StructSum: Summarization via Structured Representations

Vidhisha Balachandran **Artidoro Pagnoni** **Jay Yoon Lee**
Dheeraj Rajagopal **Jaime Carbonell** **Yulia Tsvetkov**
School of Computer Science
Carnegie Mellon University, Pittsburgh, USA
{vbalacha,apagnoni,jaylee,dheeraj,jgc,ytsvetko}@cs.cmu.edu

Abstract

Abstractive text summarization aims at compressing the information of a long source document into a rephrased, condensed summary. Despite advances in modeling techniques, abstractive summarization models still suffer from several key challenges: (i) *layout bias*: they overfit to the style of training corpora; (ii) *limited abstractiveness*: they are optimized to copying n-grams from the source rather than generating novel abstractive summaries; (iii) *lack of transparency*: they are not interpretable. In this work, we propose a framework based on document-level structure induction for summarization to address these challenges. To this end, we propose incorporating *latent and explicit dependencies across sentences in the source document* into end-to-end single-document summarization models. Our framework complements standard encoder-decoder summarization models by augmenting them with rich structure-aware document representations based on implicitly learned (latent) structures and externally-derived linguistic (explicit) structures. We show that our summarization framework, trained on the CNN/DM dataset, improves the coverage of content in the source documents, generates more abstractive summaries by generating more novel n-grams, and incorporates interpretable sentence-level structures, while performing on par with standard baselines.[1]

1 Introduction

Text summarization aims at identifying important information in long source documents and expressing it in human readable summaries. Two prominent methods of generating summaries are *extractive* (Dorr et al., 2003; Nallapati et al., 2017), where important sentences in the source article are selected to form a summary, and *abstractive* (Rush et al., 2015; See et al., 2017), where the model restructures and rephrases essential content into a paraphrased summary.

State of the art approaches to abstractive summarization employ neural encoder-decoder methods that encode the source document as a sequence of tokens producing latent document representations and decode the summary conditioned on the representations. Recent studies suggest that these models suffer from several key challenges. First, since standard training datasets are derived from news articles, model outputs are strongly affected by the layout bias of the articles, with models relying on the leading sentences of source documents (Kryscinski et al., 2019; Kedzie et al., 2018). Second, although they aim to generate paraphrased summaries, abstractive summarization systems often copy long sequences from the source, causing their outputs to resemble extractive summaries (Lin and Ng, 2019; Gehrmann et al., 2018). Finally, current methods do not lend themselves easily to interpretation via intermediate structures (Lin and Ng, 2019), which could be useful for identifying major bottlenecks in summarization models.

To address these challenges, we introduce *StructSum*: a framework that incorporates structured document representations into summarization models. StructSum complements a standard encoder-decoder architecture with two novel components: (1) a *latent-structure attention* module that adapts structured representations (Kim et al., 2017; Liu and Lapata, 2017) for the summarization task, and (2) an *explicit-structure attention* module that incorporates an external linguistic structure (e.g., coreference links). The two complementary components are incorporated and learned jointly with the encoder and decoder, as shown in Figure 1.

Encoders with induced latent structures have been shown to benefit several tasks including document classification, natural language inference

[1]Code and data available at: https://github.com/vidhishanair/structured_summarizer

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2575–2585
April 19 - 23, 2021.

(Liu and Lapata, 2017; Cheng et al., 2016), and machine translation (Kim et al., 2017). Our latent structure attention module builds upon Liu and Lapata (2017) to model the dependencies between sentences in a document. It uses a variant of Kirchhoff's matrix-tree theorem (Tutte, 1984) to model such dependencies as non-projective tree structures(§2.2). The explicit attention module is linguistically-motivated and aims to incorporate inter-sentence links from externally annotated document structures. We incorporate a coreference based dependency graph across sentences, which is then combined with the output of the latent structure attention module to produce a hybrid structure-aware sentence representation (§2.3).

We test our framework using the CNN/DM dataset (Hermann et al., 2015) and show in §4.1 that it outperforms the base pointer-generator model (See et al., 2017) by up to 1.1 ROUGE-L. We find that the latent and explicit structures are complementary, both contributing to the final performance improvement. Our modules are also orthogonal to the choice of an underlying encoder-decoder architecture, rendering them flexible to be incorporated into other advanced models.

Quantitative and qualitative analyses of summaries generated by StructSum and baselines (§4), reveal that structure-aware summarization mitigates the news corpora layout bias by improving the coverage of source document sentences. Additionally, StructSum reduces the bias of copying large sequences from the source, inherently making the summaries more abstractive by generating ∼15% more novel n-grams than a competitive baseline. We also show examples of the learned interpretable sentence dependency structures, motivating further research for structure-aware modeling.

2 StructSum Framework

Consider a source document $\boldsymbol{x}$ consisting of n sentences $\{\boldsymbol{s}\}$ where each sentence $\boldsymbol{s}_i$ is composed of a sequence of words. Document summarization aims to map the source document to a target summary $\boldsymbol{y}$ of m words $\{y\}$. A typical neural abstractive summarization system is an attentional sequence-to-sequence model that encodes the input sequence $\boldsymbol{x}$ as a continuous sequence of tokens $\{w\}$ using a standard encoder (Hochreiter and Schmidhuber, 1997; Vaswani et al., 2017). The encoder produces a set of hidden representations $\{\mathbf{h}\}$. A decoder maps the previously generated token y_{t-1} to a hidden state and computes a soft attention probability distribution $p(\mathbf{a}_t \mid \boldsymbol{x}, \boldsymbol{y}_{1:t-1})$ over encoder hidden states. A distribution p over the vocabulary is computed at every time step t and the network is trained using the negative log likelihood loss: $\text{loss}_t = -\log p(y_t)$.

StructSum modifies the above architecture as follows. We aggregate the token representations from the encoder to form sentence representations as in hierarchical encoders (Yang et al., 2016). We then use implicit- and explicit-structure attention modules to augment the sentence representations with sentence dependency information, leveraging both a learned latent structure and an external structure from other NLP modules. The attended vectors are then passed to the decoder, which produces the output abstractive summary. In the rest of this section, we describe our framework architecture, shown in Figure 1, in detail.

2.1 Sentence Representations

We consider an encoder which takes a sequence of words in a sentence $\boldsymbol{s}_i = \{w\}$ as input and produces contextual hidden representation for each word $\mathbf{h}_{w_{ik}}$, where w_{ik} is the k^{th} word of the i^{th} sentence, $k = 1 : q$ and q is the number of words in the sentence $\boldsymbol{s}_i$. The word hidden representations are max-pooled at the sentence level and passed through a sentence-encoder, which produces new hidden sentence representations for each sentence $\mathbf{h}_{\boldsymbol{s}_i}$. The sentence hidden representations are then passed as inputs to the latent and explicit structure attention modules.

2.2 Latent Structure (LS) Attention

We model the latent structure of a source document as a non-projective dependency tree of sentences and force a pairwise attention module to automatically induce this tree. We denote the marginal probability of a dependency edge as $a_{ij} = p(z_{ij} = 1)$ where z_{ij} is the latent variable representing the edge from sentence i to sentence j. We parameterize the unnormalized pairwise scores between sentences with a neural network and use the Kirchoff's matrix tree theorem (Tutte, 1984) to compute the marginal probability of a dependency edge between any two sentences.

Specifically, we decompose the representation of a sentence $\boldsymbol{s}_i$ into a *semantic* vector $\mathbf{g}_{\boldsymbol{s}_i}$ and *structure* vector $\mathbf{d}_{\boldsymbol{s}_i}$ as $\mathbf{h}_{\boldsymbol{s}_i} = [\mathbf{g}_{\boldsymbol{s}_i}; \mathbf{d}_{\boldsymbol{s}_i}]$. Using the structure vectors $\mathbf{d}_{\boldsymbol{s}_i}, \mathbf{d}_{\boldsymbol{s}_j}$, we compute a score f_{ij} between sentence pairs (i, j) (where sentence i is

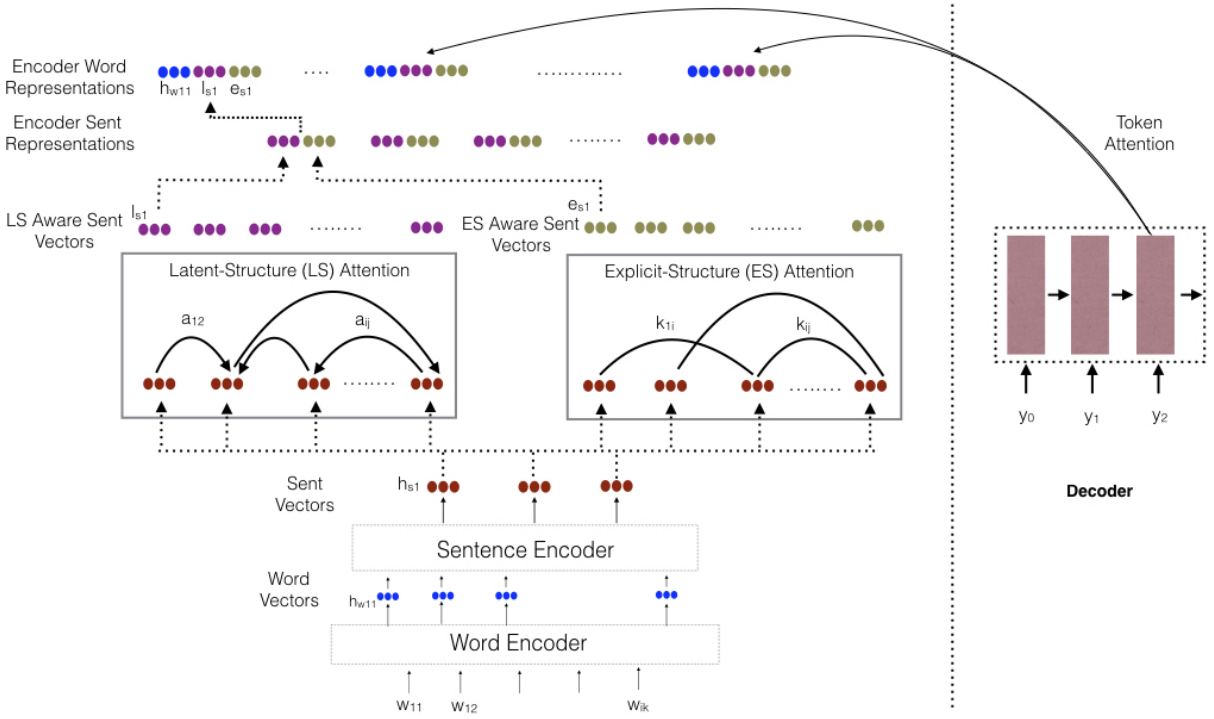

Figure 1: StructSum incorporates Latent Structure (LS) §2.2 and Explicit Structure (ES) §2.3 Attention to produce structure-aware representations. Here, StructSum augments the Pointer-Generator model, but the methodology that we proposed is general, and it can be applied to other encoder-decoder summarization systems

the parent node of sentence j) and a score r_i (where the sentence $\boldsymbol{s}_i$ is the root node):

$$f_{ij} = F_p(\mathbf{d}_{\boldsymbol{s}_i})^T W_a F_c(\mathbf{d}_{\boldsymbol{s}_j}) \text{ and } r_i = F_r(\mathbf{d}_{\boldsymbol{s}_i})$$

where F_p, F_c and F_r are linear-projection functions that build representations for the parent, child and root nodes respectively, and W_a is the weight for bilinear transformation. Here, f_{ij} is the edge weight between nodes (i, j) in a weighted adjacency graph $\mathbf{F}$ and is computed for all pairs of sentences. Using f_{ij} and r_i, we compute normalized attention scores a_{ij} and a^r_i using a variant of Kirchhoff's matrix-tree theorem where a_{ij} is the marginal probability of a dependency edge between sentences (i, j) and a^r_i is the probability of sentence i being the root.

Using these probabilistic attention weights and the semantic vectors $\{\mathbf{g}_{\boldsymbol{s}}\}$, we compute the attended sentence representations as:

$$\mathbf{p}_{\boldsymbol{s}_i} = \sum_{j=1}^{n} a_{ji}\mathbf{g}_{\boldsymbol{s}_j} + a^r_i \mathbf{g}_{root}$$

$$\mathbf{c}_{\boldsymbol{s}_i} = \sum_{j=1}^{n} a_{ij}\mathbf{g}_{\boldsymbol{s}_i}$$

$$\mathbf{l}_{\boldsymbol{s}_i} = \tanh(W_r[\mathbf{g}_{\boldsymbol{s}_i}, \mathbf{p}_{\boldsymbol{s}_i}, \mathbf{c}_{\boldsymbol{s}_i}])$$

where $\mathbf{p}_{\boldsymbol{s}_i}$ is the context vector gathered from possible parents of sentence i, $\mathbf{c}_{\boldsymbol{s}_i}$ is the context vector gathered from possible children, and $\mathbf{g}_{root}$ is a special embedding for the root node. Here, the updated sentence representation $l_{\boldsymbol{s}_i}$ incorporates the implicit structural information.

2.3 Explicit Structure (ES) Attention

Following Durrett et al. (2016), who showed that modeling coreference knowledge through anaphora constraints leads to improved clarity or grammaticality, we incorporate cross-sentence coreference links as the source of explicit structure. First, we use an off-the-shelf coreference parser[2] to identify coreferring mentions. We then build a coreference based sentence graph by adding a link between sentences $(\boldsymbol{s}_i, \boldsymbol{s}_j)$, if they have any coreferring mentions. This graph is converted into a weighted graph by incorporating a weight on the edge between two sentences that is proportional to the number of unique coreferring mentions between them. We normalize these edge weights for every sentence, effectively building a weighted adjacency matrix $\mathbf{K}$ where k_{ij} is given by:

[2] https://github.com/huggingface/neuralcoref/

$$k_{ij} = P(z_{ij} = 1) \quad (1)$$

$$= \frac{count(m_i \bigcap m_j) + \epsilon}{\sum_{v=1}^{n} count(m_i \bigcap m_v)} \quad (2)$$

where m_i denotes the set of unique mentions in sentence $\boldsymbol{s}_i$, $(m_i \bigcap m_j)$ denotes the set of co-referring mentions between the two sentences, and z is a latent variable representing a link in the coreference sentence graph. $\epsilon = 5e^{-4}$ is a smoothing hyperparameter.

Given contextual sentence representations $\{\mathbf{h}_{\boldsymbol{s}}\}$ and our explicit coreference-based weighted adjacency matrix $\mathbf{K}$, we learn an explicit structure-aware representation as follows:

$$\mathbf{u}_{\boldsymbol{s}_i} = \tanh(F_u(\mathbf{h}_{\boldsymbol{s}_i}))$$

$$\mathbf{t}_{\boldsymbol{s}_i} = \sum_{j=1}^{p} k_{ij} \mathbf{u}_{\boldsymbol{s}_j}$$

$$\mathbf{e}_{\boldsymbol{s}_i} = \tanh(F_e(\mathbf{t}_{\boldsymbol{s}_i}))$$

where F_u and F_e are linear projections and $\mathbf{e}_{\boldsymbol{s}_i}$ is an updated sentence representation which incorporates explicit structural information.

Finally, to combine the two structural representations, we concatenate the latent and explicit sentence vectors as: $\mathbf{h}_{\boldsymbol{s}_i} = [\mathbf{l}_{\boldsymbol{s}_i}; \mathbf{e}_{\boldsymbol{s}_i}]$ to form encoder sentence representations of the source document. To provide every token representation with the context of the entire document, the token representations are concatenated with their corresponding structure-aware sentence representation: $\mathbf{h}_{w_{ij}} = [\mathbf{h}_{w_{ij}}; \mathbf{h}_{\boldsymbol{s}_i}]$ where $\boldsymbol{s}_i$ is the sentence to which the word w_{ij} belongs. The resulting structure-aware token representations can be used to directly replace previous token representations as input to the decoder.

3 Experiments

Dataset: We evaluate our approach on the CNN/Daily Mail corpus[3] (Hermann et al., 2015; Nallapati et al., 2016) and use the same preprocessing steps as shown in See et al. (2017). The CNN/DM has 287226/13368/11490 train/val/test samples respectively. The reference summaries have an average of 66 tokens ($\sigma = 26$) and 4.9 sentences. Differing from See et al. (2017), we truncate source documents to 700 tokens instead of 400 in training and validation sets to model longer documents with more sentences. All our experiments were trained on Nvidia GTX Titan X GPUs.

Base Model: Although StructSum framework can be incorporated in any encoder-decoder framework with structure-aware representations, for our experiments we chose the pointer-generator model (See et al., 2017) as the base model, due to its simplicity and ubiquitous usage as a neural abstractive summarization model across different domains (Liu et al., 2019; Krishna et al., 2020). The word and sentence encoders are BiLSTM and the decoder is a BiLSTM with a pointer based copy mechanism. We re-implement the base pointer-generator model and augment it with the StructSum modules described in §2 and hence our model can be directly compared to it.

Baselines: In addition to the base model, we compare StructSum with the following baselines:
Tan et al. (2017): This is a graph-based attention model that is closest in spirit to the method we present in this work. A graph attention module is used to learn attention between sentences, but it cannot be easily used to induce interpretable document structures, since its attention scores are not constrained to learn structure. On top of latent and interpretable structured attention between sentences, StructSum introduces an explicit structure component to inject external document structure, which distinguishes it from Tan et al. (2017).
Gehrmann et al. (2018): This work introduces a separate content selector which tags words and phrases to be copied. The DiffMask variant is an end-to-end variant like ours and hence is included in our baselines. We compare StructSum with the DiffMask experiment.[4]

Hyperparameters: Our encoder uses 256 hidden states for both directions in the one-layer BiLSTM, and 512 for the single-layer decoder. We use the Adagrad optimizer (Duchi et al., 2011) with a learning rate of 0.15 and an initial accumulator value of 0.1. We do not use dropout and use

[3] https://cs.nyu.edu/~kcho/DMQA/

[4] The best results from Gehrmann et al. (2018) outperform DiffMask experiment, but they use inference-time hard masking which can be applied on ours. Our baselines also exclude Reinforcement Learning (RL) based systems as they are not directly comparable, but our approach can be introduced in an encoder-decoder based RL system. Since we do not incorporate any pretraining, we do not compare with recent contextual representation based models (Liu and Lapata, 2019).

[5] https://github.com/atulkum/pointer_summarizer

Model	ROUGE 1	ROUGE 2	ROUGE L
Pointer-Generator (See et al., 2017)	36.44	15.66	33.42
Pointer-Generator + Coverage (See et al., 2017)	39.53	**17.28**	36.38
Graph Attention (Tan et al., 2017)	38.10	13.90	34.00
Pointer-Generator + DiffMask (Gehrmann et al., 2018)	38.45	16.88	35.81
Pointer-Generator (Re-Implementation)	35.55	15.29	32.05
Pointer-Generator + Coverage (Re-Implementation)	39.07	16.97	35.87
Latent-Structure (LS) Attention	39.52	16.94	36.71
Explicit-Structure (ES) Attention	**39.63**	16.98	36.72
LS + ES Attention	39.62	17.00	**36.95**

Table 1: Evaluation of summarization models on the CNN/DM dataset. Published abstractive summarization baseline scores are on top. The bottom section shows re-implementations of See et al. (2017)[5] and StructSum results that incorporate latent and explicit document structure into the base models. StructSum's utility is on par with the base models, while introducing additional benefits of better abstractiveness and intrepretability shown in §4.

gradient-clipping with a maximum norm of 2. We selected the best model using early stopping based on the ROUGE score on the validation dataset as our criteria. We also used the coverage penalty during inference as shown in Gehrmann et al. (2018). For decoding, we use beam-search with a beam width of 3. We did not observe significant improvements with higher beam widths.

4 Evaluation

A standard ROUGE metric does not shed meaningful light into the quality of summaries across important dimensions. As a recall-based metric it is not suitable for assessing the abstractiveness of summarization; it is also agnostic to layout biases and does not facilitate intrepretability of model decisions. We thus adopt automatic metrics tailored to evaluating separately each of these aspects. We compare StructSum to our base model, the pointer-generator network with coverage (See et al., 2017) and the reference.

4.1 Automatic Metrics

We first conduct a standard comparison of generated summaries with reference summaries using ROUGE-1,2 and L (Lin, 2004) F1[6] metric. Table 1 shows the results. We first observe that introducing the latent structures and explicit structures independently improves our performance on ROUGE-L. It suggests that modeling dependencies between sentences helps the model compose better long sequences compared to baselines. We see small improvements in ROUGE-1 and ROUGE-2, hinting that we retrieve similar content words as the baseline but compose them into better contiguous sequences. As both ES and LS independently get similar performance, the results show that LS attention induces good latent dependencies that make up for pure external coreference knowledge.

Finally, our combined model which uses both Latent and Explicit structure performs the best with an improvement of **1.08 points** in ROUGE-L and **0.6 points** in ROUGE-1 over base pointer-generator model (statistically significant for 11490 samples at p=0.05 using Wilson Confidence Test). It shows that the latent and explicit information are complementary and a model can jointly leverage them to produce better summaries. Additionally, we find that structural inductive bias helps a model to converge faster. The combined LS+ES Attention model converges in 126K iterations in comparison to ∼230K iterations required for the pointer-generator network.

While ROUGE is a popular metric used for evaluating summarization models, it is limited to only evaluating n-gram overlap while ignoring semantic correctness. Hence, we compared our method with the baseline Pointer-Generator model using the BERTScore metric (Zhang et al., 2020). We observe that our model improves BERTScore by 9 points (12.3 for Pointer-Generator v/s 21.7 for StructSum) showing that our model is able to generate semantically correct content.

4.2 Abstractiveness

Despite being an abstractive model, the pointer-generator model tends to copy very long sequences of words including whole sentences from the

[6] https://pypi.org/project/pyrouge/

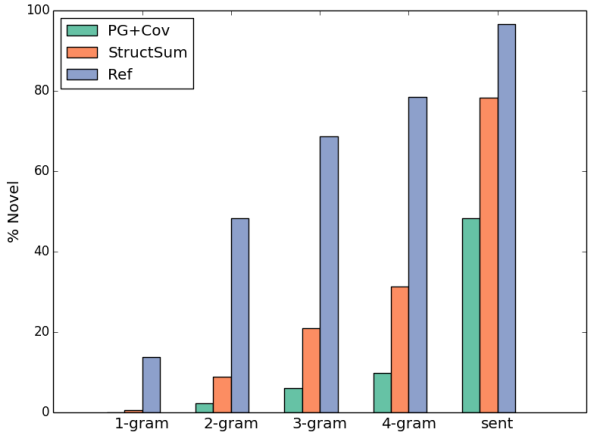

Figure 2: Comparison of % Novel n-grams between StructSum, Pointer-Generator+Coverage and the Reference. Here, "sent" indicates full novel sentences.

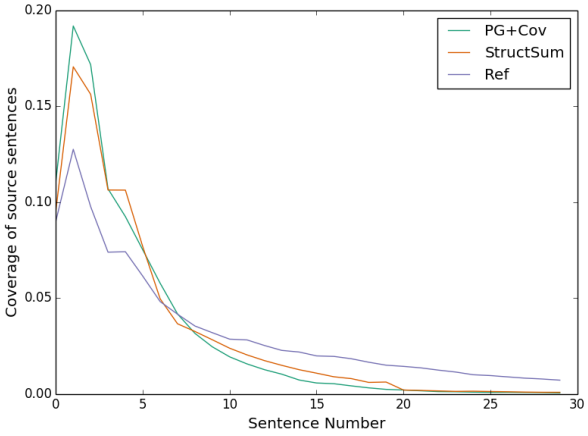

Figure 3: Coverage of source sentences in summary. Here the x-axis is the sentence position in the source article and y-axis shows the normalized count of sentences in that position copied to the summary.

source document (also observed by Gehrmann et al. (2018)). We use two metrics to evaluate the abstractiveness of the model:

Copy Length: Table 2 shows a comparison of the average length (Copy Len) of contiguous copied sequences from the source document (greater than length 3). We observe that the pointer-generator baseline on average copies 16.61 continuous tokens from the source which shows the extractive nature of the model. This indicates that pointer networks, aimed at combining advantages from abstractive and extractive methods by allowing to copy content from the input document, tend to skew towards copying, particularly in this dataset. A consequence of this is that the model fails to interrupt copying at desirable sequence length. In contrast, modeling document structure through StructSum reduces the length of copied sequences to 9.13 words on average reducing the bias of copying sentences entirely. This average is closer to the reference (5.07 words) in comparison, without sacrificing task performance. StructSum learns to stop when needed, while still generating coherent summaries.

Novel N-Grams: The proportion of novel n-grams generated has been used in the literature to measure the degree of abstractiveness of summarization models (See et al., 2017). Figure 2 compares the percentage of novel n-grams in StructSum as compared to the baseline model. Our model produces novel trigrams 21.0% of the time and copies whole sentences only 21.7% of the time. In comparison, the pointer-generator network has only 6.1% novel trigrams and copies entire sentences 51.7% of the time. This shows that StructSum on average generates 14.7% more novel n-grams in comparison to the pointer-generator baseline.

4.3 Coverage

A direct outcome of copying shorter sequences is being able to cover more content from the source document within given length constraints. We observe that this leads to better summarization performance. We compute coverage by computing the number of source sentences from which contiguous sequences greater than length 3 are copied in the summary. Table 2 shows a comparison of the coverage of source sentences in the summary content. While the baseline pointer-generator model only copies from 12.1% of the source sentences, StructSum copies content from 24.0% of the source sentences. Additionally, the average length of the summaries produced by StructSum remains mostly unchanged at 66 words on average compared to 61 of the baseline model. This indicates that StructSum produces summaries that draw from a wider selection of sentences from the original article compared to the baseline models.

4.4 Layout Bias

Neural abstractive summarization methods applied to news articles are typically biased towards selecting and generating summaries based on the first few sentences of the articles. This stems from the structure of news articles, which present the salient information of the article in the first few sentences

	Copy Len	Coverage
PG+Cov	16.61	12.1 %
StructSum	9.13	24.0 %
Reference	5.07	16.7 %

Table 2: Results of analysis of copying and coverage distribution over the source sentences on CNN/DM test set. Copy Len denotes the average length of copied sequences; Coverage – coverage of source sentences.

	Coref	NER	Coref+NER
precision	0.29	0.19	0.33
recall	0.11	0.08	0.09

Table 3: Precision and recall of ES and LS shared edges

and expand in the subsequent ones. As a result, the LEAD 3 baseline, which selects the top three sentences of an article, is widely used in the literature as a strong baseline to evaluate summarization models applied to the news domain (Narayan et al., 2018). Kryscinski et al. (2019) observed that the current summarization models learn to exploit the layout biases of current datasets and offer limited diversity in their outputs.

To analyze whether StructSum also holds the same layout biases, we compute a distribution of source sentence indices that are used for copying content (copied sequences of length 3 or more are considered). Figure 3 shows the distributions of source sentences covered in the summaries. The coverage of sentences in the reference summaries shows a high proportion of the top 5 sentences of any article being copied to the summary. Additionally, the reference summaries have a smoother tail end distribution with relevant sentences in all positions being copied. It shows that a smooth distribution over all sentences is a desirable feature. We notice that the pointer-generator framework have a stronger bias towards the beginning of the article with a high concentration of copied sentences within the top 5 sentences of the article. In contrast, StructSum improves coverage slightly having a lower concentration of top 5 sentences and copies more tail end sentences than the baselines. However, although the modeling of structure does help, our model has a reasonable gap compared to the reference distribution. We see this as an area of improvement and a direction for future work.

Depth	2	3	4	5+
StructSum	29.3%	53.7%	14.4%	2.6%

Table 4: Distribution of latent tree depth.

5 Analysis of Induced Document Structures

Similar to Liu and Lapata (2017), we also look at the quality of the intermediate structures learned by the model. We use the Chu-Liu-Edmonds algorithm (Chu and Liu, 1965; Edmonds, 1967) to extract the maximum spanning tree from the attention score matrix as our sentence structure. Table 4 shows the frequency of various tree depths. We find that the average tree depth is 2.9 and the average proportion of leaf nodes is 88%, consistent with results from tree induction in document classification (Ferracane et al., 2019). Further, we compare latent trees extracted from StructSum with undirected graphs based on coreference, on NER, or on both. These are constructed similarly to our explicit coreference based sentence graphs in §2.3 by linking sentences with overlapping coreference mentions or named entities. We measure the similarity between the learned latent trees and the explicit graphs through precision and recall over edges. The results are shown in Table 3. We observe that our latent graphs have low recall with the linguistic graphs showing that our latent graphs do not capture the coreference or named entity overlaps explicitly, suggesting that the latent and explicit structures capture complementary information.

Figure 4 shows qualitative examples of induced structures along with summaries from the StructSum. The first example shows a tree with sentence 3 chosen as root, which was the key sentence mentioned in the reference. In both examples, the sentences in the lower level of the dependency tree contribute less to the generated summary. Similarly, in the examples source sentences used to generate summaries tend to be closer to the root node. In the first summary, all source content sentences used in the summary are either the root node or within depth 1 of the root node. In the second example, 4 out of 5 source sentences were at depth=1 in the tree. In both examples, generated summaries diverged from the reference by omitting certain sentences used in the reference. These sentences are in the lower section of the tree, providing insights on which sentences were preferred for the

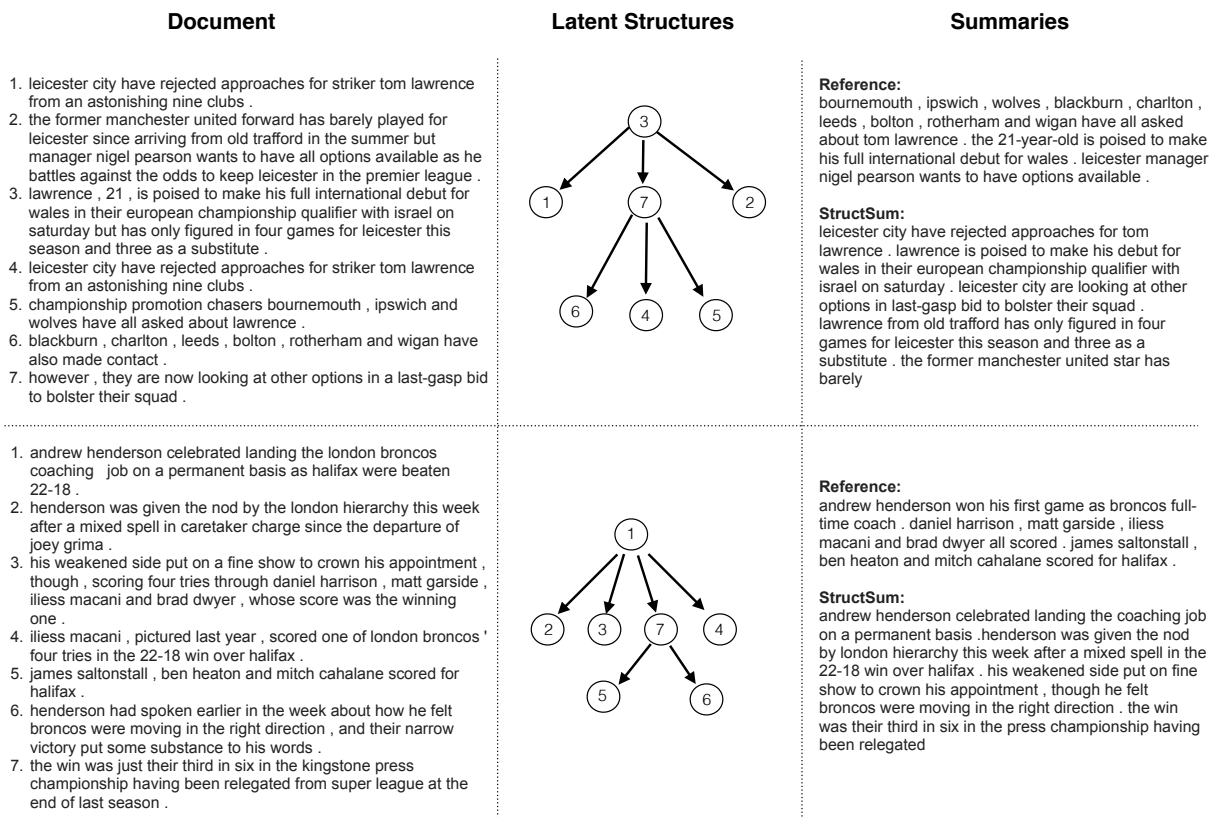

Figure 4: Examples of induced structures and generated summaries.

summary generation. We also see in example 1 that the latent structures cluster sentences based on the main topic of the document. Sentences 1,2,3 differ from sentences 5,6,7 in the topic discussed and our model clustered the two sets separately.

6 Related Work

Data-driven neural summarization falls into *extractive* (Cheng et al., 2016; Zhang et al., 2018) or *abstractive* (Rush et al., 2015; See et al., 2017; Gehrmann et al., 2018; Chen and Bansal, 2018). Pointer-generator See et al. (2017) learns to either generate novel in-vocabulary words or copy from the source. It has been the foundation for much work on abstractive summarization (Gehrmann et al., 2018; Hsu et al., 2018; Song et al., 2018). Our model extends it by incorporating latent/explicit structure, but these extensions are applicable to any other encoder-decoder architecture. For example, a follow-up study has already shown benefits of our method in multi-document summarization (Chowdhury et al., 2020).

In pre-neural era, document structure played a critical role in summarization (Leskovec et al., 2004; Litvak and Last, 2008; Liu et al., 2015; Durrett et al., 2016; Kikuchi et al., 2014). More recently Song et al. (2018) infuse source syntactic structure into the pointer-generator using word-level syntactic features and augmenting them to decoder copy mechanism. In contrast, we model sentence dependencies as latent structures and explicit coreference structures; we do not use heuristics or salient features. Li et al. (2018) propose structural compression and coverage regularizers incorporating structural bias of target summaries while we model the structure of the source document. Frermann and Klementiev (2019) induce latent structures for aspect based summarization, Cohan et al. (2018) focus on summarization of scientific papers, Isonuma et al. (2019) reviews unsupervised summarization, Mithun and Kosseim (2011) use discourse structures to improve coherence in blog summarization and Ren et al. (2018) use sentence relations for multi-document summarization. These are complementary directions to our work. To our knowledge, StructSum is the first to jointly incorporate latent and explicit document structure in a summarization framework.

7 Conclusion and Future Work

In this work, we propose the framework StructSum for incorporating latent and explicit document

structure in neural abstractive summarization. We introduce a novel explicit-attention module which incorporates external linguistic structures, instantiating it with coreference links. We show that our framework improves the abstractiveness and coverage of generated summaries, and helps mitigate layout biases associated with prior models. We present an extensive evaluation of StructSum along abstractiveness, coverage, and layout quantitatively. Future work will investigate the role of document structures in pretrained language models (Lewis et al., 2019; Liu and Lapata, 2019).

Acknowledgements

The authors are grateful to the anonymous reviewers for their invaluable feedback and to Waleed Ammar and Kathryn Mazaitis for their help and support in various stages of the project. This material is based upon work supported by the DARPA SemaFor and NNSA DOE programs. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the DARPA or NNSA. We would also like to thank Amazon for providing GPU credits.

References

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–686, Melbourne, Australia. Association for Computational Linguistics.

Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long short-term memory-networks for machine reading. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 551–561, Austin, Texas. Association for Computational Linguistics.

Tanya Chowdhury, Sachin Kumar, and Tanmoy Chakraborty. 2020. Neural abstractive summarization with structural attention. *arXiv preprint arXiv:2004.09739*.

Yau Chu and Tung Kuan Liu. 1965. On the shortest arborescence of a directed graph. *Science Sinica*.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Bonnie Dorr, David Zajic, and Richard Schwartz. 2003. Hedge trimmer: A parse-and-trim approach to headline generation. In *Proceedings of the HLT-NAACL 03 on Text summarization workshop-Volume 5*, pages 1–8. Association for Computational Linguistics.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.

Greg Durrett, Taylor Berg-Kirkpatrick, and Dan Klein. 2016. Learning-based single-document summarization with compression and anaphoricity constraints. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1998–2008, Berlin, Germany. Association for Computational Linguistics.

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

Elisa Ferracane, Greg Durrett, Junyi Jessy Li, and Katrin Erk. 2019. Evaluating discourse in structured text representations. In *ACL*.

Lea Frermann and Alexandre Klementiev. 2019. Inducing document structure for aspect-based summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6263–6273.

Sebastian Gehrmann, Yuntian Deng, and Alexander M. Rush. 2018. Bottom-up abstractive summarization. In *EMNLP*.

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.

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

Wan Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A unified model for extractive and abstractive summarization using inconsistency loss. In *ACL*.

Masaru Isonuma, Junichiro Mori, and Ichiro Sakata. 2019. Unsupervised neural single-document summarization of reviews via learning latent discourse structure and its ranking. In *ACL*.

Chris Kedzie, Kathleen McKeown, and Hal Daumé III. 2018. Content selection in deep learning models of

summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1818–1828, Brussels, Belgium. Association for Computational Linguistics.

Yuta Kikuchi, Tsutomu Hirao, Hiroya Takamura, Manabu Okumura, and Masaaki Nagata. 2014. Single document summarization based on nested tree structure. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 315–320.

Yoon Kim, Carl Denton, Luong Hoang, and Alexander M. Rush. 2017. Structured attention networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Kundan Krishna, Sopan Khosla, Jeffrey P Bigham, and Zachary C Lipton. 2020. Generating soap notes from doctor-patient conversations. *arXiv preprint arXiv:2005.01795*.

Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551, Hong Kong, China. Association for Computational Linguistics.

Jure Leskovec, Marko Grobelnik, and Natasa Milic-Frayling. 2004. Learning sub-structures of document semantic graphs for document summarization. In *LinkKDD Workshop*, pages 133–138.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *ArXiv*, abs/1910.13461.

Wei Li, Xinyan Xiao, Yajuan Lyu, and Yuanzhuo Wang. 2018. Improving neural abstractive document summarization with structural regularization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4078–4087.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.

Hui Lin and Vincent Ng. 2019. Abstractive summarization: A survey of the state of the art. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9815–9822.

Marina Litvak and Mark Last. 2008. Graph-based keyword extraction for single-document summarization. In *Proceedings of the workshop on Multi-source Multilingual Information Extraction and Summarization*, pages 17–24. Association for Computational Linguistics.

Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A. Smith. 2015. Toward abstractive summarization using semantic representations. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1077–1086, Denver, Colorado. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2017. Learning structured text representations. *Transactions of the Association for Computational Linguistics*, 6:63–75.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. *IJCNLP*, abs/1908.08345.

Zhengyuan Liu, Angela Ng, Sheldon Lee Shao Guang, AiTi Aw, and Nancy F. Chen. 2019. Topic-aware pointer-generator networks for summarizing spoken conversations. *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 814–821.

Shamima Mithun and Leila Kosseim. 2011. Discourse structures to reduce discourse incoherence in blog summarization. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 479–486, Hissar, Bulgaria. Association for Computational Linguistics.

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*.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar GuÌ‡lçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Pengjie Ren, Zhumin Chen, Zhaochun Ren, Furu Wei, Liqiang Nie, Jun Ma, and Maarten De Rijke. 2018. Sentence relations for extractive summarization with deep neural networks. *ACM Transactions on Information Systems (TOIS)*, 36(4):1–32.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 379–389.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Kaiqiang Song, Lin Zhao, and Fei Liu. 2018. Structure-infused copy mechanisms for abstractive summarization. In *COLING*.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1171–1181.

William Thomas Tutte. 1984. Graph theory, vol. 21 of. *Encyclopedia of Mathematics and its Applications*.

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 5998–6008.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. 2016. Hierarchical attention networks for document classification. In *HLT-NAACL*.

Tianyi Zhang, V. Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. *ICLR*.

Xingxing Zhang, Mirella Lapata, Furu Wei, and Ming Zhou. 2018. Neural latent extractive document summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 779–784, Brussels, Belgium. Association for Computational Linguistics.