Single Document Summarization as Tree Induction

Yang Liu, Ivan Titov and **Mirella Lapata**
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh EH8 9AB
yang.liu2@ed.ac.uk, ititov@inf.ed.ac.uk, mlap@inf.ed.ac.uk

Abstract

In this paper we conceptualize single-document extractive summarization as a tree induction problem. In contrast to previous approaches (Marcu, 1999; Yoshida et al., 2014) which have relied on linguistically motivated document representations to generate summaries, our model *induces* a multi-root dependency tree while predicting the output summary. Each root node in the tree is a summary sentence, and the subtrees attached to it are sentences whose content relates to or explains the summary sentence. We design a new iterative refinement algorithm: it induces the trees through repeatedly refining the structures predicted by previous iterations. We demonstrate experimentally on two benchmark datasets that our summarizer[1] performs competitively against state-of-the-art methods.

1 Introduction

Single-document summarization is the task of automatically generating a shorter version of a document while retaining its most important information. The task has received much attention in the natural language processing community due to its potential for various information access applications. Examples include tools which digest textual content (e.g., news, social media, reviews), answer questions, or provide recommendations.

Of the many summarization paradigms that have been identified over the years (see Mani 2001 and Nenkova and McKeown 2011 for comprehensive overviews), two have consistently attracted attention. In *abstractive* summarization, various text rewriting operations generate summaries using words or phrases that were not in the original text, while *extractive* approaches form summaries by copying and concatenating the most important spans (usually sentences) in a document. Recent approaches to (single-document) extractive summarization frame the task as a sequence labeling problem taking advantage of the success of neural network architectures (Bahdanau et al., 2015). The idea is to predict a label for each sentence specifying whether it should be included in the summary. Existing systems mostly rely on recurrent neural networks (Hochreiter and Schmidhuber, 1997) to model the document and obtain a vector representation for each sentence (Nallapati et al., 2017; Cheng and Lapata, 2016). Inter-sentential relations are captured in a sequential manner, without taking the structure of the document into account, although the latter has been shown to correlate with what readers perceive as important in a text (Marcu, 1999). Another problem in neural-based extractive models is the lack of interpretability. While capable of identifying summary sentences, these models are not able to rationalize their predictions (e.g., a sentence is in the summary because it describes important content upon which other related sentences elaborate).

The summarization literature offers examples of models which exploit the structure of the underlying document, inspired by existing theories of discourse such as Rhetorical Structure Theory (RST; Mann and Thompson 1988). Most approaches produce summaries based on tree-like document representations obtained by a parser trained on discourse annotated corpora (Carlson et al., 2003; Prasad et al., 2008). For instance, Marcu (1999) argues that a good summary can be generated by traversing the RST discourse tree structure top-down, following nucleus nodes (discourse units in RST are characterized regarding their text importance; nuclei denote central units, whereas satellites denote peripheral ones). Other work (Hirao et al., 2013; Yoshida et al., 2014) extends this idea by transforming RST trees into dependency trees and generating summaries by tree trimming. Gerani et al. (2014) summarize product reviews; their system aggregates RST trees rep-

[1] Our code is publicly available at https://github.com/nlpyang/SUMO.

Proceedings of NAACL-HLT 2019, pages 1745–1755
Minneapolis, Minnesota, June 2 - June 7, 2019.

1. One wily coyote traveled a bit too far from home, and its resulting adventure through Harlem had alarmed residents doing a double take and scampering to get out of its way Wednesday morning.
2. Police say frightened New Yorkers reported the coyote sighting around 9:30 a.m., and an emergency service unit was dispatched to find the animal.
3. The little troublemaker was caught and tranquilized in Trinity Cemetery on 155th street and Broadway, and then taken to the Wildlife Conservation Society at the Bronx Zoo, authorities said.
4. "The coyote is under evaluation and observation," said Mary Dixon, spokesperson for the Wildlife Conservation Society.
5. She said the Department of Environmental Conservation will either send the animal to a rescue center or put it back in the wild.
6. According to Adrian Benepe, New York City Parks Commissioner, coyotes in Manhattan are rare, but not unheard of.
7. "This is actually the third coyote that has been seen in the last 10 years," Benepe said.
8. Benepe said there is a theory the coyotes make their way to the city from suburban Westchester.
9. He said they probably walk down the Amtrak rail corridor along the Hudson River or swim down the Hudson River until they get to the city.

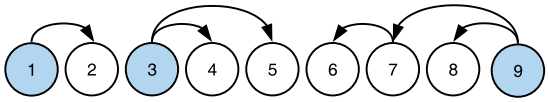

Figure 1: Dependency discourse tree for a document from the CNN/DailyMail dataset (Hermann et al., 2015). Blue nodes indicate the roots of the tree (i.e., summary sentences) and parent-child links indicate dependency relations.

resenting individual reviews into a graph, from which an abstractive summary is generated. Despite the intuitive appeal of discourse structure for the summarization task, the reliance on a parser which is both expensive to obtain (since it must be trained on labeled data) and error prone, presents a major obstacle to its widespread use.

Recognizing the merits of structure-aware representations for various NLP tasks, recent efforts have focused on learning latent structures (e.g., parse trees) while optimizing a neural network model for a down-stream task. Various methods impose structural constraints on the basic attention mechanism (Kim et al., 2017; Liu and Lapata, 2018), formulate structure learning as a reinforcement learning problem (Yogatama et al., 2017; Williams et al., 2018), or sparsify the set of possible structures (Niculae et al., 2018). Although latent structures are mostly induced for individual sentences, Liu and Lapata (2018) induce dependency-like structures for entire documents.

Drawing inspiration from this work and existing discourse-informed summarization models (Marcu, 1999; Hirao et al., 2013), we frame extractive summarization as a tree induction problem. Our model represents documents as multi-root dependency trees where each root node is a summary sentence, and the subtrees attached to it are sentences whose content is related to and covered by the summary sentence. An example of a document and its corresponding tree is shown in Figure 1; tree nodes correspond to document sentences; blue nodes represent those which should be in the summary, dependent nodes relate to or are subsumed by the parent summary sentence.

We propose a new framework that uses structured attention (Kim et al., 2017) as both the objective and attention weights for extractive summarization. Our model is trained end-to-end, it induces document-level dependency trees while predicting the output summary, and brings more interpretability in the summarization process by helping explain how document content contributes to the model's decisions. We design a new iterative structure refinement algorithm, which learns to induce document-level structures through repeatedly refining the trees predicted by previous iterations and allows the model to infer complex trees which go beyond simple parent-child relations (Liu and Lapata, 2018; Kim et al., 2017). The idea of structure refinement is conceptually related to recently proposed models for solving iterative inference problems (Marino et al., 2018; Putzky and Welling, 2017; Lee et al., 2018). It is also related to structured prediction energy networks (Belanger et al., 2017) which approach structured prediction as iterative miminization of an energy function. However, we are not aware of any previous work considering structure refinement for tree induction problems.

Our contributions in this work are three-fold: a novel conceptualization of extractive summarization as a tree induction problem; a model which capitalizes on the notion of structured attention to learn document representations based on iterative structure refinement; and large-scale evaluation studies (both automatic and human-based) which demonstrate that our approach performs competitively against state-of-the-art methods while being able to rationalize model predictions.

2 Model Description

Let d denote a document containing several sentences $[sent_1, sent_2, \cdots, sent_m]$, where $sent_i$ is the i-th sentence in the document. Extractive summarization can be defined as the task of assigning a label $y_i \in \{0, 1\}$ to each $sent_i$, indicating whether the sentence should be included in the summary. It is assumed that summary sentences represent the most important content of the document.

2.1 Baseline Model

Most extractive models frame summarization as a classification problem. Recent approaches (Zhang et al., 2018; Dong et al., 2018; Nallapati et al., 2017; Cheng and Lapata, 2016) incorporate a neural network-based encoder to build representations for sentences and apply a binary classifier over these representations to predict whether the sentences should be included in the summary. Given predicted scores r and gold labels y, the loss function can be defined as:

$$L = -\sum_{i=1}^{m}(\boldsymbol{y}_i \ln(\boldsymbol{r}_i) + (1-\boldsymbol{y}_i)\ln(1-\boldsymbol{r}_i)) \quad (1)$$

The encoder in extractive summarization models is usually a recurrent neural network with Long-Short Term Memory (LSTM; Hochreiter and Schmidhuber 1997) or Gated Recurrent Units (GRU; Cho et al. 2014). In this paper, our baseline encoder builds on the Transformer architecture (Vaswani et al., 2017), a recently proposed highly efficient model which has achieved state-of-the-art performance in machine translation (Vaswani et al., 2017) and question answering (Yu et al., 2018). The Transformer aims at reducing the fundamental constraint of sequential computation which underlies most architectures based on RNNs. It eliminates recurrence in favor of applying a self-attention mechanism which directly models relationships between all words in a sentence.

More formally, given a sequence of input vectors $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n\}$, the Transformer is composed of a stack of N identical layers, each of which has two sub-layers:

$$\tilde{\boldsymbol{h}}^l = \mathrm{LayerNorm}(\boldsymbol{h}^{l-1} + \mathrm{MHAtt}(\boldsymbol{h}^{l-1})) \quad (2)$$

$$\boldsymbol{h}^l = \mathrm{LayerNorm}(\tilde{\boldsymbol{h}}^l + \mathrm{FFN}(\tilde{\boldsymbol{h}}^l)) \quad (3)$$

where $\boldsymbol{h}^0 = \mathrm{PosEmb}(\boldsymbol{x})$ and PosEmb is the function of adding positional embeddings to the input; the superscript l indicates layer depth; LayerNorm is the layer normalization operation proposed in Ba et al. (2016); MHAtt represents the multi-head attention mechanism introduced in Vaswani et al. (2017) which allows the model to jointly attend to information from different representation subspaces (at different positions); and FFN is a two-layer feed-forward network with ReLU as hidden activation function.

For our extractive summarization task, the baseline system is composed of a sentence-level Transformer ($\mathcal{T}_\mathrm{S}$) and a document-level Transformer ($\mathcal{T}_\mathrm{D}$), which have the same structure. For each sentence $s_i = [\boldsymbol{w}_{i1}, \boldsymbol{w}_{i2}, \cdots, \boldsymbol{w}_{in}]$ in the input document, $\mathcal{T}_\mathrm{S}$ is applied to obtain a contextual representation for each word:

$$[\boldsymbol{u}_{i1}, \boldsymbol{u}_{i2}, \cdots, \boldsymbol{u}_{in}] = \mathcal{T}_\mathrm{S}([\boldsymbol{w}_{i1}, \boldsymbol{w}_{i2}, \cdots, \boldsymbol{w}_{in}]) \quad (4)$$

And the representation of a sentence is acquired by applying weighted-pooling:

$$\boldsymbol{a}_{ij} = \boldsymbol{W}_0 \boldsymbol{u}_{ij}^T \quad (5)$$

$$\boldsymbol{s}_i = \frac{1}{n}\sum_{j=1}^{n} \boldsymbol{a}_{ij}\boldsymbol{u}_{ij} \quad (6)$$

Document-level transformer $\mathcal{T}_\mathrm{D}$ takes $\boldsymbol{s}_i$ as input and yields a contextual representation for each sentence:

$$[\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_m] = \mathcal{T}_\mathrm{D}([\boldsymbol{s}_1, \boldsymbol{s}_2, \cdots, \boldsymbol{s}_m]) \quad (7)$$

Following previous work (Nallapati et al., 2017), we use a sigmoid function after a linear transformation to calculate the probability $\boldsymbol{r}_i$ of selecting s_i as a summary sentence:

$$\boldsymbol{r}_i = \mathrm{sigmoid}(\boldsymbol{W}_1 \boldsymbol{v}_i^T) \quad (8)$$

2.2 Structured Summarization Model

In the Transformer model sketched above, inter-sentence relations are modeled by multi-head attention based on softmax functions, which only capture shallow structural information. Our summarizer, which we call SUMO as a shorthand for **S**tructured **Sum**marization **Mo**del classifies sentences as summary-worthy or not, and simultaneously induces the structure of the source document as a multi-root tree. An overview of SUMO is illustrated in Figure 2. The model has the same sentence-level encoder $\mathcal{T}_\mathrm{S}$ as the baseline Transformer model (see the bottom box in Figure 2), but differs in two important ways: (a) it uses structured attention to model the roots (i.e., summary sentences) of the underlying tree (see the upper box in Figure 2); and (b) through iterative refinement it is able to progressively infer more complex structures from past guesses (see the second and third block in Figure 2).

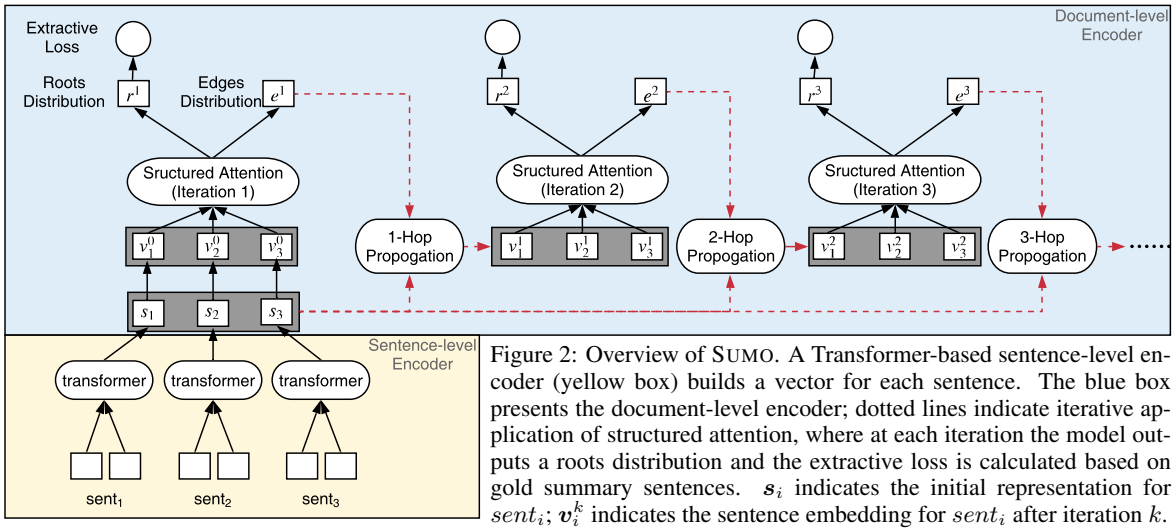

Figure 2: Overview of SUMO. A Transformer-based sentence-level encoder (yellow box) builds a vector for each sentence. The blue box presents the document-level encoder; dotted lines indicate iterative application of structured attention, where at each iteration the model outputs a roots distribution and the extractive loss is calculated based on gold summary sentences. $\boldsymbol{s}_i$ indicates the initial representation for $sent_i$; $\boldsymbol{v}_i^k$ indicates the sentence embedding for $sent_i$ after iteration k.

Structured Attention Assuming document sentences have been already encoded, SUMO first calculates the unnormalized root score $\tilde{\boldsymbol{r}}_i$ for $sent_i$ to indicate the extent to which it might be selected as root in the document tree. It also calculates the unnormalized edge score $\tilde{\boldsymbol{e}}_{ij}$ for sentence pair $\langle sent_i, sent_j \rangle$ indicating the extent to which $sent_i$ might be the head of $sent_j$ in that tree (first upper block in Figure 2). To inject structural bias, SUMO normalizes these scores as the marginal probabilities of forming edges in the document dependency tree.

We use the Tree-Matrix-Theorem (TMT; Koo et al. 2007; Tutte 1984) to calculate root marginal probability $\boldsymbol{r}_i$ and edge marginal probability $\boldsymbol{e}_{ij}$, following the procedure introduced in Liu and Lapata (2017). As illustrated in Algorithm 1, we first build the Laplacian matrix $\bar{\boldsymbol{L}}$ based on unnormalized scores and calculate marginal probabilities by matrix inverse-based operations ($\bar{\boldsymbol{L}}^{-1}$). We refer the interested reader to Koo et al. (2007) and Liu and Lapata (2017) for more details. In contrast to Liu and Lapata (2017), who compute the marginal probabilities of a single-root tree, our tree has multiple roots since in our task the summary typically contains multiple sentences. Given sentence vector $\boldsymbol{s}_i$ as input, SUMO computes:

$$\tilde{\boldsymbol{r}}_i = \boldsymbol{W}_r \boldsymbol{s}_i \tag{9}$$

$$\tilde{\boldsymbol{e}}_{ij} = \boldsymbol{s}_i \boldsymbol{W}_e \boldsymbol{s}_j^T \tag{10}$$

$$\boldsymbol{r}_i, \boldsymbol{e}_{ij} = \mathrm{TMT}(\tilde{\boldsymbol{r}}_i, \tilde{\boldsymbol{e}}_{ij}) \tag{11}$$

Iterative Structure Refinement SUMO essentially reduces summarization to a rooted-tree parsing problem. However, accurately predicting a tree in one shot is problematic. Firstly, when predicting the dependency tree, the model has solely access to labels for the roots (aka summary sentences), while tree edges are latent and learned without an explicit training signal. And as previous work (Liu and Lapata, 2017) has shown, a single application of TMT leads to shallow tree structures. Secondly, the calculation of $\tilde{\boldsymbol{r}}_i$ and $\tilde{\boldsymbol{e}}_{ij}$ would be based on first-order features alone, however, higher-order information pertaining to siblings and grandchildren has proved useful in discourse parsing (Carreras, 2007).

We address these issues with an inference algorithm which iteratively infers latent trees. In contrast to multi-layer neural network architectures like the Transformer or Recursive Neural Networks (Tai et al., 2015) where word representations are updated at every layer based on the output of previous layers, we refine only the tree structure during each iteration, word representations are not passed across multiple layers. Empirically, at early iterations, the model learns shallow and

Algorithm 1: Calculate Tree Marginal Probabilities based on Tree-Matrix-Theorem

Function TMT $(\tilde{\boldsymbol{r}}_i, \tilde{\boldsymbol{e}}_{ij})$ **l:**

$$\boldsymbol{A}_{ij} = \begin{cases} 0 & \text{if } i = j \\ exp(\hat{\boldsymbol{r}}_{ij}) & \text{otherwise} \end{cases}$$

$$\boldsymbol{L}_{ij} = \begin{cases} \sum_{i'=1}^{n} \boldsymbol{A}_{i'j} & \text{if } i = j \\ -\boldsymbol{A}_{ij} & \text{otherwise} \end{cases}$$

$$\bar{\boldsymbol{L}}_{ij} = \begin{cases} \boldsymbol{L}_{ij} + exp(\hat{\boldsymbol{r}}_i) & i = j \\ \boldsymbol{L}_{ij} & \text{otherwise} \end{cases}$$

$$\boldsymbol{e}_{ij} = (1 - \delta_{1,j}) \boldsymbol{A}_{ij} [\bar{\boldsymbol{L}}^{-1}]_{jj} - (1 - \delta_{i,1}) \boldsymbol{A}_{ij} [\bar{\boldsymbol{L}}^{-1}]_{ji}$$

$$\boldsymbol{r}_i = exp(\hat{\boldsymbol{r}}_i) [\bar{\boldsymbol{L}}^{-1}]_{i1}$$

return $\boldsymbol{r}_i, \boldsymbol{e}_{ij}$

simple trees, and information propagates mostly between neighboring nodes; as the structure gets more refined, information propagates more globally allowing the model to learn higher-order features.

Algorithm 2 provides the details of our refinement procedure. SUMO takes K iterations to learn the structure of a document. For each sentence, we initialize a structural vector $\boldsymbol{v}_i^0$ with sentence vector $\boldsymbol{s}_i$. At iteration k, we use sentence embeddings from the previous iteration $\boldsymbol{v}^{k-1}$ to calculate unnormalized root $\tilde{\boldsymbol{r}}_i^k$ and edge $\tilde{\boldsymbol{e}}_{ij}^k$ scores using a linear transformation with weight $\boldsymbol{W}_r^k$ and a bilinear transformation with weight $\boldsymbol{W}_e^k$, respectively. Marginal root and edge probabilities are subsequently normalized with the TMT to obtain $\boldsymbol{r}_i^k$ and $\boldsymbol{e}_{ij}^k$ (see lines 4–6 in Algorithm 2). Then, sentence embeddings are updated with *k-Hop Propagation*. The latter takes as input the initial sentence representations $\boldsymbol{s}$ rather than sentence embeddings $\boldsymbol{v}^{k-1}$ from the previous layer. In other words, new embeddings $\boldsymbol{v}^k$ are computed from scratch relying on the structure from the previous layer. Within the *k-Hop-Propagation* function (lines 12–19), edge probabilities $\boldsymbol{e}_{ij}^k$ are used as attention weights to propagate information from a sentence to all other sentences in k hops. $\boldsymbol{p}_i^l$ and $\boldsymbol{c}_i^l$ represent parent and child vectors, respectively, while vector $\boldsymbol{z}_i^l$ is updated with contextual information at hop l. At the final iteration (lines 9 and 10), the top sentence embeddings $\boldsymbol{v}^{K-1}$ are used to calculate the final root probabilities r^K.

We define the model's loss function as the summation of the losses of all iterations:

$$L = \sum_{k=1}^{K} [y \log(r_k) + (1-y) \log(1-r_k)] \quad (12)$$

SUMO uses the root probabilities of the top layer as the scores for summary sentences.

The *k-Hop-Propagation* function resembles the computation used in Graph Convolution Networks (Kipf and Welling, 2017; Marcheggiani and Titov, 2017). GCNs have been been recently applied to latent trees (Corro and Titov, 2019), however not in combination with iterative refinement.

3 Experiments

In this section we present our experimental setup, describe the summarization datasets we used, discuss implementation details, our evaluation protocol, and analyze our results.

Algorithm 2: Structured Summarization Model

Input: Document d
Output: Root probabilities $\boldsymbol{r}^K$ after K iterations

1 Calculate sentence vectors $\boldsymbol{s}$ using sentence-level Transformer T_S
2 $\boldsymbol{v}^0 \leftarrow \boldsymbol{s}$
3 **for** $k \leftarrow 1$ *to* $K-1$ **do**
4 Calculate unnormalized root scores: $\tilde{\boldsymbol{r}}_i^k = \boldsymbol{W}_r^k \boldsymbol{v}_i^{k-1}$
5 Calculate unnormalized edge scores: $\tilde{\boldsymbol{e}}_{ij}^k = \boldsymbol{v}_i^{k-1} \boldsymbol{W}_e^k \boldsymbol{v}_j^{k-1^T}$
6 Calculate marginal probabilities: $\boldsymbol{r}^k, \boldsymbol{e}^k = \text{TMT}(\tilde{\boldsymbol{r}}^k, \tilde{\boldsymbol{e}}^k)$
7 Update sentence representations: $\boldsymbol{v}^k = \text{k-Hop-Propagation}(\boldsymbol{e}^k, \boldsymbol{s}, k)$
8 **end**
9 Calculate final unnormalized root and edge scores: $\tilde{\boldsymbol{r}}_i^K = \boldsymbol{W}_r^K \boldsymbol{v}_i^{K-1}$, $\tilde{\boldsymbol{e}}_{ij}^K = \boldsymbol{v}_i^{K-1} \boldsymbol{W}_e^K \boldsymbol{v}_j^{K-1^T}$
10 Calculate final root and edge probabilities: $\boldsymbol{r}^K, \boldsymbol{e}^K = \text{TMT}(\tilde{\boldsymbol{r}}^K, \tilde{\boldsymbol{e}}^K)$
11
12 **Function** `k-Hop-Propagation(`$\boldsymbol{e}$, $\boldsymbol{s}$, k`)`**:**
13 $\boldsymbol{z}^0 \leftarrow \boldsymbol{s}$
14 **for** $l \leftarrow 1$ *to* k **do**
15 $\boldsymbol{p}_i^l = \frac{1}{n} \sum_{j=1}^{n} \boldsymbol{e}_{ji} \boldsymbol{z}_i^{l-1}$
16 $\boldsymbol{c}_i^l = \frac{1}{n} \sum_{j=1}^{n} \boldsymbol{e}_{ij} \boldsymbol{z}_i^{l-1}$
17 $\boldsymbol{z}_i^l = \tanh(\boldsymbol{W}_v^k [\boldsymbol{p}_i^{l-1}, \boldsymbol{c}_i^{l-1}, \boldsymbol{z}_i^{l-1}])$
18 **end**
19 **return** $\boldsymbol{z}^k$

3.1 Summarization Datasets

We evaluated SUMO on two benchmark datasets, namely the CNN/DailyMail news highlights dataset (Hermann et al., 2015) and the New York Times Annotated Corpus (NYT; Sandhaus 2008). The CNN/DailyMail dataset contains news articles and associated highlights, i.e., a few bullet points giving a brief overview of the article. We used the standard splits of Hermann et al. (2015) for training, validation, and testing (90,266/1,220/1,093 CNN documents and 196,961/12,148/10,397 DailyMail documents). We did not anonymize entities.

The NYT dataset contains 110,540 articles with abstractive summaries. Following Durrett et al. (2016), we split these into 100,834 training and 9,706 test examples, based on date of publication (test is all articles published on January 1, 2007 or later). We also followed their filtering procedure, documents with summaries that are shorter than 50 words were removed from the raw dataset. The

Model	CNN			DM			CNN+DM			NYT		
	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
LEAD-3	29.2	11.2	26.0	40.7	18.3	37.2	39.6	17.7	36.2	35.5	17.3	32.0
Narayan et al. (2018)	**30.4**	11.7	**26.9**	41.0	18.8	37.7	40.0	18.2	36.6	41.3	22.0	37.8
Marcu (1999)	25.6	6.10	19.5	31.9	12.4	23.5	26.5	9.80	20.4	29.6	11.2	23.0
Durrett et al. (2016)	—	—	—	—	—	—	—	—	—	40.8	22.3	36.7
See et al. (2017)	—	—	—	—	—	—	39.5	17.3	36.4	**42.7**	22.1	38.0
Celikyilmaz et al. (2018)	—	—	—	—	—	—	**41.7**	19.5	**37.9**	—	—	—
Transformer (no doc-att)	29.2	11.1	25.6	40.5	18.1	36.8	39.7	17.0	35.9	41.1	21.5	37.0
Transformer (1-layer doc-att)	29.5	11.4	26.0	41.5	18.7	38.0	40.6	18.1	36.7	41.8	22.1	37.8
Transformer (3-layer doc-att)	29.6	11.8	26.3	41.7	18.8	38.0	40.6	18.1	36.9	42.0	22.3	38.2
SUMO (1-layer)	29.5	11.6	26.2	41.6	18.8	37.6	40.5	18.0	36.8	42.2	22.1	38.1
SUMO (3-layer)	29.7	**12.0**	26.5	**42.0**	**19.1**	**38.0**	41.0	18.4	37.2	42.3	**22.7**	**38.6**

Table 1: Test set results on the CNN/DailyMail and NYT datasets using ROUGE F_1 (R-1 and R-2 are shorthands for unigram and bigram overlap, R-L is the longest common subsequence.

filtered test set includes 3,452 test examples out of the original 9,706. Compared to CNN/DailyMail, the NYT dataset contains longer and more elaborate summary sentences.

Both datasets contain abstractive gold summaries, which are not readily suited to training extractive summarization models. A greedy algorithm similar to Nallapati et al. (2017) was used to generate an oracle summary for each document. The algorithm explores different combinations of sentences and generates an oracle consisting of multiple sentences which maximize the ROUGE score with the gold summary. We assigned label 1 to sentences selected in the oracle summary and 0 otherwise and trained SUMO on this data.

3.2 Implementation Details

We followed the same training procedure for SUMO and various Transformer-based baselines. The vocabulary size was set to 30K. We used 300D word embeddings which were initialized randomly from $\mathcal{N}(0, 0.01)$. The sentence-level Transformer has 6 layers and the hidden size of FFN was set to 512. The number of heads in MHAtt was set to 4. Adam was used for training ($\beta_1 = 0.9$, $\beta_2 = 0.999$). We adopted the learning rate schedule from Vaswani et al. (2017) with warming-up on the first 8,000 steps. SUMO and related Transformer models produced 3-sentence summaries for each document at test time (for both CNN/DailyMail and NYT datasets).

3.3 Automatic Evaluation

We evaluated summarization quality using ROUGE F_1 (Lin, 2004). We report unigram and bigram overlap (ROUGE-1 and ROUGE-2) as a means of assessing informativeness and the longest common subsequence (ROUGE-L) as a means of assessing fluency.

Table 1 summarizes our results. We evaluated two variants of SUMO, with one and three structured-attention layers. We compared against a baseline which simply selects the first three sentences in each document (LEAD-3) and several incarnations of the basic Transformer model introduced in Section 2.1. These include a Transformer without document-level self-attention and two variants with document-level self attention instantiated with one and three layers. Several state-of-the-art models are also included in Table 1, both extractive and abstractive.

REFRESH (Narayan et al., 2018) is an extractive summarization system trained by globally optimizing the ROUGE metric with reinforcement learning. The system of Marcu (1999) is another extractive summarizer based on RST parsing. It uses discourse structures and RST's notion of nuclearity to score document sentences in terms of their importance and selects the most important ones as the summary. Our re-implementation of Marcu (1999) used the parser of Zhao and Huang (2017) to obtain RST trees. Durrett et al. (2016) develop a summarization system which integrates a compression model that enforces grammaticality and coherence. See et al. (2017) present an abstractive summarization system based on

an encoder-decoder architecture. Celikyilmaz et al.'s (2018) system is state-of-the-art in abstractive summarization using multiple agents to represent the document as well a hierarchical attention mechanism over the agents for decoding.

As far as SUMO is concerned, we observe that it outperforms a simple Transformer model without any document attention as well as variants with document attention. SUMO with three layers of structured attention overall performs best, confirming our hypothesis that document-level structure is beneficial for summarization. The results in Table 1 also reveal that SUMO and all Transformer-based models with document attention (doc-att) outperform LEAD-3 across metrics. SUMO (3-layer) is competitive or better than state-of-the-art approaches. Examples of system output are shown in Table 4.

Finally, we should point out that SUMO is superior to Marcu (1999) even though the latter employs linguistically informed document representations.

3.4 Human Evaluation

In addition to automatic evaluation, we also assessed system performance by eliciting human judgments. Our first evaluation quantified the degree to which summarization models retain key information from the document following a question-answering (QA) paradigm (Clarke and Lapata, 2010; Narayan et al., 2018). We created a set of questions based on the gold summary under the assumption that it highlights the most important document content. We then examined whether participants were able to answer these questions by reading system summaries alone without access to the article. The more questions a system can answer, the better it is at summarizing the document as a whole.

We randomly selected 20 documents from the CNN/DailyMail and NYT datasets, respectively and wrote multiple question-answer pairs for each gold summary. We created 71 questions in total varying from two to six questions per gold summary. We asked participants to read the summary and answer all associated questions as best they could without access to the original document or the gold summary. Examples of questions and their answers are given in Table 4. We adopted the same scoring mechanism used in Clarke and Lapata (2010), i.e., a correct answer was marked with a score of one, partially correct answers with a score of 0.5, and zero otherwise. Answers were elicited using Amazon's Mechanical Turk platform. Participants evaluated summaries produced by the LEAD-3 baseline, our 3-layered SUMO model and multiple state-of-the-art systems. We elicited 5 responses per summary.

Model	CNN+DM		NYT	
	Rank	QA	Rank	QA
LEAD	0.07	40.1	-0.18	36.3
Narayan et al. (2018)	**0.21**	62.4	0.12	46.1
Durrett et al. (2016)	—	—	-0.11	40.1
See et al. (2017)	-0.23	36.6	-0.44	35.3
Celikyilmaz et al. (2018)	-0.64	37.5	—	—
SUMO (3-layer)	0.15	**65.3**	**0.33**	**57.2**
GOLD	0.11	—	-0.16	—
ORACLE	0.37	74.6	0.41	67.1

Table 2: System ranking according to human judgments on summary quality and QA-based evaluation.

Table 2 (QA column) presents the results of the QA-based evaluation. Based on the summaries generated by SUMO, participants can answer 65.3% of questions correctly on CNN/DailyMail and 57.2% on NYT. Summaries produced by LEAD-3 and comparison systems fare worse, with REFRESH (Narayan et al., 2018) coming close to SUMO on CNN/DailyMail but not on NYT. Overall, we observe there is room for improvement since no system comes close to the extractive oracle, indicating that improved sentence selection would bring further performance gains to extractive approaches. Between-systems differences are all statistically significant (using a one-way ANOVA with posthoc Tukey HSD tests; $p < 0.01$) with the exception of LEAD-3 and See et al. (2017) in both CNN+DM and NTY, Narayan et al. (2018) and SUMO in both CNN+DM and NTY, and LEAD-3 and Durrett et al. (2016) in NYT.

Our second evaluation study assessed the overall quality of the summaries by asking participants to rank them taking into account the following criteria: *Informativeness* , *Fluency*, and *Succinctness*. The study was conducted on the Amazon Mechanical Turk platform using Best-Worst Scaling (Louviere et al., 2015), a less labor-intensive alternative to paired comparisons that has been shown to produce more reliable results than rating scales (Kiritchenko and Mohammad, 2017). Participants were presented with a document and

	CNN+DM			NYT		
	P	H	EA	P	H	EA
Parser	24.8	8.9	—	18.7	10.6	—
SUMO (1-layer)	69.0	2.9	23.1	54.7	3.6	20.6
SUMO (3-layer)	52.7	3.7	25.3	45.1	6.2	21.6
Left Branching	—	—	21.4	—	—	21.3
Right Branching	—	—	7.3	—	—	6.7

Table 3: Descriptive statistics Projectivity(%), Height and EdgeAgreement(%) for dependency trees produced by our model and the RST discourse parser of Zhao and Huang (2017). Results are shown on the CNN/DailyMail and NYT test sets.

summaries generated from 3 out of 7 systems and were asked to decide which summary was better and which one was worse, taking into account the criteria mentioned above. We used the same 20 documents from each dataset as in our QA evaluation and elicited 5 responses per comparison.

The rating of each system was computed as the percentage of times it was chosen as best minus the times it was selected as worst. Ratings range from -1 (worst) to 1 (best). As shown in Table 2 (Rank column), participants overwhelming prefer the extractive oracle summaries followed by SUMO and REFRESH (Narayan et al., 2018). Abstractive systems (Celikyilmaz et al., 2018; See et al., 2017; Durrett et al., 2016) perform relatively poorly in this evaluation; we suspect that humans are less forgiving to fluency errors and slightly incoherent summaries. Interestingly, gold summaries fare worse than the oracle and extractive systems. Albeit fluent, gold summaries naturally contain less detail compared to oracle-based ones; on virtue of being abstracts, they are written in a telegraphic style, often in conversational language while participants prefer the more lucid style of the extracts. All pairwise comparisons among systems are statistically significant (using a one-way ANOVA with post-hoc Tukey HSD tests; $p < 0.01$) except LEAD-3 and See et al. (2017) in both CNN+DM and NTY, Narayan et al. (2018) and SUMO in both CNN+DM and NTY, and LEAD and Durrett et al. (2016) in NYT.

3.5 Evaluation of the Induced Structures

To gain further insight into the structures learned by SUMO, we inspected the trees it produces. Specifically, we used the Chu-Liu-Edmonds algorithm (Chu and Liu, 1965; Edmonds, 1967) to extract the maximum spanning tree from the attention scores. We report various statistics on the characteristics of the induced trees across datasets in Table 3. We also examine the trees learned from different SUMO variants (with different numbers of iterations) in order to establish whether the iterative process yields better structures.

Specifically, we compared the dependency trees obtained from our model to those produced by a discourse parser (Zhao and Huang, 2017) trained on a corpus which combines annotations from the RST treebank (Carlson et al., 2003) and the Penn Treebank (Marcus et al., 1993). Unlike traditional RST discourse parsers (Feng and Hirst, 2014), which first segment a document into Elementary Discourse Units (EDUs) and then build a discourse tree with the EDUs[2] as leaves, Zhao and Huang (2017) parse a document into an RST tree along with its syntax subtrees without segmenting it into EDUs. The outputs of their parser are ideally suited for comparison with our model, since we only care about document-level structures, and ignore the subtrees within sentence boundaries. We converted the constituency RST trees obtained from the discourse parser into dependency trees using Hirao et al.'s algorithm (2013).

As can be seen in Table 3, the dependency structures induced by SUMO are simpler compared to those obtained from the discourse parser. Our trees are generally shallower, almost half of them are projective. We also calculated the percentage of head-dependency edges that are identical between learned trees and parser generated ones. Although SUMO is not exposed to any annotated trees during training, a number of edges agree with the outputs of the discourse parser. Moreover, we observe that the iterative process involving multiple structured attention layers helps generate better discourse trees. We also compare SUMO trees against a left- and right-branching baseline, where the document is trivially parsed into a left- and right-branching tree forming a chain-like structure. As shown in Table 3, SUMO outperforms these baselines (with the exception of the one-layered model on NYT). We should also point out that the edge agreement between SUMO generated trees and left/right branching trees is low (around 30% on both datasets), indicating that the trees we learn are different from a simple chain.

[2]EDUs roughly correspond to clauses.

	CNN/DM	NYT
GOLD	A company called CyArk specializes in digital preservation of threatened ancient and historical architecture. Founded by an Iraqi-born engineer, it plans to preserve 500 World Heritage sites within five years.	Louisiana officials set July 31 deadline for applicants for the Road Home, grant program for homeowners who lost their houses to hurricanes Katrina and Rita. Program is expected to cost far more than $7.5 billion provided by Federal Government, in part because many more families have applied than officials anticipated. With cutoff date, State hopes to figure out how much more money it needs to pay for program. Shortfall is projected to be $2.9 billion.
QA	Which company specializes in digital preservation of threatened ancient and historical architecture? [CyArk] How many World Heritage sites does the company plan to preserve? [500]	What is Road Home? [the Louisiana grant program for homeowners who lost their houses to hurricanes Katrina and Rita] When is the applicants' deadline for the Road Home? [July 31] Why is the program expected to cost far more than $7.5 billion? [many more families have applied than officials anticipated] What is the shortfall projected to be? [$2.9 billion]
LEAD-3	In 2001, the Taliban wiped out 1700 years of history in a matter of seconds, by blowing up ancient Buddha statues in central Afghanistan with dynamite. They proceeded to do so after an attempt at bringing down the 175-foot tall sculptures with anti-aircraft artillery had failed. Sadly, the event was just the first in a series of atrocities that have robbed the world of some of its most prized cultural heritage.	The Road Home, the Louisiana grant program for homeowners who lost their houses to hurricanes Katrina and Rita, is expected to cost far more than the $7.5 billion provided by the Federal Government, in part because many more families have applied than officials had anticipated. As a result, Louisiana officials on Tuesday night set a July 31 deadline for applicants, who can receive up to $150,000 to repair or rebuild their houses. With the cutoff date, the State hopes to be able to figure out how much more money it needs to pay for the program.
See et al. (2017)	The Taliban wiped out 1700 years of history in a matter of seconds. The thought of losing a piece of our collective history is a bleak one. But if loss can't be avoided, technology can lend a hand.	Louisiana grant program for homeowners who lost their houses to hurricanes Katrina and Rita is expected to cost far more than $7.5 billion provided by federal government. Louisiana officials set July 31 deadline for applicants, who can receive up to $150,000 to repair or rebuild their houses.
Narayan et al. (2018)	Sadly, the event was just the first in a series of atrocities that have robbed the world of some of its most prized cultural heritage. But historical architecture is also under threat from calamities which might well escape our control, such as earthquakes and climate change. The thought of losing a piece of our collective history is a bleak one.	The Road Home, the Louisiana grant program for homeowners who lost their houses to hurricanes Katrina and Rita, is expected to cost far more than the $7.5 billion provided by the federal government, in part because many more families have applied than officials had anticipated. With the cutoff date, the State hopes to be able to figure out how much more money it needs to pay for the program. The shortfall is projected to be $2.9 billion.
SUMO	In 2001, the Taliban wiped out 1700 years of history in a matter of seconds, by blowing up ancient Buddha statues in central Afghanistan with dynamite. Sadly, the event was just the first in a series of atrocities that have robbed the world of some of its most prized cultural heritage. Now Cyark, a non-profit company founded by an Iraqi-born engineer, is using groundbreaking laser scanning to ensure that – at the very least – incredibly accurate digital versions of the world's treasures will stay with us forever.	The Road Home, the Louisiana grant program for homeowners who lost their houses to hurricanes Katrina and Rita, is expected to cost far more than the $7.5 billion provided by the federal government, in part because many more families have applied than officials had anticipated. As a result, Louisiana officials on Tuesday night set a July 31 deadline for applicants, who can receive up to $150,000 to repair or rebuild their houses. The shortfall is projected to be $2.9 billion.

Table 4: GOLD human authored summaries, questions based on them (answers shown in square brackets) and automatic summaries produced by the LEAD-3 baseline, the abstractive system of See et al. (2017), REFRESH (Narayan et al., 2018), and SUMO for a CNN and NYT (test) article.

4 Conclusions

In this paper we provide a new perspective on extractive summarization, conceptualizing it as a tree induction problem. We present SUMO, a Structured Summarization Model, which induces a multi-root dependency tree of a document, where roots are summary-worthy sentences, and subtrees attached to them are sentences which elaborate or explain the summary content. SUMO generates complex trees following an iterative refinement process which builds latent structures while using information learned in previous iterations. Experiments on two datasets, show that SUMO performs competitively against state-of-the-art methods and induces meaningful tree structures.

In the future, we would like to generalize SUMO to abstractive summarization (i.e., to learn latent structure for documents *and* sentences) and perform experiments in a weakly-supervised setting where summaries are not available but labels can be extrapolated from the article's title or topics.

Acknowledgments

We thank Serhii Havrylov for helpful suggestions. This research is supported by a Google PhD Fellowship to the first author. We gratefully acknowledge the support of the European Research Council (Lapata, award number 681760, "Translating Multiple Modalities into Text"; Titov award number 678254, "Broad Coverage Semantic Parsing").

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *In Proceedings of the 3rd International Conference on Learning Representations*, San Diego, California.

David Belanger, Bishan Yang, and Andrew McCallum. 2017. End-to-end learning for structured prediction energy networks. *ICML*.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2003. Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *Current and new directions in discourse and dialogue*, pages 85–112. Springer.

Xavier Carreras. 2007. Experiments with a higher-order projective dependency parser. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

Asli Celikyilmaz, Antoine Bosselut, Xiaodong He, and Yejin Choi. 2018. Deep communicating agents for abstractive summarization. *arXiv preprint arXiv:1803.10357*.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. *arXiv preprint arXiv:1603.07252*.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar.

Yoeng-Jin Chu and Tseng-Hong Liu. 1965. On shortest arborescence of a directed graph. *Scientia Sinica*, 14(10):1396.

James Clarke and Mirella Lapata. 2010. Discourse constraints for document compression. *Computational Linguistics*, 36(3):411–441.

Caio Corro and Ivan Titov. 2019. Differentiable perturb-and-parse: Semi-supervised parsing with a structured variational autoencoder. In *ICLR*.

Yue Dong, Yikang Shen, Eric Crawford, Herke van Hoof, and Jackie Chi Kit Cheung. 2018. Banditsum: Extractive summarization as a contextual bandit. In *Proceedings of the EMNLP Conference*.

Greg Durrett, Taylor Berg-Kirkpatrick, and Dan Klein. 2016. Learning-based single-document summarization with compression and anaphoricity constraints. *arXiv preprint arXiv:1603.08887*.

Jack Edmonds. 1967. Optimum branchings. *Journal of Research of the national Bureau of Standards B*, 71(4):233–240.

Vanessa Wei Feng and Graeme Hirst. 2014. A linear-time bottom-up discourse parser with constraints and post-editing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 511–521.

Shima Gerani, Yashar Mehdad, Giuseppe Carenini, Raymond T Ng, and Bita Nejat. 2014. Abstractive summarization of product reviews using discourse structure. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1602–1613.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701.

Tsutomu Hirao, Yasuhisa Yoshida, Masaaki Nishino, Norihito Yasuda, and Masaaki Nagata. 2013. Single-document summarization as a tree knapsack problem. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1515–1520.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Yoon Kim, Carl Denton, Luong Hoang, and Alexander M Rush. 2017. Structured attention networks.

Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. *ICLR*.

Svetlana Kiritchenko and Saif Mohammad. 2017. Best-worst scaling more reliable than rating scales: A case study on sentiment intensity annotation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 465–470, Vancouver, Canada.

Terry Koo, Amir Globerson, Xavier Carreras Pérez, and Michael Collins. 2007. Structured prediction models via the matrix-tree theorem. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 141–150.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. *EMNLP*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2017. Learning structured text representations. *arXiv preprint arXiv:1705.09207*.

Yang Liu and Mirella Lapata. 2018. Learning structured text representations. *Transactions of the Association for Computational Linguistics*, 6:63–75.

Jordan J Louviere, Terry N Flynn, and Anthony Alfred John Marley. 2015. *Best-worst scaling: Theory, methods and applications*. Cambridge University Press.

Inderjeet Mani. 2001. *Automatic Summarization*. John Benjamins Pub Co.

William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text-Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1507–1516, Copenhagen, Denmark. Association for Computational Linguistics.

Daniel Marcu. 1999. Discourse trees are good indicators of importance in text. *Advances in automatic text summarization*, 293:123–136.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19(2):313–330.

Joseph Marino, Yisong Yue, and Stephan Mandt. 2018. Iterative amortized inference. In *Proceedings of the 35th International Conference on Machine Learning*, pages 3403–3412, Stockholm, Sweden.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*, pages 3075–3081.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Ranking sentences for extractive summarization with reinforcement learning. *arXiv preprint arXiv:1802.08636*.

Ani Nenkova and Kathleen McKeown. 2011. Automatic summarization. *Foundations and Trends in Information Retrieval*, 5(2–3):103–233.

Vlad Niculae, André F. T. Martins, and Claire Cardie. 2018. Towards dynamic computation graphs via sparse latent structure. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 905–911.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind K Joshi, and Bonnie L Webber. 2008. The penn discourse treebank 2.0. In *LREC*, Marrakech, Morocco. Citeseer.

Patrick Putzky and Max Welling. 2017. Recurrent inference machines for solving inverse problems. *CoRR*, abs/1706.04008.

Evan Sandhaus. 2008. The new york times annotated corpus. Linguistic Data Consortium. Philadelphia.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 2015 Annual Meeting of the Association for Computational Linguistics*, pages 1556–1566, Beijing, China.

William Thomas Tutte. 1984. Graph theory.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010.

Adina Williams, Andrew Drozdov, and Samuel R. Bowman. 2018. Do latent tree learning models identify meaningful structure in sentences? *Transactions of the Association for Computational Linguistics*, 6:253–267.

Dani Yogatama, Phil Blunsom, Chris Dyer, Edward Grefenstette, and Wang Ling. 2017. Learning to compose words into sentences with reinforcement learning. In *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France.

Yasuhisa Yoshida, Jun Suzuki, Tsutomu Hirao, and Masaaki Nagata. 2014. Dependency-based discourse parser for single-document summarization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1834–1839.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. *arXiv preprint arXiv:1804.09541*.

Xingxing Zhang, Mirella Lapata, Furu Wei, and Ming Zhou. 2018. Neural latent extractive document summarization. In *Proceedings of the EMNLP Conference*.

Kai Zhao and Liang Huang. 2017. Joint syntacto-discourse parsing and the syntacto-discourse treebank. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2117–2123, Copenhagen, Denmark.