Topological Sort for Sentence Ordering

Shrimai Prabhumoye, Ruslan Salakhutdinov, Alan W Black
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, USA
{sprabhum, rsalakhu, awb}@cs.cmu.edu

Abstract

Sentence ordering is the task of arranging the sentences of a given text in the correct order. Recent work using deep neural networks for this task has framed it as a sequence prediction problem. In this paper, we propose a new framing of this task as a constraint solving problem and introduce a new technique to solve it. Additionally, we propose a human evaluation for this task. The results on both automatic and human metrics across four different datasets show that this new technique is better at capturing coherence in documents.

1 Introduction

Sentence ordering is the task of arranging sentences into an order which maximizes the coherence of the text (Barzilay and Lapata, 2008). This is important in applications where we have to determine the sequence of pre-selected set of information to be presented. This task has been well-studied in the community due to its significance in down stream applications such as ordering of: concepts in concept-to-text generation (Konstas and Lapata, 2012), information from each document in multi-document summarization (Barzilay and Elhadad, 2002; Nallapati et al., 2017), events in storytelling (Fan et al., 2019; Hu et al., 2019), cooking steps in recipe generation (Chandu et al., 2019), and positioning of new information in existing summaries for update summarization (Prabhumoye et al., 2019). Student essays are evaluated based on how coherent and well structured they are. Hence, automated essay scoring (Burstein et al., 2010; Miltsakaki and Kukich, 2004) can use this task to improve the efficiency of their systems.

Early work on coherence modeling and sentence ordering task uses probabilistic transition model based on vectors of linguistic features (Lapata, 2003), content model which represents topics as states in an HMM (Barzilay and Lee, 2004), and entity based approach (Barzilay and Lapata, 2008). Recent work uses neural approaches to model coherence and to solve sentence ordering task. Li and Hovy (2014) introduced a neural model based on distributional sentence representations using recurrent or recursive neural networks and avoided the need of feature engineering for this task. In (Li and Jurafsky, 2017), they extend it to domain independent neural models for coherence and they introduce new latent variable Markovian generative models to capture sentence dependencies. These models used windows of sentences as context to predict sentence pair orderings. Gong et al. (2016) proposed end-to-end neural architecture for sentence ordering task which uses pointer networks to utilize the contextual information in the entire piece of text.

Recently hierarchical architectures have been proposed for this task. In (Logeswaran et al., 2018), the model uses two levels of LSTMs to first get the encoding of the sentence and then get the encoding of the entire paragraph. Cui et al. (2018) use a transformer network for the paragraph encoder to allow for reliable paragraph encoding. Prior work (Logeswaran et al., 2018; Cui et al., 2018; Kumar et al., 2020) has treated this task as a sequence prediction task where the order of the sentences is predicted as a sequence. The decoder is initialized by the document representation and it outputs the index of sentences in sequential order. Only in (Chen et al., 2016), this task is framed as a ranking problem. In this work, a pairwise score is calculated between two sentences and then the final score for an order is obtained by summing over all the scores between pairs of sentences. The order which has the maximum score is given as output. Instead of considering all possible permutations of a given order, it uses beam-search strategy to find a sub-optimal order.

Most of the recent work (Gong et al., 2016; Logeswaran et al., 2018; Cui et al., 2018) tries to leverage the contextual information but has the limitation of predicting the entire sequence of the order. This has the drawback that the prediction at the current time step is dependent on the prediction of the previous time step. Another limitation of the prior work is the availability of good sentence representations that can help in determining the relative order between two sentences.

For this work we frame the task as a constraint learning problem. We train a model which learns to predict the correct constraint given a pair of sentences. The constraint learnt by our model is the relative ordering between the two sentences. Given a set of constraints between the sentences of a document, we find the right order of the sentences by using sorting techniques. Since we don't attach a score to an order, we don't have to consider all the permutations of an order.

Our main contribution is a new framing for the sentence ordering task as a constraint solving problem. We also propose a new and simple approach for this task in this new framework. We show that a simple sorting technique can outperform the previous approaches by a large margin given that it has good sentence representations. The bottleneck for most of the hierarchical models is memory required by the representations of all the sentences and the representation of the paragraph. The new framing also obviates these memory issues. The code can be found at `https://github.com/shrimai/Topological-Sort-for-Sentence-Ordering`. Additionally, we introduce a human evaluation for this task and show that our model outperforms the state-of-the-art on all the metrics.

2 Methodology

For our task we have a set of N documents $\mathcal{D} = \{d_1 \ldots, d_N\}$. Let the number of sentences in each document d_i be denoted by v_i, where $\forall i$, $v_i >= 1$. Our task can be formulated as - If we have a set $\{s_{o_1}, \ldots, s_{o_{v_i}}\}$ of v_i sentences in a random order where the random order is $\mathbf{o} = [o_1, \ldots, o_{v_i}]$, then the task is to find the right order of the sentences $\mathbf{o}^* = [o_1^*, \ldots, o_{v_i}^*]$. Prior work (Logeswaran et al., 2018; Cui et al., 2018) learns to predict the sequence of the correct order $\mathbf{o}^*$. In this formulation of the task, we have $\mathcal{C}_i$ set of constraints for document d_i. These constraints $\mathcal{C}_i$ represent the relative ordering between every pair of sentences in d_i. Hence, we have $|\mathcal{C}_i| = \binom{v_i}{2}$. For example, if a document has four sentences in the correct order $s_1 < s_2 < s_3 < s_4$, then we have six set of constraints $\{s_1 < s_2, s_1 < s_3, s_1 < s_4, s_2 < s_3, s_2 < s_4, s_3 < s_4\}$. Constraints $\mathcal{C}_i$ are learnt using a classifier neural network described in (§2.2). We finally find the right order $\mathbf{o}^*$ using topological sort on the relative ordering between all the $\mathcal{C}_i$ pairs of sentences.

2.1 Topological Sort

Topological sort (Tarjan, 1976) is a standard algorithm for linear ordering of the vertices of a directed graph. The sort produces an ordering $\hat{\mathbf{o}}$ of the vertices such that for every directed edge $u \to v$ from vertex u to vertex v, u comes before v in the ordering $\hat{\mathbf{o}}$. We use the depth-first search based algorithm which loops through each node of the graph, in an arbitrary order. The algorithm visits each node n and prepends it to the output ordering $\hat{\mathbf{o}}$ only after recursively calling the topological sort on all descendants of n in the graph. The algorithm terminates when it hits a node that has been visited or has no outgoing edges (i.e. a leaf node). Hence, we are guaranteed that all nodes which depend on n are already in the output ordering $\hat{\mathbf{o}}$ when the algorithm adds node n to $\hat{\mathbf{o}}$.

We use topological sort to find the correct ordering $\mathbf{o}^*$ of the sentences in a document. The sentences can represent the nodes of a directed graph and the directed edges are represented by the ordering between the two sentences. The direction of the edges are the constraints predicted by the classifier. For example, if the classifier predicts the constraint that sentence s_1 precedes s_2, then the edge $s_1 \to s_2$ would be from node of s_1 to s_2.

This algorithm has time complexity of $O(v_i + |\mathcal{C}_i|)$ for a document d_i. In our current formulation, all the constraints are predicted before applying the sort. Hence, we have to consider all the $|\mathcal{C}_i| = \binom{v_i}{2}$ edges in the graph. The time complexity of our current formulation is $O(v_i^2)$. But the same technique could be adopted using a Merge Sort (Knuth, 1998) algorithm in which case the time complexity would be $O(v_i \log v_i)$. In this case, the sort algorithm is applied first and the constraint is predicted only for the two sentences for which the relative ordering is required during the sort time.

2.2 Constraint Learning

We build a classifier to predict a constraint between two sentences s_1 and s_2 (say). The constraint learnt by the classifier is the relative ordering between the two sentences. Specifically, the classifier is trained to predict whether s_2 follows s_1 or not i.e the the classifier predicts the constraint $s_1 < s_2$.

BERT based Representation. (B-TSort) We use the Bidirectional Encoder Representations from Transformers (BERT) pre-trained uncased language model (Devlin et al., 2019) and fine-tune it on each dataset using a fully connected perceptron layer. Specifically, we leverage the Next Sentence Prediction objective of BERT and get a single representation for both sentences s_1 and s_2. The input to the BERT model is the sequence of tokens of sentence s_1, followed by the separator token '[SEP]', followed by the sequence of tokens for sentence s_2. We use the pooled representation for all the time steps[1].

LSTM based Representation. (L-TSort) In this model we get two separate representations $\mathbf{h_1}$ and $\mathbf{h_2}$ for s_1 and s_2 from a bi-directional LSTM encoder, respectively. We pass the concatenation of $\mathbf{h_1}$ and $\mathbf{h_2}$ as input to two layers of perceptron for constraint prediction. This model is trained to gain insight on the contribution of pre-trained sentence representations for the constraint prediction formulation of the task.

3 Experimental Results

This section describes the datasets, the evaluation metric and the results of our experiments. The hyper-paramater settings are reported in Apendix.

3.1 Datasets

NSF. NIPS, AAN abstracts. These three datasets contain abstracts from NIPS papers, ACL papers, and the NSF Research Award Abstracts dataset respectively and are introduced in (Logeswaran et al., 2018). The paper also provides details about the statistics and processing steps for curating these three datasets.

SIND caption. We also consider the SIND (Sequential Image Narrative Dataset) caption dataset (Huang et al., 2016) used in the sentence ordering task by (Gong et al., 2016). All the stories in this dataset contain five sentences each and we only consider textual stories for this task.

[1]This code was based on (Wolf et al., 2019).

3.2 Baselines

Attention Order Network (AON). This is the current state-of-the-art model (Cui et al., 2018) which formulates the sentence ordering task as a order prediction task. It uses a LSTM based encoder to learn the representation of a sentence. It then uses a transformer network based paragraph encoder to learn a representation of the entire document. It then decodes the sequence of the order by using a LSTM based decoder.

BERT Attention Order Network (B-AON). To have a fair comparison between our model and the AON model, we replace the LSTM based sentence representation with the pre-trained uncased BERT model. This model plays a pivotal role of giving us an insight into how much improvement in performance we get only due to BERT.

3.3 Evaluation Metric

Perfect Match (PMR): calculates the percentage of samples for which the entire sequence was correctly predicted (Chen et al., 2016). $\mathsf{PMR} = \frac{1}{N}\sum_{i=1}^{N} 1\{\hat{\mathbf{o}}^i = \mathbf{o}^{*i}\}$, where N is the number of samples in the dataset. It is the strictest metric.

Sentence Accuracy (Acc): measures the percentage of sentences for which their absolute position was correctly predicted (Logeswaran et al., 2018). $\mathsf{Acc} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{v_i}\sum_{j=1}^{v_i} 1\{\hat{\mathbf{o}}_j^i = \mathbf{o}_j^{*i}\}$, where v_i is the number of sentences in the i^{th} document. It is a also a stringent metric.

Kendall Tau (Tau): quantifies the distance between the predicted order and the correct order in terms of the number of inversions (Lapata, 2006). $\tau = 1 - 2I/\binom{v_i}{2}$, where I is the number of pairs in the predicted order with incorrect relative order and $\tau \in [-1, 1]$.

Rouge-S: calculates the percentage of skip-bigrams for which the relative order is predicted correctly (Chen et al., 2016). Skip-bigrams are the total number of pairs $\binom{v_i}{2}$ in a document. Note that it does not penalize any arbitrary gaps between two sentences as long as their relative order is correct. $\texttt{Rouge-S} = \frac{1}{\binom{v_i}{2}}\texttt{Skip}(\hat{\mathbf{o}}) \cap \texttt{Skip}(\mathbf{o}^*)$, where the $\texttt{Skip}(.)$ function returns the set of skip-bigrams of the given order.

Longest Common Subsequence (LCS): calculates the ratio of longest common sub-sequence (Gong et al., 2016) between the predicted order and the given order (consecutiveness is not necessary, and higher is better).

Model	PMR	Acc	Tau	Rouge-S	LCS
NIPS abstracts					
AON	16.25	50.50	0.67	80.97	74.38
B-AON	19.90	55.23	0.73	83.65	76.29
L-TSort	12.19	43.08	0.64	80.08	71.11
B-TSort	**32.59**	**61.48**	**0.81**	**87.97**	**83.45**
SIND captions					
AON	13.04	45.35	0.48	73.76	72.15
B-AON	14.30	47.73	0.52	75.77	73.48
L-TSort	10.15	42.83	0.47	73.59	71.19
B-TSort	**20.32**	**52.23**	**0.60**	**78.44**	**77.21**

Table 1: Results on NIPS and SIND datasets

Human Evaluation We introduce a human evaluation experiment to assess the orders predicted by the models. We set up a manual pairwise comparison following (Bennett, 2005) and present the human judges with two orders of the same piece of text. The judges are asked "Pick the option which is in the right order according to you." They can also pick a third option 'No Preference' which corresponds to both the options being equally good or bad. In total we had 100 stories from the SIND dataset[2] annotated by 10 judges. We setup three pairwise studies to compare the B-TSort vs AON order, B-TSort vs Gold order and AON vs Gold order (Gold order is the actual order of the text). Each judge annotated a total of 30 stories, 10 in each of the above mentioned categories. The judges were naive annotators.

3.4 Results

Table 1 shows the results of the automated metrics for the NIPS and SIND datasets[3]. It shows that AON[4] model gains on all metrics when the sentence embeddings are switched to BERT. The L-TSort model which does not utilize BERT embeddings comes close to AON performance on Rouge-S and Tau metrics. This demonstrates that the simple L-TSort method is as accurate as AON in predicting relative positions but not the absolute positions (PMR and Acc metric). Table 1 shows that our method B-TSort does not perform better only due to BERT embeddings but also due to the design of the experiment. Note that BERT has been trained with the Next Sentence Prediction objective and not the sentence ordering objective like ALBERT (Lan et al., 2020). We believe that framing this task as a constraint solving task will further benefit from pre-trained language model like ALBERT. Table 2 shows results for the NSF and AAN datasets and the B-TSort model performs better than the AON model on all metrics.

Model	PMR	Acc	Tau	Rouge-S	LCS
NSF abstracts					
AON	13.18	38.28	0.53	69.24	61.37
B-TSort	10.44	35.21	0.66	69.61	68.50
AAN abstracts					
AON	36.62	56.22	0.70	81.52	79.06
B-TSort	50.76	69.22	0.83	87.76	85.92

Table 2: Results on NSF and AAN datasets

B-TSort	No Preference	B-AON
41.00%	28.00%	31.00%
B-TSort	**No Preference**	**Gold**
26.00%	20.00%	**54.00%**
B-AON	**No Preference**	**Gold**
24.00%	22.00%	**54.00%**

Table 3: Human Evaluation Results on B-TSort vs AON (top), B-TSort vs Gold (middle) and AON vs Gold (bottom).

Table 3 shows results for the three human evaluation studies on the SIND dataset. It shows that human judges prefer B-TSort orders 10% more number of times than the B-AON orders[5]. The reference order may not be the only correct ordering of the story. The variability in the orders produced by B-TSort and B-AON is not very high and hence in comparison with Gold orders, we don't see much difference in human preferences.

The low scores of AON could be due to the fact that it has to decode the entire sequence of the order. The search space for decoding is very high (in the order of $v_i!$). Since our framework, breaks the problem to a pairwise constraint problem, the search space for our model is in the order of v_i^2.

Discussion: We perform additional analysis to determine the displacement of sentences in the predicted orders of the models, scalability of the models for longer documents, and an understanding of quality of the human judgements.

[2]We choose SIND because all the stories contain 5 sentences and hence it is easy to read for the judges. The orders of the stories are easier to judge as compared to the orders of scientific abstracts like NSF, NIPS and AAN as they require the judges to have an informed background.

[3]We fine-tune BERT which is memory intensive. Hence, we show the results of B-AON only on these two datasets as they need 2 transformer layers for paragraph encoder (Cui et al., 2018)

[4]We use the code provided by the authors to train the AON and B-AON model. The numbers reported in Table 1 and 2 are our runs of the model. Hence, they differ from the numbers reported in the paper (Cui et al., 2018).

[5]Examples of B-TSort and B-AON orders are shown in Table 6 and 7 for SIND and NIPS dataset in Appendix.

Model	Win=1	Win=2	Win=3	% Miss	Win=1	Win=2	Win=3	% Miss
	NIPS				**SIND**			
B-AON	81.81	92.44	96.50	3.48	78.39	92.79	98.43	0.00
B-TSort	87.59	95.59	98.11	0.00	82.67	95.01	99.09	0.00
	NSF				**AAN**			
AON	50.58	63.87	72.96	5.85	82.65	92.25	96.73	0.84
B-TSort	61.41	75.52	83.87	0.00	90.56	96.78	98.71	0.00

Table 4: Sentence Displacement Analysis for all the datasets. (Win=Window size; % Miss=% mismatch)

Displacement of sentences in predicted orders is measured by calculating the percentage of sentences whose predicted location is within 1, 2 or 3 positions (in either direction) from their original location. A higher percentage indicates less displacement of sentences. We observed that in spite of lack of a global structure, B-TSort consistently performs better on all datasets for all three window sizes as shown in Table 4. Observe that as window size reduces, the difference between B-TSort and B-AON percentages increases. This implies that displacement of sentences is higher in B-AON despite taking the whole document into account.

We additionally perform a comparison of models on documents containing more than 10 sentences and the results are shown in Table 5. B-TSort consistently performs better on all the metrics. SIND dataset is omitted in these experiments as the maximum number of sentences in the story is five for all the stories in the dataset. For each dataset, the Tau difference for longer documents is much higher than the Tau difference on the overall dataset (Table 1 and 2). This implies that B-TSort performs much better for longer documents.

Note that the AON model generates the order and hence need not generate positions for all the sentences in the input. We calculate the percentage of mismatches between the length of the input document and the generated order. For AON model on the NSF dataset which has longest documents, the overall mismatch is 5.85% (Table 4), while the mismatch for documents with more than 10 sentences is 11.60%. The AON model also produces an overall mismatch of 0.84 % on AAN documents while producing a mismatch of 5.17% on longer AAN documents. Similarly, the B-AON model has an overall mismatch of 3.48% for NIPS dataset, and 33.33% mismatch for longer documents. This problem does not arise in our design of the task as it does not have to stochastically generate orders.

To better understand the choices of human judges, we observe the average length of stories calculated in number of tokens. For the B-TSort vs B-AON study, we discover that the average length of the stories for B-TSort, B-AON and 'No Preference' chosen options is 86, 65 and 47 respectively. This means that B-TSort is better according to human judges for longer stories. Similarly for B-TSort vs Gold experiment, the human judges were confused with longer stories, reiterating that B-TSort performs well with long stories.

Model	PMR	Acc	Tau	Rouge-S	LCS
	NIPS abstracts				
B-AON	0.0	29.18	0.51	74.64	63.81
B-TSort	0.0	39.43	0.74	83.26	71.68
	NSF abstracts				
AON	2.12	21.42	0.41	67.45	55.47
B-TSort	0.67	28.57	0.64	68.46	64.86
	AAN abstracts				
AON	0.0	22.70	0.40	68.90	56.19
B-TSort	0.0	36.86	0.69	78.52	72.01

Table 5: Analysis on NIPS, NSF and AAN datasets for documents longer than 10 sentences.

4 Conclusion and Future Work

We have shown a new way to design the task of sentence ordering. We provide a simple yet efficient method to solve the task which outperforms the state of the art technique on all metrics. We acknowledge that our current model has the limitation of not including the entire context of the paragraph while making the decision of the relative order of the pairs. Our future work is to include the paragraph representation in the constraint prediction model. This will help our methodology to have the benefit of making informed decision while also solving constraints.

Acknowledgments

This work was supported in part by ONR Grant N000141812861, NSF IIS1763562, and Apple. We would also like to acknowledge NVIDIAs GPU support.

References

Regina Barzilay and Noemie Elhadad. 2002. Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. *arXiv preprint cs/0405039*.

Christina L Bennett. 2005. Large scale evaluation of corpus-based synthesizers: Results and lessons from the blizzard challenge 2005. In *Ninth European Conference on Speech Communication and Technology*.

Jill Burstein, Joel Tetreault, and Slava Andreyev. 2010. Using entity-based features to model coherence in student essays. In *Human language technologies: The 2010 annual conference of the North American chapter of the Association for Computational Linguistics*, pages 681–684.

Khyathi Chandu, Eric Nyberg, and Alan W Black. 2019. Storyboarding of recipes: Grounded contextual generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6040–6046, Florence, Italy. Association for Computational Linguistics.

Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2016. Neural sentence ordering. *arXiv preprint arXiv:1607.06952*.

Baiyun Cui, Yingming Li, Ming Chen, and Zhongfei Zhang. 2018. Deep attentive sentence ordering network. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4340–4349.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Angela Fan, Mike Lewis, and Yann Dauphin. 2019. Strategies for structuring story generation. *arXiv preprint arXiv:1902.01109*.

Jingjing Gong, Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2016. End-to-end neural sentence ordering using pointer network. *arXiv preprint arXiv:1611.04953*.

Junjie Hu, Yu Cheng, Zhe Gan, Jingjing Liu, Jianfeng Gao, and Graham Neubig. 2019. What makes a good story? designing composite rewards for visual storytelling. *arXiv preprint arXiv:1909.05316*.

Ting-Hao K. Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, Aishwarya Agrawal, Ross Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, et al. 2016. Visual storytelling. In *15th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2016)*.

Donald Ervin Knuth. 1998. *The art of computer programming, , Volume III, 2nd Edition*. Addison-Wesley.

Ioannis Konstas and Mirella Lapata. 2012. Concept-to-text generation via discriminative reranking. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 369–378. Association for Computational Linguistics.

Pawan Kumar, Dhanajit Brahma, Harish Karnick, and Piyush Rai. 2020. Deep attentive ranking networks for learning to order sentences. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Mirella Lapata. 2003. Probabilistic text structuring: Experiments with sentence ordering. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 545–552. Association for Computational Linguistics.

Mirella Lapata. 2006. Automatic evaluation of information ordering: Kendall's tau. *Computational Linguistics*, 32(4):471–484.

Jiwei Li and Eduard Hovy. 2014. A model of coherence based on distributed sentence representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2039–2048.

Jiwei Li and Dan Jurafsky. 2017. Neural net models of open-domain discourse coherence. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 198–209.

Lajanugen Logeswaran, Honglak Lee, and Dragomir Radev. 2018. Sentence ordering and coherence modeling using recurrent neural networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Eleni Miltsakaki and Karen Kukich. 2004. Evaluation of text coherence for electronic essay scoring systems. *Natural Language Engineering*, 10(1):25–55.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Shrimai Prabhumoye, Chris Quirk, and Michel Galley. 2019. Towards content transfer through grounded text generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2622–2632, Minneapolis, Minnesota. Association for Computational Linguistics.

Robert Endre Tarjan. 1976. Edge-disjoint spanning trees and depth-first search. *Acta Informatica*, 6(2):171–185.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

A Appendix

Hyper-parameters. For AON model we use the code base provided by the authors in (Cui et al., 2018) and we maintain the hyper-parameters described in the paper. For the paragraph encoder of the B-AON models, we follow the same scheme of the AON model but for its sentence encoder we use hyper-parameters of the BERT setting. We use the pretrained BERT uncased base model with 12 layers for the B-AON and B-TSORT models. We fine-tune the BERT model in both cases. Hence, we replace the Adadelta optimizer with the BertAdam (Wolf et al., 2019) optimizer for the B-AON model. The LSTMs in the L-TSort model uses an RNN size of 512 and it uses the same vocabularies as the AON model. L-TSort is trained using stochastic gradient descent with dropout of 0.2, learning rate of 1.0 and learning decay rate of 0.5. For B-TSort and L-TSort we use accuracy on the validation set to stop training. For B-TSort and B-AON we use learning rate of 5e-5 with adam epsilon value of 1e-8. For all the experiments we use a maximum sequence length of 105 tokens.

Gold Order	**B-TSort Order**	**B-AON Order**
	SIND Dataset	
the family sits together for dinner on the first night of the annual reunion. the restaurant we chose had amazing food and everyone loved the presentation. gemma really adored the restaurants decorations and was always gazing at them. aunt harriot had a little trouble deciding what kind of wine she wanted tonight. bob had the whole family cracking up with his jokes.	the family sits together for dinner on the first night of the annual reunion. the restaurant we chose had amazing food and everyone loved the presentation. aunt harriot had a little trouble deciding what kind of wine she wanted tonight. gemma really adored the restaurants decorations and was always gazing at them. bob had the whole family cracking up with his jokes.	the family sits together for dinner on the first night of the annual reunion. aunt harriot had a little trouble deciding what kind of wine she wanted tonight. bob had the whole family cracking up with his jokes. gemma really adored the restaurants decorations and was always gazing at them. the restaurant we chose had amazing food and everyone loved the presentation.
he wanted to take a ride on his new bike. we went on a nice ride out to the lake. we really enjoyed the beautiful view from the dock. it was very peaceful watching the boats. we had such a busy day he needed a nap.	we went on a nice ride out to the lake. he wanted to take a ride on his new bike. we really enjoyed the beautiful view from the dock. it was very peaceful watching the boats. we had such a busy day he needed a nap.	we went on a nice ride out to the lake. he wanted to take a ride on his new bike. it was very peaceful watching the boats. we really enjoyed the beautiful view from the dock. we had such a busy day he needed a nap.
when we finally brought our son home from the hospital so many people were at home with us to see him. everyone wanted a chance to hold him! we were all so happy to have a new addition to the family. my parents were so proud to be grand parents! i am so happy and i love my son very much!	when we finally brought our son home from the hospital so many people were at home with us to see him. we were all so happy to have a new addition to the family. everyone wanted a chance to hold him! my parents were so proud to be grand parents! i am so happy and i love my son very much!	my parents were so proud to be grand parents! when we finally brought our son home from the hospital so many people were at home with us to see him. we were all so happy to have a new addition to the family. everyone wanted a chance to hold him! i am so happy and i love my son very much!

Table 6: Examples of predicted sentence orders for B-TSort and B-AON model for SIND dataset.

Gold Order	B-TSort Order	B-AON Order
	NIPS Dataset	
we study how well one can recover sparse principal components of a data matrix using a sketch formed from a few of its elements. we show that for a wide class of optimization problems, if the sketch is close (in the spectral norm) to the original data matrix, then one can recover a near optimal solution to the optimization problem by using the sketch. in particular, we use this approach to obtain sparse principal components and show that for m data points in n dimensions, o(-2k maxm, n) elements gives an - additive approximation to the sparse pca problem (k is the stable rank of the data matrix). we demonstrate our algorithms extensively on image, text, biological and financial data. the results show that not only are we able to recover the sparse pcas from the incomplete data, but by using our sparse sketch, the running time drops by a factor of five or more.	we study how well one can recover sparse principal components of a data matrix using a sketch formed from a few of its elements. we show that for a wide class of optimization problems, if the sketch is close (in the spectral norm) to the original data matrix, then one can recover a near optimal solution to the optimization problem by using the sketch. in particular, we use this approach to obtain sparse principal components and show that for m data points in n dimensions, o(-2k maxm, n) elements gives an - additive approximation to the sparse pca problem (k is the stable rank of the data matrix). the results show that not only are we able to recover the sparse pcas from the incomplete data, but by using our sparse sketch, the running time drops by a factor of five or more. we demonstrate our algorithms extensively on image, text, biological and financial data.	we study how well one can recover sparse principal components of a data matrix using a sketch formed from a few of its elements. in particular, we use this approach to obtain sparse principal components and show that for m data points in n dimensions, o(-2k maxm, n) elements gives an - additive approximation to the sparse pca problem (k is the stable rank of the data matrix). we show that for a wide class of optimization problems, if the sketch is close (in the spectral norm) to the original data matrix, then one can recover a near optimal solution to the optimization problem by using the sketch. the results show that not only are we able to recover the sparse pcas from the incomplete data, but by using our sparse sketch, the running time drops by a factor of five or more. we demonstrate our algorithms extensively on image, text, biological and financial data.
we develop a latent variable model and an efficient spectral algorithm motivated by the recent emergence of very large data sets of chromatin marks from multiple human cell types . a natural model for chromatin data in one cell type is a hidden markov model (hmm) ; we model the relationship between multiple cell types by connecting their hidden states by a fixed tree of known structure . the main challenge with learning parameters of such models is that iterative methods such as em are very slow , while naive spectral methods result in time and space complexity exponential in the number of cell types . we exploit properties of the tree structure of the hidden states to provide spectral algorithms that are more computationally efficient for current biological datasets . we provide sample complexity bounds for our algorithm and evaluate it experimentally on biological data from nine human cell types . finally , we show that beyond our specific model , some of our algorithmic ideas can be applied to other graphical models .	a natural model for chromatin data in one cell type is a hidden markov model (hmm) ; we model the relationship between multiple cell types by connecting their hidden states by a fixed tree of known structure . the main challenge with learning parameters of such models is that iterative methods such as em are very slow , while naive spectral methods result in time and space complexity exponential in the number of cell types . we develop a latent variable model and an efficient spectral algorithm motivated by the recent emergence of very large data sets of chromatin marks from multiple human cell types . we exploit properties of the tree structure of the hidden states to provide spectral algorithms that are more computationally efficient for current biological datasets . we provide sample complexity bounds for our algorithm and evaluate it experimentally on biological data from nine human cell types . finally , we show that beyond our specific model , some of our algorithmic ideas can be applied to other graphical models .	the main challenge with learning parameters of such models is that iterative methods such as em are very slow , while naive spectral methods result in time and space complexity exponential in the number of cell types . a natural model for chromatin data in one cell type is a hidden markov model (hmm) ; we model the relationship between multiple cell types by connecting their hidden states by a fixed tree of known structure .', 'we develop a latent variable model and an efficient spectral algorithm motivated by the recent emergence of very large data sets of chromatin marks from multiple human cell types . we exploit properties of the tree structure of the hidden states to provide spectral algorithms that are more computationally efficient for current biological datasets . we provide sample complexity bounds for our algorithm and evaluate it experimentally on biological data from nine human cell types . finally , we show that beyond our specific model , some of our algorithmic ideas can be applied to other graphical models .

Table 7: Examples of predicted sentence orders for B-TSort and B-AON model for NIPS dataset.