Learning to Write Stories with Thematic Consistency and Wording Novelty

Juntao Li,[1,2] **Lidong Bing,**[3] **Lisong Qiu,**[1,2] **Dongmin Chen,**[1] **Dongyan Zhao,**[1,2] **Rui Yan**[1,2*]
[1]Center for Data Science, Academy for
Advanced Interdisciplinary Studies, Peking University, Beijing, China
[2]Institute of Computer Science and Technology, Peking University, Beijing, China
[3]R&D Center Singapore, Machine Intelligence Technology, Alibaba DAMO Academy
{lijuntao,qiuls,dongminchen,zhaody,ruiyan}@pku.edu.cn
l.bing@alibaba-inc.com

Abstract

Automatic story generation is a challenging task, which involves automatically comprising a sequence of sentences or words with a consistent topic and novel wordings. Although many attention has been paid to this task and prompting progress has been made, there still exists a noticeable gap between generated stories and those created by humans, especially in terms of thematic consistency and wording novelty. To fill this gap, we propose a cache-augmented conditional variational autoencoder for story generation, where the cache module allows to improve thematic consistency while the conditional variational autoencoder part is used for generating stories with less common words by using a continuous latent variable. For combing the cache module and the autoencoder part, we further introduce an effective gate mechanism. Experimental results on ROCStories and WritingPrompts indicate that our proposed model can generate stories with consistency and wording novelty, and outperforms existing models under both automatic metrics and human evaluations.

Introduction

Story writing is the new frontier in the research of text generation, and it involves various challenges that exist in the current neural generation systems (Wiseman, Shieber, and Rush 2017). Conventionally, case-based reasoning (Gervás et al. 2005; Swanson and Gordon 2012), agent-based simulation (Brenner 2010), planning (Porteous, Cavazza, and Charles 2010), and plot graphs (Li et al. 2013) are studied for this task. Recently, deep neural networks are employed for open story generation without being limited by prior engineered domain knowledge (Martin et al. 2017; May and Knight 2018). Despite the encouraging progress, the *thematic consistency* and *creativity* of generated stories are still not satisfactory, which requires modeling long-range dependencies across the whole story and improving wording novelty respectively (Fan, Lewis, and Dauphin 2018).

For story composition, thematic consistency and wording novelty are to some extent mutually exclusive: consistent stories may have restricted word choice, while diversified wordings could lead to the risk of inconsistency. For one thing, all sentences of a well composed story are supposed to be connected with believable logic and thematic consistency. Previous story generation methods mainly focus on learning mid-level representations, e.g., event sequences (Martin et al. 2017) and prompts (keywords) (Fan, Lewis, and Dauphin 2018) to plan a story, where each sentence is generated with a specific event or keyword. Such a strategy is risky since the mid-level representations are generated in a single pass without considering the previous generated sentences. For another, stories generally contain vivid wordings, i.e. wording novelty. However, most of previous works have not formulated such an aspect and thus cannot guarantee the wording novelty of stories. These story generation models with recurrent neural networks (RNNs) are prone to generate common words with high occurrence frequencies (Zhang et al. 2017). The issue stems from the fact that RNN is inclined to learn local word co-occurrences, which fails to capture global semantic information such as topic (Bowman et al. 2015).

In fact, the above requirements, i.e. consistency and wording novelty, are not specific to the task of story generation, but also essential aspects for any text generation task outputting a long passage. In the paradigm of deep learning based text generation, one initial attempt for tackling these issues is to combine RNN with autoencoder for sequence learning, where the latent representation learned by the autoencoder has been proven appealing in modeling global attributes such as syntactic, thematic and discourse information (Li, Luong, and Jurafsky 2015). Later on, researchers enhanced autoencoder with the advantage of variational inference (Kingma and Welling 2014), also known as VAE, which can generate not only fluent but also novel word sequences (Bowman et al. 2015). To extend VAE for broader scenarios, conditional variational autoencoders (CVAE) are proposed to supervise the generation process of VAE conditioned on certain attributes while retaining the merits of VAE. It is confirmed in dialogue generation (Serban et al. 2017; Shen et al. 2017; Zhao, Zhao, and Eskenazi 2017) and poem composition (Yang et al. 2017) that CVAE can generate better responses and poems with creative words. Apart from autoencoders, researchers also explored enhancing deep neural models with memory unit, which is proven effective for modeling long-range dependencies, as demonstrated in machine translation (Feng et al. 2017; Meng et al. 2018) and poem generation (Zhang et al. 2017).

In this paper, we tackle the challenges of story genera-

*Corresponding author: Rui Yan (ruiyan@pku.edu.cn)

tion with a novel model, which simultaneously takes care of the *thematic consistency* and *wording novelty*. The fundamental module of our model is a generative model, i.e. CVAE, for improving wording novelty. As the introduced latent variable could bring in sentence-level diversity and even inferior consistency along with the accumulation of generated sentences, we propose to augment CVAE with a private cache [1] for capturing long-range dependencies so as to keep the consistency for the whole story. Moreover, we present an effective gate mechanism for integrating the cache with the CVAE part. To evaluate our proposed model, we conduct experiments on two challenging datasets, ROCStories and WritingPrompts (Fan, Lewis, and Dauphin 2018). Experimental results confirm that through generating sentences with the latent variable and the cache mechanism, the proposed model can generate stories with better consistency and wording. Quantitative and qualitative studies show that our model yields substantial improvement over state-of-the-art.

Preliminaries

VAE and CVAE

VAE comprises an encoder and a decoder, corresponding to the encoding process, i.e., mapping the input x to a latent variable z ($x \mapsto z$), and the decoding process, i.e., reconstructing x from the latent variable z ($z \mapsto x$). More concretely, the encoding process takes x as input to compute a posterior distribution $q_\theta(z \mid x)$, interpreted as the probability distribution of generating z conditioned on x. In a similar fashion, the decoding process is to compute a distribution $p_\theta(x \mid z)$, representing the probability distribution of reconstructing x conditioned on z, where z has a pre-specified prior distribution $p_\theta(z)$, i.e. standard Gaussian distribution. θ denotes the parameters of both encoder and decoder. However, in the practical situation, the integral of the marginal likelihood $p_\theta(x)$ is intractable (Kingma and Welling 2014), especially for large-scale datasets. Alternatively, the true posterior $q_\theta(z \mid x)$ is substituted by its variational approximation $q_\phi(z \mid x)$ to model the encoding process, where ϕ is the parameters of q.

For training a VAE model, the objective is to maximize the log-likelihood of reconstructing the input x, denoted as $\log p_\theta(x)$. To facilitate learning, maximizing $\log p_\theta(x)$ is converted to push up its variational lower bound:

$$\begin{aligned}\mathbb{L}(\theta, \phi; x) = &- \mathrm{KL}(q_\phi(z|x) \parallel p_\theta(z)) \\ &+ \mathbf{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]\end{aligned} \tag{1}$$

as a result of which the original objective $\log p_\theta(x)$ is optimized. Herein KL$(\cdot)$ represents the KL-divergence term, which serves as the regularization for encouraging the approximated posterior $q_\phi(z|x)$ to approach the prior $p_\theta(z)$, i.e. a standard Gaussian distribution. E$[\cdot]$ is the term of reconstruction loss, reflecting how well the decoder performs.

CVAE, as an extension of VAE, supervises the generation process under an extra condition c. Correspondingly, the objective of CVAE has the term of $\log p_\theta(x|z,c)$ (instead of $\log p_\theta(x|z)$), which represents the reconstruction log-likelihood of x conditioned on c. In consistent with VAE, the log-likelihood objective $\log p_\theta(x|c)$ is maximized through pushing up its variational lower bound:

$$\begin{aligned}\mathbb{L}(\theta, \phi; x, c) = &- \mathrm{KL}(q_\phi(z|x,c) \parallel p_\theta(z|c)) \\ &+ \mathbf{E}_{q_\phi(z|x,c)}[\log p_\theta(x|z,c)]\end{aligned} \tag{2}$$

where each item corresponds to Equation 1 except the introduced condition c, e.g., $q_\phi(z|x,c)$ and $p_\theta(z|c)$ represent the approximated conditional posterior and the conditional prior respectively, $\log p_\theta(x|z,c)$ represents the probability of reconstructing x conditioned on both z and c.

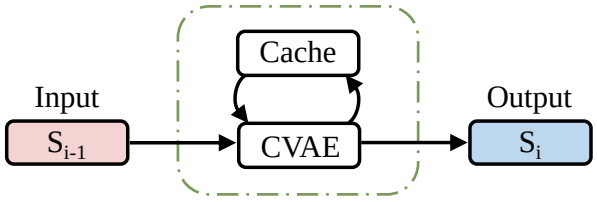

Figure 1: The overall diagram of our proposed cache augmented conditional variational auto-encoder generation model. For generating each story, the private cache of the current story will be updated and loaded to generate each sentence until the story end is generated.

Problem Formulation

We follow the same generation setting as in the previous story generation works (May and Knight 2018; Fan, Lewis, and Dauphin 2018), where each story is generated in a sentence-by-sentence fashion. As illustrated in Figure 1, the previous generated sentence serves as the input to generate the next one. We introduce some necessary notations as below to describe the generation process.

• INPUT. A title $T = (w_1, w_2, \ldots, w_N)$ is first given by a user as the input of the entire model to create a story, where w_i represents the i-th word and N is the length of the given title. Notice that the first sentence is outputted by only considering the title T. Thereafter, the model takes the previous generated sentence and the cache information to generate the next one until the story end is written.

• OUTPUT. A story $\{S_1, \ldots, S_n\}$ will be created incrementally as the output of the whole model, where there are totally n sentences in a generated story. Each sentence is represented as $S_i = (w_{i,1}, w_{i,2}, \ldots, w_{i,l})$, where $w_{i,j} \in V$ is the j-th word in the i-th sentence, and l is the length of S_i.

The Model

As shown in Figure 1, our cache-augmented CVAE model (CVAE-Cache) consists of two module, CVAE and a cache, where details are presented in the following parts.

The Cache Module

The cache module is introduced in our model for addressing the issue of thematic inconsistency for long passage generation. To achieve this goal, it provides to the decoder the

[1] We use the word "cache" here since it only stores the context information of the current story and will be cleared once the story is done.

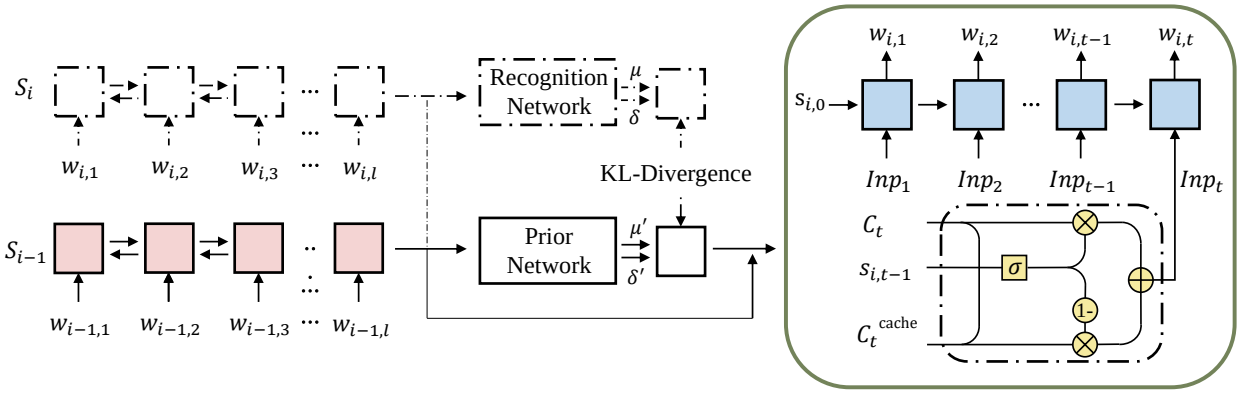

Figure 2: The detailed architecture of our CVAE-Cache model. The entire model is used during the training stage, while only the part with solid lines is used during the test stage, such as the prior network. Operations colored in yellow comprise the gate mechanism to integrate information from the cache and the decoder of CVAE.

cached information about the generated portion of the passage, thus the decoder's output, i.e. the current sentence in generation, is tied to the preceding portion of the story. Different from the global memory-like cache used in machine translation (Tu et al. 2017), our cache mechanism is a private one that works only for the current story in generation. The workflow of our private cache mechanism is as follows.

• Cache Building. In the sentence-by-sentence story generation fashion, the sentence in generation S_i is immediately tied up with the previous generated sentence S_{i-1}, while simultaneously demanding to summarize gist or logic line from the generation history as the hint for keeping thematic consistency. Thus, the generation history can be decomposed into two different parts, i.e. S_{i-1} and the rest $S_{1...i-2}$. As information in S_i is well learned by the encoder of the CVAE which is presented in the next subsection, the cache is implemented for summarizing gist from $S_{1...i-2}$. In doing so, the cache stores M words of $S_{1...i-2}$ right before the sentence S_{i-1} as the generation history, which can be denoted as $Cache = (m_1, m_2, \ldots, m_k, \ldots, m_M)$, where the cache size is M and m_k represents a word.

• Cache Reading. To read the words stored in $Cache$, we use a bidirectional GRU network (Chung et al. 2014) to encode these words into their corresponding hidden states, written as $(\boldsymbol{h}_1^{cache}, \boldsymbol{h}_2^{cache}, \ldots, \boldsymbol{h}_k^{cache}, \ldots, \boldsymbol{h}_M^{cache})$, where $\boldsymbol{h}_k^{cache}$ is the vector representation of word m_k in $Cache$. Note that this GRU network is different from the one of the encoder in CVAE part because the information in cache needs to be compressed to extract thematic hint. After acquiring the hidden state representations, we use an attention mechanism (Bahdanau, Cho, and Bengio 2014) to compute a weighted sum over all words in the cache at each decoding step, denoted as C_t^{cache}.

• Cache Updating. The cache is devised as a queue-like structure for updating: it basically follows a first-in-first-out updating mechanism, but a batched enQueue operation is adopted. Concretely, after the sentence S_i is generated from S_{i-1}, the cache enQueues the entire S_{i-1} at the front, with the cache size kept as M, the deQueue operation happens at the rear for extra cached words.

Cache-Augmented CVAE

We exploit a CVAE as the key portion of our CVAE-Cache model to improve wording novelty. The reason is that RNNs generate sentences via breaking the word sequence into next-step predictions, which can be easily trapped into local statistics, resulting in frequently occurred phrases or sentence segments (Li et al. 2015). CVAE explicitly introduces a latent variable for learning representation of global features like the topic or high-level syntactic properties to force RNNs to escape from the "local statistics" trap. As a result, sentences generated by CVAE consists of novel and meaningful words rather than common ones, which is also observed in sentence generation (Bowman et al. 2015).

The CVAE-Cache model consists of an encoder and a decoder. As demonstrated in Figure 2, we use a bidirectional GRU network (Chung et al. 2014) as the encoder to encode each sentence with shared parameters. At each step, the sentence S_{i-1} and its subsequent sentence S_i are mapped into the concatenated backward and forward vectors $h_{i-1} = [\overrightarrow{h}_{i-1}, \overleftarrow{h}_{i-1}]$ and $h_i = [\overrightarrow{h}_i, \overleftarrow{h}_i]$, respectively. Notice that h_{i-1} corresponds to the condition c while h_i corresponds to x in Equation 2. In consistent with previous works (Kingma and Welling 2014; Zhao, Zhao, and Eskenazi 2017), we hypothesize that the approximated variational posterior follows an isotropic multivariate Gaussian distribution $\mathcal{N}$, i.e. $q_\phi(z|x,c) = \mathcal{N}(\mu, \sigma^2\boldsymbol{I})$, where $\boldsymbol{I}$ represents the diagonal covariance. Thus modeling $q_\phi(z|x,c)$ is converted to learn μ and σ. Herein we parameterize μ and σ with the following neural network:

$$\begin{bmatrix} \mu \\ \log(\sigma^2) \end{bmatrix} = \boldsymbol{W}_r \begin{bmatrix} x \\ c \end{bmatrix} + b_r \tag{3}$$

which is presented as the recognition network in Figure 2. W_r and b_r are trainable parameters. Similarly, the prior follows another multivariate Gaussian distribution, i.e. $p_\theta(z|c) = \mathcal{N}(\mu', \sigma'^2\boldsymbol{I})$. Its key parameters μ' and σ' are learned by a single-layer fully-connected neural network

(the prior network in Figure 2) with $tanh(\cdot)$ as the activation function:

$$\begin{bmatrix} \mu' \\ \log\left(\sigma'^2\right) \end{bmatrix} = \mathbf{MLP}_p\left(c\right) \tag{4}$$

The decoder is a one-layer GRU network to output the sentence in generation, which is shown in the right hand of Figure 2. Take the sentence S_i as an example, the initial state of the decoder is calculated as:

$$s_{i,0} = W_d[z, c] + b_d \tag{5}$$

where W_d is a trainable matrix for dimension transformation. At each decoding step, the decoder takes Inp_t as input to compute its cell state $s_{i,t}$ and outputs a word $w_{i,t}$, where Inp_t is the combination of the previous decoding state $s_{i,t-1}$, the cache information C_t^{cache}, and the context information C_t in the previous generated sentence S_{i-1}. Specifically, the cached information is loaded by an attention mechanism, formulated by:

$$C_t^{cache} = \sum_{k=1}^{M} \alpha_{t,k} \boldsymbol{h}_k^{cache} \tag{6}$$

where $\alpha_{t,k}$ is the alignment probability (Bahdanau, Cho, and Bengio 2014) calculated by measuring the similarity between the current decoding state and each word's hidden state representation in the cache. Similarly, the context information C_t corresponds to summing up information in the previous sentence S_{i-1} through using an attention function identical with the cache reading.

To facilitate the combination of cache information and the decoder, we use a gate mechanism (Tu et al. 2017), which can be formulated as follows:

$$\lambda_t = \sigma(\boldsymbol{U} s_{i,t-1} + \boldsymbol{V} C_t + \boldsymbol{W} C_t^{cache}) \tag{7}$$

where $s_{i,t-1}$ is the decoding state of CVAE decoder at time $t-1$ and C_t is the encoder context. The final input for computing the decoding state used in generation step t is computed as:

$$Inp_t = (1 - \lambda_t) \otimes C_t^{cache} + \lambda_t \otimes C_t \tag{8}$$

For our cache-augmented CVAE model, the optimization objective is thus to maximize:

$$\begin{aligned} \mathcal{L}(\theta, \phi; c, x, Cache) = &-KL(q_\phi(z \mid x, c) \parallel P_\theta(z \mid c)) \\ &+ \mathbf{E}_{q_\phi(z|c,x)}[\log p(x \mid z, c, Cache)] \end{aligned} \tag{9}$$

where the KL terms is same as the one described in Equation 2, while the decoding process is to generate x from z, c, and the $Cache$ conditioned on the variational posterior q_ϕ. Note that θ and ϕ mentioned in the preliminary of VAE are not explicitly corresponded to a specific neural network described in this part, where ϕ refers to the parameters of the variational posterior, i.e $\phi = W_r, b_r$, while θ corresponds to all the remaining parameters.

Experiment

Datasets

To train our story generation model, we conduct experiments on two corpora: the ROCStories [2], and the WritingPrompts dataset. Specifically, the ROCStories corpus is created for the shared-task of Story Cloze Test (Cai, Tu, and Gimpel 2017; Schwartz et al. 2017), which is man-made with the following two merits: 1) It captures a rich set of commonsense relations of daily life; 2) It is a high-quality collection of life stories which can be used to learn story understanding and generation. As presented in Figure 3, each story of ROCStories comprises of exactly five sentences. The WritingPrompts dataset is collected from Reddit's WritingPrompts forum [3] for hierarchical story generation (Fan, Lewis, and Dauphin 2018), where each story has some corresponding prompts (i.e.keywords). Stories in WritingPrompts contain 734 words on average, which is substantially longer than those in ROCStories. Figure 4 shows an example story and its corresponding prompts, where the average length of prompts is 28.4.

> *Morgan and her family lived in Florida. They heard a hurricane was coming. They decided to evacuate to a relative's house. They arrived and learned from the news that it was a terrible storm. They felt lucky they had evacuated when they did.*

Figure 3: An example of ROCStories dataset. Each story contains exactly five sentences written by humans.

> Prompts:The Mage, the Warrior, and the Priest
> *A light breeze swept the ground, and carried with it still the distant scents of dust and time-worn stone. The Warrior led the way, heaving her mass of armour and muscle over the uneven terrain. She soon crested the last of the low embankments, which still bore the unmistakable fingerprints of haste and fear. She lifted herself up onto the top the rise, and looked out at the scene before her. [...]*

Figure 4: An example of WritingPrompts dataset (Fan, Lewis, and Dauphin 2018).

Totally, there are 98,159 stories in ROCStories and 303,358 in WritingPrompts. To preprocess ROCStories, we first applied NLTK for tokenization, and then we randomly split the data into 78,527/9,816/9,816 stories for training/validation/test. For WritingPrompts, we followed (Fan, Lewis, and Dauphin 2018) for preprocessing: The dataset is randomly split into 272,600/15,138/15,620 stories as training/validation/test sets; NLTK is utilized to tokenize these stories; Words with frequencies from 10 to 1000 will be loaded in the vocabulary, which results in a vocabulary size of 19,025 for the prompts and 104,960 for the stories.

[2] http://www.cs.rochester.edu/nlp/rocstories/

[3] www.reddit.com/r/WritingPrompts/

Baselines

In our experiments, we employ several highly related and strong story generation methods as our baselines.

C-LM, the conditional language model, straightforwardly utilizes an RNN language model with GRU cells to generate each word of a story in one pass conditioned on a given title, where the title is represented by a vector learned from another RNN language model.

AS2S, the standard scheme of sequence to sequence with attention (Bahdanau, Cho, and Bengio 2014). We use AS2S as our baseline since it serves as the benchmark of various language generation tasks and the recent story generation approaches are built upon it (Martin et al. 2017; Jain et al. 2017). This model creates a story incrementally in a sentence-by-sentence fashion. In details, the n-th sentence will be generated given the (n-1)-th sentence as input.

Cont-AS2S enhances the AS2S model by taking context information into account, which also creates a story in an incremental scheme, sentence-by-sentence, conditioned on the given title. Unlike AS2S, the Cont-AS2S generates the n-th sentence with all previous n-1 sentences as the input.

Hierarchical, the state-of-the-art story generation model on WritingPrompts (Fan, Lewis, and Dauphin 2018). This baseline is used to calibrate the performance of our proposed CVAE-Cache model on the WritingPrompts dataset.

CVAE, the conventional conditional variational autoencoder, utilizes the same generation pipeline with the AS2S, i.e. a whole story is generated by a sentence-by-sentence fashion with a title as the input. This baseline is used to investigate the performance of CVAE for story generation without the cache and gate mechanism.

AS2S-Cache, the combination of AS2S and our proposed cache, is utilized for studying whether our proposed cache and gate mechanism can improve the performance of AS2S.

Note that the Hierarchical model is not applicable to ROCStories dataset because there is no prompts in this dataset and the length of stories is five sentences. We thus launch all the baselines mentioned-above, except Hierarchical, to calibrate the performance of our proposed CVAE-Cache. As the Hierarchical model is proven to be better than existing models on the WritingPrompts, we only compare it with our CVAE-Cache model on this dataset.

Model Settings

We trained our model by adopting the following parameters and hyperparameters. Both encoder and decoder are formed by one layer. The hidden states dimension of encoder and decoder are both set to 500. The word embedding size is 300 and shared across everywhere. The vocabulary size is comprised of most frequent 30,000 words. The $Cache$ size is set to $\{20, 30, 50, 70\}$ for the ROCStories and $\{50, 100, 200, 300\}$ for the WritingPrompts corpus. The size of the latent variable z is 300. The prior network consists of 1 hidden layer with dimension of 400 and $tanh$ activation function. All initial weights are uniformly sampled from $[-0.08, 0.08]$. The batch size is set to 80. We use Adam optimizer (Kingma and Ba 2014) with learning rate of 0.001 and gradient clipping of 5 to train our models in an end-to-end fashion.

Readability (Rd.)	Is the story grammatically formed?
Consistency (Con.)	Does the story display a consistent theme?
Wording (Wod.)	Does the story narrate with novel words?
Overall (Ovr.)	The average score of the above three criteria.

Table 1: Criteria of human evaluation.

Evaluation

It is generally difficult to judge the quality of stories generated by computers. We put forward to evaluate the experimental results from three different aspects.

Overlap-Based Metric. The BLEU scores (Papineni et al. 2002) are designed for analyzing the word overlapping between the ground-truth and generated ones, and employed by previous works for evaluating generated stories (Martin et al. 2017; Wang et al. 2018).

Distinct Score. To evaluate the wording of generated stories, we adapt the distinct score from the dialogue task (Li et al. 2015) to measure the wording diversity of the stories, which counts the proportion of distinctive [1,4]-grams in the generated stories, shown as D-1, D-2, D-3, D-4 in Table 2. The final distinct scores are normalized to [0,100].

Human Evaluation. There are five well-educated human evaluators to score each story from three aspects: readability, consistency, and wording, as defined in Table 1. Each aspect is annotated with three score levels: 1, 2, and 3, and a higher score means a better performance. Totally, 150 randomly selected stories for each model are evaluated in a blind way. Following the conventional setting (Fan, Lewis, and Dauphin 2018), the story length is limited to 200 words for models trained on the WritingPrompts dataset.

Results

The Effect of CVAE

Table 2 presents the result of both automatic and human evaluations on ROCStories and WritingPrompts, respectively. Table 3 supplements the automatic evaluations of different cache size settings on two datasets. We analyze these results from the following perspectives.

CVAE part effectively improves wording novelty. As demonstrated in Table 2 and Table 3, CVAE outperforms all baselines in terms of distinct score, i.e. a fairly large proportion words of generated stories are distinctive. With the diversified wordings, the wording score of human evaluation also confirms that stories generated by CVAE provide better user experience. These results support the intuition that CVAE can address the issue of common wordings in RNNs through introducing a variational latent variable.

The latent variable in CVAE module introduces thematic departure. Note again that thematic consistency and wording novelty are to some extent mutually exclusive. Not surprisingly, CVAE yields a worse thematic consistency since there is no mechanism in CVAE for keeping thematic consistency while the latent variable produces more uncertainty than in RNNs, which is supported by the consistency score in human evaluation.

Datasets	Model	Automatic Evaluation								Human Evaluation			
		B-1	B-2	B-3	B-4	D-1	D-2	D-3	D-4	Rd.	Con.	Wod.	Ovr.
ROCStories	AS2S	19.7	5.42	1.91	0.78	0.28	1.46	3.71	6.92	1.28	1.13	1.10	1.17
	C-LM	23.5	8.32	3.14	1.12	1.34	10.12	28.0	49.3	2.22	2.19	1.30	1.90
	Cont-AS2S	23.6	6.79	1.86	0.58	1.42	12.3	32.0	55.5	2.43	2.19	1.37	1.95
	CVAE	25.5	7.55	2.31	0.84	**1.75**	16.8	49.0	78.7	2.65	1.47	**2.05**	2.05
	AS2S-Cache	25.7	8.54	2.52	1.15	1.48	15.3	39.6	63.6	2.38	2.32	1.48	2.06
	CVAE-Cache	**27.8**	**9.52**	**3.20**	**1.20**	1.61	**17.9**	**52.1**	**81.6**	**2.68**	**2.55**	1.61	**2.28**
WritingPrompts	Hierarchical	**31.1**	**6.05**	1.56	0.33	0.31	4.33	17.9	40.2	**2.24**	1.71	1.60	1.85
	CVAE-Cache	30.5	5.89	**1.84**	**1.03**	**1.02**	**11.4**	**24.3**	**59.2**	2.11	**1.82**	**2.04**	**1.99**

Table 2: Results of automatic ($p<0.01$) and human evaluations. **B-**n represents BLEU scores on [1,4]-grams; **D-**n corresponds to the distinct score of n-gram, with $n = 1$ to 4; **Rd., Con., Wod., Ovr.** represent readability, consistency, wording, and overall.

The Influence of the Cache

As mentioned previously, the cache module is introduced for modeling long-distance dependencies in generation history so as to improve thematic consistency. We conduct the following two groups experiments, i.e. AS2S-Cache VS AS2S and CVAE-Cache VS CVAE, as ablation study to investigate how the cache module affect the generated stories. Besides, we also explore the influence of different cache size. Following observations are concluded from these experiments, which verifies that the cache module substantially enhances existing models in thematic consistency modeling without producing other effects.

The cache module is effective for modeling thematic consistency. As illustrated by the thematic consistency score in Table 2, CVAE and AS2S can generate stories with better thematic consistency with the enhancement of cache module. Automatic evaluations, i.e. BLEU-[1,4] scores, also confirm that the cache module can improve the overlapping between generated stories and corresponding ground-truth stories, which reflects the generated stories are more relevant to thematically consistent cases.

The cache does not damnify the readability of generated stories. As the cached information is introduced as extra "hint" at each decoding step, the cache module is expected to not yield inferior fluency compared to the RNNs decoder that it is based on. The results of readability score confirm that CVAE-Cache and AS2S-Cache achieve comparable performance with CVAE and AS2S.

The cache does not affect the wording novelty. The motivation of utilizing cache module is to enhance the CVAE model with better thematic consistency while limiting inferior effect on wording novelty. The results of distinct scores in Table 2 prove that the cache module does not affect the wording novelty. Although there is a gap between the performance of CVAE and CVAE-Cache in terms of wording novelty score, the CVAE-Cache still outperforms all other baselines.

A relative small cache size is enough for modeling thematic consistency. As presented in Table 3, automatic evaluation results on ROCStories suggest that a relative small cache size, i.e. 20 or 30, achieves the best performance. A similar observation is also observed on the WritingPrompts dataset, i.e. the cache size is set to 100 which is a small one considering the average length of 734 words in stories.

Datasets	Size	BLEU-1	BLEU-2	BLEU-3	BLEU-4
ROCS	20	**27.8**	**9.52**	3.20	1.20
	30	27.7	9.49	**3.26**	**1.25**
	50	25.6	8.71	3.07	1.17
	70	24.6	7.98	2.70	0.98
WritingPs	50	30.1	5.34	1.76	0.98
	100	**30.5**	**5.89**	**1.84**	**1.03**
	200	29.6	5.28	1.70	0.94
	300	28.5	5.03	1.55	0.87

Table 3: BLEU scores of different cache size settings.

Discussion

For the ROCStories dataset, our proposed CVAE-Cache model substantially outperforms all baseline models in both human evaluations and automatic metrics. Especially for wording novelty and thematic consistency, CVAE-Cache achieves significant improvement on both sides, where wording novelty and thematic consistency are enhanced by the CVAE and Cache module, respectively. With the enhancement of the gate mechanism, the cache module and CVAE part are seamlessly integrated with their merits reserved while limiting inferior effects on other aspects. For results on WritingPrompts, we observe that our CVAE-Cache achieves comparable BLEU-1, BLEU-2, readability results with the strong baseline Hierarchical. Since there are various pre-train strategies and much more parameters in the Hierarchical story generation model, the comparable results is a promising one for a simple cache mechanism and CVAE model. Moreover, our model achieves better diversity performance and overall score in human evaluations, which represents a better user experience. We also observe that the overall human evaluation performance on the WritingPrompts is worse than those results on ROCStories, which can be explained by the difficulty of automatically generating long stories with thematic consistency and readability.

Case Study

Figure 5 shows a few example stories generated by CVAE-Cache model on the ROCStories dataset. We can observe that the model can generate stories with good consistency, which is represented by the character gender and events. With the given short titles, the generated stories are readable and correlate well with the titles. Figure 6 presents an example story generated on the WritingPrompts corpus,

Title:Bonus Round
Nola wanted to get a bonus for twenty dollars. She was sure she would not pay the bills. She waited for the price and started running. But Nola did not get the card and counted. She was unhappy for it!
Title:Picnic
Amy loved to cook in her new picnic. She took them outside and watched for the kids. She immediately made her lunch. Amy was finally able to finish the sandwich. Amy was happy she could eat potato salad for lunch.
Title:The Art Museum
We were enjoying the science fair. The teachers were thrilled to find the great art museum was there. We bought it a try. It was funny and didn't need to do anything. We made a new show and went back to class.

Figure 5: Example stories generated by the CVAE-Cache model, which is trained on the ROCStories corpus.

which is much longer than stories generated on ROCStories dataset. The result shows that our CVAE-Cache model can improve both the wording novelty and thematic consistency. Although there exists a gap between generated stories and human-comprised ones, the readable long story still demonstrates the superiority of our proposed CVAE-Cache model.

In the experiments, we also observed a few inferior cases generated by our models. A notable pattern is that there exists fine-grained inconsistency in the stories, e.g., sentiment, emotion, logic. To address such issue, one potential solution is to add corresponding mechanisms in our model such as sentiment controlling, event planning. Another error pattern is that there exist a few extraordinary worse cases for long story generation on WritingPrompts, which can be explained by the accumulation of randomness introduced by the latent variable in CVAE. To alleviate such issue, one can combine strategies such as editing or polishing during the generation process. Besides, it is worth to use generative adversarial network (GAN) (Goodfellow et al. 2014) for story generation. These results point out the direction of future work.

Related Work

Story Generation

Automated story generation involves automatically outputting a sequence of events or actions that meets a set of criteria (Martin et al. 2017), e.g., requiring thematic consistency and creativity (Fan, Lewis, and Dauphin 2018). In early years, researchers mainly focused on designing planning-based algorithms (Riedl and Young 2010), case-based reasoning methods (Gervás et al. 2005). However, these systems are limited to the topics that are covered by specific domain knowledge. To address domain limitations of story generation, the *SayAnything* system (Swanson and Gordon 2012) is proposed for collaboratively writing textual narratives with human users through using knowledge extracted from a large repository of nonfiction personal stories. Recently, researchers have paid attention to generate coherent stories with sequential information (Huang et al. 2016). Jain et al. (2017) explored to address the task of coherent story generation from independent descriptions. Martin et al. (2017) decompose the open story writing process into two steps, i.e., generating event sequences, and generating a sentence from an event to address the issue of event sparsity. Fan et.al (2018) combine hierarchical generation pipeline with the convolutional sequence to sequence model for improving the thematic consistency and creativity of stories.

Prompts:Rewrite your favorite song as a story.
There is a lot of commotion at the club, I know I am supposed to be a good person, I don' t like the girl noticed the sound of a light at the time and the door was shut off, but the look on her shoulder made me a dull. The young woman put the device back into her seat and pushed her way into the back seat of her chair. The feeling of the stomach tightened into a warm, the girl was still staring. The man dropped the knife and threw the rag into the air, the noise of the door slid away from the corner. The driver stood up and the door shut, the girl's eyes were still worried about me as well. I had the seat.

Figure 6: An example story outputted by the CVAE-Cache model, which is trained on the WritingPrompts dataset.

Conditional Variational Autoencoder

Bowman et al. (2015) first propose to employ VAE to generate sentences from a latent space. As a extend schema of VAE, the Conditional Variational Auto-Encoder is originally introduced for image generation given a certain attribute as the condition (Sohn, Yan, and Lee 2015; Yan et al. 2016). To alleviate the problem of vanishing latent variable, Zhao et al. (2017) have proposed a Bow loss. Yang et al. (2017) and Li et al.. (2018) proposed to combine CVAE for generating thematic poems. We proposed a novel model that combines CVAE with a private cache mechanism to improve the consistency of the generated story.

Conclusion

In this study, we proposed an effective method that combines cache and conditional variational autoencoder for addressing the story generation task. Specifically, we utilized CVAE to generate stories with diverse and novel words. To further improve the document-level thematic consistency, we augment the CVAE model with a cache module for capturing long-range dependencies. Through using a gate mechanism, the two parts are seamlessly integrated. Experimental results on two challenging datasets, ROCStories, and WritingPrompts, indicate that our model can generate stories with both thematic consistency and wording novelty. The qualitative study also confirms the validity of our proposed model.

Acknowledgments

We would like to thank the anonymous reviewers for their constructive comments. This work was supported by the National Key Research and Development Program of China (No. 2017YFC0804001), the National Science Foundation of China (NSFC No. 61672058; NSFC No. 61876196). Rui Yan was sponsored by CCF-Tencent Open Research Fund, Alibaba Innovative Research (AIR) fund, and Microsoft Research Asia (MSRA) Collaborative Research Program.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *Computer Science*.

Bowman, S. R.; Vilnis, L.; Vinyals, O.; Dai, A. M.; Jozefowicz, R.; and Bengio, S. 2015. Generating Sentences from a Continuous Space. *Computer Science*.

Brenner, M. 2010. Creating Dynamic Story Plots with Continual Multiagent Planning. In *AAAI*, 1529–1530.

Cai, Z.; Tu, L.; and Gimpel, K. 2017. Pay Attention to the Ending:Strong Neural Baselines for the ROC Story Cloze Task. In *ACL*, 616–622.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Fan, A.; Lewis, M.; and Dauphin, Y. 2018. Hierarchical Neural Story Generation. *arXiv preprint arXiv:1805.04833*.

Feng, Y.; Zhang, S.; Zhang, A.; Wang, D.; and Abel, A. 2017. Memory-augmented Neural Machine Translation. In *EMNLP*, 1390–1399.

Gervás, P.; Díaz-Agudo, B.; Peinado, F.; and Hervás, R. 2005. Story plot generation based on CBR. *Knowledge-Based Systems* 18(4):235–242.

Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative Adversarial Nets. In *NIPS*, 2672–2680.

Huang, T.-H. K.; Ferraro, F.; Mostafazadeh, N.; Misra, I.; Devlin, J.; Agrawal, A.; Girshick, R.; He, X.; Kohli, P.; Batra, D.; et al. 2016. Visual Storytelling. In *NAACL*.

Jain, P.; Agrawal, P.; Mishra, A.; Sukhwani, M.; Laha, A.; and Sankaranarayanan, K. 2017. Story Generation from Sequence of Independent Short Descriptions. *CoRR* abs/1707.05501.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kingma, D. P., and Welling, M. 2014. Auto-Encoding Variational Bayes. *stat* 1050:1.

Li, B.; Lee-Urban, S.; Johnston, G.; and Riedl, M. 2013. Story Generation with Crowdsourced Plot Graphs. In *AAAI*.

Li, J.; Galley, M.; Brockett, C.; Gao, J.; and Dolan, B. 2015. A Diversity-Promoting Objective Function for Neural Conversation Models. *Computer Science*.

Li, J.; Song, Y.; Zhang, H.; Chen, D.; Shi, S.; Zhao, D.; and Yan, R. 2018. Generating Classical Chinese Poems via Conditional Variational Autoencoder and Adversarial Training. In *EMNLP*, 3890–3900.

Li, J.; Luong, M.-T.; and Jurafsky, D. 2015. A Hierarchical Neural Autoencoder for Paragraphs and Documents. In *ACL*, volume 1, 1106–1115.

Martin, L. J.; Ammanabrolu, P.; Hancock, W.; Singh, S.; Harrison, B.; and Riedl, M. O. 2017. Event Representations for Automated Story Generation with Deep Neural Nets. *arXiv preprint arXiv:1706.01331*.

May, N. P. M. G. J., and Knight, K. 2018. Towards controllable story generation. *NAACL Workshop*.

Meng, F.; Tu, Z.; Cheng, Y.; Wu, H.; Zhai, J.; Yang, Y.; and Wang, D. 2018. Neural Machine Translation with Key-Value Memory-Augmented Attention. *IJCAI*.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W. J. 2002. IBM Research Report Bleu: a Method for Automatic Evaluation of Machine Translation. *ACL* 30(2):311–318.

Porteous, J.; Cavazza, M.; and Charles, F. 2010. Applying planning to interactive storytelling: Narrative control using state constraints. *ACM Transactions on Intelligent Systems and Technology* 1(2):1–21.

Riedl, M. O., and Young, R. M. 2010. Narrative planning: Balancing plot and character. *Journal of Artificial Intelligence Research* 39:217–268.

Schwartz, R.; Sap, M.; Konstas, I.; Zilles, L.; Choi, Y.; and Smith, N. A. 2017. The Effect of Different Writing Tasks on Linguistic Style: A Case Study of the ROC Story Cloze Task. In *CoNLL*, 15–25.

Serban, I. V.; Sordoni, A.; Lowe, R.; Charlin, L.; Pineau, J.; Courville, A. C.; and Bengio, Y. 2017. A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues. In *AAAI*, 3295–3301.

Shen, X.; Su, H.; Li, Y.; Li, W.; Niu, S.; Zhao, Y.; Aizawa, A.; and Long, G. 2017. A Conditional Variational Framework for Dialog Generation. In *ACL*, volume 2, 504–509.

Sohn, K.; Yan, X.; and Lee, H. 2015. Learning structured output representation using deep conditional generative models. In *NIPS*, 3483–3491.

Swanson, R., and Gordon, A. S. 2012. Say anything: Using textual case-based reasoning to enable open-domain interactive storytelling. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 2(3):16.

Tu, Z.; Liu, Y.; Shi, S.; and Zhang, T. 2017. Learning to Remember Translation History with a Continuous Cache. *arXiv preprint arXiv:1711.09367*.

Wang, X.; Chen, W.; Wang, Y.-F.; and Wang, W. Y. 2018. No metrics are perfect: Adversarial reward learning for visual storytelling. *arXiv preprint arXiv:1804.09160*.

Wiseman, S.; Shieber, S.; and Rush, A. 2017. Challenges in Data-to-Document Generation. In *EMNLP*, 2253–2263.

Yan, X.; Yang, J.; Sohn, K.; and Lee, H. 2016. Attribute2image: Conditional image generation from visual attributes. In *European Conference on Computer Vision*, 776–791. Springer.

Yang, X.; Lin, X.; Suo, S.; and Li, M. 2017. Generating Thematic Chinese Poetry with Conditional Variational Autoencoder. *arXiv preprint arXiv:1711.07632*.

Zhang, J.; Feng, Y.; Wang, D.; Wang, Y.; Abel, A.; Zhang, S.; and Zhang, A. 2017. Flexible and Creative Chinese Poetry Generation Using Neural Memory. In *ACL*, volume 1, 1364–1373.

Zhao, T.; Zhao, R.; and Eskenazi, M. 2017. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. *ACL*.