Predicting Discourse Trees from Transformer-based Neural Summarizers

Wen Xiao, Patrick Huber and Giuseppe Carenini

University of British Columbia

Motivation

Discourse Tree

Extractive Summarization

Motivation

Discourse Tree

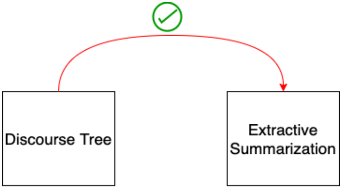
Extractive Summarization

- Discourse Tree: a document-level tree, reflects the structure, relationship and importance (nuclearity) of the document.

Motivation

- Extractive Summarization: pick the most important text units to represent the whole document.

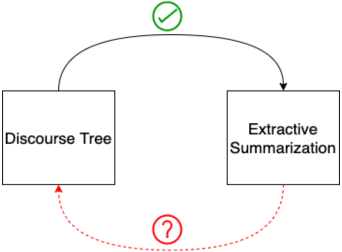
Motivation



Discourse tree is important for extractive summarization task:

- It is shown to be a good indicator of importance in text in **unsupervised method**. [Mar99]
- When **added to neural summarizers**, it helps improving the performance. [XGCL20]
- When used as **fixed attention** to replace the learnt self-attentions in transformer-based summarizer, it can achieve competitive performance.[XHC20]

Motivation

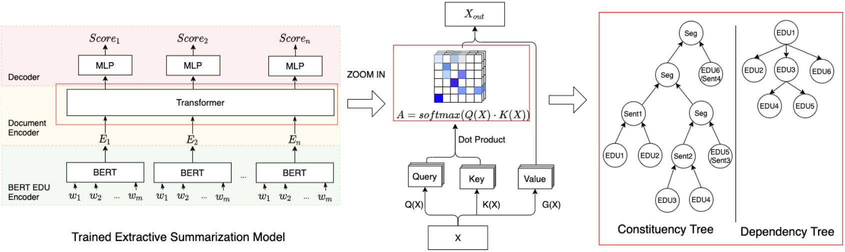


Q: Do Extractive Summarizers Learn Discourse Information?

Idea

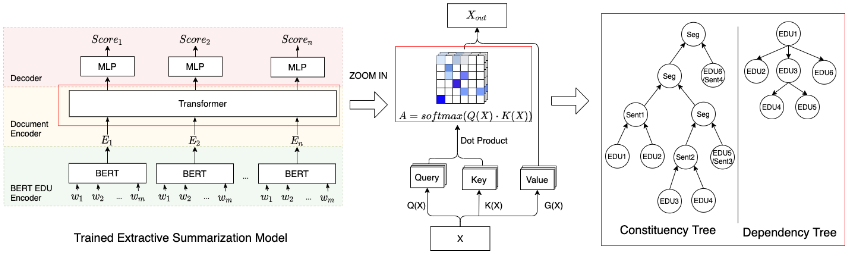
Build discourse trees based on the attention matrices of trained extractive summarization model, and verify whether and how they are aligned with human-annotated discourse trees.

Pipeline



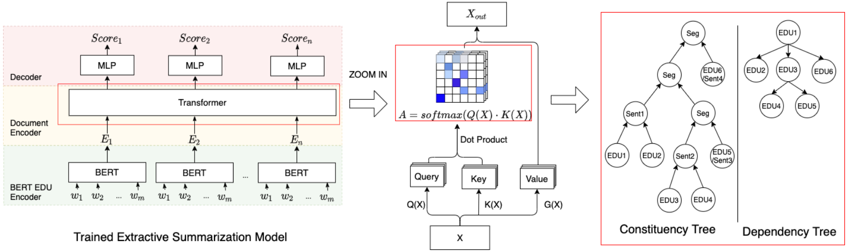
Score1
Score2
Scoren
MLP
MLP
MLP
Decoder
Transformer
Document Encoder
E1
E2
En
BERT
BERT
BERT
BERT EDU Encoder
w1 w2 ... wm
Trained Extractive Summarization Model
ZOOM IN
Xout
A = softmax(Q(X) · K(X))
Dot Product
Query
Key
Value
Q(X)
K(X)
G(X)
X
Seg
Seg
EDU6 /Sent4
Sent1
Seg
EDU1
EDU2
Sent2
EDU5 /Sent3
EDU3
EDU4
Constituency Tree
EDU1
EDU2
EDU3
EDU6
EDU4
EDU5
Dependency Tree

Pipeline



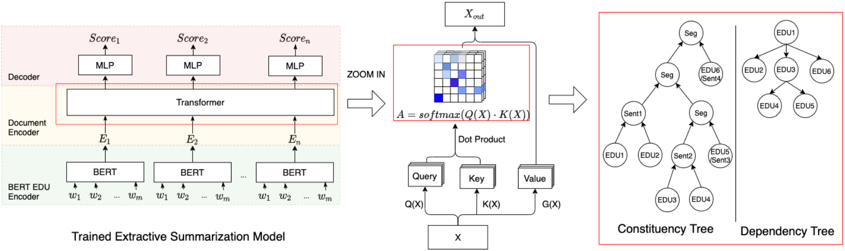
- Step 0: **Train a transformer-based extractive summarizer**

Pipeline



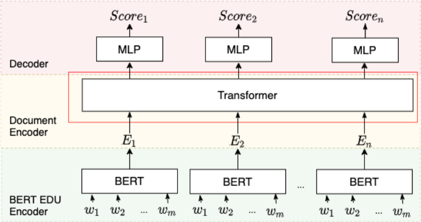
- Step 0: **Train a transformer-based extractive summarizer**
- Step 1: **Get the attention matrices** from the summarizer for any input document

Pipeline



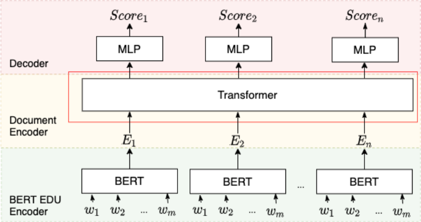
- Step 0: **Train a transformer-based extractive summarizer**
- Step 1: **Get the attention matrices** from the summarizer for any input document
- Step 2: Use the attention matrices to **build the discourse trees**

Step 0: Train the Neural Summarizer



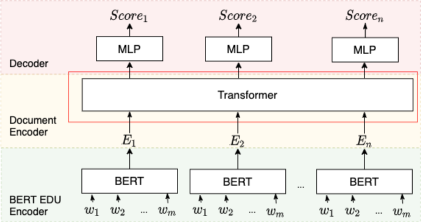
- Structure:

Step 0: Train the Neural Summarizer



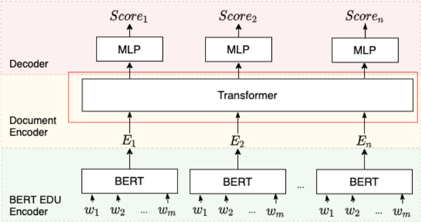
- Structure:
  - **BERT EDU Encoder**: get EDU representations from pre-trained BERT

Step 0: Train the Neural Summarizer



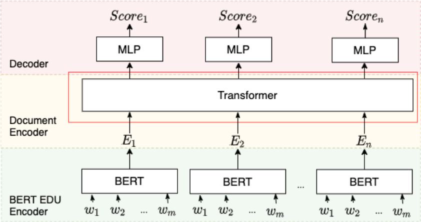
- Structure:
  - **BERT EDU Encoder**: get EDU representations from pre-trained BERT
  - **Transformer-based Document Encoder**: encode all the EDUs in the document

Step 0: Train the Neural Summarizer



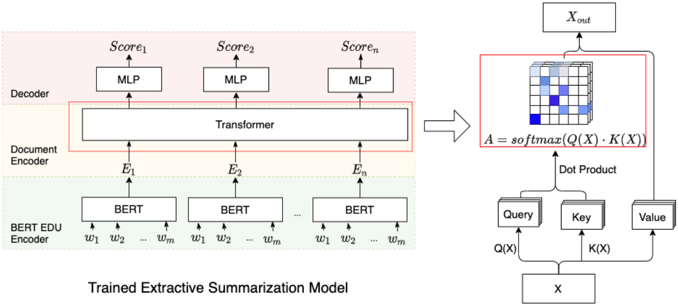
- Structure:
  - **BERT EDU Encoder**: get EDU representations from pre-trained BERT
  - **Transformer-based Document Encoder**: encode all the EDUs in the document
  - **Decoder**: a classifier to predict the score whether each EDU should be picked

Step 0: Train the Neural Summarizer



- Structure:
  - **BERT EDU Encoder**: get EDU representations from pre-trained BERT
  - **Transformer-based Document Encoder**: encode all the EDUs in the document
  - **Decoder**: a classifier to predict the score whether each EDU should be picked
- Dataset: CNNDM and NYT

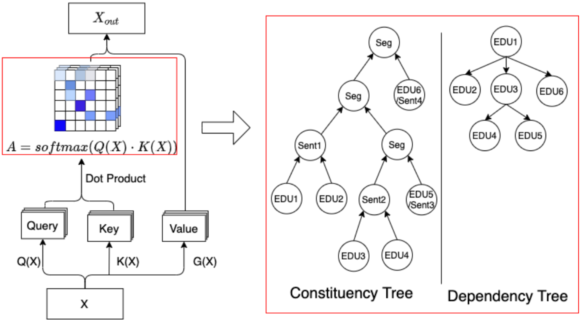
Step 1: Get Attentions



For any input document, we use two kinds of attention matrices

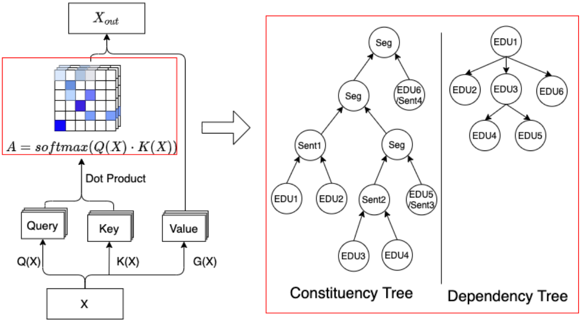
- Average attention matrices for each layer
- Attention matrices from all heads across all layers

Step 2: Build Discourse Trees



We build two kinds of discourse trees from the attention matrices:

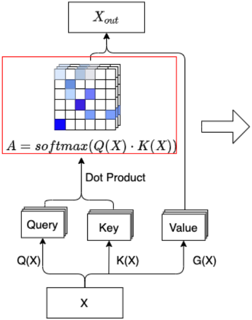
Step 2: Build Discourse Trees

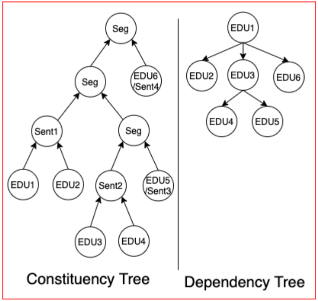


We build two kinds of discourse trees from the attention matrices:

- **Constituency Tree (structure only)**: to explore whether the structure information is captured

Step 2: Build Discourse Trees





We build two kinds of discourse trees from the attention matrices:

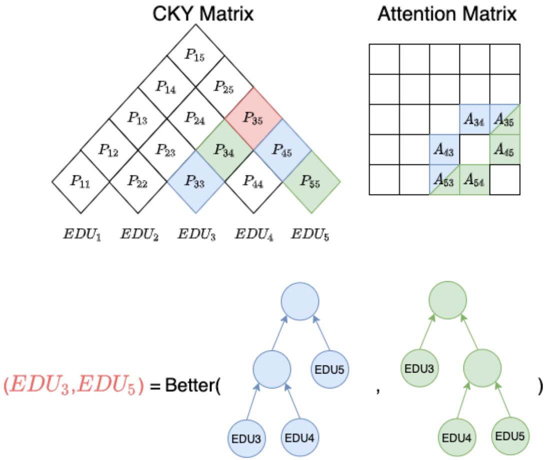
- **Constituency Tree (structure only)**: to explore whether the structure information is captured
- **Dependency Tree (projective/non-projective)**: to explore whether the dependency relationship between EDUs is captured

Step 2(a) - Build Constituency Trees

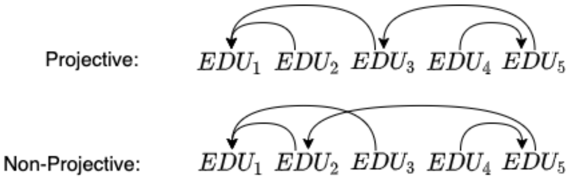
- **CKY Algorithm**[JM14]: a dynamic programming algorithm, build constituency tree in a bottom-up way.

Step 2(a) - Build Constituency Trees

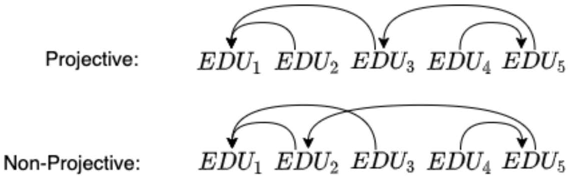
- **CKY Algorithm**[JM14]: a dynamic programming algorithm, build constituency tree in a bottom-up way.



Step 2(b) - Build Dependency Trees

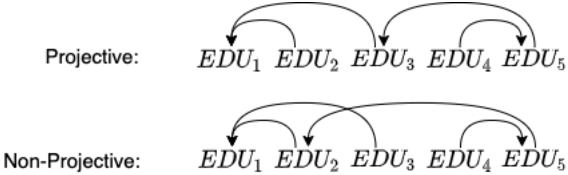


Step 2(b) - Build Dependency Trees



- **Eisner Algorithm**[Eis96]: a dynamic programming algorithm, build dependency tree in a bottom-up way, can only produce projective trees.

Step 2(b) - Build Dependency Trees



- **Eisner Algorithm**[Eis96]: a dynamic programming algorithm, build dependency tree in a bottom-up way, can only produce projective trees.
- **CLE Algorithm**[CL65, Edm67]: proposed to find the maximum spanning tree in the graph, and can produce both projective or non-projective trees.

Make Use of the Natural Structure of Documents

Sentence Constraint: units within the same sentence are aggregated before connecting with the units outside the sentence boundary

Make Use of the Natural Structure of Documents

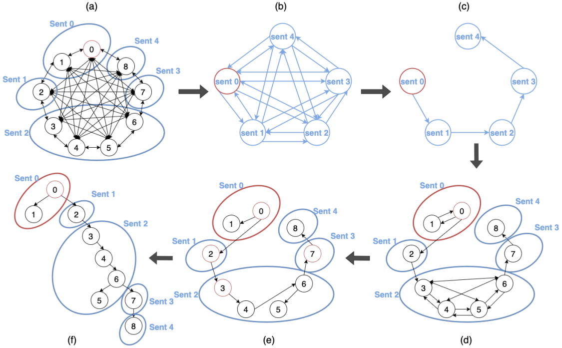
Sentence Constraint: units within the same sentence are aggregated before connecting with the units outside the sentence boundary

- **CKY Algorithm** / **Eisner Algorithm**: simply ignore options that do not meet the constraint

Make Use of the Natural Structure of Documents

Sentence Constraint: units within the same sentence are aggregated before connecting with the units outside the sentence boundary

- **CKY Algorithm / Eisner Algorithm**: simply ignore options that do not meet the constraint
- **CLE Algorithm**: construct and apply CLE on a sentence-level graph first, and then apply CLE within each sentence.



Experiments

Experiments

- Settings of summarizer:
  - 2 Layer, 1 Head
  - 2 Layer. 8 Head
  - 6 Layer, 8 Head

Experiments

- Settings of summarizer:
  - 2 Layer, 1 Head
  - 2 Layer. 8 Head
  - 6 Layer, 8 Head
- Discourse datasets with human-annotated discourse trees:

Dataset	# Docs	#EDU/doc	#Sent/doc	#words/doc
RST-DT[COM02]	385	56.6	22.5	549
Instruction[SDE09]	176	32.7	19.5	318
GUM[Zel17]	127	107	45	874

Experiments

- Settings of summarizer:
  - 2 Layer, 1 Head
  - 2 Layer. 8 Head
  - 6 Layer, 8 Head
- Discourse datasets with human-annotated discourse trees:

Dataset	# Docs	#EDU/doc	#Sent/doc	#words/doc
RST-DT[COM02]	385	56.6	22.5	549
Instruction[SDE09]	176	32.7	19.5	318
GUM[Zel17]	127	107	45	874

- Evaluation metric:
  - Constituency Tree:

$$\text{RST-Parseval Score} = \frac{\#\ \text{correct spans}}{\#\ \text{total spans}}$$

  - Dependency Tree:

$$\text{Unlabeled Attachment Score} = \frac{\#\ \text{correct dependencies}}{\#\ \text{total dependencies}}$$

Experiments - Overall

Model	CKY		Eisner		CLE	
	No Cons.	Sent Cons.	No Cons.	Sent Cons.	No Cons.	Sent Cons.
	RSTDT					
CNNDM-2-1	61.2 / 59.7	76.2 / 74.6	23.7 / 4.8	28.2 / 18.2	21.6 / 1.5	29.3 / 19.6
CNNDM-6-8	60.3 / 60.8	75.4 / 75.0	7.9 / 20.5	13.8 /27.8	7.3 / 17.3	16.1 / 28.5
Random	58.6 (0.1)	74.1 (0.1)	11.2 (0.2)	20.3 (0.2)	1.7 (0.08)	18.7 (0.1)

- Attention Matrices: the average attention matrices of the first two layers

Experiments - Overall

Model	CKY		Eisner		CLE	
	No Cons.	Sent Cons.	No Cons.	Sent Cons.	No Cons.	Sent Cons.
	RSTDT					
CNNDM-2-1	61.2 / 59.7	76.2 / 74.6	23.7 / 4.8	28.2 / 18.2	21.6 / 1.5	29.3 / 19.6
CNNDM-6-8	60.3 / 60.8	75.4 / 75.0	7.9 / 20.5	13.8 /27.8	7.3 / 17.3	16.1 / 28.5
Random	58.6 (0.1)	74.1 (0.1)	11.2 (0.2)	20.3 (0.2)	1.7 (0.08)	18.7 (0.1)

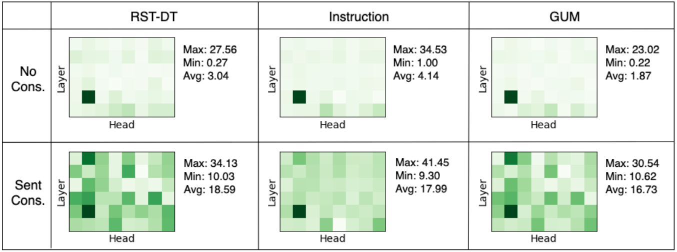
- Attention Matrices: the average attention matrices of the first two layers
- Both dependency and structural discourse information is learned implicitly in the summarization model

Experiments - Overall

Model	CKY		Eisner		CLE	
	No Cons.	Sent Cons.	No Cons.	Sent Cons.	No Cons.	Sent Cons.
	RSTDT					
CNNDM-2-1	61.2 / 59.7	76.2 / 74.6	23.7 / 4.8	28.2 / 18.2	21.6 / 1.5	29.3 / 19.6
CNNDM-6-8	60.3 / 60.8	75.4 / 75.0	7.9 / 20.5	13.8 /27.8	7.3 / 17.3	16.1 / 28.5
Random	58.6 (0.1)	74.1 (0.1)	11.2 (0.2)	20.3 (0.2)	1.7 (0.08)	18.7 (0.1)

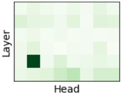
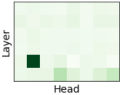
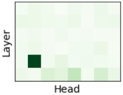
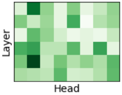
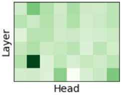
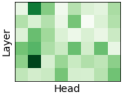
- Attention Matrices: the average attention matrices of the first two layers
- Both dependency and structural discourse information is learned implicitly in the summarization model
- More dependency information is captured, compared with the structural information.

Experiments - Per head (Dependency - CLE)



- Attention Matrices: the attention matrix from each head

Experiments - Per head (Dependency - CLE)

	RST-DT	Instruction	GUM
No Cons.	 Max: 27.56 Min: 0.27 Avg: 3.04	 Max: 34.53 Min: 1.00 Avg: 4.14	 Max: 23.02 Min: 0.22 Avg: 1.87
Sent Cons.	 Max: 34.13 Min: 10.03 Avg: 18.59	 Max: 41.45 Min: 9.30 Avg: 17.99	 Max: 30.54 Min: 10.62 Avg: 16.73

- Attention Matrices: the attention matrix from each head
- Discourse information is typically concentrated in a single head.

Experiments - Analysis of the Generated Trees (Best Head)

Measurement(%)	No Cons.	Sent Cons.
RST-DT		
Local Ratio Corr.	77.78	79.17
Instruction		
Local Ratio Corr.	81.15	84.90
GUM		
Local Ratio Corr.	77.99	80.20

$$\text{Local Ratio Corr.} = \frac{\text{\# correctly predicted local dependencies}}{\text{\# correctly predicted dependencies}}$$

Experiments - Analysis of the Generated Trees (Best Head)

Measurement(%)	No Cons.	Sent Cons.
RST-DT		
Local Ratio Corr.	77.78	79.17
Instruction		
Local Ratio Corr.	81.15	84.90
GUM		
Local Ratio Corr.	77.99	80.20

$$\text{Local Ratio Corr.} = \frac{\text{\# correctly predicted local dependencies}}{\text{\# correctly predicted dependencies}}$$

- The attention matrix works better on capturing the local dependencies (adjacent EDUs), meanwhile it also covers long distance discourse dependencies.

Experiments - Analysis of the Generated Trees (Best Head)

	Branch	Height	Leaf	Arc	vac. (%)
RST-DT					
Ours(No Cons)	1.74	25.76	0.49	0.12	3%
Ground-truth Tree	2.10	8.19	0.51	0.13	2%
Instruction					
Ours(No Cons)	1.80	14.35	0.50	0.14	3%
Ground-truth Tree	1.59	8.49	0.41	0.15	1%
GUM					
Ours(No Cons)	2.14	43.08	0.54	0.08	0%
Ground-truth Tree	2.02	12.17	0.51	0.04	0%

Experiments - Analysis of the Generated Trees (Best Head)

	Branch	Height	Leaf	Arc	vac. (%)
RST-DT					
Ours(No Cons)	1.74	25.76	0.49	0.12	3%
Ground-truth Tree	2.10	8.19	0.51	0.13	2%
Instruction					
Ours(No Cons)	1.80	14.35	0.50	0.14	3%
Ground-truth Tree	1.59	8.49	0.41	0.15	1%
GUM					
Ours(No Cons)	2.14	43.08	0.54	0.08	0%
Ground-truth Tree	2.02	12.17	0.51	0.04	0%

- The structure properties of our trees are similar to the ground-truth properties in regards to all measures except for the height of the tree

Conclusion

Answer to the question: The extractive summarization models do learn discourse information implicitly

Conclusion

Answer to the question: The extractive summarization models do learn discourse information implicitly

- More dependency information is learnt than constituency structural information.

Conclusion

Answer to the question: The extractive summarization models do learn discourse information implicitly

- More dependency information is learnt than constituency structural information.
- Most of the discourse information is concentrated on a single head.

Conclusion

Answer to the question: The extractive summarization models do learn discourse information implicitly

- More dependency information is learnt than constituency structural information.
- Most of the discourse information is concentrated on a single head.
- The generated trees have similar properties as the ground-truth trees, and it can capture not only local dependencies, but also long-distance dependencies.

Conclusion

Answer to the question: The extractive summarization models do learn discourse information implicitly

- More dependency information is learnt than constituency structural information.
- Most of the discourse information is concentrated on a single head.
- The generated trees have similar properties as the ground-truth trees, and it can capture not only local dependencies, but also long-distance dependencies.
- The consistent results across datasets and models suggest that the learned discourse information is general and transferable inter-domain.

Thanks!

References I

- Y. Chu and T. Liu, *On the shortest arborescence of a directed graph*, 1965.
- Lynn Carlson, Mary Ellen Okurowski, and Daniel Marcu, *Rst discourse treebank*, Linguistic Data Consortium, University of Pennsylvania, 2002.
- Jack Edmonds, *Optimum branchings*, 1967.
- Jason M. Eisner, *Three new probabilistic models for dependency parsing: An exploration*, COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics, 1996.
- Dan Jurafsky and James H Martin, *Speech and language processing*, vol. 3, Pearson London, 2014.
- Daniel Marcu, *Discourse Trees are Good Indicators of Importance in Text*, Advances in Automatic Text Summarization (1999), 123–136.

References II

- Rajen Subba and Barbara Di Eugenio, *An effective discourse parser that uses rich linguistic information*, Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Association for Computational Linguistics, 2009, pp. 566–574.
- 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.
- Wen Xiao, Patrick Huber, and Giuseppe Carenini, *Do we really need that many parameters in transformer for extractive summarization? discourse can help !*, Proceedings of the First Workshop on Computational Approaches to Discourse (Online), Association for Computational Linguistics, November 2020, pp. 124–134.

References III

Amir Zeldes, *The GUM corpus: Creating multilayer resources in the classroom*, Language Resources and Evaluation **51** (2017), no. 3, 581–612.