Do It With Style: A Variational Transformer Framework for Unsupervised Data2text and Text2data

Alexandre Thomas
Mines ParisTech & Sorbonne University
M2 Internship at Criteo AI Lab
alexandre.thomas@mines-paristech.fr

December 14, 2024

Abstract

The problem of converting structured data to text, and vice-versa, appears in different contexts such as voice assistants (to explain some portion of a knowledge base to the user) or online product catalogs (to generate descriptions from product specifications). These tasks are typically framed as separate supervised tasks, such as conditional natural language generation (*data-to-text*) or entity and relation extraction (*text-to-data*) and are therefore limited to small datasets. Following recent works, we consider both tasks together and do cycle training to leverage *non-parallel data*, available in much larger quantities (unsupervised setting). We augment existing pretrained seq2seq language models with latent variables to model the style and format of text and structured data, and generalize to many-to-many relations between texts and data. Indeed, for a single knowledge graph, there are multiple valid sentences that can express its content. And conversely, in practical use cases, one might have multiple formats of structured data (e.g. coming from different data sources) that can express the content of a text. We consider two probabilistic graphical models, and derive specific cycle-training objectives, maximizing the likelihood and relying on backtranslation. We use the T5 pretrained model and we make experiments on the WebNLG 2020 dataset, with knowledge graphs as the structured data. Results are promising, with one model slightly improving the performance of its non-variational counterpart, while offering more diversity and being more controllable.

Contents

1 Introduction **3**

2 Related works **4**
2.1 Data-to-text . . . 4
2.2 Text-to-data . . . 5
2.3 Cycle-training for unsupervised neural machine translation and data-to-text . . . 5
2.4 VAE for NLP and data-to-text . . . 6
2.5 VAE and cycle-training . . . 6

3 Variational Framework **6**
3.1 Supervised, no latent variables . . . 7
3.2 Unsupervised, no latent variables . . . 7
3.3 Unsupervised, with latent variables . . . 8
3.3.1 One-to-many (Fork or Fail) . . . 8
3.3.2 Many-to-many with style latent variables s_x and s_y . . . 9
3.3.3 Many-to-many with a single z (entangled style and content) . . . 10

4 Experiments **12**
4.1 Implementation and training . . . 12
4.2 Results . . . 14

5 Conclusion and Future Works **15**

Appendices **20**

A Proofs and additional details **20**
A.1 Supervised log-likelihood . . . 20
A.2 Unsupervised log-likelihood . . . 20
A.3 Generic ELBO . . . 20
A.4 Deriving the $\mathcal{L}^{\text{auto-VAE}}$ objective . . . 21

B Additional results **21**

1 Introduction

In many domains, information can be stored either as plain text or as structured data, such as tables or knowledge graphs (KG), as illustrated in figure 1a. Conversion between structured and unstructured data is an important task in natural language processing. Natural language generation from structured data is typically known as *data-to-text*, and can be useful for instance for vocal assistants, or simply to make structured information more accessible. The *text-to-data* task, and more particularly *text-to-graph*, is closely related to (open) information extraction, and often considered as a joint entity and relation extraction task. Although other methods (e.g. based on planning, or templates for data-to-text) have traditionally been used, they have been outperformed by end-to-end, neural-network based approaches, even more so with pretrained language models, for both text-to-graph and graph-to-text tasks.

Graph:

```
[
  (Albennie Jones -> genre -> Jazz),
  (Jazz -> stylisticOrigin -> Blues),
  (Jazz -> musicFusionGenre -> Afrobeat)
]
```

Text 1: `"Albennie Jones performs jazz music which has its stylistic origins in the blues and is a music fusion genre of Afrobeat."`

Text 2: `"Albennie Jones is a jazz musician whose genre is Jazz. Jazz has its stylistic origins in the Blues and its music fusion genre is Afrobeat."`

(a) Example graph and its corresponding texts (in WebNLG, a single graph typically has multiple text realizations)

Graph: `"[HEAD] Albennie Jones [TYPE] genre [TAIL] Jazz [HEAD] Jazz [TYPE] stylisticOrigin [TAIL] Blues [HEAD] Jazz [TYPE] musicFusionGenre [TAIL] Afrobeat"`

(b) Representing the graph as a sequence. The `[HEAD]`, `[TYPE]`, `[TAIL]` tokens are special tokens that we add to the existing model vocabulary.

Figure 1: Example from the WebNLG dataset

These large models greatly benefit from large amounts of training data, but manually creating a dataset with pairs of KG and corresponding text is expensive. One of the reference data-to-text datasets for instance, WebNLG (Gardent et al. [2017]), only has 50k KG-text pairs, while large language models benefit from much larger datasets, depending on their size (Kaplan et al. [2020]). A number of recent works (Jin et al. [2020], Lebret et al. [2016], Chen et al.) have proposed building larger datasets automatically, e.g. using texts from Wikipedia and KG from DBPedia or Wikidata. However these datasets are quite noisy: compared to the parallel data of WebNLG (where the KG and the texts have exactly the same content), their text-graph pairs are loosely aligned or simply non-parallel.

To overcome these issues, we consider the unsupervised framework for conversion between text and structured data, using non-parallel data and relying on cycle-training with back-translation. We use a single sequence-to-sequence (seq2seq) architecture to learn both data-to-text and text-to-data tasks in a flexible way, and we leverage large pretrained models to improve performance. We combine this with the Variational Auto-Encoder (VAE) framework (Kingma and Welling [2013]), for multiple reasons. First, this allows us to explicitly model and better handle the potential many-to-many relations between text and structured data. Indeed, for a single table or knowledge graph, the content is uniquely defined, but there are multiple ways to verbalize it into a English sentence. In the other way, even though structured data is by definition better specified, in practical use cases there may still be some ambiguity between multiple data formats. For instance, the structured data may come from different sources, mixing both knowledge graphs and tables or using different ontologies and different attributes for the same concepts. Although we only experiment on WebNLG with KG as our structured data (due to time constraints), the symmetrical seq2seq architecture allows us to model the many-to-many relations with different formats easily. Second, the VAE model can learn better representations in a smooth latent space, in addition to providing an expressive generative model. Finally, the variational framework allows us to model specific attributes such as the content, the style (in the case of text data) or the format (in the case of structured data) as latent variables, for controllable generation.

Our work is inspired by Schmitt et al. [2020] and Guo et al. [2021], which also do unsupervised graph-to-text and text-to-graph conversion but differ on several points. The model proposed in Schmitt et al. [2020] is a seq2seq model as well and requires additional care to make back-translation successful, but assumes a one-to-one mapping without latent variables. The CycleCVAE model proposed by Guo et al. [2021] does add latent variable modelling to the existing CycleGT model (Guo et al. [2020b]), but uses a less flexible text-to-graph model which only learns

to do relation classification, and only considers the text style as a latent variable. We extend their work to a many-to-many relations setting.

Our contributions can be summarized as follows:

- We review existing frameworks for joint data-to-text and text-to-data conversion, supervised or unsupervised, with or without latent variables. We uncover the implicit probabilistic assumptions being made by state-of-the-art models, starting from the data likelihood.
- We propose two new architectures and training objectives in the more general many-to-many setting, starting from different probabilistic graphical models and combining back-translation with latent variables.
- We augment the seq2seq encoder-decoder architecture of transformers with latent variables to obtain a symmetric and expressive model. In particular, we adapt the T5 pretrained model (Raffel et al. [2019]) and design a T5-VAE.

A few words on the internship This work was done during my end-of-study internship at Criteo AI Lab, in the *Learning from Text and Structured Data* research team, under the supervision of Patrick Gallinari and Alberto Lumbreras (April to September 2021). Criteo is a global technology company that enables marketers and media owners to drive better commerce outcomes with trusted and impactful advertising. In the long term, developing a better understanding of both structured and unstructured data can be very helpful for the company, potentially leading to several applications (e.g. reducing annotation effort on product description datasets, transforming logs or other structured data into natural language explanations, etc).

In addition to a couple initial trainings and meetings regarding Criteo's business and structure, most of my time was dedicated to this work with the objective of exploring the field of data-to-text. During the first few months, I learned a lot about the current NLP field (transformers, pretrained models, metrics, tasks), about variational approaches (probabilistic graphical models, EM techniques, VAE), as well as more specific approaches to data-to-text and text-to-data. In parallel to literature review, I spent some time setting up a computing environment (using PyTorch, and remote GPU-equipped machines of Criteo), reproducing results of the CycleCVAE model (Guo et al. [2021]), and finally doing my own implementation of the various models. The last months were spent formalizing the theoretical framework and deriving the new models, implementing and experimenting with them, and finally writing this report.

I would like to thank Patrick Gallinari, Alberto Lumbreras as well as Mike Gartrell for their ideas and their regular feedback, and Nada Staouite for the fruitful collaboration.

2 Related works

2.1 Data-to-text

Natural language generation aims at generating text and can be conditioned on text input (e.g. in machine translation, text summarization, or dialogue systems), but also structured data (data-to-text). The goal might be to verbalize the content of a knowledge base or a table, to generate medical reports from RDF data (Bontcheva and Wilks [2004]), generate product descriptions from a set of attributes (Wang et al. [2017]) or generate weather forecasts from weather records (Mei et al. [2015]).

These tasks can involve two separate challenges: selecting the most relevant information (*what to say*) before producing fluent sentences (*how to say it*). We focus on the latter and, similarly to machine translation, we aim at keeping all the content when converting between text and data. This allows us to do cycle training and use non-parallel data, instead of the usual supervised setting that requires pairs of associated texts and graphs.

Historically, data-to-text systems have been explicitly modelling the successive steps required for text generation: content selection, document planning, lexicalisation, aggregation, surface realisation (Narayan and Gardent [2020]). While being more easily interpretable, these rule or template-based systems are often designed for a specific domain (e.g. sports reports, restaurants descriptions, etc) and cannot be easily adapted to other domains. Recently, neural approaches have been more common, either using end-to-end models with an encoder/decoder architecture (Wiseman et al. [2017], Gardent et al. [2017]) or using neural networks as part of a pipeline system (Moryossef et al. [2019], Puduppully et al. [2019], Ferreira et al. [2019]). In particular, using large pretrained language models have

increased performance significantly, either casting data-to-text as a simple seq2seq task (Kale [2020]) or combining it with a GNN-based planner for graph-to-text tasks (Ribeiro et al. [2020], Zhao et al. [2020], Guo et al. [2020a]).

2.2 Text-to-data

Converting unstructured text to structured data can be framed as (open) information extraction tasks, such as named entity recognition for product attribute extraction (Putthividhya and Hu [2011]) or, more commonly, joint entity and relation extraction. Given a pre-defined set of relation types, joint entity and relation extraction aims at extracting all relevant triplets in a sentence (see e.g. fig. 1a). It requires identifying both the relevant entities in the text (entity recognition) and the relations between them (relation classification). Since a knowledge graph can be represented as a set of triplets (e_1, r, e_2), it is effectively *text-to-graph* conversion. There has been less work converting text to tables, although some recent approaches aim at matching tables to knowledge graphs (Jiménez-Ruiz et al. [2020]), or generating tables from queries (Gupta and Berberich [2019]) in the information retrieval domain.

Text-to-graph models can be either *extractive* or *generative*. Extractive methods typically combine a traditional entity recognition model with a relation classification model (Bekoulis et al. [2018], Guo et al. [2020b]), or do both steps simultaneously (Eberts and Ulges [2019], Wei et al. [2019]). Generative approaches consider the graph as a sequence of triples and cast the problem as a seq2seq task. They typically combine an encoder with an auto-regressive decoder, using for instance transformer-based architectures (Schmitt et al. [2020], Ye et al. [2020a], Agarwal et al. [2020], Paolini et al. [2021]). Somewhat in between generative and extractive approaches, some works propose using a copy mechanism to output entities (Zeng et al. [2018], Zeng et al. [2020]), or generating the output triplets in parallel, with a non-autoregressive decoder (Sui et al. [2020]).

2.3 Cycle-training for unsupervised neural machine translation and data-to-text

Most previous works consider data-to-text and joint entity and relation extraction as separate tasks, in a supervised framework. This requires datasets with pairs of corresponding texts and graphs, which are expensive to collect. As a result, existing human-annotated datasets are relatively small (Jin et al. [2020]). For instance, WebNLG (Gardent et al. [2017]), one of the most popular graph-to-text datasets, has around 13k training samples. The ToTTo dataset Parikh et al. [2020], recently introduced for table-to-text generation, goes up to 120k training samples. On the other hand, unsupervised datasets with non-parallel graphs and texts are much easier to collect, and allow larger scales. The GenWiki dataset (Jin et al. [2020]) provides 1.3M training samples, with non-parallel data but a shared content distribution between graphs and texts, as well as a human-annotated test set.

This issue has actually been encountered in the field of machine translation, with a similar setting and significant work already done. Indeed, converting the same content between an unstructured format (text) and a structured format (graph) could be seen as a translation task between languages. Cycle training with back-translation has been shown to be very effective (Artetxe et al. [2017], Lample et al. [2017]). Two models (from source to target language, and vice-versa) are learned jointly on monolingual data, with one model generating synthetic training samples for the other, and vice-versa. In Lample et al. [2018], 3 core principles for successful unsupervised neural machine translation are identified:

(1) good initialization so that the models are already roughly aligned,

(2) a language modelling objective to train each model to generate correct sentences in its language,

(3) iterative backtranslation to learn the mapping between languages.

One of the risks is for both models to learn a trivial mapping and simply output the exact input sentence, which would give a perfect backtranslation score. Step (2) is particularly important here to avoid this collapse, using for instance a denoising auto-encoding objective. Factors for success or failure have been further studied in subsequent works (Edunov et al. [2018], Caswell et al. [2019]). Notably, Kim et al. [2020] argues that domain mismatch and linguistic dissimilarity between source and target languages affect performance significantly, and that even a small amount of parallel data can be beneficial (semi-supervised learning). These techniques have been successfully applied to other domains, for instance in Lachaux et al. [2020], to develop a translation model between programming languages (Java, C++, Python) using non-parallel data collected from GitHub.

In the field of data-to-text as well, unsupervised training using backtranslation has recently gained traction. Notably, Schmitt et al. [2020] adapts the principles and the noise functions from Lample et al. [2018] to text-graph conversion,

using a single seq2seq model to do both tasks (LSTM with attention and a copy mechanism). Proposed by Guo et al. [2020b], the CycleGT model has separate graph-to-text and text-to-graph components. The first one is based on pretrained seq2seq model T5 and takes a linearized graph as input, while the second one uses an off-the-shelf entity extraction model and simply learns a relation classification model to obtain the graph. Since this is a not a seq2seq framework anymore, collapsing on a trivial mapping becomes harder and it seems that step (2) is not necessary anymore.

2.4 VAE for NLP and data-to-text

The VAE is a powerful generative model with great representation learning capabilities. It proposes an efficient learning framework for large variational models with continuous latent variables, and has been used successfully in many domains. Bowman et al. [2015] was the first to apply it to NLP and successfully train a text VAE, using a RNN encoder/decoder model and doing KL annealing to avoid posterior collapse. A number of later works found that the latent space of text VAEs was not so smooth and controllable compared to image VAEs for instance, and proposed solutions. In Hu et al. [2017], controllable text generation is achieved by learning disentangled latent representations for sentence attributes (sentiment and tense), and enforcing their independence. In Xu et al. [2020], authors constrain the mean μ_z of posterior $q_\phi(z|x)$ to a simplex, in order to have a more compact latent space, and better interpolation.

The Transformer model (Vaswani et al. [2017]), with its powerful attention mechanism and encoder/decoder architecture, has also been combined to the VAE framework in NLP. For instance, Lin et al. [2020] tries to use a single global latent variable or a sequence of latent variables between the encoder and the decoder, for a more compact or more informative latent representation. Several works successfully took advantage of large pretrained language models to initialize the VAE encoder and decoder, such as Li et al. [2020] using BERT and GPT2, or Fang et al. [2021] using T5.

In the data-to-text domain, variational models with latent variables have also been used to increase the diversity and control the style of generated sentences. Wiseman et al. [2018] models templates as a sequence of discrete latent variables, and learns them using hidden semi-Markov models. In Guo et al. [2021], authors augment the unsupervised CycleGT model and propose a variational framework to explicitly model the one-to-many relation between graphs and texts. However, only the graph-to-text component is variational with a single latent variable for the text style, and the text-to-graph component still requires an external NER system. In Ye et al. [2020b], authors propose a variational model with disentangled content and style latent variables. Their semi-supervised algorithm is also inspired from backtranslation but it still requires some parallel data and it focuses on data-to-text only (no text-to-graph model).

2.5 VAE and cycle-training

A couple existing works in other domains are combining back-translation and cycle-training with the VAE framework. Notably, Tobing et al. [2019] adds a cycle-reconstruction term to the original VAE training objective, to improve VAE-based voice conversion models using non-parallel data. In the domain of computer vision, Jha et al. [2018] derives a cycle-consistent VAE framework to learn disentangled latent representations.

In NLP, an additional difficulty comes from the discrete nature of text data which, unlike images, prevents back-propagation through the generated data. In Cotterell and Kreutzer [2018], authors study back-translation for semi-supervised machine translation and provide an interpretation of it under the Expectation-Maximization framework, as an instance of the Wake-Sleep algorithm (Hinton et al. [1995]). However they do not consider latent variables to explicitly model content, style, or many-to-many relations.

3 Variational Framework

In this section, we present existing frameworks for learning jointly data-to-text and text-to-data models, and we derive two novel many-to-many models and training objectives, based on different probabilistic graphical models.

We denote our text data by x, our structured data by y (e.g. knowledge graphs), and $\mathcal{T}, \mathcal{G}$ the true distributions of the data. In the *supervised setting*, pairs of texts and structured data are available from parallel datasets (e.g. WebNLG, Gardent et al. [2017]): $(x, y) \sim \mathcal{T} \times \mathcal{G}$. In the *unsupervised setting* however, text and structured data datasets are non-parallel, i.e. for each sample we typically have no corresponding structured data or text sample

with the exact same content (e.g. GenWiki-FULL, Jin et al. [2020]). In other words, we sample $x \sim \mathcal{T}$ and $y \sim \mathcal{G}$ independently. However even though we don't have pairs at the sample level, in order for backtranslation to be successful it is important that *texts and structured data share the same content distribution*: same domain, shared vocabulary, similar entities and relations, etc. We note $\mathcal{D} = (\mathcal{T}, \mathcal{G}) = (x_1, \dots, x_n, y_1, \dots y_m)$ our combined dataset (either parallel or non-parallel), and we assume i.i.d. samples.

We parameterize our model by θ and we note $p_g = p_g(y \mid s; \theta)$ and $p_t = p_t(x \mid s; \theta)$ our respective generative models for structured data and text, taking any sequence s as an input[1]. Depending on the case, input sequence s can be either some text x, structured data y, their synthetic versions $\hat{x}$ and $\hat{y}$, or their noisy versions $\widetilde{x}$ and $\widetilde{y}$. Synthetic samples are generated from our models $\hat{x} \sim p_t(x \mid y)$ and $\hat{y} \sim p_g(y \mid x)$, and are used as inputs during backtranslation. Noisy samples $\widetilde{x} \sim C(x)$ and $\widetilde{y} \sim C(y)$ are obtained by applying some stochastic noise functions C to the original sequences x and y. Noisy samples are also used as inputs during training.

3.1 Supervised, no latent variables

The supervised setting can be a useful baseline to have an idea of attainable performance. It has been successfully applied on WebNLG for instance by Agarwal et al. [2020] using T5, or by Guo et al. [2020b].

In this setting, the usual negative log-likelihood objective corresponds to maximizing:

$$\mathcal{L}(\theta) = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{T}\times\mathcal{G}} \left[\log p_t(x \mid y) + \log p_g(y \mid x)\right] \tag{1}$$

As detailed in appendix A.1, this can actually be seen as maximizing the data likelihood $p_\theta(\mathcal{D})$.

Most of the works in data-to-text or joint entity and relation extraction domain simply optimize one model, using a seq2seq framework or not.

3.2 Unsupervised, no latent variables

When data pairs (x, y) are not available, we can use backtranslation to generate synthetic data $\hat{x} \sim p_t(x \mid y)$ and $\hat{y} \sim p_g(y \mid x)$ and train our models on synthetic pairs $(\hat{x}, y)$ and $(\hat{y}, x)$. The CycleGT (Guo et al. [2020b]) and GT-BT (Schmitt et al. [2020]) models fall in this setting.

Both models optimize a cycle-training objective, which can be seen as a non-parallel approximation of eq. 1:

$$\mathcal{L}^{\text{cycle}} = \mathop{\mathbb{E}}_{x\sim\mathcal{T},\hat{y}\sim p_g(y|x)} \log p_t\left(x \mid \hat{y}\right) + \mathop{\mathbb{E}}_{y\sim\mathcal{G},\hat{x}\sim p_t(x|y)} \log p_g\left(y \mid \hat{x}\right) \tag{2}$$

An important remark is that the two models are typically optimized independently in an iterative manner, rather than jointly. That is, when maximizing the graph-to-text likelihood $\mathbb{E}_{x\sim\mathcal{T},\hat{y}\sim p_g(y|x)}\, p_t(x \mid \hat{y})$, only p_t is optimized and the weights of model p_g are "freezed" for the generation of synthetic data $\hat{y}$.

While the CycleGT model only considers this cycle loss, the more flexible seq2seq framework requires an additional training objective. Indeed, a trivial solution of $\max_\theta \mathcal{L}^{\text{cycle}}$ is for p_t and p_g to learn the identity mapping, leading to $\hat{x} = y$, $\hat{y} = x$, and a perfect score. As introduced in section 2.3, previous works in unsupervised machine translation identified solutions to this issue. In addition to good model initialization (*step (1)*), and iterative backtranslation (*step (3)*), we need an additional "language modelling" objective to teach p_t to generate valid text, and p_g to generate valid structured data (*step (2)*). Following Lample et al. [2018], Schmitt et al. [2020] considers an additional denoising auto-encoding objective:

$$\mathcal{L}^{\text{auto,denoise}} = \mathop{\mathbb{E}}_{x\sim\mathcal{T},\widetilde{x}\sim C(x)} \log p_t(x \mid \widetilde{x}) + \mathop{\mathbb{E}}_{y\sim\mathcal{G},\widetilde{y}\sim C(y)} \log p_g(y \mid \widetilde{y}) \tag{3}$$

Where $\widetilde{x}$ and $\widetilde{y}$ are our noisy samples. This leads to maximizing the following objective:

$$\mathcal{L}(\theta) = \mathcal{L}^{\text{auto}}(\theta) + \mathcal{L}^{\text{cycle}}(\theta) \tag{4}$$

[1] We give more details regarding how to cast the problem into a seq2seq task with the T5 model in section 4.1.

3.3 Unsupervised, with latent variables

Introducing latent variables in conditional text generation allows us to explicitly model attributes such as content, text style (the tense, active or passive voice, formality, etc), or the format of the structured data. This can be useful for controllable text generation, or to increase diversity of generated samples.

3.3.1 One-to-many (Fork or Fail)

In the graph-text conversion domain, where the data format is uniquely defined but a single content can be verbalized into natural language in different ways, we have a one-to-many relation between y and x. This is modelled in the CycleCVAE model (Guo et al. [2021]) by adding a latent variable z representing all the information required to construct a text x, given a graph y and its content. This transforms the one-to-many mapping between y and x into a bijective mapping between (y, z) and x.

The CycleCVAE model keeps a non-variational text-to-graph model $p_g(y \mid x)$ but, as illustrated in figure 2 it introduces z in the graph-to-text model by transforming it into a Conditional VAE (CVAE) conditioned on graph y. It now combines a generative model $p_t(x \mid y, z)$ that depends on z, and an inference model $q_\phi(z \mid x)$.

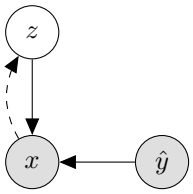


Figure 2: CycleCVAE graph-to-text probabilistic graphical model. Full arrows indicate the generative model, and dashed arrows the inference model.

Starting from eq. 2, latent variable z is introduced into the graph-to-text log-likelihood (left term):

$$\begin{aligned}
\log p_t(x \mid \hat{y}) &= \log \mathop{\mathbb{E}}_{z \sim p(z \mid \hat{y})} [p_t(x \mid \hat{y}, z)] \\
&\geq \mathop{\mathbb{E}}_{z \sim p(z \mid \hat{y})} [\log p_t(x \mid \hat{y}, z)] && \text{(using Jensen's inequality)} \\
&= \mathop{\mathbb{E}}_{z \sim q(z \mid x)} [\log p_t(x \mid \hat{y}, z)] - D_{KL}(q(z \mid x) \parallel p(z \mid \hat{y}))
\end{aligned}$$

which corresponds to the ELBO loss of a CVAE, and is valid for any variational distribution q.

For the other log-likelihood term, our text-to-graph model $p_g(y \mid x)$ should not depend on the style z of the input text $\hat{x}(y, z)$. In other words:

$$\forall z, \mathop{\mathbb{E}}_{\hat{x} \sim p_t(x \mid y, z)} \log p_g(y \mid \hat{x}) \approx \mathop{\mathbb{E}}_{z \sim p(z), \hat{x} \sim p_t(x \mid y, z)} \log p_g(y \mid \hat{x})$$

We obtain the final objective to maximize [2]:

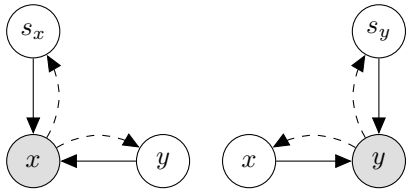
$$\begin{aligned}
\mathcal{L}^{\text{cycleCVAE}}(\theta, \phi) = &\mathop{\mathbb{E}}_{x \sim \mathcal{T}, \hat{y} \sim p_g(y \mid x)} \left[\mathop{\mathbb{E}}_{z \sim q_\phi(z \mid x)} [\log p_t(x \mid \hat{y}, z)] - D_{KL}(q_\phi(z \mid x) \parallel p(z)) \right] \\
&+ \mathop{\mathbb{E}}_{y \sim \mathcal{G}, z \sim p(z), \hat{x} \sim p_t(x \mid y, z)} [\log p_g(y \mid \hat{x})]
\end{aligned} \tag{5}$$

Here again, graph and text models are optimized in an iterative manner and are considered fixed when generating the synthetic data $\hat{x}$ and $\hat{y}$.

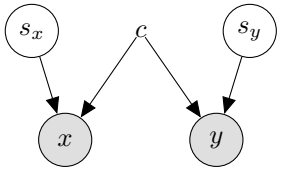
[2] Although this is the objective derived and specified in Guo et al. [2021], in their implementation available at `https://github.com/QipengGuo/CycleGT`, authors learn a conditional prior $p_\theta(z \mid y)$ instead of $p(z)$

3.3.2 Many-to-many with style latent variables s_x and s_y

Although we considered a unique format for structured data in previous section (knowledge graphs), in practical use cases structured data may come from multiple sources with various formats (graphs, logs, tables with different number of columns) or ontologies (e.g. different definitions for the list of relation types). A natural extension to the CycleCVAE framework would be to consider a latent variable s_y representing the structured data format, in addition to the previous latent variable s_x representing the text style. This way, we explicitly model the many-to-many relation between text and data.



(a) StyleVAE PGM. Plain arrows indicate the generative model, dotted arrows the inference model. This can be compared to figure 2.



(b) Alternative drawing, in the supervised setting. In our case however, we never observe x and y at the same time.

Figure 3: A semi stochastic latent variable model, with x: text, y: structured data, c: content (deterministic function of x or y), s_x: style of the text, s_y: style/format of the structured data

As illustrated in figure 3, we now consider:

- For the graph-to-text model, a decoder $p_t(x \mid y, s_x)$, an encoder $q_\phi(s_x \mid x)$ and a prior $p_\theta(s_x)$.
- For the text-to-graph model, a decoder $p_g(y \mid x, s_y)$, an encoder $q_\phi(s_y \mid y)$ and a prior $p_\theta(s_y)$.

Using this model and making a couple assumptions, it is actually possible to derive a tractable ELBO objective for cycle training. Assuming i.i.d. and non-parallel samples, with the same number of texts and graphs in our dataset, we have:

$$\max_\theta \log p_\theta(\mathcal{D}) = \max_\theta \mathop{\mathbb{E}}_{x\sim\mathcal{T}}[\log p_\theta(x)] + \mathop{\mathbb{E}}_{y\sim\mathcal{G}}[\log p_\theta(y)] \tag{6}$$

With proof in appendix A.2.

Let us now derive tractable lower bounds, and first consider the left likelihood term. Before making any assumption, we can always write the following ELBO (proof in A.3):

$$\log p_\theta(x) \geq \mathop{\mathbb{E}}_{\hat{y},s_x\sim q_\phi(y,s_x|x)}[\log p_\theta(x \mid \hat{y}, s_x)] - D_{KL}(q_\phi(y, s_x \mid x) \parallel p_\theta(y, s_x)) \tag{7}$$

With our assumptions regarding the graphical model (fig. 3a), we actually have the following:

- The prior on s_x can be simplified [3] to $p_\theta(s_x \mid y) = p_\theta(s_x)$. This implies that $p_\theta(s_x, y) = p_\theta(s_x)p_\theta(y)$.
- For the inference model q_ϕ, latent variables y and s_x are conditionally independent given x: $q_\phi(y, s_x \mid x) = q_\phi(y \mid x)q_\phi(s_x \mid x)$.

Eq. 7 becomes:

$$\begin{aligned}\log p_\theta(x) &\geq \mathop{\mathbb{E}}_{\hat{y}\sim q_\phi(y|x)}\left[\mathop{\mathbb{E}}_{s_x\sim q_\phi(s_x|x)}[\log p_\theta(x \mid \hat{y}, s_x)]\right] - \mathop{\mathbb{E}}_{y\sim q_\phi(y|x), s_x\sim q_\phi(s_x|x)}\left[\log \frac{q_\phi(y \mid x)q_\phi(s_x \mid x)}{p_\theta(y)p_\theta(s_x)}\right] \\ &= \mathop{\mathbb{E}}_{\hat{y}\sim q_\phi(y|x)}\left[\mathop{\mathbb{E}}_{s_x\sim q_\phi(s_x|x)}[\log p_\theta(x \mid \hat{y}, s_x)]\right] - D_{KL}(q_\phi(s_x \mid x) \parallel p_\theta(s_x)) - D_{KL}(q_\phi(y \mid x) \parallel p_\theta(y))\end{aligned} \tag{8}$$

And we can derive a symmetrical bound for $\log p_\theta(y)$.

[3] We could probably do without this assumption, but there is no need to have a conditional prior $p_\theta(s_x \mid y)$, especially since the inference model considers conditionally independent s_x and y

Two obstacles still prevent us from optimizing this lower bound. In order to compute the last KL divergence, we would first need to define a prior $p_\theta(y)$ in accordance with the symmetrical text-to-graph model, which already has a marginal posterior $p_\theta(y)$ (intractable). Second, y is a (sequence of) discrete latent variables, which prevents us from using the VAE framework: we cannot easily reparametrize the operation $\hat{y} \sim q_\phi(y \mid x)$, which corresponds to generation of synthetic training data. Yet, it closely resembles the cycle-training objective of eq. 19. Indeed, if we make the assumptions that our text-to-graph model is a good variational approximation $q_\phi(y \mid x) = p_g(y \mid x)$, and that *we do not need to optimize it in this step*, we find the same objective. The right KL divergence term does not need to be optimized anymore and we can pass the gradient through the first expectation (over the now fixed distribution $\hat{y} \sim p_g(y \mid x)$). This leads to the tractable cycle-training ELBO:

$$\begin{aligned}\mathcal{L}^{\text{cycle}}(\theta, \phi) = &\mathop{\mathbb{E}}_{x\sim\mathcal{T}, \hat{y}\sim p_g(y|x)} \left[\mathop{\mathbb{E}}_{s_x\sim q_\phi(s_x|x)} [\log p_t(x \mid \hat{y}, s_x)] - D_{KL}(q_\phi(s_x \mid x) \parallel p_\theta(s_x)) \right] \\ &+ \mathop{\mathbb{E}}_{y\sim\mathcal{G}, \hat{x}\sim p_t(x|y)} \left[\mathop{\mathbb{E}}_{s_y\sim q_\phi(s_y|y)} [\log p_g(y \mid \hat{x}, s_y)] - D_{KL}(q_\phi(s_y \mid y) \parallel p_\theta(s_y)) \right] \end{aligned} \tag{9}$$

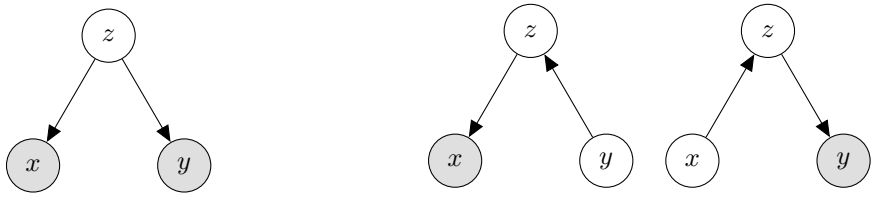
In other words, when optimizing p_t we assume that p_g is already good enough or is being trained in parallel. This assumption is rather strong, but can be justified in the back-translation framework since both models p_t and p_g are optimized to approximate the true conditional distributions $p(x \mid y)$ and $p(y \mid x)$, in alternate steps. This resonates with Cotterell and Kreutzer [2018], who interprets back-translation as a single iteration of the wake-sleep algorithm, where ones alternatively learns a forward model $p_\theta(y \mid x)$ and a variational approximation $q_\phi(x \mid y)$.

In addition to this objective and in order to avoid collapsing on the trivial identity mapping, we also consider the denoising auto-encoding objective of the non-variational unsupervised seq2seq baseline (eq. 3).

3.3.3 Many-to-many with a single z (entangled style and content)

The previous setting, considering content as a deterministic function of x or y, is useful if we are simply interested in controlling the style and format of the generated text and data. It also allows us to do only minor modifications to the non-variational transformer pretrained models. However we may also be interested in having a structured latent space for the content, for generative modelling or simply for learning smoother representations (e.g. we might want to interpolate between samples). We could extend the previous model with a content latent variable c, and make sure that s_x, s_y and c represent what they should. Due to time constraints however, we leave disentanglement for future works and we consider a first model with a single latent variable z for both style and content.

The probabilistic model is illustrated in figure 4. It can still handle many-to-many mappings, and it relies on a shared latent space for graphs and texts. Unlike the base VAE, which models data x conditioned on a latent variable z, we want to model data x and y (same content, but different formats), conditioned on z (fig. 4a).



(a) Probabilistic graphical model of the CycleVAE **(b)** An alternative, hierarchical latent variable model

Figure 4: x: text, y: graph, z: latent variable representing style and content

As seen in section 4.1, this framework works nicely with the encoder-decoder architecture of many seq2seq models. We can indeed modify the transformer encoder into a VAE encoder, and the transformer decoder into a VAE decoder. We now consider:

- For the graph-to-text model, a decoder $p_t(x \mid z)$, an encoder $q_\phi(z \mid y)$ and a prior $p_\theta(z)$. We obtain the graph-to-text model by combining encoder and decoder: $p_t(x \mid y) = p_t(x \mid z)$, with $z \sim q_\phi(z \mid y)$ or $z = \mu_z(y; \phi)$.
- For the text-to-graph model, a decoder $p_g(y \mid z)$, and encoder $q_\phi(z \mid x)$ and the same prior $p_\theta(z)$ (since we want a shared latent space). We obtain the text-to-graph model $p_g(y \mid x)$ similarly.

By swapping the encoders, we can also obtain text-to-text or graph-to-graph models which correspond to regular VAEs.

Again, as detailed in appendix A.2, we have in the unsupervised setting:

$$\max_{\theta} p_\theta(\mathcal{D}) = \max_{\theta} \mathop{\mathbb{E}}_{x\sim\mathcal{T}}[\log p_\theta(x)] + \mathop{\mathbb{E}}_{y\sim\mathcal{G}}[\log p_\theta(y)] \tag{10}$$

And, without any assumptions on x, y and z, we have (appendix A.3):

$$\log p_\theta(x) \geq \mathop{\mathbb{E}}_{\hat{y},z\sim q_\phi(y,z|x)}[\log p_\theta(x \mid \hat{y}, z)] - D_{KL}(q_\phi(y, z \mid x) \parallel p_\theta(y, z)) \tag{11}$$

Depending on our assumptions, we can actually derive different lower bounds $\mathcal{L}^{\text{auto-VAE}}$ and $\mathcal{L}^{\text{cycle}}$, which can be seen as variational equivalents of the unsupervised training objectives 3 and 2.

In particular, we can consider different probabilistic graphical models, both for inference and generation (figure 4a or figure 4b). In all cases, we have $p_\theta(x \mid z, y) = p_\theta(x \mid z)$ (since z contains all the information). Therefore the objective becomes:

$$\log p_\theta(x) \geq \mathop{\mathbb{E}}_{\hat{y},z\sim q_\phi(y,z|x)}[\log p_\theta(x \mid z)] - D_{KL}(q_\phi(y, z \mid x) \parallel p_\theta(y, z)) \tag{12}$$

We now have to choose how to express $q_\phi(y, z \mid x)$ and $p_\theta(y, z)$ using our encoder and decoder models.

The $\mathcal{L}^{\text{auto-VAE}}$ objective The $\mathcal{L}^{\text{auto-VAE}}$ objective can be obtained starting directly from eq. 10, by simply considering the left part of the probabilistic graphical models (only x and z) and deriving the usual VAE ELBO:

$$\log p_\theta(x) \geq \mathop{\mathbb{E}}_{z\sim q_\phi(z|x)}[\log p_\theta(x \mid z)] - D_{KL}(q_\phi(z \mid x) \parallel p_\theta(z)) \tag{13}$$

Doing the same for graphs y, we obtain the following objective:

$$\begin{aligned}\mathcal{L}^{\text{auto-VAE}}(\theta, \phi) = &\mathop{\mathbb{E}}_{x\sim\mathcal{T}}\left[\mathop{\mathbb{E}}_{z\sim q_\phi(z|x)}[\log p_t(x \mid z)] - D_{KL}(q_\phi(z \mid x) \parallel p_\theta(z))\right] \\ &+ \mathop{\mathbb{E}}_{y\sim\mathcal{G}}\left[\mathop{\mathbb{E}}_{z\sim q_\phi(z|y)}[\log p_g(y \mid z)] - D_{KL}(q_\phi(z \mid y) \parallel p_\theta(z))\right]\end{aligned} \tag{14}$$

As detailed in appendix A.4 this objective can also be derived starting from 12 by writing $q_\phi(y, z \mid x) = q_\phi(y \mid z, x)q_\phi(z \mid x)$ *and assuming that* $q_\phi(y \mid z, x) = q_\phi(y \mid z)$.

Eq. 14 is a valid lower bound on the data and it will teach the model to generate correct texts or graphs, but it cannot lead to successful back-translation alone. Indeed, even though we are using the same prior $p_\theta(z)$ for both models, we are not enforcing a shared latent space for the representations of x and y. Since we are basically learning two separate VAEs, there is no reason for text and graph encoders $q_\phi(z \mid x)$ and $q_\phi(z \mid y)$ to map a pair (x, y) sharing the same content to the same representation z. This motivates the cycle-training objective.

The $\mathcal{L}^{\text{cycle-VAE}}$ objective If we want to force the model to actually use synthetic samples $\hat{y} \sim p_g(y \mid x)$ and do back-translation, we can write instead $q_\phi(y, z \mid x) = q_\phi(z \mid x, y)q_\phi(y \mid x)$. A *first assumption* will be to only consider texts x to obtain z:

$$q_\phi(z \mid x, y) = q_\phi(z \mid y) \tag{15}$$

This should not be a problem since, when optimizing $\log p_\theta(y)$, we will make the symmetrical assumption.

The objective of eq. 12 becomes:

$$\begin{aligned}\log p_\theta(x) &\geq \mathop{\mathbb{E}}_{\hat{y}\sim q_\phi(y|x)}\left[\mathop{\mathbb{E}}_{z\sim q_\phi(z|\hat{y})}[\log p_\theta(x \mid z)]\right] - D_{KL}(q_\phi(y, z \mid x) \parallel p_\theta(y, z)) \\ &= \mathop{\mathbb{E}}_{\hat{y}\sim q_\phi(y|x)}\left[\mathop{\mathbb{E}}_{z\sim q_\phi(z|\hat{y})}[\log p_\theta(x \mid z)] - \mathop{\mathbb{E}}_{z\sim q_\phi(z|\hat{y})}\left[\log \frac{q_\phi(z \mid \hat{y})q_\phi(\hat{y} \mid x)}{p_\theta(\hat{y}, z)}\right]\right] \\ &= \mathop{\mathbb{E}}_{\hat{y}\sim q_\phi(y|x)}\left[\mathop{\mathbb{E}}_{z\sim q_\phi(z|\hat{y})}[\log p_\theta(x \mid z)] - D_{KL}(q_\phi(z \mid \hat{y}) \parallel p_\theta(\hat{y}, z)) + \log q_\phi(\hat{y} \mid x)\right]\end{aligned} \tag{16}$$

Eq. 16 is interesting and actually introduces back-translation, since it uses a synthetic sample $\hat{y}$ to reconstruct the original x. However, we encounter the same issue as in section 3.3.2: the discrete nature of y makes the objective intractable, due to the expectation over $\hat{y} \sim q_\phi(y \mid x)$. By making the *same assumption* that we do not optimize $q_\phi(y \mid x) = p_g(y \mid x)$ in this step, and writing $p_\theta(y, z) = p_\theta(y \mid z)p_\theta(z)$ (fig. 4a), we obtain this lower bound:

$$\log p_\theta(x) \geq \mathop{\mathbb{E}}_{\hat{y}\sim q(y|x)} \left[\mathop{\mathbb{E}}_{z\sim q_\phi(z|\hat{y})} [\log p_\theta(x \mid z) + \log p_\theta(\hat{y} \mid z)] - D_{KL}(q_\phi(z \mid \hat{y}) \parallel p_\theta(z)) \right] \tag{17}$$

Where all terms are well known and tractable. The total resulting objective, combining text and graph losses, is:

$$\begin{aligned} \mathcal{L}^{\text{cycle-VAE-joint}}(\theta, \phi) = & \mathop{\mathbb{E}}_{x\sim\mathcal{T}, \hat{y}\sim p_g(y|x)} \left[\mathop{\mathbb{E}}_{z\sim q_\phi(z|\hat{y})} [\log p_t(x \mid z) + \log p_g(\hat{y} \mid z)] - D_{KL}(q_\phi(z \mid \hat{y}) \parallel p_\theta(z)) \right] \\ & + \mathop{\mathbb{E}}_{y\sim\mathcal{G}, \hat{x}\sim p_t(x|y)} \left[\mathop{\mathbb{E}}_{z\sim q_\phi(z|\hat{x})} [\log p_g(y \mid z) + \log p_t(\hat{x} \mid z)] - D_{KL}(q_\phi(z \mid \hat{x}) \parallel p_\theta(z)) \right] \end{aligned} \tag{18}$$

Interestingly, eq. 18 does enforce a shared latent representation for texts and graphs, not only through the back-translation mechanism, but also through the joint reconstruction term $\log p_t(x \mid z) + \log p_g(\hat{y} \mid z)$, from the same latent variable z.

In practice however, synthetic targets $\hat{y}$ are noisy at the beginning of training, which made training less stable in our preliminary experiments. In a couple existing works using back-translation in the voice conversion domain (Tobing et al. [2019], Lee et al. [2019]), the terms $\log p_g(\hat{y} \mid z)$ and $\log p_t(\hat{x} \mid z)$ are removed and a slightly different objective is considered:

$$\begin{aligned} \mathcal{L}^{\text{cycle-VAE-single}}(\theta, \phi) = & \mathop{\mathbb{E}}_{x\sim\mathcal{T}, \hat{y}\sim p_g(y|x)} \left[\mathop{\mathbb{E}}_{z\sim q_\phi(z|\hat{y})} \log p_t(x \mid z) - D_{KL}(q_\phi(z \mid \hat{y}) \parallel p_\theta(z)) \right] \\ & + \mathop{\mathbb{E}}_{y\sim\mathcal{G}, \hat{x}\sim p_t(x|y)} \left[\mathop{\mathbb{E}}_{z\sim q_\phi(z|\hat{x})} \log p_g(y \mid z) - D_{KL}(q_\phi(z \mid \hat{x}) \parallel p_\theta(z)) \right] \end{aligned} \tag{19}$$

While eq. 19 leads to more stable learning and performs similarly to the non-variational unsupervised model, the objective of eq. 18 should in principle enforce a tighter correspondence in the latent space between similar graphs and texts. We leave for future works a more thorough experimental comparison of these two objectives.

4 Experiments

4.1 Implementation and training

Dataset Although our framework easily allows multiple types of structured data together (e.g. graphs, tables formatted differently), we only experiment on graph-to-text and text-to-graph tasks, with the WebNLG 2020 dataset[4]. As illustrated in figure 1, we cast both tasks into a seq2seq framework by linearizing the knowledge graphs in an arbitrary order. Like Schmitt et al. [2020] and Guo et al. [2020b], we break up the pairs and consider texts and graphs independently to simulate the unsupervised setting. We also consider the supervised setting as a baseline, where we keep the data pairs.

Adapting T5 to text-graph conversion We build our model around T5 (Raffel et al. [2019]) whose architecture is close to the original Transformer architecture (Vaswani et al. [2017], [5]). It combines an encoder and a decoder component, which stack several "blocks" of self-attention layers followed by feed-forward networks. The output of the encoder is given as input to the decoder, which also has standard attention layers attending to the encoder outputs. The T5 model has been very successful in transfer learning and multi-task learning, including graph-to-text and text-to-graph in multiple languages (Agarwal et al. [2020]). We follow this practice and we finetune a single T5-based model to do generate both texts (with $p_t(x|\cdot)$) and graphs (with $p_g(y|\cdot)$). Note that, by using

[4] Data is available at `https://gitlab.com/shimorina/webnlg-dataset/-/tree/master/release_v3.0`

[5] `http://jalammar.github.io/illustrated-transformer/`

the same pretrained model for both ways, we likely encode graphs and texts with the same content to a similar latent representation, and we satisfy the initialization condition for successful backtranslation (step (1)). We use the `t5-small` version.

We specify the target format by adding a prefix to the input sequence given to the encoder: either `"Generate text:"` or `"Generate graph:"` [6]. After successful training, the encoder outputs should contain at least the content of the input text or graph.

Non-variational baseline We implement a baseline without any latent variables in this setting, that we can train both in supervised and unsupervised settings, on WebNLG. Given the seq2seq framework, we add a denoising auto-encoding objective, and we use all noise functions from Schmitt et al. [2020][7], randomly sampling one of them at each training step. That is we either swap, drop, repeat, blank some tokens, or generate a noisy translation with a simple rule-based algorithm.

Many-to-many with s_x and s_y (deterministic content) In this setting, we only need to add style latent variables, without modifying the way our model represents content. This allows us to do only minor modifications to the original transformer architecture. We need, for our variational model:

- *Encoders $q_\phi(s_x|x)$ and $q_\phi(s_y|y)$.* We add a special token `[STYLE]` to the input sequence, that the encoder will process alongside the input text x or graph y. We take its hidden representation in encoder outputs, and we add text- or graph-specific linear layers to obtain the parameters of our variational distributions $q_\phi(s_x|x) = \mathcal{N}(s_x; \mu_{s_x}, \sigma_{s_x})$ and $q_\phi(s_y|y) = \mathcal{N}(s_y; \mu_{s_y}, \sigma_{s_y})$. We use the same dimension than encoder outputs (`model_dim`) for our style latent variables.
- *Decoders $p_t(x|s_x, y)$ and $p_g(y|s_y, x)$.* In the case of graph-to-text conversion, we give the input graph y to the encoder as usual, in order to encode its content. Before giving the sequence of encoder outputs to the decoder however, we concatenate the style latent variable s_x to the sequence. For text-to-graph, we do the same but with input text x and latent variable s_y.
- *Priors $p_\theta(s_x)$ and $p_\theta(s_y)$.* We set them to the standard normal distribution $\mathcal{N}(0, I)$.

Many-to-many with a single z (entangled style and content) In this setting, we consider content as a latent variable as well, therefore we choose to map the sequence of encoder outputs $h = (h_1, \dots, h_T)$ to a sequence of latent variables $z = (z_1, \dots, z_T)$, representing both content and style. We model z as a full sequence with the same dimension than h to have a more information-rich representation, compared to a single compact latent variable like in Li et al. [2020]. Our variational model is made of:

- *Encoders $q_\phi(z|x)$ and $q_\phi(z|y)$.* We consider independent normal distributions for each latent variable z_t of the sequence: $q_\phi(z|\cdot) = \mathcal{N}(z_t; \mu_{z_t}, \sigma_{z_t})$. We obtain parameters μ_{z_t} and σ_{z_t} by simply applying a shared linear transformation to the encoder output h_t.
- *Decoders $p_t(x|z)$ and $p_g(y|z)$.* Given our sequence $z = (z_1, \dots, z_T)$ (e.g. sampled from posterior or prior), we use the decoder as usual, simply feeding it z instead of encoder outputs h.
- *Prior $p_\theta(z)$.* We consider each latent variable z_t independently, and choose the standard normal distribution: $z_t \sim \mathcal{N}(0, I)$. We leave the study of more complex priors for future works.

Solving the KL vanishing issue A common issue encountered when training VAE models is *KL vanishing* (or KL collapse). At the beginning of training, an easy solution to decrease the ELBO loss is simply to match the variational posterior to the prior ($\forall x, q_\phi(z|x) = p_\theta(z)$) to obtain a zero KL divergence term. This leads to a local minima where the encoder becomes uninformative (giving the same latent representation no matter the input), and the decoder tries to reconstruct the data without using z. This can happen easily when the decoder is powerful enough to do well without the latent variable, especially in NLP with large generative models and auto-regressive decoding (Fu et al. [2019]). Several works have proposed solutions to this issue, but we found the MMD-VAE (Zhao et al. [2017]) particularly effective and simple to implement. In practice, this means replacing the exact KL

[6] We also tried defining two special tokens that we could give to the decoder as a start token, to indicate the target format (instead of the default `<pad>` token). This would remove the need to alter the encoder input sequence, but it made training too unstable in our preliminary experiments.

[7] Their original implementation is available at `https://github.com/mnschmit/unsupervised-graph-text-conversion`

divergence term $D_{KL}(q_\phi(z|x)||p_\theta(z))$ with a sample estimate of the Maximum-Mean Discrepancy (MMD) measure between the aggregated posterior $q_\phi(z) = \mathbb{E}_{x\sim p_\mathcal{D}(x)}[q_\phi(z|x)]$ and the prior $p_\theta(z)$:

$$D_{MMD}(q_\phi(z)||p_\theta(z)) = \mathop{\mathbb{E}}_{z,z'\sim q_\phi(z)}[k(z,z')] + \mathop{\mathbb{E}}_{z,z'\sim p_\theta(z)}[k(z,z')] - 2\mathop{\mathbb{E}}_{z\sim q_\phi(z),z'\sim p_\theta(z')}[k(z,z')]$$

We used a coefficient $\lambda = 10$ in front of the MMD regularization term. We also experimented with the β-VAE and cyclical schedule proposed in Fu et al. [2019] and used e.g. in Li et al. [2020], but preliminary experiments were not as successful.

Training We train our models for 10 epochs, using 2 Tesla V100 GPUs with 16GB memory and an effective batch size of $2 \times 20 = 40$ for the `t5-small` model. With this setting, unsupervised training takes approximately 9 hours. For the learning rate, we use a linear schedule that decreases to 0 over training, starting from $10e^{-4}$, and we clip the gradients norm to 1.0. Finally, we use top-k sampling to generate the synthetic graphs and texts for most models in the unsupervised setting, following Edunov et al. [2018] who argues that sampling improves backtranslation in unsupervised machine translation. We use greedy decoding for the FullVAE model only. For evaluation, we use beam search decoding.

4.2 Results

Baseline models We compare our models against current state of the art supervised and unsupervised models on WebNLG 2020 (English only), on both graph-to-text and text-to-graph tasks.

In the supervised setting, the P2 model proposed by Guo et al. [2020a] achieved best results in the WebNLG 2020 challenge (Ferreira et al. [2020]) in both text-to-graph and graph-to-text tasks. The graph-to-text model combines a R-GCN planner trained to linearize graphs in the desired order, with `t5-large` pretrained model (without using additional pretraining data). The text-to-graph model however makes use of external data: it queries relations between entities from the DBpedia database, after identifying entities in the text. Since it is the most similar to our seq2seq T5-based approach, we also include results of the bt5 model (Agarwal et al. [2020]).

In the unsupervised setting, we can only compare to CycleGT (Guo et al. [2020b]) on WebNLG 2020. The graph-to-text model simply uses `t5-base`, while the text-to-graph model combines an external NER tool with a BiLSTM-based relation classification module.

We implement our own non-variational baseline by using T5 for both graph-to-text and text-to-graph tasks, in our seq2seq framework with graphs linearized in an arbitrary order. We train a supervised version as well as an unsupervised version (by breaking up the pairs), as detailed in section 4.1.

StyleVAE refers to our proposed model in section 3.3.2 (using only style latent variables s_x and s_y), while *FullVAE* refers to section 3.3.3 (using a single latent variable z for both content and style). For the FullVAE model, we compare the *joint loss* (eq. 18, the actual lower bound including the joint reconstruction terms) with the *single loss* (eq. 19, the empirical objective without the synthetic data reconstruction term).

Metrics We compute standard text generation and joint entity and relation extraction metrics on the test sets. The WebNLG 2020 test set actually has 3 subsets: *seen categories* for data in categories seen during training, *unseen entities* for data with entities absent from the training set, and *unseen categories* for data in categories not in the training set.

For graph-to-text, we compute automated metrics BLEU-4 (Papineni et al. [2002]), METEOR (Lavie and Agarwal [2007]) and BERTScore (Zhang et al. [2019]), using official evaluation scripts[8]. For text-to-graph, we compute F1, Precision and Recall for relations (*exact matching of triples*) and for entities. We also compute the graph accuracy (to measure exact matching of all the graph triples), and format error (fraction of generated sequences that could not be parsed successfully into a (e_1, rel, e_2) triple).

[8] `https://github.com/WebNLG/GenerationEval`

Interpretation Results on the full test set are presented in tables 1 and 2, and results on *seen categories*, *unseen categories* and *unseen entities* are in tables 3 and 4 (appendix).

Despite being preliminary results (e.g. we used `t5-small`, while P2 uses `t5-large`), these results look promising. As could be expected, the supervised models always perform better than their unsupervised counterparts, and can be useful to give performance upper bounds.

In the unsupervised setting, the seq2seq framework seems to perform less well in the text-to-graph task (cf. our unsupervised baseline, compared to CycleGT), but is especially effective in graph-to-text. Most remarkably, the StyleVAE (close to the unsupervised baseline, but augmented with style latent variables) performs similarly, or slightly better than its non-variational counterpart (the unsupervised baseline). This concurs with experimental results from Guo et al. [2021], where the CycleCVAE has slightly better performance than CycleGT. It should be noted however that the main motivation behind the variational models is that they are more controllable and can produce more diverse output. Finally, the FullVAE performance is on par or a bit lower than StyleVAE (especially the *joint* loss), and should be studied further.

		BLEU	METEOR	BERTScore		
				F1	P	R
	P2 (supervised)	54.0	0.417	0.958	0.960	0.957
	bt5 (supervised)	51.7	0.411	0.954	0.955	0.954
	CycleGT (unsupervised)	44.6	0.387	0.948	0.949	0.949
	Baseline (supervised)	49.31	0.391	0.952	0.956	0.950
	Baseline (unsupervised)	45.58	0.372	0.947	0.951	0.944
Ours	StyleVAE	46.27	0.379	0.948	0.951	0.946
	FullVAE (joint loss)	40.430	0.346	0.938	0.946	0.932
	FullVAE (single loss)	45.760	0.382	0.946	0.949	0.945

Table 1: Data-to-Text results, full test set

		Relations			Entities				
		F1	P	R	F1	P	R	Graph Acc.	Format err.
	P2 (supervised)	0.689	0.689	0.690	-	-	-	-	-
	bt5 (supervised)	0.682	0.670	0.701	-	-	-	-	-
	CycleGT (unsupervised)	0.342	0.338	0.349	-	-	-	-	-
	Baseline (supervised)	0.414	0.427	0.403	0.70	0.71	0.69	0.19	0.037
	Baseline (unsupervised)	0.271	0.294	0.252	0.584	0.611	0.559	0.093	0.039
Ours	StyleVAE	0.273	0.294	0.255	0.588	0.613	0.565	0.099	0.044
	FullVAE (joint loss)	0.176	0.182	0.170	0.499	0.503	0.495	0.037	0.363
	FullVAE (single loss)	0.245	0.259	0.232	0.568	0.581	0.555	0.077	0.179

Table 2: Text-to-Graph results, full test set

5 Conclusion and Future Works

After reviewing existing literature on data-to-text, text-to-data, unsupervised machine translation using cycle-training and back-translation, and some variational extensions, we have focused on the different training frameworks. In particular, we reviewed existing training objectives for joint data-to-text and text-to-data conversion, supervised or unsupervised, with or without latent variables. Focusing on the unsupervised setting, we showed how back-translation can be derived in a variational framework, starting from the data likelihood. Motivated by the fact that the same content can be verbalized in multiple ways into natural language (due to the text style), or into structured data (to support different formats), we derive two new models handling the many-to-many relations with latent variables. We validate the approach experimentally on WebNLG 2020.

With this exploratory work in the data-to-text field, we left several doors open for future works. First, the analysis of our proposed models can be pushed further by experimenting on data with actual many-to-many relations. This requires building a new dataset (that could be non-parallel), e.g. by assembling different formats from existing datasets on the same domains. Experimenting on the larger dataset GenWiki would also be interesting, as well as

studying more in-depth the structure of our latent representation (e.g. sampling, interpolating, measuring diversity, etc).

Second, regarding the latent variable model, a natural extension would be to consider both content and style as separate latent variables, and enforce disentanglement (see e.g. Ye et al. [2020b]).

Finally, regarding the architecture, one could try using an extractive approach for the text-to-graph component. Although less flexible, it would also likely suffer less from the identity collapse issue of seq2seq models in back-translation. If keeping the seq2seq framework instead, one could adapt the default positional encoding of language models to be permutation invariant regarding the graph input triples (although this is not trivial to do with the relative positional embeddings of the T5 model).

References

O. Agarwal, M. Kale, H. Ge, S. Shakeri, and R. Al-Rfou. Machine translation aided bilingual data-to-text generation and semantic parsing. In *Proceedings of the 3rd WebNLG Workshop on Natural Language Generation from the Semantic Web (WebNLG+ 2020), Dublin, Ireland (Virtual). Association for Computational Linguistics*, 2020.

M. Artetxe, G. Labaka, E. Agirre, and K. Cho. Unsupervised neural machine translation. *arXiv preprint arXiv:1710.11041*, 2017.

G. Bekoulis, J. Deleu, T. Demeester, and C. Develder. Adversarial training for multi-context joint entity and relation extraction. *arXiv preprint arXiv:1808.06876*, 2018.

K. Bontcheva and Y. Wilks. Automatic report generation from ontologies: the miakt approach. In *International conference on application of natural language to information systems*, pages 324–335. Springer, 2004.

S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

I. Caswell, C. Chelba, and D. Grangier. Tagged back-translation. *arXiv preprint arXiv:1906.06442*, 2019.

M. Chen, S. Wiseman, and K. Gimpel. Wikitablet: A large-scale data-to-text dataset for generating wikipedia article sections.

R. Cotterell and J. Kreutzer. Explaining and generalizing back-translation through wake-sleep. *arXiv preprint arXiv:1806.04402*, 2018.

M. Eberts and A. Ulges. Span-based joint entity and relation extraction with transformer pre-training. *arXiv preprint arXiv:1909.07755*, 2019.

S. Edunov, M. Ott, M. Auli, and D. Grangier. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381*, 2018.

L. Fang, T. Zeng, C. Liu, L. Bo, W. Dong, and C. Chen. Transformer-based conditional variational autoencoder for controllable story generation. *arXiv preprint arXiv:2101.00828*, 2021.

T. Ferreira, C. Gardent, N. Ilinykh, C. van der Lee, S. Mille, D. Moussallem, and A. Shimorina. The 2020 bilingual, bi-directional webnlg+ shared task overview and evaluation results (webnlg+ 2020). In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, 2020.

T. C. Ferreira, C. van der Lee, E. van Miltenburg, and E. Krahmer. Neural data-to-text generation: A comparison between pipeline and end-to-end architectures. *arXiv preprint arXiv:1908.09022*, 2019.

H. Fu, C. Li, X. Liu, J. Gao, A. Celikyilmaz, and L. Carin. Cyclical annealing schedule: A simple approach to mitigating kl vanishing. *arXiv preprint arXiv:1903.10145*, 2019.

C. Gardent, A. Shimorina, S. Narayan, and L. Perez-Beltrachini. The webnlg challenge: Generating text from rdf data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, 2017.

Q. Guo, Z. Jin, N. Dai, X. Qiu, X. Xue, D. Wipf, and Z. Zhang. \cal p2: A plan-and-pretrain approach for knowledge graph-to-text generation: A plan-and-pretrain approach for knowledge graph-to-text generation. In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+).*, pages 100–106, 2020a.

Q. Guo, Z. Jin, X. Qiu, W. Zhang, D. Wipf, and Z. Zhang. Cyclegt: Unsupervised graph-to-text and text-to-graph generation via cycle training. *arXiv preprint arXiv:2006.04702*, 2020b.

Q. Guo, Z. Jin, Z. Wang, X. Qiu, W. Zhang, J. Zhu, Z. Zhang, and W. David. Fork or fail: Cycle-consistent training with many-to-one mappings. In *International Conference on Artificial Intelligence and Statistics*, pages 1828–1836. PMLR, 2021.

D. Gupta and K. Berberich. Jigsaw: Structuring text into tables. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 237–244, 2019.

G. E. Hinton, P. Dayan, B. J. Frey, and R. M. Neal. The" wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, 1995.

Z. Hu, Z. Yang, X. Liang, R. Salakhutdinov, and E. P. Xing. Toward controlled generation of text. In *International Conference on Machine Learning*, pages 1587–1596. PMLR, 2017.

A. H. Jha, S. Anand, M. Singh, and V. Veeravasarapu. Disentangling factors of variation with cycle-consistent variational auto-encoders. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 805–820, 2018.

E. Jiménez-Ruiz, O. Hassanzadeh, V. Efthymiou, J. Chen, K. Srinivas, and V. Cutrona. Results of semtab 2020. In *CEUR Workshop Proceedings*, volume 2775, pages 1–8, 2020.

Z. Jin, Q. Guo, X. Qiu, and Z. Zhang. Genwiki: A dataset of 1.3 million content-sharing text and graphs for unsupervised graph-to-text generation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2398–2409, 2020.

M. Kale. Text-to-text pre-training for data-to-text tasks. *arXiv preprint arXiv:2005.10433*, 2020.

J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Y. Kim, M. Graça, and H. Ney. When and why is unsupervised neural machine translation useless? *arXiv preprint arXiv:2004.10581*, 2020.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

M.-A. Lachaux, B. Roziere, L. Chanussot, and G. Lample. Unsupervised translation of programming languages. *arXiv preprint arXiv:2006.03511*, 2020.

G. Lample, A. Conneau, L. Denoyer, and M. Ranzato. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*, 2017.

G. Lample, M. Ott, A. Conneau, L. Denoyer, and M. Ranzato. Phrase-based & neural unsupervised machine translation. *arXiv preprint arXiv:1804.07755*, 2018.

A. Lavie and A. Agarwal. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the second workshop on statistical machine translation*, pages 228–231, 2007.

R. Lebret, D. Grangier, and M. Auli. Neural text generation from structured data with application to the biography domain. *arXiv preprint arXiv:1603.07771*, 2016.

K. Lee, I.-C. Yoo, and D. Yook. Many-to-many voice conversion using cycle-consistent variational autoencoder with multiple decoders. *arXiv preprint arXiv:1909.06805*, 2019.

C. Li, X. Gao, Y. Li, B. Peng, X. Li, Y. Zhang, and J. Gao. Optimus: Organizing sentences via pre-trained modeling of a latent space. *arXiv preprint arXiv:2004.04092*, 2020.

Z. Lin, G. I. Winata, P. Xu, Z. Liu, and P. Fung. Variational transformers for diverse response generation. *arXiv preprint arXiv:2003.12738*, 2020.

H. Mei, M. Bansal, and M. R. Walter. What to talk about and how? selective generation using lstms with coarse-to-fine alignment. *arXiv preprint arXiv:1509.00838*, 2015.

A. Moryossef, Y. Goldberg, and I. Dagan. Step-by-step: Separating planning from realization in neural data-to-text generation. *arXiv preprint arXiv:1904.03396*, 2019.

S. Narayan and C. Gardent. Deep learning approaches to text production. *Synthesis Lectures on Human Language Technologies*, 13(1):1–199, 2020.

G. Paolini, B. Athiwaratkun, J. Krone, J. Ma, A. Achille, R. Anubhai, C. N. d. Santos, B. Xiang, and S. Soatto. Structured prediction as translation between augmented natural languages. *arXiv preprint arXiv:2101.05779*, 2021.

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

A. P. Parikh, X. Wang, S. Gehrmann, M. Faruqui, B. Dhingra, D. Yang, and D. Das. ToTTo: A controlled table-to-text generation dataset. In *Proceedings of EMNLP*, 2020.

R. Puduppully, L. Dong, and M. Lapata. Data-to-text generation with content selection and planning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6908–6915, 2019.

D. Putthividhya and J. Hu. Bootstrapped named entity recognition for product attribute extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1557–1567, 2011.

C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

L. F. Ribeiro, M. Schmitt, H. Schütze, and I. Gurevych. Investigating pretrained language models for graph-to-text generation. *arXiv preprint arXiv:2007.08426*, 2020.

M. Schmitt, S. Sharifzadeh, V. Tresp, and H. Schütze. An unsupervised joint system for text generation from knowledge graphs and semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7117–7130, 2020.

D. Sui, Y. Chen, K. Liu, J. Zhao, X. Zeng, and S. Liu. Joint entity and relation extraction with set prediction networks. *arXiv preprint arXiv:2011.01675*, 2020.

P. L. Tobing, Y.-C. Wu, T. Hayashi, K. Kobayashi, and T. Toda. Non-parallel voice conversion with cyclic variational autoencoder. *arXiv preprint arXiv:1907.10185*, 2019.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

J. Wang, Y. Hou, J. Liu, Y. Cao, and C.-Y. Lin. A statistical framework for product description generation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 187–192, 2017.

Z. Wei, J. Su, Y. Wang, Y. Tian, and Y. Chang. A novel cascade binary tagging framework for relational triple extraction. *arXiv preprint arXiv:1909.03227*, 2019.

S. Wiseman, S. M. Shieber, and A. M. Rush. Challenges in data-to-document generation. *arXiv preprint arXiv:1707.08052*, 2017.

S. Wiseman, S. M. Shieber, and A. M. Rush. Learning neural templates for text generation. *arXiv preprint arXiv:1808.10122*, 2018.

P. Xu, J. C. K. Cheung, and Y. Cao. On variational learning of controllable representations for text without supervision. In *International Conference on Machine Learning*, pages 10534–10543. PMLR, 2020.

H. Ye, N. Zhang, S. Deng, M. Chen, C. Tan, F. Huang, and H. Chen. Contrastive triple extraction with generative transformer. *arXiv preprint arXiv:2009.06207*, 2020a.

R. Ye, W. Shi, H. Zhou, Z. Wei, and L. Li. Variational template machine for data-to-text generation. *arXiv preprint arXiv:2002.01127*, 2020b.

D. Zeng, H. Zhang, and Q. Liu. Copymtl: Copy mechanism for joint extraction of entities and relations with multi-task learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9507–9514, 2020.

X. Zeng, D. Zeng, S. He, K. Liu, and J. Zhao. Extracting relational facts by an end-to-end neural model with copy mechanism. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 506–514, 2018.

T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.

C. Zhao, M. Walker, and S. Chaturvedi. Bridging the structural gap between encoding and decoding for data-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2481–2491, 2020.

S. Zhao, J. Song, and S. Ermon. Infovae: Information maximizing variational autoencoders. *arXiv preprint arXiv:1706.02262*, 2017.

Appendices

A Proofs and additional details

A.1 Supervised log-likelihood

If we define some priors $p(x)$ and $p(y)$ with separate parameters from $p_t(x \mid y)$ and $p_g(y \mid x)$, then we obtain the following equivalence:

$$\begin{aligned}
\max_\theta \log p_\theta(\mathcal{D}) &= \max_\theta \log p_\theta((x_1, y_1), \dots, (x_n, y_n)) \\
&= \max_\theta \sum_{i=1}^{n} \log p_\theta(x_i, y_i) \\
&= \max_\theta \frac{1}{2} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{T}\times\mathcal{G}} \left[\log \frac{p_\theta(x,y)}{p(x)} p(x) + \log \frac{p_\theta(x,y)}{p(y)} p(y) \right] \\
&= \max_\theta \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{T}\times\mathcal{G}} [\log p_\theta(y|x) + \log p_\theta(x|y) + p(x) + p(y)] \\
&= \max_\theta \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{T}\times\mathcal{G}} [\log p_g(y|x) + \log p_t(x|y)]
\end{aligned} \tag{20}$$

We are only interested in the parameters of p_t and p_g, and in the supervised case they can be directly optimized, independently of $p(x)$ and $p(y)$.

A.2 Unsupervised log-likelihood

In the unsupervised setting, we have non-parallel text and graphs datasets: $\mathcal{T} = (x_1, \dots x_n)$ and $\mathcal{G} = (y_1, \dots y_m)$. As always, our objective is to maximize the likelihood of our model on the data $p_\theta(\mathcal{D})$. With the assumption of i.i.d. and non-parallel samples, we can write:

$$\begin{aligned}
\log p_\theta(\mathcal{D}) &= \log p_\theta(x_1, \dots, x_n, y_1, \dots, y_m) \\
&= \sum_{i=1}^{n} \log p_\theta(x_i) + \sum_{j=1}^{m} \log p_\theta(y_j) \\
&= n \mathop{\mathbb{E}}_{x\sim\mathcal{T}} [\log p_\theta(x)] + m \mathop{\mathbb{E}}_{y\sim\mathcal{G}} [\log p_\theta(y)]
\end{aligned} \tag{21}$$

Note that the i.i.d. assumption is commonly made for datasets with pairs of inputs and labels $(x_i, y_i)_i$, in which case x_i and y_i are not independent. In our case however, the data is non-parallel, and can be viewed as a dataset with "missing pairs": $\mathcal{D} = ((x_1, \cdot), \dots, (x_n, \cdot), (\cdot, y_1), \dots, (\cdot, y_m))$, with x_i and y_j independent.

In the case $m = n$, our objective becomes

$$\max_\theta \mathop{\mathbb{E}}_{x\sim\mathcal{T}} [\log p_\theta(x)] + \mathop{\mathbb{E}}_{y\sim\mathcal{G}} [\log p_\theta(y)] \tag{22}$$

A.3 Generic ELBO

Without making any assumptions on the probabilistic model, for any latent variables x, y and z, we can derive a lower bound of the likelihood:

$$
\begin{aligned}
\log p_\theta(x) &= \log \frac{p_\theta(x,y,z)}{p_\theta(y,z \mid x)} \frac{q(y,z \mid x)}{q(y,z \mid x)} && \forall q, \forall y, z \\
&= \mathop{\mathbb{E}}_{y,z \sim q_\phi(y,z|x)} \left[\log \frac{p_\theta(x,y,z)}{q_\phi(y,z \mid x)} + \log \frac{q_\phi(y,z \mid x)}{p_\theta(y,z \mid x)} \right] && \forall q_\phi \\
&= \mathop{\mathbb{E}}_{y,z \sim q_\phi(y,z|x)} \left[\log \frac{p_\theta(x,y,z)}{q_\phi(y,z \mid x)} \right] + D_{KL}(q_\phi(y,z \mid x) \parallel p_\theta(y,z \mid x)) \\
&\geq \mathop{\mathbb{E}}_{y,z \sim q_\phi(y,z|x)} \left[\log \frac{p_\theta(x,y,z)}{q_\phi(y,z \mid x)} \right] && \text{Since KL divergence is always non-negative} \\
&= \mathop{\mathbb{E}}_{y,z \sim q_\phi(y,z|x)} \left[\log \frac{p_\theta(x \mid y,z) p_\theta(y,z)}{q_\phi(y,z \mid x)} \right] \\
&= \mathop{\mathbb{E}}_{y,z \sim q_\phi(y,z|x)} [\log p_\theta(x \mid y,z)] - D_{KL}(q_\phi(y,z \mid x) \parallel p_\theta(y,z)) && (23)
\end{aligned}
$$

A.4 Deriving the $\mathcal{L}^{\text{auto-VAE}}$ objective

We can start by writing $q_\phi(y,z \mid x) = q_\phi(y \mid z,x) q_\phi(z \mid x)$. Since the symmetrical text-to-graph model verifies $p_\theta(y \mid z,x) = p_\theta(y \mid x)$, we can *make the assumption that* $q_\phi(y \mid z,x) = q_\phi(y \mid z)$. This implies that:

$$
q_\phi(y,z \mid x) = q_\phi(y \mid z) q_\phi(z \mid x)
$$

Since we also have $p_\theta(y,z) = p_\theta(y \mid z) p_\theta(z)$, the objective becomes:

$$
\log p_\theta(x) \geq \mathop{\mathbb{E}}_{z \sim q_\phi(z|x), \hat{y} \sim q_\phi(y|z)} [\log p_\theta(x \mid z)] - D_{KL}(q_\phi(z \mid x) \parallel p_\theta(z)) - \mathop{\mathbb{E}}_{z \sim q_\phi(z|x)} [D_{KL}(q_\phi(y \mid z) \parallel p_\theta(y \mid z))
$$

By writing the inference model this way, we directly use $z \sim q_\phi(z \mid x)$ to reconstruct x and we never use our synthetic sample $\hat{y} \sim q_\phi(y \mid z)$. The term $q_\phi(y \mid z)$ will be matched exactly to a prior $p_\theta(y \mid z)$, due to the second KL divergence. In our setting, it is not clear what should be $q_\phi(y \mid z)$ and $p_\theta(y \mid z)$, except using our text-to-graph decoder $p_g(y \mid z)$ for both. In this case the second KL term disappears, and we obtain:

$$
\log p_\theta(x) \geq \mathop{\mathbb{E}}_{z \sim q_\phi(z|x)} [\log p_\theta(x \mid z)] - D_{KL}(q_\phi(z \mid x) \parallel p_\theta(z))
$$

B Additional results

Seen categories						
		BLEU	METEOR	BERTScore		
				F1	P	R
	P2 (supervised)	60.4	0.434	0.962	0.964	0.961
	bt5 (supervised)	61.1	0.433	0.963	0.965	0.961
	CycleGT (unsupervised)	47.4	0.394	0.950	0.951	0.950
	Baseline (supervised)	57.030	0.412	0.959	0.963	0.955
	Baseline (unsupervised)	50.58	0.380	0.951	0.957	0.946
Ours	StyleVAE	51.890	0.389	0.952	0.957	0.949
	FullVAE (joint loss)	44.140	0.347	0.938	0.947	0.930
	FullVAE (single loss)	51.880	0.397	0.951	0.954	0.949
Unseen categories						
		BLEU	METEOR	BERTScore		
				F1	P	R
	P2 (supervised)	49.2	0.404	0.954	0.957	0.953
	bt5 (supervised)	44.0	0.393	0.947	0.948	0.948
	CycleGT (unsupervised)	40.9	0.379	0.945	0.945	0.946
	Baseline (supervised)	42.650	0.373	0.946	0.949	0.944
	Baseline (unsupervised)	40.94	0.362	0.942	0.946	0.940
Ours	StyleVAE	41.450	0.369	0.944	0.947	0.943
	FullVAE (joint loss)	37.430	0.347	0.937	0.944	0.932
	FullVAE (single loss)	40.290	0.367	0.941	0.944	0.939
Unseen entities						
		BLEU	METEOR	BERTScore		
				F1	P	R
	P2 (supervised)	52.3	0.413	0.961	0.963	0.960
	bt5 (supervised)	50.8	0.415	0.959	0.961	0.959
	CycleGT (unsupervised)	46.6	0.390	0.955	0.956	0.954
	Baseline (supervised)	49.010	0.396	0.958	0.960	0.956
	Baseline (unsupervised)	45.38	-	0.952	0.955	0.950
Ours	StyleVAE	45.870	0.384	0.951	0.953	0.950
	FullVAE (joint loss)	39.400	0.342	0.941	0.950	0.934
	FullVAE (single loss)	45.720	0.389	0.953	0.955	0.952

Table 3: Data-to-Text results on seen categories, unseen categories, and unseen entities

		Relations			Entities				
Seen categories		F1	P	R	F1	P	R	Graph Acc.	Format err.
	P2 (supervised)	0.693	0.693	0.694	-	-	-	-	-
	bt5 (supervised)	0.877	0.875	0.880	-	-	-	-	-
	CycleGT (unsupervised)	0.548	0.541	0.560	-	-	-	-	-
	Baseline (supervised)	0.694	0.725	0.665	0.866	0.896	0.838	0.337	0.008
	Baseline (unsupervised)	0.479	0.531	0.437	0.717	0.771	0.670	0.128	0.018
Ours	StyleVAE	0.483	0.535	0.440	0.724	0.776	0.678	0.154	0.013
	FullVAE (joint loss)	0.271	0.280	0.261	0.572	0.580	0.564	0.025	0.287
	FullVAE (single loss)	0.425	0.455	0.399	0.706	0.735	0.678	0.105	0.097
Unseen categories		Relations			Entities				
		F1	P	R	F1	P	R	Graph Acc.	Format err.
	P2 (supervised)	0.658	0.657	0.660	-	-	-	-	-
	bt5 (supervised)	0.551	0.540	0.568	-	-	-	-	-
	CycleGT (unsupervised)	0.223	0.222	0.227	-	-	-	-	-
	Baseline (supervised)	0.185	0.188	0.182	0.581	0.579	0.584	0.118	0.057
	Baseline (unsupervised)	0.089	0.094	0.084	0.485	0.494	0.476	0.075	0.052
Ours	StyleVAE	0.094	0.098	0.090	0.487	0.495	0.480	0.071	0.059
	FullVAE (joint loss)	0.091	0.094	0.088	0.437	0.437	0.437	0.045	0.424
	FullVAE (single loss)	0.101	0.105	0.097	0.476	0.476	0.477	0.063	0.223
Unseen entities		Relations			Entities				
		F1	P	R	F1	P	R	Graph Acc.	Format err.
	P2 (supervised)	0.746	0.746	0.747	-	-	-	-	-
	bt5 (supervised)	0.649	0.617	0.701	-	-	-	-	-
	CycleGT (unsupervised)	0.239	0.238	0.247	-	-	-	-	-
	Baseline (supervised)	0.437	0.452	0.422	0.697	0.714	0.680	0.169	0.024
	Baseline (unsupervised)	0.318	0.349	0.292	0.591	0.629	0.558	0.09	0.036
Ours	StyleVAE	0.312	0.341	0.288	0.595	0.630	0.564	0.095	0.051
	FullVAE (joint loss)	0.195	0.202	0.188	0.515	0.523	0.506	0.034	0.316
	FullVAE (single loss)	0.249	0.270	0.232	0.551	0.579	0.525	0.075	0.183

Table 4: Text-to-Graph results on seen categories, unseen categories, and unseen entities