Sentence-Level Content Planning and Style Specification for Neural Text Generation

Xinyu Hua and **Lu Wang**
Khoury College of Computer Sciences
Northeastern University
Boston, MA 02115
`hua.x@husky.neu.edu`   `luwang@ccs.neu.edu`

Abstract

Building effective text generation systems requires three critical components: content selection, text planning, and surface realization, and traditionally they are tackled as separate problems. Recent all-in-one style neural generation models have made impressive progress, yet they often produce outputs that are incoherent and unfaithful to the input. To address these issues, we present an end-to-end trained two-step generation model, where a sentence-level content planner first decides on the keyphrases to cover as well as a desired language style, followed by a surface realization decoder that generates relevant and coherent text. For experiments, we consider three tasks from domains with diverse topics and varying language styles: persuasive argument construction from Reddit, paragraph generation for normal and simple versions of Wikipedia, and abstract generation for scientific articles. Automatic evaluation shows that our system can significantly outperform competitive comparisons. Human judges further rate our system generated text as more fluent and correct, compared to the generations by its variants that do not consider language style.

1 Introduction

Automatic text generation is a long-standing challenging task, as it needs to solve at least three major problems: (1) *content selection* ("what to say"), identifying pertinent information to present, (2) *text planning* ("when to say what"), arranging content into ordered sentences, and (3) *surface realization* ("how to say it"), deciding words and syntactic structures that deliver a coherent output based on given discourse goals (McKeown, 1985). Traditional text generation systems often handle each component separately, thus requiring extensive effort on data acquisition and system engineering (Reiter and Dale, 2000). Recent progress

Figure 1: [Upper] Sample counter-argument from Reddit. Argumentative stylistic language for persuasion is in italics. [Bottom] Excerpts from Wikipedia, where sophisticated concepts and language of higher complexity used in the standard version are not present in the corresponding simplified version. Both: key concepts are in bold.

has been made by developing end-to-end trained neural models (Rush et al., 2015; Yu et al., 2018; Fan et al., 2018), which naturally excel at producing fluent text. Nonetheless, limitations of model structures and training objectives make them suffer from low interpretability and substandard generations which are often incoherent and unfaithful to the input material (See et al., 2017; Wiseman et al., 2017; Li et al., 2017).

To address the problems, we believe it is imperative for neural models to gain adequate control on content planning (i.e., content selection and ordering) to produce coherent output, especially for long text generation. We further argue that, in order to achieve desired discourse goals, it is beneficial to enable style-controlled surface realization by explicitly modeling and specifying proper linguistic styles. Consider the task of producing counter-arguments to the topic "US should cut off foreign aid completely". A sample argument in Figure 1 demonstrates how human selects
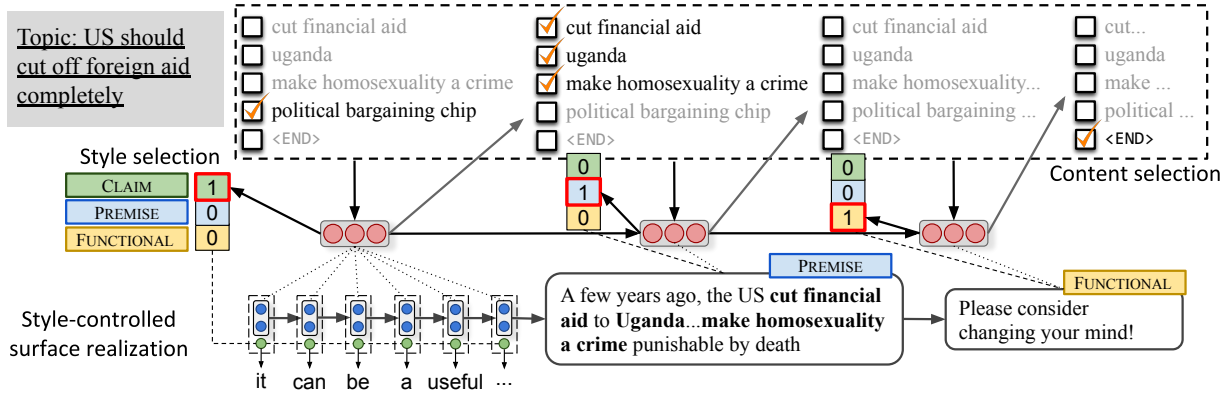
Figure 2: Overview of our framework. The LSTM content planning decoder (§ 3.2) first identifies a set of keyphrases from the memory bank conditional on previous selection history, based on which, a style is specified. During surface realization, the hidden states of the planning decoder and the predicted style encoding are fed into the realizer, which generates the final output (§ 3.3). Best viewed in color.

a series of talking points and a proper style based on the argumentative function for each sentence. For instance, the argument starts with a *proposition* on "foreign aid as a political bargaining chip", followed by a concrete *example* covering several key concepts. It ends with *argumentative stylistic language*, which differs in both content and style from the previous sentences. Figure 1 shows another example on Wikipedia articles: compared to a topic's standard version where longer sentences with complicated concepts are constructed, its simplified counterpart tends to explain the same subject with plain language and simpler concepts, indicating the interplay between content selection and language style.

We thus present an end-to-end trained neural text generation framework that includes the modeling of traditional generation components, to promote the control of content and linguistic style of the produced text.[1] Our model performs *sentence-level content planning* for information selection and ordering, and *style-controlled surface realization* to produce the final generation. We focus on conditional text generation problems (Lebret et al., 2016; Colin et al., 2016; Dušek et al., 2018): As shown in Figure 2, the input to our model consists of a topic statement and a set of keyphrases. The output is a relevant and coherent paragraph to reflect the salient points from the input. We utilize two separate decoders: for each sentence, (1) a planning decoder selects relevant keyphrases and a desired style conditional on previous selections, and (2) a realization decoder produces the text in the specified style.

We demonstrate the effectiveness of our framework on three challenging datasets with diverse topics and varying linguistic styles: persuasive argument generation on Reddit Change-MyView (Hua and Wang, 2018); introduction paragraph generation on a *newly* collected dataset from Wikipedia and its simple version; and scientific paper abstract generation on AGENDA dataset (Koncel-Kedziorski et al., 2019).

Experimental results on all three datasets show that our models that consider content planning and style selection achieve significantly better BLEU, ROUGE, and METEOR scores than non-trivial comparisons that do not consider such information. Human judges also rate our model generations as more fluent and correct compared to the outputs produced by its variants without style modeling.

2 Related Work

Content selection and text planning are critical components in traditional text generation systems (Reiter and Dale, 2000). Early approaches separately construct each module and mainly rely on hand-crafted rules based on discourse theory (Scott and de Souza, 1990; Hovy, 1993) and expert knowledge (Reiter et al., 2000), or train statistical classifiers with rich features (Duboue and McKeown, 2003; Barzilay and Lapata, 2005). Advances in neural generation models have alleviated human efforts on system engineering, by combining all components into an end-to-end trained conditional text generation framework (Mei et al., 2016; Wiseman et al., 2017). However, without

---

[1]Data and code are available at `xinyuhua.github.io/Resources/emnlp19/`.

proper planning and control (Rambow and Korelsky, 1992; Stone and Doran, 1997; Walker et al., 2001), the outputs are often found to be incoherent and hallucinating. Recent work (Moryossef et al., 2019) separates content selection from the neural generation process and shows improved generation quality. However, their method requires an exhaustive search for content ordering and is therefore hard to generalize and scale. In this work, we improve the content selection by incorporating past selection history and directly feeding the predicted language style into the realization module.

Our work is also inline with concept-to-text generation, where sentences are produced from *structured* representations, such as database records (Konstas and Lapata, 2013; Lebret et al., 2016; Wiseman et al., 2017; Moryossef et al., 2019), knowledge base items (Luan et al., 2018; Koncel-Kedziorski et al., 2019), and AMR graphs (Konstas et al., 2017; Song et al., 2018; Koncel-Kedziorski et al., 2019). Shared tasks such as WebNLG (Colin et al., 2016) and E2E NLG challenges (Dušek et al., 2019) have been designed to evaluate single sentence planning and realization from the given structured inputs with a small set of fixed attribute types. Planning for multiple sentences in the same paragraph is nevertheless much less studied; it poses extra challenges for generating coherent long text, which is addressed in this work. Moreover, structured inputs are only available in a limited number of domains (Tanaka-Ishii et al., 1998; Chen and Mooney, 2008; Belz, 2008; Liang et al., 2009; Chisholm et al., 2017). The emerging trend is to explore less structured data (Kiddon et al., 2016; Fan et al., 2018; Martin et al., 2018). In our work, keyphrases are used as input to our generation system, which offer flexibility for concept representation and generalizability to broader domains.

3 Model

Our model tackles conditional text generation tasks where the input is comprised of two major parts: (1) a topic statement, $\mathbf{x} = \{x_i\}$, which can be an argument, the title of a Wikipedia article, or a scientific paper title, and (2) a keyphrase memory bank, $\mathcal{M}$, containing a list of talking points, which plays a critical role in content planning and style selection. We aim to produce a sequence of words, $\mathbf{y} = \{y_t\}$, to comprise the output,

which can be a counter-argument, a paragraph as in Wikipedia articles, or a paper abstract.

3.1 Input Encoding

The input text $\mathbf{x}$ is encoded via a bidirectional LSTM (biLSTM), with its last hidden state used as the initial states for both content planning decoder and surface realization decoder. To encode keyphrases in the memory bank $\mathcal{M}$, each keyphrase is first converted into a vector $\boldsymbol{e}_k$ by summing up all its words' embeddings from GloVe (Pennington et al., 2014). A biLSTM-based *keyphrase reader*, with hidden states $\boldsymbol{h}_k^e$, is used to encode all keyphrases in $\mathcal{M}$. We also insert entries of <START> and <END> into $\mathcal{M}$ to facilitate learning to start and finish selection.

3.2 Sentence-Level Content Planning and Style Specification

Content Planning: Context-Aware Keyphrase Selection. Our content planner selects a set of keyphrases from the memory bank $\mathcal{M}$ for each sentence, indexed with j, conditional on keyphrases that have been selected in previous sentences, allowing topical coherence and content repetition avoidance. The decisions are denoted as a *selection vector* $\boldsymbol{v}_j \in \mathbb{R}^{|\mathcal{M}|}$, with each dimension $\boldsymbol{v}_{j,k} \in \{0, 1\}$, indicating whether the k-th phrase is selected for the j-th sentence generation. Starting with a <START> tag as the input for the first step, our planner predicts $\boldsymbol{v}_1$ for the first sentence, and recurrently makes predictions per sentence until <END> is selected, as depicted in Figure 2.

Formally, we utilize a sentence-level LSTM f, which consumes the summation embedding of selected keyphrases, $\boldsymbol{m}_j$, to produce a hidden state $\boldsymbol{s}_j$ for the j-th sentence step:

$$\boldsymbol{s}_j = f(\boldsymbol{s}_{j-1}, \boldsymbol{m}_j) \tag{1}$$

$$\boldsymbol{m}_j = \sum_{k=1}^{|\mathcal{M}|} \boldsymbol{v}_{j,k} \boldsymbol{h}_k^e \tag{2}$$

where $\boldsymbol{v}_{j,k} \in \{0, 1\}$ is the selection decision for the k-th keyphrase in the j-th sentence.

Our recent work (Hua et al., 2019) utilizes a similar formulation for sentence representations. However, the prediction of $\boldsymbol{v}_{j+1}$ is estimated by a bilinear product between $\boldsymbol{h}_k^e$ and $\boldsymbol{s}_j$, which is agnostic to what have been selected so far. While in reality, content selection for a new sentence should depend on previous selections. For in-

stance, keyphrases that have already been utilized many times are less likely to be picked again; topically related concepts tend to be mentioned closely. We therefore propose a vector $\boldsymbol{q}_j$ that keeps track of what keyphrases have been selected up to the j-th sentence:

$$\boldsymbol{q}_j = (\sum_{r=0}^{j} \boldsymbol{v}_r)^T \times \mathbb{E} \qquad (3)$$

where $\mathbb{E} = [\boldsymbol{h}_1^e, \boldsymbol{h}_2^e, \ldots \boldsymbol{h}_{|\mathcal{M}|}^e]^T \in \mathbb{R}^{|\mathcal{M}| \times H}$ is the matrix of keyphrase representations, H is the hidden dimension of the keyphrase reader LSTM.

Then $\boldsymbol{v}_{j+1}$ is calculated in an attentive manner with $\boldsymbol{q}_j$ as the attention query:

$$P(\boldsymbol{v}_{j+1,k} = 1|\boldsymbol{v}_{1:j}) = \sigma(\mathbf{w}_v^T \boldsymbol{s}_j + \boldsymbol{q}_j \mathbf{W}^c \boldsymbol{h}_k^e) \qquad (4)$$

where σ is the sigmoid funciton, and $\mathbf{w}_*$, $\mathbf{W}^*$, and $\mathbf{W}^{**}$ are trainable parameters throughout the paper. Bias terms are all omitted for simplicity.

As part of the learning objective, we utilize the binary cross-entropy loss with the gold-standard selection $\boldsymbol{v}_j^*$ as criterion over the training set D:

$$\mathcal{L}_{\text{sel}} = - \sum_{(\mathbf{x},\mathbf{y}) \in D} \sum_{j=1}^{J} (\sum_{k=1}^{|\mathcal{M}|} \log(P(\boldsymbol{v}_{j,k}^*))) \qquad (5)$$

Style Specification. As discussed in § 1, depending on the content (represented as selected keyphrases in our model), humans often choose different language styles adapted for different discourse goals. Our model characterizes such stylistic variations by assigning a categorical style type $\boldsymbol{t}_j$ for each sentence, which is predicted as follows:

$$\hat{\boldsymbol{t}}_j = \text{softmax}(\mathbf{w}_s^T (\tanh (\mathbf{W}^s [\boldsymbol{m}_j; \boldsymbol{s}_j]))) \qquad (6)$$

$\hat{\boldsymbol{t}}_j$ is the estimated distribution over all types. We select the one with the highest probability and use a one-hot encoding vector, $\boldsymbol{t}_j$, as the input to our realization decoder (§ 3.3). The estimated distributions $\hat{\boldsymbol{t}}_j$ are compared against the gold-standard labels $\boldsymbol{t}_j^*$ to calculate the cross-entropy loss $\mathcal{L}_{\text{style}}$:

$$\mathcal{L}_{\text{style}} = - \sum_{(\mathbf{x},\mathbf{y}) \in D} \sum_{j=1}^{J} \boldsymbol{t}_j^* \log \hat{\boldsymbol{t}}_j \qquad (7)$$

3.3 Style-Controlled Surface Realization

Our surface realization decoder is implemented with an LSTM with state calculation function g to get each hidden state $\boldsymbol{z}_t$ for the t-th generated token. To compute $\boldsymbol{z}_t$, we incorporate the content

planning decoder hidden state $\boldsymbol{s}_{J(t)}$ for the sentence to be generated, with $J(t)$ as the sentence index, and previously generated token $\boldsymbol{y}_{t-1}$:

$$\boldsymbol{z}_t = g(\boldsymbol{z}_{t-1}, \tanh(\mathbf{W}^{ws} \boldsymbol{s}_{J(t)} + \mathbf{W}^{ww} \boldsymbol{y}_{t-1})) \qquad (8)$$

For word prediction, we calculate two attentions, one over the input statement $\mathbf{x}$, which produces a context vector $\boldsymbol{c}_t^w$ (Eq. 10), the other over the keyphrase memory bank $\mathcal{M}$, which generates $\boldsymbol{c}_t^e$ (Eq. 11). To better reflect the control over word choice by language styles, we directly append the predicted style $\boldsymbol{t}_{J(t)}$ to the context vectors and hidden state $\boldsymbol{z}_t$, to compute the distribution over the vocabulary[2]:

$$P(y_t|y_{1:t-1}) = \text{softmax}(\tanh(\mathbf{W}^o[\boldsymbol{z}_t; \boldsymbol{c}_t^w; \boldsymbol{c}_t^e; \boldsymbol{t}_{J(t)}]))$$
$$(9)$$

$$\boldsymbol{c}_t^w = \sum_{i=1}^{L} \alpha_i^w \boldsymbol{h}_i, \quad \alpha_i^w = \text{softmax}(\boldsymbol{z}_t \mathbf{W}^{wa} \boldsymbol{h}_i) \qquad (10)$$

$$\boldsymbol{c}_t^e = \sum_{k=1}^{|\mathcal{M}|} \alpha_k \boldsymbol{h}_k^e, \quad \alpha_k^e = \text{softmax}(\boldsymbol{z}_t \mathbf{W}^{we} \boldsymbol{h}_k^e) \qquad (11)$$

We further adopt a copying mechanism from See et al. (2017) to enable direct reuse of words from the input $\mathbf{x}$ and keyphrase bank $\mathcal{M}$ to allow out-of-vocabulary words to be included.

3.4 Training Objective

We jointly learn to conduct content planning and surface realization by aggregating the losses over (i) word generation: $\mathcal{L}_{\text{gen}} = -\sum_D \sum_{t=1}^{T} \log P(y_t^*|\mathbf{x}; \theta)$, (ii) keyphrase selection: $\mathcal{L}_{\text{sel}}$ (Eq. 5), and (iii) style prediction $\mathcal{L}_{\text{style}}$ (Eq. 7):

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{gen}}(\theta) + \gamma \cdot \mathcal{L}_{\text{style}}(\theta) + \eta \cdot \mathcal{L}_{\text{sel}}(\theta) \qquad (12)$$

where θ denotes the trainable parameters. γ and η are set to 1.0 in our experiments for simplicity.

4 Tasks and Datasets

4.1 Task I: Argument Generation

Our first task is to generate a counter-argument for a given statement on a controversial issue. The input keyphrases are extracted from automatically retrieved and reranked passages with queries constructed from the input statement.

---

[2]The inclusion of style variables is different from our prior style-aware generation model (Hua et al., 2019), where styles are predicted but not encoded for word production.

	Argument	Wikipedia	AGENDA
	# Args (# Threads)	(Nor. / Sim.)	
# Train	272,147 (11,434)	125,136	38,720
# Dev	40,291 (1,784)	21,004	1,000
# Test	46,757 (1,706)	23,534	1,000
# Tokens	54.87	70.57 / 48.60	141.34
# Sent.	2.48	3.15 / 3.20	5.59
# KP (candidates)	55.80	23.56	12.23
# KP (selected)	11.61	16.01/11.11	12.23

Table 1: Statistics of the three datasets. Average numbers are reported. For argument dataset, number of unique threads is also shown. On AGENDA, entities are extracted from abstract as keyphrases, hence all candidates are "selected".

We reuse the dataset from our previous work (Hua et al., 2019), but annotate with newly designed style scheme. We first briefly summarize the procedures for data collection, keyphrase extraction and selection, and passage reranking; more details can be found in our prior work. Then we describe how to label argument sentences with style types that capture argumentative structures.

The dataset is collected from Reddit /r/ChangeMyView subcommunity, where each thread consists of a multi-paragraph original post (OP), followed by user replies with the intention to change the opinion of the OP user. Each OP is considered as the input, and the root replies awarded with delta (Δ), or with positive karma (# upvotes > # downvotes) are target counter-arguments to be generated. A domain classifier is further adopted to select politics related threads. Since users often have separate arguments in different paragraphs, we treat each paragraph as one *target argument* by itself. Statistics are shown in Table 1.

Input Keyphrases and Label Construction. To obtain the input keyphrase candidates and their sentence-level selection labels, we first construct queries to retrieve passages from Wikipedia and news articles collected from commoncrawl. org.[3] For training, we construct a query per target argument sentence using its content words for retrieval, and keep top 5 passages per query. For testing, the queries are constructed from the sentences in OP (input statement).

We then extract keyphrases from the retrieved passages based on topic signature words (Lin and Hovy, 2000) calculated over the given OP. These

---

	CLAIM	PREMISE	FUNCTIONAL
# Arguments	29.1%	62.2%	8.7%
# Tokens	17.0	26.2	10.0

Length $\in$	(0, 10]	(10, 20]	(20, 30]	(30, ∞)
Normal Wikipedia	9.9%	40.5%	29.8%	19.8%
Simple Wikipedia	29.3%	51.7%	14.6%	4.4%

Table 2: Sentence style distribution for argument and Wikipedia datasets.

words, together with their related terms from WordNet (Miller, 1994), are used to determine whether a phrase in the passage is a *keyphrase*. Specifically, a keyphrase is (1) a noun phrase or verb phrase that is shorter than 10 tokens; (2) contains at least one content word; (3) has a topic signature or a Wikipedia title. For each keyphrase candidate, we match them with the sentences in the target counter-argument, and we consider it to be "selected" for the sentence if there is any overlapping content word.

During test time, we further adopt a stance classifier from Bar-Haim et al. (2017) to produce a stance score for each passage. We retain passages that have a negative stance towards OP, and a greater than 5 stance score. They are further ordered based on the number of overlapping keyphrases with the OP. Top 10 passages are used to construct the input keyphrase bank, and as optional input to our model.

Sentence Style Label Construction. For argument generation, we define three sentence styles based on their argumentative discourse functions (Persing and Ng, 2016; Lippi and Torroni, 2016): CLAIM is a proposition, usually containing one or two talking points, e.g., "*I believe foreign aid is a useful bargaining chip*"; PREMISE contains supporting arguments with reasoning or examples; FUNCTIONAL is usually a generic statement, e.g., "*I understand what you said*". For training, we employ a list of rules extended from the claim detection method by Levy et al. (2018) to automatically construct a style label for each sentence. Statistics are displayed in Table 2, and sample rules are shown below, with the complete list in the Supplementary:

- CLAIM: must be shorter than 20 tokens and matches any of the following patterns: (a) i (don't)? (believe|agree|...); (b) (anyone|all|everyone|nobody...) (should|could|need|must|might...);

```
(c) (in my opinion|my view|...)
```

- PREMISE: must be longer than 5 tokens, contains at least one noun or verb content word, and matches any of the following patterns: (a) `(for (example|instance)|e.g.)`; (b) `(increase|reduce|improve|...)`

- FUNCTIONAL: contains fewer than 5 alphabetical words and no noun or verb content word

Paragraphs that only contain FUNCTIONAL sentences are removed from our dataset.

4.2 Task II: Paragraph Generation for Normal and Simple Wikipedia

The second task is generating introduction paragraphs for Wikipedia articles. The input consists of a title, a user-specified global style (normal or simple), and a list of keyphrases collected from the gold-standard paragraphs of both normal and simple Wikipedia. During training and testing, the global style is encoded as one extra bit appended to m_j (Eq. 2).

We construct a *new* dataset with topically-aligned paragraphs from normal and simple English Wikipedia.[4] For alignment, we consider it a match if two articles share exactly the same title with at most two non-English words. We then extract the first paragraphs from both and filter out the pair if one of the paragraphs is shorter than 10 words or is followed by a table.

Input Keyphrases and Label Construction. Similar to argument generation, we extract noun phrases and verb phrases and consider the ones with at least one content word as keyphrase candidates. After de-duplication, there are on average 5.4 and 3.7 keyphrases per sentence for the normal and simple Wikipedia paragraphs, respectively. For each sample, we merge the keyphrases from the aligned paragraphs as the input. The model is then trained to select the appropriate ones conditioned on the global style.

Sentence Style Label Construction. We distinguish sentence-level styles based on language complexity, which is approximated by sentence length. The distribution of sentence styles is displayed in Table 2.

4.3 Task III: Paper Abstract Generation

We further consider a task of generating abstracts for scientific papers (Ammar et al., 2018), where

---
[4] We download the dumps of 2019/04/01 for both dataset.

the input contains a paper title and scientific entities mentioned in the abstract. We use the AGENDA data processed by Koncel-Kedziorski et al. (2019), where entities and their relations in the abstracts are extracted by SciIE (Luan et al., 2018). All entities appearing in the abstract are included in our keyphrase bank. The state-of-the-art system (Koncel-Kedziorski et al., 2019) exploits the scientific entities, their relations, and the relation types. In our setup, we ignore the relation graph, and focus on generating the abstract with only entities and title as the input. Due to the dataset's relatively uniform language style and smaller size, we do not experiment with our style specification component.

5 Experiments

5.1 Implementation Details

For argument generation, we truncate the input OP and retrieved passages to 500 and 400 words. Passages are optionally appended to OP as our encoder input. The keyphrase bank size is limited to 70 for argument, and 30 for Wikipedia and AGENDA data (based on the average numbers in Table 1), with keyphrases truncated to 10 words. We use a vocabulary size of 50K for all tasks.

Training Details. Our models use a two-layer LSTM for both decoders. They all have 512-dimensional hidden states per layer and dropout probabilities (Gal and Ghahramani, 2016) of 0.2 between layers. Wikipedia titles are encoded with the summation of word embeddings due to their short length. The learning process is driven by AdaGrad (Duchi et al., 2011) with 0.15 as the learning rate and 0.1 as the initial accumulator. We clip the gradient norm to a maximum of 2.0. The mini-batch size is set to 64. And the optimal weights are chosen based on the validation loss.

For argument generation, we also pre-train the encoder and the lower layer of realization decoder using language model losses. We collect all the OP posts from the training set, and an extended set of reply paragraphs, which includes additional counter-arguments that have non-negative `karma`. For Wikipedia, we consider the large collection of 1.9 million unpaired normal English Wikipedia paragraphs to pre-train the model for both normal and simple Wikipedia generation.

Beam Search Decoding. For inference, we utilize beam search with a beam size of 5. We disallow the repetition of trigrams, and replace the UNK

with the keyphrase of the highest attention score.

5.2 Baselines and Comparisons

For all three tasks, we consider a SEQ2SEQ with attention baseline (Bahdanau et al., 2015), which encodes the input text and keyphrase bank as a sequence of tokens, and generates the output.

For argument generation, we implement a RETRIEVAL baseline, which returns the highest reranked passage retrieved with OP as the query. We also compare with our prior model (Hua and Wang, 2018), which is a multi-task learning framework to generate both keyphrases and arguments.

For Wikipedia generation, a RETRIEVAL baseline obtains the most similar paragraph from the training set with input title and keyphrases as the query, measured with bigram cosine similarity. We further train a logistic regression model (LOGREGSEL), which takes the summation of word embeddings in a phrase and predicts its inclusion in the output for a normal or simple Wiki paragraph.

For abstract generation, we compare with the state-of-the-art system GRAPHWRITER (Koncel-Kedziorski et al., 2019), which is a transformer model enabled with knowledge graph encoding mechanism to handle both the entities and their structural relations from the input.

We also report results by our model variants to demonstrate the usefulness of content planning and style control: (1) with gold-standard[5] keyphrase selection for each sentence (Oracle Plan.), and (2) without style specification.

6 Results and Analysis

6.1 Automatic Evaluation

We report precesion-oriented BLEU (Papineni et al., 2002), recall-oriented ROUGE-L (Lin, 2004) that measures the longest common subsequence, and METEOR (Denkowski and Lavie, 2014), which considers both precision and recall.

Argument Generation. For each input OP, there can be multiple possible counter-arguments. We thus consider the best matched (i.e., highest scored) reference when reporting results in Table 3. Our models yield significantly higher BLEU and ROUGE scores than all comparisons while

---

[5]"Gold-standard" indicates the keyphrases that have content word overlap with the reference sentence.

	BLEU	ROUGE	MTR	Len.
RETRIEVAL	7.81	15.68	**10.59**	150.0
SEQ2SEQ	3.64	19.00	9.85	51.7
H&W (2018)	5.73	14.44	3.82	36.5
OURS (Oracle Plan.)	16.30*	20.25*	11.61	65.5
OURS	**13.19***	20.15*	10.42	65.2
w/o Style	12.61*	**20.28***	10.15	64.5
w/o Passage	11.84*	19.90*	9.03	62.6

Table 3: Results on argument generation with BLEU (up to bigrams), ROUGE-L, and METEOR (MTR). Best systems without oracle planning are in **bold** per metric. Our models that are significantly better than all comparisons are marked with $*$ ($p < 0.001$, approximate randomization test (Noreen, 1989)).

producing longer arguments than generation-based approaches. Furthermore, among our model variants, oracle content planning further improves the performance, indicating the importance of content selection and ordering. Taking out style specification decreases scores, indicating the influence of style control on generation.[6]

Wikipedia Generation. Results on Wikipedia (Table 4) show similar trends, where our models almost always outperform all comparisons across metrics. The significant performance drop on ablated models without style prediction proves the effectiveness of style usage. Our model, if guided with oracle keyphrase selection per sentence, again achieves the best performance.

We further show the *effect of content selection on generation* on Wikipedia and abstract data in Figure 3, where we group the test samples into 10 bins based on F1 scores on keyphrase selection.[7] We observe a strong correlation between keyphrase selection and generation performance, e.g., for BLEU, Pearson correlations of 0.95 ($p < 10^{-4}$) and 0.85 ($p < 10^{-2}$) are established for Wikipedia and abstract. For ROUGE, the values are 0.99 ($p < 10^{-8}$) and 0.72 ($p < 10^{-1}$).

Abstract Generation. Lastly, we compare with the state-of-the-art GRAPHWRITER model on AGENDA dataset in Table 5. Although our model does not make use of the relational graph encod-

---

[6]We do not compare with our recent model in Hua et al. (2019) due to the training data difference caused by our new sentence style scheme. However, the newly proposed model generates arguments with lengths closer to human arguments, benefiting from the improved content planning module.

[7]We calculate F1 by aggregating the selections across all sentences. For argument generation, keyphrases are often paraphrased, making it difficult to calculate F1 reliably, therefore omitted here.

	BLEU	ROUGE	METEOR	Length	BLEU	ROUGE	METEOR	Length
	Normal Wikipedia				Simple Wikipedia			
RETRIEVAL	20.10	28.60	12.23	44.5	21.99	33.44	12.97	34.7
SEQ2SEQ	22.62	27.49	14.74	52.9	21.98	29.36	16.94	52.8
LOGREGSEL	29.28	28.65	**27.76**	34.3	5.59	23.21	13.27	13.0
OURS (Oracle Plan.)	37.70*	45.41*	31.65*	79.8	34.22*	45.48*	32.84*	70.5
OURS	**33.76***	**40.08***	25.70	65.4	**31.22***	**40.76***	**26.76***	58.7
w/o Style	31.06*	37.72*	24.56	71.0	27.94*	38.20*	25.87*	64.5

Table 4: Results on Wikipedia generation. Best results without oracle planning are in **bold**. ∗: Our models that are significantly better than all comparisons ($p < 0.001$, approximate randomization test).
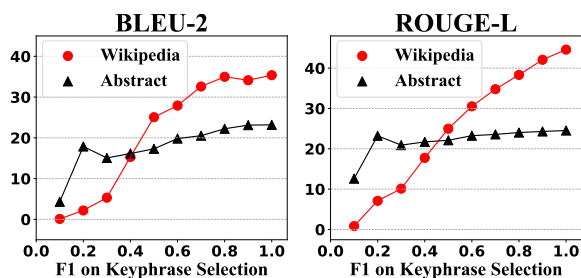

Figure 3: Effect of keyphrase selection (F1 score) on generation performance, measured by BLEU and ROUGE. Positive correlations are observed.

	BLEU	ROUGE	MTR	Len.
GRAPHWRITER	29.95	28.56	19.90	130.1
SEQ2SEQ	18.13	21.03	13.95	134.8
OURS (Oracle Plan.)	25.03	26.18	19.21	125.8
OURS	20.32	23.30	15.95	128.3

Table 5: Results on paper abstract generation. Notice that GRAPHWRITER models rich information about relations and relation types among entities, which is not utilized by our model.

ing, we achieve competitive ROUGE-L and ME-TEOR scores given the oracle plans. Our model also outperforms the seq2seq baseline, which has the same input, indicating the applicability of our method across different domains.

6.2 Human Evaluation

We further ask three proficient English speakers to assess the quality of generated arguments and Wikipedia paragraphs. Human subjects are asked to rate on a scale of 1 (worst) to 5 (best) on **grammaticality**, **correctness** of the text (for arguments, the stance is also considered), and **content richness** (i.e., coverage of relevant points). Detailed guidelines for different ratings are provided to the raters (see Supplementary). For both tasks, we randomly choose 30 samples from the test set; out-

	Argument			Wikipedia		
	Gram.	Corr.	Cont.	Gram.	Corr.	Cont.
HUMAN	4.81	3.90	3.48	4.84	4.73	4.49
OURS	**3.99***	**2.78***	**2.61***	**3.38**	**3.24***	3.43
w/o Style	3.03	2.26	2.03	2.99	2.89	**3.50**
Krippendorff's α	0.75	0.69	0.33	0.70	0.56	0.55

Table 6: Human evaluation on argument generation (Upper) and Wikipedia generation (Bottom). Grammaticality (**Gram**), correctness (**Corr**), and content richness (**Cont**) are rated on Likert scale ($1-5$). We mark our model with ∗ to indicate statistically significantly better ratings over the variant without style specification ($p < 0.001$, approximate randomization test).

puts from two variants of our models and a human written text are presented in random order.

According to Krippendorff's α, the raters achieve substantial agreement on grammaticality and correctness, while the agreement on content richness is only moderate due to its subjectivity. As shown in Table 6, on both tasks, our models with style specification produce more fluent and correct generations, compared to the ones without such information. However, there is still a gap between system generations and human edited text.

We further show **sample outputs** in Figure 4. The first example is on the topic of abortion, our model captures the relevant concepts such as "*fetuses are not fully developed*" and "*illegal to kill*". It also contains fewer repetitions than the seq2seq baseline. For Wikipedia, our model is not only better at controlling the global simplicity style, but also more grammatical and coherent than the seq2seq output.

6.3 Further Analysis and Discussions

We further investigate the usage of different styles, and show the top frequent patterns for each argument style from human arguments and our system generation (Table 7). We first calculate the most

	Human	Our model
C	It doesn't mean that; everyone should be able to	I don't believe that it is necessary; don't need to be able to
P	have the freedom to; is leagal in the US; imagine for a moment if; Let's say (you/your partner/a friend)	have the right to (bear arms/cast a ballot vote); For example, (if you look at/let's look at/I don't think)
F	Why is that?; that's ok; Would it change your mind?	I'm not sure (why/if) this is; TLDR: I don't care about this

Table 7: Top frequent patterns captured in style CLAIM (C), PREMISE (P), and FUNCTIONAL (F) from arguments by human and our model.

frequent 4-grams per style, then extend it with context. We manually cluster and show the representative ones. For both columns, the popular patterns reflect the corresponding discourse functions: CLAIM is more evaluative, PREMISE lists out details, and FUNCTIONAL exhibits argumentative stylistic languages. Interestingly, our model also learns to paraphrase popular patterns, e.g., "*have the freedom to*" vs. "*have the right to*".

For Wikipedia, the sentence style is defined by length. To validate its effect on content selection, we calculate the average number of keyphrases per style type. The results on human written paragraphs are 2.0, 3.8, 5.8, and 9.0 from the simplest to the most complex. A similar trend is observed in our model outputs, which indicates the challenge of content selection in longer sentences.

For future work, improvements are needed in both model design and evaluation. As shown in Figure 4, system arguments appear to overfit on stylistic languages and rarely create novel concepts like humans do. Future work can lead to improved model guidance and training methods, such as reinforcement learning-based explorations, and better evaluation to capture diversity.

7   Conclusion

We present an end-to-end trained neural text generation model that considers sentence-level content planning and style specification to gain better control of text generation quality. Our content planner first identifies salient keyphrases and a proper language style for each sentence, then the realization decoder produces fluent text. We consider three tasks of different domains on persuasive argument generation, paragraph generation for normal and simple versions of Wikipedia, and abstract generation for scientific papers. Ex-

Topic: Aborting a fetus has some non-zero negative moral implications

Human: It's not the birthing process that changes things. It's the existence of the baby. Before birth, the baby only exists inside another human being. After birth, it exists on its own in the world like every other person in the world.

Seq2seq: i 'm not going to try to change your view here , but i do n't want to change your position . i do n't think it 's fair to say that a fetus is not a person . it 's not a matter of consciousness .

Our model: tl ; dr : i agree with you , but i think it 's important to note that fetuses are not fully developed . i do n't know if this is the case , but it does n't seem to be a compelling argument to me at all , so i 'm not going to try to change your view by saying that it should be illegal to kill

Topic: Moon Jae-in

Simple Wikipedia: Moon Jae-in is a South Korean politician. He is the 12th and current President of South Korea since 10 May 2017 after winning the majority vote in the 2017 presidential election.

Seq2seq: moon election park is a election politician who served as prime minister of korea from 2007 to 2013 . he was elected as a member of the house of democratic party in the moon 's the the moon the first serving president of jae-in , in office since 2010 .

Our model: moon jae-in is a south korean politician and current president of south korea from 2012 to 2017 and again from 2014 to 2017.

Normal Wikipedia: Moon Jae-in is a South Korean politician serving as the 19th and current President of South Korea since 2017. He was elected after the impeachment of Park Geun-hye as the candidate of the Democratic Party of Korea.

Seq2seq: moon winning current is a current politician who served as prime minister of korea from 2007 to 2013 . he was elected as a member of the house of democratic party in the moon 's the the current the first president of pakistan , in office . prior to that , he also served on the democratic republic of germany .

Our model: moon jae-in is a south korean politician serving as the 19th and current president of south korea , since 2019 to 2019 and 2019 to 2017 respectively he has been its current president ever since .

Figure 4: Sample outputs for argument generation and Wikipedia generation.

perimental results demonstrate the effectiveness of our model, where it obtains significantly better BLEU, ROUGE, and METEOR scores than nontrivial comparisons. Human subjects also rate our model generations as more grammatical and correct when language style is considered.

References

Waleed Ammar, Dirk Groeneveld, Chandra Bhagavatula, Iz Beltagy, Miles Crawford, Doug Downey, Jason Dunkelberger, Ahmed Elgohary, Sergey Feldman, Vu Ha, Rodney Kinney, Sebastian Kohlmeier, Kyle Lo, Tyler Murray, Hsu-Han Ooi, Matthew Peters, Joanna Power, Sam Skjonsberg, Lucy Wang, Chris Wilhelm, Zheng Yuan, Madeleine van Zuylen, and Oren Etzioni. 2018. Construction of the literature graph in semantic scholar. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, pages 84–91, New Orleans - Louisiana. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Roy Bar-Haim, Indrajit Bhattacharya, Francesco Dinuzzo, Amrita Saha, and Noam Slonim. 2017. Stance classification of context-dependent claims. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 251–261. Association for Computational Linguistics.

Regina Barzilay and Mirella Lapata. 2005. Collective content selection for concept-to-text generation. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 331–338, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Anja Belz. 2008. Automatic generation of weather forecast texts using comprehensive probabilistic generation-space models. *Natural Language Engineering*, 14(4):431–455.

David L Chen and Raymond J Mooney. 2008. Learning to sportscast: a test of grounded language acquisition. In *Proceedings of the 25th international conference on Machine learning*, pages 128–135. ACM.

Andrew Chisholm, Will Radford, and Ben Hachey. 2017. Learning to generate one-sentence biographies from Wikidata. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 633–642, Valencia, Spain. Association for Computational Linguistics.

Emilie Colin, Claire Gardent, Yassine M'rabet, Shashi Narayan, and Laura Perez-Beltrachini. 2016. The WebNLG challenge: Generating text from DBPedia data. In *Proceedings of the 9th International Natural Language Generation conference*, pages 163–167, Edinburgh, UK. Association for Computational Linguistics.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

Pablo Ariel Duboue and Kathleen R. McKeown. 2003. Statistical acquisition of content selection rules for natural language generation. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 121–128.

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.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2018. Findings of the E2E NLG challenge. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 322–328, Tilburg University, The Netherlands. Association for Computational Linguistics.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2019. Evaluating the state-of-the-art of end-to-end natural language generation: The E2E NLG Challenge. *arXiv preprint arXiv:1901.11528*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1019–1027. Curran Associates, Inc.

Eduard H Hovy. 1993. Automated discourse generation using discourse structure relations. *Artificial intelligence*, 63(1-2):341–385.

Xinyu Hua, Zhe Hu, and Lu Wang. 2019. Argument generation with retrieval, planning, and realization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2661–2672, Florence, Italy. Association for Computational Linguistics.

Xinyu Hua and Lu Wang. 2018. Neural argument generation augmented with externally retrieved evidence. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 219–230, Melbourne, Australia. Association for Computational Linguistics.

Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. 2016. Globally coherent text generation with neural

checklist models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 329–339, Austin, Texas. Association for Computational Linguistics.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text Generation from Knowledge Graphs with Graph Transformers. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2284–2293, Minneapolis, Minnesota. Association for Computational Linguistics.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural AMR: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157, Vancouver, Canada. Association for Computational Linguistics.

Ioannis Konstas and Mirella Lapata. 2013. Inducing document plans for concept-to-text generation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1503–1514, Seattle, Washington, USA. Association for Computational Linguistics.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213, Austin, Texas. Association for Computational Linguistics.

Ran Levy, Ben Bogin, Shai Gretz, Ranit Aharonov, and Noam Slonim. 2018. Towards an argumentative content search engine using weak supervision. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2066–2081, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. 2017. Adversarial learning for neural dialogue generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2157–2169, Copenhagen, Denmark. Association for Computational Linguistics.

Percy Liang, Michael Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 91–99, Suntec, Singapore. Association for Computational Linguistics.

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

Chin-Yew Lin and Eduard Hovy. 2000. The automated acquisition of topic signatures for text summarization. In *COLING 2000 Volume 1: The 18th International Conference on Computational Linguistics*.

Marco Lippi and Paolo Torroni. 2016. Argumentation mining: State of the art and emerging trends. *ACM Transactions on Internet Technology (TOIT)*, 16(2):10.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium. Association for Computational Linguistics.

Lara J Martin, Prithviraj Ammanabrolu, Xinyu Wang, William Hancock, Shruti Singh, Brent Harrison, and Mark O Riedl. 2018. Event representations for automated story generation with deep neural nets. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Kathleen R. McKeown. 1985. *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text*. Cambridge University Press, New York, NY, USA.

Hongyuan Mei, Mohit Bansal, and Matthew R. Walter. 2016. What to talk about and how? selective generation using LSTMs with coarse-to-fine alignment. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 720–730, San Diego, California. Association for Computational Linguistics.

George A. Miller. 1994. Wordnet: A lexical database for english. In *HUMAN LANGUAGE TECHNOLOGY: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019. Step-by-step: Separating planning from realization in neural data-to-text generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2267–2277, Minneapolis, Minnesota. Association for Computational Linguistics.

Eric W Noreen. 1989. *Computer-intensive methods for testing hypotheses*. Wiley New York.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Isaac Persing and Vincent Ng. 2016. End-to-end argumentation mining in student essays. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1384–1394, San Diego, California. Association for Computational Linguistics.

Owen Rambow and Tanya Korelsky. 1992. Applied text generation. In *Proceedings of the Third Conference on Applied Natural Language Processing*, pages 40–47, Trento, Italy. Association for Computational Linguistics.

Ehud Reiter and Robert Dale. 2000. *Building applied natural language generation systems*. Cambridge University Press.

Ehud Reiter, Roma Robertson, and Liesl Osman. 2000. Knowledge acquisition for natural language generation. In *INLG'2000 Proceedings of the First International Conference on Natural Language Generation*, pages 217–224, Mitzpe Ramon, Israel. Association for Computational Linguistics.

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*, pages 379–389, Lisbon, Portugal. Association for Computational Linguistics.

Donia Scott and Clarisse Sieckenius de Souza. 1990. Getting the message across in rst-based text generation. *Current research in natural language generation*, 4:47–73.

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.

Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2018. A graph-to-sequence model for AMR-to-text generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1616–1626, Melbourne, Australia. Association for Computational Linguistics.

Matthew Stone and Christine Doran. 1997. Sentence planning as description using tree adjoining grammar. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pages 198–205, Madrid, Spain. Association for Computational Linguistics.

Kumiko Tanaka-Ishii, Kôiti Hasida, and Itsuki Noda. 1998. Reactive content selection in the generation of real-time soccer commentary. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*, pages 1282–1288. Association for Computational Linguistics.

Marilyn A. Walker, Owen Rambow, and Monica Rogati. 2001. SPoT: A trainable sentence planner. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, Copenhagen, Denmark. Association for Computational Linguistics.

Zhiwei Yu, Jiwei Tan, and Xiaojun Wan. 2018. A neural approach to pun generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1650–1660, Melbourne, Australia. Association for Computational Linguistics.