An Unsupervised Joint System for Text Generation from Knowledge Graphs and Semantic Parsing

Martin Schmitt[1] **Sahand Sharifzadeh**[2] **Volker Tresp**[2,3] **Hinrich Schütze**[1]

[1]Center for Information and Language Processing (CIS), LMU Munich
[2]Department of Informatics, LMU Munich
[3]Siemens AG Munich
martin@cis.lmu.de

Abstract

Knowledge graphs (KGs) can vary greatly from one domain to another. Therefore supervised approaches to both graph-to-text generation and text-to-graph knowledge extraction (semantic parsing) will always suffer from a shortage of domain-specific parallel graph-text data; at the same time, adapting a model trained on a different domain is often impossible due to little or no overlap in entities and relations. This situation calls for an approach that (1) does not need large amounts of annotated data and thus (2) does not need to rely on domain adaptation techniques to work well in different domains. To this end, we present the *first approach to unsupervised text generation from KGs* and show simultaneously how it can be used for *unsupervised semantic parsing*. We evaluate our approach on WebNLG v2.1 and a new benchmark leveraging scene graphs from Visual Genome. Our system outperforms strong baselines for both text↔graph conversion tasks without any manual adaptation from one dataset to the other. In additional experiments, we investigate the impact of using different unsupervised objectives.[1]

1 Introduction

Knowledge graphs (KGs) are a general-purpose approach for storing information in a structured, machine-accessible way (Van Harmelen et al., 2008). They are used in various fields and domains to model knowledge about topics as different as lexical semantics (Fellbaum, 2005; van Assem et al., 2006), common sense (Speer et al., 2017; Sap et al., 2019), biomedical research (Wishart et al., 2018) and visual relations in images (Lu et al., 2016).

This ubiquity of KGs necessitates interpretability because diverse users – both experts and non-experts – work with them. Even though, in principle, a KG is human-interpretable, non-experts may have difficulty making sense of it. Thus, there is a need for methods, such as automatic natural language generation ("graph→text"), that support them.

Semantic parsing, i.e., the conversion of a text to a formal meaning representation, such as a KG, ("text→graph") is equally important because it makes information that only exists in text form accessible to machines, thus assisting knowledge base engineers in KG creation and completion.

As KGs are so flexible in expressing various kinds of knowledge, separately created KGs vary a lot. This unavoidably leads to a shortage of training data for both graph↔text tasks. We therefore propose an unsupervised model that (1) easily adapts to new KG domains and (2) only requires unlabeled (i.e., non-parallel) texts and graphs from the target domain, together with a few fact extraction heuristics, but no manual annotation.

To show the effectiveness of our approach, we conduct experiments on the latest release (v2.1) of the WebNLG corpus (Shimorina and Gardent, 2018) and on a new benchmark we derive from *Visual Genome* (Krishna et al., 2016). While both of these datasets contain enough annotations to train supervised models, we evaluate our unsupervised approach by ignoring these annotations. The datasets are particularly well-suited for our evaluation as both graphs and texts are completely human-generated. Thus for both our tasks, models are evaluated with natural, i.e., human-generated targets.

Concretely, we make the following contributions: (1) We present the first unsupervised non-template approach to text generation from KGs (graph→text). (2) We jointly develop a new unsupervised approach to semantic parsing that automatically adjusts to a target KG schema (text→graph). (3) In contrast to prior unsupervised graph→text and text→graph work, our model does not re-

[1] https://github.com/mnschmit/unsupervised-graph-text-conversion

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 7117–7130,
November 16–20, 2020. ©2020 Association for Computational Linguistics

quire manual adaptation to new domains or graph schemas. (4) We provide a thorough analysis of the impact of different unsupervised objectives, especially the ones we newly introduce for text↔graph conversion. (5) We create a new large-scale dataset for text↔graph transformation tasks in the visual domain.

2 Related Work

graph → text. Our work is the first attempt at fully unsupervised text generation from KGs. In this respect it is only comparable to traditional rule- or template-based approaches (Kukich, 1983; McRoy et al., 2000). However, in contrast to these approaches, which need to be manually adapted to new domains and KG schemas, our method is generally applicable to all kinds of data without modification.

There is a large body of literature about supervised text generation from structured data, notably about the creation of sports game summaries from statistical records (Robin, 1995; Tanaka-Ishii et al., 1998). Recent efforts make use of neural encoder-decoder mechanisms (Wiseman et al., 2017; Puduppully et al., 2019). Although text creation from relational databases is related and our unsupervised method is, in principle, also applicable to this domain, in our work we specifically address text creation from graph-like structures such as KGs.

One recent work on supervised text creation from KGs is (Bhowmik and de Melo, 2018). They generate a short description of an entity, i.e., a single KG node, based on a set of facts about the entity. We, however, generate a description of the whole KG, which involves multiple entities and their relations. Koncel-Kedziorski et al. (2019) generate texts from whole KGs. They, however, do not evaluate on human-generated KGs but automatically generated ones from the scientific information extraction tool SciIE (Luan et al., 2018). Their supervised model is based on message passing through the topology of the incidence graph of the KG input. Such graph neural networks (Kipf and Welling, 2017; Veličković et al., 2018) have been widely adopted in supervised graph-to-text tasks (Beck et al., 2018; Damonte and Cohen, 2019; Ribeiro et al., 2019, 2020).

Even though Marcheggiani and Perez-Beltrachini (2018) report that graph neural networks can make better use of graph input than RNNs for supervised learning, for our unsupervised approach we follow the line of research that uses RNN-based sequence-to-sequence models (Cho et al., 2014; Sutskever et al., 2014) operating on serialized triple sets (Gardent et al., 2017b; Trisedya et al., 2018; Gehrmann et al., 2018; Castro Ferreira et al., 2019; Fan et al., 2019). We make this choice because learning a common semantic space for both texts and graphs by means of a shared encoder and decoder is a central component of our model. It is a nontrivial, separate research question whether and how encoder-decoder parameters can effectively be shared for models working on both sequential and non-sequential data. We thus leave the adaptation of our approach to graph neural networks for future work.

text → graph. Converting a text into a KG representation, our method is an alternative to prior work on open information extraction (Niklaus et al., 2018) with the advantage that the extractions, though trained without labeled data, automatically adjust to the KGs used for training. It is therefore also related to relation extraction in the unsupervised (Yao et al., 2011; Marcheggiani and Titov, 2016; Simon et al., 2019) and distantly supervised setting (Riedel et al., 2010; Parikh et al., 2015). However, these systems merely predict a single relation between two given entities in a single sentence, while we translate a whole text into a KG with potentially multiple facts.

Our text→graph task is therefore most closely related to semantic parsing (Kamath and Das, 2019), but we convert statements into KG facts whereas semantic parsing typically converts a question into a KG or database query. Poon and Domingos (2009) proposed the first unsupervised approach. They, however, still need an additional KG alignment step, i.e., are not able to directly adjust to the target KG. Other approaches overcome this limitation but only in exchange for the inflexibility of manually created domain-specific lexicons (Popescu et al., 2004; Goldwasser et al., 2011). Poon (2013)'s approach is more flexible but still relies on preprocessing by a dependency parser, which generally means that language-specific annotations to train such a parser are needed. Our approach is end-to-end, i.e., does not need any language-specific preprocessing during inference and only depends on a POS tagger used in the rule-based text→graph system to bootstrap training.

Unsupervised sequence generation. Our unsupervised training regime for both text↔graph tasks is inspired by (Lample et al., 2018b). They used self-supervised pretraining and backtranslation for unsupervised translation from one language to another. We adapt these principles and their noise model to our tasks, and introduce two new noise functions specific to text↔graph conversion.

3 Preliminaries

3.1 Data structure

We formalize a KG as a labeled directed multigraph (V, E, s, t, l) where entities are nodes V and edges E represent relations between entities. The lookup functions $s, t : E \to V$ assign to each edge its source and target node. The labeling function l assigns labels to nodes and edges where node labels are entity names and edge labels come from a predefined set $\mathcal{R}$ of relation types.

An equivalent representation of a KG is the set of its facts. A fact is a triple consisting of an edge's source node (the subject), the edge itself (the predicate), and its target node (the object). So the set of facts $\mathcal{F}$ of a KG can be obtained from its edges:

$$\mathcal{F} := \{\, (s(e), e, t(e)) \mid e \in E \,\} .$$

Applying l to all triple elements and writing out $\mathcal{F}$ in an arbitrary order generates a serialization that makes the KG accessible to sequence models otherwise used only for text. This has the advantage that we can train a sequence encoder to embed text and KGs in the same semantic space. Specifically, we serialize a KG by writing out its facts separated with end-of-fact symbols (`EOF`) and elements of each fact with special `SEP` symbols. We thus define our task as a sequence-to-sequence (seq2seq) task.

3.2 Scene Graphs

The Visual Genome (VG) repository is a large collection of images with associated manually annotated **scene graphs**; see Fig. 1. A scene graph formally describes image objects with their attributes, e.g., (hydrant, `attr`, yellow), and their relations to other image objects, e.g., (woman, in, shorts). Each scene graph is organized into smaller subgraphs, known as **region graphs**, representing a subpart of a more complex larger picture that is interesting on its own. Each region graph is associated with an English text, the **region description**. Texts and graphs were not automatically produced from each other, but were collected from crowdworkers who were presented an image region and then generated text and graph. So although the graphs were not specifically designed to closely resemble the texts, they describe the same image region. This semantic correspondence makes scene graph↔text conversion an interesting and challenging problem because text and graph are not simple translations of each other.

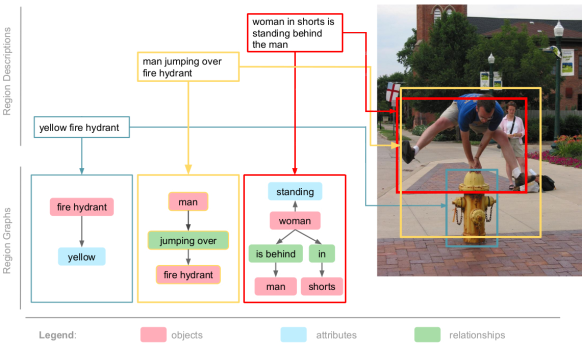

Figure 1: Region graphs and textual region descriptions in Visual Genome (VG). Image regions serve as common reference for text and graph creation but are disregarded in our work. We solely focus on the pairs of corresponding texts and graphs. Illustration adapted from (Krishna et al., 2016).

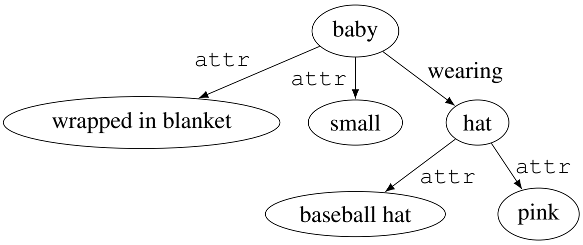

Figure 2: Example graph in our new VG benchmark.

Scene graphs are formalized in the same way as other KGs: V here contains image objects and their attributes, and $\mathcal{R}$ contains all types of visual relationships and the special label `attr` for edges between attribute and non-attribute nodes. Fig. 2 shows an example.

VG scene graphs have been used before for traditional KG tasks, such as KG completion (Wan et al., 2018), but we are the first to use them for a text↔graph conversion dataset.

4 Approaches

4.1 Rule-based systems

We propose a rule-based system as unsupervised baseline for each of the text↔graph tasks. Note that they both assume that the texts are in English.

R$^{\text{graph}\to\text{text}}$. From a KG serialization, we remove

noise function	behavior
swap	applies a random permutation σ of words or facts with $\forall i \in \{1, \ldots, n\}, \lvert\sigma(i) - i\rvert \leq k$; $k = 3$ for text, $k = +\infty$ for knowledge graphs.
drop	removes each fact/word with a probability of p_{drop}.
blank	replaces each fact/word with a probability of p_{blank} by a special symbol blanked.
repeat	inserts repetitions with a probability of p_{repeat} in a sequence of facts/words.
rule	generates a noisy translation by applying $\mathrm{R}^{\text{graph}\rightarrow\text{text}}$ to a graph or $\mathrm{R}^{\text{text}\rightarrow\text{graph}}$ to a text.

Table 1: Noise functions and their behavior on graphs and texts.

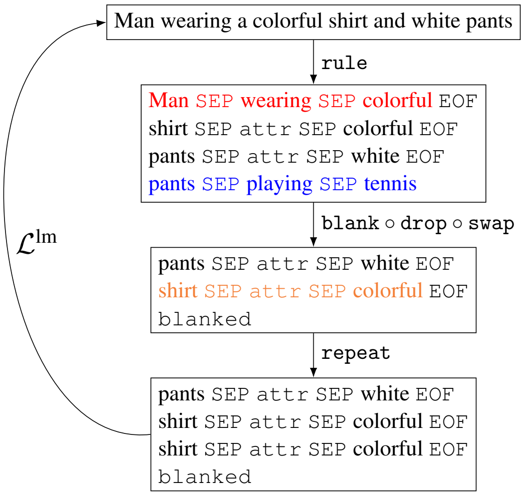
Man wearing a colorful shirt and white pants
rule
Man SEP wearing SEP colorful EOF
shirt SEP attr SEP colorful EOF
pants SEP attr SEP white EOF
pants SEP playing SEP tennis
blank ∘ drop ∘ swap
pants SEP attr SEP white EOF
shirt SEP attr SEP colorful EOF
blanked
repeat
pants SEP attr SEP white EOF
shirt SEP attr SEP colorful EOF
shirt SEP attr SEP colorful EOF
blanked
$\mathcal{L}^{\text{lm}}$

Figure 3: Example noisy training instance for the graph→text task in the composed noise setting. The fact highlighted in red is removed by drop, the one in blue is replaced with blanked by blank, the one in orange is repeated by repeat.

SEP symbols and replace EOF symbols by the word *and*. The special label attr is mapped to *is*. This corresponds to a template-based enumeration of all KG facts. See Table 5 for an example.

$\mathbf{R}^{\text{text}\rightarrow\text{graph}}$. After preprocessing a text with NLTK's default POS tagger (Loper and Bird, 2004) and removing stop words, we apply two simple heuristics to extract facts: (1) Each verb becomes a predicate; *is* creates facts with predicate attr. The content words directly before and after such a predicate word become subject and object. (2) Adjectives a form attributes, i.e., build facts of the form $(X, \texttt{attr}, a)$ where X is filled with the first noun after a. These heuristics are similar in nature to a rudimentary parser. See Table 8 for an example.

4.2 Neural seq2seq systems

Our main system is a neural seq2seq architecture. We equip the standard encoder-decoder model with attention (Bahdanau et al., 2014) and copy mechanism (Gu et al., 2016). Allowing the model to directly copy from the source to the target side is beneficial in data to text generation (Wiseman et al., 2017; Puduppully et al., 2019). The encoder (resp. decoder) is a bidirectional (resp. unidirectional) LSTM (Hochreiter and Schmidhuber, 1997). Dropout (Hinton et al., 2012) is applied at the input of both encoder and decoder (Britz et al., 2017). We combine this model with the following concepts:

Multi-task model. In unsupervised machine translation, systems are trained for both translation directions (Lample et al., 2018b). In the same way, we train our system for both conversion tasks text↔graph, sharing encoder and decoder. To tell the decoder which type of output should be produced (text or graph), we initialize the cell state of the decoder with an embedding of the desired output type. The hidden state of the decoder is initialized with the last state of the encoder as usual.

Noisy source samples. Lample et al. (2018a) introduced denoising auto-encoding as pretraining and auxiliary task to train the decoder to produce well-formed output and make the encoder robust to noisy input. The training examples for this task consist of a noisy version of a sentence as source and the original sentence as target. We adapt this idea and propose the following noise functions for the domains of graphs and texts: swap, drop, blank, repeat, rule. Table 1 describes their behavior. swap, drop and blank are adapted from (Lample et al., 2018a) with facts in graphs taking the role of words in text. As order should be irrelevant in a set of facts, we drop the locality constraint in the swap permutation for graphs by setting $k = +\infty$.

Denoising samples generated by repeat requires the model to learn to remove redundant information in a set of facts. In the case of text, repeat mimics a behavior often observed with insufficiently trained neural models, i.e., repeating words considered important.

Unlike the other noise functions, rule does not "perturb" its input, but rather noisily backtranslates it. We will see in Section 7 that bootstrapping with these noisy translations is essential.

We consider two fundamentally different noise injection regimes: (1) The **composed noise** setting is an adaptation of Lample et al. (2018a)'s noise model ($\texttt{blank}\circ\texttt{drop}\circ\texttt{swap}$) where our newly introduced noise functions `rule` and `repeat` are added to the start and end of the pipeline, i.e., all data samples are treated equally with the same noise function $C_{\text{comp}} := \texttt{repeat}\circ\texttt{blank}\circ\texttt{drop}\circ\texttt{swap}\circ\texttt{rule}$. Figure 3 shows an example. (2) In the **sampled noise** setting, we do not use all noise functions at once but sample a single one per data instance.

4.3 Training regimes

We denote the sets of graphs and corresponding texts by $\mathcal{G}$ and $\mathcal{T}$. The set of available supervised examples $(x, y) \in \mathcal{G} \times \mathcal{T}$ is called $\mathcal{S} \subset \mathcal{G} \times \mathcal{T}$. P_g and P_t are probabilistic models that generate, conditioned on any input, a graph (g) or a text (t).

Unsupervised training. We first obtain a language model for both graphs and text by training one epoch with the denoising auto-encoder objective:

$$\mathcal{L}^{\text{denoise}} = \mathop{\mathbb{E}}_{x\sim\mathcal{G}}[-\log P_g(x|C(x))] + \mathop{\mathbb{E}}_{y\sim\mathcal{T}}[-\log P_t(y|C(y))]$$

where $C \in \{C_{\text{comp}}\}$ for composed noise and $C \in \{\texttt{swap}, \texttt{blank}, \texttt{drop}, \texttt{repeat}, \texttt{rule}\}$ for sampled noise. In this pretraining epoch only, we use all possible noise functions individually on all available data. As sampled noise incorporates five different noise functions and composed noise only one, this results in five times more pretraining samples for sampled noise than for composed noise.

In subsequent epochs, we additionally consider $\mathcal{L}^{\text{back}}$ as training signal:

$$\mathcal{L}^{\text{back}} = \mathop{\mathbb{E}}_{x\sim\mathcal{G}}[-\log P_g(x|z^*(x))] + \mathop{\mathbb{E}}_{y\sim\mathcal{T}}[-\log P_t(y|w^*(y))]$$

$$z^*(x) = \arg\max_z P_t(z|x)$$

$$w^*(y) = \arg\max_w P_g(w|y)$$

This means that, in each iteration, we apply the current model to backtranslate a text (graph) to obtain a potentially imperfect graph (text) that we can use as noisy source with the clean original input being the target. This gives us a pseudo-parallel training instance for the next iteration – recall that we address unsupervised generation, i.e., without access to parallel data.

The total loss in these epochs is $\mathcal{L}^{\text{back}} + \mathcal{L}^{\text{denoise}}$, where now $\mathcal{L}^{\text{denoise}}$ only samples one possible type of noise independently for each data instance.

Supervised training. Our intended application is an unsupervised scenario. For our two datasets, however, we have labeled data (i.e., a "parallel corpus") and so can also compare our model to its supervised variant. Although supervised performance is generally better, it serves as a reference point and gives us an idea of the impact of supervision as opposed to factors like model architecture and hyperparameters. The supervised loss is simply defined as follows:

$$\mathcal{L}^{\text{sup}} = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{S}}[-\log P_t(y|x) - \log P_g(x|y)]$$

	VG	VG_{ball}	WebNLG
train split size	2,412,253	151,790	34,338
val split size	323,478	21,541	4,313
test split size	324,664	20,569	4,222
#relation types	36,506	5,167	373
avg #facts in graph	2.7	2.5	3.0
avg #tokens in text	5.4	5.5	22.8
avg % text tokens in graph	49.3	50.6	49.4
avg % graph tokens in text	52.3	54.7	75.6

Table 2: Statistics of WebNLG v2.1 and our newly created benchmark VG; VG_{ball} is a subset of VG representing images from ball sports events. Data split sizes are given as number of graph-text pairs.

5 Experiments

5.1 Data

For our experiments, we randomly split the VG images 80/10/10 into train/val/test. We then remove all graphs from train that also occur in one of the images in val or test. Finally, we unify graph serialization duplicates with different texts to single instances with multiple references for graph→text and proceed analogously with text duplicates for text→graph. For WebNLG v2.1, we use the data splits as provided. Following (Gardent et al., 2017a), we resolve the camel case of relation names and remove underscores from entity names in a preprocessing step. For both datasets, the order of facts in graph serializations corresponds to the order of triples in the original dataset. Because of VG's enormous size and limited computation power, we additionally create a closed-domain ball

graph → text	Visual Genome						WebNLG					
	BLEU		METEOR		CHRF++		BLEU		METEOR		CHRF++	
	val	test	val	test	val	test	val	test	val	test	val	test
Rgraph→text	5.9	5.9	28.2	28.1	43.4	43.3	18.3	18.3	33.5	33.6	55.0	55.2
Ours w/ sampled noise	19.8	19.5	31.4	31.2	50.9	50.7	**39.1**	**37.7**	**35.4**	**35.5**	**61.9**	**62.1**
Ours w/ composed noise	**23.2**	**23.2**	**33.0**	**32.9**	**53.7**	**53.6**	30.8	30.5	30.2	30.0	53.1	52.8
Ours *supervised*	26.5	26.4	32.3	32.2	53.7	53.6	35.1	34.4	39.6	39.5	64.1	64.0

Table 3: Results for unsupervised and supervised text generation. Note that training a supervised model on millions of labeled samples is usually not an option. Best unsupervised models are identified by best BLEU on $\mathcal{V}_{100}$. BLEU and METEOR are computed with scripts from (Lin et al., 2018); the CHRF++ script is from (Popović, 2017b).

sports subset of VG, called VG$_{\text{ball}}$, which we can use to quickly conduct additional experiments (see Section 7). We identify all images where at least one region graph contains at least one fact that mentions an object ending with *ball* and take all regions from them (keeping data splits the same). In contrast to alternatives like random subsampling, we consider this domain-focused construction more realistic.

Table 2 shows relevant statistics for all datasets. While VG and WebNLG have similar statistics, VG is around 70 times larger than WebNLG, which makes it an interesting benchmark for future research, both supervised and unsupervised. Apart from size, there are two important differences: (1) The VG graph schema has been freely defined by crowd workers and thus features a large variety of different relations. (2) The percentage of graph tokens occurring in the text, a measure important for the text→graph task, is lower for VG than for WebNLG. Thus, VG graphs contain more details than their corresponding texts, which is a characteristic feature of the domain of image captions: they mainly describe the salient image parts.

5.2 Training details

We train all models with the Adam optimizer (Kingma and Ba, 2015) for maximally 30 epochs. We stop supervised models early when $\mathcal{L}^{\text{sup}}$ does not decrease on val for 10 epochs. Unsupervised models are stopped after 5 iterations on VG because of its big size and limited computational resources. All hyperparameters and more details are described in Appendices A and B. Our implementation is based on AllenNLP (Gardner et al., 2017).

In unsupervised training, input graphs and texts are the same as in supervised training – only the gold target sides are ignored. While it is an artificial setup to split paired data and treat them as unpaired, this not only makes the supervised and unsupervised settings more directly comparable, but also ensures that the text data resemble the evaluation texts in style and domain. For the purpose of experiments on a benchmark, this seems appropriate to us. For a concrete use case, it would be an important first step to find adequate texts that showcase the desired language style and that are about a similar topic as the KGs that are to be textualized. As KGs are rarely the only means of storing information, e.g., in an industrial context, such texts should not be hard to come by in practice.

	sampled noise				composed noise			
#	$\mathcal{U}$	$\mathcal{V}_{100}$	val	test	$\mathcal{U}$	$\mathcal{V}_{100}$	val	test
1	**80.4**	7.8	10.1	9.9	**72.2**	15.9	19.8	19.7
2	50.7	7.2	9.2	9.1	41.2	14.0	15.2	15.1
3	67.6	19.5	19.4	19.2	61.0	22.7	**23.5**	**23.4**
4	56.4	**21.2**	**19.8**	**19.5**	51.9	22.2	21.4	21.3
5	62.9	20.0	19.6	19.4	60.5	**24.5**	23.2	23.2

Table 4: BLEU scores on VG for our unsupervised models evaluated for graph→text at different iterations. $\mathcal{U}$ is calculated on all unlabeled data used for training. $\mathcal{V}_{100}$ is a 100-size random sample from val. All results are computed with scripts from (Lin et al., 2018).

6 Results and Discussion

6.1 Text generation from graphs

Model selection. Table 4 shows how performance of our unsupervised model changes at every back-translation iteration, measured in BLEU (Papineni et al., 2002), a common metric for natural language generation. For model selection, we adopt the two methods proposed by Lample et al. (2018b), i.e., a small validation set (we take a 100-size random subset of val, called $\mathcal{V}_{100}$) and a fully unsupervised criterion ($\mathcal{U}$) where BLEU compares an unlabeled sample with its back-and-forth translation. We confirm their finding that $\mathcal{U}$ is not reliable for neural

(a) Reference text	a baseball cap on a baby's head
(b) $\mathrm{R}^{\text{graph}\rightarrow\text{text}}$	baby is small and baby is wrapped in blanket and hat is pink and hat is baseball hat and baby wearing hat
(c) Unsuperv. neural model	small baby wrapped in blanket with pink baseball hat
(d) Superv. neural model	baby wearing a pink hat

Table 5: Texts generated from graph in Fig. 2.

text generation models whereas $\mathcal{V}_{100}$ correlates better with performance on the larger test sets. We use $\mathcal{V}_{100}$ for model selection in the rest of this paper.

Quantitative evaluation. Table 3 shows BLEU, METEOR (Banerjee and Lavie, 2005) and CHRF++ (Popović, 2017a) for our unsupervised models and the rule baseline $\mathrm{R}^{\text{graph}\rightarrow\text{text}}$, which is in many cases, i.e., if parallel graph-text data are scarce, the only alternative.

First, we observe that $\mathrm{R}^{\text{graph}\rightarrow\text{text}}$ performs much better on WebNLG than VG, indicating that our new benchmark poses a tougher challenge. Second, our unsupervised models consistently outperform this baseline on all metrics and on both datasets, showing that our method produces textual descriptions much closer to human-generated ones. Third, noise composition, the general default in unsupervised machine translation, does not always perform better than noise sampling. Thus, it is worthwhile to try different noise settings for new tasks or datasets.

Surprisingly, supervised and unsupervised models perform nearly on par. Real supervision does not seem to give much better guidance in training than our unsupervised regime, as measured by our three metrics on two different datasets. Some metric-dataset combinations even favor one of the unsupervised models. Our qualitative observations provide a possible explanation for that.

Qualitative observations. Taking a look at example generations (Table 5), we also see qualitatively how much easier it is to grasp the content of our natural language summarization than reading through a simple enumeration of KG facts. We find that the unsupervised model (c) seems to output the KG information in a more complete manner than its supervised counterpart (d). The supervision probably introduces a bias present in the training data that image captions focus on salient image parts and therefore the supervised model is encouraged to omit information. As it never sees a corresponding text-graph pair together, the unsupervised model cannot draw such a conclusion.

	sampled noise				composed noise			
#	$\mathcal{U}$	$\mathcal{V}_{100}$	val	test	$\mathcal{U}$	$\mathcal{V}_{100}$	val	test
1	19.1	1.0	1.2	1.2	17.0	2.0	2.2	2.2
2	**71.0**	**21.7**	**19.1**	**18.8**	49.3	**22.1**	**22.1**	**21.7**
3	58.2	19.3	18.6	18.3	45.9	18.7	19.7	19.4
4	62.3	18.3	**19.1**	**18.8**	**54.4**	19.9	20.8	20.5
5	63.7	19.8	19.0	18.7	49.0	18.8	19.0	18.8

Table 6: F1 scores on VG for our models from Table 4 evaluated on text→graph at different iterations.

text → graph	VG		WebNLG	
	val	test	val	test
$\mathrm{R}^{\text{text}\rightarrow\text{graph}}$	13.4	13.1	0.0	0.0
Stanford SG Parser	19.5	19.3	0.0	0.0
Ours w/ sampled noise	19.1	18.8	**38.5**	**39.1**
Ours w/ composed noise	**22.1**	**21.7**	32.5	33.1
Ours *supervised*	23.5	23.0	52.8	52.8

Table 7: F1 scores of facts extracted by our unsupervised semantic parsing (text→graph) systems and our model trained with supervision.

6.2 Graph extraction from texts

We evaluate semantic parsing (text→graph) performance by computing the micro-averaged F1 score of extracted facts. If there are multiple reference graphs (cf. Section 5.1), an extracted fact is considered correct if it occurs in at least one reference graph. For the ground truth number of facts to be extracted from a given text, we take the maximum number of facts of all its reference graphs.

Model selection. Table 6 shows that (compared to text generation quality) $\mathcal{U}$ is more reliable for text→graph performance. For sampled noise, it correctly identifies the best iteration, whereas for composed noise it chooses second best. In both noise settings, $\mathcal{V}_{100}$ perfectly chooses the best model.

Quantitative observations. Table 7 shows a comparison of our unsupervised models with two rule-based systems, our $\mathrm{R}^{\text{text}\rightarrow\text{graph}}$ and the highly domain-specific Stanford Scene Graph Parser (SSGP) by Schuster et al. (2015).

We choose these two baselines to adequately represent the state of the art in the unsupervised setting. Recall from Section 2 that the only previous unsupervised works either cannot adapt to a target graph schema (open information extraction), which means their precision and recall of retrieved facts is always 0, or have been created for SQL query

Input sentence	Man wearing a colorful shirt and white pants playing tennis
Reference (RG)	(shirt, `attr`, colorful) (pants, `attr`, white) (man, wearing, shirt) (man, wearing, pants)
$\text{R}^{\text{text}\rightarrow\text{graph}}$	🟥 (Man, wearing, colorful) 🟩 (shirt, `attr`, colorful) 🟩 (pants, `attr`, white) 🟥 (pants, playing, tennis)
Stanford Scene Graph Parser	🟥 (shirt, play, tennis), 🟥 (pants, play, tennis), 🟩 (shirt, `attr`, colorful), 🟩 (pants, `attr`, white)
Unsuperv. model w/ composed noise	🟥 (pants, `attr`, colorful) 🟩 (pants, `attr`, white) 🟩 (man, wearing, shirt) 🟨 (man, playing, tennis)
Supervised model	🟩 (shirt, `attr`, colorful) 🟩 (pants, `attr`, white) 🟩 (Man, wearing, shirt) 🟩 (Man, wearing, pants)

Table 8: Example fact extractions and evaluation wrt reference graph (RG). Green: correct ($\in$ RG). Yellow: acceptable fact, but $\notin$ RG. Red: incorrect ($\notin$ RG).

	VG_{ball}		WebNLG	
	g→t BLEU	t→g F1	g→t BLEU	t→g F1
No noise	$\underline{0.9}$	$\underline{0.0}$	$\underline{14.8}$	$\underline{0.0}$
sample all noise funs	**19.9**	17.3	**39.1**	**38.5**
compose all noise funs	19.6	**19.0**	30.8	32.5
use only `rule`	**19.5**	**18.5**	37.4	**31.0**
use only `swap`	$\underline{0.9}$	$\underline{0.0}$	$\underline{13.1}$	$\underline{0.0}$
use only `drop`	$\underline{0.9}$	$\underline{0.0}$	**39.9**	30.1
use only `blank`	$\underline{0.9}$	$\underline{0.0}$	$\underline{14.9}$	$\underline{0.0}$
use only `repeat`	$\underline{1.1}$	$\underline{0.0}$	$\underline{15.7}$	$\underline{0.0}$
sample all but `rule`	$\underline{0.9}$	$\underline{0.0}$	$\underline{14.9}$	$\underline{0.0}$
sample all but `swap`	19.2	17.0	39.6	**37.3**
sample all but `drop`	19.5	16.0	39.2	35.3
sample all but `blank`	19.9	**17.5**	**41.0**	37.0
sample all but `repeat`	**20.4**	16.6	36.7	37.1
comp. all but `rule`	$\underline{0.9}$	$\underline{0.0}$	$\underline{13.5}$	$\underline{0.0}$
comp. all but `swap`	20.2	16.3	35.9	40.8
comp. all but `drop`	21.5	18.6	36.4	41.1
comp. all but `blank`	20.2	16.3	34.8	40.4
comp. all but `repeat`	**21.1**	**20.1**	**38.5**	**42.3**

Table 9: Ablation study of our models on val of VG_{ball} and WebNLG v2.1. Models selected based on $\mathcal{V}_{100}$. Bold: best performance per column and block. Underlined: worse than corresponding rule-based system.

generation from natural language questions (Poon, 2013), a related task that is yet so different that an adaptation to triple set generation from natural language statements is nontrivial. While rule-based systems do not automatically adapt to new graph schemas either, $\text{R}^{\text{text}\rightarrow\text{graph}}$ and SSGP were at least designed with the scene graph domain in mind.

Although SSGP was not optimized to match the scene graphs from VG, its rules were still engineered to cover typical idiosyncrasies of textual image descriptions and corresponding scene graphs. Besides, we evaluate it with lemmatized reference graphs because it only predicts lemmata as predicates. All this gives it a major advantage over the other presented systems but it is nonetheless outperformed by our best unsupervised model – even on VG. This shows that our automatic method can beat even hand-crafted domain-specific rules.

Both $\text{R}^{\text{text}\rightarrow\text{graph}}$ and SSGP fail to predict any fact from WebNLG. The DBpedia facts from WebNLG often contain multi-token entities while $\text{R}^{\text{text}\rightarrow\text{graph}}$ only picks single tokens from the text. Likewise, SSGP models multi-token entities as two nodes with an `attr` relation. This illustrates the importance of automatic adaptation to the target KG. Although our system uses $\text{R}^{\text{text}\rightarrow\text{graph}}$ during unsupervised training and is similarly not adapted to the WebNLG dataset, it performs significantly better.

Supervision helps more on WebNLG than on VG. The poor performance of $\text{R}^{\text{text}\rightarrow\text{graph}}$ on WebNLG is probably a handicap for unsupervised learning.

Qualitative observations. Table 8 shows example facts extracted by different systems. $\text{R}^{\text{text}\rightarrow\text{graph}}$ and SSGP are both fooled by the proximity of the noun *pants* and the verb *play* whereas our model correctly identifies *man* as the subject. It, however, fails to identify *shirt* as an entity and associates the two attributes *colorful* and *white* to *pants*. Only the supervised model produces perfect output.

6.3 Noise and translation completeness

Sampled noise only creates training pairs that either are complete rule-based translations or reconstruction pairs from a noisy graph to a complete graph or a noisy text to a complete text. In contrast, composed noise can introduce translations from a noisy text to a complete graph or vice versa and thus encourage a system to omit input information (cf. Fig. 3). This difference is mirrored nicely in the results of our unsupervised systems for both tasks: composed noise performs better on VG where omitted information in an image caption is common and sampled noise works better on WebNLG where the texts describe their graphs completely.

7 Noise Ablation Study

Our unsupervised objectives are defined by different types of noise models. Hence, we examine their impact in a noise ablation study. Table 9 shows results for text→graph and graph→text on the validation splits of VG_{ball} and WebNLG.

For both datasets and tasks, introducing variation via noise functions is crucial for the success of unsupervised learning. The model without noise (i.e., $C(x) = x$) fails completely as do all models lacking `rule` as type of noise, the only exception being the only-`drop` system on WebNLG. Even though `drop` seems to work equally well in this one case, the simple translations delivered by our rule-based systems clearly provide the most useful information for the unsupervised models – notably in combination with the other noise functions: removing `rule` and keeping all other types of noise (cf. "sample all but `rule`" and "comp. all but `rule`") performs much worse than leaving out `drop`.

We hypothesize that our two rule systems provide two important pieces of information: (1) $\text{R}^{\text{graph}\rightarrow\text{text}}$ helps distinguish data format tokens from text tokens and (2) $\text{R}^{\text{text}\rightarrow\text{graph}}$ helps find probable candidate words in a text that form facts for the data output. As opposed to machine translation, where usually every word in a sentence is translated into a fluent sentence in the target language, identifying words that probably form a fact is more important in data-to/from-text generation.

We moreover observe that our unsupervised models always improve on the rule-based systems even when `rule` is the only type of noise: graph→text BLEU increases from 6.2/18.3 to 19.5/37.4 on VG_{ball}/WebNLG and text→graph F1 from 14.4/0.0 to 18.5/31.0.

Finally, our ablation study makes clear that there is no best noise model for all datasets and tasks. We therefore recommend experimenting with both different sets of noise functions and noise injection regimes (sampled vs. composed) for new data.

8 Conclusion

We presented the first fully unsupervised approach to text generation from KGs and a novel approach to unsupervised semantic parsing that automatically adapts to a target KG. We showed the effectiveness of our approach on two datasets, WebNLG v2.1 and a new text↔graph benchmark in the visual domain, derived from Visual Genome. We quantitatively and qualitatively analyzed our method on text↔graph conversion. We explored the impact of different unsupervised objectives in an ablation study and found that our newly introduced unsupervised objective using rule-based translations is essential for the success of unsupervised learning.

Acknowledgments

We thank the anonymous reviewers for their helpful comments and gratefully acknowledge a Ph.D. scholarship awarded to the first author by the German Academic Scholarship Foundation (Studienstiftung des deutschen Volkes). This work was supported by by the BMBF as part of the project MLWin (01IS18050).

References

Mark van Assem, Aldo Gangemi, and Guus Schreiber. 2006. Conversion of wordnet to a standard rdf/owl representation. In *Proceedings of the Fifth Edition of the International Conference on Language Resources and Evaluation (LREC 2006)*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *Computing Research Repository*, arXiv:1409.0473.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. Graph-to-sequence learning using gated graph neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 273–283, Melbourne, Australia. Association for Computational Linguistics.

Rajarshi Bhowmik and Gerard de Melo. 2018. Generating fine-grained open vocabulary entity type descriptions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 877–888, Melbourne, Australia. Association for Computational Linguistics.

Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc Le. 2017. Massive exploration of neural machine translation architectures. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark. Association for Computational Linguistics.

Thiago Castro Ferreira, Chris van der Lee, Emiel van Miltenburg, and Emiel Krahmer. 2019. Neural data-to-text generation: A comparison between pipeline and end-to-end architectures. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 552–562, Hong Kong, China. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Marco Damonte and Shay B. Cohen. 2019. Structural neural encoders for AMR-to-text generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3649–3658, Minneapolis, Minnesota. Association for Computational Linguistics.

Angela Fan, Claire Gardent, Chloé Braud, and Antoine Bordes. 2019. Using local knowledge graph construction to scale Seq2Seq models to multi-document inputs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4184–4194, Hong Kong, China. Association for Computational Linguistics.

Christiane Fellbaum. 2005. Wordnet and wordnets. In Keith Brown et al., editor, *Encyclopedia of Language and Linguistics*, second edition, pages 665–670. Elsevier, Oxford.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017a. Creating training corpora for NLG micro-planners. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 179–188, Vancouver, Canada. Association for Computational Linguistics.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017b. The WebNLG challenge: Generating text from RDF data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, Santiago de Compostela, Spain. Association for Computational Linguistics.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform. *Computing Research Repository*, arXiv:1803.07640.

Sebastian Gehrmann, Falcon Dai, Henry Elder, and Alexander Rush. 2018. End-to-end content and plan selection for data-to-text generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 46–56, Tilburg University, The Netherlands. Association for Computational Linguistics.

Dan Goldwasser, Roi Reichart, James Clarke, and Dan Roth. 2011. Confidence driven unsupervised semantic parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1486–1495, Portland, Oregon, USA. Association for Computational Linguistics.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1631–1640, Berlin, Germany. Association for Computational Linguistics.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *Computing Research Repository*, arXiv:1207.0580.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Aishwarya Kamath and Rajarshi Das. 2019. A survey on semantic parsing. In *Automated Knowledge Base Construction (AKBC)*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text Generation from Knowledge Graphs with Graph Transformers. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human*

Language Technologies, Minneapolis, Minnesota. Association for Computational Linguistics.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Fei-Fei Li. 2016. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *Computing Research Repository*, arXiv:1602.07332.

Karen Kukich. 1983. Design of a knowledge-based report generator. In *21st Annual Meeting of the Association for Computational Linguistics*, pages 145–150, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018a. Unsupervised machine translation using monolingual corpora only. In *International Conference on Learning Representations*.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018b. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium. Association for Computational Linguistics.

Tsung-Yi Lin, Xinlei Chen, Hao Fang, and Ramakrishna Vedantam. 2018. GitHub repository: tylin/coco-caption (Microsoft COCO caption evaluation). `https://github.com/tylin/coco-caption`.

Edward Loper and Steven Bird. 2004. Nltk: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.

Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. 2016. Visual relationship detection with language priors. In *European Conference on Computer Vision*.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium. Association for Computational Linguistics.

Diego Marcheggiani and Laura Perez-Beltrachini. 2018. Deep graph convolutional encoders for structured data to text generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 1–9, Tilburg University, The Netherlands. Association for Computational Linguistics.

Diego Marcheggiani and Ivan Titov. 2016. Discrete-state variational autoencoders for joint discovery and factorization of relations. *Transactions of the Association for Computational Linguistics*, 4:231–244.

Susan W. McRoy, Songsak Channarukul, and Syed S. Ali. 2000. YAG: A template-based generator for real-time systems. In *INLG'2000 Proceedings of the First International Conference on Natural Language Generation*, pages 264–267, Mitzpe Ramon, Israel. Association for Computational Linguistics.

Christina Niklaus, Matthias Cetto, André Freitas, and Siegfried Handschuh. 2018. A survey on open information extraction. In *Proceedings of the 27th International Conference on Computational Linguistics*, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

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

Ankur P. Parikh, Hoifung Poon, and Kristina Toutanova. 2015. Grounded semantic parsing for complex knowledge extraction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 756–766, Denver, Colorado. Association for Computational Linguistics.

Hoifung Poon. 2013. Grounded unsupervised semantic parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 933–943, Sofia, Bulgaria. Association for Computational Linguistics.

Hoifung Poon and Pedro Domingos. 2009. Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1–10, Singapore. Association for Computational Linguistics.

Ana-Maria Popescu, Alex Armanasu, Oren Etzioni, David Ko, and Alexander Yates. 2004. Modern natural language interfaces to databases: Composing statistical parsing with semantic tractability. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 141–147, Geneva, Switzerland. COLING.

Maja Popović. 2017a. chrF++: words helping character n-grams. In *Proceedings of the Second Conference on Machine Translation*, pages 612–618, Copenhagen, Denmark. Association for Computational Linguistics.

Maja Popović. 2017b. GitHub repository: m-popovic/chrf (chrF). `https://github.com/m-popovic/chrF`.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019. Data-to-Text Generation with Content Selection and Planning. In *Proceedings of the 33rd Conference on Artificial Intelligence*.

Leonardo F. R. Ribeiro, Claire Gardent, and Iryna Gurevych. 2019. Enhancing AMR-to-text generation with dual graph representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3181–3192, Hong Kong, China. Association for Computational Linguistics.

Leonardo F. R. Ribeiro, Yue Zhang, Claire Gardent, and Iryna Gurevych. 2020. Modeling global and local node contexts for text generation from knowledge graphs. *Computing Research Repository*, arXiv:2001.11003.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Machine Learning and Knowledge Discovery in Databases*, pages 148–163, Berlin, Heidelberg. Springer Berlin Heidelberg.

Jacques Pierre Robin. 1995. *Revision-based Generation of Natural Language Summaries Providing Historical Background: Corpus-based Analysis, Design, Implementation and Evaluation*. Ph.D. thesis, Columbia University, New York, NY, USA. UMI Order No. GAX95-33653.

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

Sebastian Schuster, Ranjay Krishna, Angel Chang, Li Fei-Fei, and Christopher D. Manning. 2015. Generating semantically precise scene graphs from textual descriptions for improved image retrieval. In *Proceedings of the Fourth Workshop on Vision and Language*, pages 70–80, Lisbon, Portugal. Association for Computational Linguistics.

Anastasia Shimorina and Claire Gardent. 2018. Handling rare items in data-to-text generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 360–370, Tilburg University, The Netherlands. Association for Computational Linguistics.

Étienne Simon, Vincent Guigue, and Benjamin Piwowarski. 2019. Unsupervised information extraction: Regularizing discriminative approaches with relation distribution losses. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1378–1387, Florence, Italy. Association for Computational Linguistics.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, pages 4444–4451. AAAI Press.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Kumiko Tanaka-Ishii, Koiti Hasida, and Itsuki Noda. 1998. Reactive content selection in the generation of real-time soccer commentary. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 2*, pages 1282–1288, Montreal, Quebec, Canada. Association for Computational Linguistics.

Bayu Distiawan Trisedya, Jianzhong Qi, Rui Zhang, and Wei Wang. 2018. GTR-LSTM: A triple encoder for sentence generation from RDF data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1627–1637, Melbourne, Australia. Association for Computational Linguistics.

Frank Van Harmelen, Vladimir Lifschitz, and Bruce Porter. 2008. *Handbook of knowledge representation*, volume 1. Elsevier.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations*.

Hai Wan, Yonghao Luo, Bo Peng, and Wei-Shi Zheng. 2018. Representation learning for scene graph completion via jointly structural and visual embedding. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 949–956. International Joint Conferences on Artificial Intelligence Organization.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, Copenhagen, Denmark. Association for Computational Linguistics.

David S Wishart, Yannick D Feunang, An C Guo, Elvis J Lo, Ana Marcu, Jason R Grant, Tanvir Sajed, Daniel Johnson, Carin Li, Zinat Sayeeda, Nazanin Assempour, Ithayavani Iynkkaran, Yifeng Liu, Adam Maciejewski, Nicola Gale, Alex Wilson, Lucy Chin, Ryan Cummings, Diana Le, Allison Pon, Craig Knox, and Michael Wilson. 2018. DrugBank 5.0: a major update to the DrugBank database for 2018. *Nucleic Acids Research*, 46(D1):D1074–D1082.

Limin Yao, Aria Haghighi, Sebastian Riedel, and Andrew McCallum. 2011. Structured relation discovery using generative models. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1456–1466, Edinburgh, Scotland, UK. Association for Computational Linguistics.

A Hyperparameters

We use the following settings for all our experiments: learning rate of 10^{-4}, word embeddings of size 300, an LSTM hidden size of 250, a dropout rate of 0.2 and a batch size of 10. Following Lample et al. (2018b), we set $p_{\text{blank}} = p_{\text{repeat}} = 0.2$, $p_{\text{drop}} = 0.1$. For inference, we decode greedily with a maximum number of 40 decoding steps. To speed up unsupervised learning, we increase the batch size to 64 when creating backtranslations.

B Model details

We train with homogeneous batches of one target output type (text or graph) at a time. We use a single GeForce GTX 1080 GPU for training and inference. In this environment, pure training takes approximately 9 ms per instance and inference, which also means backtranslation, takes approximately 21 ms per instance. This means that unsupervised learning approximately needs 30 ms per instance. WebNLG models use 10.6 million parameters, VG models have 60.7 million parameters. The difference is due to a larger vocabulary size of 70,800 for VG compared to 8,171 for WebNLG.

C Results of all iterations on WebNLG

See Table 10 for all intermediate graph→text results of unsupervised training on WebNLG and Table 11 for text→graph. We find similar trends as for VG (Tables 4 and 6) except for $\mathcal{U}$ being a less reliable performance indicator for text→graph in the sampled noise setting.

	sampled noise			composed noise		
#	$\mathcal{U}$	$\mathcal{V}_{100}$	val	$\mathcal{U}$	$\mathcal{V}_{100}$	val
1	91.7	12.8	13.0	23.0	15.9	15.5
2	**94.0**	14.7	15.8	53.2	22.2	20.7
3	85.2	25.5	26.0	71.0	23.2	22.8
4	65.9	27.7	28.8	75.2	25.3	26.2
5	65.5	31.4	30.7	69.2	25.9	27.2
6	58.1	31.5	31.0	71.5	27.6	27.7
7	48.0	31.3	32.3	**79.2**	29.0	27.7
8	48.3	32.8	33.4	52.5	28.1	27.5
9	37.5	33.2	34.0	57.1	30.5	30.0
10	42.1	32.8	33.4	52.4	30.6	29.9
11	38.7	34.7	34.8	59.9	32.0	**31.6**
12	38.7	36.4	36.2	42.1	30.4	30.8
13	39.3	33.5	35.1	50.0	30.7	30.7
14	40.5	36.9	36.6	46.7	30.9	30.7
15	41.8	36.5	37.5	48.2	31.1	30.3
16	43.2	36.9	38.0	43.7	30.3	29.6
17	39.1	35.6	36.6	43.1	29.0	29.7
18	38.5	37.5	38.3	31.1	29.7	29.8
19	38.8	37.8	38.4	39.5	29.0	29.8
20	37.5	37.2	38.6	36.2	31.3	29.8
21	36.4	36.8	38.4	35.2	30.0	30.8
22	44.8	36.3	39.7	37.6	32.4	30.7
23	40.8	35.8	38.2	39.6	31.4	30.3
24	35.8	39.2	39.6	39.6	32.4	30.3
25	40.6	38.5	39.5	37.0	33.2	30.9
26	36.8	38.9	40.3	41.3	32.3	30.2
27	44.1	39.7	**40.6**	37.3	33.0	30.4
28	39.3	36.9	38.9	39.0	**34.7**	30.8
29	36.1	37.6	38.6	41.5	31.0	30.6
30	38.7	**40.7**	39.1	42.9	30.6	30.0

Table 10: BLEU scores on WebNLG for our unsupervised models evaluated for graph→text at different iterations. $\mathcal{U}$ is calculated on all unlabeled data used for training. $\mathcal{V}_{100}$ is a 100-size random sample from val. All results are computed with scripts from (Lin et al., 2018).

	sampled noise			composed noise		
#	$\mathcal{U}$	$\mathcal{V}_{100}$	val	$\mathcal{U}$	$\mathcal{V}_{100}$	val
1	**69.4**	0.0	0.0	0.0	0.0	0.0
2	64.0	0.0	0.1	16.2	1.2	1.6
3	35.6	0.9	0.3	7.5	3.3	3.0
4	47.8	2.6	2.3	37.5	5.5	5.5
5	39.2	5.7	3.4	35.3	7.0	6.6
6	39.2	6.2	5.6	44.9	9.7	8.0
7	45.8	9.8	7.9	58.3	8.0	10.3
8	50.0	12.6	10.0	51.1	14.0	12.8
9	54.9	13.6	12.9	53.1	12.5	14.0
10	58.3	14.9	14.3	51.1	15.9	16.8
11	62.5	19.3	17.8	53.8	15.6	17.3
12	54.2	20.3	18.2	58.3	16.7	18.0
13	57.1	23.1	20.2	47.8	19.8	20.6
14	37.5	25.5	21.4	49.0	20.6	22.1
15	48.0	25.7	22.4	54.2	23.0	22.8
16	52.0	27.9	24.3	46.2	22.5	25.4
17	50.0	26.7	25.1	35.6	26.8	26.8
18	48.0	32.1	27.7	52.2	27.8	27.7
19	56.0	32.3	28.9	58.3	26.4	28.1
20	60.0	31.0	30.1	55.3	26.4	29.2
21	51.0	32.3	30.4	59.3	27.6	30.7
22	55.3	34.9	32.0	**62.5**	31.7	32.0
23	44.9	34.3	32.7	54.9	34.0	32.6
24	58.8	38.4	33.7	61.2	31.5	32.4
25	46.8	39.6	34.1	58.3	33.3	33.1
26	53.8	40.6	36.3	54.2	**34.4**	32.5
27	62.5	41.8	36.4	50.0	33.9	33.3
28	55.3	41.0	37.4	40.8	32.6	**33.7**
29	56.0	40.7	37.0	58.8	29.5	**33.7**
30	59.6	**41.9**	**38.5**	53.8	31.6	33.4

Table 11: F1 scores on WebNLG for our unsupervised models evaluated for text→graph at different iterations. $\mathcal{U}$ is calculated on all unlabeled data used for training. $\mathcal{V}_{100}$ is a 100-size random sample from val.