CHAPTER 5
STRUCTURAL SYNCHRONIZATION WITH LONG INPUT SEQUENCE FOR BOOK QA

5.1 Overview

In the previous chapter, we have revealed the event-centric nature of the questions in the book QA task and explored a set of distant supervision methods and event-aware methods for the ranker module in a book QA system. In this chapter, we continue the discussion on the book QA task and show the deficiency of the latest pretrained language models (PLMs) in reasoning over book content and the importance of modeling event structures at various scales to improve the machine reader (Section 1.2.3).

Transformer-based encoders [153] have been widely used in natural language processing, with success. The pretrained language models based on Transformer, such as BERT [176], GPT-3 [178], T5 [104], and BART [103], further make it a dominating architecture in NLP.

Despite their successes, the Transformer models suffer from a major challenge in encoding long sequences because the self-attention mechanism used in each Transformer layer requires computing attention for each pair of input words. Such computations lead to $O(l^2)$ complexity in time and space in each Transformer layer, where l is the sequence length. This limits Transformer's roles in the increasingly important long-sequence encoding for two common scenarios: (1) *encoding a single long document* with lengths exceeding the input limitation, and (2) *joint encoding of multiple related documents* for tasks that require synthesizing scattered pieces of evidence (e.g., multihop reasoning and multi-document summarization). Fig. 5.1 gives an example of the necessary information exchange in multihop QA. Each paragraph provides a partial clue to solve the task (shown as the connected entities). Intuitively, an effective global encoding should allow the entities (e.g., *Dutch-Belgian*) appearing in multiple paragraphs to share information across all of their occurrences. In this way, the embedding of *Dutch-Belgian* in the second paragraph can be aware of partial evidence from the first one and resolve the required information of *House of Anubis*.

Portions of this chapter have previously appeared as: X. Mou, C. Yang, M. Yu, B. Yao, X. Guo, S. Potdar and H. Su, "Narrative question answering with cutting-edge open-domain QA techniques: a comprehensive study," *Trans. Assoc. Comput. Linguistics (TACL)*, vol. 9, pp. 1032-1046, Sep. 2021.

Portions of this chapter have previously appeared as: X. Mou, M. Yu, B. Yao, C. Yang, X. Guo, S. Potdar, H. Su, "Frustratingly hard evidence retrieval for QA over books," In *Proc. 1st Workshop Narrative Understanding Storylines Events (NUSE)*, 2020, pp. 108-113.

In this chapter, we first benchmark book QA performance on the NarrativeQA dataset, with methods created or adapted based on the ideas of state-of-the-art ODQA methods [6], [8], [122], [163], [187], [190]. We supplement the missing study and comparison among pretrained generative models for book QA, such as GPT-2 [177] and BART [103], and we then investigate approaches to adapt to the book writing styles and employ more evidence paragraphs.

This chapter also proposes an orthogonal direction toward an efficient encoding of long sequences. Our method begins with local segments in the post-hoc aggregation approaches and relies on our proposed **synchronization mechanisms** to swap useful information from the other relevant segments during encoding, so as to maintain global information. Formally, our approach first identifies a set of anchors in the segments and divides them into groups based on the similarity of their semantic units or the roles they play in the original input sequence. The identified anchors and groups connect different segments logically and naturally. Our synchronization is applied only to the encoding stage, where inside each Transformer layer of the encoder, we perform an additional embedding update for each anchor using other anchor embeddings in the same group after a normal local encoding. The local encoding and anchor synchronization occur iteratively, so that the global information is propagated deeply among segments with anchors as bridges.

Compared to previous hierarchical encoding approaches with fixed communication designs, our approach is both more powerful and flexible. First, our approach provides a more finely grained information exchange mechanism; second, it is a general framework that reduces the problem of global encoding to synchronization schema design. For any new applications or tasks, it is simple to infuse human prior knowledge into the model by identifying task-specific anchors and anchor grouping.

Following the previous chapter, we evaluate our approach on the benchmark NarrativeQA dataset [31] for the book QA task and advance the published state-of-the-art with $\sim$7% absolute improvement in Rouge-L. To further verify the generalization of the proposed method, we also experiment with the wild multihop QA task adapted from HotpotQA [10], where the evidence annotation is assumed to be unavailable, and the input documents are treated more independently. The two settings correspond to the representative examples of the aforementioned long sequence encoding scenarios (1) and (2). The results show that our approach significantly improves performance while maintaining efficiency. Moreover, building on top of Fusion-in-Decoder (FiD, Izacard and Grave [217]), the state-of-the-art

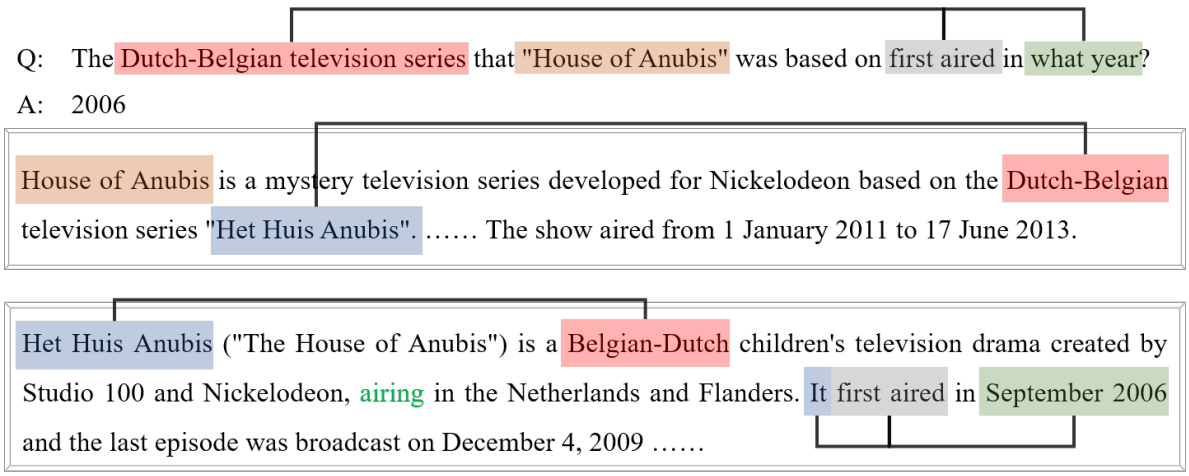

Figure 5.1: An example from HotpotQA. Different entities are color-coded. The solid lines indicate the correct partial evidence chain toward the true answer. *Dutch-Belgian* is an example of anchors that can pass the partial evidence to the others for collecting full evidence.

hierarchical method ETC [218] does not bring further improvements, but our approach improves consistently.

5.2 Related Work

To overcome difficulties in encoding long sequences with Transformer, many techniques have been proposed, and existing solutions can be categorized into two classes. The first class is **hierarchical encoding**, where the idea is either to explicitly split the input into multiple short segments for the quick encoding of each segment and then exchange information on top of their embeddings following sample-agnostic strategies [218], [219] or to implicitly constrain the information exchange among tokens with a sparse attention map [220], [221]. The essence of these methods is to identify efficient methods to pass information among segments and compensate for the loss of crucial global context across segments. One solution is introducing a pseudo-token for each segment and encouraging the pseudo-tokens to attend to one another during encoding [218] for inter-segment interactions. However, the attended positions of this family of methods are either random or following a simple strategy. Comparably, our method provides a more flexible and semantic attention mechanism. The second class is **post-hoc aggregation** in the generative framework, such as FiD with the BART model. Here, the input is split into segments that are encoded independently by the encoder, and the decoder then casts global attention over all of the segments and generates a prediction. This approach

allows for only shallow information exchange because the encoding is purely localized; yet it has proven empirically powerful in many NLP tasks. Our proposed synchronization is designed to compensate for the lack of global information exchange.

5.3 Adapted Reader for Long Context

This section describes our efforts in applying or adapting the latest ODQA ideas to improve book QA reader models. Fig. 5.1 summarizes the examined approaches. The experimental results quantify the challenges in book QA beyond ODQA.

Table 5.1: Summary of the approaches we inspected for the reader.

Approach	Original Idea in ODQA	Our Improved Version for book QA
Book Prereading	Sun et al. [191] adapt GPT on general QA datasets; Xiong et al. [192] adapt BERT on Wikipedia as entity prediction.	We propose to adapt BART to the narrative style with the text-infilling objective.
Fusion-in-Decoder	Proposed by [217] as a new type of ODQA reader.	We improve the decoder with attention on all the encoder states to capture cross-passage interaction.
Synchronization	Ainslie et al. [218], Beltagy et al. [220] reduce the memory usage with self-attention to extend models' input length limits.	We make improvements over the Fusion-in-Decoder by exchanging the information across the split segments.

5.3.1 Method 1: Book Prereading

Inspired by the literature on the unsupervised adaptation of pretrained LMs [191], [192], we tasked the reader to **preread** the training books in an additional pretraining step, prior to fine-tuning the QA task. This technique helped the reader better adapt to the narrative writing styles.

In particular, we extracted random passages from all of the training books to build a passage pool. For each training iteration, we masked random spans from each passage, following the setting in [103]. The starting positions of spans were sampled from a uniform distribution, without overlap. The length of each span was taken from a Poisson distribution, with $\lambda = 3$. Each span was then replaced by a single (`[mask]`) token, regardless of span length. We masked 15% of the total tokens in each passage. During the prereading stage, we used the masked passage as the encoder input and the raw passage as the decoder output to restore the raw passage in an autoregressive manner.

5.3.2 Method 2: Fusion-in-Decoder

Recently, Izacard and Grave [217] scaled a BART reader up to a large number of input paragraphs. The method, FiD, first concatenates each paragraph to the question to obtain a question-aware encoded vector; then, it merges these vectors from all paragraphs and feeds them to a decoder for answer prediction. FiD reduces memory and time costs for encoding the concatenation of all paragraphs and improves on multiple ODQA datasets. FiD is an interesting alternative for book QA, since it can be viewed as an integration of the ranker and reader, with the ranker absorbed in the separate paragraph encoding step.

FiD trades cross-paragraph interactions for the increased encoding of paragraphs. The single encoded vector per passage works well for extractive ODQA because the vector only needs to encode information regarding candidate answers. However, in book QA, the answers may not be inferred from a single paragraph, and the integration of multiple paragraphs is necessary. As such, in our approach, we concatenated the encoded vectors of all paragraphs and relied on the decoder's focus on these vectors to capture cross-paragraph interactions.

5.3.3 Method 3: Transformer with Synchronization Framework

In this section, we propose our TranSync framework which extends the Transformer layer with an embedding synchronization module attached to the end. Given a long context sequence C, we divide it into segments similar to what we do in FiD (i.e. $C = [s_1; s_2; ...; s_n]$ where s_i is the i-th segment of C and n is the number of segments). A segment can represent a natural sentence or a sequence of a certain length. Together with the question q, we re-organize the input to the Transformer and form a set of question-prefixed segments $\{s_i^q\}_{i=1}^n$, s.t.

$$s_i^q = [q; < \text{SEP} >; s_i] \tag{5.1}$$

where <SEP> is a special token. An embedding layer converts the text segments $\{s_i^q\}_{i=1}^n$ into their corresponding question-aware embeddings $\{\mathbf{e}_i^q\}_{i=1}^n$, s.t.

$$\mathbf{e}_i^q = [\mathbf{t}_i^1; \mathbf{t}_i^2; ...; \mathbf{t}_i^{l_i}] \in \mathbb{R}^{l_i \times d} \tag{5.2}$$

where l_i is the length of s_i^q's token sequence, d is the dimension of the feature vector and $\mathbf{t}_i^j \in \mathbb{R}^d$ is the embedding for the j-th token in the i-th segment.

For genericity, our synchronization is performed between the target anchor and the incoming anchors, following the idea of message passing. The values of the target anchor embedding $\mathbf{a}_t$ are updated with the weighted sum of the incoming anchor embeddings and itself,

$$\mathbf{a}'_t = \sum_k \alpha_k \mathbf{a}^k, \quad s.t. \sum_k \alpha_k = 1 \tag{5.3}$$

where $\{\mathbf{a}^k\}$ are the embedding spans[16] of the same length within the same anchor group, and α_k is the normalized weight. In this work, for each anchor group, we form a new sequence from the selected anchor embeddings, i.e., $[\mathbf{a}^1; \mathbf{a}^2; ...; \mathbf{a}^k]$, and use a self-attention module to compute the weights and update the embedding values.

Our TranSync framework is embedded in Transformer layer. The synchronization is performed between the local self-attention and normalization steps to achieve deep information exchange. At the end of the last Transformer layer, the synchronized segment embeddings are fused into one by concatenating one another, as follows:

$$[\mathbf{e}^q_1; \mathbf{e}^q_2; ...; \mathbf{e}^q_n] \in \mathbb{R}^{\sum_1^n l_i \times d} \tag{5.4}$$

The flexibility of our TranSync framework is granted by the manifold strategies of identifying anchors in the segments and the heterogeneous message passing directions. The schema is further detailed in Section 5.4.3.

5.4 Task Setup

5.4.1 Datasets

Following [31], we define the **book QA** task as finding the answer **A** to a question **Q** from a book, where each book contains a number of logically consecutive passages, $\mathcal{C}$. The number of paragraphs $|\mathcal{C}|$ from different books varies from a few hundred to thousands.

Parts of our experiments are conducted on the NarrativeQA dataset [31], which has a collection of 783 books and 789 movie scripts (we use books to refer both of them), each containing an average of 62K words. In addition, each book has 30 question–answer pairs generated by human annotators in free-form natural language. Hence, the exact answers are

[16]Some words may correspond to multiple tokens due to the byte pair encoding (BPE) algorithm.

not guaranteed to appear in the books. NarrativeQA provides two settings: the **summary** setting and the **full-story** setting. The former requires answering questions from book summaries from Wikipedia, while the latter requires answering questions from the original books, assuming that the summaries do not exist. We use both settings to verify the feasibility of our readers. In the rest of the paper, if it is not explicitly mentioned, the book QA task corresponds to the full-story setting.

Following [31], we tokenize the books with SpaCy[17] and split each book into a sequence of non-overlapping trunks of 200 tokens. We also construct a wild multihop reasoning task from the HotpotQA dataset [10], which provides two evidence documents and eight distractor documents for each question. We adopt the realistic assumption with no evidence annotation provided to investigate the models' ability to sort out the reasoning chains from multiple documents. We designed two settings on the HotpotQA dataset, intending to verify the effect of various context lengths on different models. **MultiHop-10** uses all eight distractors in the dataset; the concatenation of the documents thus exceeds the length limit of BART. **MultiHop-6** uses only four, which is within BART's limit, on average.

With the wild multihop reasoning, we hope to justify that our TranSync can effectively pass important messages across segments. For consistency, we also formulate the wild multihop reasoning as a generative QA task with the goal of predicting a free-form or YES/NO answer, given the question and context.

5.4.2 Baseline Readers

Following the formulation of the open-domain setting, we employed the dominating ranker-reader pipeline that first utilizes a ranker model to select the most relevant passages $\mathcal{C}_{\mathbf{Q}}$ to $\mathbf{Q}$ as evidence,

$$\mathcal{C}_{\mathbf{Q}} = \text{top-k}(\{P(\mathbf{C}_i|\mathbf{Q})|\forall\ \mathbf{C}_i \in \mathcal{C}\}); \tag{5.5}$$

and then a *reader model* predicts answer $\tilde{\mathbf{A}}$ given $\mathbf{Q}$ and $\mathcal{C}_{\mathbf{Q}}$. In this chapter, we focus only on the reader module.

We studied the usage of different pretrained LMs on book QA, including BART [103], GPT-2 [177], T5 [104], and BERT [176]. The first three are *generative* readers and can be directly trained with free-form answers as supervision. In particular, during training, we

[17] `https://spacy.io/`

Table 5.2: Characteristics of the compared systems, following Mou et al. [168]. †/‡ refers to generative/extractive QA systems. In addition to the standard techniques, Wang et al. [6] used reinforcement learning to train the ranker, and Tay et al. [185] used curriculum to train the reader to overcome the divergence of retrieval qualities between training and testing.

System	trained ranker	pretrained LM	extra data
IR+AttSum† [31]			
IR+BiDAF‡ [31]			
IAL-CPG† [185]			
R^{3}‡ [6]	✓		
BERT-heur‡ [186]	✓	✓	✓
DS Ranker+GPT2† [168]	✓	✓	
DS Ranker+BERT‡ [168]	✓	✓	
Our best QA system†	✓	✓	

treated $\mathbf{Q} \oplus [\text{SEP}] \oplus \mathcal{C}_{\mathbf{Q}}$ as input to generate answer $\mathbf{A}$, where [SEP] represented the special separation token and $\oplus$ the concatenation operator.

For the extractive reader BERT, we predicted the most likely span in $\mathcal{C}_{\mathbf{Q}}$, given the concatenation of the question and the evidence $\mathbf{Q} \oplus [\text{SEP}] \oplus \mathcal{C}_{\mathbf{Q}}$. Due to the generative nature of book QA, the true answer may not have an exact match in the context. As such, we followed Mou et al. [168] to find the span $\mathbf{S}$ that has the maximum Rouge-L score with the ground truth $\mathbf{A}$ as the weak label, subject to $\mathbf{A}$ and $\mathbf{S}$ having the same length (i.e., $|\mathbf{S}| = |\mathbf{A}|$).

In addition to our own baselines, we also compared competitive public book QA systems as baselines following [31], [168], [185], [186] under the narrative full-story setting. Frermann [186] used exterior supervision from the book summaries, which is considered unavailable in the book QA task. Because the summaries are written by humans, the system can be assumed to benefit from the human comprehension of books. Fig. 5.2 outlines the details of the compared systems.

5.4.3 Synchronization Setup

We customize the preprocessing steps for the synchronization and adapt them to two different tasks.

Book QA Synchronization Schema We split each summary into natural sentences as the segments $\{s_i\}$ and prefix them with the question q, following Section 5.3.3. This separation of the continuous summary sentences drops the global context across segments during encoding. As compensation, we apply a segment-level synchronization, which adopts the preceding question sequence as the anchor. We simply use the special token ¡SEP¿, which connects to the question in each segment as the representative; this significantly reduces synchronization cost. The segment-level synchronization occurs only among the closest neighboring segments, inspired by their natural order in the summary text. Intuitively, it provides each segment compressed contextual information from its neighbors and makes the question embedding aware of its matched contents across multiple segments. As such, we expect it can better address questions that require multiple sentences to answer.

Multihop QA Synchronization Schema We split each concatenated document into segments of similar lengths containing various numbers of natural sentences. We apply two methods of synchronization according to the task's unique properties. A similar segment-level synchronization schema is first applied. However, due to the differing segment splitting strategies, no continuation is guaranteed between neighboring segments. As such, we synchronize across all segments rather than only among neighbors. Secondly, we take the titles of the original documents as word-level anchors. For simplicity, the titles are added to the input[18], immediately following the question sequence. Similarly, we perform synchronization among all of the title-associated special tokens to reduce computational costs. Due to the multihop nature of the samples, we expect the token-level synchronization to help build latent connections among the evidence.

5.4.4 Metrics

Following previous works [31], [185], [186], we used the study of Rouge-L [212] as the main metric for the reasoning results on the NarrativeQA dataset.[19] We compare the overlap between the true answers and the retrieved passages or the predicted answers by calculating their rolling Rouge-L. The rolling Rouge-L is calculated as follows: we went through the context, iteratively computed the Rouge-L score between the true answer and each candidate

[18]The title words already appear in the document, hence adding them to the input does not introduce new information and is regarded as a fair comparison.

[19]For fair comparison, we lower-cased the answers and removed the punctuation, and used the open-source nlg-eval library [222].

span of the same length as the true answer having overlapping considered, and finally took the maximum value as our rolling Rouge-L score. Appendix A.1 provides the results, along with other metrics used in previous works, including Bleu-1/4 [223] and Meteor [224] as well as the exact match (EM) and F1 scores commonly used in extractive QA. We use EM and F1 scores on the wild HotpotQA dataset.

5.4.5 Model Selection

We select the best models in the development set according to their average scores of Rouge-L and EM. For ranker model selection, we use the average score of upper-bound EM and Rouge-L of the top-five ranked paragraphs.

5.5 Evaluations

In implementation, we initialized readers using `bert-base-uncased`, `gpt2-medium`, `T5-base`, and `bart-large`. These readers use the top-three retrieved passages as input, except for the FiD reader, which uses the top-10, giving the readers comparable time and space complexities [20].

5.5.1 Reader Model Validation

We first compare our readers under the summary setting in Table 5.3 to verify the correctness of our readers. Our BERT reader achieves performance close to the public state-of-the-art in this setting.

Our GPT-2 reader outperforms the existing systems without the use of pointer generators (PG) but lags behind the state-of-the-art with PG. Despite the large gap between systems with and without PG in this setting, Tay et al. [185] demonstrates that it did not contribute much in the full-story setting in the ablation study. Nonetheless, we will investigate the usage of PG in pretrained LMs in future work. Table 5.4 provides some examples of the generated results by the GPT-2 reader.

5.5.2 On Closed-Domain Setting

Table 5.5 shows the overall results on all three tasks. Our proposed TranSync achieves the best results on both NarrativeQA and our new wild multihop QA tasks.

[20] We used the top-10 so that different readers show similar time and space complexities.

Table 5.3: Results under NarrativeQA summary setting on dev/test set (%). PG refers to the usage of pointer generator. For extractive models, we compare with the best public result [187] and its BERT-only ablation. The latter corresponds to the same setting as ours. For generative model, we compare with the best public models with and without pointer generators.

System	Bleu-1	Bleu-4	Meteor	Rouge-L
Extractive Readers				
BERT-only [187]	-	-	-	55.8/56.1
BERT + Hard EM [187]	-	-	-	**58.1/58.8**
BERT w/ full training signals [**Ours**]	49.35/49.02	25.76/25.85	23.93/24.14	52.62/52.02
BERT w/ exact answer match only [**Ours**]	**49.78/49.64**	**27.01/28.94**	**25.22/25.12**	57.19/56.35
Generative Readers				
Attention Sum [31] (w/o PG)	23.54/23.20	5.90/6.39	8.02/7.77	23.28/22.26
Masque [118] (w/ PG)	-/48.70	-/20.98	-/21.95	-/54.74
GPT-2 Reader(w/o PG) [**Ours**]	33.63/35.49	11.87/14.33	13.71/14.36	34.32/35.65

Table 5.4: Generative result examples. The model tends to generate shorter answers in general. The longer answer it generates, the less likely the answer tends to be correct. The grammatical correctness and fluency of the long generative answers are approaching to human level, regardless of the problematic logic between the generated answer and question. The majority of the generative results do not make sense logically which leads to the low scores in different metrics.

Question	Gold Answer 1	Gold Answer 2	Generative Result
Where is Millicent sent to boarding school?	Millicent is sent to a boarding school in France	France	France
What is Morgan's relationship to Wyatt?	Morgan is Wyatt's brother	Brothers	Brother
What illness does Doc Holiday suffer from?	Tuberculosis	Tuberculosis	Lung cancer
How does Carl make his house fly?	He attaches thousands of helium balloons to it	Balloons	He uses a parachute to climb up the side of the dirigible
How does Felipe die?	Suicide	He suffers a physical breakdown	He is killed by a bullet in the head
What was the great stone face and how did it appear?	A natural rock formation on the side of a mountain	A natural rock formation which appeared when viewed at a proper distance	It was a stone face

To our surprise, splitting the long sentences into question-aware segments alone (FiD) elicits strong results against the BART baseline. This indicates that the post-hoc aggregation

Table 5.5: Overall results under the NarrativeQA summary setting on test set (%) and the two wild multihop setting on dev set (%).

System	NarQA	MultiHop-10		MultiHop-6	
	Rouge-L	EM	F1	EM	F1
BART	64.78	41.63	54.85	55.62	69.96
FiD	66.57	55.65	69.35	57.42	71.30
FiD+ETC	65.89	55.46	69.31	57.52	71.66
TranSync	**67.58**	**56.49**	**70.32**	**58.30**	**72.61**

of local embeddings can handle a significant portion of testing cases, which reflects the absence of global reasoning in many existing datasets. Our synchronization mechanism compensates for the loss of global context resulting from the sequence splitting and brings a consistent 1% improvement over FiD across all three tasks. ETC does not provide a further improvement over FiD as our approach does. This demonstrates empirically that ETC's synchronization mechanism does not provide complementary global information to the post-hoc aggregation approach.

Finally, in addition to Table 5.5, we also experiment with different segment lengths and find that the split context length should be at least two times longer than the prefixing question for effective encoding; otherwise, the question would dominate in the segment, which would lead to a significant drop in performance. Together with the observations that FiD with short segments outperforms BART with long sequences in both settings, we conclude that splitting length is a hyper-parameter worth fine-tuning.

5.5.3 On Open-Domain Setting

We then experimented our whole QA pipelines in the full-story setting. Tables 5.6 and 5.7 compare our results with public state-of-the-art generative and extractive QA systems. Overall, we significantly raise the bar on NarrativeQA by 4.7% over our best baseline (and by 11.5% over the best published one), which confirms the advantages of pretrained LMs, as observed in most QA tasks. The use of pretrained LMs also helps outperform the method in [186] on multiple metrics without the usage of the strong extra supervision from the summaries. Among all of the generative reader models, BART outperforms GPT-2 and T5 by notable margins. Our best reader (see Section 5.3 for details) brings an additional >3% Rouge-L improvement on development and test sets compared to the BART reader.

Table 5.6: Generative performance in NarrativeQA full-story setting (book QA setting) dev/test set(%). *Oracle IR* utilizes question and true answers for retrieval.

System	Bleu-1	Meteor	Rouge-L	EM	F1
Public Generative Baselines					
AttSum (top-10) [31]	20.00/19.09	4.45/4.29	14.47/14.03	-	-
AttSum (top-20) [31]	19.79/19.06	4.60/4.37	14.86/14.02	-	-
IAL-CPG [185]	23.31/22.92	5.68/5.59	17.33/17.67	-	-
- curriculum	20.75/-	4.65/-	15.42/-	-	-
Our Generative QA Models					
BM25 + BART Reader	24.52/–	8.68/–	23.16/–	6.28/–	21.16/–
BM25 + GPT-2 Reader	24.54/–	7.32/–	20.25/–	5.12/–	17.72/–
BM25 + T5 Reader	19.28/–	6.62/–	16.89/–	4.17/–	15.47/–
ORQA Ranker + BART Reader	27.06/27.68	9.35/9.74	25.83/26.95	8.57/8.95	23.80/25.08
+ Preread	28.54/–	9.59/–	26.82/–	10.21/–	25.06/–
+ FiD	28.04/–	9.49/–	26.27/–	9.20/–	24.29/–
+ FiD + TranSync	29.05/–	10.10/–	29.07/–	11.25/–	27.59/–
+ FiD + Preread	**29.56**/–	10.03/–	27.91/–	10.45/–	26.09/–
+ TranSync	29.09/**30.21**	**10.34/10.92**	**29.75/31.51**	**12.30/13.48**	**28.33/30.24**
Oracle IR + BART Reader	*35.04/36.41*	*14.78/15.07*	*37.75/39.32*	*15.78/17.27*	*37.71/38.73*

Table 5.7: Extractive performance in NarrativeQA full-story setting (book QA setting) dev/test set(%). † indicates that the method uses additional supervision. *Oracle IR* utilizes question and true answers for retrieval.

System	Bleu-1	Meteor	Rouge-L	EM	F1
Public Extractive Baselines					
BiDAF [31]	5.82/5.68	3.84/3.72	6.33/6.22	-	-
R^3 [6]	**16.40/15.70**	3.52/3.47	11.40/11.90	-	-
BERT-heur† [186]	*-/12.26*	-/5.28	*-/15.15*	-	-
Our Extractive QA Models					
BM25 + BERT Reader	13.27/-	4.29/-	12.59/-	4.67/-	11.57/-
ORQA Ranker + BERT Reader	15.06/14.25	**5.28/5.06**	**15.42/15.22**	**6.25/6.19**	**14.58/14.30**

Table 5.6 shows how the various reader techniques in Section 5.3 contribute to QA performance. First, switching the BART reader to FiD gives large improvements (2.8%) when using our best ranker from the previous chapter, which approaches the result of "our ranker + BART". This agrees with our hypothesis in Section 5.3 Method 2: that FiD fulfils the roles of both ranker and reader. Second, although the above result shows that FiD's ranking

ability does not add much to our best ranker, our cross-paragraph attention enhancement still improves FiD due to better retrieval results (0.5% improvement over "our ranker + BART"). Third, our synchronization mechanism provides an additional 1.2% increase in the Rouge-L score on the basis of FiD, which matches our observations outlined in Table 5.5. The gains stem from our compensation on the loss of global context caused by FiD. Finally, the book prereading brings consistent improvement, and the combination of our orthogonal reader improvements provides the best results. We also confirm that prereading mostly helps decoders, as only training the decoder gives comparable results.

We conducted a significance test on the results to find the best system. No agreement exists on best practices to validate the improvements [225]–[227] in the field of natural language processing. We chose the nonparametric bootstrap test because it is a more general approach and does not assume specific distributions over the samples. For bootstrapping, we sampled 10K subsets, the size of each being 1K. The small p-value (< 0.01) shows the effectiveness of our best model.

5.5.4 Synchronization Efficiency

Table 5.8 provides an analysis of the efficiency of our TranSync framework. The complexity comparison shows that the TranSync is more memory efficient than the BART baseline in theory and becomes more superior when $l_q \ll \frac{l_c}{n}$. We also compare the runtime speed empirically by measuring the average time used for encoding per token. Although the synchronization introduces extra complexities to the encoding procedure, our experiments on the NarrativeQA dataset verify that the overall speed of our methods remains double that of the BART baseline.

Table 5.8: Efficiency comparison. l_q and l_c are the lengths of the question sequence and context sequence; n is the number of split segments. The encoding time per token is averaged over 100 QA samples.

System	$\mathcal{O}(f)$	Time/Token
BART Baseline	$(l_q + l_c)^2$	292 μs
TranSync	$(l_q + \frac{l_c}{n})^2 \cdot n$	134 μs

5.6 Chapter Summary

In this chapter, we first experimented with the latest large-scale pretrained LM readers on the book QA task and supplemented the study of their applications on the book QA task. The experimental results demonstrated the advantages of using pretrained LMs on the task but also showed the considerable gap from the human performance at the same time. Inspired by the success of the FiD method in encoding long sequences and our observations of the loss of global context in FiD, we further proposed TranSync framework with flexible synchronization mechanisms to encourage the information exchange among local embeddings and to compensate for the loss of global context. We presented the feasibility of our methods in reasoning tasks with long context and showed its high adaptability to different scenarios. Meanwhile, our experiments and analysis proved that the combination of FiD and our synchronization could speed up the computation for raw long input sequences. We consider our work valuable as a simple solution to address the long context issue in QA and potentially applicable to other long sequence modeling tasks.