PAIR: Planning and Iterative Refinement in Pre-trained Transformers for Long Text Generation

Xinyu Hua
Khoury College of Computer Sciences
Northeastern University
Boston, MA
huax@northeastern.edu

Lu Wang
Computer Science and Engineering
University of Michigan
Ann Arbor, MI
wangluxy@umich.edu

Abstract

Pre-trained Transformers have enabled impressive breakthroughs in generating long and fluent text, yet their outputs are often “rambling” without coherently arranged content. In this work, we present a novel content-controlled text generation framework, PAIR, with planning and iterative refinement, which is built upon a large model, BART. We first adapt the BERT model to automatically construct the content plans, consisting of keyphrase assignments and their corresponding sentence-level positions. The BART model is employed for generation without modifying its structure. We then propose a refinement algorithm to gradually enhance the generation quality within the sequence-to-sequence framework. Evaluation with automatic metrics shows that adding planning consistently improves the generation quality on three distinct domains, with an average of 20 BLEU points and 12 METEOR points improvements. In addition, human judges rate our system outputs to be more relevant and coherent than comparisons without planning.

1 Introduction

Large pre-trained language models are the cornerstone of many state-of-the-art models in various natural language understanding and generation tasks (Devlin et al., 2019; Liu et al., 2019; Lewis et al., 2020), yet they are far from perfect. In generation tasks, although models like GPT-2 (Radford et al., 2019) are able to produce plausible text, their spontaneous nature limits their utility in actual applications, e.g., users cannot specify what contents to include, and in what order.

To make large models more useful in practice, and to improve their generation quality, we believe it is critical to inform them of when to say what, which is addressed as content planning in traditional generation systems (Duboue and McKeown, 2001; Stent et al., 2004). Specially designed control codes and auxiliary planning modules have been integrated into neural models (Keskar et al., 2019; Morcossef et al., 2019; Hua and Wang, 2019), yet those solutions require model architecture modification or retraining, making text generation with large models a very costly endeavor.

To this end, this work aims to bring new insights into how to effectively incorporate content plans into large models to generate more rele-
vant and coherent text. We first study a planning model trained from BERT (Devlin et al., 2019) to produce the initial content plan, which assigns keyphrases to different sentences and predicts their positions. Next, we propose a content-controlled text generation framework, built upon the pre-trained sequence-to-sequence (seq2seq) Transformer model BART (Lewis et al., 2020). As shown in Figure 1, our generation model takes in a content plan consisting of keyphrase assignments and their corresponding positions for each sentence. The plan is encoded as a template, with masked tokens added at positions where no content is specified. Our model then outputs a fluent and coherent multi-sentence text (draft) to reflect the plan. This is done by fine-tuning BART without modifying its architecture.

Furthermore, we present an iterative refinement algorithm to improve the generation in multiple passes, within the seq2seq framework. At each iteration, tokens with low generation confidence are replaced with masked tokens to compose a new template, from which a new output is produced. Unlike prior refinement algorithms that only permit editing in place, our solution offers more flexibility. Figure 1 exemplifies the refinement outcome.

We call our system PAIR (Planning And Iterative Refinement). It is experimented on three distinct domains: counter-argument generation with Reddit ChangeMyView data, opinion article writing with the New York Times (NYT) corpus (Sandhaus, 2008), and news report production on NYT. Automatic evaluation with BLEU, ROUGE, and METEOR shows that, by informing the generation model with sentence-level content plans, our model significantly outperforms a BART model fine-tuned with the same set of keyphrases as input (§ 5.1). Human judges also rate our system outputs as more relevant and coherent (§ 5.2). Additionally, our iterative refinement strategy consistently improves the generation quality according to both automatic scores and human evaluation. Finally, our model achieves better content control by reflecting the specified keyphrases in the content plan, whose outputs are preferred by human to another version with weaker control.

To summarize, our major contributions include:

- We propose a novel content planner built upon BERT to facilitate long-form text generation.
- We present a novel template mask-and-fill method to incorporate content planning into generation models based on BART.
- We devise an iterative refinement algorithm that works within the seq2seq framework to flexibly improve the generation quality.

2 Related Work

Content Planning as a Generation Component.
Despite the impressive progress made in many generation tasks, neural systems are known to produce low-quality content (Wiseman et al., 2017; Rohrbach et al., 2018), often with low relevance (Li et al., 2016) and poor discourse structure (Zhao et al., 2017; Xu et al., 2020). Consequently, planning modules are designed and added into neural systems to enhance content relevance (Wiseman et al., 2018; Morcos et al., 2019; Yao et al., 2019; Hua and Wang, 2019). However, it is still an open question to include content plans in large models, given the additional and expensive model retraining required. This work innovates by adding content plans as masked templates and designing refinement strategy to further boost generation performance, without architectural change.

Controlled Text Generation.
Our work is also in line with the study of controllability of neural text generation models. This includes manipulating the syntax (Dušek and Jurčiček, 2016; Goyal and Durrett, 2020) and semantics (Wen et al., 2015; Chen et al., 2019) of the output. Specific applications encourage the model to cover a given topic (Wang et al., 2017; See et al., 2019), mention specified entities (Fan et al., 2018), or display a certain attribute (Hu et al., 2017; Luo et al., 2019; Balakrishnan et al., 2019). However, most existing work relies on model engineering, limiting the generalizability to new domains and adaptability to large pre-trained Transformers. One exception is the Plug and Play model (Dathathri et al., 2020), which directly modifies the key and value states of GPT-2 (Radford et al., 2019). However, since the signal is derived from the whole generated text, it is too coarse to provide precise sentence-level content control. Here, we instead gain fine-grained controllability through keyphrase assignment and positioning per sentence, which can be adapted to any off-the-shelf pre-trained Transformer generators.

Iterative Refinement has been studied in machine translation (Lee et al., 2018; Freitag et al., 2019;
Mansimov et al., 2019; Kasai et al., 2020) to gradually improve translation quality. Refinement is also used with masked language models to improve fluency of non-autoregressive generation outputs (Ghazvininejad et al., 2019; Lawrence et al., 2019). Our work uses BART (Lewis et al., 2020), a state-of-the-art seq2seq model that offers better generalizability and stronger capacity for long text generation. Our proposed strategy substantially differs from prior solutions that rely on in-place word substitutions (Novak et al., 2016; Xia et al., 2017; Weston et al., 2018), as we leverage the seq2seq architecture to offer more flexible edits.

3 Content-controlled Text Generation with PAIR

Task Description. Our input consists of (1) a sentence-level prompt \( x \), such as a news headline, or a proposition in an argument, and (2) a set of keyphrases \( m \) that are relevant to the prompt. The system aims to generate \( y \) that contains multiple sentences, as in a news report or an argument, by reflecting the keyphrases in a coherent way.

In this section, we first introduce content planning built upon BERT, that assigns keyphrases into sentences and predicts their positions (§ 3.1). Then we propose a seq2seq generation framework with BART fine-tuning that includes a given content plan derived from keyphrases \( m \) (§ 3.2). Finally, § 3.3 discusses improving generation quality by iteratively masking the less confident predictions and regenerating within our framework.

3.1 Content Planning with BERT

Our content planner is trained from BERT to assign keyphrases to different sentences and predict their corresponding positions. As shown in Figure 2, the concatenation of prompt \( x \) and unordered keyphrases \( m \) is encoded with bidirectional self-attentions. Keyphrase assignments are produced autoregressively as a sequence of tokens \( m' = \{w_j\} \), with their positions in the sentence \( s = \{s_j\} \) predicted as a sequence tagging task.

We choose BERT because it has been shown to be effective at both language modeling and sequence tagging. Moreover, we leverage its segment embedding to distinguish the input and output sequences. Specifically, we reuse its pre-trained language model output layer for keyphrase assignment. We further design a separate keyphrase positioning layer to predict token position \( s_j \) as the relative distance from each sentence’s beginning:

\[
p(s_j | w_{\leq j}) = \text{softmax}(H^L W_s) \tag{1}
\]

where \( H^L \) is the last layer hidden states of the Transformer, and \( W_s \) are the newly added keyphrase positioning parameters learned during BERT fine-tuning. The range of allowed positions is from 0 to 127.

Noticeably, as our prediction is done autoregressively, attentions should only consider the generated tokens, but not the future tokens. However, BERT relies on bidirectional self-attentions to attend to both left and right. To resolve this discrepancy, we apply causal attention masks (Dong et al., 2019) over \( m' \) to disallow attending to the future (gray arrows in Figure 2).

Training the Planner. We extract keyphrases and acquire their ground-truth positions from human-written references, and fine-tune BERT with cross-entropy losses for both assignment and positioning, with a scaling factor 0.1 over the positioning loss.

Inference. A [BOK] token signals the beginning of keyphrase assignment generation. We employ a greedy decoding algorithm, and limit the output vocabulary to tokens in \( m \) and ensure each keyphrase is generated at most once. To allow sentence-level content planning, a special [SEN] token is generated to represent the sentence boundary, with its predicted position indicating the length. The planning process terminates when [EOS] is produced.
3.2 Adding Content Plan with a Template

Mask-and-Fill Procedure

Given a content planning model, we invoke it to output keyphrase assignments to different sentences \((m')\), their corresponding positions \(s\), along with each sentence’s length (based on the prediction of \([\text{SEN}]\)). We first employ a post-processing step to convert between different tokenizers, and correct erroneous position predictions that violate the assignment ordering or break the consecutivity of the phrase (Appendix A). We then convert the plan into a template \(t^{(0)}\) as follows: For each sentence, the assigned keyphrases are placed at their predicted positions, and empty slots are filled with \([\text{MASK}]\) symbols. Figure 3 illustrates the template construction process and our seq2seq generation model. In Appendix B, we show statistics on the constructed templates.

The input prompt \(x\), keyphrase assignments \(m'\), and template \(t^{(0)}\) are concatenated as the input to the encoder. The decoder then generates an output \(y^{(1)}\) according to the model’s estimation of \(p(y^{(1)} | x, m', t^{(0)})\). \(y^{(1)}\) is treated as a draft, to be further refined as described in the next section.

Our method is substantially different from prior work that uses constrained decoding to enforce words to appear at specific positions (Hokamp and Liu, 2017; Post and Vilar, 2018; Hu et al., 2019), which is highly biased by the surrounding few words and suffers from disfluency. Since BART is trained to denoise the masked input with contextual symbols. Figure 3 illustrates the template construction process and our seq2seq generation model. In Appendix B, we show statistics on the constructed templates.

Decoding. We employ the nucleus sampling strategy (Holtzman et al., 2019), which is shown to yield superior output quality in long text generation. In addition to the standard top-k sampling from tokens with the highest probabilities, nucleus sampling further limits possible choices based on a cumulative probability threshold (set to 0.9 in all experiments below). We also require the keyphrases to be generated at or nearby their predicted positions. Concretely, for positions that match any keyphrase token, we force the decoder to copy the keyphrase unless it has already been generated in the previous five tokens. We sample three times to choose the one with the lowest perplexity, as estimated by GPT-2\(_{\text{base}}\) (Radford et al., 2019).

3.3 Iterative Refinement

Outputs generated in a single pass may suffer from incorrectness and incoherence (see Figure 1), therefore we propose an iterative refinement procedure to improve the quality. In each pass, tokens with low generation confidence are masked (Algorithm 1). This is inspired by iterative decoding designed for inference acceleration in non-autoregressive generation (Lee et al., 2018; Lawrence et al., 2019), though their refinement mostly focuses on word substitution and lacks the flexibility for other operations. Moreover, our goal is to improve fluency while ensuring the generation of given keyphrases.

At each iteration, the \(n\) least confident tokens are replaced with \([\text{MASK}]\). Similar as the mask-predict algorithm (Ghazvininejad et al., 2019), we gradually reduce the number of masks. In our experiments, each sample is refined for 5 iterations, with \(n\) decaying linearly from 80% of \(|y^{(r)}|\) to 0.

Training the Generator. Our training scheme is similar to masked language model pre-training. Given the training corpus \(D = \{(x_i, m'_i, y_i)\}\), we consider two approaches that add noise to the target \(y_i\) by randomly masking a subset of (1) any tokens, or (2) tokens that are not within the span...
We evaluate our generation and planning models (Hua et al., 2019). This dataset contains material-type opinions (e.g., the minimum wage should be abolished), to form the input prompt $x$. In our prior work, only the first paragraphs of high-quality counter-arguments are used for generation. Here we consider generating the full post, which is significantly longer. Keyphrases are identified as noun phrases and verb phrases that contain at least one topic signature word (Lin and Hovy, 2000), which is determined by a log-likelihood ratio test that indicates word salience. Following our prior work, we expand the set of topic signatures with their synonyms, hyponyms, hypernyms, and antonyms according to WordNet (Miller, 1994). The keyphrases longer than 10 tokens are further discarded.

**Task 2: Opinion Article Generation.** We collect opinion articles from the New York Times (NYT) corpus (Sandhaus, 2008). An article is selected if its taxonomies label has a prefix of Top/Opinion. We eliminate articles with an empty headline or less than three sentences. Keyphrases are extracted in a similar manner as done in argument generation. Samples without any keyphrase are removed. The article headline is treated as the input, and our target is to construct the full article. Table 1 shows that opinion samples have shorter input than arguments, and the keyphrase set also covers fewer content words in target covered by the keyphrases (KP Cov.).

**4 Experiment Setups**

**4.1 Tasks and Datasets**

We evaluate our generation and planning models on datasets from three distinct domains for multi-paragraph-level text generation: (1) argument generation (ARGGEN) (Hua et al., 2019), to produce a counter-argument to refute a given proposition; (2) writing opinionated articles (OPINION), e.g., editorials and op-eds, to show idea exchange on a given subject; and (3) composing news reports (NEWS) to describe events. The three domains are selected with diverse levels of subjectivity and various communicative goals (persuading vs. informing), with statistics shown in Table 1.

**Task 1: Argument Generation.** We first evaluate our models on persuasive argument generation, based on a dataset collected from Reddit r/ChangeMyView (CMV) in our prior work (Hua et al., 2019). This dataset contains pairs of original post (OP) statement on a controversial issue about politics and filtered high-quality counter-arguments, covering 14,833 threads from 2013 to 2018. We use the OP title, which contains a proposition (e.g. the minimum wage should be abolished), to form the input prompt $x$. In our prior work, only the first paragraphs of high-quality counter-arguments are used for generation. Here we consider generating the full post, which is significantly longer. Keyphrases are identified as noun phrases and verb phrases that contain at least one topic signature word (Lin and Hovy, 2000), which is determined by a log-likelihood ratio test that indicates word salience. Following our prior work, we expand the set of topic signatures with their synonyms, hyponyms, hypernyms, and antonyms according to WordNet (Miller, 1994). The keyphrases longer than 10 tokens are further discarded.

**Table 1:** Statistics of the three datasets. We report average lengths of the prompt and the target generation, number of unique keyphrases (# KP) used in the input, and the percentage of content words in target covered by the keyphrases (KP Cov.).

<table>
<thead>
<tr>
<th>Domain</th>
<th># Sample</th>
<th>Prompt</th>
<th>Target</th>
<th># KP</th>
<th>KP Cov.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARGGEN</td>
<td>56,504</td>
<td>19.4</td>
<td>116.6</td>
<td>20.6</td>
<td>30.5%</td>
</tr>
<tr>
<td>OPINION</td>
<td>104,610</td>
<td>6.1</td>
<td>205.6</td>
<td>19.0</td>
<td>26.0%</td>
</tr>
<tr>
<td>NEWS</td>
<td>239,959</td>
<td>7.0</td>
<td>282.7</td>
<td>30.3</td>
<td>32.6%</td>
</tr>
</tbody>
</table>

[Algorithm 1: Iteratively refinement via template mask-and-fill. The sample with the lowest perplexity (thus with better fluency) is selected for each iteration.]
We consider two baselines, both are fine-tuned WordPiece (Wu et al., 2016) for BERT-based planning and avoid overfitting. More details are in Appendix A.

Table 2: Key results on argument generation, opinion article writing, and news report generation. BLEU-4 (B-4), ROUGE-L (R-L), METEOR (MTR), and average output lengths are reported (for references, the lengths are 100, 166, and 250, respectively). PAIRlight, using keyphrase assignments only, consistently outperforms baselines; adding keyphrase positions, PAIRfull further boosts scores. Improvements by our models over baselines are all significant ($p < 0.0001$, approximate randomization test). Iterative refinement helps on both setups.

<table>
<thead>
<tr>
<th></th>
<th>ARGGEN</th>
<th></th>
<th></th>
<th>OPINION</th>
<th></th>
<th></th>
<th>NEWS</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>B-4</td>
<td>R-L</td>
<td>MTR</td>
<td>Len.</td>
<td>B-4</td>
<td>R-L</td>
<td>MTR</td>
<td>Len.</td>
<td>B-4</td>
</tr>
<tr>
<td>seq2seq</td>
<td>0.76</td>
<td>13.80</td>
<td>9.36</td>
<td>97</td>
<td>1.42</td>
<td>15.97</td>
<td>10.97</td>
<td>156</td>
<td>1.11</td>
</tr>
<tr>
<td>kpseq2seq</td>
<td>6.78</td>
<td>19.43</td>
<td>15.98</td>
<td>97</td>
<td>11.38</td>
<td>22.75</td>
<td>18.38</td>
<td>164</td>
<td>11.61</td>
</tr>
<tr>
<td>Pairlight w/o refine</td>
<td>25.17</td>
<td>46.84</td>
<td>31.31</td>
<td>120</td>
<td>15.45</td>
<td>32.35</td>
<td>24.11</td>
<td>214</td>
<td>27.32</td>
</tr>
<tr>
<td>Pairfull w/o refine</td>
<td>34.09</td>
<td>55.42</td>
<td>32.74</td>
<td>101</td>
<td>22.17</td>
<td>39.71</td>
<td>24.65</td>
<td>169</td>
<td>33.48</td>
</tr>
<tr>
<td>Pairfull</td>
<td>36.09</td>
<td>56.86</td>
<td>33.30</td>
<td>102</td>
<td>23.12</td>
<td>40.53</td>
<td>24.73</td>
<td>167</td>
<td>34.37</td>
</tr>
</tbody>
</table>

Table 2: Key results on argument generation, opinion article writing, and news report generation. BLEU-4 (B-4), ROUGE-L (R-L), METEOR (MTR), and average output lengths are reported (for references, the lengths are 100, 166, and 250, respectively). PAIRlight, using keyphrase assignments only, consistently outperforms baselines; adding keyphrase positions, PAIRfull further boosts scores. Improvements by our models over baselines are all significant ($p < 0.0001$, approximate randomization test). Iterative refinement helps on both setups.

Data Split and Preprocessing. For argument generation, we split the data into 75%, 12.5%, and 12.5% for training, validation, and test sets. To avoid test set contamination, the split is conducted on thread level. For opinion and news generation, we reserve the most recent 5k articles for testing, another 5k for validation, and the rest (23k for news and 10k for opinion) are used for training. We apply the BPE tokenization (Sennrich et al., 2016) for the generation model as BART does, and use WordPiece (Wu et al., 2016) for BERT-based planner. To fit the data into our GPUs, we truncate the target size to 140 tokens for argument, sizes of 243 and 335 are applied for opinion and news, for both training and inference.

4.2 Implementation Details
Our code is written in PyTorch (Paszke et al., 2019). For fine-tuning, we adopt the standard linear warmup and inverse square root decaying scheme for learning rates, with a maximum value of $5 \times 10^{-5}$. Adam (Kingma and Ba, 2014) is used as the optimizer, with a batch size of 10 for refinement and 20 for content planning, and a maximum gradient clipped at 1.0. All hyperparameters are tuned on validation set, with early stopping used to avoid overfitting. More details are in Appendix A.

4.3 Baselines and Comparisons
We consider two baselines, both are fine-tuned from BART as in our models: (1) SEQ2SEQ directly generates the target from the prompt; (2) KPSEQ2SEQ encodes the concatenation of the prompt and the unordered keyphrase set. To study if using only sentence-level keyphrase assignments helps, we include a model variant (PAIRlight) by removing keyphrase position information ($s$) from the input of our generator and using an initial template with all [MASK] symbols. Our model with full plans is denoted as PAIRfull. We first report generation results using ground-truth content plans constructed from human-written text, and also show the end-to-end results with predicted content plans by our planner.

5 Results
5.1 Automatic Evaluation
We report scores with BLEU (Papineni et al., 2002), which is based on n-gram precision (up to 4-grams); ROUGE-L (Lin, 2004), measuring recall of the longest common subsequences; and METEOR (Lavie and Agarwal, 2007), which accounts for paraphrase. For our models PAIRfull and PAIRlight, we evaluate both the first draft and the final output after refinement. Table 2 lists the results when ground-truth content plans are applied.

First, our content-controlled generation model with planning consistently outperforms comparisons and other model variants on all datasets, with or without iterative refinement. Among our model variants, PAIRfull that has access to full content plans obtains significantly better scores than PAIRlight that only includes keyphrase assignments but not their positions. Lengths of PAIRfull’s outputs are also closer to those of human references. Both imply the benefit of keyphrase positioning.

Table 2 also shows that the iterative refinement strategy can steadily boost performance on both of our setups. By inspecting the performance of refinement in different iterations (Figure 4), we observe that both BLEU and ROUGE-L scores gradually increase while perplexity lowers as the
refinement progresses. This indicates that iterative post-editing improves both content and fluency.

**Results with Predicted Content Plans.** We further report results by using content plans predicted by our BERT-based planner. Figure 5 compares PAIR\textsubscript{full} and PAIR\textsubscript{light} with KPSEQ\textsubscript{2SEQ}. Our models yield better METEOR scores on all three domains. That said, the improvement from predicted plans is not as pronounced as that from ground-truth plans. Upon inspection, we find that our planner often falls short of accurately positioning the given keyphrases, leading to degraded generation performance. This points to a potential direction for future work where better positioning model should be developed.

### 5.2 Human Evaluation

We hire four proficient English speakers\textsuperscript{3} to rate three aspects of the generated arguments on a scale of 1 (worst) to 5 (best): fluency, coherence—if the information organization is natural and logical, and relevance—if the topic is related to the prompt and whether the stance is correct. 50 samples are randomly selected, with system outputs by KPSEQ\textsubscript{2SEQ}, PAIR\textsubscript{full} and PAIR\textsubscript{light} shown to human judges in random order. The evaluation guideline is in the supplementary material.

Table 3 shows that both of our models are rated with better coherence and relevance than KPSEQ\textsubscript{2SEQ} which uses the same but unordered keyphrases as input. Interestingly, outputs by PAIR\textsubscript{light} are regarded as more fluent and coherent, though the difference is not significant. However, discourse analysis in § 6 reveals that clauses produced by PAIR\textsubscript{light} are more locally related, compared to PAIR\textsubscript{full}, which can be perceived as easier to read. In addition to the sample argument in Figure 1, Table 4 shows PAIR\textsubscript{full}’s output in the news and opinion domains. More samples by different systems are in the supplementary material.

Table 3: Human evaluation for argument generation on fluency, coherence, and relevance, with 5 as the best. The Krippendorff’s α are 0.28, 0.30, and 0.37, respectively. Our model outputs are significantly more coherent and relevant than KPSEQ\textsubscript{2SEQ} (*: p < 0.0001), with comparable fluency.

**Prompt (News): 4 Arrested in Theft of Baby Jesus Figurines**

PAIR\textsubscript{full}: Four New Jersey teenagers arrested yesterday were accused of stealing more than 25 plastic baby Jesus figurines from a church before they burn in a bonfire, the police said. The police in Sayreville, N.J., arrested Michael Payne, 18, and T.J. Jones, 18, of Union City; Nicholas M. Hess, 18, of Matawan; and Jason L. O’Neill, 18, of Port New York, N.J., and charged them with burglary. Their vandals removed more than 100 figurines of Jesus from a cemetery outside St. Stanislaus Kostka Church in Sayreville, the police said. Detectives said the four had taken their flashlights to the cemetery and jotted down the license plates of several cars in the vicinity […]

**Prompt (Opinion): Drowning in a Sea of College Debt**

PAIR\textsubscript{full}: Once again, the appalling problem of increasing debt in financing higher education that has taken place is more intolerable, particularly because of the arrogance of our colleges and universities. The burden of saddling college students with debt in the middle of their teenage years, when they were in debt, is essential for a good education. Our educational system is designed to allow kids to develop the skills necessary, but it does not create optimal conditions for mature students who know they will not be able […]

Table 4: Sample outputs in the news and opinion domain. Keyphrases assigned to different sentences are in boldface and color-coded.

---

\textsuperscript{3}They are all US-based college students. Each of them is paid $15 hourly for the task.
ment. We further ask whether human judges prefer the refined text and whether enforcing keyphrases to be generated yields noticeable content improvement. In a second study, we present the same 50 prompts from the previous evaluation on argument generation, and an additional 50 samples for opinion article writing to the same group of human judge. For each sample, PAIRfull’s outputs with and without refinement are shown in random order. Judges indicate their preference based on the overall quality. The same procedure is conducted to compare with a version where we do not enforce keyphrases to be copied at their predicted positions during decoding. Table 5 demonstrates that the refined text is preferred in more than half of the cases, for both domains. Enforcing keyphrase generation based on their positions is also more favorable than not enforcing such constraint.

<table>
<thead>
<tr>
<th></th>
<th>ArgGen</th>
<th>Opinion</th>
<th>News</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAIRfull w/o refine</td>
<td>52.7%</td>
<td>52.7%</td>
<td>50.0%</td>
</tr>
<tr>
<td>PAIRfull w/ enforce</td>
<td>45.3%</td>
<td>30.7%</td>
<td>29.3%</td>
</tr>
<tr>
<td>ArgGen</td>
<td>33.3%</td>
<td>40.0%</td>
<td></td>
</tr>
<tr>
<td>Opinion</td>
<td>30.7%</td>
<td>29.3%</td>
<td></td>
</tr>
<tr>
<td>News</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5: Percentages of samples preferred by human judges before and after refinement [Left]; with and without enforcing keyphrases to appear at the predicted positions [Right]. Ties are omitted.

What is updated during iterative refinement? Since refinement yields better text, we compare generations before and after the refinement. First, we find that masks are regularly put on “functional” words and phrases. For example, stopwords and punctuation along with their bigrams are often swapped out, with new words filled in to improve fluency. Moreover, about 85% of the refinement operations result in new content being generated. This includes changing prepositions and paraphrasing, e.g., replacing “a research fellow” with “a graduate student.” On both news and opinion domains, numerical and temporal expressions are often incorrectly substituted, suggesting that better fact control needs to be designed to maintain factuality.

6 Further Discussions on Discourse

Prior work’s evaluation mainly focuses on fluency and content relevance, and largely ignores the discourse structure exposed by the generated text. However, unnatural discourse and lack of focus are indeed perceived as major problems of long-form neural generations, as identified by human experts. Here, we aim to investigate whether content-controlled generation with ground-truth content plans resembles human-written text by studying discourse phenomena.

Are PAIR generations similar to human-written text in discourse structure? We utilize DPLP (Ji and Eisenstein, 2014), an off-the-shelf Rhetorical Structure Theory (RST) discourse parser. DPLP converts a given text into a binary tree, with elementary discourse units (EDUs, usually clauses) as nucleus and satellite nodes. For instance, a relation NS-elaboration indicates the second node as a satellite (S) elaborating on the first nucleus (N) node. DPLP achieves F1 scores of 81.6 for EDU detection and 71.0 for relation prediction on news articles from the annotated RST Discourse Treebank (Carlson et al., 2001). We run this trained model on our data for both human references and model generations.

First, we analyze the depth of RST parse trees, which exhibits whether the text is more locally or globally connected. For all trees, we truncate at a maximum number of EDUs based on the 90 percentile of EDU count for human references. Distributions of tree depth are displayed in Figure 6. As can be seen, generations by PAIRfull show similar patterns to human-written arguments and articles. We also find that trees by PAIRlight tend to have a more “linear” structure, highlighting the dominance of local relations between adjacent EDUs, compared with PAIRfull which uses knowledge of keyphrases positions. This implies that content positioning helps with structure at a more global level. We further look into the ratios of NS, NN, SN relations, and observe that most model outputs have similar trends as human-written texts, except for KPSeq2Seq which has more SN relations, e.g., it produces twice as many SNs than others on arguments.

Can PAIR correctly generate discourse markers? Since discourse markers are crucial for coherence (Grote and Stede, 1998; Callaway, 2003) and have received dedicated research efforts in rule-based systems (Reed et al., 2018; Balakrishnan et al., 2019), we examine if PAIR\textsubscript{full} can properly generate them. For each sample, we construct sentence pairs based on content word overlaps between system generation and human reference. We manually select a set of unambiguous discourse markers from Appendix A of the Penn Discourse Treebank manual (Prasad et al., 2008). When a marker is present in the first three words in a reference sentence, we check if the corresponding system output does the same.

Figure 7 displays the numbers of generated sentences with markers produced as the same in human references (correct) or not (wrong). The markers are grouped into three senses: CONTINGENCY, COMPARISON, and EXPANSION. The charts indicates that PAIR\textsubscript{full} does better at reproducing markers for CONTINGENCY, followed by COMPARISON and EXPANSION. Manual inspections show that certain missed cases are in fact plausible replacements, such as using at the same time for in addition or also for further, while in other cases the markers tend to be omitted. Overall, we believe that content control alone is still insufficient to capture discourse relations, motivating future work on discourse planning.

7 Ethics Statement

We recognize that the proposed system can generate fabricated and inaccurate information due to the systematic biases introduced during model pretraining based on web corpora. We urge the users to cautiously examine the ethical implications of the generated output in real world applications.

8 Conclusion

We present a novel content-controlled generation framework that adds content planning to large pretrained Transformers without modifying model architecture. A BERT-based planning model is first designed to assign and position keyphrases into different sentences. We then investigate an iterative refinement algorithm that works with the sequence-to-sequence models to improve generation quality with flexible editing. Both automatic evaluation and human judgments show that our model with planning and refinement enhances the relevance and coherence of the generated content.

Acknowledgements

This research is supported in part by National Science Foundation through Grant IIS-1813341 and Nvidia GPU gifts. We thank three anonymous reviewers for their constructive suggestions on many aspects of this work.

References


A Reproducibility

Computing Infrastructure. Our model is built upon the PyTorch transformers-2.6.0 library by Wolf et al. (2019), with Pytorch-Lightning-0.7.3 (Falcon, 2019) for training routines. To improve training efficiency, we adopt mixed-precision floating point (FP16) computation using the O2 option of NVIDIA apex5. For both training and decoding, we utilize the Titan RTX GPU card with 24 GB memory.

Model Sizes. Our generation model has the same architecture as BART (Lewis et al., 2020) with 406M parameters. The content planner is built on top of BERTbase, which has 110M parameters.

Running Time. Training the generation model takes 2.5 hours for argument, 5 hours for opinion, and 24 hours for news. The content planning model converges in 2.5-4 hours for three domains.

Decoding Settings. At inference time, we set \( k = 50, \) temperature=1.0, and \( p = 0.9 \) for nucleus sampling. The relatively large \( k \) value is determined based on a pilot study, where we find that the refinement lacks diversity if \( k \) is set to small values. Moreover, since the Transformer states need to be cached during autoregressive decoding and we perform three complete nucleus sampling runs in each refinement iteration, the GPU memory consumption is substantially increased. We therefore limit the maximum generation steps to 140 for argument, 243 and 335 for opinion and news.

Auto-Correction for Content Plan. When the content plan is predicted by the planner, the following post-processing steps are employed prior to the

---

5https://github.com/NVIDIA/apex

### Table 6: Statistics on generated templates by our content planner. Tokens are measured in units of WordPiece (Sennrich et al., 2016). KP distance denotes the average number of tokens between two keyphrases that are in the same sentence. Both system output (sys) and human reference (ref) are reported.

<table>
<thead>
<tr>
<th></th>
<th>ARGGEN</th>
<th>OPINION</th>
<th>NEWS</th>
</tr>
</thead>
<tbody>
<tr>
<td># tokens</td>
<td>133.3</td>
<td>130.2</td>
<td>228.5</td>
</tr>
<tr>
<td># sentences</td>
<td>8.6</td>
<td>5.6</td>
<td>11.1</td>
</tr>
<tr>
<td># KP per sent.</td>
<td>2.96</td>
<td>3.77</td>
<td>2.22</td>
</tr>
<tr>
<td>KP distance</td>
<td>2.61</td>
<td>2.95</td>
<td>5.70</td>
</tr>
</tbody>
</table>

masked template construction: (1) For a predicted keyphrase, its token positions are adjusted to a consecutive segment, so that the phrase is kept intact in the template. (2) If the predicted positions are not monotonic to the assignment ordering, they will be rearranged. For instance, if the assignment contains \( KP_1 \succ KP_2 \), but position of \( KP_2 \) is not strictly larger than that of \( KP_1 \), we instead place \( KP_2 \) immediately after \( KP_1 \) in the template. (3) Finally, since the planner and generator have different subword vocabularies, it is necessary to detokenize the predicted keyphrase assignment, and re-tokenize with the BPE vocabulary of the generator.

B Template Construction Statistics

We characterize the content planning results in Table 6. Specifically, we show the statistics on the automatically created templates based on the planner’s output. As we can see, our system predicted templates approach human reference in terms of length, per sentence keyphrase count, and the average keyphrase spacing. Sentence segmentation occurs more often in our templates than the reference text, likely due to the frequent generation of [SEN] tokens.