A Causal Approach for Counterfactual Reasoning in Narratives

Feiteng Mu, Wenjie Li
The Department of Computing, The Hong Kong Polytechnic University, Hong Kong
{csfmu,cswjli}@comp.polyu.edu.hk

Abstract

Counterfactual reasoning in narratives requires predicting how alternative conditions, contrary to what actually happened, might have resulted in different outcomes. One major challenge is to maintain the causality between the counterfactual condition and the generated counterfactual outcome. In this paper, we propose a basic VAE module for counterfactual reasoning in narratives. We further introduce a pre-trained classifier and external event commonsense to mitigate the posterior collapse problem in the VAE approach, and improve the causality between the counterfactual condition and the generated counterfactual outcome. We evaluate our method on two public benchmarks. Experiments show that our method is effective. Code is available at https://github.com/mufeiteng/CausalCRN.

1 Introduction

Counterfactual reasoning in narratives (CRN) is commonly known as predicting how alternative events, contrary to what actually happened, might have resulted in different outcomes (Qin et al., 2019; Ashida and Sugawara, 2022). Specifically, given the observed narrative $S = (c, x, y)$, where c, x, and y denote the context, condition, and outcome, respectively, CRN considers how y' would be if keeping the context c unchanged while perturbing x to a similar but different x'. Figure 1 presents a case of CRN.

Even though it is considered a crucial component of intelligent systems (Pearl, 2009; Pearl and Mackenzie, 2018), only a few resources have been devoted to CRN. Some of the works (Hao et al., 2021; Chen et al., 2022; Li et al., 2023b) design dataset-specific heuristic methods, but they are actually abusing unique patterns, i.e., the feature of minimum editing, in the dataset, which limits the generality of their methods. Other works (Qin et al., 2019; Zhou et al., 2022) take advantage of the progress of pre-trained language models (PLMs), and fine-tune PLMs for CRN, i.e., learning the conditional distribution $p(\mathbf{y}'|\mathbf{c}, \mathbf{x}', \mathbf{S})$. Despite the success of simulating real examples, the conditional distribution is notorious for being susceptible to exploiting artifacts of the dataset, instead of learning to robustly reason about counterfactuals (Qin et al., 2019). For example, the models often directly copy the original y or learn to paraphrase y without acknowledging the counterfactual condition (Qin et al., 2019; Hao et al., 2021). As a result, the predicted counterfactual outcome y' usually conflicts with the counterfactual condition x'.

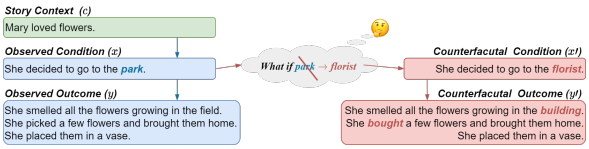


Figure 1: An example of counterfactual reasoning in narratives. The example comes from TimeTravel (Qin et al., 2019). The colored text in the counterfactual outcome denotes the modified parts.

Generally, CRN relies on the ability to find causality in narratives (Chen et al., 2022), i.e., y' should express a clear causal relation to x' to make it clear how the perturbation makes the observed outcome change. This problem naturally fits to be formulated with a causal mechanism (Pearl et al., 2016), which requires us to infer the background knowledge that is compatible with (c, x', y'). However, this is non-trivial as it involves estimating the posterior of the background knowledge. Luckily, with the variational technique (Kingma and Welling, 2013), we are able to use the background compatible with the observed S to approximate the posterior distribution. In fact, the variational process provides an approximation of the background of (c, x', y'), but it may face the problem of posterior collapse (Razavi et al., 2019). As a result, the generated y' may not be the precise effect of

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 6556–6569
August 11-16, 2024 ©2024 Association for Computational Linguistics

x', and the resulting model may be sub-optimal. To mitigate this problem, we further propose two intuitive strategies, which introduce a pre-trained classifier and commonsense causality, to enhance the causality between (c, x') and the generated y'.

In this work, we propose a causal approach for CRN. We utilize the variational process to approximate the *implicit* background of counterfactual scenarios. In addition, we devise two strategies to alleviate the posterior collapse problem in this variational process. First, inspired by research on natural language inference (Kang et al., 2018; Dziri et al., 2019), we want to ensure that the generated y' entails its true condition x'. In other words, the model should correctly learn the influence of the condition on the outcome. Therefore, we introduce a pre-trained classifier that estimates the likelihood of a text y entails an input (c, x). We use the Gumbel-softmax technique (Jang et al., 2016; Hu and Li, 2021) to enable gradient back-propagation. Second, we exploit COMeT (Hwang et al., 2021) to retrieve diverse event causality tailored for (c, x'), which allows for deducing plausible event sequences and provides an *explicit* background for the unobserved counterfactual outcome y'.

To summarize, we formulate CRN in a variational framework and introduce event causality and a pre-trained classifier to further improve the causality between x' and the generated y'. Our method is a general approach that is applicable to multiple tasks. The experiment proves the effectiveness of our method. We also study the practicality of the generated counterfactual narratives via a data augmentation experiment.

2 Related Work

Causality for NLP The research of causality aims to explore the causal relationships in the data (Yao et al., 2021). Recently, there has been a strong interest in utilizing causal inference to enhance current natural language understanding and generation. These works are mainly studied in event detect (Chen et al., 2021), text classification (Mu and Li, 2023), relation extraction (Liu et al., 2021a), etc. Another line of research attempts to equip the current text generation with counterfactual reasoning ability. These works involve fields such as dialogue generation (Ou et al., 2022), machine translation (Liu et al., 2021b), style transferring (Hu and Li, 2021), etc. Yet there have been few works that apply causal perspective to counterfactual reasoning in narratives. We adapt the ideas of the above works and propose additional strategies to improve the causality between the counterfactual condition and the generated counterfactual outcome.

Counterfactual Story Generation Counterfactual story generation aims to revise an original story ending guided by a modified condition (Qin et al., 2019). Previous works (Chen et al., 2022; Li et al., 2023b) usually utilize a two-stage approach. Generally, in the first stage, each token in the original story ending is determined if it requires modification. In the second stage, the identified words are modified to align with the story logic under the counterfactual condition. However, it is difficult to migrate this dataset-specific framework to other datasets (Ashida and Sugawara, 2022). Instead, motivated by counterfactual reasoning (Pearl and Mackenzie, 2018), we propose a general framework for counterfactual reasoning in narratives.

Knowledge-Enhanced Narrative Generation Narrative generation requires models to produce fluent and coherent stories under predefined conditions. Many studies inject structured knowledge into the generation process. For example, Ji et al. (2020) introduces explicit knowledge from ConceptNet, and Mu and Li (2022) introduces structural event causality to improve narrative generation. These works prove that external knowledge helps to enhance the coherence between the input and output text. Motivated by these works, we introduce commonsense causality tailored for the counterfactual condition to improve the causality between (c, x') and the generated y'.

3 Methods

3.1 Problem Setting with Causal Mechanism

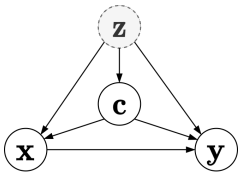


Figure 2: The proposed structural causal model. The dashed circle indicates that the variable is latent, while the solid circle indicates that the variable is observed.

Given a narrative $S = (c, x, y)$, we perturb x into a counterfactual condition x' and want to predict the new outcome y'. To solve this problem, we need to speculate on the background knowledge compatible with (c, x', y'), which allows us

to predict the precise effect of x'. This problem is naturally suitable to be expressed with a causal mechanism. Figure 2 shows the structural causal model (SCM) (Pearl, 2009) that describes the generation process of narratives. Here the latent variable $\mathbf{z}$ denotes the unobserved background knowledge. When people write stories, they usually generate narrative events one by one based on their knowledge. This is reflected in the graph as $\mathbf{z}$ affects all of $(\mathbf{c}, \mathbf{x}, \mathbf{y})$. The similar setting can be seen in (Chen et al., 2022), in which they unify the unobserved knowledge and the context into one confounder variable. On the contrary, we model the unobserved knowledge independently to directly infer the information of $\mathbf{z}$.

The SCM thus defines a joint distribution:

$$p(\mathbf{y}, \mathbf{x}, \mathbf{c}, \mathbf{z}) = p(\mathbf{y}|\mathbf{x}, \mathbf{c}, \mathbf{z})p(\mathbf{x}, \mathbf{c}|\mathbf{z})p(\mathbf{z}), \quad (1)$$

where $p(\mathbf{z})$ is a standard Gaussian distribution following common practices. Similarly, conditioned on the observed S, the joint distribution of the counterfactual scenario is defined as:

$$\begin{aligned} &p(\mathbf{y}', \mathbf{x}', \mathbf{c}, \mathbf{z}|\mathbf{S}) = \\ &\quad p(\mathbf{y}'|\mathbf{x}', \mathbf{c}, \mathbf{z}, \mathbf{S})p(\mathbf{x}', \mathbf{c}|\mathbf{z}, \mathbf{S})p(\mathbf{z}|\mathbf{S}), \end{aligned} \quad (2)$$

where $p(\mathbf{y}'|\mathbf{x}', \mathbf{c}, \mathbf{z}, \mathbf{S})$ is the decoder model requiring us to infer $\mathbf{z}$ from all (S, c, x', y') data. However, the inference of $\mathbf{z}$ involves estimating the posterior distribution of the knowledge, i.e., $p(\mathbf{z}|\mathbf{c}, \mathbf{x}', \mathbf{y}', \mathbf{S})$. We next introduce our basic variational process to approximate the distribution.

3.2 The Basic Variational Objective

3.2.1 Variational Inference

Our basic objective follows the common VAE approach (Kingma and Welling, 2013). By introducing the approximate network $q(\mathbf{z}|\mathbf{c}, \mathbf{x}', \mathbf{y}', \mathbf{S})$, a lower bound of the model's marginal log-likelihood (that marginalizes out $\mathbf{z}$) is:

$$\begin{aligned} &\log p(\mathbf{y}'|\mathbf{c}, \mathbf{x}', \mathbf{S}) = \log \int_{\mathbf{z}} p(\mathbf{y}', \mathbf{z}|\mathbf{c}, \mathbf{x}', \mathbf{S}) \\ &\geq \textbf{ELBO} = \mathbf{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{c}, \mathbf{x}', \mathbf{y}', \mathbf{S})} \log \frac{p(\mathbf{y}', \mathbf{z}|\mathbf{c}, \mathbf{x}', \mathbf{S})}{q(\mathbf{z}|\mathbf{c}, \mathbf{x}', \mathbf{y}', \mathbf{S})}. \end{aligned} \quad (3)$$

For simplicity, we denote $q(\mathbf{z}|\mathbf{c}, \mathbf{x}', \mathbf{y}', \mathbf{S})$ as $q(\mathbf{z}|\cdot)$. Then, according to Equation 2, we have:

$$\begin{aligned} &\textbf{ELBO} \\ &= \mathbf{E}_{\mathbf{z}\sim q(\mathbf{z}|\cdot)}[\log \frac{p(\mathbf{y}', \mathbf{z}, \mathbf{c}, \mathbf{x}'|\mathbf{S})}{q(\mathbf{z}|\cdot)} - \log p(\mathbf{c}, \mathbf{x}'|\mathbf{S})] \\ &\approx \mathbf{E}_{\mathbf{z}\sim q(\mathbf{z}|\cdot)}[\log p(\mathbf{y}'|\mathbf{z}, \mathbf{c}, \mathbf{x}', \mathbf{S}) + \log p(\mathbf{c}, \mathbf{x}'|\mathbf{z}, \mathbf{S})] \\ &\qquad - \mathrm{KL}[q(\mathbf{z}|\cdot)||p(\mathbf{z}|\mathbf{S})], \end{aligned} \quad (4)$$

where $p(\mathbf{c}, \mathbf{x}'|\mathbf{S})$ is a constant for the given dataset and independent of the parameterized model. Hence, given the labeled set $\mathcal{D}$ which contains all (S, c, x', y') examples, our basic objective is:

$$\begin{aligned} \mathcal{L}_{\text{VAE}} = -\frac{1}{|\mathcal{D}|} \sum_{\mathcal{D}} &\mathbf{E}_{\mathbf{z}\sim q(\mathbf{z}|\cdot)}[\log p(\mathbf{y}'|\mathbf{z}, \mathbf{c}, \mathbf{x}', \mathbf{S}) \\ &+ \lambda_x \log p(\mathbf{c}, \mathbf{x}'|\mathbf{z}, \mathbf{S}) + \lambda_k \mathrm{KL}[q(\mathbf{z}|\cdot)||p(\mathbf{z}|\mathbf{S})]. \end{aligned} \quad (5)$$

λ_x and λ_k are hyper-parameters. We use the cyclic schedule (Li et al., 2020) to anneal λ_k from 0 to 1 to avoid excessive regularization of the KL term.

3.2.2 $p(\mathbf{y}'|\mathbf{z}, \mathbf{c}, \mathbf{x}', \mathbf{S})$ vs. $p(\mathbf{y}'|\mathbf{c}, \mathbf{x}', \mathbf{S})$

Current generative models follow the autoregressive paradigm, but suffer from exposure bias. Note that (c, x, y) and (c, x', y') have similar content. When performing inference, given the input (S, c, x'), $p(\mathbf{y}'|\mathbf{c}, \mathbf{x}', \mathbf{S})$ has no information about the gold y', so it may paraphrase y. Different from $p(\mathbf{y}'|\mathbf{c}, \mathbf{x}', \mathbf{S})$, we encode y' into $q(\mathbf{z}|\cdot)$, and use the KL term to bridge the gap between $q(\mathbf{z}|\cdot)$ and $p(\mathbf{z}|\mathbf{S})$. When performing inference, we sample $\mathbf{z} \sim p(\mathbf{z}|\mathbf{S})$ and feed it into $p(\mathbf{y}'|\mathbf{z}, \mathbf{c}, \mathbf{x}', \mathbf{S})$. This can somewhat alleviate the issue of exposure bias and mitigate the problem of paraphrasing y.

3.2.3 Model Implementation

We use PLMs, e.g., BART (Lewis et al., 2019), as backbone to implement $p(\mathbf{y}'|\mathbf{z}, \mathbf{c}, \mathbf{x}', \mathbf{S})$. We first encode the input part (S, c, x') into the context vectors $\mathbf{H}_C = \text{BARTEncoder}(S, c, x')$, where $\mathbf{H}_C \in \mathcal{R}^{l\times d}$, l is the total length of $[S; c, x']$, d is the hidden size. To fuse $\mathbf{z} \sim q(\mathbf{z}|\cdot)$ into PLMs, as suggested in (Li et al., 2020), we concatenate $\mathbf{z}$ with $\mathbf{H}_C$, and pass it into the decoder for autoregressive learning. The hidden state of t-th time step of the target sequence $\mathbf{h}_{y_t}$ is computed by:

$$\mathbf{h}_{y_t} = \text{BARTDecoder}(Y_{<t}, [\mathbf{H}_C; \mathbf{z}]). \quad (6)$$

The word distribution of t-th time-step over the standard vocabulary V is:

$$P(y_t|Y_{<t}) = \text{softmax}_V(\mathbf{W}_v \mathbf{h}_{y_t} + b). \quad (7)$$

To implement $q(\mathbf{z}|\cdot)$ and $p(\mathbf{z}|\mathbf{S})$, we approximate them to Gaussian distributions. We use the pre-trained BARTEncoder to initialize different text encoders, which are used to encode S and (c, x', y', S). Following several linear layers, we obtain the mean and log-variance of two distributions, which are used to calculate the KL loss. To implement $P(\mathbf{c}, \mathbf{x}'|\mathbf{z}, \mathbf{S})$, we adopt the in-batch contrastive learning. For the positive example (c, x', z, S), we collect different $\bar{x'}$ from the mini-batch and regard $(c, \bar{x'}, z, S)$ as negative examples. This actually matches the $z \sim q(\mathbf{z}|\cdot)$ with the counterfactual x', and differitiates it from negatives.

Then the representations of examples are projected into scalar values for binary classification.

Training with the above base objective alone can lead to posterior collapse, i.e., the KL term tends to be zero. As a result, the decoder will ignore the information from $q(\mathbf{z}|\cdot)$, and the generated text is not the precise result of x'. We next introduce our two strategies, which introduce the pre-trained classifier and external event causality to improve the causality between (c, x') and the generated y'.

3.3 Introducing the Pre-trained Classifier

Intuitively, we expect that the generated y' truly entails its condition x'. To achieve this goal, we pre-train a classifier $f([c, x, y])$ that estimates the likelihood of the input (c, x) entailed by the output y. Motivated by (Chen et al., 2022), we use the training set of the used datasets to obtain positive and negative examples. For example, given the example (c, x, y, x', y'), (c, x') should entail by y' but contradict with y, and (c, x) should entail by y but contradict with y'. That is, (c, x, y) and (c, x', y') are positive, and (c, x', y) and (c, x, y') are negative. We initialize $f(\cdot)$ with BARTEncoder to keep the embedding space the same as the generator. Once the classifier is pre-trained, it is frozen in the later process of training our generator.

Then, we train the generator so that its predicted outcome entails the corresponding condition with a high likelihood measured by the classifier:

$$\mathcal{L}_{\text{Cla}} = -\frac{1}{|\mathcal{D}|}\sum_{\mathcal{D}} \mathbf{E}_{\mathbf{z}\sim q(\mathbf{z}|\cdot),\tilde{y}\sim p(\mathbf{y}|\mathbf{z},\mathbf{c},\mathbf{x},\mathbf{S}')}[\log f([c, x, \tilde{y}]) + \log(1 - f([c, x', \tilde{y}]))], \tag{8}$$

where $S' = (c, x', y')$ and $p(\mathbf{y}|\mathbf{z}, \mathbf{c}, \mathbf{x}, \mathbf{S}')$ is the mirror of $p(\mathbf{y}'|\mathbf{z}, \mathbf{c}, \mathbf{x}', \mathbf{S})$. Here, we consider $p(\mathbf{y}|\mathbf{z}, \mathbf{c}, \mathbf{x}, \mathbf{S}')$ rather than $p(\mathbf{y}'|\mathbf{z}, \mathbf{c}, \mathbf{x}', \mathbf{S})$ because it has been optimized in Equation 4. Equation 8 can be explained in more detail as following. That is, we first use our generator to generate examples, then we use the frozen classifier to calculate the coherence of the examples, and then we update the parameters of the generator to maximize the coherence. In fact, we use the classifier to restrict the generated $\tilde{y}$ entails its true condition x but contradicts with x'. As in (Hu et al., 2017; Hu and Li, 2021), we use Gumbel-softmax to enable gradient backpropagation for the discrete text.

3.4 Utilizing External Event Causalities

The variational process provides an *implicit* background for unobserved counterfactual outcomes, this further motivates us to utilize external event causalities which allows for introducing diverse event commonsense and providing an *explicit* background for generating counterfactual outcomes.

3.4.1 Retrieving Event Causality

We use COMeT (Hwang et al., 2021) as the event knowledge base. We first feed the zero-hop events (c, x') into COMeT to generate one-hop events with corresponding relations. The one-hop events are then fed into COMeT to generate two-hop events. We leave the details in Appendix A. We next organize the retrieved knowledge into an event graph $G = (V, E)$ where V denotes the node set and E denotes the edge set. Each node $e \in V$ is an event which is a word sequence. Each edge in E is a tuple (e_h, r, e_t) containing a head event e_h, a relation r, and a tail event e_t. Then, we perform reasoning on G to select guided events, which are the possible effects of (c, x'). We use the selected events as guidance for generating y'.

3.4.2 Selecting Guided Events

Motivated by (Ji et al., 2020; Mu and Li, 2022), we perform multi-hop reasoning on G to select important event nodes. We iteratively compute the relevance scores of multi-hop events with respect to (c, x'), as shown in Figure 3(a). In each iteration, we parallelly calculate the scores of events in the same hop. For the tail event e_t, the score $s(e_t)$ is calculated by polymerizing information from its neighbors $\mathcal{N}_{e_t}$ including pairs of (e_h, r):

$$s(e_t) = \frac{1}{|\mathcal{N}_{e_t}|}\sum_{(e_h,r)\in\mathcal{N}_{e_t}} (s(e_h) + R(e_h, r, e_t)). \tag{9}$$

At the beginning, zero-hop events, i.e., (c, x'), are assigned a score of 1, e.g., $s(c) = s(x') = 1$, while other events are assigned a score of 0. $R(\cdot)$ is the relevance of the edge (e_h, r, e_t) with respect to the (c, x'), which is calculated by:

$$R(e_h, r, e_t) = \sigma(\mathbf{h}^T_{(c,x')}\mathbf{W}_k \cdot [\mathbf{h}_{e_h}; \mathbf{h}_r; \mathbf{h}_{e_t}]), \tag{10}$$

where $\mathbf{W}_k \in \mathcal{R}^{d\times 3d}$, $[\cdot;\cdot]$ denotes the concatenation, $\mathbf{h}_{(c,x')} \in \mathcal{R}^d$ is the embedding of (c, x'), $\mathbf{h}_{e_h}, \mathbf{h}_r, \mathbf{h}_{e_t}$ are the embeddings of e_h, r, e_t.

We select the top-k events according to their scores: $E_k = \text{topk}_i(s(e_i))$. k is set to 4 after searching on the dev set. To fuse the guided E_k into the generation, we concatenate (S, c, x') with E_k, and pass them to BARTEncoder to obtain knowledge-enhanced context vectors $\mathbf{H}_{\tilde{C}} = \text{BARTEncoder}(S, c, x', E_k)$. Then, we concatenate $\mathbf{H}_{\tilde{C}}$ with $\mathbf{z} \sim q(\mathbf{z}|\cdot)$, and feed it into the

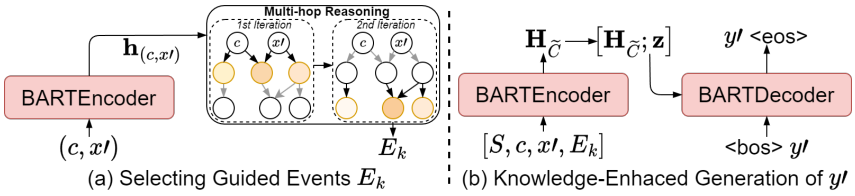


Figure 3: (a) The scores of one-hop and two-hop events are parallelly calculated in each iteration. Color intensity indicates the score difference. In each iteration, the black arrows denote used edges, while the grey arrows denote unused edges. (b) We concatenate E_k with (S, c, x') for the auto-regressive decoding.

decoder for autoregressive learning, i.e., $y' \sim p(\mathbf{y}'|\mathbf{z}, \mathbf{c}, \mathbf{x}', \mathbf{S}, \mathbf{E}_k)$, as shown in Figure 3(b).

3.5 Training and Inference

Similar to (Mu and Li, 2022), we add an additional object to guide the event selection. We maximize the probability of selecting positive events by:

$$\mathcal{L}_E = \frac{\sum_{\mathcal{D}}}{|\mathcal{D}|} \sum_i -l_i \log p(e_i) - (1-l_i) \log(1-p(e_i)), \quad (11)$$

where $p(e_i) = \sigma(s(e_i))$ is the probability that the event e_i is selected. l_i is the label of e_i which is subject to the overlap between e_i and the gold y'. The details are in Appendix A. The final object is:

$$\mathcal{L} = \mathcal{L}_{VAE} + \alpha \mathcal{L}_{Cla} + \beta \mathcal{L}_E, \quad (12)$$

where α and β are hyper-parameters.

When inference, given input (S, c, x'), we first sample $\mathbf{z} \sim p(\mathbf{z}|\mathbf{S})$, then we select guided event E_k according to (c, x'), at last we generate the counterfactual outcome $y' \sim p(\mathbf{y}'|\mathbf{z}, \mathbf{c}, \mathbf{x}', \mathbf{S}, \mathbf{E}_k)$.

4 Experiment

4.1 Datasets

We evaluate our method on two datasets.

(1) **TimeTravel** (Qin et al., 2019) is a counterfactual story rewriting dataset. It is built on the ROCStories (Mostafazadeh et al., 2016) corpus, which consists of a large set of five-sentence stories $S = s_{1:5}$. s_1 is set as the context c, s_2 is set as the condition x, and $s_{3:5}$ sets up the outcome y. In TimeTravel, the initial condition x is rewritten by humans into a counterfactual condition x', and then annotators perform minimal edits to the original ending y to create the counterfactual outcome y'. In TimeTravel, the major challenge is the trade-off between generating natural stories and modifying the original y with minimal edits.

(2) **PossibleStories** (Ashida and Sugawara, 2022), also built on the ROCStories corpus, considers the problem that: for the same context, if the current situation is different, the final consequence may be different. It is originally a multiple-choice dataset, where each example consists of the original context c, the original ending y, the counterfactual question x', and candidate options including the counterfactual ending y'. To adapt it to text generation, we set the original condition x as a simple text "*what's the most likely story ending?*", then we generate y' according to (c, x, y, x').

The statistics of two datasets are in Appendix B.

4.2 Baselines

We produce the following kinds of baselines:

- **Prompting large chat models**, e.g., ChatGLM2(6B) (Zeng et al., 2022), Llama2Chat(7B) (Touvron et al., 2023), ChatGPT (OpenAI. , 2023). We use one-shot prompting for experiments, the used prompts are in Appendix C.
- **Supervised fine-tuning**. We fine-tune several pre-trained language models, including GPT2(base) (Radford et al., 2019), T5(base) (Raffel et al., 2020), BART(base) (Lewis et al., 2019), and Llama2(7B) (Touvron et al., 2023). We use QLoRA (Dettmers et al., 2023) to adapt Llama2(7B) on a single 3090 GPU.

For TimeTravel, we additionally compare our method with some task-specific methods:

- DELOREAN (Qin et al., 2020) and EDUCAT (Chen et al., 2022) which regard the task as a controllable generation problem, and unsupervised edit the original y to the counterfactual y.
- CLICK (Li et al., 2023a), a two-stage method, first detects which words in the original ending need to be modified, and then implements the modification.

4.2.1 Implementation Details

We use the train set of the two datasets to train the classifier. We use the AdamW optimizer and set lr to 5e-6. We select checkpoint according to F1 on the dev set. The best checkpoint achieves the F1 scores of 66.1 and 70.1 in the test set of two

datasets. When training the counterfactual generator, we use the AdamW optimizer and set lr to 5e-5. We linearly decrease lr to zero with a 10% warmup ratio. We search for the best hyper-parameters according to ENTScore on the dev set of each dataset. The searched parameters are in Appendix D. When inference, we adopt the multinomial sampling strategy to generate y', and we repeat for 5 times to calculate the average performance.

4.3 Automatic Evaluation

Metrics For Timetravel, we follow the previous works and use BLEU (Papineni et al., 2002), BertScore (Zhang et al., 2019), ENTScore (Chen et al., 2022), and HMean $= \frac{2 \cdot \text{BLEU} \cdot \text{ENTScore}}{\text{BLEU}+\text{ENTScore}}$ (Chen et al., 2022) as metrics. BLEU and BertScore evaluate the similarity between the generated y' and the ground truth. ENTScore evaluates the coherence between (c, x') and the generated y'. For PossibleStories, we use BLEU, BertScore, and ENTScore as metrics.

4.3.1 Our Method vs. Baselines

TimeTravel				
Methods	BLEU	BertS.	ENTS.	HMean
Prompting Chat Models				
ChatGLM2(6B)	16.5	60.0	66.2	26.4
Llama2Chat(7B)	16.9	58.8	77.8	27.8
ChatGPT	36.4	69.8	**82.6**	50.6
Unsupervised Editing-based Methods				
DELOREAN	23.9	59.9	51.4	32.6
EDUCAT	44.1	74.1	32.3	37.3
Supervised Fine-tuning				
CLICK	46.7	73.2	36.7	41.1
GPT2	63.5(0.2)	77.8(0.3)	43.5(1.0)	51.6(0.7)
T5	71.2(0.3)	**80.1(0.1)**	42.7(0.8)	53.3(0.6)
BART	66.5(0.3)	79.4(0.2)	52.0(1.0)	58.3(0.6)
Llama2(7B)	**70.3(0.4)**	79.9(0.2)	54.1(0.7)	60.9(0.5)
Ours	67.0(0.1)	79.5(0.1)	56.2(0.4)	**61.1(0.2)**
Ablation Experiment				
w/o Clas	67.5(0.2)	79.8(0.1)	54.6(0.6)	60.4(0.4)
w/o Event	65.6(0.4)	79.0(0.1)	55.2(0.5)	60.0(0.4)
w/ VAE	65.9(0.3)	79.2(0.1)	54.1(0.6)	59.4(0.4)

Table 1: The automatic and ablation-study result on TimeTravel. We report the mean(std) under 5 random experiments. Scores with **bold** denote the best results.

The automatic evaluation result is shown in Table 1 and 2. We can see BART generally performs better than GPT2 and T5, therefore we use BART as the backbone. In addition, we observe that:

- In Table 1, unsupervised editing-based methods have poor performances, indicating that this kind of unsupervised approach is unable to produce qualified counterfactual stories.
- Compared with BART, our method achieves an obvious improvement, especially in the ENTScore metric, e.g., obtaining a 4.2/4.0 gain on two datasets. In addition, our method outperforms Llama2(7B), which indicates that our method is effective in improving the causality between (c, x') and the generated y'.
- Due to the extremely large-scale pre-training, Chat models, e.g., ChatGLM2, Llama2Chat, and ChatGPT, have a strong ability to generate coherent stories. However, chat models get a low BLEU and BertScore, indicating that they tend to less consider what has happened.
- On TimeTravel, the ENTScore result of our method is not as good as the results of chat models, but our method achieves the best trade-off between BLEU and ENTScore. On PossibleStories, our method approximates ChatGPT and surpasses ChatGLM2 by a large margin. This indicates that the small-model-based sophisticated method is expected to be comparable to LLM-based prompting, indicating that it still has research value in the era of LLMs.

PossibleStories			
Methods	BLEU	BertScore	ENTScore
Prompting Chat Models			
ChatGLM2(6B)	1.9	48.4	38.8
Llama2Chat(7B)	3.0	49.9	43.8
ChatGPT	5.0	53.5	**48.5**
PLMs-based Finetuning			
GPT2	6.0(0.7)	49.4(0.3)	37.3(0.4)
T5	5.7(0.3)	49.2(0.3)	35.8(0.7)
BART	13.2(0.5)	53.8(0.2)	42.9(1.0)
Llama2(7B)	**16.3(1.1)**	54.4(0.6)	45.1(0.9)
Ours	16.1(0.2)	**56.2(0.1)**	46.9(1.0)
Ablation Experiment			
w/o Clas	15.7(0.4)	55.6(0.3)	45.6(0.7)
w/o Event	15.5(0.4)	55.8(0.2)	46.0(0.5)
w/ VAE	15.5(0.5)	55.9(0.3)	45.0(0.4)

Table 2: The automatic and ablation-study result on PossibleStories. We report the mean(std) under 5 random experiments. Scores with **bold** denote the best results.

4.3.2 Ablation Study

Settings To investigate the effectiveness of different components, we devise the following ablated variants to compare with our full model. (1) "w/o Event" means we do not use event causality. (2) "w/o Cla" means we remove the pre-trained clas-

sifier. (3) “w/ VAE" means we ablate both event causality and the pre-trained classifier. In this case, this variant degenerates into the basic VAE module.

Result The ablation study result is shown in Table 1 and 2. We have the following observations.

- Compared with BART, “w/ VAE" achieves a 2.1/2.1 gain in ENTScore on both datasets. This demonstrates the effectiveness of the variational process. The possible reason is that the variational process learns an approximation of the latent $\mathbf{z}$, which provides an implicit background for generating counterfactual outcomes.
- Compared with “w/ VAE", “w/o Clas" and “w/o Event" achieve higher ENTScore on both datasets, indicating the two strategies contribute to improving the causality between x' and the generated y'. This makes sense because (1) external event causality provides a causal background for generating y' and (2) the classifier will punish the unqualified generation. The best result is achieved when combining two strategies, these show that two strategies complement each other.
- “w/o Clas" performs better than “w/o Event" on both datasets. This shows that the classifier is more important than event knowledge. The possible reason lies in two aspects. (1) Though retrieved event causality may contain useful information for generation, unrelated and noisy knowledge may also be retrieved. (2) The classifier directly measures the causality between the condition and the generated outcome, and penalizes incoherent generation.

Methods	TimeTravel				PossibleStories			
	MinEdits		Coherence		Similarity		Coherence	
	W	L	W	L	W	L	W	L
vs. w/o Clas	14.7	23.7	28.0	10.3	10.0	9.0	16.3	7.0
vs. w/o Event	17.0	25.3	37.3	10.0	13.3	7.0	24.3	6.7
vs. Llama2Chat(7B)	61.0	11.0	24.3	37.7	32.3	12.7	41.7	20.7
vs. ChatGPT	52.3	15.7	16.7	47.0	21.7	13.0	23.7	27.3

Table 3: The manual evaluation result. MinEdits denotes Minimal-Edits.

4.4 Manual Evaluation

Setting For TimeTravel, we follow (Chen et al., 2022) and use *Minimal-Edits* and *Coherence* as manual evaluation metrics. *Coherence* denotes the logical consistency between the counterfactual context (c, x') and generated y'. *Minimal-Edits* denotes the extent of minimal revision between the original y and the generated y'. For PossibleStories, we use Similarity and Coherence as metrics, where *Similarity* evaluates the similarity between the generated y' and the ground truth. We carry out pairwise comparisons between our method with some baselines, including Llama2Chat, ChatGPT, and two ablated models “w/o Event" and “w/o Clas". We randomly sample 100 cases from the two test sets for each pair of models, respectively. Three annotators are recruited to make a preference among Win, Tie, and Lose given the input and two outputs generated by our model and a baseline respectively. The annotators are research students from the field of commonsense text generation to make sure they have a fair judgment of used metrics.

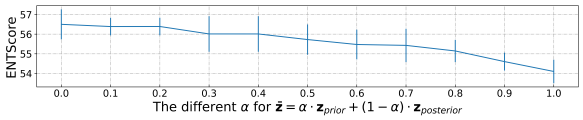


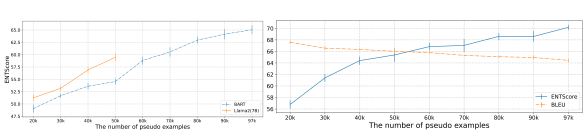
Figure 4: Linearly interpolating $\mathbf{z}_{prior}$ and $\mathbf{z}_{posterior}$ for the VAE decoding, i.e., $y' \sim p(\mathbf{y}'|\mathbf{z}, \mathbf{c}, \mathbf{x}', \mathbf{S})$.

Result The result is shown in Table 3. Compared with the two ablated variants, our full method shows an increase in Coherence, but a decrease in Minimal-Edits. This is because both of the two strategies prevent copying the original ending y. On TimeTravel, our full model performs better in Minimal-Edits, but not as well in Coherence as the chat models. This is consistent with automatic evaluation. We calculate Fleiss's kappa reliability as the inter-rater agreement. For TimeTravel, the agreement of Minimal-Edits and Coherence is 0.43 and 0.56. For PossibleStories, the agreement of Similarity and Coherence is 0.50 and 0.52.

4.5 Further Discussion

4.5.1 Analyzing the VAE Module by Manipulating z

Since the strength of our VAE module lies in its ability to approximate the posterior distribution, we are interested in whether the encoded $\mathbf{z}$ benefits counterfactual narrative generation. We conduct a pilot study on TimeTravel. We first replace $\mathbf{z} \sim p(\mathbf{z}|\mathbf{S})$ with a random noise $\mathbf{z}_{noise} \sim \mathcal{N}(0, 1)$, and feed $\mathbf{z}_{noise}$ into the VAE decoder $p(\mathbf{y}'|\mathbf{z}, \mathbf{c}, \mathbf{x}', \mathbf{S})$ for generation. We get a 43.8 ENTScore, which is significantly worse than the result of BART. This is reasonable because the random noise disrupts the model structure and brings about a significant negative impact. Next, we make a linearly interpolation $\bar{\mathbf{z}} = \alpha \cdot \mathbf{z}_{prior} + (1-\alpha) \cdot \mathbf{z}_{posterior}$, where $\mathbf{z}_{prior} \sim$



(a) (b)

Figure 5: (a): Fine-tuning BART and Llama2(7B) with a different number of pseudo examples. (b): Fine-tuning BART via mixing the labeled set $\mathcal{D}$ and a different number of pseudo examples.

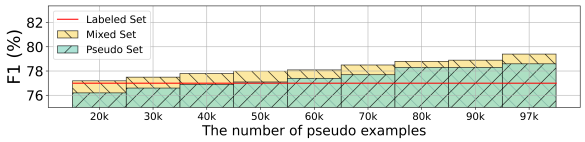


Figure 6: Fine-tuning RoBERTa-large with different types of training examples.

$p(\mathbf{z}|\mathbf{S})$ and $\mathbf{z}_{posterior} \sim q(\mathbf{z}|\mathbf{c}, \mathbf{x}', \mathbf{y}', \mathbf{S})$, and feed $\bar{\mathbf{z}}$ for generation. The result is shown in Figure 4. The larger the proportion of $\mathbf{z}_{posterior}$, that is, the more posterior information about the gold y', the better the result is achieved. When using $\mathbf{z}_{posterior}$ for generation, we get a 56.5 ENTScore, but the value is still not satisfactory enough. The possible reason is that too much information is lost during the process of encoding the sequence into a vector $\mathbf{z}$, making it hard for the model to reconstruct the gold y'. According to above results, we speculate that the more effective solution to this problem is to introduce more diverse and more large-scale data. Therefore, we conduct the following data augmentation experiment.

4.5.2 Generating Counterfactual Stories for Data Augmentation

(Qin et al., 2019) provides an additional data partition that only has counterfactual conditions x' but no counterfactual outcomes. This partition contains about 97k examples. We use this partition to study the practicality of the generated counterfactual stories via a data augmentation experiment. Specifically, we use our ablated variant "w/o Event" as the generator since its performance is not significantly worse than our full model, and there is no need for external knowledge, making it easy to use. For each (S, c, x'), we use "w/o Event" to generate 60 candidates and keep the one with the highest ENTScore as the pseudo counterfactual outcome, denoted as $\widetilde{y'}$. Finally, we obtain the pseudo set $\mathcal{D}_P = \{(S, c, x', \widetilde{y'})\}$. We test this set for both generation and classification tasks.

Testing for the Generation Task First, We only use $\mathcal{D}_P$ to directly fine-tune BART and Llama2, i.e., learning $p(\mathbf{y}'|\mathbf{c}, \mathbf{x}', \mathbf{S})$. The result is shown in Figure 5(a). When training with about 32k pseudo examples, BART achieves a 52.0 ENTScore, which is obtained using the labeled set $\mathcal{D}$. When using more pseudo examples for training, the result continuously improves. We have a similar observation from the result of Llama2(7B). Because finetuning Llama2(7B) is time-consuming, e.g., it takes about 1.5 hours to train an epoch with 30k samples, we use a maximum of 50k samples for fine-tuning. Next, we mix the labeled set $\mathcal{D}$ and a different number of pseudo examples to fine-tune BART. The result is shown in Figure 5(b). When mixing $\mathcal{D}$ and all pseudo examples, BART obtained a 70.1 ENTScore, which is better than ChatGLM2 and closer to Llama2Chat. However, as the number of pseudo examples increases, BLEU continues to decline, but overall the decline is acceptable.

Testing for the Classification Task Motivated by (Chen et al., 2022), we construct a binary classification task to test the quality of pseudo examples, which is the same as training the classifier. We explore three types of training examples to train RoBERTa-large (Liu et al., 2019) and then validate on the test set of TimeTravel: (1) the labeled set $\mathcal{D}$, (2) the pseudo set $\mathcal{D}_P$ in which we randomly sample a different number of pseudo examples, and (3) the mixed set in which we mix the label set $\mathcal{D}$ and a different number of pseudo examples. The F1 result on the test set of TimeTravel is shown in Figure 6. When training with more pseudo examples, F1 achieves a stable improvement. The F1 under the mixed set is better than that under the labeled set, indicating that the pseudo set is an effective supplement to the labeled set.

Overall, these results demonstrate the practicality of the generated pseudo examples, which further proves the effectiveness of our method.

4.5.3 Case Study and Error Analysis

Table 4 presents a case study. The counterfactual ending generated by ChatGPT is coherent, but it differs significantly from the original ending. Our model changes the "land" in the original ending to "fossil" and generates a coherent counterfactual ending with minimal-edits. Since the context has changed from "oil" to "fossils", the counterfactual endings containing fossil-related information better

Original Story	A man dug a well on his farm. Instead of water, though, he struck oil. Jubilant, he placed some calls. A large oil company arrived the next day. They bought the land from the man for a million dollars.
Counterfactual Context	A man dug a well on his farm. Instead of water, though, he struck fossils.
w/o Clas	Jubilant, he placed some calls. A large fossils company arrived the next day. They bought the land from the man for a million dollars.
w/o Event	Jubilant, he placed some calls. A large oil company arrived the next day. They bought the land from the man for a million dollars.
Llama2Chat	Excited, he placed some calls. A paleontology company arrived the next day. They bought the land from the man for a large sum of money!
ChatGPT	Jubilant, he contacted a local museum to share his discovery. A team of paleontologists arrived the next day. They offered to buy the fossils from the man for a substantial amount of money.
Ours	Jubilant, he placed some calls. A large fossil company arrived the next day. They bought the fossils from the man for a million dollars.

Table 4: A case study with the generated texts by different models. The case is from the test set of TimeTravel.

reflect the causal relationship between the perturbation and story endings. However, we find that the issue of paraphrasing y still exists in our method, as shown in Appendix E, Table 5. But this issue less-likely occurs in large chat models. We speculate that there are two reasons: (1) the problem of exposure bias cannot be completely eliminated; (2) The used model is small, and the scale and diversity of data are insufficient. More examples can be seen in Appendix E, Table 9.

5 Conclusion

In this work, we formulate counterfactual reasoning in narratives in a VAE framework. In addition, we introduce a pre-train classifier and external event causality to further improve the causality between the counterfactual condition and the generated counterfactual outcome. The experiment proves the effectiveness of our method. We also conduct a data augmentation experiment to verify the practicality of our method.

Acknowledgements

We thank the anonymous reviewers for their encouraging feedback. This work is supported by Research Grants Council of Hong Kong(PolyU/15207920, PolyU/15207821, PolyU/15213323) and National Natural Science Foundation of China (62076212).

Limitations

We construct our method based on small-scale datasets and the small model, e.g., BART, therefore our method cannot outperform large language models. But it is inherently unfair to directly compare small models with LLMs, as large models are obtained with massive resources, e.g., data, hardware, funding, etc. Although large models have strong capabilities, their disadvantage is that once trained, it is especially difficult to adjust them. Therefore, LLMs may not be able to meet task-specific constraints that have not been seen during training. However, the problem is that different tasks may have additional requirements. For example, TimeTravel requires the generated y' to exhibit Minimal-Edits characteristics. However, the output of chat models conflicts with the requirement, indicating that chat models still face the problem of precisely following instructions for generation.

In the experiment, fine-tuned Llama2 performs better than fine-tuned BART. This indicates that constructing our method upon larger models may have a better performance. In addition, the generated counterfactual stories are beneficial for counterfactual narrative reasoning. This foreshadows the future direction, that is, we can transform x into different x' through different perturbations, thus generating diverse counterfactual stories for data augmentation. In a bootstrapping manner, more and more qualified examples can be created for training neural models. Different from predicting counterfactual outcomes, it is easy to perturb x into x', and there have been a lot of related research works. We leave this in the future work.

In addition, we demonstrate our method using five-sentences RocStories datasets, but it does not mean that our method is not applicable for stories of variable length. Our SCM contains three variables: the context c, the treatment x, and the effect y. Actually, c, x, and y may contain one or several narrative events. That is, given a narrative story with variable length, we can divide it into three parts (c, x, y) to adapt our approach.

Ethical Considerations

This paper mainly focuses on narrative reasoning. It describes event relationships in human daily life, and does not involve sensitive, biased, or harmful content. The datasets used in this work does not involve any sensitive data, but only crowd-sourced datasets released in previous works, including RocStories (Mostafazadeh et al., 2016), TimeTravel (Qin et al., 2019), and PossibleStories (Ashida and Sugawara, 2022). We believe that our research

work meets the ethics of ACL.

References

Mana Ashida and Saku Sugawara. 2022. Possible stories: Evaluating situated commonsense reasoning under multiple possible scenarios. *arXiv preprint arXiv:2209.07760*.

Jiangjie Chen, Chun Gan, Sijie Cheng, Hao Zhou, Yanghua Xiao, and Lei Li. 2022. Unsupervised editing for counterfactual stories. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10473–10481.

Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. 2021. Honey or poison? solving the trigger curse in few-shot event detection via causal intervention. *arXiv preprint arXiv:2109.05747*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*.

Nouha Dziri, Ehsan Kamalloo, Kory W Mathewson, and Osmar Zaiane. 2019. Evaluating coherence in dialogue systems using entailment. *arXiv preprint arXiv:1904.03371*.

Changying Hao, Liang Pang, Yanyan Lan, Yan Wang, Jiafeng Guo, and Xueqi Cheng. 2021. Sketch and customize: A counterfactual story generator. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12955–12962.

Zhiting Hu and Li Erran Li. 2021. A causal lens for controllable text generation. *Advances in Neural Information Processing Systems*, 34:24941–24955.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *International conference on machine learning*, pages 1587–1596. PMLR.

Jena D. Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. Comet-atomic 2020: On symbolic and neural commonsense knowledge graphs. In *AAAI*.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Haozhe Ji, Pei Ke, Shaohan Huang, Furu Wei, Xiaoyan Zhu, and Minlie Huang. 2020. Language generation with multi-hop reasoning on commonsense knowledge graph. *arXiv preprint arXiv:2009.11692*.

Dongyeop Kang, Tushar Khot, Ashish Sabharwal, and Eduard Hovy. 2018. Adventure: Adversarial training for textual entailment with knowledge-guided examples. *arXiv preprint arXiv:1805.04680*.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Chunyuan Li, Xiang Gao, Yuan Li, Baolin Peng, Xiujun Li, Yizhe Zhang, and Jianfeng Gao. 2020. Optimus: Organizing sentences via pre-trained modeling of a latent space. *arXiv preprint arXiv:2004.04092*.

Dandan Li, Ziyu Guo, Qing Liu, Li Jin, Zequen Zhang, Kaiwen Wei, and Feng Li. 2023a. Click: Integrating causal inference and commonsense knowledge incorporation for counterfactual story generation. *Electronics*.

Dandan Li, Ziyu Guo, Qing Liu, Li Jin, Zequn Zhang, Kaiwen Wei, and Feng Li. 2023b. Click: Integrating causal inference and commonsense knowledge incorporation for counterfactual story generation. *Electronics*, 12(19):4173.

Fangchao Liu, Lingyong Yan, Hongyu Lin, Xianpei Han, and Le Sun. 2021a. Element intervention for open relation extraction. *arXiv preprint arXiv:2106.09558*.

Qi Liu, Matt Kusner, and Phil Blunsom. 2021b. Counterfactual data augmentation for neural machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 187–197, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and evaluation framework for deeper understanding of commonsense stories. *arXiv preprint arXiv:1604.01696*.

Feiteng Mu and Wenjie Li. 2022. Enhancing text generation via multi-level knowledge aware reasoning. IJCAI.

Feiteng Mu and Wenjie Li. 2023. Enhancing event causality identification with counterfactual reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 967–975.

OpenAI. (2023). ChatGPT (gpt-3.5-turbo) [Large language model]. Https://chat.openai.com/chat.

Jiao Ou, Jinchao Zhang, Yang Feng, and Jie Zhou. 2022. Counterfactual data augmentation via perspective transition for open-domain dialogues. *arXiv preprint arXiv:2210.16838*.

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.

Judea Pearl. 2009. *Causality*. Cambridge university press.

Judea Pearl, Madelyn Glymour, and Nicholas P Jewell. 2016. *Causal inference in statistics: A primer*. John Wiley & Sons.

Judea Pearl and Dana Mackenzie. 2018. *The book of why: the new science of cause and effect*. Basic books.

Lianhui Qin, Antoine Bosselut, Ari Holtzman, Chandra Bhagavatula, Elizabeth Clark, and Yejin Choi. 2019. Counterfactual story reasoning and generation. *arXiv preprint arXiv:1909.04076*.

Lianhui Qin, Vered Shwartz, Peter West, Chandra Bhagavatula, Jena D Hwang, Ronan Le Bras, Antoine Bosselut, and Yejin Choi. 2020. Backpropagation-based decoding for unsupervised counterfactual and abductive reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 794–805.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Ali Razavi, Aäron van den Oord, Ben Poole, and Oriol Vinyals. 2019. Preventing posterior collapse with delta-vaes. *arXiv preprint arXiv:1901.03416*.

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 *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3027–3035.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Liuyi Yao, Zhixuan Chu, Sheng Li, Yaliang Li, Jing Gao, and Aidong Zhang. 2021. A survey on causal inference. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(5):1–46.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. 2022. Glm-130b: An open bilingual pre-trained model. *arXiv preprint arXiv:2210.02414*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Yucheng Zhou, Tao Shen, Xiubo Geng, Guodong Long, and Daxin Jiang. 2022. Claret: Pre-training a correlation-aware context-to-event transformer for event-centric generation and classification. *arXiv preprint arXiv:2203.02225*.

A Details for Constructing Event Causality Graph

To obtain event causality knowledge for counterfactual narrative generation, we use COMeT as event knowledge base. COMeT is a transformer model trained on ATOMIC (Sap et al., 2019) that generates nine kinds of inferences of events in natural language. In our work, we select 8 relations that are causal related for retrieving event knowledge. They are *xNeed*, *xIntent*, *Causes*, *HasSubEvent*, *oEffect*, *oWant*, *xEffect*, *xWant*. Given a query event, we retrieve 5 knowledge tuples for each relation. We first feed the central events (c, x') into COMeT to generate one-hop events with corresponding relations. The one-hop events are then fed into COMeT to generate two-hop events. After that, we obtain many event paths. Several heuristic rules are applied to filter low-quality paths. For example, if an event in a path contains less than 2 words or more than 8 words, the path will be discarded. Finally, we transform the retained event chains into an event graph that simulates the evolution tendency of events centered on (c, x'), and provides a explicit background for the unseen counterfactual outcome. We heuristically use the word-overlap ratio to label event nodes. That is, an event is labeled as positive if 60% of event tokens are contained by the gold y'. The event labels are used as supervision for selecting guided events.

Original Story	Megan loved her sock monkey. She took it to her grandad's house when she visited him. Megan got home and realized she had left her monkey. I had to meet grandad halfway to his house and pick up her monkey.
Original ending	Megan was so happy and she was then able to go to bed.
Counterfactual question:	Why was it so important to get the sock monkey back before bedtime?
w/o Clas	The monkey needed to be taken back before Megan got to play with it.
w/o Event	Megan was so happy and she was then able to go to bed.
Llama2Chat	Megan needs to play with her sock monkey before going to bed.
ChatGPT	Megan couldn't sleep without her sock monkey by her side.
Ours	Megan was so happy and she was then able to go to bed.

Table 5: An example for error analysis. The case is from the test set of PossibleStories.

B Statistics of the Used Datasets

Datasets	Train	Dev	Test
TimeTravel	28363	1871	1871
PossibleStories	3404	458	671

Table 6: Statistics of the datasets used in this work.

The statistics of the used datasets are shown in Table 6.

Datasets	bs	lr	α	β	λ_x
TimeTravel	8	5e-5	1.0	0.5	1.0
PossibleStories	8	5e-5	0.5	0.5	0.5

Table 7: The searched hyper-parameters.

C The Prompts for Different Tasks

Table 8 presents the prompts we used for utilizing chat models.

D Hyper-Parameters Used in This Work

The searched hyper-parameters on two datasets are shown in Table 7.

E Supplementary Cases

Table 5 shows a case for error analysis, in which our method still copies the original y. We count the statistics on the test set of TimeTravel. The number of our model directly copies the original is approximately 8% of the test set. Table 9 presents some cases with the generated counterfactual outcome by different models. The case #1 is from the test set of TimeTravel, and the case #2 is from the test set of PossibleStories.

Tasks	Prompt
TimeTravel	Each story contains 5 sentences, where the first two sentences are the story premise, and the last 3 sentences are the story ending. I will apply subtle a perturbation to the second sentence, making the first two sentences a counterfactual story premise. Due to the slight perturbation, the counterfactual premise is very similar to the original premise, with only some words being different. According to the original story and the counterfactual story premise, you are required to predict the counterfactual story ending. Note that the counterfactual story ending should be similar to the original story ending, as well as being coherent with the counterfactual story premise. Here is one example: ### <Original 5-sentences story> 1. Bella wanted to cook some spaghetti and meatballs. 2. She discovered she had no pasta noodles. 3. She found a recipe online that used spaghetti squash instead. 4. Bella luckily had a spaghetti squash on hand. 5. She was surprised to find the spaghetti and meatballs delicious! <Counterfactual story premise> 1. Bella wanted to cook some spaghetti and meatballs. 2. She realized she didn't have the time to make it properly so she changed made an omelette instead. <Counterfactual story ending > 3. She found a recipe online that used egg whites instead. 4. Bell luckily had many eggs on hand. \\ 5. She was surprised to find the egg white omelette delicious! ### Now, given the following example, please write the counterfactual story ending. There should be only three sentences at the counterfactual story ending. <Original 5-sentences story> {original_story} <Counterfactual story premise> {counterfactual_premise} <Counterfactual story ending>
PossibleStories	You will observe a story that consists of a context and an ending. Then given the counterfactual question, please generate a new story ending that is compatible with the question. Here is an example: ###<Observed story context> Fred and James both claimed they were the best basketball player. One day they decided to find out who was better. James loved to brag, but Fred was focused on the game. Eventually Fred beat James by 1 point. <Observed story ending> James learned that day to focus on the game, not on bragging. <Counterfactual question> What is most likely to happen if Fred has a lot of empathy for others? <Counterfactual story ending> Fred felt bad that he won, so the next game he eased up and let James win. ### Now, given the following example, please write the counterfactual story ending. You can only generate one sentence, do not add additional content. <Observed story context> {original_context} <Observed story ending> {original_ending} <Counterfactual question> {cf_context} <Counterfactual story ending>

Table 8: The prompts used for different tasks.

#1	Original Story	Myla was asked to be in a play. She a had a leading role. Myla was struggling to remember her lines. Her mother began helping her practice daily. With her mother's help Myla memorized her lines quickly.
	Counterfactual Context	Myla was asked to be in a play. She declined because she wasn't interested.
	w/o Clas	Myla was struggling to remember her lines. Her mother began helping her practice daily. With her mother's help Myla memorized her lines quickly.
	w/o Event	Myla was struggling with her singing. Her mother began watching her practice daily. With her mother's help Myla practiced her singing quickly.
	Llama2Chat	She wished she had taken the opportunity when it was offered. Myla realized her mistake and tried to find another chance.
	ChatGPT	Myla's friend offered to take her place in the play. Her friend's mother began helping her practice daily. With her friend's mother's help, Myla's friend memorized her lines quickly.
	Ours	Myla decided to audition for a different play, and Myla was struggling to remember her lines. Her mother began helping her practice daily. With her mother's help Myla memorized her lines quickly.
#2	Original Story	Anne liked to bake. She bought a new pie pan for baking pies. She planned on using it a lot this fall. When it was time to test it out, she couldn't find it.
	Original ending	Her son had taken it and used it in the sandbox!
	Counterfactual question:	What is most likely to happen after this if she was still able to make the pies?
	w/o Clas	Her son had taken it and used it in the sandbox!
	w/o Event	When she tried to bake them last year, the butter burnt off the pan and they all melted.
	Llama2Chat	She made a delicious pie that everyone loved.
	ChatGPT	She would have baked a delicious pie using a different pan.
	Ours	She could try it out a little more time later and it gave her better results.

Table 9: Some cases with the generated counterfactual outcomes by different models.