Plot Writing From Scratch Pre-Trained Language Models

Anonymous ACL submission

Abstract

Pre-trained language models (PLMs) fail to generate long-form narrative text because they do not consider global structure. As a result, the generated texts are often incohesive, repetitive, or lack content. Recent work in story generation reintroduced explicit content planning in the form of prompts, keywords, or semantic frames. Trained on large parallel corpora, these models can generate more logical event sequences and thus more contentful stories. However, these intermediate representations are often not in natural language and cannot be utilized by PLMs without fine-tuning. We propose generating story plots using off-the-shelf PLMs while maintaining the benefit of content planning to generate cohesive and contentful stories. Our proposed method, SCRATCHPLOT, first prompts a PLM to compose a content plan. Then, we generate the story’s body and ending conditioned on the content plan. Furthermore, we take a generate-and-rank approach by using additional PLMs to rank the generated (story, ending) pairs. We benchmark our method with various baselines and achieved superior results in both human and automatic evaluation.\(^1\)

1 Introduction

Long-form story generation is challenging because language models lack global planning (Hua and Wang, 2020; Tan et al., 2021), discourse coherence (Bosselut et al., 2018; Ji and Huang, 2021), and common sense knowledge (Xu et al., 2020; Ji et al., 2020). While individual sentences appear fluent and logical, they do not fit together as a whole and the stories often have no clear content (See et al., 2019; Goldfarb-Tarrant et al., 2020).

Interestingly, recent work in long-form story generation relied on explicit content planning (Reiter and Dale, 1997), contrary to the prevalent trend of end-to-end learning across NLP tasks. The content plan usually takes the form of prompts (Fan et al., 2018), keywords/keyphrases (Xu et al., 2018; Yao et al., 2019), semantic frames (Fan et al., 2019), or summaries (Sun et al., 2020).

These content plans are usually not in the form of natural language\(^2\) and cannot be understood by pre-trained language models (PLMs) without fine-tuning using parallel data. Another subtle problem of modeling story generation as a supervised learning task is that the model learns common sense and frequently occurring action sequences, like morning routines (Fan et al., 2019). Such an action plan may not be interesting and surprising, which are crucial characteristics of stories.

We propose generating stories using off-the-

\(^1\)Code and data available at https://github.com/anonymized-for-review.

\(^2\)Except for using summaries as the content plan.
shelf PLMs without fine-tuning. We tap on DINO (Schick and Schütze, 2021), a framework
to generate datasets using instructions, to compose
stories progressively. Our method, ScratchPlot, is
depicted in Figure 1. We firstly prompt a PLM to
perform content planning, including the location,
characters, genre, and theme. We then generate a
story conditioned on these attributes. Finally, we
generate story endings and rank them.

2 Related Work

Fan et al. (2018) introduced hierarchical story gen-
eration by first generating a prompt, then trans-
forming it into a full story. They introduced a
novel fusion-based architecture to improve the
relevance between the generation and the input
prompt. Subsequent works used sequences of key-
words (Yao et al., 2019; Rashkin et al., 2020),
semantic frames (Fan et al., 2019), or paragraph
summaries (Sun et al., 2020) as the content plan
because they have a higher capacity than a sin-
gle prompt. See et al. (2019) compared Fan et al.
(2018) with a fine-tuned GPT-2 model (Radford
et al., 2019) and showed that GPT-2 writes better
stories and conditions on the prompt more strongly.

Recently, Tan et al. (2021) proposed ProGen,
a multi-stage seq2seq model, by extracting key-
words at different granularities. Each stage takes
the output from the previous stage and adds finer-
grained details. Unlike previous works, we use heterogenous plot elements sampled from a PLM
as the content plan. We also do not require any
fine-tuning and rely solely on off-the-shelf PLMs.

3 Plot Generation From Scratch

DINO (Schick and Schütze, 2021) is a framework
to generate labeled NLI datasets (Bowman et al.,
2015) using a pre-trained GPT-2 model (Radford
et al., 2019). Schick and Schütze (2021) formu-
lated different task descriptions to generate sen-
tence pairs for each category. Instead of generating
the sentence pairs at once, they sample the first
sentence \(x_1\), then incorporate it into the task de-
scription to sample the second sentence, as shown
in Figure 2.

Factorizing Elements of a Story We factor-
ize story generation into multiple stages analogous
to generating NLI sentence pairs. We define four
main plot elements: location, characters, genre, and
theme. These elements are not entirely inde-
pendent. For example, the genre will influence the

Figure 2: Task description to generate a similar sen-
tence by incorporating the first generated sentence, \(x_1\).

Figure 3: Task description for generating genre and
theme. \(<X_1>\) denotes the generated genre. The exam-
ple continuations are generated by GPT2-XL.

We sample one value for each plot element ex-
cept for characters, where we generate a male and a
female character. After sampling all plot elements,
we fuse them into a single task description to gen-
erate the story, as depicted in Step 2 of Figure 1.

Generating Coherent Story Ending A co-
herent and thoughtful ending is crucial to stories.
However, it is not obvious how to write the story
ending with PLMs. One challenge is that GPT-2
does not have an \(<\text{EOS}>\) (end-of-sentence) token.
Schick and Schütze (2021) always end the task de-
scription with an open quotation mark and treat the
first quotation mark generated by the PLM as the
\(<\text{EOS}>\) token. However, the PLM usually generates
the quotation mark after a couple of sentences, mak-
ing it unsuitable for generating long-form stories.
Therefore, we ignore the artificial \(<\text{EOS}>\) token
and generate the story with a fixed length. Then,
we truncate it till the last complete sentence.

We design a separate task description for story
ending generation by providing the story body and
asking the PLM to write what happens in the end.
As the story ending is usually short, we treat the
first quotation mark as the \(<\text{EOS}>\) token.
We observe that PLMs sometimes ignore the task descriptions and write generic or irrelevant story endings. Therefore, we propose two methods to rank the story endings. Firstly, we use the next sentence prediction (NSP) task of BERT (Devlin et al., 2019) to measure the coherence between the story and the ending. Specifically, we calculate $P_{NSP}(b,e)$, where $b$ denotes the story body and $e$ denotes the story ending.

Inspired by previous works in fact-checking with PLMs (Lee et al., 2020, 2021), we use the perplexity score as another metric for the story ending’s quality. Specifically, we concatenate the story body and ending to form the input to the PLM: $X = \{x_{b0}, ..., x_{bB}, x_{e0}, ..., x_{eE}\}$, where $B$ and $E$ denote the number of tokens in the story body and ending separately. We then calculate the conditioned perplexity by

$$PPL(X) = \frac{1}{Z} \prod_{i=1}^{E} \frac{1}{p(x_{ei}|x_{b0}, ..., x_{bB}, ..., x_{ei-1})}$$

Note that we use the story body tokens to condition the perplexity, but they do not contribute to the $PPL(X)$.

We hypothesize that NSP and PPL attend to different aspects. NSP performs sentence-level classification and may pay attention to discourse markers or lexical overlap to determine whether a sentence is a valid continuation of another sentence. On the other hand, PPL aggregates scores for individual tokens and measures the intrinsic quality of the story ending.

We sample multiple story bodies and their corresponding endings and rank them using NSP and PPL separately. We select the story body/ending pair with the lowest mean rank as the final output.

4 Experiments

Experimental Details We use the official implementation of DINO (Schick and Schütze, 2021) with the default GPT2-XL language model. We follow the default parameters except setting $k=30$ for top-$k$ sampling and blocking repeating trigrams during generation. We use self-debiasing (Schick et al., 2021) to differentiate different geographical units and male/female names. The rest of the generations use a vanilla PLM without self-debiasing.

For story ending ranking, we use HuggingFace (Wolf et al., 2020) bert-base-uncased checkpoint to calculate the NSP probabilities and gpt2 (base) to calculate the perplexity.

We perform simple post-processing to clean or filter the continuations, such as removing tailing punctuations and filtering continuations that repeat words from the prompt or contain 1st or 2nd person pronouns. The story body must also contain some plot elements to ensure it is contentful and respects the task description. The post-processing is detailed in Appendix B.

Baselines We compare with two conditional story generation baselines. The first one is a Fusion model with a convolutional encoder and a self-attention decoder (Fan et al., 2018). The second model is ProGen (Tan et al., 2021), a multi-stage seq2seq model using salient keywords as intermediate representations. We provide the same generated content plans to the baselines to make the comparison fair. We use the generated theme as input to the Fusion model, which is analogous to the prompt. On the other hand, we extract keywords using TF-IDF following Tan et al. (2021) from all plot elements to prepare the input to ProGen. We also experiment with a baseline GPT2-XL without content planning where we sample a list of stories by providing the instruction “Task: Write a plot summary.”

Plot summary:” We limit the story length to 150 tokens in all models for ease of human evaluation.

RQ1: How does SCRATCHPlot compare to the baselines? We generate 50 stories using each model and invite three crowdworkers to evaluate each story on the following fine-grained aspects: naturalness, interestingness, and cohesiveness. We take the average of the scores assigned by the annotators as the final score. Appendix E provides full details of the crowdsourcing evaluation.

Table 1 overviews the result. SCRATCHPlot outperformed all baselines by a large margin. We notice that the Fusion model tends to generate common narratives, thus receiving the lowest interestingness score. ProGen sometimes generates ungrammatical text, likely due to its intermediate content representation, which is a list of keywords.

---

1NSP the higher, the better. PPL($X$) the lower, the better.
2https://github.com/tanyuqian/progressive-generation
3Additional experimental details can be found in Appendix C.
4https://github.com/pytorch/fairseq/tree/main/examples/stories
5https://github.com/tanyuqian/progressive-generation
6https://github.com/tanyuqian/CRATCH
Table 1: Human evaluation result of various models on different aspects. The columns denote naturalness, interestingness, cohesiveness. All scores are on a scale from 1 (worst) to 5 (best). Best scores for each aspect are highlighted in bold. * indicates statistical significance compared with the second best system using two-sided paired t-test with $p=0.01$.

Table 9 shows a randomly sampled content plan with stories generated by each model.

**RQ2: Does unsupervised content planning help?** Different from previous work in story content planning, SCRATCHPLOT is entirely unsupervised, relying only on task descriptions. Table 2 presents the average expert rating for each generated plot element. As we can see, the PLM generates simple fields like locations and person names with high quality. However, some generated themes are ambiguous or nonsensical.

Table 2: Expert rated scores for generated plot elements normalized to the range of 0 (worst) to 1 (perfect).

SCRATCHPLOT outperformed the baseline without content planning in all aspects, demonstrating the contribution of content plans in story generation even when they are imperfect.

Furthermore, we measure intra-story lexical diversity using self-BLEU (Zhu et al., 2018) and within-story lexical diversity (or repetition) using distinct-n (Li et al., 2016) and summarize the result in Table 3. Unsurprisingly, the baseline without explicit content planning generates less diverse stories because they are sampled by conditioning on the same instruction. It also generates more within-story repetitions than SCRATCHPLOT.

**RQ3: Which story ending ranking performs best?** We compare our story ending ranking with three alternatives, selecting the best NSP score, the best PPL score, and a random story body and ending pair. We randomly sample 50 content plans where each method selects a different story ending and use them to conduct a pair-wise crowd-source evaluation. Each time, we present the annotators two stories generated using the same content plan, one of which uses our story ending ranking (with randomized order). We highlight the story endings for ease of comparison and ask them to rate which story ends better. We take the majority vote from three annotators for each comparison and present the result in Table 4.

Table 3: Result of self-BLEU scores to measure intra-story diversity (the lower the better) and distinct-n scores to measure repetitions (the higher the better). The best results are highlighted in bold.

Using NSP to select story endings does not perform better than random when compared with our proposed method. Due to the unreliability of NSP scoring, using the mean rank also performs worse than relying on the perplexity alone. However, the difference is not statistically significant. Based on this empirical result, it seems perplexity is a robust metric to select appropriate story endings.

Table 4: Pair-wise comparison of the story ending quality. “Win” indicates our method is rated better than the alternative and vice versa for “Lose”. * indicates statistical significance using two-sided Wilcoxon signed-rank test with $p=0.01$.

5 Conclusion

We introduced SCRATCHPLOT, a framework to perform unsupervised content planning for story generation using only pretrained language models (PLM). SCRATCHPLOT achieved strong results compared to supervised baselines fine-tuned on large parallel corpora and a PLM without access to content plans. In future work, we plan to generalize the framework to other types of long-form text.

---

8Detailed evaluation on the content plan quality can be found in Appendix D.

9They may or may not have the same story body.
Ethical Considerations

Our proposed method is intended for creative text composition. The generated stories can be either consumed by readers or help writers to come up with new ideas. There are several potential risks if the proposed method is not deployed with care. However, they are inherent from large pre-trained language models (PLMs) instead of intrinsic to our method.

First, PLMs may recall partially from the training data instead of composing stories from scratch. Due to the vast size of the pre-training data, it is not feasible to measure what percentage of the generated stories are “original”. Secondly, the system sometimes generates real person names of famous people as the main characters. It should be noted that the system is for literature purposes and is not meant to be a factual report of real persons or anecdotes. Lastly, the system might generate inappropriate or disrespectful stories to a particular population, such as the genres “biblical epic” and “erotic”. Manual curation or automatic content filtering can be deployed to mitigate this problem.

We relied on crowdworkers to conduct human evaluations in this work. The crowdworkers are from various countries, and the adequate payment differs drastically. Therefore, We target paying $6.0 per hour. Some of the tasks took longer than we initially estimated, and we issued all crowdworkers a one-time bonus of $0.2 to compensate.

Although we use a relatively large PLM (GPT2-XL; 1.5 billion parameters), our approach does not require training. Generating a single story takes around 1 minute, consuming 0.003 kWh power based on the max power consumption of the Quadro P5000 we used in the experiment.

References


A Full List of Task Descriptions

Table 5 shows the complete list of task descriptions to generate various plot elements for content planning.

We use self-debiasing (Schick et al., 2021) when generating locations and casts to ensure the generated texts using each task description are distinct. The intuition of self-debiasing is to calculate the averaged texts using each task description are distinct.

The intuition of self-debiasing is to calculate the averaged texts using each task description are distinct.

The intuition of self-debiasing is to calculate the averaged texts using each task description are distinct.

A Full List of Task Descriptions

Table 5 shows the complete list of task descriptions to generate various plot elements for content planning.

We use self-debiasing (Schick et al., 2021) when generating locations and casts to ensure the generated texts using each task description are distinct. The intuition of self-debiasing is to calculate the averaged texts using each task description are distinct.
**Element** | **Task Description**
---|---
Location | **Task:** Write the name of a country. | **Country:** “
**Task:** Write the name of a province. | **Province:** “
**Task:** Write the name of a city. | **City:** “
**Task:** Write the name of a county. | **County:** “

Cast | **Task:** Write the male character’s full name in a story happened in \(<X_1>\). | **Full name:** “
**Task:** Write the female character’s full name in a story happened in \(<X_1>\). | **Full name:** “

Genre | **Task:** Write a story genre. | **Story genre:** “
**Task:** Write a literary genre. | **Literary genre:** “
**Task:** Write a novel genre. | **Novel genre:** “

Theme | **Task:** Write the main point from a \(<X_1>\) story. | **Main point:** “
**Task:** Write the twist in a \(<X_1>\) story. | **Twist:** “
**Task:** Write the lesson learned from a \(<X_1>\) story. | **Lesson learned:** “
**Task:** Write the spectacle of a \(<X_1>\) story. | **Spectacle:** “

Table 5: Full list of task descriptions to generate each element. \(<X_1>\) denotes the previously generated element. Please refer to Figure 1 for the dependency among plot elements and the task descriptions to generate the story body and ending.

The token’s probability \(p_y\) assigned by the PLM using each task description. For each label \(y\) (specified by a task description), the token’s final logit is as follows:

\[
\delta_y = p_y - \max_{y' \neq y} p_{y'} \quad (1)
\]

We do not use self-debiasing when generating other plot elements because the task descriptions are complementary to each other.

**B Post-Processing**

We perform various post-processing depending on the plot elements. We rely on simple heuristics based on common errors we observe.

**Including tailing punctuations** For some plot elements, we expect a phrase instead of a whole sentence. However, the PLM sometimes inserts a punctuation mark, such as a full stop or a comma. Therefore, we recursively remove punctuations at the end till the last character is a letter.

**Repeating the prompt** We observe that the PLM sometimes repeats or rephrases the task description instead of trying to perform the task. Therefore, we filter out continuations that contain any word in the task description (excluding stop words and the text replacing the placeholder \(<X_1>\)).

**Generating 1st or 2nd person pronouns** We usually do not expect first or second-person pronouns in a story plot. When the model generates plot elements containing first or second-person pronouns, it is often generic or opinionated, such as “I’ll try not to think about it” or “You will not fail me.” Therefore, we filter continuations containing a first or second-person pronoun of any case.

**Ignoring task description** When generating the story, we want to ensure that it includes essential plot elements specified in the task description. Therefore, we filter out a story if it contains fewer than two of the following \{male character’s first name, female character’s first name, location\}.

Table 6 overviews the post-processing applied when generating each type of output.

**C Additional Experimental Setups**

We generate a large set of plot elements offline in batch and store them to speed up the inference. The number of each plot element is shown in Table 8. When generating stories, we randomly sample plot elements and combine them to form a content plan. Table 7 presents the detailed parameters used for each type of generation.

For the Fusion model (Fan et al., 2018), we use the checkpoint provided in the official repository, which is fine-tuned on the WritingPrompts dataset with 300k prompt-story pairs. For ProGen (Tan et al., 2021), we use a two-stage seq2seq architecture, where the first seq2seq model takes the input keywords and generates a refined intermediate representation containing keywords with finer-grained details. The second seq2seq model then uses it as input and generates the final story. We fine-tune a BART-base model for both stages using 1k examples randomly sampled from the WritingPrompts dataset following Tan et al. (2021). We truncate...
## D Evaluation on Generated Content Plan

We invite an expert annotator to rate each generated plot element\(^{10}\). We use binary rating (acceptable/unacceptable) for location, cast, and genre. We use a scale from 1 to 5 for theme because it is more subjective. We present the result in Table 8.

<table>
<thead>
<tr>
<th>Element</th>
<th>Count</th>
<th>Location</th>
<th>Cast</th>
<th>Genre</th>
<th>Theme</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>43</td>
<td>493</td>
<td>24</td>
<td>117</td>
<td></td>
</tr>
</tbody>
</table>

### E Details of Crowdsource Evaluation

We conducted the crowdsource evaluations on the Toloka platform\(^{11}\). In this section, we detail the specification of the annotation tasks, the quality control measures, and the stats of the annotation.

#### E.1 Annotation Task Specifications

For the fine-grained evaluation, we decompose it into a separate annotation task per aspect so that the annotators can focus on evaluating a single aspect and avoid context switching.

**Fine-grained evaluation**  
Rate each story in the following aspects on a scale of 1 (worse) to 5 (best).

- **Naturalness**: is the story fluent and understandable? The language should be natural. Minor grammatical errors are acceptable if they do not affect understanding the story.
- **Interestingness**: is the story interesting to readers? Rate this aspect as objective as possible. Assuming someone familiar with the particular genre, will the story interest them?

---

\(^{10}\)Plot elements are much shorter and faster to rate. Therefore, we use an expert annotator for superior accuracy.  
\(^{11}\)https://toloka.ai/
• **Cohesiveness**: is the story cohesive and logical? Common problems include mixing up the characters and introducing illogical event sequences (unless it appears like a deliberate choice).

Figure 5 shows the annotation interface and Figure 6, 7, 8 shows the detailed annotation instructions.

**Story ending evaluation**  Indicate which of the two stories has a better ending. A good story ending should be relevant to the story, logical, conclusive, and thoughtful. Figure 9 shows the full annotation instructions and Figure 10 shows the annotation UI.

**E.2 Quality Control**

We select crowdworkers who are fluent in English and among the 20% top-rated performers. Figure 4 shows a screenshot of the annotator filter. Additionally, they have to pass a short training session and correctly answer 3 out of 4 training questions to be selected for the main evaluation. If they answer a question wrongly during training, the system will show a hint to help them improve, as shown in Figure 11.

During annotation, we apply various quality control rules, including limiting each annotator to no more than 50 tasks, adding occasional captcha to block bots, banning users who consistently submit tasks too fast (less than 5 seconds for fine-grained evaluation and less than 10 seconds for story ending evaluation), and banning users who skip more than 5 tasks in a row.

**E.3 Annotation Task Stats**

We paid $0.05 for each fine-grained evaluation task. On average, it took around 30 seconds to complete each task, making the average earning $6 an hour. There are around 40 crowdworkers evaluating for each aspect. Figure 12 shows an example pool stats for the naturalness evaluation.

We paid $0.1 for each story ending evaluation task, which takes on average 1 minute 13 seconds to complete. There are in total 20 crowdworkers participating in this evaluation task.

The overall budget we spent on all crowdsourced evaluations is $200 (including payment and bonus to crowdworkers and platform fees).

**F Sample Generated Stories**

We present randomly sampled stories generated using different models in Table 9 and story endings selected by different ranking metrics in Table 10.
Figure 5: Annotation interface for the interestingness aspect. The interface for other aspects are analogous and we omit them for brevity.

Figure 6: Annotation instructions for the naturalness aspect.
Instructions

Please rate how interesting a short story (~100 words) is on a scale of 1-5.

Evaluation scale
There are five categories on the scale: 1 (very dull), 2 (somewhat dull), 3 (acceptable), 4 (somewhat interesting) and 5 (very interesting).

To make a correct evaluation, read the whole story instead of skimming through the content. Ignore minor grammatical errors and focus on the overall quality.

- **1 (very dull)**: the story is by no means interesting. It's a waste of time to read it. If the text isn't understandable or isn't even a story, you should also assign the score 1.
- **2 (somewhat dull)**: the story is boring. It would fail to interest the majority of the readers.
- **3 (acceptable)**: the story has some twists. However, overall it's not there yet.
- **4 (somewhat interesting)**: the story has an interesting or unexpected plot. It may interest some audiences.
- **5 (very interesting)**: the story is very interesting. People would love to read it.

Figure 7: Annotation instructions for the interestingness aspect.

Instructions

Please rate how coherent a short story (~100 words) is on a scale of 1-5.

Evaluation scale
There are five categories on the scale: 1 (very incoherent), 2 (somewhat incoherent), 3 (acceptable), 4 (somewhat coherent) and 5 (very coherent).

To make a correct evaluation, read the whole story instead of skimming through the content. The coherence problem usually appears across multiple sentences. While individual sentences may seem meaningful, sometimes, they don't fit together as a whole story.

- **1 (very incoherent)**: the story doesn't make sense. It doesn't have clearly defined characters and a plot.
- **2 (somewhat incoherent)**: it's difficult to understand the main plot. Some events seem out of order.
- **3 (acceptable)**: the story has obvious logical errors, but I can understand what it's about.
- **4 (somewhat coherent)**: the story plot is more or less clear. However, there are minor logical errors.
- **5 (very coherent)**: the story is very coherent. There's a clear progression of events. It could have been written by a human.

Figure 8: Annotation instructions for the cohesiveness aspect.
Instructions

Please rate which of the two stories has a better ending. While the story endings are highlighted in bold for your convenience, please read the whole story instead of only the endings. A good story ending should be:

- **Relevant**: relevant and coherent with the story. If it seems irrelevant, it's a poor story ending.
- **Logical**: the story ending should be logically derived from the story. While surprises might occur in stories, it should be intentional and understandable.
- **Conclusive**: ideally, story endings should conclude the story and don't leave any open loops. However, certain genres of stories tend to leave an open question to the reader. Please perform your judgement on whether a story ending is appropriate.
- **Thoughtful**: ideally, a story ending should convey a message or insight besides continuing the narrative.

Additional Notes:

- "better" doesn't imply it must be a 'happy ending'. A tragic ending, if appropriate should be rated better than a low-quality happy ending.
- While the qualities mentioned above are ideal, they may not satisfy in every story. Please contrast the two stories and pick the one that has a relatively better ending.
- Sometimes, the story bodies contain minor errors. While you should read them to get the context, the evaluation should be focused on the story ending.

Figure 9: Annotation instructions for the story ending evaluation.

Figure 10: Annotation interface for the pair-wise story ending evaluation.
Figure 11: Screenshot of the training session. A hint is displayed for the examples where the annotator provides the wrong answer.

Figure 12: The annotation pool stats for the naturalness evaluation.
Location: Alameda County
Cast: John and Cheryl
Genre: literary fiction
Theme: The only problem with the house was the people.

Fusion:
“What?” she whispered, as she pulled a small piece from her coat. It was a small piece of paper, with a picture of a young man sitting on that couch. “What?” he said, as he flipped the picture around. “My name is John, and I’m here to give you a story.” he said, as the photo of his wife came out of his hand and he pulled out a book. The picture was of some kind of book, with the Briggs on it. “What?” she said, as she looked at it. “I don’t know,” John said, as he read the book. “It was about to get pretty bad. I don’t like it here.” “It’s not good,” she said. “What did you tell me?” he asked, as the picture was on the couch. “I just can’t believe it

ProGen:
The first son I held inside my wife skin and body, I rested my head on her chest and prayed to the waiting Forbiddone, loved by the boy and girl alike, and that the house would bring smiles to all. Jerry Fairey, anxious and excited, pushed his face into my naked flesh. Jerry sounded just like his daddy and chirped, “Grr!” and they laughed together. Jerry took a deep breath, thought of again, “Fuck this. This place is empty, people.” Jerry slept.

ScratchPlot: John and Cheryl had decided that they’d get a divorce. They’d never been married, and they’d always been in the midst of an argument that had escalated to violence. John had left the house, locked the door behind him, and walked into the woods. Cheryl had followed him, but after a brief conversation with her father and a call to an ex-boyfriend from his stepmother, she had walked out to the car. After John had walked down the driveway, he came up behind Cheryl in her driveway and punched her in the face with both fists.

- content plan: The plot summary tells the reader what the story is about. Plot summaries are very long and sometimes the authors need to explain some things and show the readers something. I use a simple one here: The plot summaries of most of the books I read are in the order of their publication date (as seen above). The authors do not like the readers to have to remember this order, so they just list the events in a very long way. I have seen plots summary in the form of a list in the back of a book, but that has problems: the reader has to find the book. A plot summary in a book will be much more readable and more likely to get across the events.

Table 9: Stories generated by different models. All models except for “- content plan” use the same plot element(s) listed above.
ScratchPlot: Jack Ryan was a detective for the Gotham City Police Department who specialized in counter-terrorism and intelligence work. Doreen Green was a journalist who specialized on terrorism, and was a regular contributor to Gothamist. Dormammu was a powerful villain with a reputation that reached far into the Gotham underground. Jack began investigating Dormammu and the mysterious Ornaments of Death. [SEP] After his first interview with Dormammu, Jack Ryan’s mind is shattered, and it’s not long before he has no memory of who he is.

NSP: Doreen’s sister is in prison for murder. Jack Ryan, the most brilliant and ruthless man the government has ever produced, is trying to rescue her. He’s in Gotham and the place is swarming with criminals. Jack, with the help of two friends, the brilliant and deadly Jack Ryder and the deadly and brilliant Tommy Elliot, sets out to rescue Doree, and make things right. [SEP] They have the world at their feet but they have the eyes of a hawk.

PPL: Jack and Dora Green, the daughter of a major newspaper publisher, live in the Gotham Mansion, the home of Jack’s business partner, Doree Green, and the mother of their three-year old daughter, Dora. Their home is a large home with a large lawn. A large stone sculpture of a man stands at the front of the front door, which is marked with a red paint that is difficult to remove without destroying the painting. [SEP] It was a beautiful day, sunny and warm. The grass was green and lush. The air was clear, and crisp. It was the perfect summer day.

Random: Jack Ryan was a detective for the Gotham City Police Department who specialized in counter-terrorism and intelligence work. Doree Green was a journalist who specialized on terrorism, and was a regular contributor to Gothamist. Dormammu was a powerful villain with a reputation that reached far into the Gotham underground. Jack began investigating Dormammu and the mysterious Ornaments of Death. [SEP] And then there was a loud thud, like the sound of an iron fist smashing into concrete, and then the sky was a blood red.

Table 10: Story body and ending selected by different algorithms. We manually insert a [SEP] token to indicate the boundary between the story body and ending.