The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)

Hierarchical Attention Networks for Sentence Ordering

Tianming Wang, Xiaojun Wan
Institute of Computer Science and Technology, Peking University
The MOE Key Laboratory of Computational Linguistics, Peking University
{wangtm,wanxiaojun}@pku.edu.cn

Abstract

Modeling discourse coherence is an important problem in natural language generation and understanding. Sentence ordering, the goal of which is to organize a set of sentences into a coherent text, is a commonly used task to learn and evaluate the model. In this paper, we propose a novel hierarchical attention network that captures word clues and dependencies between sentences to address this problem. Our model outperforms prior methods and achieves state-of-the-art performance on several datasets in different domains. Furthermore, our experiments demonstrate that the model performs very well even though adding noisy sentences into the set, which shows the robustness and effectiveness of the model. Visualization analysis and case study show that our model captures the structure and pattern of coherent texts not only by simple word clues but also by consecution in context.

Introduction

Modeling discourse coherence is an essential problem in NLP as evidenced by its importance on many downstream tasks like summarization, question answering and text planning. Sentence ordering is a commonly used task to build and evaluate such models. The goal of it is to organize a given set of sentences into a coherent text in a clear and consistent manner.

The task requires models to learn which ordering of sentences is likely to enhance understanding and avoid confusion. By learning to order sentences, models can characterize the properties which make text coherent, such as topical relevancy, chronological sequence and cause-effect. Thus ordering models can help to generate a coherent text in other tasks like multi-document summarization since the relative ordering of the sentences extracted from different documents might be unclear.

A variety of researches tackling coherence have been done. Lexical cohesion focuses on chains of related words that contribute to the continuity of lexical meaning (Morris and Hirst 1991). Mann and Thompson (1988) define relations called RST that are used in linking successive sentences. Lapata (2003) represents sentences by vectors and learns the transition probabilities of adjacent sentences. The Entity-Grid model captures local coherence by modeling entities transitions between adjacent sentences (Barzilay and Lapata 2008). Most recent works focus on addressing the sentence ordering task. Chen, Qiu, and Huang (2016) and Agrawal et al. (2016) propose pair-wise models and use beam search to seek the ground truth order. Li and Jurafsky (2017) employ the graph-based model to address this problem. The end-to-end approach using RNN based pointer network outperforms prior methods and achieves a great success (Gong et al. 2016; Logeswaran, Lee, and Radev 2018). But the vanilla pointer network has an obvious disadvantage that it treats the out-of-order set of sentences as a sequential input, which is contrary to intuition.

In this paper, we propose a novel hierarchical attention network which captures word clues and dependencies between sentences to address the sentence ordering task. Prior neural approaches rarely take the advantage of word clues in sentence encoding and usually encode each sentence individually, which ignores that the semantic of the sentence in a coherent text is dependent on the context. We design two levels of attentions in our model to tackle these problems - one at the word level and one at the sentence level. We first use a LSTM-based word encoder to get an initial representation of each sentence and a word attention is employed over it, which lets the model to pay more attention to word clues when encoding sentences. Then we use a sentence encoder to capture the global dependencies between sentences and adjust their representations by a sentence self-attention mechanism. The sentence encoder is RNN-free and composed of stacked attention networks, which is well adapted to the situation where input sentences are unordered. Finally we use a sentence decoder utilizing the information of ordered subsequence to select the next sentence one by one by pointer mechanism.

Our model strongly outperforms prior methods and achieves state-of-the-art performance on several datasets in different domains. We further evaluate the robustness and effectiveness of our model by adding unrelated noisy sentences to the given sentences set. Both noise discrimination and sentence ordering on noisy data are studied. The results show that our model can discriminate unrelated sentence with a very high accuracy and achieves excellent performance on the sentence ordering task. Finally, through visualization analysis and case study, we show that our model

captures the structure and pattern of coherent texts not only by simple word clues but also by consecution in context. In summary, our key contributions are as follows:

- We propose a novel hierarchical attention network, which captures both word clues and dependencies between sentences, to tackle the problem of organizing a set of sentences into a coherent text.
- Our model achieves the state-of-the-art performance on several datasets in different domains and the experiment results on noisy data show the robustness and effectiveness of the model.
- The results of visualization analysis indicate that our model captures the structure and pattern of coherent texts.

Related Work

Discourse Coherence

Coherence is an essential aspect in NLP. Various coherence theories have been developed over years. Mann and Thompson (1988) call a text coherent when it can be explained what role each clause plays with regard to the whole. Rhetorical Structure Theory(RST) defines about 25 relations that organizes text structure (Mann and Thompson 1988). Centering Theory extensively studies local coherence at the early time (Grosz, Weinstein, and Joshi 1995; Walker and Prince 1998; Strube and Hahn 1999; Poesio et al. 2004). Centering Theory claims that entities in coherent discourses exhibit certain regularities. But centering approaches suffer from a severe dependency on manual annotation. Motivated from centering theory, the Entity-Grid model is proposed, which represents sentences by vectors of entities appearing in the document along with their syntactic roles (Barzilay and Lapata 2008). Global graph models typically use HMMs to model coherence between adjacent sentences (Barzilay and Lee 2004; Louis and Nenkova 2012).

Two typical tasks are commonly used to build and evaluate models that understand coherence. One is a discrimination task that identifies the more coherent ordering given a document and a permuted version of it. Another is the sentence ordering task that arranges a set of sentences into a coherent text. The latter is more challenging and meaningful and attracts more attention recently. The RNN based pointer network has the most outstanding performance on the ordering task (Gong et al. 2016; Logeswaran, Lee, and Radev 2018). These neural approaches use a word-level RNN encoder to produce sentence representations and a sentence-level RNN encoder to construct context representation. A sentence-level decoder is designed to select the sentences sequentially.

RNN/CNN-Free Network

Recurrent neural networks and convolution neural networks have been widely used on NLP tasks as their advantages at capturing the long-term and local dependency respectively. Compared to these networks, attention mechanism shows superiority on parallelism and flexibility in modeling dependencies. Attention-based but RNN/CNN-free networks have attracted more interests recently. Vaswani et al. (2017) propose an encoder-decoder structure model called "Transformer", which is solely based on attention mechanism, on the neural machine translation task. Gu et al. (2018) introduce a non-autoregressive translation model based on the Transformer network. Shen et al. (2018) propose a model named Directional Self-Attention Network to produce sentence representation for natural language inference and sentiment analysis tasks.

Inspired by the Transformer network, we employ stacked attention networks with fusion gate, which is designed to fuse local and global information, as the sentence encoder-decoder in our model. Note that our whole model is not RNN/CNN-free.

Our Approach

Our model is a hierarchical structure composed of a word encoder and a sentence encoder-decoder. The word encoder is used to learn the sentence embedding and the sentence encoder-decoder is used to adjust representations of sentences in context and arrange these sentences into a coherent text. We begin by formulating the sentence ordering task. Our full model is described in parts for easy understanding. We will describe the word encoder and the word attention employed for getting the sentence representation. Then the sentence encoder for capturing global dependencies between sentences is described. Finally we will introduce the decoder utilizing the information captured by the encoder and predicting the next sentence one by one. Figure 1 shows the overall architecture of our model.

Task Description

Given an out-of-order set of N sentences $\{s_1, s_2, ..., s_N\}$, sentence s_i is denoted by $s_i = \{w_{i,1}, w_{i,2}, ..., w_{i,T_i}\}$, where T_i is the number of words in the sentence and $w_{i,j}$ is the j-th word. Our goal is to find the gold order O for these sentences. O is denoted as $O = \{o_1, o_2, ..., o_N\}$, where s_{o_i} denotes the i-th sentence in the coherent text. Our model is trained to maximize the probability:

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

Word Encoder

Given a sentence s_i with words $w_{i,t}$, $t \in [0, T_i]$, we first embed the words to vectors $x_{i,t}$ through an embedding matrix W_e. We use a bidirectional LSTM (Hochreiter and Schmidhuber 1997) to encode contextual information for words from both directions. The bidirectional LSTM contains a forward LSTM $\overrightarrow{f}$ which reads the sentence from $w_{i,1}$ to w_{i,T_i} and a backward LSTM $\overleftarrow{f}$ which reads from w_{i,T_i} to $w_{i,1}$:

$$\begin{aligned} x_{i,t} &= w_{i,t} W_e \\ \overrightarrow{h}_{i,t} &= \overrightarrow{LSTM}(x_{i,t}) \\ \overleftarrow{h}_{i,t} &= \overleftarrow{LSTM}(x_{i,t}) \end{aligned} \tag{2}$$

where $\overrightarrow{h}_{i,t}$ and $\overleftarrow{h}_{i,t}$ are the hidden states of the LSTMs. We concatenate them and get the contextual representation

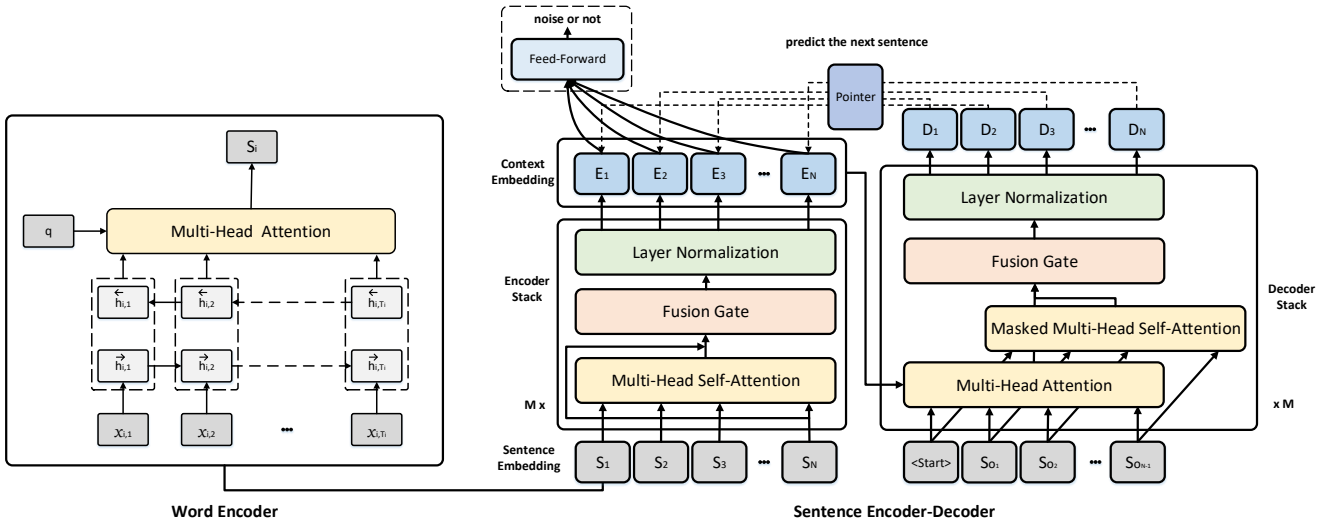

Figure 1: Architecture of our approach.

$h_{i,t} = [\overrightarrow{h}_{i,t}; \overleftarrow{h}_{i,t}]$ for word $w_{i,t}$.

Observing that some keywords like "first" and "then" provide clues for ordering, we employ a word attention, which allows the word encoder to pay more attention to these word clues. We use a modified multi-head attention in our model, and its original version is proposed by Vaswani et al. (2017). Specifically, we project the d-dimensional queries, keys and values H times with different non-linear projections to d/H dimensions respectively. On each projected subspace, we perform the dot-product attention with a scaling factor of $1/\sqrt{d/H}$. These outputs are concatenated together as the final output of attention. The original multi-head attention uses lineal projections and projects once again after concatenated, which is different from ours[1].

$$\begin{aligned} \hat{Q}^j &= ReLU(QW_Q^j) \\ \hat{K}^j &= ReLU(KW_K^j) \\ \hat{V}^j &= ReLU(VW_V^j) \\ \hat{C}^j &= softmax(\frac{\hat{Q}^j \hat{K}^{j\top}}{\sqrt{d/H}})\hat{V}^j \\ C &= [\hat{C}^1; \hat{C}^2; ...; \hat{C}^H] \end{aligned} \tag{3}$$

where Q, K, V are the packages of a set of queries, keys and values, $W_Q^j, W_K^j, W_V^j \in \mathbb{R}^{d \times d/H}$ are parameter matrices, $\hat{C}^j$ is the output of attention in the j-th subspace and $j \in [1, H]$. For convenience, we denote multi-head attention as $C = MultiHead(Q, K, V)$.

For the word attention, we use a context vector q (random initialized) as a single query and the hidden states $h_{i,t}$ are packed as keys and values simultaneously. The output C of the word attention is thus a single vector, which is regarded as the embedding of the sentence s_i. We pack all the embeddings of the sentences in the set together into a matrix S.

[1]The modified attention mechanism has a more stable performance than the vanilla one for our task.

Sentence Encoder

Our sentence encoder is composed of a stack of M attention layers. Each layer has three sub-layers. Multi-head self-attention mechanism is used for capturing the global dependencies between the sentences in the set. We further employ a fusion gate to combine the input and output of the attention layer, which yields a self-aware and global-aware vector representation for each sentence. Layer normalization (Mullery and Whelan 2018) is implemented as the last part of the layer. We denote the input and output of the j-th layer as E_{in}^j and E_{out}^j respectively. Particularly, the embeddings of the sentences S are fed as E_{in}^1 of the first layer, E_{out}^M is the final output of the encoder stack. For convenience, we abbreviate E_{out}^M as E in Figure 1.

$$\begin{aligned} E_{in}^j &= E_{out}^{j-1} \\ C &= MultiHead(E_{in}^j, E_{in}^j, E_{in}^j) \\ G &= sigmoid(E_{in}^j W_{in}^j + CW_{out}^j) \\ F &= GE_{in}^j + (1-G)C \\ E_{out}^j &= LayerNorm(F) \end{aligned} \tag{4}$$

where $W_{in}^j, W_{out}^j \in \mathbb{R}^{d \times 1}$ are parameter matrices and $Layernorm$ is the layer normalization which we omit the formula here.

For noise discrimination, we employ an extra layer to judge for each sentence whether it is irrelevant to others. A feed-forward network is used for prediction:

$$P^* = softmax(ReLU(E_{out}^M W_{f1})W_{f2}) \tag{5}$$

where P^* is the output of the feed-forward network and $W_{f1} \in \mathbb{R}^{d \times d}, W_{f2} \in \mathbb{R}^{d \times 2}$ are parameter matrices. Specif-

ically, P_i^* is the probability distribution of whether sentence s_i is a noise or not.

Sentence Decoder

Our sentence decoder is also composed of a stack of M attention layers. Different from the encoder stack, we employ two attention mechanisms in the decoding phase.

One is the masked multi-head self-attention. We need to prevent earlier decoding steps from accessing information from later steps. The masking ensures that the attention and prediction for position y can depend only on the known sentences at positions preceding y. It also allows the decoder to utilize the information of ordered subsequence and construct a context for predicting the next sentence. We modify Eq(3) and denote it as $MaskedMultiHead(Q, K, V)$:

$$\begin{aligned} Mask_{x,y} &= \begin{cases} 0 & x <= y \\ -\infty & otherwise \end{cases} \\ \hat{C}^j &= softmax(\frac{\hat{Q}^j \hat{K}^{j\top} + Mask}{\sqrt{d/H}})\hat{V}^j \\ C &= [\hat{C}^1; \hat{C}^2; ...; \hat{C}^H] \end{aligned} \tag{6}$$

where $Mask$ is the mask and $x, y \in [1, N]$.

Another is the global attention, which utilizes the global information captured by the encoder and allows the decoder to model global dependencies in the context. These attention outputs are fed to a fusion gate, followed by a layer normalization. Similar to the encoder, this yields

$$\begin{aligned} D_{in}^j &= D_{out}^{j-1} \\ C &= MaskedMultiHead(D_{in}^j, D_{in}^j, D_{in}^j) \\ C^* &= MultiHead(D_{in}^j, E_{out}^M, E_{out}^M) \\ G &= sigmoid(CW_{in}^j + C^* W_{out}^j) \\ F &= GC + (1 - G)C^* \\ D_{out}^j &= LayerNorm(F) \end{aligned} \tag{7}$$

where $j \in [1, M]$ and D_{in}^j and D_{out}^j are the input and output of the j-th layer of the decoder. Particularly, S is fed as D_{in}^1 and D_{out}^M is the final output of the decoder stack. For convenience, we abbreviate D_{out}^M as D in Figure 1.

Over the decoder stack, a pointer layer is implemented for predicting the next sentence, which works as

$$\begin{aligned} Q &= ReLU(D_{out}^M W_Q) \\ K &= ReLU(E_{out}^M W_K) \\ P &= softmax(\frac{QK^\top}{\sqrt{d}}) \end{aligned} \tag{8}$$

where P is the output of the pointer layer. Specifically, $P_{i,j}$ represents the probability for the sentence s_j being the correct sentence choice at position i. During inference, we use beam search to select sentences sequentially.

Experiment

Datasets

We evaluate the proposed model on three datasets, the arXiv dataset (Chen, Qiu, and Huang 2016), the VIST dataset (Huang et al. 2016) and the ROCStory dataset (Mostafazadeh et al. 2016).

The **arXiv** dataset is a very large dataset for sentence ordering, which contains 884912 training abstracts, 110614 validation abstracts and 110615 testing abstracts of papers on arXiv website. The abstracts are composed of 2 to 20 sentences and the average word count per abstract is around 135. This dataset is very suitable for the sentence ordering task since the abstracts are well-organized and strict in logic.

The **VIST** dataset is a visual storytelling dataset, which is previously known as "SIND", the Sequential Image Narrative Dataset. We only use story texts for sentence ordering. It includes 40155 training stories, 4990 validation stories and 5055 testing stories. Each story is composed of 5 sentences and its average word count is around 57. The story dataset is also suitable for our task since the story is usually coherent in chronological sequence and cause-effect.

The **ROCStory** dataset is a commonsense story dataset, which contains 98162 stories with 50 words per story on average. Each story is composed of 5 sentences. We randomly split the dataset by 8:1:1 to get the training, validation and testing datasets of 78529, 9816 and 9817 stories respectively. Compared to the VIST dataset, the story in this dataset is more logical since there is no extra image as information supplementary.

Baselines and Metrics

We directly compare the results[2] of several models developed on the first two datasets. We also reproduce two recently proposed models for comparison. Besides, we modifies a prior method as baseline. All the baselines are described as follows:

LSTM+Pairwise Chen, Qiu, and Huang (2016) propose a pairwise ranking model which uses LSTM as the word encoder. It is developed on the arXiv dataset.

SkipThought+Pairwise Agrawal et al. (2016) propose a pairwise model which takes a pair of SkipThought embeddings as input. It is developed on the VIST dataset.

LSTM+Ptr Gong et al. (2016) propose an end-to-end approach based on pointer network to address the task. It treats the out-of-order set of sentences as a sequential input for encoder and predicts the next sentence by pointer mechanism. The model is developed on the arXiv and the VIST dataset.

Seq2Seq+Pairwise Li and Jurafsky (2017) propose a generative model which is trained to predict the next sentence given the current sentence for the sentence ordering task. This sequence-to-sequence method is also pairwise. We reproduce the bi-directional model in the paper as baseline.

LSTM+Set2Seq Logeswaran, Lee, and Radev (2018) also use pointer network to tackle the problem. It uses LSTM

[2]Pairwise metric is used in these papers and it can be converted to Kendall's τ easily.

Table 1: Comparison of results on three datasets

Methods	arXiv		VIST		ROCStory	
	τ	PMR	τ	PMR	τ	PMR
random	0	0.0827	0	0.0083	0	0.0083
LSTM+Pairwise (Chen, Qiu, and Huang 2016)	0.6594	0.3343	-	-	-	-
SkipThought+Pairwise (Agrawal et al. 2016)	-	-	0.4640	-	-	-
LSTM+PtrNet (Gong et al. 2016)	0.7158	0.4044	0.4842	0.1234	-	-
Seq2Seq+Pairwise (Li and Jurafsky 2017)	0.0593	0.1370	0.1892	0.1250	0.3419	0.1793
LSTM+Set2Seq (Logeswaran, Lee, and Radev 2018)	0.7281	0.4157	0.4919	0.1380	0.7112	0.3581
WordAtt+PtrNet	0.7367	0.4210	0.4925	0.1346	0.7024	0.3285
Our	**0.7536**	**0.4455**	**0.5021**	**0.1501**	**0.7322**	**0.3962**

to produce sentence embeddings and constructs a context representation by a sentence-level set encoder which iteratively attends to these embeddings. A sentence-level pointer network selects the sentences sequentially. We reproduce it for comparison.

WordAtt+Ptr We modify the word encoder in Gong et al. (2016)'s model and employ the word attention on it. We treat this method as our last strong baseline.

Evaluating the absolute positions of sentences is too harsh for the sentence ordering task since it does not take the relative position between sentences into account. Through maintaining relative position, local coherence of a text can be manifested. Therefore we use Kendall's τ, a metric of rank correlation for evaluation.

$$\tau = 1 - \frac{2(InvertPairs)}{N(N-1)/2} \tag{9}$$

where N is the number of sentences being ordered and $(InvertPairs)$ is the number of interchanges of consecutive elements necessary to arrange them in their natural order. Besides, we also use Perfect Match Ratio(PMR) as a metric, which calculates the ratio of cases of exact match of the whole sequence.

Setup

We use Tensorflow to implement our model. We use the Adam Optimizer with an initial learning rate of 10^{-4}, momentum $\beta_1 = 0.9$, $\beta_2 = 0.98$ and weight decay $\epsilon = 10^{-9}$. We set batch size to 64 and stop training when the metric Kendall's τ on the validation set does not improve for 3 epochs. The size of hidden states of LSTM is set to 300 and dimension d is set to 600. The head of attention H is set to 4 and the number of layers M is set to 3. We apply dropout to the output of each multi-head attention sub-layer. We use a rate $P_{drop} = 0.05$ for the arXiv dataset and $P_{drop} = 0.15$ for the VIST and the ROCStory datasets. We use beam search for all models with a beam size of 16 and initialize all models (include baselines) with 300-dimensional GloVe (Pennington, Socher, and Manning 2014) word vectors.

Results and Analysis

Table 1 shows the results of all methods on three datasets. Among all prior methods, LSTM+Set2Seq has the best performance. We can see that our model strongly outperforms it and achieves the state-of-the-art scores on all datasets. On the arXiv dataset, our model improves the state of art from 72.81% to 75.36% on τ and achieves 2.98% improvement on PMR. On the VIST dataset and the ROCStory dataset, our model also outperforms LSTM+Set2Seq by 1.02% and 2.10% on τ, and 1.21% and 3.81% on PMR respectively. In general, end-to-end models outperform pairwise models. Seq2Seq+Pairwise performs badly, especially on the arXiv dataset due to the data diversity and longer sentences. WordAtt+PtrNet performs better than LSTM+Ptr, which indicates that the word attention can improve the quality of sentence representations. Comparing the results on different datasets, we can see that our proposed methods perform better on the arXiv dataset than on the VIST dataset and the ROCStory dataset. It might be caused by that abstracts of papers are more rigorous and contain more word clues than stories and arranging sentences of the story requires a stronger capability of reasoning. In terms of the story datasets , there is a clear gap between the results on VIST and ROCStory. These two datasets both contain 5-sentence stories but our model achieve 72.43% on the ROCStory dataset against 50.21% on the VIST dataset. This is because that the story in the VIST dataset is accompanied by images. Discarding the visual information makes the ordering task difficult.

Table 2: Variations of our model. Unlisted values are identical to those of base model. All metrics are on the arXiv dataset.

	M	H	P_{drop}	WordAtt	τ	PMR
base	3	4	0.05	yes	**0.7536**	**0.4455**
(A)	2				0.7484	0.4419
	4				0.7515	0.4409
(B)		1			0.7437	0.4368
		2			0.7496	0.4420
		8			0.7481	0.4407
(C)			0		0.7475	0.4413
			0.1		0.7526	0.4442
(D)				no	0.7399	0.4301

To evaluate the importance of the different parameters

Table 3: Performance of noise discrimination.

Strategy	Methods	arXiv	VIST	ROCStory
		acc	acc	acc
1 noise	random	0.1819	0.1667	0.1667
	Our	0.9664	0.8462	0.9382
0/1 noise	random	0.2955	0.5833	0.5833
	Our	0.9330	0.9151	0.9698

and components of our model, we vary the number of the encoder-decoder layers M, the number of attention heads H, the dropout rate P_{drop} and whether to use the word attention(WordAtt)[3]. The result of variations is shown in Table 2. In Table 2 rows(A), we can see that the encoder-decoder stack with 2 layers or 4 layers suffers a loss of 0.52% and 0.21% on τ respectively. We further observe in rows(B) that too few or too many heads both hurt model quality, which is the same situation for dropout. In Table 2(D), we can see that the model's performance drops by 1.37% without word attention. It verifies that word clues help ordering sentence and word attention can let the model pay more attention to these clues. In general, all variations of our model outperform prior methods on both two metrics.

Adding Noise

As mentioned earlier, word clues play an important role in sentence ordering. However if the model learns the structure of coherence only by word clues, it might collapse when we add noisy sentence to the set since the noisy sentence might also contain word clues. To test the robustness and effectiveness of our model, we add a noisy sentence, which is randomly chosen from another abstract or story, into the set. In this case, the model needs to judge for each sentence whether it is relevant and coherent with other sentences in the set. We evaluate our model on such noisy set by two experiments.

First we test our model by a noise discrimination task. We feed the output of our sentence encoder to a classifier and require it to determine whether each sentence is a noisy sentence or not. We compare two different strategies in adding noise: (1) add 1 noisy sentence (1 noise); (2) add 1 noisy sentence with a probability of 50% (0/1 noise), in other words, 50% sets contain 1 noisy sentence. We use the set-level accuracy as the metric, which means the score is 1 only if all the sentences in the set are classified correctly, otherwise 0. Table 3 shows the results on three datasets. Our model shows a strong capability of discriminating noise. We can see that our model achieves very high accuracy scores on the arXiv and ROCStory datasets. On the VIST dataset, the accuracy is relatively lower but still over 84%. This gap is caused by the particularity of this dataset as we mentioned before. In terms of two strategies in adding noise, 0/1 noise seems to be more difficult than 1 noise for our model on the arXiv dataset while the opposite occurs on two story datasets. Since all texts in two story datasets contain 5 sentences, it is easy for model to tell if there is a noisy sentence. All the errors come from discriminating which sentence is noise. But the 0/1 noise case is difficult for the discrimination on the arXiv dataset since the number of sentences in abstracts varies. In general, the results indicate that the sentence encoder of our model can deal with noise while modeling the global dependencies between sentences.

Further we test our model by the sentence ordering task on datasets with noise. In this case, it is required to arrange sentences on the basis of noise discrimination. Since it is not a typical ordering task, Kendall's τ is not suitable for measuring correlation. We introduce Pairwise Metrics (PM), which is the fraction of pairs of sentences whose predicted relative order is the same as the ground truth order. Here we use the F-score of PM, which is denoted as PM_F.

$$\begin{aligned} PM_P &= \frac{(CorrectPairs)}{N^*(N^*-1)/2} \\ PM_R &= \frac{(CorrectPairs)}{N(N-1)/2} \\ PM_F &= \frac{2 * PM_P * PM_R}{PM_P + PM_R} \end{aligned} \tag{10}$$

where $(CorrectPairs)$ is the number of correct pairs, and N^* is the number of sentences in predicted set. In typical ordering task, $PM_F = PM_P = PM_R = (1+\tau)/2$.

Gong et al. (2016) also did this experiment so we compare our model with their proposed model. The strong baseline WordAtt+PtrNet is also evaluated. For comparison, the result on datasets without noise(0 noise) is also presented. Table 4 shows the results. Compared to the results on datasets without noise, the performance of the models drops, which indicates that noisy sentences confuse the models to some extent. But our model still performs comparably to that trained on datasets without noise. On the arXiv dataset, our model only drops by 1.82% and 2.52% on PM-F when adding 1 noise and 0/1 noise respectively. On the ROCStory dataset, our model achieves 38.83% and 38.79% on PMR with 1 noise and 0/1 noise respectively, which are very close to the result without noise. As we can see, our model outperforms all baselines and is more robust than other methods. This experiment verifies that our model can deal with the case well when additional noisy sentence exists.

In general, our model shows its robustness and effectiveness and it performs the ordering task not only by word clues but also by dependencies between sentences.

Visualization Analysis

We now visualize the word attention weights of several cases in the word encoder to show what word clues the model uses to perform the ordering task. The attention weight is calculated as

$$\alpha = \frac{1}{H}\sum_{j=1}^{H} softmax(\frac{\hat{Q}^j \hat{K}^{j\top}}{\sqrt{d/H}}) \tag{11}$$

Darker shades correspond to higher attention weights. Figure 2 shows the visualizations of two abstracts and one story.

[3]If the word attention is discarded, we use the final hidden state of the LSTM as the embedding of sentence.

Table 4: Performance of sentence ordering on three datasets with noise.

Strategy	Methods	arXiv		VIST		ROCStory	
		PM_F	PMR	PM_F	PMR	PM_F	PMR
0 noise	random	0.5000	0.0827	0.5000	0.0083	0.5000	0.0083
	LSTM+PtrNet (Gong et al. 2016)	0.8579	0.4044	-	-	-	-
	WordAtt+PtrNet	0.8683	0.4210	0.7463	0.1346	0.8512	0.3285
	Our	**0.8768**	**0.4455**	**0.7510**	**0.1520**	**0.8661**	**0.3962**
1 noise	random	0.3178	0.0238	0.3357	0.0011	0.3326	0.0010
	LSTM+PtrNet (Gong et al. 2016)	0.8228	0.3733	-	-	-	-
	WordAtt+PtrNet	0.8271	0.3805	0.6432	0.0980	0.7852	0.2930
	Our	**0.8586**	**0.4325**	**0.6992**	**0.1283**	**0.8376**	**0.3883**
0/1 noise	random	0.3830	0.0259	0.4096	0.0049	0.4227	0.0069
	LSTM+PtrNet (Gong et al. 2016)	0.8344	0.3675	-	-	-	-
	WordAtt+PtrNet	0.8407	0.3740	0.6706	0.1064	0.7967	0.3055
	Our	**0.8516**	**0.4094**	**0.6974**	**0.1300**	**0.8293**	**0.3879**

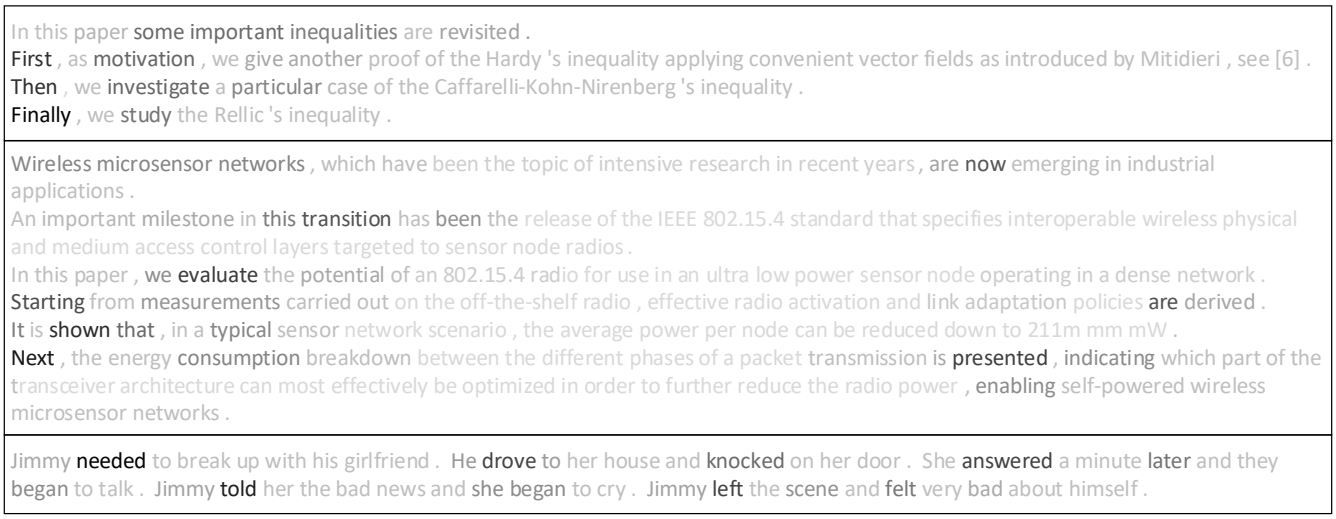

Figure 2: Visualizing word clues

In the first two samples, word clues "first", "then", "next" and "finally" contribute a lot to the representations of sentences. These words give strong signal for ordering. Besides, the model also pays attention to words like "motivation", "investigate", "evaluate" and "show". These clues are implicit. For example, we usually "evaluate" the model first and then "show" the result. In the story example, the model relies heavily on verbs. We guess that verbs represents actions that drive the story and reflect order in a way. Subjects are also helpful, e.g., the subject of the first sentence in a story can not be a personal pronoun. In general, our model can learn patterns from word clues.

Further, we present PCA embeddings of sentence representations learned by the word encoder and the sentence encoder, i.e. S and E_M^{out} on the arXiv dataset in Figure 3. The embeddings are color coded by the positions of the sentences in the abstracts. Figure (a) and (b) present the embeddings S and E_M^{out} without noisy sentence respectively. Figure (c) and (d) present the embeddings S and E_M^{out} with noise respectively. As we can see, both the word encoder and the sentence encoder can learn the structure when no noise exists. When adding noisy sentences to the set, the embeddings of noisy sentences learned by the word encoder are distributed in space uniformly as expected, since these embeddings are context-free. But through context encoding, the structure learned by the sentence encoder is more clear and the embeddings of noises are gathered together. It verifies that our model captures the structure and pattern of coherent texts in context.

Conclusion

We investigate the problem of arranging sentences into a coherent text, which is known as the sentence ordering task. Our proposed hierarchical attention network strongly out-

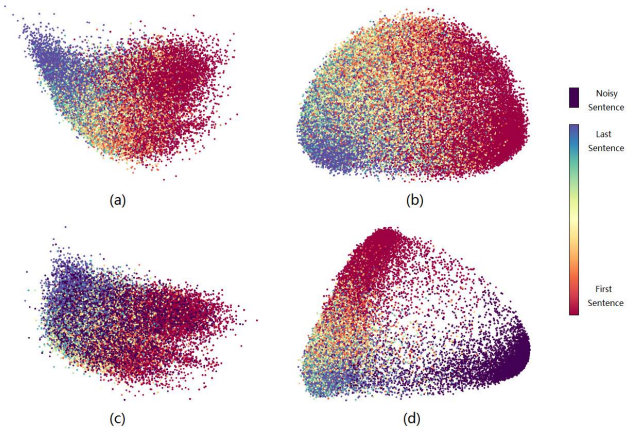

Figure 3: PCA embeddings of sentences from the arXiv dataset.

performs prior methods. We further evaluate our model in the case with noisy sentence and our model achieves excellent performance, which shows the robustness and effectiveness of our model. Our future work will focus on applying model trained on the sentence ordering task to downstream tasks like multi-document summarization and text planning.

Acknowledgments

This work was supported by National Natural Science Foundation of China (61772036, 61331011) and Key Laboratory of Science, Technology and Standard in Press Industry (Key Laboratory of Intelligent Press Media Technology). We thank the anonymous reviewers for their helpful comments. Xiaojun Wan is the corresponding author.

References

Agrawal, H.; Chandrasekaran, A.; Batra, D.; Parikh, D.; and Bansal, M. 2016. Sort story: Sorting jumbled images and captions into stories. *empirical methods in natural language processing* 925–931.

Barzilay, R., and Lapata, M. 2008. *Modeling local coherence: An entity-based approach*. MIT Press.

Barzilay, R., and Lee, L. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. *Computer Science* 113–120.

Chen, X.; Qiu, X.; and Huang, X. 2016. Neural sentence ordering. *arXiv: Computation and Language*.

Gong, J.; Chen, X.; Qiu, X.; and Huang, X. 2016. End-to-end neural sentence ordering using pointer network. *arXiv: Computation and Language*.

Grosz, B. J.; Weinstein, S.; and Joshi, A. K. 1995. *Centering: a framework for modeling the local coherence of discourse*. MIT Press.

Gu, J.; Bradbury, J.; Xiong, C.; Li, V. O. K.; and Socher, R. 2018. Non-autoregressive neural machine translation. *international conference on learning representations*.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural Computation* 9(8):1735–1780.

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

Lapata, M. 2003. Probabilistic text structuring: Experiments with sentence ordering. In *Meeting of the Association for Computational Linguistics*, 545–552.

Li, J., and Jurafsky, D. 2017. Neural net models of open-domain discourse coherence. *empirical methods in natural language processing* 198–209.

Logeswaran, L.; Lee, H.; and Radev, D. R. 2018. Sentence ordering and coherence modeling using recurrent neural networks. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*.

Louis, A., and Nenkova, A. 2012. A coherence model based on syntactic patterns. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 1157–1168.

Mann, W. C., and Thompson, S. A. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text* 8(3):243–281.

Morris, J., and Hirst, G. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics* 17(1):21–48.

Mostafazadeh, N.; Chambers, N.; He, X.; Parikh, D.; Batra, D.; Vanderwende, L.; Kohli, P.; and Allen, J. F. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. 839–849.

Mullery, S., and Whelan, P. F. 2018. Batch normalization in the final layer of generative networks. *CoRR* abs/1805.07389.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, 1532–1543.

Poesio, M.; Stevenson, R. J.; Eugenio, B. D.; and Hitzeman, J. 2004. Centering: A parametric theory and its instantiations. *Computational Linguistics* 30(3):309–363.

Shen, T.; Zhou, T.; Long, G.; Jiang, J.; Pan, S.; and Zhang, C. 2018. Disan: Directional self-attention network for rnn/cnn-free language understanding. *national conference on artificial intelligence*.

Strube, M., and Hahn, U. 1999. *Functional centering: grounding referential coherence in information structure*. MIT Press.

Vaswani, A.; Shazeer, N.; Parmar, N.; Jones, L.; Uszkoreit, J.; Gomez, A. N.; and Kaiser, u. 2017. Attention is all you need. *neural information processing systems* 5998–6008.

Walker, M. A., and Prince, E. F. 1998. *Centering Theory in Discourse*. Clarendon Press , Oxford University Press.