Ranked Keywords to Story Generation

Stanford CS224N Custom Project, Mentor: Mina Lee

WenXin Dong
Department of Computer Science
Stanford University
wxd@stanford.edu

Abstract

This project attempts the following task: given a set of ranked keywords, construct a coherent short story. The goal is for the model to use all the keywords, while still being grammatically and logically correct. To solve this task, we try both the traditional method of finetuning large pretrained language models as well as the recently introduced Plug and Play Language Model(PPLM) strategy, which leverages the power of pretrained language models without finetuning them. For the Plug and Play strategy, we introduce custom attribute models to guide language models to generate stories containing the desired keywords, especially those with higher rank. Unlike the original PPLM paper which focuses on perturbing the generation of zero-shot unconditioned language models, we experiment with zero-shot, low-resource, and fine-tuned language model choices, and compare the relative improvement in the PPLM generations. We find that finetuned language models perform much better than the default PPLM approach, but our custom combination of finetuned language model + attribute model performed the best overall. Finally, we perform error analysis on all our approaches and find that in spite of introducing more grammar mistakes, PPLM improves keywords usage, reduces the number of contradictory sentences in the stories, and generates stories with better endings.

1 Introduction

The goal of this project is to address the following task: Given a set of ranked keywords, construct short story that uses all the keywords in a logically consistent way. The ranking determines the relative importance of each word, and the model is expected to use high-rank keywords to guide the story. For example, given the words 'josh, streets, living, adopted, happy', the model could output:

Josh is a black dog. He was living on the streets. A nice man stopped when he saw Josh. He became attached to Josh. So the man adopted Josh, and Josh is very happy with his new family.

The goal is for the model to be creative with using all the keywords, while still being grammatically and logically correct. One could imagine this task being used to inspire writers with creative story ideas. This project uses both the traditional method of finetuning large pretrained language models, and the PPLM [1] approach (which will be explained later in the Approach section) to solve this task. We are interested in which approach performs better, and how much improvement can PPLM add unto zero-shot, low-resource, and finetuned language models.

This task itself is novel and different to previous tasks in controllable story generation:

- **There is no fixed number of keywords.** There will only be a soft constraint, e.g. 10 words max, and 1 word min, and we aim to use all keywords without compromising length of the story. Thus, this task is different from most outline-based story generations, where the input is a sequence such as words and event representations, and each sentence is built using one element in the sequence at

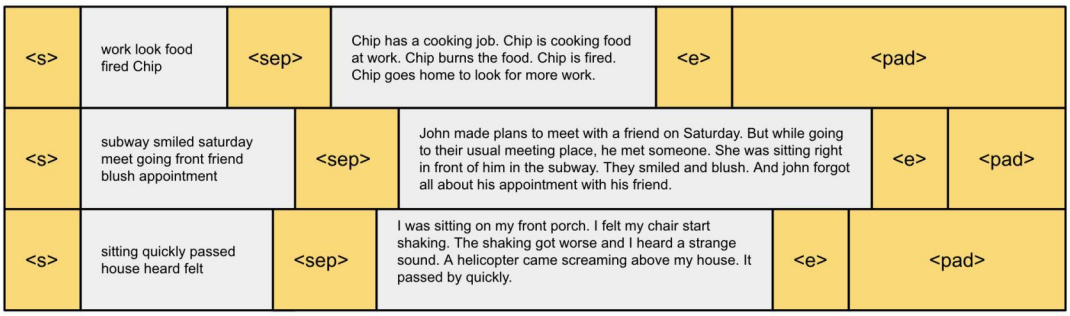

Figure 1: **Training examples.** <s>, <e>, <sep> and <pad> are special tokens. Between <s> and <sep> are the input tokens, and between <sep> and <e> are the output tokens.

a time, [2] and hierarchical story generations [3] [4] work, where the the number of words/events in the storyline is usually fixed [3] [2].

- **The keywords need not be logically connected**. The motivation of this task is to help writes to connect seemingly random ideas together, which is arguably harder than connecting already logically related keywords or generating a story from scratch. Therefore, this task is different to prompt based story generation, where the input is a beginning sentence, title[3], planned storyline[5], or a short description[6].
- **The keywords are ranked in order of importance.** Work has been done on variable length keywords to story/text generation [7] [5] [8], including PPLM[1]. This task is different because the importance of each keywords matters. The model should come up different logic based on different ranking of the keywords. Moreover, regarding PPLM specifically, the keywords per sequence ratio is expected to be much higher for the current task, meaning the attribute becomes a lot harder to satisfy than the high-level attributes, such as sentiment and topic, presented in the PPLM paper.

2 Related Work

Controllable text generation. Conditional text generation encapsulates controllable story generations, and is an active research area in NLP. For the task of keyword-conditioned short text generation, template-based approaches include filling POS templates with keywords [9] and combining short keyword-included phrases using dependency trees[10]. Work has been done on using RNN and LSTM to generate text based on context words. [11] explored how to best choose context words to create stronger semantic relationship between input and output. CTRL is a large transformer based model trained from end to end, that outputs sentiment or topic specific text conditioned on input attribute codes [12]. Uber AI introduced a Plug and Play language model for controllable text generation that leverages the power of large pretrained models without finetuning them [1].

Controllable story generation. With the rise of RNN and deep neural networks, works on controllable story generation increasingly explored seq2seq models with attention mechanisms [3] [4] [7] [6]. Recently, the release of large pretrained language models such as GPT-2 and T5 led to the exploration of transfer learning [13] [2] [5] [14][15]. In particular, [15] is a work on commonsense story generation where GPT-2 is further trained on external commonsense knowledge base.

Hierarchical story generation. A notably popular research focus in controllable story generation is "hierarchical story generation", which is highly relevant to this project. Yao et.al proposed a plan-and-write hierarchical generation framework that first plans a story line and then generates a story based on the story line. Their three major problems were off-topic, repetitive, and logically inconsistent [3]. Martin et.al proposed to first generate a sequence of 4-tuple event representations and then a sequence of sentences based on the events tuples [4]. Fan et.al collected 300K human-written stories paired with writing prompts from Reddit, and trained a model that first generates a prompt sentence and a "fusion model" that generates passage conditioned on the prompt. Problems with this approach included repetition and generation of overly-generic prompts [7]. Jain et.al generated stories based on an input sequence of independent one-line descriptions describing a scene or an event. They reported semantically disconnection between input descriptions and generated stories [6].

3 Approach

To exact keywords from stories, we use an algorithm called Rake[1]. Rake extracts ranked keywords from the training set and the generated stories, and we compare the similarity of the two sets of keywords at evaluation time. We allow the number of input keywords to range anywhere between 1 and 10, providing additional flexibility. Since the average number of words in the training stories is 42.5, we want a relative sparse keywords to story ratio, hence choosing a cap of 10. The cap needs not to apply to out-of-domain inputs.

We frame this problem as a sequence to sequence problem, and approach it in three ways.

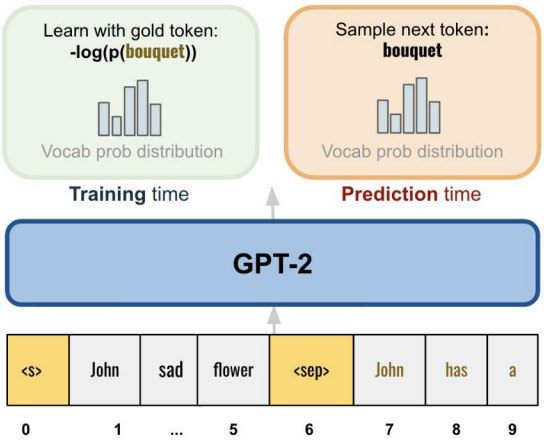

Figure 2: **Finetune pretrained GPT-2.** Assuming the gold tokens are "John has a bouquet of ...", during training we calculate the cross entropy loss on the gold token, while during prediction we perform top-k sampling.

Seq2Seq Bi-LSTM with Attention Mechanism For our baseline, we use the OpenNMT library to train a seq2seq model with 2 bidirectional encoder LSTM layers and 2 unidirectional decoder LSTM layers, with the default attention mechanism. The attention score for each decoder query q is calculated using the bidirectional hidden states of the input sequence H_j's and a parameterized matrix W_a: $\text{score}(H_j, q) = H_j^T W_a q$. We use early-stopping on both accuracy and perplexity. We input tokenized keywords and train on gold stories.

Finetune pretrained GPT-2 We use the Huggingface Transfomer library to finetune the pretrained GPT-2 model as shown in Figure 2. Figure 1 shows the format of the input data, which is <|eos|> keyword tokens <|sep|> story tokens <|eos|>.An important implementation note is that the loss is only calculated for text generated after the <|sep|> token.

Plug and Play Language Model. PPLM [1] is a controllable text generation strategy which emphasizes on not needing to fine-tune large language models. The key idea is to use a small attribute model to control a large, unconditioned language model to generate text satisfying a particular attribute. The intuition is that the attribute model moves the generated sequence toward a higher probability region for satisfying the attribute, while the language model maintains fluency of the generated text. In this project, our attribute is keywords usage, and we experiment with multiple custom attribute models, as explained in Section 4.2. Let us first describe the PPLM approach.

Let LM be a language model, such as pretrained GPT-2 or finetuned GPT-2. Let H_t represent all hidden states of LM that influence the prediction of the next token, so that

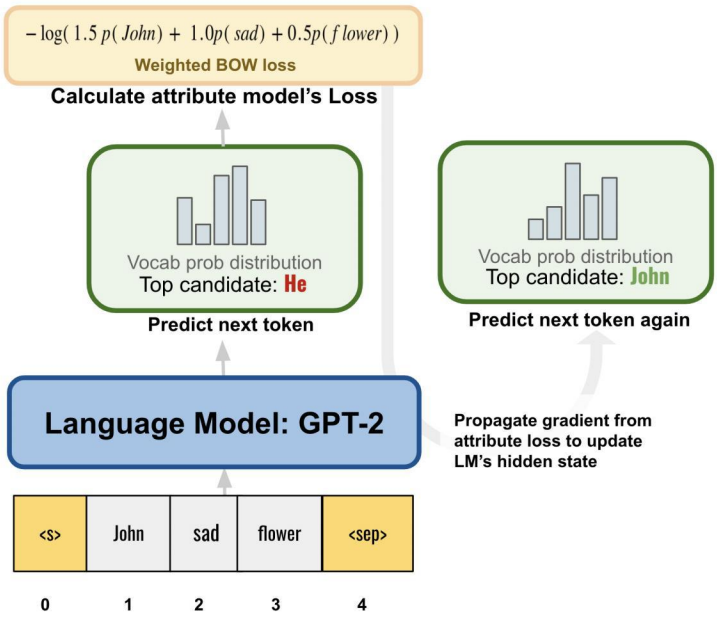

Figure 3: **PPLM approach**. Assume the input keywords are John, sad, flower. The language model (LM) first generates an unperturbed probability distribution. Then, gradient of the attribute loss propagates back to LM, updating its hidden states. Next, LM generates a perturbed probability distribution.

[1]https://pypi.org/project/rake-nltk/

$o_{t+1}, H_{t+1} = LM(x_t, H_t)$, where x_t is the input token and o_{t+1} a logit vector.

Instead of just using o_{t+1} to predict the next token, PPLM uses an attribute model to perturb H_t to $H_t + \Delta H_t$, and uses the perturbed hidden states to generate a new logit vector $\tilde{o}_{t+1}$ to predict the next token, which presumably generates a better token at satisfying the attribute.

PPLM computes ΔH_t through m iterations, with ΔH_t initialized to zero. In each iteration, PPLM first take derivative of ΔH_t with respect to loss of the attribute model,

$$\Delta H_t \leftarrow \Delta H_t + \alpha \frac{\nabla_{\Delta H_t} AttributeLoss(H_t + \Delta H_t)}{||\nabla_{\Delta H_t} AttributeLoss(H_t + \Delta H_t)||^{\gamma}}$$

where α is the step size, and γ is a hyperparameter. Next, PPLM moves ΔH_t to minimize the KL divergence between $P(x) = softmax(\tilde{o}_{t+1})$, the perturbed distribution, and $Q(x) = softmax(o_{t+1})$, the unperturbed distribution:

$$\Delta H_t \leftarrow \Delta H_t + \lambda_{KL} k_t$$

where k_t is the KL coefficient between $P(x)$ and $Q(x)$, and λ_{KL} a scalar hyperparameter.

Although our approach is inspired by PPLM, it differs from the default PPLM approach, as we don't restrict our language model to zero-shot language models. Specifically, we first compare the default PPLM approach (zero-shot GPT-2 + attribute model) against our finetuned GPT-2 model. Secondly, we improve the performance of our finetuned GPT-2 by substituting it as the language model in the PPLM approach.

4 Experiments

4.1 Data

We use 5-sentence English short stories from the ROC story dataset [16]. We split our training, validation and testing dataset into 42,132, 4,978, and 5,555 stories correspondingly. The vocabulary size is 31,492. Some training examples are shown in Figure 1.

4.2 Evaluation Metrics

We use automatic and human evaluation to measure model performance.

For automatic evaluation metrics, we use **Perplexity**, a proxy measure of fluency. Similar to the authors of PPLM, we use a separate language model, pretrained GPT-2 medium, to evaluate perplexity of the generated stories; **Keywords Coverage**, the ratio of input keywords being extracted by Rake as keywords in the generated story. **Rank Biased Overlap (RBO)**, a similarity metric for any two ranked lists. In this case, we compare the keyword list, extracted by Rake, of the generated story to that of the the gold story. RBO score rewards correct keywords ranking in addition to keywords coverage; It is a more comprehensive score but harder to reason about, that's why we have also included Keywords Coverage. **Repeat-4-grams**, ratio of generated stories that repeated at least one 4-gram. This is to measure the level of repetition of the generated text. **Distinct-4-grams**, ratio of distinct 4-grams out of all 4-grams. It measures corpus-level diversity, the higher the score the more diverse the stories being generated are.

For human evaluation metric, we first identified the main categories of errors, as listed in Table 3. We sampled 15 set of keywords and generated one story from each model per keyword set, and evaluate those stories (a total of 150 stories). We believe 150 examples are sufficient for identifying common errors, but we would like to evaluate on more examples to get more reliable results. We would also like to have more human annotators as the writer was the single annotator, to reduce implicit bias. The error categories are

Grammar mistake, the number of sentences that contained one or more grammar mistakes

E.g. Jose did not like trumpet music ***very good****.*

Logical contradiction, the number of sentences that contradict with itself or previous sentences

E.g. ...Jose's friend asked him why he ***did not play trumpet*** *at school with him. Jose* ***admitted he was playing the trumpet.***

Repetition, the number of sentences that repeat or paraphrase previous sentences

*E.g. The dog **ate breakfast** at his parents' house. **He ate his breakfast** .*

Entailed but illogical progression, the number of sentences that introduce no contradictions but that are illogical continuation of the story.

*E.g.Tome had a lot of friends over. One night while moving he got lost. It was very annoying. **Tom was able to call back a lot of people over and over.***;

Incomprehensible, the number of sentences that may or may not be grammatically correct and are incomprehensible to a human, both when in story context and when read as a stand-alone sentence.

E.g. He had problems with his drink and it was not going to fix the problems on his drink.

No or Invalid Ending Although our approach did not account for endings, we noticed that our models were able to implicitly learn this property from the training data, to various extents. Since having a sound ending significantly improves the quality of the generated story, we decide to add this metric. This is a rather subjective category and measures whether a story ends on a conclusive or assertive remark that makes a human feel like the story has reached to a logical endpoint. For example, I assigned this following story With Ending:

Jose was playing at school with his class. His friend invited him over for a trumpet solo. Jose practiced for about ten minutes. The trumpet solo was great! Jose was a great trumpet player!

I assigned this following story No Ending:

Jose was playing at school with his class. His friend invited him over for a trumpet lesson. Jose did not listen to his classmate, who was good at it. Jose's friend asked him why he did not play at school with him. Jose admitted he was playing the trumpet.

Multiple error categories could overlap. For example, a sentence could be both grammatically incorrect, incomprehensible, and an illogical progression. In fact, it was common for a sentence to introduce more than one error. When counting the number of errors, we assigned a count of $\frac{1}{2}$ when in doubt.

Substitute for LM in PPLM	Model description
Zero-shot GPT-2(124M)	Pretrained GPT-2 without any additional training
Low-resource GPT-2(124M)	Pretrained GPT-2 further trained on 200 examples, for 1 epoch [2]
Finetuned GPT-2(124M)	Pretrained GPT-2 further trained on 42k examples, for 3 epochs

PPLM: Language Model Options. Unlike the original PPLM paper which focuses on zero-shot models, we experiment with zero-shot, low-resource, and finetuned language model. We train GPT-2 on the same task as described in Figure 2.

4.3 Experimental details

As mentioned in the Approach section, we first train a seq2seq baseline using OpenNMT's bidirectional RNN model with early stopping.

For our second approach **Finetuned GPT-2** (124M), we manually evaluated checkpoints for 500 steps, 2000 steps, and 4000 steps, and found the 4000 steps (full 3 epochs) checkpoint performed the best on both human and automatic metrics. We used top-k = 10 during prediction time and found that without top-k the generated stories tend to repeat short phrases. We used batch size of 32 and a learning rate of 0.00005, the default hyperparameters from the library.

The PPLM approach. We experimented with different combinations for the language model and the attribute model, as listed in Table 1 and 4.2, to show the effectiveness of the PPLM approach. . We used $\alpha = 0.02$, $\lambda_{KL} = 0.01$ and $\gamma = 1.5$ as our hyperparameter setting, as suggested by the PPLM authors, throughout the project.

We introduced our three original attribute models, with the following motivations and justifications. We include the loss functions of the attribute models in Table 1.

[2]We train on 200 examples (randomly selected from the training set) because the model does not output understandable output when trained on less than 200 examples.

Substitute for attribute model in PPLM	**Loss function**
Weighted BOW model	$$-log(\sum_{k \in \{k_1,...,k_n\}} \alpha_i p(k_i))$$ $p(k_i)$ is the probability of the next token being k'th keyword, using the logits from LM. α_i is weight assigned the k'th keyword, which decreases linearly with k. We define $\alpha_i = \frac{2(n-i+1)}{n+1}$, which makes all the α_i's sum to n
Weighted BOW model with Popping mechanism	$$-log(\sum_{\substack{k_i \in \{k_1,...,k_n\} \\ k_i \notin \{x_1,...,x_t\}}} \alpha_i p(k_i))$$ x_i's consists of the generated sequence so far.
Weighted Synonym-BOW model with Popping mechanism	$$-log(\sum_{\substack{w \in \{k_1, syns(k_1),...,k_n, syns(k_n)\} \\ w \notin \{x_1,...,x_t\}}} \alpha_w p(w))$$ For each keyword k_i, $\alpha_{k_i} = \frac{(n-i+1)}{n+1}$. For each synonym in $syns(k_i)$, $\alpha_{syns(k_i)_j} = \frac{(n-i+1)}{(n+1)(\lvert syns(k_i) \rvert}$. In words, we assign half of the weight to the keyword, and splitting the other half evenly among the synonyms of the keywords, such that all weights sum to n

Table 1: **PPLM: Attribute models options**: Weighted BOW model (**Weighted**), Weighted BOW with Popping mechanism (**Weighted+Pop**), Weighted Synonym-BOW with Popping mechanism (**Weighted+Pop+Syns**)

Weighted BOW model This is the simplest attribute model. Weighted dynamic BOW penalises the LM for not generating words from the set of input keywords, and uses weighting to penalize the model more for not generating high-rank keywords. We choose to use linear weighting, assuming the n keyword have weights $n, n-1, ..., 1$ respectively. In practice, we could allow users themselves to assign a weight to each keyword.

Weighted BOW model with Popping mechanism This attribute model removes keywords what have already been generated in the sequence from the bag. This is an effective improvement as the naive weighted BOW model leads to repetition of higher rank keywords and keywords that also happen to be common English words.

Weighted Synonym-BOW model with Popping mechanism Popping mechanism alleviates repetition. However, we want to further improve keywords usage. To do so, we add synonyms of the input keywords to the bag of words, hoping to send stronger signals to the LM to move in the desired directions. An example of synonyms added is shown in Table 5. We use the NLTK library to find synonyms for each keyword. We keep the weights to enforce correct ranking, and make synonyms weigh less than the actual keyword to encourage the model to generate the actual keywords. Detailed loss function is shown in Table 1.

4.4 Results

Detailed metric results is shown in Table 2. The following are key observations.

- **Zero-shot GPT-2. Since we use the same random seed at generation time, the generated stories are nearly identical for all the test prompts, suggesting that zero-shot GPT-2 is not picking signals from the attribute model**. This could be seen through examples provided in the Appendix, for example in Table 4. This is in contrast to the PPLM paper where the authors managed to simultaneously satisfy the attribute and coherency. We finetuned the the hyperparameters, namely α, λ_{KL} and γ, to force stronger control of the attribute model on the language model. However, this led the language model to simply output keywords repetitively (e.g. "joe water water water"). We were not able to generate the keywords while maintaining fluency and coherency. We believe this difference is caused by tasks themselves. Higher-level attributes like topic and sentiment in the PPLM paper are defined with large bags of words, for example BOW for the Religion topic contains 196 words, and are therefore much easier to satisfy. Our bag of words is much smaller, the keywords are not similar to each other, and thus the attribute is much harder to satisfy. We hypothesis this is why in order to satisfy the attribute, the model sacrificed fluency.

- **Low-resource GPT-2. Low-resource GPT-2 by itself generated stories that made no logical sense and are many times just 1 sentence long. When using the PPLM approach on low-resource GPT-2, we observe slight improvement in keywords usage, with slight decrease in fluency and coherency.**

- **Finetune GPT-2.** Finetuned GPT-2 on itself already performs very well on the task - using more than 70 percent of keywords on average. The generated stories are much more fluent, coherent, and understandable. **Using finetuned GPT-2 as the language model in the PPLM approach, we observe a maximum of 3.8% increase in Keywords Coverage and a maximum of 3.5% in RBO score. However, we also observe a maximum of 2.17 points increase in perplexity as well as increase in repetition as compared to finetuned GPT-2 on its own.** Importantly, however, we notice improvement in perplexity with the Weighted Syns-BOW model with popping mechanism, as compared to the other two attribute models. We believe the addition of synonyms into the bag of words makes the bag larger and thus reduces the relative importance of each word, as all the weights had to sum up to the number of keywords. Therefore, the perturbation of hidden states is less "screwed" or "biased" towards very specific words, making it easier for the language model to keep the top candidates of it's original probability distribution.

Model	PPL ↓	RBO ↑	KC ↑	Rep4 ↓	Dist4 ↑	Length
Seq2Seq bi-LSTM	21.6	**0.708**	0.717	0.834	0.594	46.5
Zero-shot GPT-2						
Weighted	12.54	0.048	0.063	0.373	0.514	86.68
Weighted + Pop + Synys	**12.19**	0.041	0.046	0.332	0.526	87.56
Low-resource GPT-2						
Unperturbed	28.84	0.220	0.255	0.206	0.941	34.90
Weighted	32.67	0.261	0.296	0.197	0.959	35.41
Weighted + Pop + Synys	33.22	0.259	0.298	0.184	0.949	34.70
Fine-tuned GPT-2						
Unperturbed	23.99	0.646	0.708	**0.115**	0.958	45.50
Weighted	26.16	0.666	0.729	0.144	**0.962**	45.43
Weighted +Pop	26.02	0.671	0.738	0.124	**0.962**	45.43
Weighted + Pop + Synys	25.74	0.681	**0.746**	0.142	**0.962**	45.50

Table 2: **Experiment automatic metrics result** on **Perplexity**, calculated on generated stories using pretrained GPT2-medium, average **Ranked Biased Overlap** score, average **Keywords Coverage Ratio**, **Repetition-4**, **Distinct-4**, and average **Story Length**.

Model	Grmmr ↓	Contrad ↓	Rep ↓	No Prog ↓	Incomp ↓	No End ↓
Seq2Seq bi-LSTM	5.5	7.5	7.0	19.0	**7.0**	11.0
Zero-shot GPT-2						
Weighted	4.0	18.0	4.0	**11.0**	11.5	12.5
Weighted + Pop + Synys	4.5	21.5	2.0	16.0	10.0	14.0
Low-resource GPT-2						
Unperturbed	5.0	9.5	5.0	23.5	13.0	13.0
Weighted	6.5	**4.0**	5.5	15.0	10.0	10.5
Weighted + Pop + Synys	5.0	7.0	5.0	18.5	13.0	13.5
Fine-tuned GPT-2						
Unperturbed	**1.0**	14.5	**0.0**	17.0	9.5	8.0
Weighted	3.0	9.0	**0.0**	20.0	9.0	9.5
Weighted +Pop	3.5	8.0	**0.0**	18.5	9.5	**5.5**
Weighted + Pop + Synys	3.0	7.0	**0.0**	20.0	7.5	**5.5**

Table 3: **Experiment manual metrics result** on number of **Grammar Mistake**, **In-sentence or In-context contradiction**, **Repetition of previous sentences**, **Entailed but illogical progression**, **Grammatically correct but incomprehensible sentence**, **No valid ending in story**.

5 Analysis

An overarching trend in our automatic metrics results is that PPLM improves keywords usage but sacrifices coherency. This is shown through the relative increase in RBO and Keyword Coverage and relative increase in Perplexity. This is not surprising as perturbing the hidden states and forcing the LM to generate a particular word is not organic and could mess up the internal logic of the unconditioned model.

However, our human evaluation, shown in Table 3, provides a more nuanced picture. Although more grammar mistakes are generated, the PPLM models contain fewer contradictory sentences and more sentences that made logical progression in the stories compared to language models on them own. Moreover, when using finetuned GPT-2 as the language model in PPLM, the number of stories with valid ending significantly increased. Therefore, to some extent, PPLM perturbed models generated better stories regardless of keywords usage.

PPLM approach increases performance of low-resource GPT-2 and finetuned GPT-2, but not of zero-shot GPT-2. As explained above in the experimental observation, zero-shot GPT-2 was failed to act on signals from the attribute models due to high attribute specification.

The baseline model in fact performs very well on some automatic metrics. It achieves the highest RBO score and uses 72% of all the keywords. However, it fails in human evaluation, as the generated sentences are too repetitive, and each sentence is likely to not be a logical continuation of the previous sentences. This suggests that high RBO and Keyword coverage alone are trivial and easy to satisfy, as models could simply repeat the input keywords, suggesting the importance of other evaluation metrics.

6 Conclusion

The goal of the this project is generate short stories using ranked keywords as input. We approached this task in three ways: baseline RNN, finetuning GPT-2, and Plug and Play Language Model (PPLM) with custom attribute models. PPLM is a strategy to connect a language model and an attribute model, such that the attribute model guides the generation of the language model. We found that PPLM used on zero-shot pretrained language models fails to generate coherent stories containing the keywords, but PPLM used on finetuned pretrained language models improve attribute satisfiability and story quality overall. We designed three custom attribute models. Our best performing attribute model is a variant of the naived weighted BOW model which pops used keywords and expands the bag of words with keyword synomyns.

The following are some limitations in this project that we would like to overcome in the future. Firstly, we used the same set of hyperparamters through out the study. Although these choices are recommended by the PPLM authors, it is possible for us to get another set of results using a different set of hyperparameters. Secondly, we trained on GPT-2 and only evaluated checkpoints for a maximum of 3 epochs. We believe training for more epochs, or training on GPT-2 medium or T5 would further improve model performance. Thirdly, we did not test on out-of-domain examples. Given time, we could like to test the models on actual user inputs. Fourthly, we believe there are better attribute model designs. For example, a better attribute model could dynamically assign perturbation strength depending on the current generation and the remaining keywords, perhaps increasing perturbation strength after the end a sentence or decrease perturbation strength after a keyword is just being used.

References

[1] Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: A simple approach to controlled text generation, 2020.

[2] Nanyun Peng, Marjan Ghazvininejad, Jonathan May, and Kevin Knight. Towards controllable story generation. In *Proceedings of the First Workshop on Storytelling*, pages 43–49, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[3] Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. Plan-and-write: Towards better automatic storytelling, 2019.

[4] Lara J. Martin, Prithviraj Ammanabrolu, Xinyu Wang, William Hancock, Shruti Singh, Brent Harrison, and Mark O. Riedl. Event representations for automated story generation with deep neural nets, 2017.

[5] Le Fang, Tao Zeng, Chaochun Liu, Liefeng Bo, Wen Dong, and Changyou Chen. Outline to story: Fine-grained controllable story generation from cascaded events, 2021.

[6] Parag Jain, Priyanka Agrawal, Abhijit Mishra, Mohak Sukhwani, Anirban Laha, and Karthik Sankaranarayanan. Story generation from sequence of independent short descriptions, 2017.

[7] Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation, 2018.

[8] Jian Guan, Fei Huang, Zhihao Zhao, Xiaoyan Zhu, and Minlie Huang. A knowledge-enhanced pretraining model for commonsense story generation. *Transactions of the Association for Computational Linguistics*, 8:93–108, 2020.

[9] Abhijit Mishra, Md Faisal Mahbub Chowdhury, Sagar Manohar, Dan Gutfreund, and Karthik Sankaranarayanan. Template controllable keywords-to-text generation, 2020.

[10] Kiyotaka Uchimoto, Hitoshi Isahara, and Satoshi Sekine. Text generation from keywords. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, page 1–7, USA, 2002. Association for Computational Linguistics.

[11] Sivasurya Santhanam. Context based text-generation using lstm networks, 2020.

[12] Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation, 2019.

[13] Lianhui Qin, Antoine Bosselut, Ari Holtzman, Chandra Bhagavatula, Elizabeth Clark, and Yejin Choi. Counterfactual story reasoning and generation, 2019.

[14] Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. Commongen: A constrained text generation challenge for generative commonsense reasoning, 2020.

[15] Jian Guan, Fei Huang, Zhihao Zhao, Xiaoyan Zhu, and Minlie Huang. A knowledge-enhanced pretraining model for commonsense story generation, 2020.

[16] Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. A corpus and evaluation framework for deeper understanding of commonsense stories, 2016.

A Appendix

Model	Keywords: **thought take safely paranoid nervous made luckily grace flight crashes**
Seq2Seq bi-LSTM	**Grace** was **nervous** about her **flight**. She **thought** she would make a **flight** to the airport. **Luckily** she **made** a **flight** to the airport. She was afraid she had to **take** a **flight** to the airport. **Luckily** she **made** the **flight**!
Zero-shot GPT-2	
Weighted	The US is planning to spend a record $ 1.25 billion on "smart" technology this year, the Wall Street Journal reports, citing sources familiar with the matter and speaking on condition of anonymity to describe its plans. The Wall Street Journal, which published the article, reports that the plan to use a new generation of computer-aided surveillance technology [trimmed]
Weighted + Pop + Synys	This past summer the city of Toronto took its first steps on a $15-million renovation of the downtown core. With the addition of new sidewalks and sidewalks at both the east and west corner of the city, the new project was announced as a $2-billion, four kilometre project. [trimmed]
low-resource GPT-2	
Unperturbed	The **flight** attendants **made nervous flight** plans. They had been taking too much time to prepare for **flight**. **instead** they **made** a risky **flight** to the airport. The **flight** attendants **made** a mistake and flew back to the United States.
Weighted	I had been **paranoid** about flying and had been **nervous** about flying for a long time. I had taken a few **nervous nervous** shots in **flight**. I decided to leave the plane for a few days. I was worried that I might get caught out in the crash . I flew again on Thursday night. I had an anxiety attack that night. I **thought** my anxiety would have been worse if I took a risky shot. However, I took a few **nervous** shots in **flight**. I was **nervous** again.
Weighted + Pop + Synys	The **flight** was safe and sound. The plane had been crashed in an unsafe condition. The **flight** manager **made** the emergency call for an emergency call. The **flight** manager told the **flight** crew that they would **take** the precautionary step of taking the safety precautions.
Fine-tuned GPT-2	
Unperturbed	**Grace** was at her **flight** to LA. She was **nervous** about flying so **safely**. One morning she **made** a sudden stopover. Her **flight** was delayed by ten minutes. **Luckily** she was able to **go** home **safely**.
Weighted	**Grace** was at her **flight** to LA. She was **nervous** about flying so **safely**. One morning she **made** a sudden stopover. Her **flight** was delayed by ten minutes. **Luckily** she was alright and didn't **take** any more dangerous flights.
Weighted +Pop	**Grace** is on her **flight**. She **thought** her seat belt was too unsafe. When her friend came down on board, she was so **nervous** and **paranoid**. **Luckily** her friend managed to **safely** and **safely take** her seat belt off. **Grace** was very **relieved** with a good night's sleep.
Weighted + Pop + Synys	**Grace** was at her **flight** to LA. She was **nervous** about flying so **safely**. One morning **Grace** was **nervous** about taking off **safely**. When the plane landed she **made** an angry face. **Luckily** she **thought** it was safe and **thought** she did the right thing.

Table 4: **Examples of generated stories** Underlined words are alternative forms of the keywords, which we did not count when evaluating the Keywords Coverage and RBO.

Keyword	Synonyms
thought	remember, guess, consider,imagine, cogitate, opinion, recall, cerebration, intend, recollect, believe, thinking, mean, opine, view, idea, retrieve, conceive, persuasion, reckon, sentiment, suppose, think, cerebrate
take	hold, learn, engage, consider, pack, hire, subscribe, remove, shoot, admit, demand, occupy, choose, adopt, guide, submit, contain, postulate, ingest, direct, bring, withdraw, study, make, lead, assume, proceeds, return, aim, train, carry, exact, necessitate, have, issue, yield, accept, deal, read, ask, consume, strike, film, drive, select, takings, acquire, involve, payoff, contract, get, fill, require, lease, claim, convey, conduct, charter, rent, need
safely	
paranoid	paranoiac
nervous	aflutter, anxious, flighty, neural, uneasy, queasy, unquiet, spooky, skittish
made	hold, realise, take, constitute, throw, stimulate, seduce, cook, clear, puddle, piddle, crap, earn, build, do, pee, defecate, construct, fix, wee, make, give, stool, piss, urinate, attain, nominate, prepare, score, form, have, draw, establish, gain, create, pretend, induce, name, produce, reach, hit, cause, get, shit, realize, work, ready
luckily	fortunately, fortuitously
grace	Grace, decorate, ornament, deck, beautify, goodwill, thanksgiving, blessing, adorn, embellish, seemliness
flight	escape, flying,trajectory
crashes	dash, clangour, clang, clangor, wreck, crash, collapse, ram, clash, doss, clank, smash

Table 5: **Example of words in a Syns-BOW model**. We take use NLTK to find synonyms of the keywords, and filter those that are out of the vocabulary of our GPT-2 tokenizer.

Model	Keywords: **week test studying relieved Katie got class**
Seq2Seq bi-LSTM	**Katie** has a math **test** tomorrow. She has been **studying** all **week**. When she **got** to **class**, her teacher dropped her off. **Katie** was **relieved**. She was **relieved** when she **got** her **test**.
Zero-shot GPT-2	
Weighted	The US is planning to spend a record $1.25 billion on "smart" technology to improve the ability of soldiers to detect and fight back against the threat they face, US Defense secretary Ash Carter revealed on Thursday. The new military intelligence, which the US military said is designed for combat training, will be tested at Fort Still, Texas. [trimmed]
Weighted + Pop + Synys	This past summer the city of Toronto took its first steps on a $15-million renovation of the downtown core. With the addition of new sidewalks and sidewalks at both the entrance into downtown and the exit from the city's downtown core, the city's downtown is poised to become one of Canada's most livable cities. [trimmed]
low-resource GPT-2	
Unperturbed	Katherine **got** sick of being **relieved**. Katherine had been **studying** hard for weeks. She felt **relieved**. The **class** would be classed with anxiety. Catherine had a **class** tomorrow.
Weighted	**Katie got** a great exam. She **got** an hour to prepare for her new exam. **Katie got** her **class** time back. Her **test** exam **day** had been **studying** and **studying**. Kat's exam are now being used for **studying**. **Katie got relieved** and went to **class**!**Katie** was **relieved** and **relieved** that she was able to **take** classes. **Katie got** back to work **studying** again!
Weighted + Pop + Synys	Katherine **got** sick of being **relieved**. Katherine had been **studying** hard for **week**s. She felt **relieved**. The **class** would be **class** time. Katherine **got relieved**.
Fine-tuned GPT-2	
Unperturbed	**Katie** was taking her science **test** on Wednesday. After **studying** for about two weeks, **Katie** decided to do math homework. **Katie** was **relieved** when the **test** was finally done. **Katie got** a B+ in her math **test**. **Katie** is now **studying** very hard for the **test** tomorrow.
Weighted	Kate was always taking her classes to pass. On Tuesday she **got** a **test** for her math **test**. She was very **relieved**! She started thinking about her **test** tomorrow. Kate had a good **test**!
Weighted +Pop	**Katie** was doing an SAT. But her **test** was coming up soon. Her math teacher came over. She was **relieved** to have finally passed her **test**. **Katie** was **relieved** that she **got** a good grade on her **test**.
Weighted + Pop + Synys	Kate was always taking her classes to pass. On Tuesday she **got** a **test** for her math **test**. She was very **relieved**! She started thinking about her **test** tomorrow. Kate had a good **week**!

Table 6: **Examples of generated stories** Underlined words are alternative forms of the keywords, which we did not count when evaluating the Keywords Coverage and RBO.

Model	Keywords: **woke wished vomiting snowstorm school running instead ill hurricane day**
Seq2Seq bi-LSTM	I **woke** up one morning and was very **ill**. I **wished** I had to **go** back to **school** for a long time. **Instead**, I **woke** up with a fever. I **wished** I had to **go** back to **school** for a long time. **Instead** of staying home, I **wished** I had to **go** home .
Zero-shot GPT-2	
Weighted	The US is set to begin a review of the military's new drone program next month, and it appears that a few members of Congress are already taking the plunge. The US is to begin its first drone review of its new unmanned military aircraft, which the US military said is designed for combat purposes. US Senator John McCain is also reportedly considering [trimmed]
Weighted + Pop + Synys	This past summer the city of Toronto was given its first major public housing expansion. The new Toronto Centre for Community Development is the first of many to take shape as part of the City's Vision for Community Development program, which is intended to provide affordable, [trimmed]
low-resource GPT-2	
Unperturbed	The **school**'s teacher had been sick for days. He had been **vomiting** heavily. His cough had been worse than a flu attack. He needed to vomit heavily. The teachers had been able to cure him. However, he still felt sick. The **school**'s nurse was sick. She had been **vomiting**. Her doctor had been sick for months and needed to be replaced. The **school** decided to replace the teacher with one that was sicker.
Weighted	I **woke** up in the middle of the **day**. It turned out my **vomiting** had been a **hurricane day**. My teacher was sick at the **school** and sick for a month. I **woke** up in a feverish mood. I **woke** up **vomiting**. I didn't get sick. I was worried that I could never be better.
Weighted + Pop + Synys	The **school**'s teacher had been sick for days. He had been **vomiting** heavily. His cough had been worse than a normal **school day**.
Fine-tuned GPT-2	
Unperturbed	My daughter **woke** up this morning to a thunderstorm. She **wished** that she could sleep **instead**. In the storm it had snowed, but the **school** was **running**. The storm was **running** and she was sick. She was **vomiting** when she **woke** up and she was in the hospital.
Weighted	My daughter **woke** up this morning to snow. I **wished** she had not had a storm. When she did, her hair fell off as she was **running** for **school**. I had a horrible nightmare about being sick for days. **Instead** she **woke** up very badly from the bad weather.
Weighted +Pop	I was walking around campus one evening. I **woke** up feeling sick but my stomach felt alright. After **school** my friend invited me to bed **instead** of **running** into the **snowstorm**. I **wished** that **day** was better than this one.
Weighted + Pop + Synys	My daughter **woke** up this morning to snow. I **wished** she had not had a storm. When she did, her hair fell off as she was **running** for **school**. I had a horrible nightmare about being sick for days. **Instead** she **woke** up very badly from the illness.

Table 7: **Examples of generated stories** Underlined words are alternative forms of the keywords, which we did not count when evaluating the Keywords Coverage and RBO.

Model	Keywords: **walked turned park hot got go**
Seq2Seq bi-LSTM	I **walked** to the **park** to **go** to the **park**. When I **got** there, it was too **hot**. I **walked** to the **park** and **turned** on the air. It was too **hot**. I had to **go** to the **park** and **go** home.
Zero-shot GPT-2	
Weighted	The US is planning to spend a record $1.25 billion on a nuclear plant by 2020, the latest in a series of moves aimed at staving off a nuclear disaster. A joint US/Japan nuclear cooperation program is under way and a new US $1 billion contract for a nuclear plant in Japan is expected to be unveiled shortly. [trimmed]
Weighted + Pop + Synys	This past summer the city of Toronto took its first steps on a $15-million renovation of the downtown core. With the addition of new parks and sidewalks, new bike lanes, and a green space for businesses and residents, the new development is the first of its kind on the Eastside. [trimmed]
low-resource GPT-2	
Unperturbed	The walker **walked** up to the curb and began walking through the **park**. The walker **got** up off the grass and began to walk down the path. The walker was sweating and had trouble walking. The walker **got** off of the grass and **walked** down the walk path.
Weighted	The **park got hot**. They had a lot of **hot** drinks and they were going to be **hot**. They **walked** around the block. They **walked** up the steps into the **park**.
Weighted + Pop + Synys	The walker **walked** up to the curb and began walking through the **park**. The walker **turned** on its heel when it **turned** off the heat.
Fine-tuned GPT-2	
Unperturbed	My dog always goes to the **park**. One **day** we **got** to **park** outside. We **walked** in a **hot** and very **hot** spot. We **walked** out for lunch instead of going back. It **turned** out that it was too **hot** so she had to **go** home.
Weighted	My dog always goes to the **park**. One **day** we **got** to **park** outside. We **walked** in a **hot** and very **hot** spot. We **walked** out for lunch instead of going back home. We **turned** in to **go** for a walk instead.
Weighted +Pop	I **got** to the **park**. The temperature dropped to 95 degrees. I **walked** to the picnic table. It **turned** out to be too **hot** at that point.
Weighted + Pop + Synys	My dog always goes to the **park**. One **day** we **got** to **park** outside. We **walked** in a **hot** and very **hot** spot. We **walked** out for the walk anyway. I **turned** off the **hot** and **turned** off the **hot**.

Table 8: **Examples of generated stories** Underlined words are alternative forms of the keywords, which we did not count when evaluating the Keywords Coverage and RBO.