Event Transition Planning for Open-ended Text Generation

Qintong Li[♡,*] **Piji Li**[♣] **Wei Bi**[♣] **Zhaochun Ren**[◇] **Yuxuan Lai**[♡,†] **Lingpeng Kong**[♡,♠]
[♡]Department of Computer Science, The University of Hong Kong
[♣]Tencent AI Lab [◇]Shandong University
[♠]Shanghai Artificial Intelligence Laboratory
qtli@connect.hku.hk
{lipiji.pz, erutan.pkuicst}@gmail.com
victoriabi@tencent.com, zhaochun.ren@sdu.edu.cn
lpk@cs.hku.hk

Abstract

Open-ended text generation tasks, such as dialogue generation and story completion, require models to generate a coherent continuation given limited preceding context. The open-ended nature of these tasks brings new challenges to the neural auto-regressive text generators nowadays. Despite these neural models are good at producing human-like text, it is difficult for them to arrange causalities and relations between given facts and possible ensuing events. To bridge this gap, we propose a novel two-stage method which explicitly arranges the ensuing events in open-ended text generation. Our approach can be understood as a specially-trained coarse-to-fine algorithm, where an event transition planner provides a "coarse" plot skeleton and a text generator in the second stage refines the skeleton. Experiments on two open-ended text generation tasks demonstrate that our proposed method effectively improves the quality of the generated text, especially in coherence and diversity. The code is available at: https://github.com/qtli/EventPlanforTextGen.

1 Introduction

With the fast development of large-scale pre-trained models, considerable progress has been made in improving the quality of machine generated text (Radford et al., 2019; Rashkin et al., 2019a; Zhang et al., 2020b; Brown et al., 2020; Guan et al., 2021; Bakhtin et al., 2021). Today, machine learning models can do extremely well in generating text that looks human (Clark et al., 2021). The problem is still far from solved, however, as further reading of the machine-generated text often exposes defects such as self-contradiction and topic drifting (Bisk et al., 2020; Gao et al., 2020; Tan et al., 2021; Fan et al., 2019; Dou et al., 2021; Dziri et al., 2021). These issues are particularly serious in open-ended text generation tasks (e.g., story completion), where the model is asked to produce a coherent continuation which often involves multiple events, given limited preceding context.

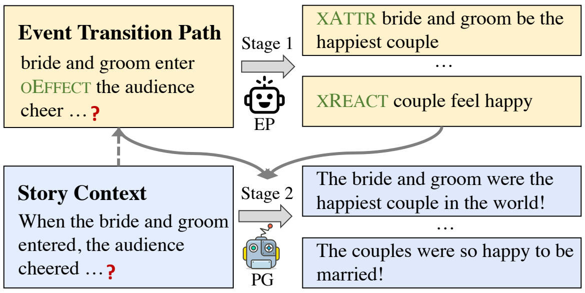
Event Transition Path
bride and groom enter
oEffect the audience
cheer ... ?
Stage 1
EP
xAttr bride and groom be the happiest couple
...
xReact couple feel happy
Story Context
When the bride and groom entered, the audience cheered ... ?
Stage 2
PG
The bride and groom were the happiest couple in the world!
...
The couples were so happy to be married!

Figure 1: An illustration of our planning based framework in story completion task. Given story context, we extract corresponding event transition path, and use model EP to develop potential ensuing event transition paths. The planned paths accordingly guide the path-aware text generation model PG.

To bridge this gap, we propose a two-stage method which explicitly models the event transitions in open-ended text generation. Multi-step generation has been adopted to control the generated content at a high level (Dong and Lapata, 2018; Ji et al., 2020; Xu, 2021). Different from previous works that rely on inflexible pattern retrieval, we leverage a generative model as an event transition planner in the first stage to boost the high-level coherence and causalities in open-ended text generation.

Specifically, in stage one, an event transition planner (§3.1) outlines a transition path of events starting from the ones extracted from the input context. In stage two, this path is used to ensure a relevant and sound continuation from the actual text generator (§3.2). This method can be understood as a specially-trained coarse-to-fine algorithm, where

[*]The majority of this work was done while the first author was interning at Tencent AI Lab.

[†]Corresponding author.

	Dialogue Generation	**Story Completion**
Input Context — Events	[1] my husband lost a job but i'm hoping he can find a full time job soon. — my husband lost job, I hope he find job [2] He will , I have faith. — I have faith [3] thank you so much! — thank you	[1] John got laid off from his company. — john get laid off [2] He was close to retirement age. —john is close retirement [3] John felt bored and listless his first week of unemployment. — john feel bored and listless [4] John decided to start a business of his own. — john decide start business
Target Output — Events	No problem. What kind of work does he do? — what work he do	He now has a flourishing online company. — john have a company
Event Transition Path	my husband lost job XATTR i hope he find job OREACT i have faith XREACT thank you OREACT what work he do	john get laid off XATTR john is close to retirement XREACT john feel bored and listless XREACT john decide start business XEFFECT john have a company

Table 1: Examples of event transition paths acquired from downstream tasks, i.e., dialogue generation and story completion. Events are marked in blue box.

an event transition planner provides a "coarse" plot skeleton and a path-aware text generator refines the skeleton. Figure 1 shows an illustration of our approach.

There are two main challenges in this method. First, the planer should produce high-quality and diverse paths that can generalize well to the unseen events at test time. For this challenge, we fine-tune a GPT-2 (Radford et al., 2019) on a large amount of event paths extracted from commonsense graphs (Sap et al., 2019), as well as from the training set of the specific task, aiming to extrapolate to event sequences that never appeared in these sources with the help of general knowledge stored in the large pre-trained model (Petroni et al., 2019; Lee et al., 2021). For the second challenge, the auto-regressive text generator need to work effectively under the supervision of the even transition path. We thus design an event query layer to absorb information from the planned paths and use the query layer to guide the text generation process.

We validate our method thorough extensive experiments on two standard open-ended text generation tasks, dialogue generation (Rashkin et al., 2019b) and story completion (Mostafazadeh et al., 2016). Our two-stage approach outperforms a strong knowledge enhanced GPT-2 baseline (Guan et al., 2020a) in both automatic and human evaluation metrics. Further analysis shows that the improvements of the event transition planning model come in particular from the high-level consistency and diversity in long and difficult generation cases.

2 Event Transition Path.

In this work, the event transition path is defined as an alternating sequence between events and relations, where an *event* is a subject-verb phrase, a *relation* is chosen from a pre-defined label set (e.g., OREACT - object reaction; XATTR - subject attribute) of a commonsense atlas (Sap et al., 2019). Table 1 shows some text examples and their corresponding event transition paths. We collect event transition paths from a commonsense atlas ATOMIC (Sap et al., 2019), as well as from the training set of the specific task, to train an event transition planner.

Sampling Paths from ATOMIC. We use everyday commonsense atlas ATOMIC (Sap et al., 2019) to acquire plenty event paths. ATOMIC is organized through 9 relations and 877k events (textual descriptions) of inferential commonsense, e.g., if "PersonX pays PersonY a compliment", then "PersonY will likely return the compliment". It has been demonstrated that ATOMIC is useful for open-ended text generation tasks, such as story generation (Guan et al., 2020b).

Besides, to increase the flexibility, we introduce a reverse relation (e.g., _XATTR) for each original relation (e.g., XATTR) so that a sampled path can contain reverse triplets. The intuition is that, in open-ended text generation, the narrative maybe in a reverse order. After explaining the event A, the author may want to introduce the subsequent event B. Meanwhile, if the author introduce the event B first, she/he may want to describe the event A as an explanation for the reason/motivation.

Finally, we collect sufficient event paths of variant lengths from ATOMIC via random walk-

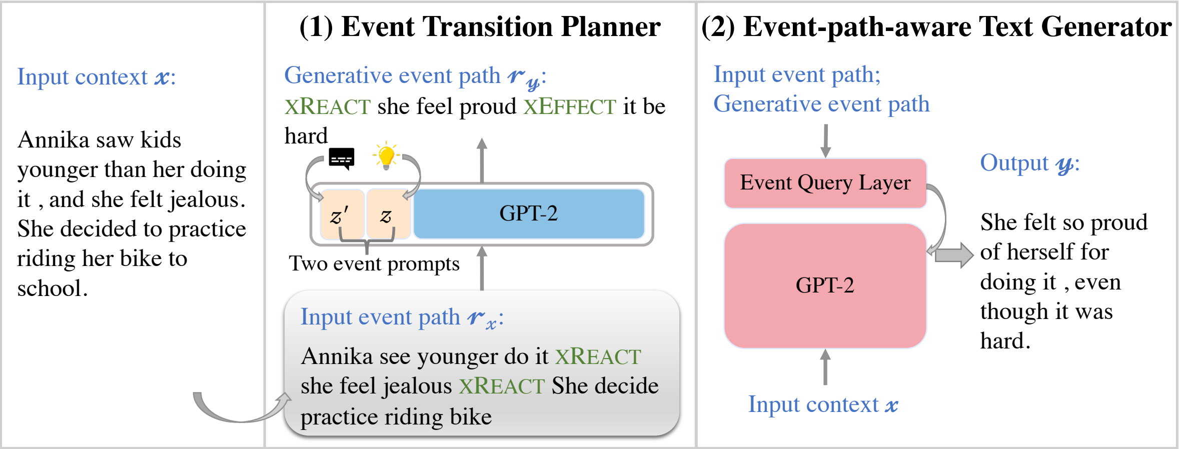
Input context $\boldsymbol{x}$:
Annika saw kids younger than her doing it , and she felt jealous. She decided to practice riding her bike to school.
(1) Event Transition Planner
Generative event path $\boldsymbol{r_y}$:
XREACT she feel proud XEFFECT it be hard
z'
z
GPT-2
Two event prompts
Input event path $\boldsymbol{r_x}$:
Annika see younger do it XREACT she feel jealous XREACT She decide practice riding bike
(2) Event-path-aware Text Generator
Input event path;
Generative event path
Event Query Layer
GPT-2
Input context $\boldsymbol{x}$
Output $\boldsymbol{y}$:
She felt so proud of herself for doing it , even though it was hard.

Figure 2: Overall architecture of the proposed coarse-to-fine framework. It consists of two components. (1) **Event Transition Planner**: given a input context, it first extracts corresponding event path and then generates possible ensuing event path. The planner directly inherits the pre-trained parameters from GPT-2; (2) **Event-path-aware Text Generator**: another GPT-2-based generator is applied to generate a natural language sentence by attending to input context and explicit event transition path.

ing[1]. We split the sampled paths into training/validation/test with the ratio of 18:1:1. We use these sampled paths to optimize the event transition planner which is responsible for generative event planning (see §3.1). The statistics of sampled paths are shown in Table 6 of Appendix A. We display several examples of the randomly sampled event transition paths in Table 7 of Appendix A.

Extracting Paths from Specific Dataset. We use two kinds of event transition paths. A general kind is obtained from random walking on a daily commonsense graph, ATOMIC, as mentioned above. Another kind is extracted from the natural language instances of downstream datasets, which is used for the training and prediction stage of task-specific event planning. For example, given the inputs, "When the bride and groom entered, the audience cheered", the extracted event path is "bride and groom enter OEFFECT audience cheer".

In detail, for each sentence, to ensure the extracted events have complete semantics and keep a similar format with the events in ATOMIC, we use ASER event extractor tool[2] to distil events for all sentences of downstream datasets. We further predict the relations between these events, linking these isolated events as event transition paths. Specifically, we train a BERT (Devlin et al., 2019) classifier using event triples and relations in ATOMIC. The sizes of training/validation/test instances are 639,045/35,503/35,502, respectively. We finally achieve a accuracy score of 85% on the test set for the relation prediction.

3 Methodology

We focus on the conditional language modeling problem in open-ended text generation tasks. Formally, given an input context $\boldsymbol{x}$, models are required to generate a sentence $\boldsymbol{y}$ that is consistent with input context and not contradicts itself.

In this work, we propose a two-stage model for the generation process. In the first stage, we extract the starting event sequence $\boldsymbol{r}_x$ from the input context and employ the event transition planer to generates subsequent event transition path $\boldsymbol{r}_y$ based on $\boldsymbol{r}_x$. In the second stage, the output text is generated from an auto-regressive model conditioning on the path and the preceding context $\boldsymbol{x}$.

Figure 2 gives an overview of our coarse-to-fine framework for open-ended text generation. In a nutshell, we first fine-tune a GPT-2 on event transition sequences as an event planner (i.e., a conditional generative model for event paths). This fine-tuning involves event transition sequences extracted from both commonsense graphs and the training set. We then build a path-aware text generator with an event query layer specifically designed to refer to the planned path when generating the output.

3.1 Generative Event Transition Planner

In this section, we describe the event transition planner which completes the partial event path

[1] The hops of these sampled paths fall in between 1 and 5.
[2] https://hkust-knowcomp.github.io/ASER/html/index.html

given certain input context. Pre-trained language models can be good representation learners of relational knowledge (Petroni et al., 2019; Bosselut et al., 2019). In our model, we choose GPT-2 (Radford et al., 2019) as the backbone of our event transition planner.

Specifically, we first fine-tune GPT-2 with large-scaled event transition paths sampled from ATOMIC (Sap et al., 2019). After that, we fine-tune the resulting model in addition on the event transitions extracted from the training corpus, so that the planner is aware of general transitions in the commonsense while focusing on the transitions in the specific domain in the meantime.

In preliminary experiments, we find that directly running a full fine-tuning (i.e., updating all GPT-2 parameters) leads to a drop in the final performance. We suspect the reason is the full fine-tuning flushes out the original general knowledge from the large-scale pre-training (Chen et al., 2019; Lee et al., 2020; Chen et al., 2020).

To overcome this drawback, we prepend a trainable continuous event prompt $\boldsymbol{z}$ to the input path $\boldsymbol{r} = [\boldsymbol{r}_x; \boldsymbol{r}_y]$ of every transformer layer in event transition planner, as prefix-tuning (Li and Liang, 2021) does. A trainable matrix $\mathbf{U}_\theta$ with parameters θ is randomly initialized to embed event prompt $\boldsymbol{z}$. The aim is to use parameters θ introduced by $\boldsymbol{z}$ to store event transition patterns from ATOMIC. Then the representation of each input event transition path $\boldsymbol{r}$ is prompted as $\boldsymbol{r}' = [\boldsymbol{z}; \boldsymbol{r}]$. To increase training speed and performance robustness, we apply an additional linear reparameterization function on $\mathbf{U}_\theta$.

$$\mathbf{U}_\theta = \mathit{FFN}_\theta\ (\mathbf{U}_\theta^{'}), \tag{1}$$

where $\boldsymbol{U}_\theta^{'}$ is another randomly initialized matrix with smaller dimension, *FFN* is a large feedforward neural network (Vaswani et al., 2017). We perform gradient updates on the following log-likelihood objective:

$$\max_\theta \ \log(\boldsymbol{r}_y \mid [\boldsymbol{z}; \boldsymbol{r}_{<y}]) = \sum_{y \in \boldsymbol{z}_{\text{idx}}} \log \ \mathit{EP}_{\phi,\theta}(\boldsymbol{r}_y \mid \mathbf{h}_{<y}), \tag{2}$$

where ϕ denotes the pre-trained parameters from the backbone LM of event transition planner, θ denotes the newly introduced parameters for the event prompt, $\boldsymbol{z}_{\text{idx}}$ denotes the index sequence of the event prompt, *EP* is short for event transition planner, and $\mathbf{h}_{<y}$ denotes the hidden states calculated by the trainable event prompt matrix and activation layers of the backbone LM:

$$\mathbf{h}_y = \begin{cases} \mathbf{U}_\theta[y, :], & \text{if } y \in \boldsymbol{z}_{\text{idx}}, \\ \boldsymbol{LM}_\phi(\boldsymbol{r}_y \mid \mathbf{h}_{<y}) & \text{otherwise.} \end{cases} \tag{3}$$

Similar to the above event prompting technique, for the paths from downstream dataset, we prepend another event prompt $\boldsymbol{z}'$ to the $\boldsymbol{r}'$ and only optimize the parameters introduced by $\boldsymbol{z}'$. This effectively preserves the newly-learned event transition patterns from ATOMIC and continuously adapts the event transition planner to different downstream event transition patterns.

3.2 Event-path-aware Text Generation

Current state-of-the-art systems for open-ended text generation are based on fine-tuning pre-trained language models with different downstream datasets. Although text generation fluency is usually not a crucial issue nowadays, topic-related mistakes (Dou et al., 2021) such as off-prompt and self-contradiction are common. We therefore integrate the event transition paths produced by the planner into the text generation model via an event query layer using the multi-head attention mechanism (*MHA*).

The event query layer is built on top of the stacked transformer layers, aiming to explicitly induce the expected output with event transition paths. The input of the event query layer is the event transition path $\boldsymbol{r}$ given the current input $\boldsymbol{x}$. $\boldsymbol{r}$ not only summarizes the event transition in $\boldsymbol{x}$, also indicates possible event path following $\boldsymbol{x}$. The structure of the event query layer resembles the transformer layer. Its output serves as the key and value vectors in the multi-head attention mechanism, which computes another attention vector $\mathit{MHA}(\boldsymbol{r})$. We concatenate two multi-head attention vectors and derive the final event-path-aware attention vector $\mathbf{m}$:

$$\mathbf{m} = \mathit{MLP}([\mathit{MHA}(\boldsymbol{x}); \mathit{MHA}(\boldsymbol{r})]), \tag{4}$$

where $\mathit{MHA}(\boldsymbol{x})$ is the output from the multi-head attention function of the original transformer layer, $\mathit{MHA}(\boldsymbol{r})$ is the output from the event query layer. The event-path-aware attention vector $\mathbf{m}$ replaces the original multi-head attention vector $\mathit{MHA}(\boldsymbol{x})$ and participates the remaining calculation of the language model.

The optimization of the event-path-aware text generator is the standard cross-entropy objective: $\textit{CrossEntropy}(\boldsymbol{y}_j \mid \boldsymbol{y}_{<j}, \boldsymbol{x}, \boldsymbol{r})$.

3.3 Implementation Details

We base our event planner and event-plan-aware text generator on pre-trained GPT-2-small models[3]. The event prompt length during training ATOMIC event transition paths are set to 5 according to pilot study. We inject and optimize the event query layer on the last layer of the stacked Transformers. When training the event-path-aware text generator, event path $\boldsymbol{r}_y$ is derived from the ground truth. During inference, $\boldsymbol{r}_y$ is the prediction from event transition planner given the input event transition path $\boldsymbol{r}_x$. More details are elaborated in Appendix B.

4 Experiments

We conduct experiments on two open-ended text generation tasks, dialogue generation and story completion, to answer the following questions:

- **RQ1**: How to develop a better event transition planner?
- **RQ2**: Whether the integration of event transition paths enhances the open-ended text generation?
- **RQ3**: How do the event transition paths benefit text generation?

4.1 Evaluated Tasks

- **Story Completion** requires models to complete a story given the first few sentences. We evaluated our framework on ROCSTORIES (Mostafazadeh et al., 2016), which contains 98k five-sentence stories. Our default setting is to predict the last sentence given the first four ones.
- **Dialogue Generation** aims to generate reasonable and human-like responses given the dialogue history. We evaluated our framework on EMPATHETICDIALOGUES (Rashkin et al., 2019b) which consists of 25k conversations grounded in prespecified situations.

4.2 Event Transition Planning (RQ1)

We compare our event transition planner, named as PLANGENERATION, with fine-tuned pre-trained GPT-2 (Radford et al., 2019) and several ablation settings, investigating *how to develop a better event transition planner*.

Specifically, the compared settings include:

- **GPT-2** is a pre-trained GPT-2 model (Radford et al., 2019) directly fine-tuned on the event paths extracted from specific tasks, i.e., dialogue generation or story completion in our work.
- **PLANGENERATION** is our proposed event planning method, which explores a two-stage fine-tuning on event transition paths from ATOMIC (Sap et al., 2019) and the downstream task, equipping with the proposed event prompting module.
- **w/o PROMPT** is our proposed method without the event prompting module, but still using the two-stage fine-tuning strategy.
- **w/o TUNING ON ATOMIC** is our proposed method without the first-stage fine-tuning on the event paths extracted from external commonsense atlas ATOMIC.
- **PLANRETRIEVAL** is a retrieval based planning methods, which employs the BM25 ranking function (ROBERTSON et al., 1995) to retrieve from the paths extracted from the training sets according to the given context.

Results. We use **BLEU** (Papineni et al., 2002) and **DIST** (Li et al., 2016) as the automatic metrics to evaluate the generated sentences in terms of the coherence and diversity, respectively. BLEU evaluates n-gram overlap between generation and ground truth. BLEU scores will become extremely low for large n. We thus experiment with $n \in \{1, 2, 4\}$. DIST measures the ratio of distinct n-grams to all the generated n-grams from the perspective of the generation diversity. For DIST metric, we adopt $n \in \{1, 2\}$. The experimental results are shown in Table 2. The dataset needed in this section consists of event transition paths sampled from ATOMIC and extracted from downstream datasets. i.e., ROCSTORIES and EMPATHETICDIALOGUE. The details of event transition paths are shown in §2 and Appendix A.

On both dialogue generation and story completion tasks, our proposed PLANGENERATION greatly outperforms baseline GPT-2 on event planning coherence (BLEU) and event path diversity (DIST). Specifically, on two downsteam tasks, our event transition planner PLANGENERATION surpasses the fine-tuned GPT-2 by 3.09 and 3.53 on BLEU-1, 0.31 and 0.30 on DIST-1. This improvement indicates that (1) the two-stage event prompting module could endow event transition planner powerful abilities on predicting the ensuing event paths; (2) enhanced with the large-scale event transition patterns from ATOMIC, our event transition

[3] https://huggingface.co/gpt2

Tasks	Methods	BELU-1	BLEU-2	BLEU-4	DIST-1	DIST-2
Dialogue Generation	GPT-2	23.43	11.50	3.31	1.57	4.18
	PlanGeneration (Ours)	**26.52**	**12.38**	3.29	**1.88**	**5.52**
	w/o Prompt	23.58	11.85	**3.58**	1.80	5.13
	w/o Tuning on Atomic	19.82	7.90	1.81	1.16	2.54
	PlanRetrieval	0.75	0.14	0.00	13.05	39.52
Story Completion	GPT-2	15.98	7.19	1.08	5.53	17.44
	PlanGeneration (Ours)	**19.51**	**9.01**	**1.35**	**5.83**	**17.48**
	w/o Prompt	13.64	6.14	1.12	4.71	15.77
	w/o Tuning on Atomic	12.74	4.61	0.47	6.08	12.27
	PlanRetrieval	1.28	0.15	0.00	11.88	37.70

Table 2: Experimental results on event transition planning. For detailed description about the compared models, please refer to §4.2.

planner becomes more creative and produces more diverse outcomes.

Considering the ablation settings, without tuning on Atomic (w/o Tuning on Atomic) or without the event prompting module (w/o Prompt), the method performs worse on both tasks and across almost all metrics. The limited performance of w/o Tuning on Atomic suggests the necessity and effectiveness of learning general event transition patterns from Atomic before optimizing on task-specific event paths. Tuning on Atomic event patterns could make event transition planner get familiar with the event-path-like language and generalize well on unseen event patterns. Compared to ablation model w/o Prompt, EventPlanning is comparatively more effective. This is because when optimizing on event paths of target tasks, the proposed event prompt protects the parameters of pre-trained language model from drastic change when training with event transition paths. This comparison confirms our intuition that event prompting module could improve event planning performance without destroying the eventual commonsense stored in pre-trained parameters. It provides a more flexible approach to blend the event transition patterns in both Atomic and specific tasks with the pre-trained GPT-2 model.

We also attempt a variation of our PlanGeneration method, i.e., PlanRetrieval. We can see that the BELU scores of PlanRetrieval are substantially lower that the generation based methods. The main reason is that the target event paths are flexible, infinite, and task-related. Many transition patterns are not seen in the training data or external commonsense graph.

4.3 Event-path-aware Text Generation (RQ2)

In this section, we compare our overall framework EP-PG with several baselines to investigate *whether the integration of generative event transition paths benefits the open-ended text generation.*

We consider the following settings:

- **GPT-2** is a pre-trained GPT-2 model (Radford et al., 2019) fine-tuned on the task-specific dataset.
- **GPT-2-CS-FT** is a commonsense-enhanced GPT-2 model. By following Guan et al. (2020b), we conduct a first-stage post-training on the Atomic commonsense triples and then fine-tuning on task-specific dataset.
- **EP-PG** is our proposed framework, which is a fine-tuned GPT-2 model integrated with the event transition path produced from event transition planner PlanGeneration via an event query layer.
- **R-EP-PG** is another version of **EP-PG** to explore the proposed event query layer. The input event transition paths are produced by PlanRetrieval in a retrieval way.

Results. We consider the same evaluation metrics as in §4.2. As demonstrated in Table 3, EP-PG achieves the most satisfying performance among all settings on both tasks[4]. Integrated with the explicit guidance of the event transition paths, EP-PG produces more accurate open-ended generations with higher diversity.

Particularly, our proposed framework EP-PG consistently and significantly improves GPT-2 baseline for all tasks on content quality (BLEU) and diversity (DIST), showcasing the advantage of injecting event query layer on fine-tuned GPT-2. Without the explicit modeling of event transition

[4] P-value < 0.05 on BLEU-1, according to Padó (2006).

Tasks	Models	BLEU-1	BLEU-2	BLEU-4	DIST-1	DIST-2
Dialogue Generation	GPT-2	16.07	6.41	2.13	2.06	7.70
	GPT-2-CS-FT (Guan et al.)	16.43	6.83	2.31	2.16	8.28
	R-EP-PG	16.68	6.71	2.27	**2.21**	**8.44**
	EP-PG (Ours)	**16.74**	**6.94**	**2.39**	2.19	8.25
Story Completion	GPT-2	25.03	9.58	2.70	8.38	31.33
	GPT-2-CS-FT (Guan et al.)	25.09	9.64	2.72	8.07	30.68
	R-EP-PG	24.72	9.27	2.63	7.01	26.49
	EP-PG (Ours)	**25.47**	**9.71**	**2.74**	**8.99**	**34.48**

Table 3: Results of experiments on open-ended text generations. For detailed information about each compared model, please refer to §4.3.

paths, GPT-2-CS-FT which post-trains on commonsense triples only obtains a slight improvement or even performs comparable with GPT-2 model. EP-PG further improves generation performance from GPT-2-CS-FT across all metrics on the two tasks, highlighting the efficacy of long-range event planning via an additional event query layer.

Particularly remarkable are the relative differences between R-EP-PG and EP-PG. Although R-EP-PG manages to bring generations more diversity, but in most cases, EP-PG is more effective on content planning and informativeness due to generative event transition patterns in higher qualities. Moreover, R-EP-PG performs even worse than GPT-2 on story completion. This implies that low-quality event paths even damage the generations. Thus, a reliable event path is a key guarantee for effective downstream text generation.

4.4 Analysis: Event Transition Planning for Different Generation Scenarios (RQ3)

To further investigate *how do the event paths benefit text generation*, we analyse the effectiveness of event paths on differently difficult levels of generation, i.e., token-level and sentence-level.

Token-level. We first separate the test set into 5 groups according to the averaged target sentence lengths, and then observe the improvements of our proposed EP-PG over GPT-2 on BLUE-1 score. We find that our framework gains more on the longer instances in both story completion (from 0.4 to 1.3 on instances with more than 15 non-stop-words) and dialogue generation (from 0.3 to 0.9 on instances with more than 5 non-stop-words). We argue that the longer targets imply more sophisticated upcoming event transitions, where the guidance from the event transition planner becomes more important.

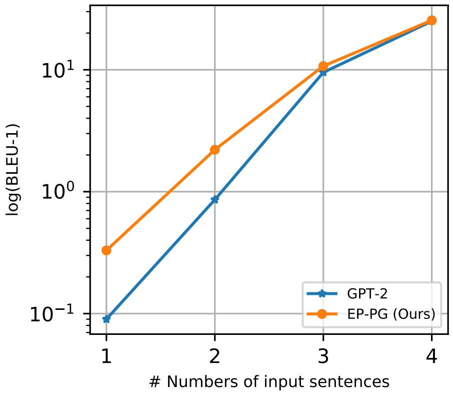

Figure 3: The log of BLEU-1 scores on story completion with different numbers of sentences as input.

Sentence-level. For story completion on five-sentence story dataset ROCStories, we further conduct experiments on EP-PG with various input sentences and output sentences, i.e., the numbers of input (output) sentence are 1 (4), 2 (3), 3 (2), and 4 (1), respectively. Figure 3 shows that, compared to GPT-2, the relative improvement proportion of EP-PG is nearly doubled on the most difficult setting where there is only one sentence as input. This improvement is much larger than the easiest situation where 4 sentences are input to the model. Despite less input context, EP-PG with event transition planning manages to performs better with smaller performance drop.

4.5 Human Evaluation

We set up a human evaluation as a complementary evaluation beyond automatic evaluation. For both tasks, we randomly select 100 samples from test set. For each sample, we compare three pairs of models: EP-PG versus GPT-2, GPT-2-CS-FT, and R-EP-PG. Each comparison is rated by three crowd workers, who are asked to give a preference (win, lose or tie) from two perspectives:

Tasks	Models	Coherence				Diversity			
		Win	Lose	Tie	κ	Win	Lose	Tie	κ
Dialogue Generation	Ours vs. GPT-2	45%	11%	44%	0.290	71%	10%	19%	0.226
	Ours vs. GPT-2-CS-FT	34%	10%	56%	0.286	54%	7%	39%	0.288
	Ours vs. R-EP-PG	32%	8%	60%	0.472	67%	11%	22%	0.291
Story Completion	Ours vs. GPT-2	45%	12%	42%	0.397	59%	10%	31%	0.220
	Ours vs. GPT-2-CS-FT	47%	17%	36%	0.387	56%	17%	27%	0.210
	Ours vs. R-EP-PG	43%	17%	40%	0.393	61%	6%	33%	0.340

Table 4: Manual evaluation results on downstream text generation. The scores indicate the percentages of Win, Lose or Tie when our model is compared with other baselines. κ denotes Fleiss' kappa (all are *fair agreement* or *moderate agreement*).

• **Coherence.** It indicates whether the inference is natural, relevant, and follows logically from the given context.

• **Diversity.** Particularly, for baseline models, we use beam search decoding with a top-k ($k = 5$) sampling scheme (Fan et al., 2018) and a softmax temperature τ ($\tau = 0.7$) to generate three inferences per sample. For our method EP-PG, its event transition planner first predicts three paths via the same beam decoding, then its text generator uses greedy decoding based on the generated three paths to produce three inferences per sample. During pair-wise comparison, we ask annotators to evaluate which model's predictions contain more reasonable and coherent event transition patterns.

The two aspects are independently evaluated and results are shown in Table 4. According to human evaluations, our proposed EP-PG significantly outperforms compared baselines in terms of both criteria on the test set of all datasets. Overall inter-rater agreement measured by Fleiss' kappa (Fleiss, 1971) and all the results show fair agreement ($0.2 \leq k \leq 0.4$) or moderate agreement ($0.4 \leq k \leq 0.6$). The results indicate that explicit incorporating event transition patterns yields significant improvement in generating coherence texts given the input context. Specifically, with guidance from different event transition paths, our method could produce more diverse and reasonable inferences.

4.6 Qualitative Study

Table 5 illustrates how our model tends to produce more contentful and coherent predictions compared to the other systems. In this story completion case, the generated event path successfully captures the correlations between *working out* and *pass physical test*, which further helps our model produce the most reasonable output, *Alex was able to pass the physical exam.* For the baseline without commonsense knowledge, GPT-2, is instead not related to the core context *failed the physical assessment*. Tuning on commonsense atlas Atomic, GPT-2-CS-FT produces informative inference but contradicts the context. The retrieval-based model R-EP-PG searches a related event transition *police officer*. However, its flexibility is limited by search space and cannot maintain a long-range event path, which is easy to produce hallucinated inference. More case analysis are stated in the Appendix C.

Story Context:
Alex was in training to be a police officer.
He was not in the best shape.
Alex failed the physical assessment.
Alex started working out.

Golden Event Path:
xEffect he take the test again xEffect he pass
Retrieved Event Path:
wants to be best police officer xWant tells person to stop
Generated Event Path:
xEffect Alex able get good shape xEffect Alex able pass physical test

Reference:
He took the test again and passed .
GPT-2:
Alex was able to get a good job.
GPT-2-CS-FT:
Alex made the squad.
R-EP-PG:
Alex was able to become a police officer.
EP-PG:
Alex was able to pass the physical exam.

Table 5: Case study on story completion. The three sections from top to bottom are the input context, the event transition plans, and inferences from our model and baseline models, respectively.

5 Related Work

Recent advances in pre-trained language models have resulted in impressive performances on open-domain text generation, such as story completion (See et al., 2019; Yao et al., 2019; Fan et al., 2019; Ippolito et al., 2020), dialogue generation (Rashkin et al., 2019b; Zhang et al., 2020b; Li, 2020; Vulić et al., 2021), question generation (Cheng et al., 2021; Wang et al., 2021), and so on. For example, in dialogue generation, Zhang et al. (2020b) design a trainable generative pre-trained transformer by training an autoregressive language model on large-scale Reddit context-response pairs with a maximum mutual information scoring function to improve diversity. Goldfarb-Tarrant et al. (2020) integrate semantic role labels and prompts into pre-trained BART (Lewis et al., 2020) during fine-tuning for prompt based story telling. In this paper, we focus on story completion and dialogue generation and build a generative coarse-to-fine method to generate open-ended text with explicit event transition paths.

Despite the success of generative pre-trained language models on a series of open-ended text generation tasks, they still suffer in maintaining coherence throughout multiple sentences due to the left-to-right word-by-word generation style (Fan et al., 2019; Yu et al., 2020). To alleviate this problem, one research direction adopts coarse-to-fine progressive text generation (Tan et al., 2021). This generation paradigm has been studied in many text generation systems for specific tasks, such as data-to-text generation (Moryossef et al., 2019; Puduppully and Lapata, 2021), storytelling (Goldfarb-Tarrant et al., 2020; Orbach and Goldberg, 2020), and dialogue generation (Xu et al., 2020a). Our work adopts a generative event transition planner that is trained on a large amount of event transition paths, aiming to arrange the ensuing events in open-ended text generation.

Another research direction incorporates external entities to guide the open-ended text generation (Guan et al., 2019; Zhang et al., 2020a; Dziri et al., 2021; Peng et al., 2021). Ji et al. (2020) and Xu et al. (2020b) retrieve entities from knowledge bases to control the generated content. However, the retrieval-based methods also suffer from the sparsity problem and the domain shift between external sources and downstream tasks (Wang et al., 2020). Guan et al. (2020b) integrate entity relations into pre-trained language model via additional tuning on entity triples. Even with such specialized learning, the resulted model still often stuck in logical errors or repeats pieces of narratives (Guan et al., 2020b; Peng et al., 2021). This phenomenon demonstrates the need for an intact inductive bias on organizing event transition patterns for open-ended text generation. Different from using event triples as additional training instances, our method explicitly maintains generative event transition paths to make the generation process more explainable and improve the coherence.

6 Conclusion

In this paper, we propose a novel two-stage method to improve high-level consistency and diversity in open-ended text generation. We design a special-trained event transition planner to explicitly arrange the ensuing events and introduce an event-path-aware text generator to exploit the event transition guidance for language generation. We investigate two open-ended text generation tasks, i.e., story completion and dialogue generation. Thorough experiments demonstrate that the explicit arrangement of event transition path indeed facilitate models to generate more coherent and diverse text in open-ended scenery. Besides, with the proposed event prompt and event query layer, our method could be extended to any other language models and open-ended generation tasks. A future line of investigation is to explore the effect of the proposed method on other open-ended tasks, such as commonsense question answering.

Acknowledgments

We thank the anonymous reviewers whose suggestions helped clarify this work. This research is supported in part by the National Natural Science Foundation of China (Grant No. 62106105, 61902219), the Shanghai Committee of Science and Technology, China (Grant No. 21DZ1100100), and the Tencent AI Lab Rhino-Bird Focused Research Program.

References

Anton Bakhtin, Yuntian Deng, Sam Gross, Myle Ott, Marc'Aurelio Ranzato, and Arthur Szlam. 2021. Residual energy-based models for text. *J. Mach. Learn. Res.*, 22:40:1–40:41.

Yonatan Bisk, Ari Holtzman, Jesse Thomason, Jacob Andreas, Yoshua Bengio, Joyce Chai, Mirella Lapata, Angeliki Lazaridou, Jonathan May, Aleksandr Nisnevich, Nicolas Pinto, and Joseph Turian. 2020. Experience grounds language. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8718–8735, Online. Association for Computational Linguistics.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Sanyuan Chen, Yutai Hou, Yiming Cui, Wanxiang Che, Ting Liu, and Xiangzhan Yu. 2020. Recall and learn: Fine-tuning deep pretrained language models with less forgetting. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7870–7881, Online. Association for Computational Linguistics.

Xinyang Chen, Sinan Wang, Bo Fu, Mingsheng Long, and Jianmin Wang. 2019. Catastrophic forgetting meets negative transfer: Batch spectral shrinkage for safe transfer learning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 1906–1916.

Yi Cheng, Siyao Li, Bang Liu, Ruihui Zhao, Sujian Li, Chenghua Lin, and Yefeng Zheng. 2021. Guiding the growth: Difficulty-controllable question generation through step-by-step rewriting. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5968–5978, Online. Association for Computational Linguistics.

Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A. Smith. 2021. All that's 'human' is not gold: Evaluating human evaluation of generated text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7282–7296, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Li Dong and Mirella Lapata. 2018. Coarse-to-fine decoding for neural semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 731–742, Melbourne, Australia. Association for Computational Linguistics.

Yao Dou, Maxwell Forbes, Rik Koncel-Kedziorski, Noah A Smith, and Yejin Choi. 2021. Scarecrow: A framework for scrutinizing machine text. *ArXiv preprint*, abs/2107.01294.

Nouha Dziri, Andrea Madotto, Osmar Zaïane, and Avishek Joey Bose. 2021. Neural path hunter: Reducing hallucination in dialogue systems via path grounding. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2197–2214, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

Angela Fan, Mike Lewis, and Yann Dauphin. 2019. Strategies for structuring story generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2650–2660, Florence, Italy. Association for Computational Linguistics.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Jianfeng Gao, Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Heung-Yeung Shum. 2020. Robust conversational AI with grounded text generation. *CoRR*, abs/2009.03457.

Seraphina Goldfarb-Tarrant, Tuhin Chakrabarty, Ralph Weischedel, and Nanyun Peng. 2020. Content planning for neural story generation with aristotelian rescoring. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4319–4338, Online. Association for Computational Linguistics.

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

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

Jian Guan, Xiaoxi Mao, Changjie Fan, Zitao Liu, Wenbiao Ding, and Minlie Huang. 2021. Long text generation by modeling sentence-level and discourse-level coherence. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6379–6393, Online. Association for Computational Linguistics.

Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6473–6480. AAAI Press.

Daphne Ippolito, David Grangier, Douglas Eck, and Chris Callison-Burch. 2020. Toward better storylines with sentence-level language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7472–7478, Online. Association for Computational Linguistics.

Haozhe Ji, Pei Ke, Shaohan Huang, Furu Wei, Xiaoyan Zhu, and Minlie Huang. 2020. Language generation with multi-hop reasoning on commonsense knowledge graph. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 725–736, Online. Association for Computational Linguistics.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. 2020. Mixout: Effective regularization to finetune large-scale pretrained language models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Kenton Lee, Kelvin Guu, Luheng He, Tim Dozat, and Hyung Won Chung. 2021. Neural data augmentation via example extrapolation. *ArXiv preprint*, abs/2102.01335.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pretraining for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Piji Li. 2020. An empirical investigation of pre-trained transformer language models for open-domain dialogue generation. *ArXiv preprint*, abs/2003.04195.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

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

Nasrin Mostafazadeh, Lucy Vanderwende, Wen-tau Yih, Pushmeet Kohli, and James Allen. 2016. Story cloze evaluator: Vector space representation evaluation by predicting what happens next. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 24–29, Berlin, Germany. Association for Computational Linguistics.

Eyal Orbach and Yoav Goldberg. 2020. Facts2Story: Controlling text generation by key facts. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2329–2345, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Sebastian Padó. 2006. User's guide to sigf: Significance testing by approximate randomisation.

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

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Xiangyu Peng, Siyan Li, Sarah Wiegreffe, and Mark Riedl. 2021. Inferring the reader: Guiding automated story generation with commonsense reasoning. *CoRR*, abs/2105.01311.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Ratish Puduppully and Mirella Lapata. 2021. Data-to-text generation with macro planning. *Transactions of the Association for Computational Linguistics*, 9:510–527.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2019a. Towards empathetic open-domain conversation models: A new benchmark and dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5370–5381, Florence, Italy. Association for Computational Linguistics.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2019b. Towards empathetic open-domain conversation models: A new benchmark and dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5370–5381, Florence, Italy. Association for Computational Linguistics.

SE ROBERTSON, S WALKER, S JONES, MM HANCOCK-BEAULIEU, and M GATFORD. 1995. Okapi at trec-3. *NIST special publication*, (500225):109–123.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. 2019. ATOMIC: an atlas of machine commonsense for if-then reasoning. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3027–3035. AAAI Press.

Abigail See, Aneesh Pappu, Rohun Saxena, Akhila Yerukola, and Christopher D. Manning. 2019. Do massively pretrained language models make better storytellers? In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 843–861, Hong Kong, China. Association for Computational Linguistics.

Bowen Tan, Zichao Yang, Maruan Al-Shedivat, Eric Xing, and Zhiting Hu. 2021. Progressive generation of long text with pretrained language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4313–4324, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Ivan Vulić, Pei-Hao Su, Samuel Coope, Daniela Gerz, Paweł Budzianowski, Iñigo Casanueva, Nikola Mrkšić, and Tsung-Hsien Wen. 2021. ConvFiT: Conversational fine-tuning of pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1151–1168, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bingning Wang, Ting Yao, Weipeng Chen, Jingfang Xu, and Xiaochuan Wang. 2021. Multi-lingual question generation with language agnostic language model. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2262–2272, Online. Association for Computational Linguistics.

Peifeng Wang, Nanyun Peng, Filip Ilievski, Pedro Szekely, and Xiang Ren. 2020. Connecting the dots: A knowledgeable path generator for commonsense question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4129–4140, Online. Association for Computational Linguistics.

Jun Xu, Zeyang Lei, Haifeng Wang, Zheng-Yu Niu, Hua Wu, and Wanxiang Che. 2020a. Enhancing dialog coherence with event graph grounded content planning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3941–3947. ijcai.org.

Peng Xu, Mostofa Patwary, Mohammad Shoeybi, Raul Puri, Pascale Fung, Animashree Anandkumar, and Bryan Catanzaro. 2020b. Controllable story generation with external knowledge using large-scale language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2831–2845.

Yang Xu. 2021. Global divergence and local convergence of utterance semantic representations in dialogue. In *Proceedings of the Society for Computation in Linguistics 2021*, pages 116–124, Online. Association for Computational Linguistics.

Lili Yao, Nanyun Peng, Ralph M. Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7378–7385. AAAI Press.

Changlong Yu, Hongming Zhang, Yangqiu Song, and Wilfred Ng. 2020. Cocolm: Complex commonsense enhanced language model. *CoRR*, abs/2012.15643.

Houyu Zhang, Zhenghao Liu, Chenyan Xiong, and Zhiyuan Liu. 2020a. Grounded conversation generation as guided traverses in commonsense knowledge graphs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2031–2043, Online. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020b. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

A Event Transition Path

The statistics about event transition paths sampled from ATOMIC are shown in Table 6. We display several examples of the sampled event transition paths in Table 7.

Total	**Training**	**Validation**	**Test**
4,016,468	3,614,981	200,752	200,735

Table 6: Numbers of the sampled event transition paths from ATOMIC.

Sampled Event Transition Paths of Variant Lengths
[1] PersonX earns a bachelor's degree xWANT PersonX wants to find a good job
[2] PersonX asks PersonY to join oWANT PersonY wants to be friends xREACT PersonY feels loved
[3] PersonX is inebriated _xATTR PersonX loses control of PersonX's car xREACT PersonX feels scared _oREACT PersonY takes PersonX by force xREACT PersonY feels triumphant

Table 7: Event transition paths sampled from daily commonsense reasoning atlas ATOMIC (Sap et al., 2019).

B Implementation Details

For all the systems, including the event transition planner and text generator in our proposed method, we employ the small version of GPT-2 model[5] which is a Transformer with 12-head, 12-layer, and hidden size of 768. The total parameter scalse is 117M. We use pre-trained GPT-2 Byte Pair Encoding (BPE) tokenizer with an extended vocabulary of 50,282 tokens to tokenize texts.

The event prompt length during training ATOMIC event transition paths, EMPATHETICDIALOGUES paths, and ROCSTORIES paths are 5, 5, and 10, respectively. The dimension of the randomly initialized smaller matrix $\mathbf{U}'_\theta$ in Eq.1 is 512.

The batch size is 128 using AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of 5e-5. We select the best checkpoint according to the perplexity on the development set of each task and apply early stopping on training where the patient value is set to 2. We adopt the pre-trained BERT-base model[6] to train the event relation classifier. All experiments are implemented by PyTorch framework (Paszke et al., 2017) and run on NVIDIA V100 GPUs. The training time of the event transition planner and event-path-aware text generator are less than 5 hours and 3 hours with 8 GPUs.

C Case Study

We qualitatively analyze our model predictions and find that although the proposed model outperforms the state-of-the-art baselines, many of predictions are still wrong. Table 8 shows several satisfying and unsatisfying predictions on the two datasets. One significant error originates from the weak alignment between event transition path and final prediction. For example, in the second case, despite "xEFFECT tommy be happy" is imperfect, the prediction "bought it" do not convey its information and makes co-reference mistake (the expected output is "bought them"). Another serious error type is event transition hallucination, where both the predicted event path and its corresponding

[5] `https://huggingface.co/gpt2`

[6] `https://huggingface.co/bert-base-uncased`

inference fail to maintain the logic coherence, such as the fourth case. These problems could be alleviated if we design a better format of event transition path which is easier to learn or improve the relation modeling between events and sentences.

	Input	**Corresponding Event Transition Path**
Good Case on Story Completion	**Context**:	
	Our granddaughter is two.	Our granddaughter be two
	Today she went to the doctor for a blood draw.	xEffect she go doctor for draw
	She did very well.	xEffect she do well
	Our daughter sent a photo of her licking a lollypop afterward.	oReact we daughter send a photo xEffect she lick lollypop
	Target:	
	We were very proud of her.	oEffect we be proud
	Prediction:	
	We were amused by the photo.	xEffect we get good photo xEffect we be happy
***Bad* Case on Story Completion**	**Context**:	
	Tommy wanted to buy a new computer.	tommy want buy new computer
	After some research he decided to build a PC himself.	xEffect After research decide build PC
	He found a site that spelled out compatible parts lists.	xEffect he find site
	He shopped around for the cheapest parts he could get.	xEffect he shop around part _xEffect he could get
	Target:	
	The PC he made was more powerful than computers twice its price.	xEffect he make powerful computer
	Prediction:	
	He finally found the perfect parts list and bought it.	xReact he find part xEffect tommy be happy
Good Case on Dialogue Generation	**Context**:	
	Hi, I joined a firm 6 months ago and then I got a promotion for Junior Manager.	i join firm ago xEffect i get promotion for manager
	Target:	
	Congratulations. That sounds like the fast track.	oReact that like fast track
	Prediction:	
	That's awesome! I bet you are excited!	oReact that be great xEffect you must be proud
***Bad* Case on Dialogue Generation**	**Context**:	
	I got my four year old daughter her first tricycle yesterday.	i get my daughter tricycle yesterday
	Target:	
	thats so sweet of you.	oReact that sweet
	Prediction:	
	Wow, that's a lot of fun. What kind of tricycle?	oEffect that be great xEffect what be tricycle

Table 8: Summary table of issues found through a qualitative analysis of our model predictions. The errors that occur in our model predictions are colored in blue.