Do We Really Need That Many Parameters In Transformer For Extractive Summarization? Discourse Can Help !

Wen Xiao, Patrick Huber and Giuseppe Carenini

University of British Columbia

Motivation

- The current summarization models are too large to train/finetune (e.g. BERTSUM: 118M [LL19])

Motivation

- The current summarization models are too large to train/finetune (e.g. BERTSUM: 118M [LL19])
  - Can we reduce the number of parameter to train/finetune?

Motivation

- The current summarization models are too large to train/finetune (e.g. BERTSUM: 118M [LL19])
  - Can we reduce the number of parameter to train/finetune?

- Light-weight attention modules have been proposed and applied on other tasks. [RST20][TBM+20]

Motivation

- The current summarization models are too large to train/finetune (e.g. BERTSUM: 118M [LL19])
  - > Can we reduce the number of parameter to train/finetune?

- Light-weight attention modules have been proposed and applied on other tasks. [RST20][TBM+20]
  - > Is it necessary to use heavy-weight dot-product self-attention in extractive summarization?

Motivation

- The current summarization models are too large to train/finetune (e.g. BERTSUM: 118M [LL19])
  - Can we reduce the number of parameter to train/finetune?

- Light-weight attention modules have been proposed and applied on other tasks. [RST20][TBM+20]
  - Is it necessary to use heavy-weight dot-product self-attention in extractive summarization?

- *Discourse trees are good indicators of importance in the text.* [Mar99]

Motivation

- The current summarization models are too large to train/finetune (e.g. BERTSUM: 118M [LL19])
  - Can we reduce the number of parameter to train/finetune?
- Light-weight attention modules have been proposed and applied on other tasks. [RST20][TBM+20]
  - Is it necessary to use heavy-weight dot-product self-attention in extractive summarization?
- *Discourse trees are good indicators of importance in the text.* [Mar99]
  - Applying discourse in the attention module might help reducing number of learnable parameters in the extractive summarization model.

What is Extractive Summarization?

Sent 1
(1) A 6.3-magnitude earthquake struck early sunday off Indonesia ,
(2) according to the U.S. geological survey.

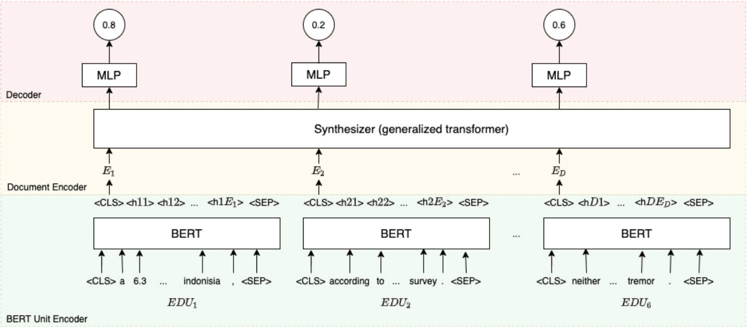
Sent 2
(3) The quake rattled a remote swath of sea between the Pacific and Indian oceans , north of Australia and east of Timor-leste, some 5.6 miles (9 kilometers) deep,
(4) according to the U.S. agency.

Sent 3
(5) It was centered approximately 212 miles (340 kilometers) west-northwest of Saumlaki in Indonesia 's Tanimbar Islands, 217 miles east-northeast of Dili, Timor-leste, and 226 miles of Ambon, Indonesia.

Sent 4
(6) Neither the Pacific Tsunami Warning Center nor the Japan Meteorological Agency issued Tsunami Warnings or advisories immediately after the tremor.

- Select **units** (e.g. EDUs,sentences,...) that can best represent the whole document
- Can be regarded as a sequence labeling problem

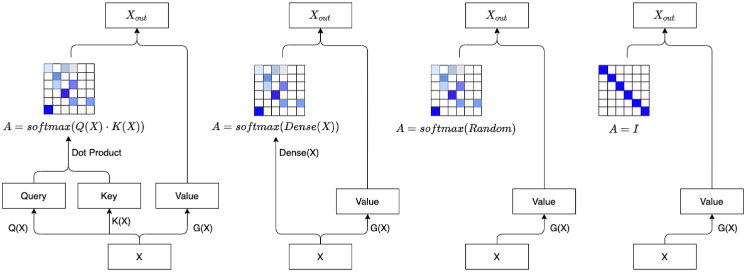
Extractive Summarization Model



- BERT Unit Encoder to get unit representation (EDU/Sentence)
- Synthesizer (generalized transformer) based Document Encoder to encode the units
- MLP Decoder to get the importance score of each unit

Synthesizer - A Generalized Transformer

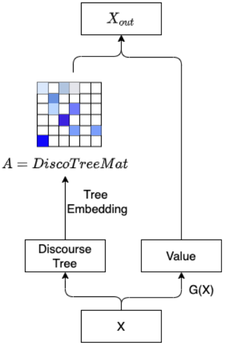
- Same structure as transformer
- It supports different attention modules [TBM+20]
  - (a) Dot-Product Self-Attention (original transformer)
  - (b) Dense Self-Attention
  - (c) Random Self-Attention (fixed or learnt)
  - (d) No attention (baseline model)
- We propose another self-attention module: Discourse Tree Attention.



(a) Dot Product (b) Dense (c) Random (d) No Attention

Discourse Tree Attention

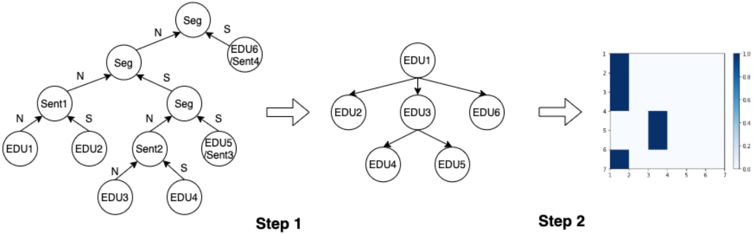
- Fixed attention, as the embedding of discourse tree
- Three variants of tree-to-matrix encoding:
  - Dependency Tree (mainly nuclearity information)
  - Constituency Tree (structure information only)
  - Constituency Tree with Nuclearity (structure + nuclearity)



Discourse Tree Attention

Variant #1 - Dependency Tree

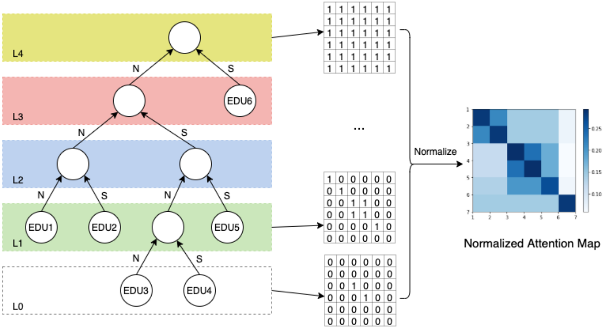
- Most downstream applications for discourse use the transformed dependency trees over constituency trees [Mar99, HYN+13, XGCL20]
- Step 1: constituency tree → dependency tree [HYN+13]
- Step 2: dependency tree → attention matrix [XGCL20]



Variant #2 - Constituency Tree

- Encode the compositional structure of the document
- The closer the units are in the discourse tree, the more attention they should pay to each other

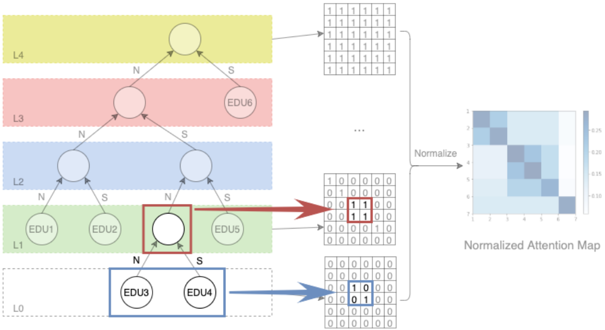
$$M_{ij}^{L} = \begin{cases} 1, & \text{if } EDU_i \text{ and } EDU_j \text{ in the same constituent at level } L \\ 0, & \text{otherwise} \end{cases}$$



Variant #2 - Constituency Tree

- Encode the compositional structure of the document
- The closer the units are in the discourse tree, the more attention they should pay to each other

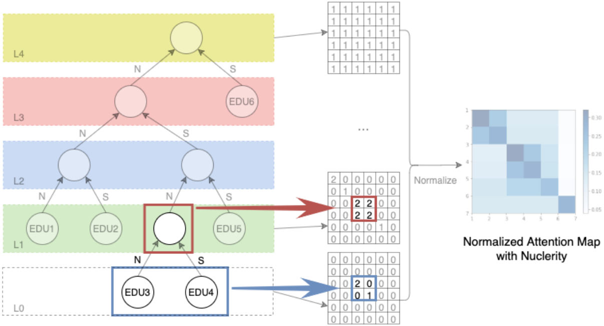
$$M_{ij}^{L} = \begin{cases} 1, & \text{if } EDU_i \text{ and } EDU_j \text{ in the same constituent at level } L \\ 0, & \text{otherwise} \end{cases}$$



Variant #3 - Constituency Tree with Nuclearity

- We also take Nuclearity into consideration

$$M_{ij}^{L} = \begin{cases} 2, & \text{if } EDU_i \text{ and } EDU_j \text{ in the same constituent at level } L \\ & \text{\& the node is } \textbf{Nucleus} \\ 1, & \text{if } EDU_i \text{and } EDU_j \text{ in the same constituent at level } L \\ & \text{\& the node is } \textbf{Satellite} \\ 0, & \text{otherwise} \end{cases}$$



Normalized Attention Map with Nuclerity

Experiments - Settings

- Dataset: CNNDM

#token/doc	#EDU/doc	#Sent/doc	#EDU(Oracle)	#Sent(Oracle)
546	70.2	27.2	6.4	3.1

- EDU Segmentor: top performing EDU Segmentor on RST-DT [WLY18]
- Discourse Parser: top performing Discourse Parser on RST-DT [WLW17]
- Evaluation Metric: ROUGE score
- We select the **top 6 EDUs** or **top 3 sentences** based on importance scores
- For all the models, we use two-layer synthesizer with 8 heads or single head.
- Hyper-parameter Setting can be found in the paper.

Experiment Results - EDU Level

	Model	Rouge-1	Rouge-2	Rouge-L	# Heads	# Params(attn)	# Params
	Default Setting ($d_k = d_v = d_q = 64$, $d_{inner} = 3072$)						
Learned	Dot Product(8)	**41.02**	**18.78**	**37.96**	8	3.2M	12.7M
Learned	Dot Product(1)	**40.92**‡	**18.69**‡	**37.85**‡	1	0.4M	9.9M
Learned	Dense	40.70	18.65†	37.74†	1	1.5M	11.0M
Learned	Learned Random	40.24	18.28	37.32	1	0.7M	10.3M
Fixed	Fixed Random	40.36	18.35	37.40	1	0.2M	9.7M
Fixed	No attention	39.89	17.98	36.99	1	0.2M	9.7M
Fixed	D-Tree	40.43	18.32	37.45	1	0.2M	9.7M
Fixed	C-Tree	40.80†	18.56	37.74†	1	0.2M	9.7M
Fixed	C-Tree w/Nuc	40.76	18.59†	37.73	1	0.2M	9.7M

Experiment Results - EDU Level

	Model	Rouge-1	Rouge-2	Rouge-L	# Heads	# Params(attn)	# Params
	Default Setting ($d_k = d_v = d_q = 64$, $d_{inner} = 3072$)						
Learned	Dot Product(8)	**41.02**	**18.78**	**37.96**	8	3.2M	12.7M
	Dot Product(1)	**40.92**‡	**18.69**‡	**37.85**‡	1	0.4M	9.9M
	Dense	40.70	18.65†	37.74†	1	1.5M	11.0M
	Learned Random	40.24	18.28	37.32	1	0.7M	10.3M
Fixed	Fixed Random	40.36	18.35	37.40	1	0.2M	9.7M
	No attention	39.89	17.98	36.99	1	0.2M	9.7M
	D-Tree	40.43	18.32	37.45	1	0.2M	9.7M
	C-Tree	40.80†	18.56	37.74†	1	0.2M	9.7M
	C-Tree w/Nuc	40.76	18.59†	37.73	1	0.2M	9.7M

- The C-Tree discourse tree attentions are better than all the other fixed attentions.

Experiment Results - EDU Level

	Model	Rouge-1	Rouge-2	Rouge-L	# Heads	# Params(attn)	# Params
	Default Setting ($d_k = d_v = d_q = 64$, $d_{inner} = 3072$)						
Learned	Dot Product(8)	**41.02**	**18.78**	**37.96**	8	3.2M	12.7M
	Dot Product(1)	**40.92**‡	**18.69**‡	**37.85**‡	1	0.4M	9.9M
	Dense	40.70	18.65†	37.74†	1	1.5M	11.0M
	Learned Random	40.24	18.28	37.32	1	0.7M	10.3M
Fixed	Fixed Random	40.36	18.35	37.40	1	0.2M	9.7M
	No attention	39.89	17.98	36.99	1	0.2M	9.7M
	D-Tree	40.43	18.32	37.45	1	0.2M	9.7M
	C-Tree	40.80†	18.56	37.74†	1	0.2M	9.7M
	C-Tree w/Nuc	40.76	18.59†	37.73	1	0.2M	9.7M

- They are competitive with the single-head learned attentions with less learnable parameters in the attention mechanism.

Experiment Results - EDU Level

	Model	Rouge-1	Rouge-2	Rouge-L	# Heads	# Params(attn)	# Params
	Default Setting ($d_k = d_v = d_q = 64$, $d_{inner} = 3072$)						
Learned	Dot Product(8)	**41.02**	**18.78**	**37.96**	8	3.2M	12.7M
	Dot Product(1)	**40.92**‡	**18.69**‡	**37.85**‡	1	0.4M	9.9M
	Dense	40.70	18.65†	37.74†	1	1.5M	11.0M
	Learned Random	40.24	18.28	37.32	1	0.7M	10.3M
Fixed	Fixed Random	40.36	18.35	37.40	1	0.2M	9.7M
	No attention	39.89	17.98	36.99	1	0.2M	9.7M
	D-Tree	40.43	18.32	37.45	1	0.2M	9.7M
	C-Tree	40.80†	18.56	37.74†	1	0.2M	9.7M
	C-Tree w/Nuc	40.76	18.59†	37.73	1	0.2M	9.7M

- The parameters in the attention module is only a small portion in the whole model, so we also test with a more balanced setting.

Experiment Results - EDU Level

Model	Rouge-1	Rouge-2	Rouge-L	# Heads	# Params(attn)	# Params
Balanced Models ($d_k = d_v = d_q = 512$, $d_{inner} = 512$)						
Dot Product(8)	**40.95**	**18.52**	**37.78**	8	25.2M	27M
Dot Product(1)	40.64	18.33	37.54	1	3.2M	4.8M
C-Tree w/Nuc	40.70	18.46†	37.63	1	1.6M	3.2M

- In this setting, the C-Tree w/Nuc is better than the single-head dot-product attention, and is competitive with the 8-head dot-product attention.

Experiment Results - Sentence Level

Model	Rouge-1	Rouge-2	Rouge-L	# Heads	# Params(attn)	# Params
Balanced Models ($d_k = d_v = d_q = 512$, $d_{inner} = 512$)						
Dot Product(8)	41.45	18.88	37.84	8	25.2M	27M
Dot Product(1)	41.51	18.95	37.94	1	3.2M	4.8M
C-Tree	**41.68**	**19.11**	**38.12**	1	1.6M	3.2M
C-Tree w/Nuc	41.64†	19.02†	38.06†	1	1.6M	3.2M

- C-tree discourse tree attentions achieves the best performance, and it's significantly better than single-head/8-head Dot-Product attentions.

Conclusion

- We extend and adapt the “Synthesizer” framework for extractive summarization by proposing a new discourse tree self-attention method.
- The empirical results show that our fixed tree attentions are significanly better than other fixed attention baselines, and comparable with the learned attentions on both EDU level and Sentence level.

Future Work

- Explore ways to also incorporate rhetorical relations into discourse tree attention.
- The C-Tree with Nuclearity doesn’t perform better than C-Tree, which may suggest more exploration should be done in terms of the representation of nuclearity.
- Explore the combination of different kinds of learned and fixed attentions to see if it helps improving the performance.
- Instead of two-level encoder, inject the tree attentions directly to the BERT Document Encoder.

Thanks!

References I

- Tsutomu Hirao, Yasuhisa Yoshida, Masaaki Nishino, Norihito Yasuda, and Masaaki Nagata, *Single-Document Summarization as a Tree Knapsack Problem*, Tech. report, 2013.
- Yang Liu and Mirella Lapata, *Text Summarization with Pretrained Encoders*, EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference (2019), 3730–3740.
- Daniel Marcu, *Discourse Trees are Good Indicators of Importance in Text*, Advances in Automatic Text Summarization (1999), 123–136.
- Alessandro Raganato, Yves Scherrer, and Jörg Tiedemann, *Fixed Encoder Self-Attention Patterns in Transformer-Based Machine Translation*.

References II

- Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng, *Synthesizer: Rethinking self-attention in transformer models*, 2020.
- Yizhong Wang, Sujian Li, and Houfeng Wang, *A two-stage parsing method for text-level discourse analysis*, Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), 2017, pp. 184–188.
- Yizhong Wang, Sujian Li, and Jingfeng Yang, *Toward fast and accurate neural discourse segmentation*, arXiv preprint arXiv:1808.09147 (2018).
- Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu, *Discourse-aware neural extractive text summarization*, Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (Online), Association for Computational Linguistics, July 2020, pp. 5021–5031.