Latent Constraints on Unsupervised Text-Graph Alignment with Information Asymmetry

Jidong Tian[1, 2*], Wenqing Chen[3*], Yitian Li[1, 2], Caoyun Fan[1, 2], Hao He[1, 2†], Yaohui Jin[1, 2†]

[1] MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
[2] State Key Lab of Advanced Optical Communication System and Network,
School of Electronic Information and Electrical Engineering,
Shanghai Jiao Tong University
[3] School of Software Engineering, Sun Yat-sen University
frank92@sjtu.edu.cn, chenwq95@mail.sysu.edu.cn, {yitian_li, fcy3649, hehao, jinyh}@sjtu.edu.cn

Abstract

Unsupervised text-graph alignment (UTGA) is a fundamental task that bidirectionally generates texts and graphs without parallel data. Most available models of UTGA suffer from information asymmetry, a common phenomenon that texts and graphs include additional information invisible to each other. On the one hand, these models fail to supplement asymmetric information effectively due to the lack of ground truths. On the other hand, it is challenging to indicate asymmetric information with explicit indicators because it cannot be decoupled from the data directly. To address the challenge posed by information asymmetry, we propose the assumption that asymmetric information is encoded in unobservable latent variables and only affects the one-way generation processes. These latent variables corresponding to asymmetric information should obey prior distributions recovered approximately from original data. Therefore, we first propose a taxonomy of the latent variable that classifies the latent variable into transferrable (TV) and non-transferable (NTV) variables and further distinguish NTV as the dependent variable (DV) and the independent variable (IV). Next, we propose three latent VAE-based regularizations on TV, DV, and IV to constrain their distributions to well-designed prior distributions to introduce asymmetric information into models and enhance the preservation of shared contents. Finally, we impose the three proposed constraints on a cycle-consistent learning framework, back-translation (BT), named ConstrainedBT. Experiments on three UTGA tasks demonstrate the effectiveness of ConstrainedBT on the information-asymmetric challenge.

Introduction

Unsupervised text-graph alignment (UTGA) is the task of bidirectional text-graph generation without parallel data (Jin et al. 2020; Schmitt et al. 2020; Ke et al. 2021), which is fundamental to producing readable explanations (Cai and Lam 2020; Saha et al. 2021; Chairatanakul et al. 2021; Gai et al. 2021; Tian et al. 2022) and is used to make reasoning (Sinha et al. 2019; Tian et al. 2021; Huang et al. 2021; Li et al. 2022b) in NLU. However, most available methods on unsupervised alignment (Schmitt et al. 2020; Guo et al. 2020; Prabhumoye et al. 2018; Yi et al. 2020; Xiao et al. 2021; Ma et al. 2021) cannot solve a critical problem that is very common in UTGA: information asymmetry. We first illustrate information asymmetry in UTGA defined in Definition 1.

Information Asymmetry: Yang et al. (Yang et al. 2019) summarize three information conditions in text generation tasks: Source $\approx$ Target, Source $>$ Targe, and Source $<$ Target. However, there exists a more challenging condition in unsupervised alignment tasks: Source $\neq$ Target, the phenomenon of which is defined as information asymmetry in Definition 1. For the example in Figure 1, the text includes the information of the story scene (such as "visit" and "movie") that is not visible to the graph, while the graph also contains the logic (such as $\mathrm{Mother(x, y)} \wedge \mathrm{Mother(y, z)} \rightarrow \mathrm{Grandmother(x, z)}$) that is not necessary to be expressed in the text. It is challenging to eliminate its impacts on UTGA. Due to the lack of labels, models can hardly extract asymmetric information during training, like supervised learning (Yang et al. 2019; Liu, Wang, and Li 2021). In addition, asymmetric information is usually entangled in forms and contents so that we cannot operate on explicit spaces directly (Schmitt et al. 2020).

Definition 1. *Information asymmetry in UTGA is the phenomenon that the paired data contain additional information with complex forms invisible to each other.*

To address the information asymmetry problem in UTGA, we assume that all information, including asymmetric information, is encoded in the latent variable. This latent variable can be divided into three parts: transferable variable (TV) and non-transferable parts (NTV) with asymmetric information of the text and the graph, respectively. Furthermore, NTV is classified into the dependent variable (DV) and the independent variable (IV) according to its correlation to TV. Aiming at TV, we design a VAE-based transferable constraint (Kingma and Welling 2014) to preserve shared information between two generation processes. Considering NTV, we propose two other CVAE-based regularizations (Kingma et al. 2014; Jain, Zhang, and Schwing 2017) for DV and IV that force their distributions to approximate

[*] These authors contributed equally.

[†] Corresponding authors.

prior distributions, which can be recovered from the original data. These two constraints can effectively introduce the missing asymmetric information into one-way generation models, allowing the better convergence of models. Finally, we impose the three constraints on a cycle-consistent learning framework, back-translation (BT) (Sennrich, Haddow, and Birch 2016; Hoang et al. 2018), to achieve UTGA. The proposed model is named ConstrainedBT.

We conduct experiments on three available UTGA tasks. Logic2Text (Chen et al. 2020) includes a task without asymmetric information, while LogicNLI (Tian et al. 2021) and CLUTRR (Sinha et al. 2019) contain two information-asymmetric tasks. Results show that ConstrainedBT can converge faster than naive BT on Logic2Text and outperforms other unsupervised baselines on the information-asymmetric datasets, which supports that our proposed constraints can indeed solve the information asymmetric problem and enhance the unsupervised alignment model. To further understand these constraints, we analyze their priority and interactions from the view of curriculum learning (Wang, Chen, and Zhu 2021). Finally, we validate ConstrainedBT's benefits to downstream NLU tasks based on prompt learning (Liu et al. 2021b; Li and Liang 2021).

Contributions: Firstly, we analyze a critical problem of UTGA, information asymmetry. Secondly, we distinguish transferable and non-transferable latent variables and impose three implicit constraints on BT to introduce information asymmetry in latent spaces. Thirdly, experimental results show that our proposed method can effectively alleviate the negative impacts of asymmetric information on UTGA.

Related Works

Information Asymmetry in NLG

Information asymmetry is a common phenomenon in NLG. Some tasks, such as text style transfer,machine translation, and multi-modal tasks (Chen et al. 2021a), includes asymmetric information at the formal level. For example, text style transfer shows asymmetric styles (Prabhumoye et al. 2018; Yi et al. 2020; Xiao et al. 2021; Ma et al. 2021) while machine translation should bridge the gap between different languages (Wang et al. 2021; Li et al. 2022a; Nguyen et al. 2021). Other tasks, including text summarization and topic-to-essay generation, exhibit asymmetric information at the content level. Text summarization is a typical one-way asymmetric task that the information of the target needs to be filtered from the source (Lin and Ng 2019; Cao 2022). Inversely, topic-to-essay generation requires supplementary information to generate the target (Yang et al. 2019). Although information asymmetry is not a prominent problem in supervised learning (Chen et al. 2021b), it is a dominant challenge in unsupervised scenarios. Considering unsupervised text style transfer, it is widely studied due to its difficulty in introducing asymmetric information of style into the content (Prabhumoye et al. 2018; Ma et al. 2021).

Text-graph alignment is the bidirectional generation of the graph and the text from each other. Previous studies treat them as two independent supervised tasks: text-to-graph generation and graph-to-text generation (Song et al. 2020; Hoyle, Marasovic, and Smith 2021; Ren et al. 2021; Ke et al. 2021). Unsupervised text-graph alignment (UTGA) is a more challenging task because it includes bidirectional asymmetric information, as shown in Figure 1.

Cycle-Consistent Learning

Cycle-consistent learning (CCL) is commonly used to solve unsupervised tasks without parallel data (Nguyen et al. 2021; West et al. 2019; Ju et al. 2021), which trains unsupervised models through internally consistent loss functions, including adversarial loss (Shen et al. 2017), reconstruction loss (Luo et al. 2019), and auxiliary loss (Prabhumoye et al. 2018). In particular, it achieves surprising performance on unsupervised text style transfer. Shen et al. (Shen et al. 2017) first propose a general CCL framework that uses an adversarial loss to achieve unsupervised transferring. Based on the framework, the following works (Prabhumoye et al. 2018; Yi et al. 2020; Xiao et al. 2021; Ma et al. 2021) design new CCL frameworks introducing auxiliary tasks and diverse methods to balance style conversion and content preservation. Among these works, back-translation (BT) (Luo et al. 2019; Lample et al. 2019; He et al. 2020; Lai, Toral, and Nissim 2021) is a simple but effective framework for arbitrary unsupervised alignment tasks, which is first proposed to achieve data augmentation in NMT (Sennrich, Haddow, and Birch 2016; Pham et al. 2021). Hoang et al. (Hoang et al. 2018) and Cotterell and Kreutzer (Cotterell and Kreutzer 2018) improve the optimization strategy to an iterative one, named iterative back-translation (IBT). Aiming at UTGA, Schmitt et al. (Schmitt et al. 2020) also propose an LSTM-based cycle-consistent learning method (GT-BT), which adopts different initializations to control the generation of text sequence or the graph sequence with three unsupervised auxiliary objectives. On this basis, Guo et al. (Guo et al. 2020, 2021) introduce a general BT method to text-graph alignment directly and further optimize such the method through a CVAE module (CycleGT), achieving comparable performance with supervised methods on WebNLG (Gardent et al. 2017; Ferreira et al. 2020) without much asymmetric information. However, these available methods cannot deal with UTGA with information asymmetry due to the lack of effective means to introduce or indicate such complex asymmetric information.

Formulation

The generation process of the text-graph pair can be described under two hypotheses.

Hypothesis 1 (Transferable Hypothesis). *A graph $\boldsymbol{g}$ is generated from the conditional distribution $p(\boldsymbol{g}|\boldsymbol{z}, \boldsymbol{z_g})$, while a text $\boldsymbol{t}$ is generated from the conditional distribution $p(\boldsymbol{g}|\boldsymbol{z}, \boldsymbol{z_t})$, where $\boldsymbol{z}$ is the transferable latent variable, and $\boldsymbol{z_g}$ and $\boldsymbol{z_t}$ are non-transferable latent variables.*

Hypothesis 2 (Disentangling Hypothesis). *There are only two cases of dependency between $\boldsymbol{z_g}$ ($\boldsymbol{z_t}$) and $\boldsymbol{z}$. If $\boldsymbol{z_g}$ ($\boldsymbol{z_t}$) and $\boldsymbol{z}$ can be totally disentangled, then $\boldsymbol{z_g}$ ($\boldsymbol{z_t}$) and $\boldsymbol{z}$ are independent; otherwise, $\boldsymbol{z_g}$ ($\boldsymbol{z_t}$) is dependent on $\boldsymbol{z}$.*

Based on the two hypotheses, latent variables can be divided into three types. Definition 2 distinguishes transferable (TV) and non-transferable variables (NTV) clearly,

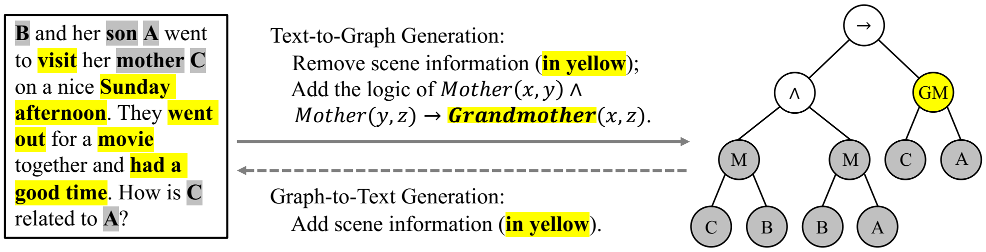
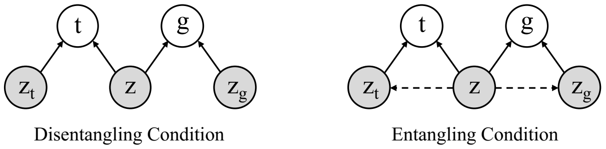

B and her son A went to visit her mother C on a nice Sunday afternoon. They went out for a movie together and had a good time. How is C related to A?
Text-to-Graph Generation:
Remove scene information (in yellow);
Add the logic of $Mother(x, y) \wedge Mother(y, z) \rightarrow \boldsymbol{Grandmother}(x, z)$.
Graph-to-Text Generation:
Add scene information (in yellow).
→
∧
GM
M
M
C
A
C
B
B
A

Figure 1: An example of information asymmetry in text-graph alignment. “M” and “GM” represent “Mother” and “Grandmother”, respectively. Hightlights in yellow represent asymmetric information of the text/graph, while highlights in grey represent information that is transferable.



Disentangling Condition

Entangling Condition

Figure 2: Variable dependencies under disentangling and entangling conditions. Line arrows represent dependencies defined in Hypothesis 1, while dashed arrows represent ones in Hypothesis 2. White circles mean observed variables, while grey circles mean latent variables.

while Definition 3 further distinguishes two different non-transferable variables (IV and DV).

Definition 2. ***Transferable Variable (TV)*** *is defined as the part of the latent variable that can be extracted by both “perfect” text and graph encoders, while* ***Non-transferable Variable (NTV)*** *is the remained part that can only be encoded by either “perfect” text or graph encoder.*

Definition 3. *If the non-transferable variable is totally independent of the transferable variable, it is defined as* ***Independent Variable (IV)****; otherwise, it is named* ***Dependent Variable (DV).***

According to the definitions, there are two conditions among latent variables in UTGA, as shown in Figure 2, where $\boldsymbol{z}$ represents TVs, while $\boldsymbol{z_t}$ and $\boldsymbol{z_g}$ are NTVs. As a result, the text-graph pair can be sampled from joint distributions of $p(\boldsymbol{t}, \boldsymbol{g})$ (in Equation 1 and Equation 2 where independent conditions are also given) based on $\boldsymbol{z}$, $\boldsymbol{z_t}$, and $\boldsymbol{z_g}$, under the disentangling and entangling conditions. The essence of the sampling process includes four steps: 1) sampling the transferable variable $\boldsymbol{z}$ from a prior distribution $p(\boldsymbol{z})$; 2) sampling non-transferable variables $\boldsymbol{z_t}$ and $\boldsymbol{z_g}$ from prior distributions or conditional distributions; 3) Sampling the consistent graph g and text t according to $\boldsymbol{z}$, $\boldsymbol{z_t}$, and $\boldsymbol{z_g}$.

$$\begin{aligned} p(\boldsymbol{t}, \boldsymbol{g}) =& \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z})} \mathbb{E}_{\boldsymbol{z_t} \sim p(\boldsymbol{z_t})} \mathbb{E}_{\boldsymbol{z_g} \sim p(\boldsymbol{z_g})} [p(\boldsymbol{t}, \boldsymbol{g} | \boldsymbol{z_t}, \boldsymbol{z_g}, \boldsymbol{z})] \\ & (\boldsymbol{z_t} \perp \boldsymbol{z}, \boldsymbol{z_g} \perp \boldsymbol{z}, \boldsymbol{z_t} \perp \boldsymbol{z_g}) \end{aligned} \quad (1)$$

$$\begin{aligned} p(\boldsymbol{t}, \boldsymbol{g}) =& \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z})} \mathbb{E}_{\boldsymbol{z_t} \sim p(\boldsymbol{z_t} | \boldsymbol{z})} \mathbb{E}_{\boldsymbol{z_g} \sim p(\boldsymbol{z_g} | \boldsymbol{z})} [p(\boldsymbol{t}, \boldsymbol{g} | \boldsymbol{z_t}, \boldsymbol{z_g}, \boldsymbol{z})] \\ & (\boldsymbol{z_t} \perp \boldsymbol{z_g} | \boldsymbol{z}) \end{aligned} \quad (2)$$

In UTGA, we can only observe two unpaired datasets of texts $\boldsymbol{T} = \{\boldsymbol{t_1}, \boldsymbol{t_2}, \cdots, \boldsymbol{t_n}\}$ and graphs $\boldsymbol{G} = \{\boldsymbol{g_1}, \boldsymbol{g_2}, \cdots, \boldsymbol{g_m}\}$, and latent variables ($\boldsymbol{z}$, $\boldsymbol{z_t}$, and $\boldsymbol{z_g}$) are unobservable. Therefore, our objective is to learn the conditional distributions of $p(\boldsymbol{t}|\boldsymbol{g})$ and $p(\boldsymbol{g}|\boldsymbol{t})$ under the generative assumption (Shen et al. 2017) shown in Equation 3 and Equation 4 under two conditions in Figure 2.

$$\begin{aligned} p(\boldsymbol{t}|\boldsymbol{g}) =& \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z}|\boldsymbol{g})} \mathbb{E}_{\boldsymbol{z_t} \sim p(\boldsymbol{z_t})} [p(\boldsymbol{t} | \boldsymbol{z_t}, \boldsymbol{z})] \\ & (\boldsymbol{z_t} \perp \boldsymbol{g}, \boldsymbol{t} \perp \boldsymbol{g} | \boldsymbol{z}, \boldsymbol{z_t} \perp \boldsymbol{z}) \\ p(\boldsymbol{g}|\boldsymbol{t}) =& \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z}|\boldsymbol{t})} \mathbb{E}_{\boldsymbol{z_g} \sim p(\boldsymbol{z_g})} [p(\boldsymbol{g} | \boldsymbol{z_g}, \boldsymbol{z})] \\ & (\boldsymbol{z_g} \perp \boldsymbol{t}, \boldsymbol{g} \perp \boldsymbol{t} | \boldsymbol{z}, \boldsymbol{z_g} \perp \boldsymbol{z}) \end{aligned} \quad (3)$$

$$\begin{aligned} p(\boldsymbol{t}|\boldsymbol{g}) =& \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z}|\boldsymbol{g})} \mathbb{E}_{\boldsymbol{z_t} \sim p(\boldsymbol{z_t}|\boldsymbol{z})} [p(\boldsymbol{t} | \boldsymbol{z_t}, \boldsymbol{z})] \\ & (\boldsymbol{z_t} \perp \boldsymbol{g} | \boldsymbol{z}, \boldsymbol{t} \perp \boldsymbol{g} | \boldsymbol{z}) \\ p(\boldsymbol{g}|\boldsymbol{t}) =& \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z}|\boldsymbol{t})} \mathbb{E}_{\boldsymbol{z_g} \sim p(\boldsymbol{z_g}|\boldsymbol{z})} [p(\boldsymbol{g} | \boldsymbol{z_g}, \boldsymbol{z})] \\ & (\boldsymbol{z_g} \perp \boldsymbol{t} | \boldsymbol{z}, \boldsymbol{g} \perp \boldsymbol{t} | \boldsymbol{z}) \end{aligned} \quad (4)$$

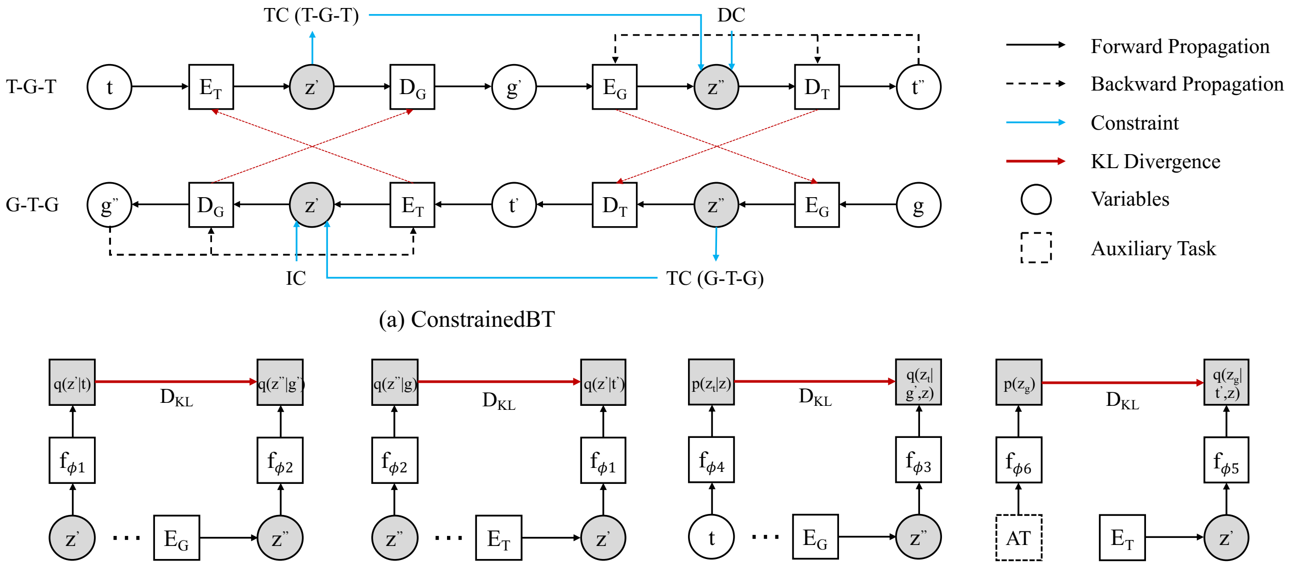
The objective suggests recovering conditional distributions based on $\boldsymbol{T}$ and $\boldsymbol{G}$ in three steps: 1) recovering the transferable conditional distributions based on $\boldsymbol{T}$ and $\boldsymbol{G}$ and sampling transferable latent variables; 2) constructing the distribution of the non-transferable distributions and sampling non-transferable variables; 3) modeling the probability of $\boldsymbol{g}$ and $\boldsymbol{t}$ conditioning on latent variables.

Methodology

Figure 3 shows the overall design of ConstrainedBT. We first illustrate the selection of neural architectures of E_T, E_G, D_T, and D_G and then explain how BT works on UTGA in Figure 3 (a). Finally, we focus on our proposed three VAE-based constraints shown in Figure 3 (b-e).

Neural Architectures

We first introduce the neural architectures to encode latent variables and formulate probabilities of $\boldsymbol{t}$ and $\boldsymbol{g}$ in Figure 3. In text-to-graph generation, we use a text encoder $E_T(\theta_1)$ to extract latent variables and a graph decoder $D_G(\theta_2)$ to formulate $p(\boldsymbol{g}|\boldsymbol{z}, \boldsymbol{z_g})$, where $\theta = [\theta_1, \theta_2]$ represents trainable parameters. We adopt commonly used models, RoBERTa (Liu et al. 2019) with adapter (Houlsby et al. 2019; Gururangan et al. 2020) and GraphRNN (You et al. 2018), as the text encoder and the graph decoder, respectively. Similarly, in graph-to-text generation, we introduce GAT (Velickovic et al. 2018) and GPT-2 (Radford et al. 2019) with adapter (Houlsby et al. 2019; Gururangan et al.



(a) ConstrainedBT

(b) TC (T-G-T)

(c) TC (G-T-G)

(d) DC

(e) IC

Figure 3: The framework of ConstrainedBT. (a) shows a global design based on BT, while (b), (c), (d), and (e) provide local structures of three constraints. The legend is on the right. E, D, T, G mean encoder, decoder, text, and graph, respectively. TC, DC, and IC represent transferable constraint, dependent constraint, and independent constraint, respectively.

2020) as the graph encoder ($E_G(\psi_1)$) and the text decoder ($D_T(\psi_2)$), which extracts latent variables from graphs and approximate $p(\boldsymbol{t}|\boldsymbol{z}, \boldsymbol{z_t})$, respectively. $\psi = [\psi_1, \psi_2]$ means trainable parameters. We use $\boldsymbol{z'}$ and $\boldsymbol{z''}$ to represent outputs of E_T and E_G, respectively. Ideally, $\boldsymbol{z'}$ encodes $\boldsymbol{z}$ and $\boldsymbol{z_t}$, while $\boldsymbol{z''}$ implies $\boldsymbol{z}$ and $\boldsymbol{z_g}$.

Back-Translation

From Figure 3 (a), BT includes two processes of recovering $\boldsymbol{t}$ (T-G-T) and $\boldsymbol{g}$ (G-T-G) by 1) translating the original input to the target, and 2) recovering the input based on the generated target. Therefore, the reconstruction objectives of T-G-T and G-T-G are to minimize losses shown in Equation 5, where $\boldsymbol{z'} = E_T(\boldsymbol{t}; \theta_1)$, $\boldsymbol{g'} = D_G(\boldsymbol{z'}; \theta_2)$, $\boldsymbol{z''} = E_G(\boldsymbol{g}; \psi_1)$, and $\boldsymbol{t'} = D_T(\boldsymbol{z''}; \psi_2)$. $p_{D_G}(\cdot; \theta_2)$ and $p_{D_T}(\cdot; \psi_2)$ are conditional probabilities of $p(\boldsymbol{g}|\boldsymbol{z}, \boldsymbol{z_g})$ and $p(\boldsymbol{t}|\boldsymbol{z}, \boldsymbol{z_t})$ approximated by D_G and D_T.

$$\begin{aligned}\mathcal{L}_T(\psi) &= -\mathbb{E}_{\boldsymbol{t}\sim \boldsymbol{T}}[\mathbb{E}_{\boldsymbol{z''}\sim q(\boldsymbol{z''}|\boldsymbol{t})} \log p(\boldsymbol{t}|\boldsymbol{z''})] \\ &\approx \mathbb{E}_{\boldsymbol{t}\sim \boldsymbol{T}}[-\log p_{D_T}(\boldsymbol{t}|E_G(D_G(E_T(\boldsymbol{t};\theta_1);\theta_2);\psi_1);\psi_2)] \\ \mathcal{L}_G(\theta) &= -\mathbb{E}_{\boldsymbol{g}\sim \boldsymbol{G}}[\mathbb{E}_{\boldsymbol{z'}\sim q(\boldsymbol{z'}|\boldsymbol{g})} \log p(\boldsymbol{g}|\boldsymbol{z'})] \\ &\approx \mathbb{E}_{\boldsymbol{g}\sim \boldsymbol{G}}[-\log p_{D_G}(\boldsymbol{g}|E_T(D_T(E_G(\boldsymbol{g};\psi_1);\psi_2);\theta_1);\theta_2)]\end{aligned} \tag{5}$$

Iterative Training: The equations imply an assumption that $p(\boldsymbol{g'}|\boldsymbol{t}) \approx 1$ and $p(\boldsymbol{t'}|\boldsymbol{g}) \approx 1$ for the generated $\boldsymbol{g'}$ and $\boldsymbol{t'}$. This assumption requires that D_G and D_T are functions of greedy search based on p_{D_T} and p_{D_G}, which cannot backpropagate gradients. As a result, θ and ψ cannot be updated simultaneously in a single direction. Instead of multi-task learning on bidirectional generation, we adopt the iterative strategy to train modules alternately, which has been proven effective (Cotterell and Kreutzer 2018). Therefore, the total training process of one iteration includes four steps: 1) Execute forward propagation of T-G-T; 2) Update ψ; 3) Execute forward propagation of G-T-G; 4) Update θ.

VAE-Based Constraints

Constraints on Latent Variables in UTGA: According to Definition 2, $\boldsymbol{z}$ belongs to TV. According to Definition 3, IV requires a more strict condition that all parts of NTV should be independent of TV, while DV only needs the verification of an existential condition. Considering $\boldsymbol{z_t}$, it encodes syntactic and asymmetric information of texts. Syntactic information is dependent on TV so that the existential condition can be satisfied, and thereby $\boldsymbol{z_t}$ belongs to DV. As for $\boldsymbol{z_g}$, it must contain parts that are not independent of $\boldsymbol{z}$. However, the core problem in text-to-graph generation is the introduction of logic, so the asymmetric information of the graph can be approximately equivalent to the logic rules R. Under this condition, $\boldsymbol{z_g}$ can be modeled as IV because R is independent of contents absolutely. We design three kinds of VAE-based constraints, named transferable constraint (TC), dependent constraint (DC), and independent constraint (IC) for $\boldsymbol{z}$ (TV), $\boldsymbol{z_t}$ (DV), and $\boldsymbol{z_g}$ (IV), respectively, which are illustrated in Figure 3 (b-e).

Transferable Constraint (TC): VAE imposes a distributional constraint on the reconstruction objective (Kingma and Welling 2014). In BT, maximizing the reconstruction terms of T-G-T and G-T-G is equivalent to minimizing losses in Equation 5. Inspired by VAE, TC's objectives are to minimize the second terms of $\mathrm{D}_{KL}(q(\boldsymbol{z}|\boldsymbol{t})||p(\boldsymbol{z})) = \mathrm{D}_{KL}(q(\boldsymbol{z}|\boldsymbol{g'}(\boldsymbol{t}))||p(\boldsymbol{z}))$ and $\mathrm{D}_{KL}(q(\boldsymbol{z}|\boldsymbol{g})||p(\boldsymbol{z})) = \mathrm{D}_{KL}(q(\boldsymbol{z}|\boldsymbol{t'}(\boldsymbol{g}))||p(\boldsymbol{z}))$ on $\boldsymbol{z}$, where

$\boldsymbol{g'}(\boldsymbol{t}) = D_G(E_T(\boldsymbol{t};\theta_1);\theta_2)$, $\boldsymbol{t'}(\boldsymbol{g}) = D_T(E_G(\boldsymbol{g};\psi_1);\psi_2)$, and $p(\boldsymbol{z})$ is the prior distribution of $\boldsymbol{z}$ (TV). To extract $\boldsymbol{z}$ from $\boldsymbol{z'}$/$\boldsymbol{z''}$ and calculate the posterior distributions of $\boldsymbol{z}$, we introduce two more neural functions: f_{ϕ_1} and f_{ϕ_2}, with trainable parameters ϕ_1 and ϕ_2. Therefore, $q(\boldsymbol{z}|\boldsymbol{g'}(\boldsymbol{t})) \approx q_{f_{\phi_2}}(E_G(\boldsymbol{g'}(\boldsymbol{t});\psi_1);\phi_2)$ and $q(\boldsymbol{z}|\boldsymbol{t'}(\boldsymbol{g})) \approx q_{f_{\phi_1}}(E_T(\boldsymbol{t'}(\boldsymbol{g});\theta_1);\phi_1)$. The remained challenge is to design the prior distribution $p(\boldsymbol{z})$ directly. In reality, TC allows $q(\boldsymbol{z}|\boldsymbol{t}) = p(\boldsymbol{z}) = q(\boldsymbol{z}|\boldsymbol{g})$, so we adopt approximate constraints making the two conditional distributions approximate each other in BT and allowing $q(\boldsymbol{z}|\boldsymbol{t}) = q(\boldsymbol{z}|\boldsymbol{g})$, which avoids to design $p(\boldsymbol{z})$ directly. TC's objectives of T-G-T and G-T-G are to minimize KL divergences in Equation 6.

$$\begin{aligned}\mathcal{L}_{TC(TGT)}(\phi_2,\psi_1) \approx& \mathbb{E}_{\boldsymbol{t}\sim\boldsymbol{T}}[\mathrm{D}_{KL}(q_{f_{\phi_2}}(E_G(\boldsymbol{g'}(\boldsymbol{t});\psi_1);\phi_2)|| \\ & q_{f_{\phi_1}}(E_T(\boldsymbol{t};\theta_1);\phi_1))] \\ \mathcal{L}_{TC(GTG)}(\phi_1,\theta_1) \approx& \mathbb{E}_{\boldsymbol{g}\sim\boldsymbol{G}}[\mathrm{D}_{KL}(q_{f_{\phi_1}}(E_T(\boldsymbol{t'}(\boldsymbol{g});\theta_1);\phi_1)|| \\ & q_{f_{\phi_2}}(E_G(\boldsymbol{g};\psi_1);\phi_2))]\end{aligned} \tag{6}$$

Dependent Constraint (DC): To constrain $\boldsymbol{z_t}$ (DV), we follow the previous work (Guo et al. 2021) that adopts CVAE to design the prior distribution based on the ground truth. DC assumes that the text $\boldsymbol{t}$ is generated from the latent variable $\boldsymbol{z_t}$ conditional on $\boldsymbol{z}$ implied by $\boldsymbol{g'}(\boldsymbol{t})$. The objective of DC is to minimize the second term of $\mathrm{D}_{KL}(q(\boldsymbol{z_t}|\boldsymbol{z},\boldsymbol{t})||p(\boldsymbol{z_t}|\boldsymbol{z}))$. Following the setting of amortized variational inference (Gershman and Goodman 2014), we introduce a neural function f_{ϕ_3} with trainable parameters ϕ_3 to approximate the posterior distribution based on $\boldsymbol{z''}$, which means that $q(\boldsymbol{z_t}|\boldsymbol{z},\boldsymbol{t}) \approx q_{f_{\phi_3}}(E_G(\boldsymbol{g'}(\boldsymbol{t});\psi_1);\phi_3)$. The prior distribution $p(\boldsymbol{z_t}|\boldsymbol{z})$ is inferred based on the ground truth $\boldsymbol{t}$ through another neural function f_{ϕ_4} with trainable parameters ϕ_4 representing $p(\boldsymbol{z_t}|\boldsymbol{z}) \approx p_{f_{\phi_4}}(\boldsymbol{t};\phi_4)$. Therefore, DC's objective is to minimize the KL divergence in Equation 7.

$$\begin{aligned}\mathcal{L}_{DC}(\phi_3,\phi_4,\psi_1) \approx& \mathbb{E}_{\boldsymbol{t}\sim\boldsymbol{T}}[\mathrm{D}_{KL}(q_{f_{\phi_3}}(E_G(\boldsymbol{g'}(\boldsymbol{t});\psi_1);\phi_3)|| \\ & p_{f_{\phi_4}}(\boldsymbol{t};\phi_4))]\end{aligned} \tag{7}$$

Independent Constraint (IC): Similar to DC, IC can also be established by CVAE. The only difference is that $\boldsymbol{z_g} \perp \boldsymbol{z}$, so the objective of IC is to minimize $\mathrm{D}_{KL}(q(\boldsymbol{z_g}|\boldsymbol{z},\boldsymbol{g})||p(\boldsymbol{z_g}))$ (Pandey and Dukkipati 2016). Symmetrically, the posterior distribution $q(\boldsymbol{z_g}|\boldsymbol{z},\boldsymbol{g})$ is approximated by $q_{f_{\phi_5}}(E_T(\boldsymbol{t'}(\boldsymbol{g});\theta_1);\phi_5)$. To design $p(\boldsymbol{z_g})$, we introduce an auxiliary task (AT) to generate $\boldsymbol{g}$ in an auto-regressive manner (You et al. 2018). The latent variable of the AT model ($\boldsymbol{z_{at}}$) has been proven to contain non-transferable information without transferable content (You et al. 2018) and can be used to recover $p(\boldsymbol{z_g})$. As a result, $p(\boldsymbol{z_g}) \approx p_{f_{\phi_6}}(\boldsymbol{z_{at}};\phi_6)$ where f_{ϕ_6} is a neural function with trainable parameters ϕ_6. IC's loss is shown in Equation 8.

$$\begin{aligned}\mathcal{L}_{IC}(\phi_5,\phi_6,\theta_1) \approx& \mathbb{E}_{\boldsymbol{g}\sim\boldsymbol{G}}[\mathrm{D}_{KL}(q_{f_{\phi_5}}(E_T(\boldsymbol{t'}(\boldsymbol{g});\theta_1);\phi_5)|| \\ & p_{f_{\phi_6}}(\boldsymbol{z_{at}};\phi_6))]\end{aligned} \tag{8}$$

Multi-Task Training: During training, we adopt multi-task learning for complete ConstrainedBT to optimize $\mathcal{L}_{TGT}$ and $\mathcal{L}_{GTG}$ in Equation 9 alternately with hyper-parameters of λ_1, λ_2, λ_3, and λ_4.

$$\begin{aligned}\mathcal{L}_{TGT}(\psi,\phi_2,\phi_3,\phi_4) =& \mathcal{L}_T(\psi) + \lambda_1\mathcal{L}_{TC(TGT)}(\phi_2,\psi_1) \\ & + \lambda_3\mathcal{L}_{DC}(\phi_3,\phi_4,\psi_1) \\ \mathcal{L}_{GTG}(\theta,\phi_1,\phi_5,\phi_6) =& \mathcal{L}_G(\theta) + \lambda_2\mathcal{L}_{TC(GTG)}(\phi_1,\theta_1) \\ & + \lambda_4\mathcal{L}_{IC}(\phi_5,\phi_6,\theta_1)\end{aligned} \tag{9}$$

Experiments and Results

Experimental Settings and Baselines

Dataset: We experiment on three available datasets: Logic2Text (Chen et al. 2020), LogicNLI (Tian et al. 2021), and CLUTRR (Sinha et al. 2019). Logic2Text is relatively easy because the information between its text and graph is almost symmetric. Different from Logic2Text, both LogicNLI and CLUTRR include asymmetric information, which has been illustrated in Figure 1.

Metric: Following the previous work (Guo et al. 2021), we adopt F1 scores (node/edge) to measure text-to-graph generation, while we use BLEU (Papineni et al. 2002), ROUGE (Lin 2004), CIDEr (Vedantam, Zitnick, and Parikh 2015), and METEOR (Banerjee and Lavie 2005) in graph-to-text generation.

Training Setting: We pretrain our model with only one constraint and impose the other two later to avoid interference among them. We execute the pretraining with IC on graph generation, which will be illustrated in Section .

Baselines: We adopt two generative language models and two unsupervised baselines as the following. 1) GPT-2 (Adapter) (Radford et al. 2019) and 2) BART (Adapter) (Lewis et al. 2020) with BT framework. 3) Graph-Text Back Translator (GT-BT) (Schmitt et al. 2020): This baseline adopts the same encoder and decoder in both directions. As a result, the graph should be transferred to a sequence that does not involve the prediction of edges. In this work, we reproduce its neural architecture and apply it to our tasks in the same setting as the original. 4) CycleGT (Guo et al. 2021): We reconstruct CycleGT with our proposed neural architectures so CycleGT can be regarded as a basic BT version. Furthermore, CycleCVAE is equivalent to BT+DC. Besides, we also show the results of graph generation and text generation in a supervised manner.

Results

Text-to-Graph Generation: Table 1 shows the performance of various models for all metrics. We can observe that unsupervised LMs (GPT-2 and BART) perform poorly on all three datasets. GT-BT performs well on CLUTRR but cannot achieve alignment tasks on the other two datasets. Practically, we adopt depth-first search (DFS) to simplify graphs to sequences. However, only on CLUTRR among the three can sequences effectively represent their corresponding graph because of simple logic and stable structures. Therefore, GT-BT can only capture such stable features on CLUTRR. According to F1 scores of the node and

Dataset	Model	Graph Generation		Text Generation				
		F1 (Node)	F1(Edge)	BLEU-1	BLEU-4	ROUGE-L	CIDEr	METEOR
Logic2Text (Symmetric)	Supervised*	99.7	99.8	96.3	91.7	96.5	8.16	66.0
	GPT-2(BT) (Radford et al. 2019)	34.8	-	26.0	13.8	25.9	0.1	19.1
	BART(BT) (Lewis et al. 2020)	64.9	-	63.8	50.5	59.2	3.15	41.4
	GT-BT (Schmitt et al. 2020)	0.2	-	14.4	0	13.9	0	15.9
	CycleGT (Guo et al. 2021)	97.2	99.7	89.1	84.3	89.3	7.31	52.9
	CycleCVAE (Guo et al. 2021)	96.9	99.2	**91.8**	87.2	**91.1**	**7.50**	**54.0**
	ConstrainedBT	**97.8**	**99.8**	91.7	**87.3**	90.9	7.49	**54.0**
LogicNLI (Asymmetric)	Supervised*	99.1	99.9	76.7	36.7	46.8	0.89	41.2
	GPT-2(BT) (Radford et al. 2019)	37.1	-	17.0	0	18.8	0	5.4
	BART(BT) (Lewis et al. 2020)	36.4	-	19.0	1.0	20.0	0	8.0
	GT-BT (Schmitt et al. 2020)	0	-	7.8	0	6.4	0	18.6
	CycleGT (Guo et al. 2021)	38.5	58.3	62.3	28.3	41.5	0.57	36.6
	CycleCVAE (Guo et al. 2021)	53.9	64.5	69.9	36.7	59.7	1.56	33.1
	ConstrainedBT	**65.9**	**74.0**	**85.3**	**56.8**	**76.6**	**2.47**	**40.2**
CLUTRR (Asymmetric)	Supervised*	96.0	98.0	38.3	17.7	35.9	0.17	25.4
	GPT-2(BT) (Radford et al. 2019)	8.1	-	2.9	0	3.0	0	9.7
	BART(BT) (Lewis et al. 2020)	28.4	-	19.1	0	19.9	0	8.0
	GT-BT (Schmitt et al. 2020)	42.3	-	34.3	19.6	44.6	0.78	27.5
	CycleGT (Guo et al. 2021)	24.4	68.1	15.9	4.3	13.9	0	14.4
	CycleCVAE (Guo et al. 2021)	44.6	68.9	36.6	17.2	38.2	0.69	25.2
	ConstrainedBT	**48.5**	**69.6**	**46.6**	**23.9**	**46.4**	**1.27**	**29.2**

Table 1: Results of ConstrainedBT and baselines. GT-BT does not predict edges according to our settings. All metrics except CIDEr are in percent (%). * means the supervised method.

the edge, we find that our method (ConstrainedBT) outperforms all unsupervised baselines on all three tasks. On LogicNLI and CLUTRR, constraints bring improvements, especially on node prediction with 12 points and 3.9 points, respectively. However, their performance is far from supervised learning. All models except GT-BT reach a comparable performance with supervised learning on Logic2Text.

Graph-to-Text Generation: Unsupervised LMs (GPT-2 and BART) are barely able to generate valid text. CycleGT, CycleCVAE, and ConstrainedBT perform similarly on Logic2Text. However, consistent with our intuition, our proposed constraints significantly improve the graph-to-text generation performance on LogicNLI and CLUTRR with asymmetric information. We can observe two phenomena from Table 1: 1) Constraints bring more significant improvements to text generation than graph generation; 2) ConstrainedBT even beats the supervised method on these two tasks with information asymmetry. Relatively, graph generation is much simpler than text generation, so models are likely to capture critical features of graph generation. Meanwhile, supervised learning regards bidirectional generation as two independent processes, making it perform almost perfectly on the graph generation but fail on text generation. Instead, ConstrainedBT provides intrinsic dependent constraints on the bidirectional generation that allows it to exchange information between two tasks, which provides additional information for graph-to-text generation.

Training Efficiency: Due to the similar performance on Logic2Text, we explore the training efficiency by loss curves of CycleGT, CycleCVAE, and ConstrainedBT. Although losses of all three models finally converge to a similar level, CycleGT and ConstrainedBT converge within 20 iterations, while CycleGT does not converge until 60 iterations. This evidence supports that our proposed constraints benefit BT from completing training more efficiently. Moreover, we can also observe from partial views that naive BT has a slightly better loss on graph generation, while BT with constraints has a slightly smaller loss on text generation.

Priority Analysis via Curriculum Learning

Curriculum learning is a strategy that defines a curriculum for model training. The definition of the curriculum is a sequence of training criteria $C = [c_1, c_2, \cdots, c_t]$, where c_i represents the i^{th} criterion with the design of the elements in training (Wang, Chen, and Zhu 2021). The original study constrains c_i as a criterion whose complexity is monotonically increasing (Bengio et al. 2009), meaning that the model learns from simple scenes to more complex ones. From the perspective of curriculum learning, we regard three constraints and their combinations as criteria. To obey the monotonically increasing principle, we stipulate that in a curriculum, if c_i includes a constraint, $c_j (j > i)$ will not be allowed to remove this constraint. We only discuss curricula with sequence lengths not greater than 2 in this work. Table 2 summarizes the results on CLUTRR.

Priority: Considering the priority of constraints (Red lines in Table 2), IC is the constraint with the highest priority, whose curricula result in the best performance on almost all text-graph alignment tasks. Essentially, the auxiliary task introduced by IC can stably capture the prior distribution

Criterion 1	Criterion 2	Graph Generation		Text Generation				
		F1 (Node)	F1(Edge)	BLEU-1	BLEU-4	ROUGE-L	CIDEr	METEOR
TC	***TC***	*36.7*	*68.9*	*23.9*	*10.4*	*22.5*	*0.28*	*15.4*
	TC+DC	37.7	69.1	**25.1**	**11.2**	**23.8**	**0.31**	16.2
	TC+IC	**38.6**	69.7	24.3	9.9	22.6	0.29	**16.8**
	TC+DC+IC	36.8	**69.9**	23.5	9.8	22.1	0.28	15.2
DC	***DC***	*44.6*	*68.9*	*36.6*	*17.2*	*38.2*	***0.69***	*25.2*
	DC+TC	45.9	68.9	36.6	**17.4**	**38.1**	0.67	25.4
	DC+IC	**46.9**	**69.1**	**37.2**	17.3	37.7	0.67	**25.6**
	DC+TC+IC	44.9	68.8	36.9	16.6	37.5	0.67	24.8
IC	***IC***	*43.8*	*69.1*	*43.7*	*21.2*	*43.3*	*0.95*	*27.8*
	IC+TC	45.5	69.4	46.5	23.3	45.4	1.09	29.0
	IC+DC	46.0	69.4	45.4	22.2	44.5	1.05	28.5
	IC+TC+DC	**48.5**	**69.6**	**46.6**	**23.9**	**46.4**	**1.27**	**29.2**

Table 2: Results of curriculum learning on CLUTRR. We categorize different settings into three groups based on Criterion 1. Italics represent experiments with only one criterion.

so that IC can provide the most accurate constraints on IV. Compared with IC, DC can provide similar improvements on noise-free tasks (Logic2Text and CLUTRR) but is not as good as IC on LogicNLI. Actually, DC adopts an approximate method to model the prior distribution that is not as accurate as of the IC's distribution. TC is the criterion with the lowest priority because it provides the weakest constraint without the true prior distribution $p(\boldsymbol{z})$ but promotes the two posterior distributions to be close to each other. Overall, the priority of the single constraint is IC $>$ DC $>$ TC.

Interaction: It is necessary to understand interactions among the three. Based on results of curriculum learning, DC and IC bring positive gains to TC as DC/IC further constrains the prior distribution to enhance TC. However, imposing DC and IC simultaneously in Criterion 2 does not necessarily lead to better performance because of their mutual interference. Considering DC, TC tends to bring negative gains, while IC benefits DC in most scenarios, which matches their priority. However, a surprising result is that the combination of TC and IC brings significant improvements to DC on LogicNLI. IC itself can provide a stable constraint so that TC and DC hardly impact IC negatively. Besides, the combination of TC and DC can bring stable positive gains to IC. Based on the analysis, we select the most stable curriculum of C=[IC, IC+TC+DC] as ConstrainedBT.

Downstream Applications

To further explore our proposed constraints' impacts on downstream applications, we adopt prompt learning to experiment on the generalization NLU tasks of LogicNLI and CLUTRR. Specifically, we use the latent variables out of the text encoder E_T as prompts and adopt P-Tuning v2 with reparameterization (Liu et al. 2021a) initialized by the prompts. We compare the randomly initialized prompt (Random), prompts out of CycleGT and CycleCVAE, and the prompt of ConstrainedBT. Results are shown in Table 3. On LogicNLI, if models achieve 51.3% accuracies, they predict all instances to the same label. Therefore, ConstrainedBT provides the only valid prompt that benefits the NLI task with an accuracy of 74.4%, while other prompts do not work. As for CLUTRR, accuracies are 56.9% (Random), 59.5% (CycleGT), 64.3% (CycleCVAE), and 66.6% (ConstrainedBT). These results prove that our proposed constraints can help capture critical information that benefits downstream tasks through prompt learning.

Dataset	Random	CycleGT	CycleCVAE	ConstrainedBT
LogicNLI	51.3	51.3	51.3	74.4
CLUTRR	56.9	59.5	64.3	66.6

Table 3: Results of prompt learning on downstream NLU tasks (Accuracy %).

Limitations of ConstrainedBT

There are three main limitations of ConstrainedBT. In theory, when revisiting Equation 7 and $q(\boldsymbol{z_t}|\boldsymbol{g'}(\boldsymbol{t}),\boldsymbol{z}) = q_{f_{\phi_3}}(E_G(\boldsymbol{g'}(\boldsymbol{t});\psi_1);\phi_3)$, we actually adopt an implicit assumption that $\boldsymbol{z} = f(\boldsymbol{z''}), \boldsymbol{z}_t = h(\boldsymbol{z''})$, where f and h can be seen as feature extractors. Nevertheless, $\boldsymbol{z''}$ should be the mixture of $\boldsymbol{z}$ and $\boldsymbol{z}_g$ which means that $\boldsymbol{z''}$ should not contain the information of $\boldsymbol{z}_t$ under ideally decoupling conditions. The same problem also occurs in Equation 8. However, this limitation does not affect the effectiveness of DC and IC because encoders in CVAE also receive feedback information from decoders during the back-propagation processes. Therefore, E_G (E_T) can encode the invisible information of z_t (z_g). In addition, interactions among the three constraints are not controllable under a fixed ratio given by λ. In practice, we cannot exhaust all possible curricula to find the best curriculum for three constraints.

Conclusions

To solve UTGA, we assume that asymmetric information is encoded in latent variables and propose three VAE-based constraints to introduce asymmetric information into the latent space and apply them to BT. Experiments on three tasks show that our proposed method can effectively improve the performance of UTGA with information asymmetry. Future works focus on: 1) designing more reasonable constraints to solve information asymmetry; 2) introducing explicit rules into UTGA.

Acknowledgements

This work was supported by the National Key Research and Development Program of China (2022YFC3340904), and the Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102).

References

Banerjee, S.; and Lavie, A. 2005. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *IEEvaluation@ACL*.

Bengio, Y.; Louradour, J.; Collobert, R.; and Weston, J. 2009. Curriculum learning. In *ICML*.

Cai, D.; and Lam, W. 2020. AMR Parsing via Graph-Sequence Iterative Inference. In *ACL*.

Cao, M. 2022. A Survey on Neural Abstractive Summarization Methods and Factual Consistency of Summarization. *CoRR*.

Chairatanakul, N.; Sriwatanasakdi, N.; Charoenphakdee, N.; Liu, X.; and Murata, T. 2021. Cross-lingual Transfer for Text Classification with Dictionary-based Heterogeneous Graph. In *EMNLP(Findings)*.

Chen, W.; Tian, J.; Fan, C.; He, H.; and Jin, Y. 2021a. Dependent Multi-Task Learning with Causal Intervention for Image Captioning. In *IJCAI*.

Chen, W.; Tian, J.; Li, Y.; He, H.; and Jin, Y. 2021b. De-Confounded Variational Encoder-Decoder for Logical Table-to-Text Generation. In *ACL*.

Chen, Z.; Chen, W.; Zha, H.; Zhou, X.; Zhang, Y.; Sundaresan, S.; and Wang, W. Y. 2020. Logic2Text: High-Fidelity Natural Language Generation from Logical Forms. In *EMNLP(Findings)*.

Cotterell, R.; and Kreutzer, J. 2018. Explaining and Generalizing Back-Translation through Wake-Sleep. *CoRR*.

Ferreira, T. C.; Gardent, C.; Ilinykh, N.; van der Lee, C.; Mille, S.; Moussallem, D.; and Shimorina, A. 2020. The 2020 Bilingual, Bi-Directional WebNLG+ Shared Task: Overview and Evaluation Results (WebNLG+ 2020). In *WEBNLG*.

Gai, Y.; Jain, P.; Zhang, W.; Gonzalez, J.; Song, D.; and Stoica, I. 2021. Grounded Graph Decoding improves Compositional Generalization in Question Answering. In *EMNLP(Findings)*.

Gardent, C.; Shimorina, A.; Narayan, S.; and Perez-Beltrachini, L. 2017. The WebNLG Challenge: Generating Text from RDF Data. In *INLG*.

Gershman, S.; and Goodman, N. D. 2014. Amortized Inference in Probabilistic Reasoning. In *CogSci*.

Guo, Q.; Jin, Z.; Qiu, X.; Zhang, W.; Wipf, D.; and Zhang, Z. 2020. CycleGT: Unsupervised Graph-to-Text and Text-to-Graph Generation via Cycle Training. In *INLG*.

Guo, Q.; Jin, Z.; Wang, Z.; Qiu, X.; Zhang, W.; Zhu, J.; Zhang, Z.; and Wipf, D. 2021. Fork or Fail: Cycle-Consistent Training with Many-to-One Mappings. In *AISTATS*.

Gururangan, S.; Marasovic, A.; Swayamdipta, S.; Lo, K.; Beltagy, I.; Downey, D.; and Smith, N. A. 2020. Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. In *ACL*.

He, J.; Wang, X.; Neubig, G.; and Berg-Kirkpatrick, T. 2020. A Probabilistic Formulation of Unsupervised Text Style Transfer. In *ICLR*.

Hoang, C. D. V.; Koehn, P.; Haffari, G.; and Cohn, T. 2018. Iterative Back-Translation for Neural Machine Translation. In *NMT Workshop of ACL*.

Houlsby, N.; Giurgiu, A.; Jastrzebski, S.; Morrone, B.; de Laroussilhe, Q.; Gesmundo, A.; Attariyan, M.; and Gelly, S. 2019. Parameter-Efficient Transfer Learning for NLP. In *ICML*.

Hoyle, A. M.; Marasovic, A.; and Smith, N. A. 2021. Promoting Graph Awareness in Linearized Graph-to-Text Generation. In *ACL(Findings)*.

Huang, Y.; Fang, M.; Cao, Y.; Wang, L.; and Liang, X. 2021. DAGN: Discourse-Aware Graph Network for Logical Reasoning. In *NAACL*.

Jain, U.; Zhang, Z.; and Schwing, A. G. 2017. Creativity: Generating Diverse Questions Using Variational Autoencoders. In *CVPR*.

Jin, Z.; Guo, Q.; Qiu, X.; and Zhang, Z. 2020. GenWiki: A Dataset of 1.3 Million Content-Sharing Text and Graphs for Unsupervised Graph-to-Text Generation. In *COLING*.

Ju, J.; Liu, M.; Koh, H. Y.; Jin, Y.; Du, L.; and Pan, S. 2021. Leveraging Information Bottleneck for Scientific Document Summarization. In *EMNLP(Findings)*.

Ke, P.; Ji, H.; Ran, Y.; Cui, X.; Wang, L.; Song, L.; Zhu, X.; and Huang, M. 2021. JointGT: Graph-Text Joint Representation Learning for Text Generation from Knowledge Graphs. In *ACL(Findings)*.

Kingma, D. P.; Mohamed, S.; Rezende, D. J.; and Welling, M. 2014. Semi-supervised Learning with Deep Generative Models. In *NeurIPS*.

Kingma, D. P.; and Welling, M. 2014. Auto-Encoding Variational Bayes. In *ICLR*.

Lai, H.; Toral, A.; and Nissim, M. 2021. Generic resources are what you need: Style transfer tasks without task-specific parallel training data. In *EMNLP*.

Lample, G.; Subramanian, S.; Smith, E. M.; Denoyer, L.; Ranzato, M.; and Boureau, Y. 2019. Multiple-Attribute Text Rewriting. In *ICLR*.

Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; and Zettlemoyer, L. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *ACL*.

Li, L.; Fan, K.; Li, H.; and Yuan, C. 2022a. Structural Supervision for Word Alignment and Machine Translation. In *ACL(Findings)*.

Li, X. L.; and Liang, P. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *ACL*.

Li, Y.; Tian, J.; Chen, W.; Fan, C.; He, H.; and Jin, Y. 2022b. To What Extent Do Natural Language Understanding Datasets Correlate to Logical Reasoning? A Method for Diagnosing Logical Reasoning. In *COLING*.

Lin, C.-Y. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*.

Lin, H.; and Ng, V. 2019. Abstractive Summarization: A Survey of the State of the Art. In *AAAI*.

Liu, X.; Ji, K.; Fu, Y.; Du, Z.; Yang, Z.; and Tang, J. 2021a. P-Tuning v2: Prompt Tuning Can Be Comparable to Finetuning Universally Across Scales and Tasks. *CoRR*.

Liu, X.; Zheng, Y.; Du, Z.; Ding, M.; Qian, Y.; Yang, Z.; and Tang, J. 2021b. GPT Understands, Too. *CoRR*.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*.

Liu, Z.; Wang, J.; and Li, Z. 2021. Topic-to-Essay Generation with Comprehensive Knowledge Enhancement. In *ECML/PKDD*.

Luo, F.; Li, P.; Zhou, J.; Yang, P.; Chang, B.; Sun, X.; and Sui, Z. 2019. A Dual Reinforcement Learning Framework for Unsupervised Text Style Transfer. In *IJCAI*.

Ma, Y.; Chen, Y.; Mao, X.; and Li, Q. 2021. Collaborative Learning of Bidirectional Decoders for Unsupervised Text Style Transfer. In *EMNLP*.

Nguyen, X.; Joty, S. R.; Nguyen, T.; Wu, K.; and Aw, A. T. 2021. Cross-model Back-translated Distillation for Unsupervised Machine Translation. In *ICML*.

Pandey, G.; and Dukkipati, A. 2016. Variational methods for Conditional Multimodal Learning: Generating Human Faces from Attributes. *CoRR*.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *ACL*.

Pham, H.; Wang, X.; Yang, Y.; and Neubig, G. 2021. Meta Back-Translation. In *ICLR*.

Prabhumoye, S.; Tsvetkov, Y.; Salakhutdinov, R.; and Black, A. W. 2018. Style Transfer Through Back-Translation. In *ACL*.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language Models are Unsupervised Multitask Learners. *CoRR*.

Ren, L.; Sun, C.; Ji, H.; and Hockenmaier, J. 2021. HySPA: Hybrid Span Generation for Scalable Text-to-Graph Extraction. In *ACL(Findings)*.

Saha, S.; Yadav, P.; Bauer, L.; and Bansal, M. 2021. ExplaGraphs: An Explanation Graph Generation Task for Structured Commonsense Reasoning. In *EMNLP*.

Schmitt, M.; Sharifzadeh, S.; Tresp, V.; and Schütze, H. 2020. An Unsupervised Joint System for Text Generation from Knowledge Graphs and Semantic Parsing. In *EMNLP*.

Sennrich, R.; Haddow, B.; and Birch, A. 2016. Improving Neural Machine Translation Models with Monolingual Data. In *ACL*.

Shen, T.; Lei, T.; Barzilay, R.; and Jaakkola, T. S. 2017. Style Transfer from Non-Parallel Text by Cross-Alignment. In *NeurIPS*.

Sinha, K.; Sodhani, S.; Dong, J.; Pineau, J.; and Hamilton, W. L. 2019. CLUTRR: A Diagnostic Benchmark for Inductive Reasoning from Text. In *EMNLP*.

Song, L.; Wang, A.; Su, J.; Zhang, Y.; Xu, K.; Ge, Y.; and Yu, D. 2020. Structural Information Preserving for Graph-to-Text Generation. In *ACL*.

Tian, J.; Li, Y.; Chen, W.; Xiao, L.; He, H.; and Jin, Y. 2021. Diagnosing the First-Order Logical Reasoning Ability Through LogicNLI. In *EMNLP*.

Tian, J.; Li, Y.; Chen, W.; Xiao, L.; He, H.; and Jin, Y. 2022. Weakly Supervised Neural Symbolic Learning for Cognitive Tasks. In *AAAI*.

Vedantam, R.; Zitnick, C. L.; and Parikh, D. 2015. CIDEr: Consensus-based image description evaluation. In *CVPR*.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *ICLR*.

Wang, R.; Tan, X.; Luo, R.; Qin, T.; and Liu, T. 2021. A Survey on Low-Resource Neural Machine Translation. In *IJCAI*.

Wang, X.; Chen, Y.; and Zhu, W. 2021. A Survey on Curriculum Learning. *TNNLS*.

West, P.; Holtzman, A.; Buys, J.; and Choi, Y. 2019. BottleSum: Unsupervised and Self-supervised Sentence Summarization using the Information Bottleneck Principle. In *EMNLP*.

Xiao, F.; Pang, L.; Lan, Y.; Wang, Y.; Shen, H.; and Cheng, X. 2021. Transductive Learning for Unsupervised Text Style Transfer. In *EMNLP*.

Yang, P.; Li, L.; Luo, F.; Liu, T.; and Sun, X. 2019. Enhancing Topic-to-Essay Generation with External Commonsense Knowledge. In *ACL*.

Yi, X.; Liu, Z.; Li, W.; and Sun, M. 2020. Text Style Transfer via Learning Style Instance Supported Latent Space. In *IJCAI*.

You, J.; Ying, R.; Ren, X.; Hamilton, W. L.; and Leskovec, J. 2018. GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models. In *ICML*.