INVESTIGATION OF TRANSFORMER MODIFICATIONS FOR VARIOUS LONG DOCUMENTS PROCESSING TASKS

Fourth year Ph.D Student
Arij Al Adel

Moscow Institute of Physics and Technology

Directivity: 09.06.01 Informatics and Computer Engineering

Specialization: 2.3.5 Mathematical and software of computers, complexes and computer networks

Supervisor:
PhD of physical and mathematical sciences
Valentin Malykh

8 February, 2024

Outline

1. Motivation and background
2. Scientific actuality of the research
3. Goal
4. Tasks
5. Scientific novelty
6. Theoretical and practical value of the work
7. Model modifications details with applications
   - translation
   - MLM task
   - question answering task
   - summarization
8. Publications submitted for defense
9. Conclusion

Motivation and background

How to scale up the attention to process long inputs?

- Divide long input into segments (blocks, chunks)
  - Hierarchical (HIBERT)
  - Recurrence (Transformer-XL)
- Process the long input without dividing it.
  - Sparse attention models(Longformer, ETC, BigBird, GMAT, LongT5, terratransformer...etc)

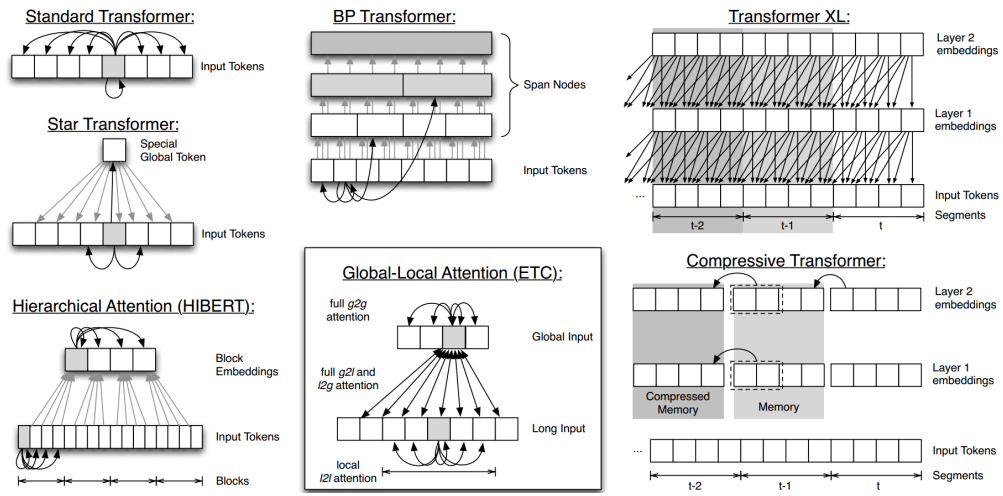
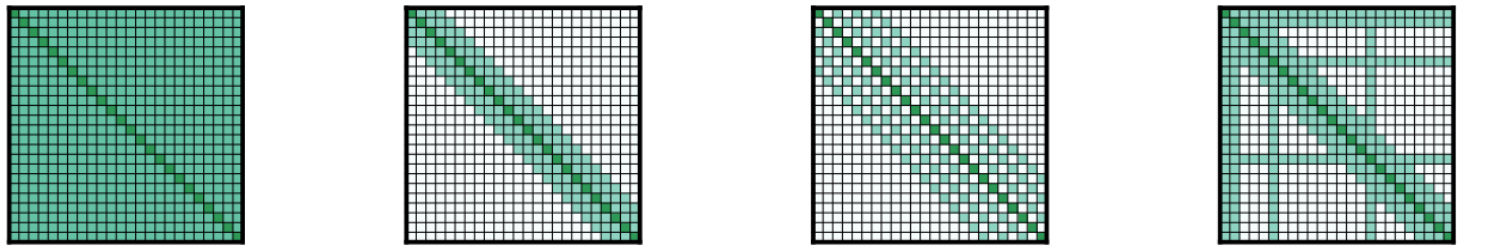


Figure 1: Summary of attention approaches to handle long documents [1]

Motivation and backgrond

Sparse attention



(a) Full n^2 attention (b) Sliding window attention (c) Dilated sliding window (d) Global+sliding window

Figure 2: Comparing the full self-attention pattern and configuration of attention patterns in longformer [2]

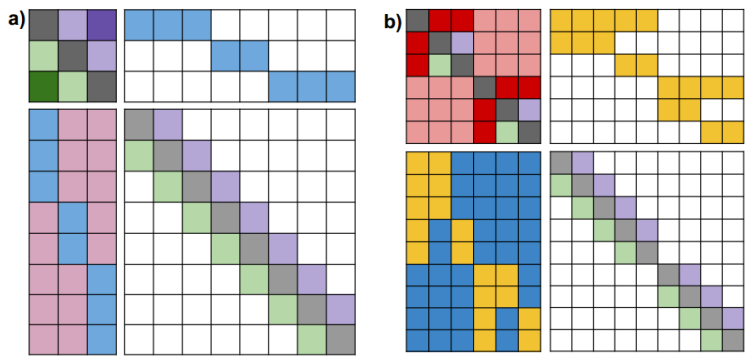


Figure 3: Example attention patterns for handling (a) long inputs and (b) structured inputs. White background means attention is masked via M, and the other colors indicate different relative position labels [1].

Scientific actuality

Cope with long texts using transformer.

Goal the study

The main goal of this dissertation is to suggest and study different structural modifications of the encoder-decoder model to process longer input.

Tasks

1. Review publications on the application of attention mechanism in the transformer on long text inputs;
2. Propose and implement new transformer structures using different masking patterns, position bias algorithms, additional memory tokens to inputs, and new attention schema so the model can process long inputs;
3. Investigate the efficacy of the resulting modifications on the problems of translation, mask language modeling, question answering with long-range context, and summarization of long inputs;
4. Develop and publish in the public domain the implementation of the proposed structural modifications and results with analysis.

Scientific novelty

1. Propose structural modifications to the T5 encoder-decoder architecture to process longer inputs than the standard input length.
2. An investigation was conducted to evaluate the effectiveness of modifications made to the T5 model in the context of translation, mask language modeling, question answering with long-range context, and summarization of long inputs.

Theoretical and practical value of the work

- Exploration of the proposed variants of transformer modifications for processing long input;
- These new modifications led to breaking attention computational barriers for processing long-range inputs;
- This thesis presents a new usage for the internal global tokens called memory tokens, proper masking technique, and proper usage for the original relative positional encoding to relate chunked input with related memory slots;
- The modifications resulted in consuming long-range input, information interchange, and data compression;
- Models trained as part of the dissertation work are made publicly available.

First proposed model design details

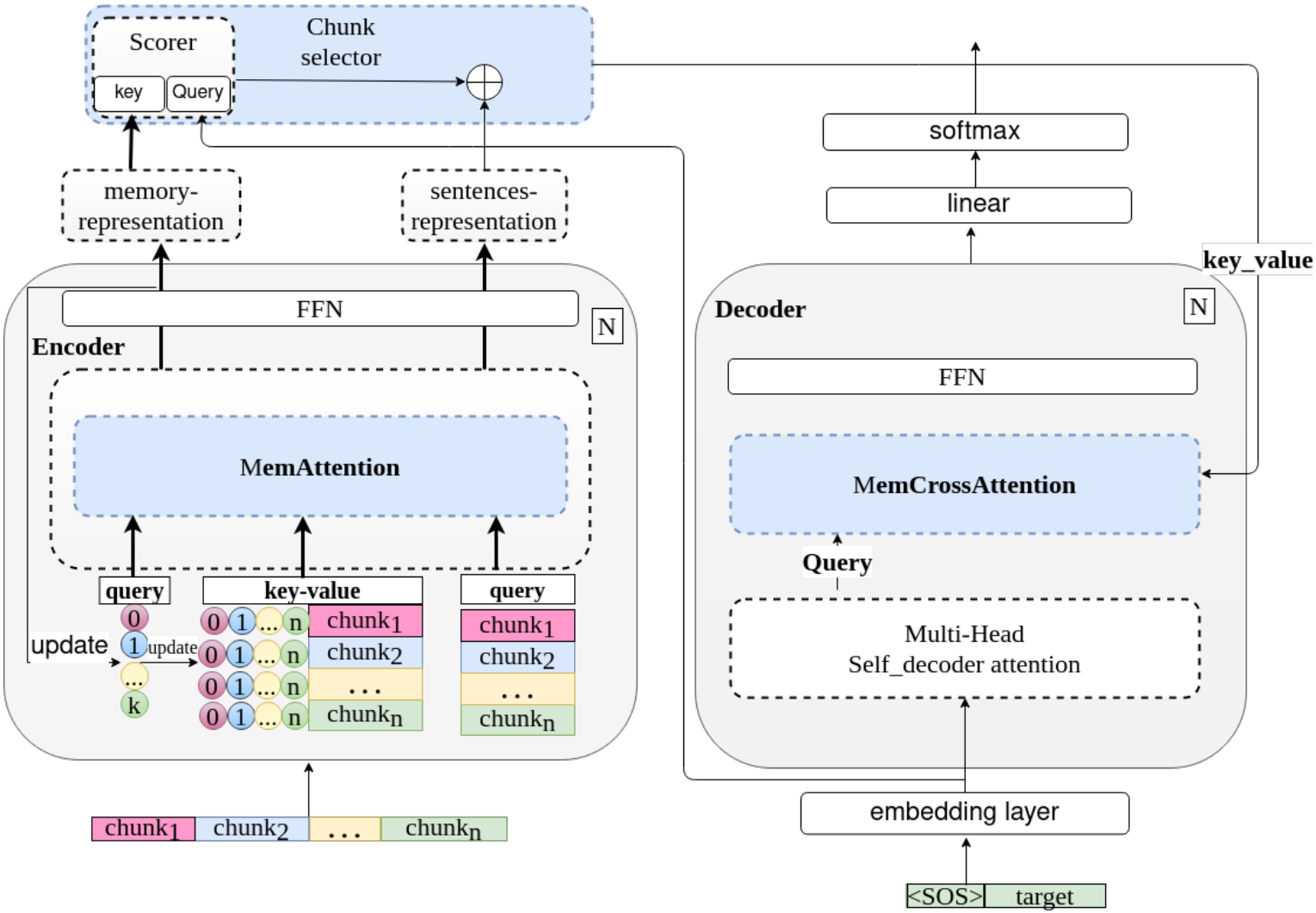


Figure 4: The main implemented model

Source by: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9681776&tag=1

First proposed model design details: input data flow

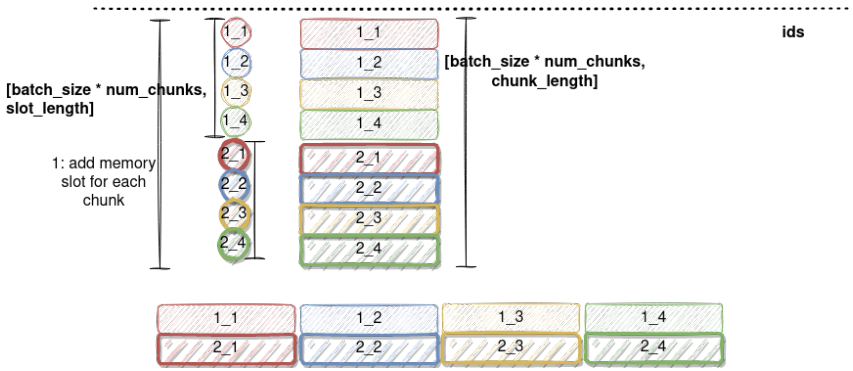


Figure 5: One memory slot for each chunk.

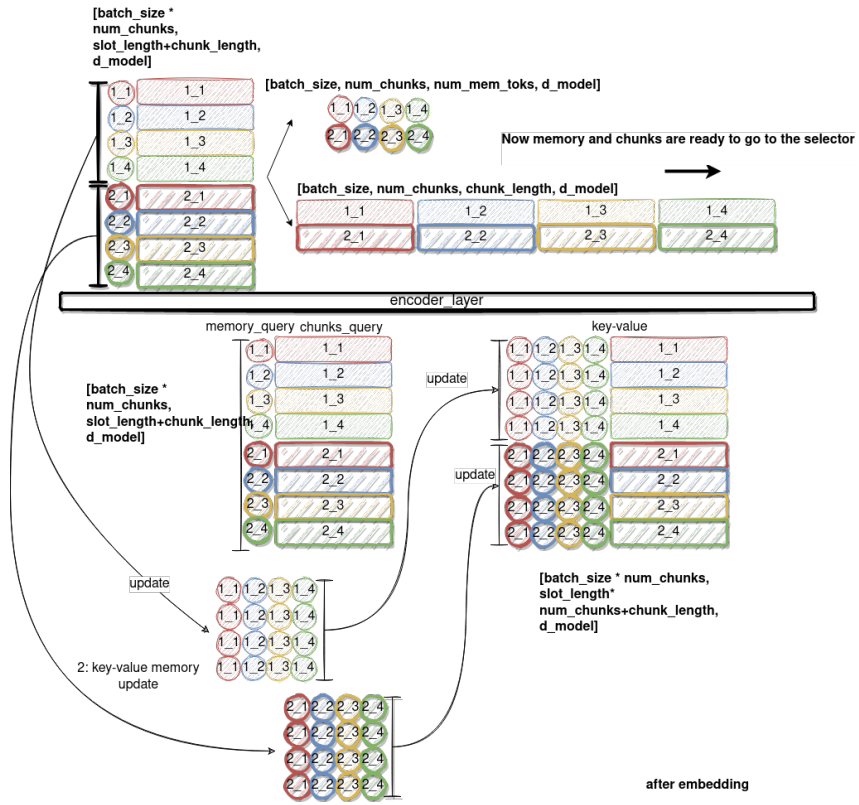


Figure 6: memory slot and chunks processing.

Models details: explanation of memory slot-chunk attention inside the encoder

Theoretical and practical value of the work!

The modifications resulted in consuming long-range input, information interchange, and data compression;

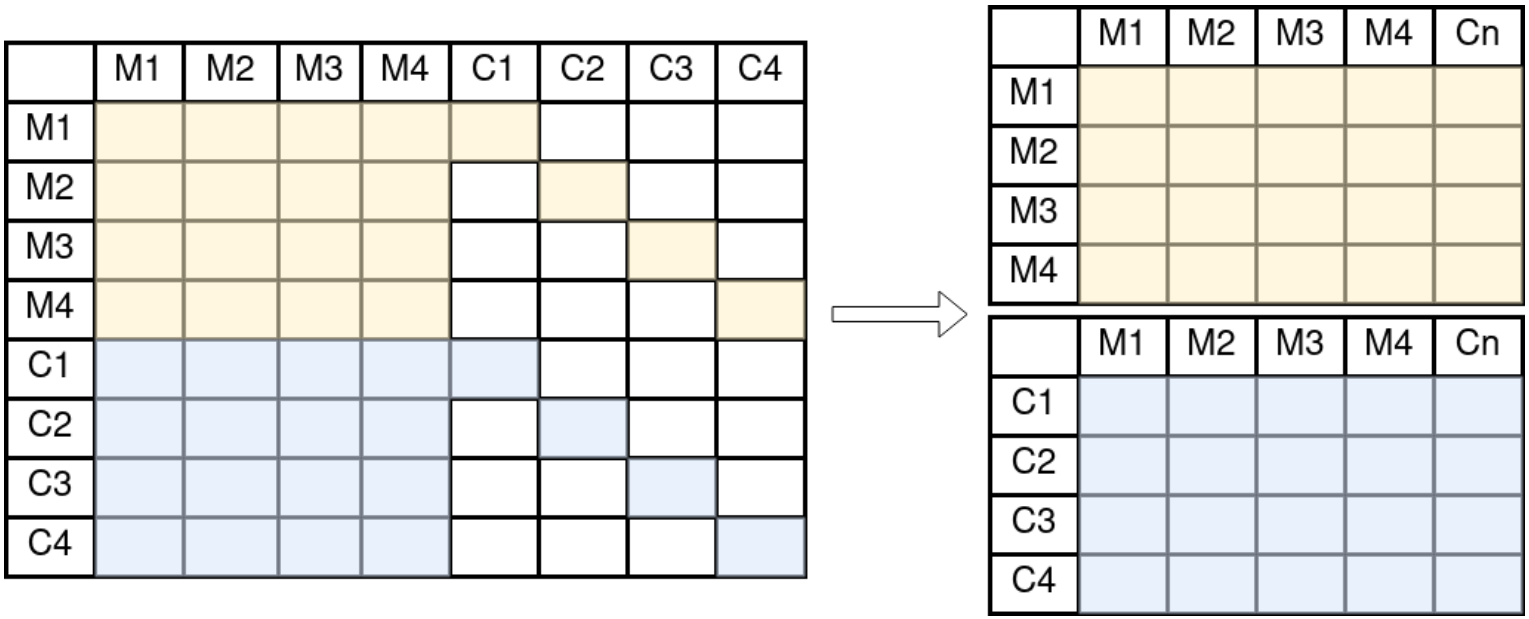
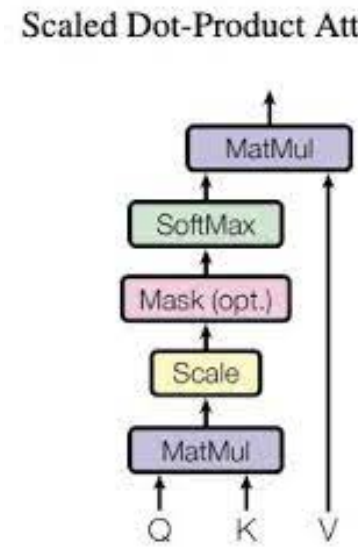
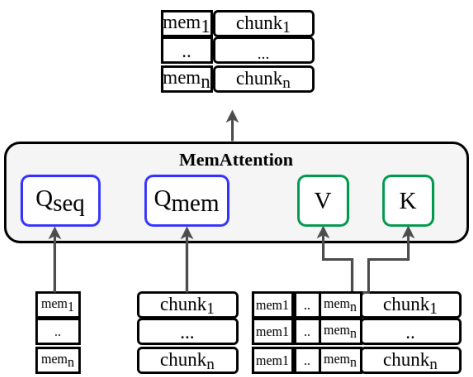


Figure 7: Example attention pattern for handling input chunks in which dense attention is used for each chunk.

Models details: MemAttention



(a) Scaled dot product attention in transformer



(b) MemAttention overview

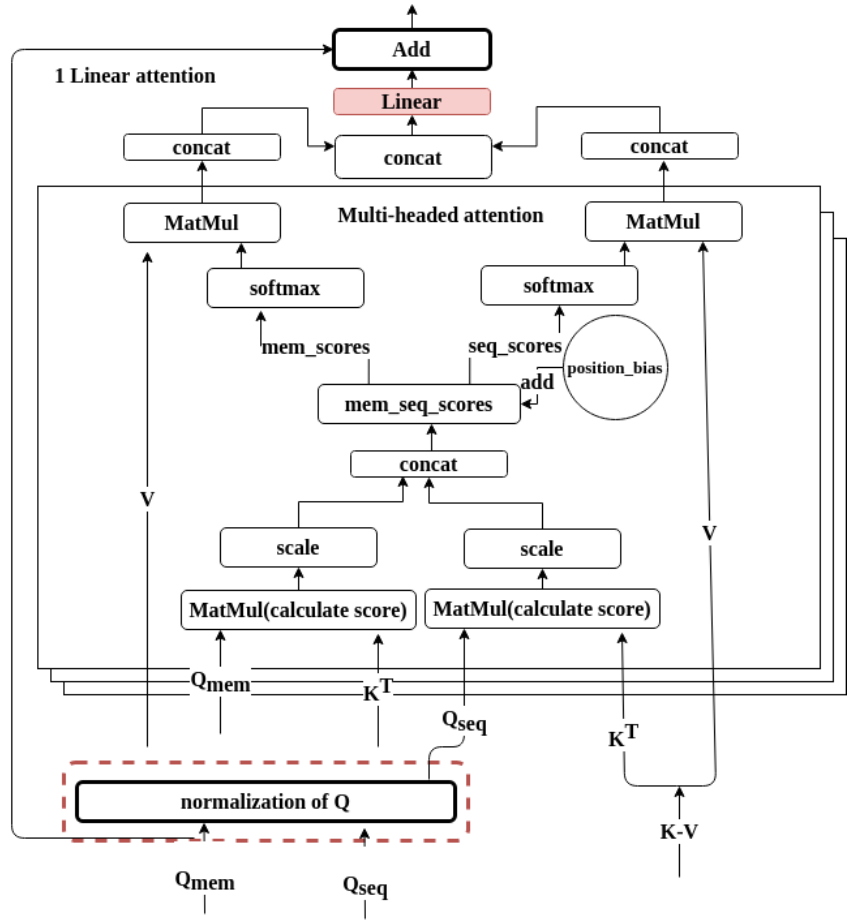


Figure 9: MemAttention structure ith one linear layer for both memory slot and its segment

Model details: MemAttention anf FFN

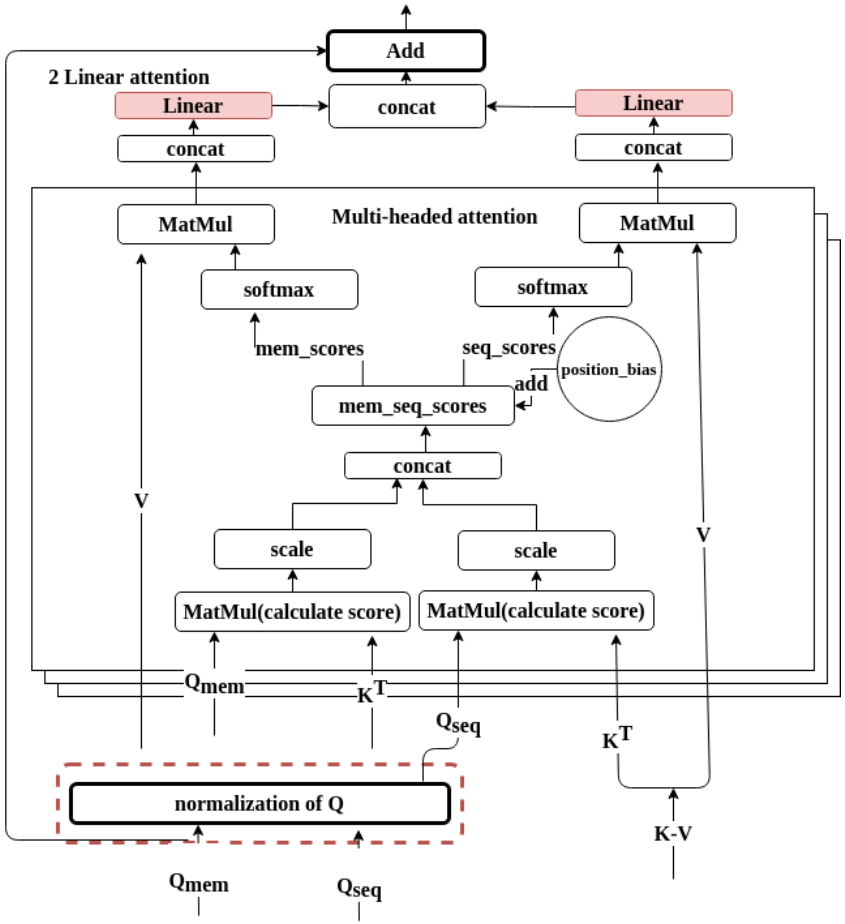


Figure 10: The detailed design of the MemAttention using 2 linear layers for memory slots and chunk output in the attention

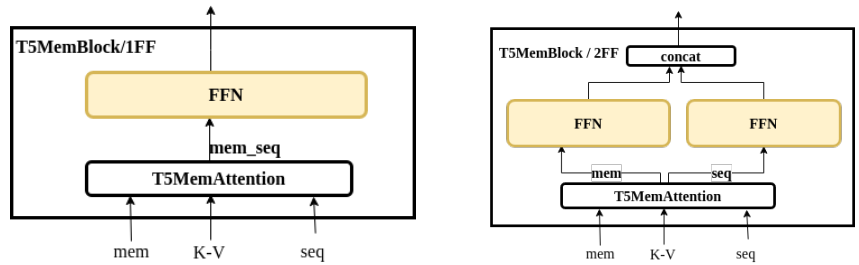


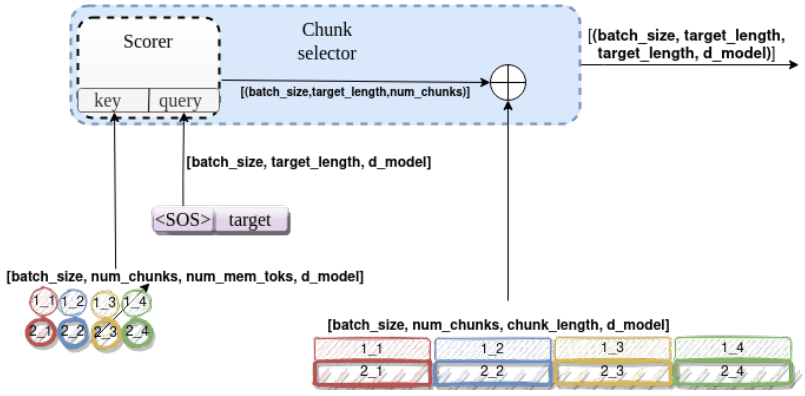
Figure 11: Figure (a) for using one feedforward for the concatenation of memory and chunk representations and figure (b) using separate feedforward for memory and related chunk.

Model details: chunk selector and cross attention

- Selector:

$$S = softmax(Query * Key^T) \quad (1)$$

$$Z = \sum_{1<=i<=k} Sent_i * S_i \quad (2)$$



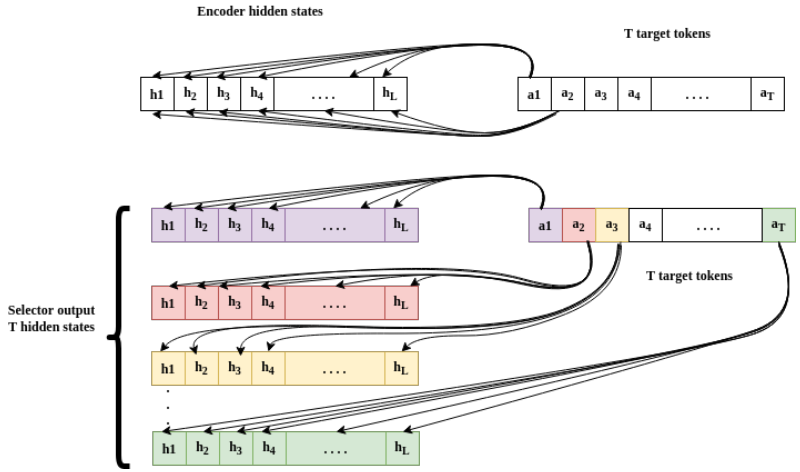


Figure 12: The selector

Model details: chunk selector and cross attention

- Selector:

$$S = softmax(Query * Key^T) \quad (1)$$

$$Z = \sum_{1<=i<=k} Sent_i * S_i \quad (2)$$

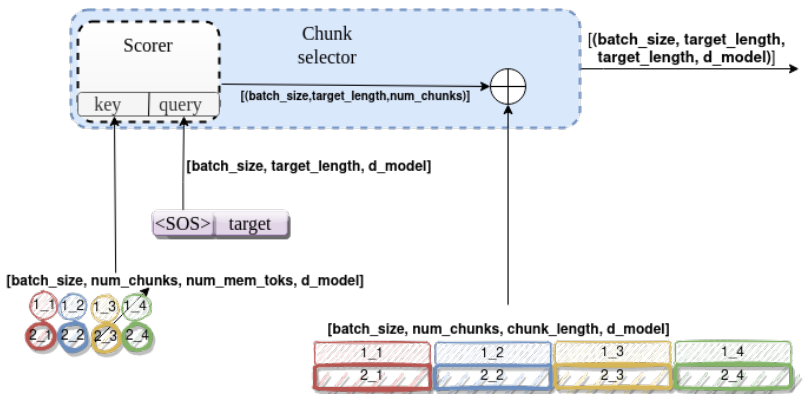
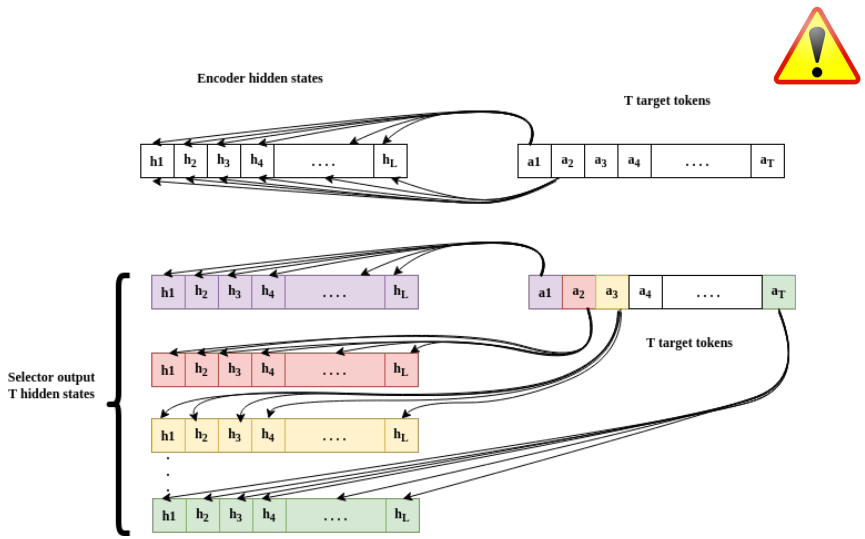


Figure 12: The selector



Translation data sets

Agnostic aware data set sample

] 1. Do you know anyone who's got any strawberry milk? (English)
1. У кого-нибудь есть клубничное молоко ? (Russian)

Context aware dataset samples

(English)

1. But I have to watch her shows . _eos You going somewhere ? _eos I don 't believe that . _eos but it 's true .
2. You going somewhere ? _eos I don 't believe that . _eos but it 's true . _eos They never stopped .
3. I don 't believe that . _eos but it 's true . _eos They never stopped . _eos All those years , all those protests , they kept at it .
4. but it 's true . _eos They never stopped . _eos All those years , all those protests , they kept at it . _eos just like a couple of rabbits .

(Russian)

1. Но мне приходиця смотреть ее передачи . _eoc Ты куда-то уезжаешь ? _eoc Не могу поверить в это . _eoc Но это правда .
2. Ты куда-то уезжаешь ? _eoc Не могу поверить в это . _eoc Но это правда . _eoc Они никогда не прекращали .
3. Не могу поверить в это . _eoc Но это правда . _eoc Они никогда не прекращали . _eoc Все эти годы , все эти преграды , они держались за это .
4. Но это правда . _eoc Они никогда не прекращали . _eoc Все эти годы , все эти преграды , они держались за это . _eoc Как парочка кроликов .

Application: translation experiments setup

- SentencePiece tokenizer was trained on training data files for both languages Russian and English.
- Optimizer: adam
- Scheduler: linear
- Learning rate 5-5e
- batch size 160
- early stopping
- Num_encoder layers: 2
- Num_decoder layers:2
- Num_experiment_runs: 2 ; except where otherwise noted.

Application: translation

Design variants for translation task

- v1: one linear layer(attention), two separated feedforward
- v2: one linear layer(attention), one feedforward
- v3: two linear layers(attention), two separated feedforward
- v4: two linear layers (attention), one feedforward

Agnostic aware translation

variant	val loss	test loss	val BLEU	test BLEU
$T5(baseline)$	1.596	2.349	26.12	25.648
$v1$	1.632	1.634	26.34	**26.725**
$v2$	1.642	1.639	26.09	26.437
$v3$	1.629	1.621	**26.5**	26.706
$v4$	1.636	1.639	26.34	26.354

Table 1: Loss and blue scores on Agnostic aware dataset.

Application: translation

Context aware translation (traning from scratch)

variant	val loss	test loss	val BLEU	test BLEU
*v*1	3.859	3.8	1.533	1.019
*v*2	3.86	3.837	1.944	0.882
*v*3	3.787	3.753	1.597	1.543
*v*4	3.823	3.806	1.698	1.291

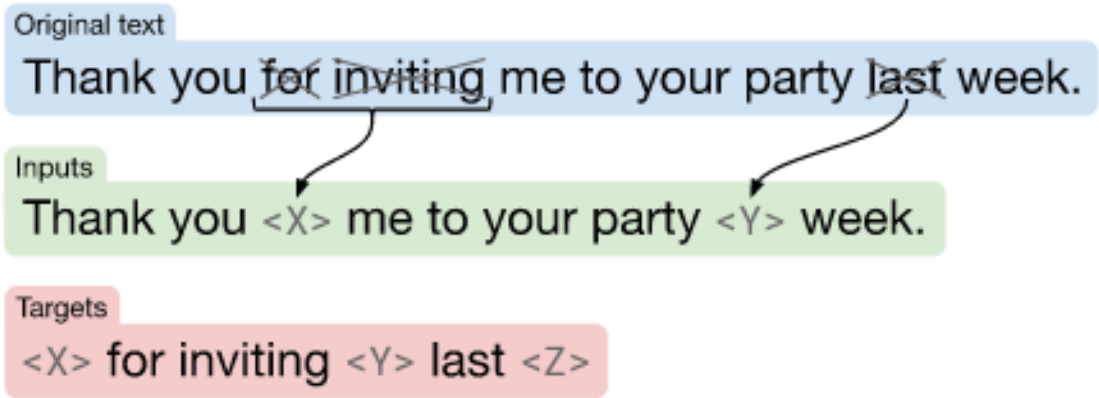
Table 2: Loss and blue scores on Context aware dataset before finetuning, training the the context aware from scratch.

Context-aware translation (finetuning)

variant	val loss	test loss	val BLEU	test BLEU
*v*1	1.678	1.665	26.12	**26.615**
*v*2	1.7	1.685	26.03	26.120
*v*3	1.698	1.667	25.88	26.356
*v*4	1.656	1.652	**26.64**	26.582

Table 3: Loss and blue scores on Context-aware data set after finetuning.

Application: MLM-masking pipeline and pre-training objective



Schematic of the objective we use in our baseline model. In this example, we process the sentence "Thank you for inviting me to your party last week." The words "for", "inviting" and "last" (marked with an ×) are randomly chosen for corruption. Each consecutive span of corrupted tokens is replaced by a sentinel token (shown as <X> and <Y>) that is unique over the example. Since "for" and "inviting" occur consecutively, they are replaced by a single sentinel <X>. The output sequence then consists of the dropped-out spans, delimited by the sentinel tokens used to replace them in the input plus a final sentinel token <Z>.

Figure 13: Schematic of the objective as original T5.

Application: MLM experiments setup

- Optimizer: adafactor
- Scheduler: linear
- perdevice 32, 8 gpus ere used; 32*8 = 256
- Learning rate 5-5e
- Epoches: 100
- Num_encoder layers: 2
- Num_decoder layers:2
- Num_experiment_runs: 2 ; except where otherwise noted.
- early stopping were not used

Application: results of pretraining on MLM task

All reported experiments used memory slot of two memory tokens and just two encoder layers and two decoder layers.

Table 4: Experimental results on train and dev set of wikitext-103-raw-v1 dataset for Masked language modeling task using just one chunk as input length 128. Best results for the proposed model are highlighted.

Expriment name	Valid loss	Perplexity
T5_baseline_128	3.455	32.265
T5Mem_MLM_1_chunk_128	**3.417**	**22.78**

Table 5: Experimental results on train and dev set of wikitext-103-raw-v1 dataset for Masked language modeling task using input length 512. Input length for all models is 512 length as one chunk or divided into four chunks of 128 chunk length. 128_to_512 means continue pre-training the model using 512 input length after training it for 100 epoches using input length 128 since the model is resilient to the input length.

no	Experiment name	Valid loss	Perplexity
1	T5_baseline_512	4.204	66.94
2	T5_128_to_512	4.589	98.46
3	T5Mem_MLM_1_chunk_512	**4.186**	**60.422**
4	T5Mem_MLM_1_to_4_chunks[a]	4.5147	91.35
5	T5Mem_MLM_4_chunks[a]	**4.184**	**65.66**

[a] Model has been trained once.

Model details: modifications

1. Dropping out chunk selector

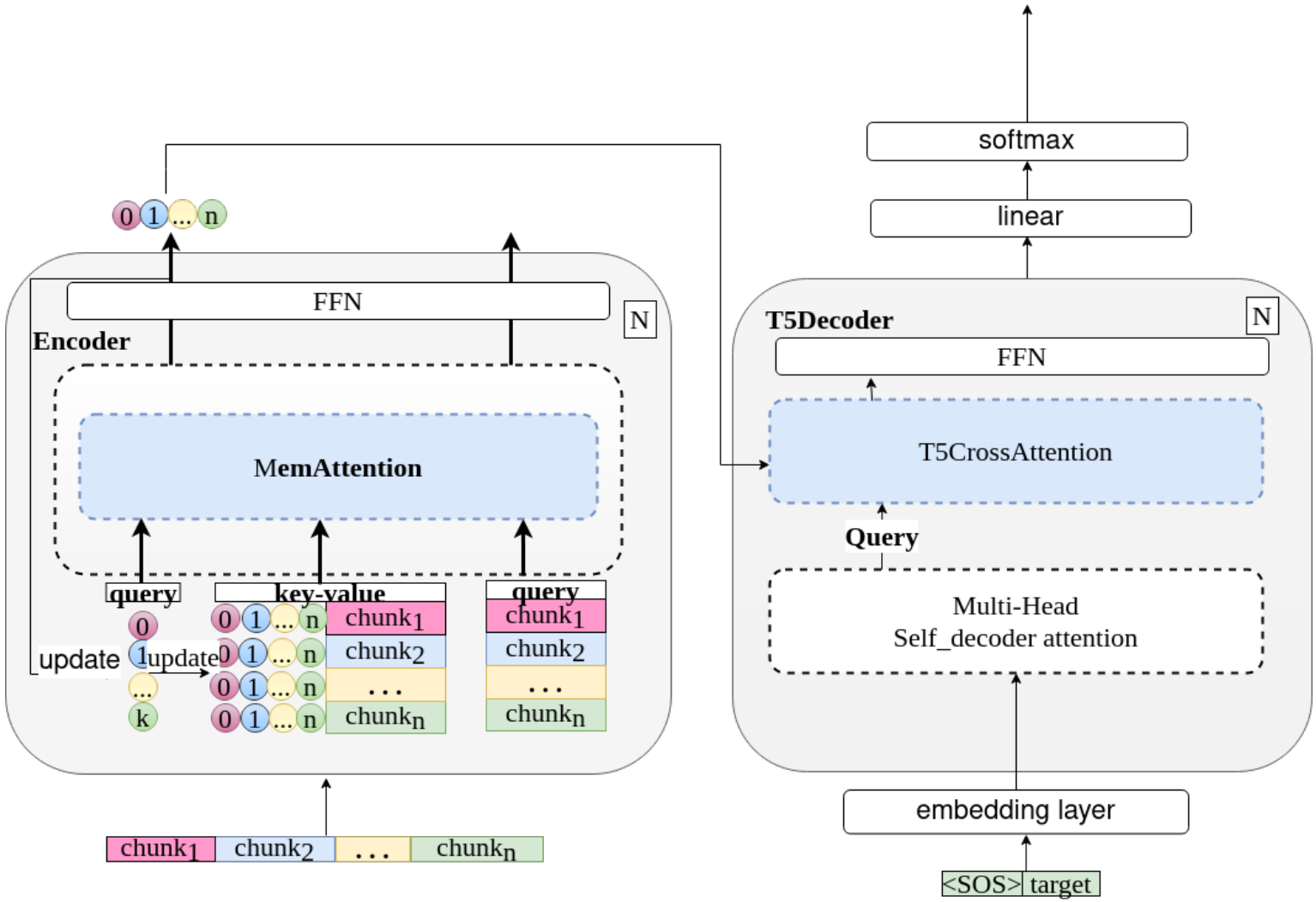


Figure 14: The main implemented model with dropped selector

Model details: modifications

2. Dropping out chunk selector and MemAttention

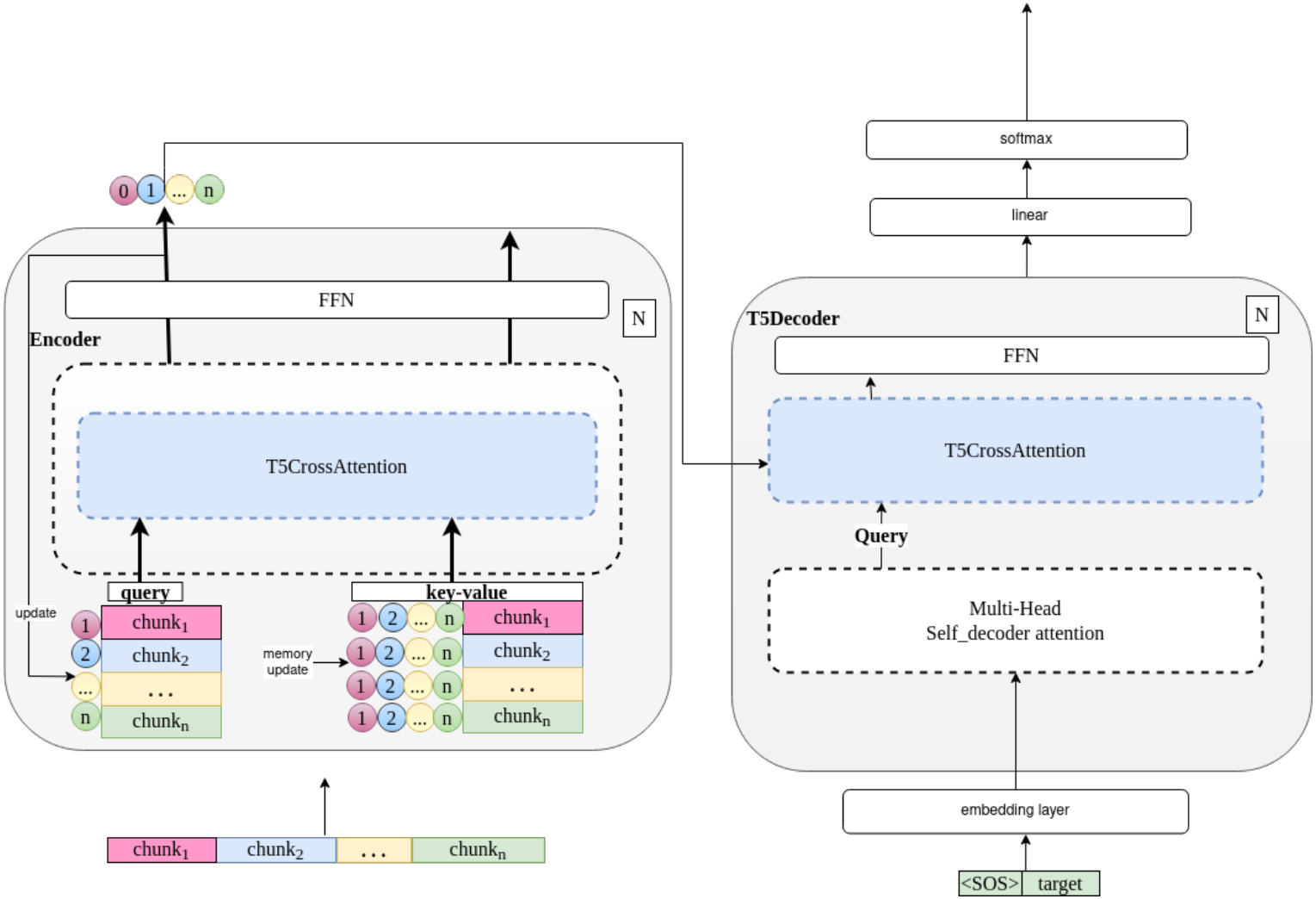


Figure 15: The main implemented model with dropped selector

Application: setup of experiments for modifications on MLM task

- Optimizer: adafactor for short AF, Adam for short Ad or linear from previous experiments
- Scheduler: linear from previous experiments , constant for short const
- Learning rate 5e-5
- Epoches: 100
- Num_encoder layers: 2
- Num_decoder layers: 2
- early stopping
- Num_experiment_runs: 2 ; except where otherwise noted.

Models:

- T5: as baseline
- T5Mem: the proposed model.
- T5MemWs_2mem: the proposed model without selector.
- T5MemWsWMA_2mem: the proposed model without selector and without MemAttention(T5Cross Attention is used instead).

Application: MLM results

Table 6: Experimental MLM task results using just 2 layers - training for 100 epochs input length is 512 for T5. For T5Mem model and its variants the input length 512 is divided into 4 chunks. optimizers(Adafactor "AF" and AdamW "Adm"), the used scheduler is **linear**. All models use two memory tokens for each chunk as a slot, except for T5MemWsWMA_1mem uses 1 memory token. Lines in grey use Adam Optimizer. The best results for the Adafactor optimizer are underlined. The best results for Adam Optimizer are bold.

Model	Modification	Valid loss.	val_acc. ↑	val_PPL ↓	Test loss.	test_acc. ↑	test_PPL↓
T5. AF,linear[1]		4.196	30.060	66.394	4.163	30.218	64.289
T5. Adm,linear		4.258	27.605	70.648	4.234	27.675	68.991
T5Mem. AF, linear[2,3]	original	4.171	29.775	64.769	4.140	29.808	62.816
T5Mem Adm,linear[2]	original	4.301	27.620	73.754	4.280	27.919	72.258
T5MemWs_2mem. AF,linear	1st	4.093	30.090	59.979	4.091	30.891	59.779
T5MemWs_2mem. Adm,linear	1st	4.143	**30.210**	**62.998**	4.113	**30.397**	**61.158**
T5MemWsWMA_1mem. AF,linear	2nd	4.050	31.145	57.417	3.999	31.355	54.589
T5MemWsWMA_1mem. Adm,linear	2nd	4.202	28.316	66.873	4.176	28.316	65.117
T5MemWsWMA_2mem. AF,linear	2nd	4.069	31.010	58.497	4.0217	31.262	55.813
T5MemWsWMA_2mem. Adm,linear	2nd	4.161	29.985	64.111	4.136	30.103	62.548

[1] It is the model T5_baseline_512 the name in table just to make it easy to compare in this table.

[2] Model has been trained once.

[3] It is the model T5Mem_MLM_4_chunks name was changed in the table to make it easy to compare

The proposed model outperformed the baseline model, notably using memory representation instead of using chunk representation in cross attention showed better results.

Application: MLM results

Table 7: Experimental MLM task results using just 2 layers - training for 100 epochs input length is 512 for T5. For T5Mem model and its variants the input length 512 is divided into 4 chunks. optimizers(Adafactor "AF" and AdamW "Adm"), the used scheduler is constant. All models use two memory tokens for each chunk as a slot, except for T5MemWsWMA_1mem uses 1 memory token. Lines in grey use Adam optimizer. Best results for Adafactor optimizer are underlined. Best results for Adam optimizer are bold. For all best results baseline is superior.

Model	modification	Valid loss.	val_acc. ↑	val_PPL ↓	Test loss.	test_acc. ↑	test_PPL↓
T5. AF,const	-	2.308	59.225	10.0595	2.265	60.150	9.634
T5. Adm,const	-	2.309	59.59	10.060	2.269	60.523	9.673
T5Mem. AF,const[a]	original	2.892	52.47	18.020	2.864	52.742	17.533
T5Mem. Adm,const[a]	original	3.078	47.77	21.720	3.757	33.659	42.822
T5MemWs_2mem. AF,const	1st	2.889	50.87	17.990	2.865	51.175	17.555
T5MemWs_2mem. Adm,const	1st	2.886	**51.005**	**17.916**	2.864	**51.213**	**17.531**
T5MemWsWMA_1mem. AF,const	2nd	3.157	44.805	23.495	3.126	45.216	22.789
T5MemWsWMA_1mem. Adm,const	2nd	3.104	47.275	22.290	3.076	47.656	21.677
T5MemWsWMA_2mem. AF,const	2nd	3.086	47.585	21.541	3.055	47.961	21.225
T5MemWsWMA_2mem. Adm,const	2nd	3.085	47.465	21.88	3.0535	47.806	21.195

[a] Model has been trained once.

Using constant optimizer the baseline outperform the proposed model and its variants, noting that using memory representation outcomes better results than original proposed model. Proposed model

Application: HotpotQA sample

HotPotQA dataset:

- proposed to encourage systems to learn more complex reasoning where the pieces of evidence to answer a question are scattered among different documents;
- consists of two parts full wiki dataset and distractor dataset (the gold paragraphs are available to a model in the distractor setting,).

Column	Value
id	'5a7a06935542990198eaf050'
question	"Which magazine was started first Arthur's Magazine or First for Women?"
answer	"Arthur's Magazine"
context	['Radio City is India's first private FM radio station and was started on 3 July 2001.', ' It broadcasts on 91.1 (earlier 91.0 in most cities) megahertz from Mumbai...]
type	'comparison'
level	'medium'
Supporting_facts	["Arthur's Magazine", 'First for Women']

Table 8: Application: example of HotpotQA dataset records.

Application: results of fine tuned modified models on HotpotQA

Table 9: Experimental results of fine tuning all pre-trained models and modified models on HotpotQA dataset(distractor part) using input length as 512 even as one chunk(T5 take input as chunk of 512 length) or 512 input length is divide it to 4 chunks. Lines in grey use Adam optimizer. All experiments were pretrained and finetuned using constant learning rate

Experiment name	Modification	Valid loss	Test loss	exact match	f1	recall	precission
T5_hp_AF_const_512		3.231	3.231	17.008	24.619	24.995	25.825
T5_hp_Adm_const_512		3.285	3.285	16.819	24.175	24.522	25.279
T5Mem_hp_4_chunks_AF_const	original	4.024	4.024	8.727	14.007	13.977	14.894
T5Mem_hp_4_chunks_Adm_const	original	4.150	4.150	4.090	5.507	5.480	5.781
T5MemWS2mem_hp_4_chunks_AF_const	1st	4.090	4.090	7.930	14.227	14.190	15.190
T5MemWS2mem_hp_4_chunks_Adm_const	1st	4.133	4.133	8.140	14.797	14.369	15.372
T5MemWSWMA1mem_hp_4_chunks_AF_const	2nd	4.052	4.052	7.876	13.905	13.911	14.904
T5MemWSWMA1mem_hp_4_chunks_Adm_const	2nd	4.331	4.331	6.965	12.657	12.610	13.587
T5MemWSWMA2mem_hp_4_chunks_AF_const	2nd	4.217	4.218	7.843	13.964	13.966	14.87
T5MemWSWMA2mem_hp_4_chunks_Adm_const	2nd	4.401	4.401	6.959	12.888	12.941	13.658

Baseline overcomes the model and its variants noting that the results of the proposed model were best when using memory representation.

Model modifications details: summarization task

	M1	M1	M2	M2	M3	M3	A	B	C	D	E	F	G	0	0
M1	0	0	1	1	2	2	1	2	3	4	5	6	7	8	9
M1	0	0	1	1	2	2	1	2	3	4	5	6	7	8	9
M2	-1	-1	0	0	1	1	-3	-2	-1	1	2	3	4	5	6
M2	-1	-1	0	0	1	1	-3	-2	-1	1	2	3	4	5	6
M3	-2	-2	-1	-1	0	0	-6	-5	-4	-3	-2	-1	1	2	3
M3	-2	-2	-1	-1	0	0	-6	-5	-4	-3	-2	-1	1	2	3
A	0	0	1	1	2	2	0	1	2						
B	0	0	1	1	2	2	-1	0	1						
C	0	0	1	1	2	1	-2	-1	0						
D	-1	-1	0	0	1	1				0	1	2			
E	-1	-1	0	0	1	1				-1	0	1			
F	-1	-1	0	0	1	1				-2	-1	0			
G	-2	-2	-1	-1	0	0							0	1	2
0	-2	-2	-1	-1	0	0							-1	0	1
0	-2	-2	-1	-1	0	0							-2	-1	0

Figure 16: overview of attention used beteen chunks and memory slots

	M1	M1	M2	M2	M3	M3	A	B	C
M1	0	0	1	1	2	2	1	2	3
M1	0	0	1	1	2	2	1	2	3
M2	-1	-1	0	0	1	1	1	2	3
M2	-1	-1	0	0	1	1	1	2	3
M3	-2	-2	-1	-1	0	0	1	2	3
M3	-2	-2	-1	-1	0	0	1	2	3
A	0	0	1	1	2	2	0	1	2
B	0	0	1	1	2	2	-1	0	1
C	0	0	1	1	2	2	-2	-1	0

Figure 17: example of masking and positional encoding used for each chunk

Application: used summarization datasets

Datasets used for summarization:

1. SAMSum: this dataset written by linguists used for abstractive summarization. It contains about 16k messenger-like conversations with summaries.
2. GovReport: originally proposed for summarizing long documents. It consists of U.S. government reports with expert-written abstractive summaries. This dataset is challenging, because the summary itself is long and scattered over very long document.
3. cnn_dailymail: this dataset can be used for both abstractive and extractive summarization. It has two fields article which is long text and highlights which is one or two sentences summary. These articles written by journalists at CNN between April 2007 and April 2015, and at Daily Mail between June 2010 and April 2015.

Dataset	Number of Instances in Split			Length of tokenized input			Length of tokenized target		
	Train	Validation	Test	Mean	Median	Max	Mean	Median	Max
SAMSum	14732	818	819	148	119	1153	28	25	94
GovReport	17517	973	973	10305	8571	324004	637	658	2360
cnn_dailymail	287113	13368	11490	985	898	5269	75	70	3151

Table 10: Statistics of the used summarization datasets. Input length is measured in tokens using a pre-trained T5 tokenizer.

Dataset	Maximum tokenized input length	Maximum tokenized target length	Maximum tokenized generated tokens length
SAMSum	no limit	94	94
GovReport	3072	384	384
cnn_dailymail	1024	128	128

Table 11: Length of models input, output, and length of generated text for evaluation and test. "no limit" means that the whole tokenized input was consumed by the models in all reported results.

Application: results of summarization task

Results on SAMSum dataset:

Model	Validation				Test				Loss		
	R1	R2	RL	RLsum	R1	R2	RL	RLsum	train	validation	test
T5-base(base line)	**52.543**	28.181	43.734	48.433	50.895	25.852	42.068	46.316	**1.199**	1.328	1.234
SLED_256	43.023	20.202	35.232	39.198	42.084	18.893	34.345	38.15	2.485	1.74	1.622
T5mem-base_256_32	52.092	27.855	43.46	47.994	50.394	25.904	42.032	45.97	1.201	1.332	1.241
T5mem-base_256_16	51.84	27.654	43.087	47.76	51.086	**26.672**	42.643	46.76	1.224	**1.328**	**1.233**
T5mem-base_256_8	52.084	27.815	43.664	48.047	**51.305**	26.346	**42.685**	**46.916**	1.22	1.347	1.24
T5mem-base_384_8	52.503	**28.23**	**43.853**	**48.507**	50.996	26.002	42.251	46.456	1.203	1.331	1.236

Table 12: Rouge metrics for the proposed model with T5 as baseline and SLED as another model for processing long documents on SAMSum dataset. 256 and 364 represent the block size, and 32,16,8 represent the number of memory tokens in each slot.

The proposed model outperformed the baseline and the contemporary model SLED.

Application: results of summarization task

Results on CNN/Dailymail dataset:

Model	Validation				Test				Loss		
	R1	R2	RL	RLsum	R1	R2	RL	RLsum	train	validation	test
T5-base (baseline)	44.074	21.421	31.154	41.024	43.21	20.647	30.566	40.102	1.262	1.409	1.443
SLED_256	42.196	19.99	29.865	39.14	-	-	-	-	1.609	-	-
T5mem-base_8mem_384b	43.67	21.023	30.759	40.603	42.905	20.483	30.381	39.871	1.255	1.394	1.422
T5mem-base_32mem_256b	43.547	26.971	30.76	40.532	42.777	20.332	30.226	39.723	1.295	1.442	1.455

Table 13: Comparing Rouge metrics for the proposed model with T5 as baseline and SLED as another model for processing long documents on CNN/DailyMail dataset.

The proposed model outperformed the baseline(T5) using R2 or on par with the baseline and outperforms SLED model results for all additional memory slots with different sizes.

Results on Govreport dataset:

Model	Validation				Test				Loss		
	R1	R2	RL	RLsum	R1	R2	RL	RLsum	train	validation	test
T5-base (baseline)	52.796	22.655	27.96	48.866	53.35	23.328	28.614	49.578	1.773	1.751	1.808
SLED1024_256	38.59	14.579	25.357	33.801	38.82	15.058	25.785	34.254	2.216	1.963	1.959
SLED3072_256	36.02	11.545	21.463	31.852	36.305	11.809	21.704	32.175	2.381	2.044	2.104
T5mem-base_8mem_256b	49.727	20.241	26.442	45.668	50.058	20.833	26.838	46.115	1.825	1.785	1.843

Table 14: Comparing Rouge metrics for the proposed model with T5 as baseline and SLED as another model for processing long documents on GovReport dataset.

The baseline outperformed both models but still our proposed model was better than SLED model(note that SLED as originally suggested for processing long documents)

Publications submitted for defense

1. **Al Adel A.**, Burtsev M.S. (2021). Memory transformer with hierarchical attention for long document processing. 2021 International Conference Engineering and Telecommunication (En&T), 1-7. Url: `https://ieeexplore.ieee.org/document/9681776`
2. **Al Adel, A.** (2022). Global Memory Transformer for Processing Long Documents. In Advances in Neural Computation, Machine Learning, and Cognitive Research VI. NEUROINFORMATICS 2022. Studies in Computational Intelligence, vol 1064. Springer, Cham. https://doi.org/10.1007/978-3-031-19032-2_36
3. **Al Adel, A.** (2023). SAMDIT: Systematic Study of Adding Memory to Divided Input in the Transformer to Process Long Documents. In: Kryzhanovsky, B., Dunin-Barkowski, W., Redko, V., Tiumentsev, Y., Klimov, V. (eds) Advances in Neural Computation, Machine Learning, and Cognitive Research VII. NEUROINFORMATICS 2023. Studies in Computational Intelligence, vol 1120. Springer, Cham. https://doi.org/10.1007/978-3-031-44865-2_10

Conclusion

In conclusion, the following theoretical and practical outcomes of overall model modification development stages are:

1. A review of proposed models for processing long documents for different NLP tasks;
2. New usage of the global tokens as memory slots to relate chunked inputs and as compressed representation;
3. Developing a proper masking mechanism and proper usage of the relative positional encoding for binding memory slots with related segments(chunks);

References I

[1] Joshua Ainslie et al. "ETC: Encoding Long and Structured Data in Transformers". In: *ArXiv* abs/2004.08483 (2020).

[2] Iz Beltagy, Matthew E. Peters, and Arman Cohan. "Longformer: The Long-Document Transformer". In: *ArXiv* abs/2004.05150 (2020).

[3] Sebastian Jaszczur et al. "Sparse is Enough in Scaling Transformers". In: *CoRR* abs/2111.12763 (2021).

[4] Tsvetomila Mihaylova and André F. T. Martins. "Scheduled Sampling for Transformers". In: *CoRR* abs/1906.07651 (2019).

[5] Xinwei Geng et al. "How Does Selective Mechanism Improve Self-Attention Networks?" In: *CoRR* abs/2005.00979 (2020).