rehm, Jamal Al Qundus, and Lydia Pintscher (Eds.). CEUR Workshop Proceedings. Volume 2535.

[24] Georg Rehm, Karolina Zaczynska, Julián Moreno-Schneider, Malte Ostendorff, Peter Bourgonje, Maria Berger, Jens Rauenbusch, André Schmidt, and Mikka Wild. 2020. Towards Discourse Parsing-inspired Semantic Storytelling. *arXiv e-prints* (2020), arXiv–2004.

[25] Julian Risch, A. Stoll, Marc Ziegele, and Ralf Krestel. 2019. hpiDEDIS at GermEval 2019: Offensive Language Identification using a German BERT model. In *KONVENS*. KONVENS, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany.

[26] Lior Rokach. 2010. Ensemble-based classifiers. *Artificial intelligence review* 33, 1-2 (2010), 1–39.

[27] Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, 1073–1083. https://doi.org/10.18653/v1/P17-1099

[28] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2* (Montreal, Canada) *(NIPS'14)*. MIT Press, Cambridge, MA, USA, 3104–3112.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 6000–6010.

[30] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer Networks. In *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (Eds.), Vol. 28. Curran Associates, Inc., Palais des Congrès de Montréal, Montréal CANADA. https://proceedings.neurips.cc/paper/2015/file/29921001f2f04bd3baee84a12e98098f-Paper.pdf

[31] Tianming Wang and Xiaojun Wan. 2019. Hierarchical Attention Networks for Sentence Ordering. *Proceedings of the AAAI Conference on Artificial Intelligence* 33, 01 (Jul. 2019), 7184–7191. https://doi.org/10.1609/aaai.v33i01.33017184

[32] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv* abs/1910.03771 (2019).