Enhancing AMR-to-Text Generation with Dual Graph Representations

Leonardo F R Ribeiro, Claire Gardent, Iryna Gurevych

Enhancing AMR-to-Text Generation with Dual Graph Representations

Leonardo F. R. Ribeiro[†], **Claire Gardent**[‡] **and Iryna Gurevych**[†]

[†]Research Training Group AIPHES and UKP Lab, Technische Universität Darmstadt
www.ukp.tu-darmstadt.de

[‡]CNRS/LORIA, Nancy, France
claire.gardent@loria.fr

Abstract

Generating text from graph-based data, such as Abstract Meaning Representation (AMR), is a challenging task due to the inherent difficulty in how to properly encode the structure of a graph with labeled edges. To address this difficulty, we propose a novel graph-to-sequence model that encodes different but complementary perspectives of the structural information contained in the AMR graph. The model learns parallel top-down and bottom-up representations of nodes capturing contrasting views of the graph. We also investigate the use of different node message passing strategies, employing different state-of-the-art graph encoders to compute node representations based on incoming and outgoing perspectives. In our experiments, we demonstrate that the dual graph representation leads to improvements in AMR-to-text generation, achieving state-of-the-art results on two AMR datasets[1].

1 Introduction

Abstract Meaning Representation (AMR; Banarescu et al. (2013)) is a linguistically-grounded semantic formalism that represents the meaning of a sentence as a rooted directed graph, where nodes are concepts and edges are semantic relations. As AMR abstracts away from surface word strings and syntactic structure producing a language neutral representation of meaning, its usage is beneficial in many semantic related NLP tasks, including text summarization (Liao et al., 2018) and machine translation (Song et al., 2019).

The purpose of AMR-to-text generation is to produce a text which verbalises the meaning encoded by an input AMR graph. This is a challenging task as capturing the complex structural information stored in graph-based data is not trivial, as these are non-Euclidean structures, which implies that properties such as global parametrization, vector space structure, or shift-invariance do not hold (Bronstein et al., 2017). Recently, Graph Neural Networks (GNNs) have emerged as a powerful class of methods for learning effective graph latent representations (Xu et al., 2019) and graph-to-sequence models have been applied to the task of AMR-to-text generation (Song et al., 2018; Beck et al., 2018; Damonte and Cohen, 2019; Guo et al., 2019).

In this paper, we propose a novel graph-to-sequence approach to AMR-to-text generation, which is inspired by pre-neural generation algorithms. These approaches explored alternative (top-down, bottom-up and mixed) traversals of the input graph and showed that a hybrid traversal combining both top-down (TD) and bottom-up (BU) information was best as this permits integrating both global constraints top-down from the input and local constraints bottom-up from the semantic heads (Shieber et al., 1990; Narayan and Gardent, 2012).

Similarly, we present an approach where the input graph is represented by two separate structures, each representing a different view of the graph. The nodes of these two structures are encoded using separate graph encoders so that each concept and relation in the input graph is assigned both a TD and a BU representation.

Our approach markedly differs from existing graph-to-sequence models for MR-to-Text generation (Marcheggiani and Perez Beltrachini, 2018; Beck et al., 2018; Damonte and Cohen, 2019) in that these approaches aggregate all the immediate neighborhood information of a node in a single representation. By exploiting parallel and complementary vector representations of the AMR graph, our approach eases the burden on the neural model in encoding nodes (concepts) and edges (relations) in a single vector representation. It also elimi-

[1]Code is available at https://github.com/UKPLab/emnlp2019-dualgraph

nates the need for additional positional information (Beck et al., 2018) which is required when the same graph is used to encode both TD and BU information, thereby making the edges undirected.

Our main contributions are the following:

- We present a novel architecture for AMR-to-text generation which explicitly encodes two separate TD and BU views of the input graph.
- We show that our approach outperforms recent AMR-to-text generation models on two datasets, including a model that leverages additional syntactic information (Cao and Clark, 2019).
- We compare the performance of three graph encoders, which have not been studied so far for AMR-to-text generation.

2 Related Work

Early works on AMR-to-text generation employ statistical methods (Flanigan et al., 2016b; Pourdamghani et al., 2016; Castro Ferreira et al., 2017) and apply linearization of the graph by means of a depth-first traversal.

Recent neural approaches have exhibited success by linearising the input graph and using a sequence-to-sequence architecture. Konstas et al. (2017) achieve promising results on this task. However, they strongly rely on named entities anonymisation. Anonymisation requires an ad hoc procedure for each new corpus. The matching procedure needs to match a rare input item correctly (e.g., "United States of America") with the corresponding part in the output text (e.g., "USA") which may be challenging and may result in incorrect or incomplete delexicalisations. In contrast, our approach omits anonymisation. Instead, we use a copy mechanism (See et al., 2017), a generic technique which is easy to integrate in the encoder-decoder framework and can be used independently of the particular domain and application. Our approach further differs from Konstas et al. (2017) in that we build a dual TD/BU graph representation and use graph encoders to represent nodes.

Cao and Clark (2019) factor the generation process leveraging syntactic information to improve the performance. However, they linearize both AMR and constituency graphs, which implies that important parts of the graphs cannot well be represented (e.g., coreference).

Several graph-to-sequence models have been proposed. Marcheggiani and Perez Beltrachini (2018) show that explicitly encoding the structure of the graph is beneficial with respect to sequential encoding. They evaluate their model on two tasks, WebNLG (Gardent et al., 2017) and SR11Deep (Belz et al., 2011), but do not apply it to AMR benchmarks. Song et al. (2018) and Beck et al. (2018) apply recurrent neural networks to directly encode AMR graphs. Song et al. (2018) use a graph LSTM as the graph encoder, whereas Beck et al. (2018) develop a model based on GRUs. We go a step further in that direction by developing parallel encodings of graphs which are able to highlight different graph properties.

In a related task, Koncel-Kedziorski et al. (2019) propose an attention-based graph model that generates sentences from knowledge graphs. Schlichtkrull et al. (2018) use Graph Convolutional Networks (GCNs) to tackle the tasks of link prediction and entity classification on knowledge graphs.

Damonte and Cohen (2019) show that off-the-shelf GCNs cannot achieve good performance for AMR-to-text generation. To tackle this issue, Guo et al. (2019) introduce dense connectivity to GNNs in order to integrate both local and global features, achieving good results on the task. Our work is related to Damonte and Cohen (2019), that use stacking of GCN and LSTM layers to improve the model capacity and employ anonymization. However, our model is substantially different: (i) we learn dual representations capturing top-down and bottom-up adjuvant views of the graph, (ii) we employ more effective graph encoders (with different neighborhood aggregations) than GCNs and (iii) we employ copy and coverage mechanisms and do not resort to entity anonymization.

3 Graph-to-Sequence Model

In this section, we describe (i) the representations of the graph adopted as inputs, (ii) the model architecture, including the Dual Graph Encoder and (iii) the GNNs employed as graph encoders.

3.1 Graph Preparation

Let $G = (V, E, R)$ denote a rooted and directed AMR graph with nodes $v_i \in V$ and labeled edges $(v_i, r, v_j) \in E$, where $r \in R$ is a relation type.

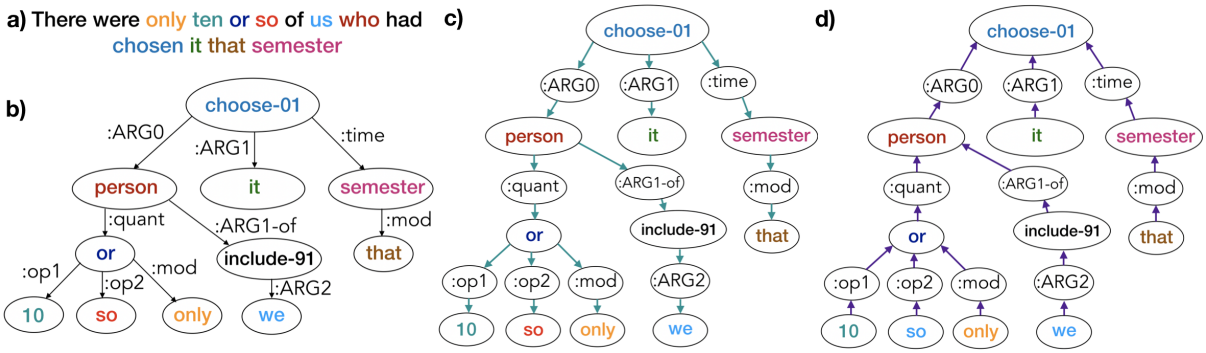

Figure 1: (a) an example sentence, (b) its original AMR graph (G) and different graph perspectives: (c) top-down (G_t) and (d) bottom-up (G_b).

Let $n = |V|$ and $m = |E|$ denote the numbers of nodes and edges, respectively.

We convert each AMR graph into an unlabeled and connected bipartite graph $G_t = (V_t, E_t)$, transforming each labeled edge $(v_i, r, v_j) \in E$ into two unlabeled edges $(v_i, r), (r, v_j) \in E_t$, with $|V_t| = n + m$ and $|E_t| = 2m$. This process, called Levi Transformation (Beck et al., 2018), turns original edges into nodes creating an unlabeled graph. For instance, the edge between `semester` and `that` with label `:mod` in Figure 1(b) is replaced by two edges and one node in 1(c): an edge between `semester`, and the new node `:mod` and another one between `:mod` and `that`. The new graph allows us to directly represent the relationships between nodes using embeddings. This enables us to encode label edge information using distinct message passing schemes employing different GNNs.

G_t captures a TD view of the graph. We also create a BU view of the graph $G_b = (V_t, E_b)$, where each directed edge $e_k = (v_i, v_j) \in E_t$ becomes $e_k = (v_j, v_i) \in E_b$, that is, we reverse the direction of original edges. An example of a sentence, its AMR graph and the two new graphs G_t and G_b is shown in Figure 1.

3.2 Dual Graph Encoder

We represent each node $v_i \in V_t$ with a node embedding $\mathbf{e}_i \in \mathbb{R}^d$, generated from the node label. In order to explicitly encode structural information, our encoder starts with two graph encoders, denoted by GE_t and GE_b, that compute representations for nodes in G_t and G_b, respectively.

Each GE learns node representations based on the specific view of its particular graph, G_t or G_b. Since G_t and G_b capture distinct perspectives of the graph structure, the information flow is propagated throughout TD and BU directions, respectively. In particular, for each node v_i, the GE receives the node embeddings of v_i and its neighbors, and computes its node representation:

$$\mathbf{h}_i^t = GE_t(\{\mathbf{e}_i, \mathbf{e}_j : j \in \mathcal{N}_t(i)\}),$$
$$\mathbf{h}_i^b = GE_b(\{\mathbf{e}_i, \mathbf{e}_j : j \in \mathcal{N}_b(i)\}),$$

where $\mathcal{N}_t(i)$ and $\mathcal{N}_b(i)$ are the immediate incoming neighborhoods of v_i in G_t and G_b, respectively.

Each node v_i is represented by two different hidden states, $\mathbf{h}_i^t$ and $\mathbf{h}_i^b$. Note that we learn two representations per relation and node of the original AMR graph. The hidden states $\mathbf{h}_i^t$ and $\mathbf{h}_i^b$, and embedding $\mathbf{e}_i$ contain different information regarding v_i. We concatenate them building a final node representation:

$$\mathbf{r}_i = \left[\, \mathbf{h}_i^t \,\|\, \mathbf{h}_i^b \,\|\, \mathbf{e}_i \,\right].$$

This approach is similar to bidirectional RNNs (Schuster and Paliwal, 1997). Bidirectional RNNs benefit from left-to-right and right-to-left propagation. They learn the hidden representations separately and concatenate them at the end. We perform a similar encoding: first we learn TD and BU representations independently, and lastly, we concatenate them.

The final representation $\mathbf{r}_i$ is employed in a sequence input of a bidirectional LSTM. For each AMR graph, we generate a node sequence by depth-first traversal order. In particular, given a representation sequence from $\mathbf{r}_1$ to $\mathbf{r}_n$, the hidden forward and backward states of $\mathbf{r}_i$ are defined as:

$$\overrightarrow{\mathbf{h}}_i = LSTM_f(\mathbf{r}_i, \overrightarrow{\mathbf{h}}_{i-1}),$$
$$\overleftarrow{\mathbf{h}}_i = LSTM_b(\mathbf{r}_i, \overleftarrow{\mathbf{h}}_{i-1}),$$

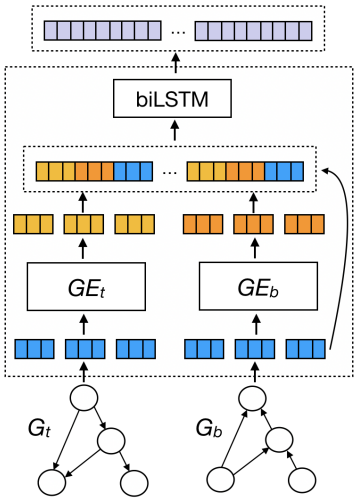

Figure 2: Dual Graph Encoder. The encoder receives the two graph views and generates structural node representations that are used by the decoder. Representations in blue, yellow and orange are $\mathbf{e}_i$, $\mathbf{h}_i^t$ and $\mathbf{h}_i^b$, respectively.

where $LSTM_f$ is a forward LSTM and $LSTM_b$ is a backward LSTM. Note that, for the backward LSTM, we feed the reversed input as the order from $\mathbf{r}_n$ to $\mathbf{r}_1$. Lastly, we obtain the final hidden state by concatenating them as:

$$\mathbf{h}_i = [\overrightarrow{\mathbf{h}}_i \,\|\, \overleftarrow{\mathbf{h}}_i].$$

The resulting hidden state $\mathbf{h}_i$ encodes the information of both preceding and following nodes.

Stacking layers was demonstrated to be effective in graph-to-sequence approaches (Marcheggiani and Perez Beltrachini, 2018; Koncel-Kedziorski et al., 2019; Damonte and Cohen, 2019) and allows us to test for their contributions to the system performance more easily. We employ different GNNs for both graph encoders (Section 3.3). Figure 2 shows the proposed encoder architecture.

3.3 Graph Neural Networks

The GEs incorporate, in each node representation, structural information based on both views of the graph. We explore distinct strategies for neighborhood aggregation, adopting three GNNs: Gated Graph Neural Networks (GGNN, Li et al. (2016)), Graph Attention Networks (GAT, Veličković et al. (2018)) and Graph Isomorphic Networks (GIN, Xu et al. (2019)). Each GNN employs a specific message passing scheme which allows capturing different nuances of structural information.

Gated Graph Neural Networks GGNNs employ gated recurrent units to encode node representations, reducing the recurrence to a fixed number of steps. In particular, the l-th layer of a GGNN is calculated as:

$$\mathbf{h}_i^{(l)} = GRU\left(\mathbf{h}_i^{(l-1)}, \sum_{j \in \mathcal{N}(i)} \mathbf{W_1} \mathbf{h}_j^{(l-1)}\right),$$

where $\mathcal{N}(i)$ is the immediate neighborhood of v_i, $\mathbf{W_1}$ is a parameter and GRU is a gated recurrent unit (Cho et al., 2014). Different from other GNNs, GGNNs use back-propagation through time (BPTT) to learn the parameters. GGNNs also do not require to constrain parameters to ensure convergence.

Graph Attention Networks GATs apply attentive mechanisms to improve the exploitation of non-trivial graph structure. They encode node representations by attending over their neighbors, following a self-attention strategy:

$$\mathbf{h}_i^{(l)} = \alpha_{i,i} \mathbf{W_2} \mathbf{h}_i^{(l-1)} + \sum_{j \in \mathcal{N}(i)} \alpha_{i,j} \mathbf{W_2} \mathbf{h}_j^{(l-1)},$$

where attention coefficients $\alpha_{i,j}$ are computed as:

$$\alpha_{i,j} = \text{softmax}\left(\sigma\left(\mathbf{a}^\top [\mathbf{W_2}\mathbf{h}_i^{(l-1)} \,\|\, \mathbf{W_2}\mathbf{h}_j^{(l-1)}]\right)\right),$$

where σ is the activation function and $\|$ denotes concatenation. $\mathbf{W_2}$ and $\mathbf{a}$ are model parameters. The virtue of the attention mechanism is its ability to focus on the most important parts of the node neighborhood. In order to learn attention weights in different perspectives, GATs can employ multi-head attentions.

Graph Isomorphic Networks GIN is a GNN as powerful as the Weisfeiler-Lehman (WL) graph isomorphism test (Weisfeiler and Lehman, 1968) in representing isomorphic and non-isomorphic graphs with discrete attributes. Its l-th layer is defined as:

$$\mathbf{h}_i^{(l)} = h_{\mathbf{W}}\left(\mathbf{h}_i^{(l-1)} + \sum_{j \in \mathcal{N}(i)} \mathbf{h}_j^{(l-1)}\right),$$

where $h_{\mathbf{W}}$ is a multi-layer perceptron (MLP). In contrast to other GNNs, which combine node feature with its aggregated neighborhood feature, GINs do not apply the combination step and simply aggregate the node along with its neighbors.

Each of these GNNs applies different approaches to learn structural features from graph data and has achieved impressive results on many graph-based tasks (Li et al., 2016; Veličković et al., 2018; Xu et al., 2019).

	LDC2015E86			LDC2017T10		
training, dev and test instances	16,833	1,368	1,371	36,521	1,368	1,371
min, average and max graph diameter	0	6.9	20	0	6.7	20
min, average and max node degree	0	2.1	18	0	2.1	20
min, average and max number of nodes	1	17.7	151	1	16.8	151
min, average and max number of edges	0	18.6	172	0	17.7	172
number of DAG and non-DAG graphs	18,679	893		37,284	1,976	
min, average and max length sentences	1	21.3	225	1	20.4	225

Table 1: Data statistics of LDC2015E86 and LDC2017T10 datasets. The values are calculated for all splits (train, development and test sets). DAG stands for directed acyclic graph.

3.4 Decoder

An attention-based unidirectional LSTM decoder is used to generate sentences, attending to the hidden representations of edges and nodes. In each step t, the decoder receives the word embedding of the previous word (during training, this is the previous word of the reference sentence; at test time it is the previously generated word), and has the decoder state $\mathbf{s}_t$. The attention distribution $\mathbf{a}^t$ is calculated as in See et al. (2017):

$$\mathbf{e}_i^t = \mathbf{v} \cdot \tanh(\mathbf{W}_h \mathbf{h}_i + \mathbf{W}_s \mathbf{s}_t + \mathbf{w}_c \mathbf{s}_c + \mathbf{b}),$$
$$\mathbf{a}^t = \text{softmax}(\mathbf{e}^t),$$

where $\mathbf{s}_c$ is the coverage vector and $\mathbf{v}$, $\mathbf{W}_h$, $\mathbf{W}_s$, $\mathbf{w}_c$ and $\mathbf{b}$ are learnable parameters. The coverage vector is the accumulation of all attention distributions so far.

Copy and Coverage Mechanisms Previous works (Damonte and Cohen, 2019; Cao and Clark, 2019) use anonymization to handle names and rare words, alleviating the data sparsity. In contrast, we employ copy and coverage mechanisms to address out-of-vocabulary issues for rare target words and to avoid repetition (See et al., 2017).

The model is trained to optimize the negative log-likelihood:

$$\mathcal{L} = -\sum_{t=1}^{|Y|} \log\, p(y_t | y_{1:t-1}, X; \theta),$$

where $Y = y_1, \ldots, y_{|Y|}$ is the sentence, X is the AMR graph and θ represents the model parameters.

4 Data

We use two AMR corpora, LDC2015E86 and LDC2017T10[2]. In these datasets, each instance contains an AMR graph and a sentence. Table 1 shows the statistics for both datasets. Figure 3 shows the distribution of the AMR graph diameters and node degrees for both datasets. The AMR graph structures are similar for most examples. Note that 90% of AMR graphs in both datasets have the diameter less than or equal to 11 and 90% of nodes have the degree of 4 or less. Very structurally similar graphs pose difficulty for the graph encoder by making it harder to learn the differences between their similar structures. Therefore, the word embeddings used as additional input play an important role in helping the model to deal with language information. That is one of the reasons why we concatenate this information in the node representation $\mathbf{r}_i$.

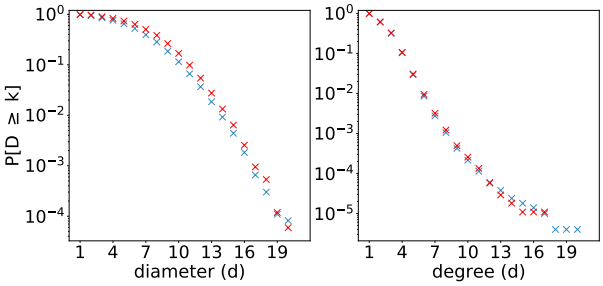

Figure 3: Distribution of the AMR graph diameter (left) and node degree (right) in the training set for LDC2015E86 (red) and LDC2017T10 (blue) datasets.

5 Experiments and Discussion

Implementation Details We extract vocabularies (size of 20,000) from the training sets and initialize the node embeddings from GloVe word embeddings (Pennington et al., 2014) on Common Crawl. Hyperparameters are tuned on the development set of the LDC2015E86 dataset. For GIN, GAT, and GGNN graph encoders, we set the number of layers to 2, 5 and 5, respectively. To regu-

[2]The datasets can be found at https://amr.isi.edu/download.html

Model	BLEU	METEOR
LDC2015E86		
Konstas et al. (2017)	22.00	-
Song et al. (2018)	23.28	30.10
Cao et al. (2019)	23.50	-
Damonte et al.(2019)	24.40	23.60
Guo et al. (2019)	**25.70**	-
S2S	22.55 ± 0.17	29.90 ± 0.31
G2S-GIN	22.93 ± 0.20	29.72 ± 0.09
G2S-GAT	23.42 ± 0.16	29.87 ± 0.14
G2S-GGNN	24.32 ± 0.16	**30.53** ± 0.30
LDC2017T10		
Back et al. (2018)	23.30	-
Song et al. (2018)	24.86	31.56
Damonte et al.(2019)	24.54	24.07
Cao et al. (2019)	26.80	-
Guo et al. (2019)	27.60	-
S2S	22.73 ± 0.18	30.15 ± 0.14
G2S-GIN	26.90 ± 0.19	32.62 ± 0.04
G2S-GAT	26.72 ± 0.20	32.52 ± 0.02
G2S-GGNN	**27.87** ± 0.15	**33.21** ± 0.15

Table 2: BLEU and METEOR scores on the test set of LDC2015E86 and LDC2017T10 datasets.

larize the model, during training we apply dropout (Srivastava et al., 2014) to the graph layers with a rate of 0.3. The graph encoder hidden vector sizes are set to 300 and hidden vector sizes for LSTMs are set to 900.

The models are trained for 30 epochs with early stopping based on the development BLEU score. For our models and the baseline, we used a two-layer LSTM decoder. We use Adam optimization (Kingma and Ba, 2015) as the optimizer with an initial learning rate of 0.001 and 20 as the batch size. Beam search with the beam size of 5 is used for decoding.

Results We call the models G2S-GIN (isomorphic encoder), G2S-GAT (graph-attention encoder), and G2S-GGNN (gated-graph encoder), according to the graph encoder utilized. As a baseline (S2S), we train an attention-based encoder-decoder model with copy and coverage mechanisms, and use a linearized version of the graph generated by depth-first traversal order as input. We compare our models against several state-of-the-art results reported on the two datasets (Konstas et al., 2017; Song et al., 2018; Beck et al., 2018; Damonte and Cohen, 2019; Cao and Clark, 2019; Guo et al., 2019).

Model	External	BLEU
Konstas et al. (2017)	200K	27.40
Song et al. (2018)	200K	28.20
Guo et al. (2019)	200K	31.60
G2S-GGNN	200K	**32.23**

Table 3: Results on LDC2015E86 test set when models are trained with additional Gigaword data.

We use both BLEU (Papineni et al., 2002) and METEOR (Denkowski and Lavie, 2014) as evaluation metrics[3]. In order to mitigate the effects of random seeds, we report the averages for 4 training runs of each model along with their standard deviation. Table 2 shows the comparison between the proposed models, the baseline and other neural models on the test set of the two datasets.

For both datasets, our approach substantially outperforms the baselines. In LDC2015E86, G2S-GGNN achieves a BLEU score of 24.32, 4.46% higher than Song et al. (2018), who also use the copy mechanism. This indicates that our architecture can learn to generate better signals for text generation. On the same dataset, we have competitive results to Damonte and Cohen (2019). However, we do not rely on preprocessing anonymisation not to lose semantic signals. In LDC2017T10, G2S-GGNN achieves a BLEU score of 27.87, which is 3.33 points higher than Damonte and Cohen (2019), a state-of-the-art model that does not employ external information. We also have competitive results to Guo et al. (2019), a very recent state-of-the-art model.

We also outperform Cao and Clark (2019) improving BLEU scores by 3.48% and 4.00%, in LDC2015E86 and LDC2017T10, respectively. In contrast to their work, we do not rely on (i) leveraging supplementary syntactic information and (ii) we do not require an anonymization preprocessing step. G2S-GIN and G2S-GAT have comparable performance on both datasets. Interestingly, G2S-GGNN has better performance among our models. This suggests that graph encoders based on gating mechanisms are very effective in text generation models. We hypothesize that the gating mechanism can better capture long-distance dependencies between nodes far apart in the graph.

[3]For BLEU, we use the multi-BLEU script from the MOSES decoder suite (Koehn et al., 2007). For METEOR, we use the original meteor-1.5.jar script (https://github.com/cmu-mtlab/meteor).

Model	BLEU	METEOR	Size
biLSTM	22.50	30.42	57.6M
GE_t + biLSTM	26.33	32.62	59.6M
GE_b + biLSTM	26.12	32.49	59.6M
GE_t + GE_b + biLSTM	27.37	33.30	61.7M

Table 4: Results of the ablation study on the LDC2017T10 development set.

Additional Training Data Following previous works (Konstas et al., 2017; Song et al., 2018; Guo et al., 2019), we also evaluate our models employing additional data from English Gigaword corpus (Napoles et al., 2012). We sample 200K Gigaword sentences and use JAMR[4] (Flanigan et al., 2016a) to parse them. We follow the method of Konstas et al. (2017), which is fine-tuning the model on the LDC2015E86 training set after every epoch of pretraining on the Gigaword data. G2S-GGNN outperforms others with the same amount of Gigaword sentences (200K), achieving a 32.23 BLEU score, as shown in Table 3. The results demonstrate that pretraining on automatically generated AMR graphs enhances the performance of our model.

Ablation Study In Table 4, we report the results of an ablation study on the impact of each component of our model on the development set of LDC2017T10 dataset by removing the graph encoders. We also report the number of parameters (including embeddings) used in each model. The first thing we notice is the huge increase in metric scores (17% in BLEU) when applying the graph encoder layer, as the neural model receives signals regarding the graph structure of the input. The dual representation helps the model with a different view of the graph, increasing BLEU and METEOR scores by 1.04 and 0.68 points, respectively. The complete model has slightly more parameters than the model without graph encoders (57.6M vs 61.7M).

Impact of Graph Size, Arity and Sentence Length The good overall performance on the datasets shows the superiority of using graph encoders and dual representations over the sequential encoder. However, we are also interested in estimating the performance of the models concerning different data properties. In order to evaluate how the models handle graph and sentence features, we perform an inspection based on different sizes of graph diameter, sentence length, and max node out-degree. Table 5 shows METEOR[5] scores for the LDC2017T10 dataset.

Model	Graph Diameter					
	0-7	Δ	7-13	Δ	14-20	Δ
S2S	33.2		29.7		28.8	
G2S-GIN	35.2	+6.0%	31.8	+7.4%	31.5	+9.2%
G2S-GAT	35.1	+5.9%	32.0	+7.8%	31.5	+9.51%
G2S-GGNN	36.2	+9.0%	33.0	+11.4%	30.7	+6.7%
	Sentence Length					
	0-20	Δ	20-50	Δ	50-240	Δ
S2S	34.9		29.9		25.1	
G2S-GIN	36.7	+5.2%	32.2	+7.8%	26.5	+5.8%
G2S-GAT	36.9	+5.7%	32.3	+7.9%	26.6	+6.1%
G2S-GGNN	37.9	+8.5%	33.3	+11.2%	26.9	+6.8%
	Max Node Out-degree					
	0-3	Δ	4-8	Δ	9-18	Δ
S2S	31.7		30.0		23.9	
G2S-GIN	33.9	+6.9%	32.1	+6.9%	25.4	+6.2%
G2S-GAT	34.3	+8.0%	32.0	+6.7%	22.5	-6.0%
G2S-GGNN	35.0	+10.3%	33.1	+10.4%	22.2	-7.3%

Table 5: METEOR scores and differences to the S2S, in the LDC2017T10 test set, with respect to the graph diameter, sentence length and max node out-degree.

The performances of all models decrease as the diameters of the graphs increase. G2S-GGNN has a 17.9% higher METEOR score in graphs with a diameter of at most 7 compared to graphs with diameters higher than 13. This is expected as encoding a bigger graph (containing more information) is harder than encoding smaller graphs. Moreover, 71% of the graphs in the training set have a diameter less than or equal to 7 and only 2% have a diameter bigger than 13 (see Figure 3). Since the models have fewer examples of bigger graphs to learn from, this also leads to worse performance when handling graphs with higher diameters. We also investigate the performance with respect to the sentence length. The models have better results when handling sentences with 20 or fewer tokens. Longer sentences pose additional challenges to the models.

G2S-GIN has a better performance in handling graphs with node out-degrees higher than 9. This indicates that GINs can be employed in tasks where the distribution of node degrees has a long

[4] https://github.com/jflanigan/jamr

[5] METEOR score is used as it is a sentence-level metric.

	REF ⇒ GEN		
Model	**ENT**	**CON**	**NEU**
S2S	38.45	11.17	50.38
G2S-GIN	49.78	9.80	40.42
G2S-GAT	49.48	8.09	42.43
G2S-GGNN	51.32	8.82	39.86
	GEN ⇒ REF		
Model	**ENT**	**CON**	**NEU**
S2S	73.79	12.75	13.46
G2S-GIN	76.27	10.65	13.08
G2S-GAT	77.54	8.54	13.92
G2S-GGNN	77.64	9.64	12.72

Table 6: Entailment (ENT), contradiction (CON) and neutral (NEU) average percentages for the LDC2017T10 test set. **(Top)** The premise and the hypothesis are the generated (GEN) and reference (REF) sentences, respectively. **(Bottom)** The hypothesis and the premise are the generated (GEN) and reference (REF) sentences, respectively.

tail. Surprisingly, S2S has a better performance than G2S-GGNN and G2S-GAT when handling graphs that contain high degree nodes.

Semantic Equivalence We perform an entailment experiment using BERT (Devlin et al., 2019) fine-tuned on the MultiNLI dataset (Williams et al., 2018) as a NLI model. We are interested in exploring whether a generated sentence (hypothesis) is semantically *entailed* by the reference sentence (premise). In a related text generation task, Falke et al. (2019) employ NLI models to rerank alternative predicted abstractive summaries.

Nevertheless, uniquely verifying whether the reference (REF) entails the generated sentence (GEN) or vice-versa (GEN entails REF) is not sufficient. For example, suppose that *"Today Jon walks"* is the REF and *"Jon walks"* is the GEN. Even though REF entails GEN, GEN does not entail REF, that is, GEN is too general (missing information). Furthermore, suppose that *"Jon walks"* is the REF and *"Today Jon walks"* is the GEN, GEN entails REF but REF does not entail GEN, that is, GEN is too specific (added information). Therefore, in addition to verify whether the reference entails the generated sentence, we also verify whether the generated sentence entails the reference.

Table 6 shows the average probabilities for entailment, contradiction and neutral classes on the LDC2017T10 test set. All G2S models have higher entailment compared to S2S. G2S-GGNN has 33.5% and 5.2% better entailment performances than S2S, when REF entails GEN and GEN entails REF, respectively. G2S models also generate sentences that contradict the reference sentences less. This suggests that our models are capable of capturing better semantic information from the graph generating outputs semantically related to the reference sentences.

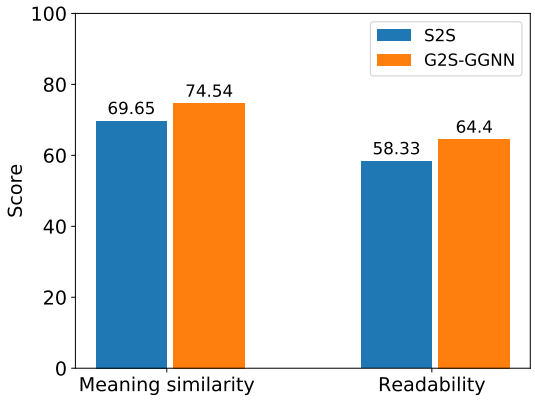

Figure 4: Human evaluation of the sentences generated by S2S and G2S-GGNN models. Results are statistically significant with $p < 0.05$, using Wilcoxon rank-sum test.

Human Evaluation To further assess the quality of the generated sentences, we conduct a human evaluation. We employ the *Direct Assessment* (DA) method (Graham et al., 2017) via Amazon Mechanical Turk. Using the DA method inspired by Mille et al. (2018), we assess two quality criteria: (i) *meaning similarity*: how close in meaning the generated text is to the gold sentence; and (ii) *readability*: how well the generated sentence reads (Is it good fluent English?).

We randomly select 100 sentences generated by S2S and G2S-GGNN and randomly assign them to HITs (following Mechanical Turk terminology). Human workers rate the sentences according to meaning similarity and readability on a 0-100 rating scale. The tasks are executed separately and workers were first given brief instructions. For each sentence, we collect scores from 5 workers and average them. Models are ranked according to the mean of sentence-level scores. We apply a quality control step filtering workers who do not score some faked and known sentences properly.

Figure 4 shows the results. In both metrics, G2S-GGNN has better human scores for meaning similarity and readability, suggesting a higher

(a / agree :ARG0 (a2 / and :op1 (c / country :wiki China :name (n / name :op1 China)) :op2 (c2 / country :wiki Kyrgyzstan :name (n2 / name :op1 Kyrgyzstan))) :ARG1 (t / threaten-01 :ARG0 (a3 / and :op1 (t2 / terrorism) :op2 (s / separatism) :op3 (e / extremism)) :ARG2 (a4 / and :op1 (s3 / security :mod (r / region)) :op2 (s4 / stability :mod r)) :time (s2 / still) :ARG1-of (m / major-02)) :medium (c3 / communique :mod (j / joint)))	
GOLD	China and Kyrgyzstan agreed in a joint communique that terrorism, separatism and extremism still pose major threats to regional security and stability.
S2S	In the joint communique, China and Kyrgyzstan still agreed to threaten terrorism, separatism, extremism and regional stability.
Song et. al (2018)	In a joint communique, China and Kyrgyzstan have agreed to still be a major threat to regional security, and regional stability.
G2S-GGNN	At a joint communique, China and Kyrgyzstan agreed that terrorism, separatism and extremism are still a major threat to region security and stability.

Table 7: An example of an AMR graph and generated sentences. GOLD refers to the reference sentence.

Model	ADDED	MISS
S2S	47.34	37.14
G2S-GIN	48.67	33.64
G2S-GAT	48.24	33.73
G2S-GGNN	48.66	34.06
GOLD	50.77	28.35

Table 8: Fraction of elements in the output that are not present in the input (ADDED) and the fraction of elements in the input graph that are missing in the generated sentence (MISS), for the test set of LDC2017T10. The token lemmas are used in the comparison. GOLD refers to the reference sentences.

quality of the generated sentences regarding S2S. The Pearson correlations between meaning similarity and readability scores, and METEOR[6] scores are 0.50 and 0.22, respectively.

Semantic Adequacy We also evaluate the semantic adequacy of our model (how well does the generated output match the input?) by comparing the number of added and missing tokens that occur in the generated versus reference sentences (GOLD). An added token is one that appears in the generated sentence but not in the input graph. Conversely, a missing token is one that occurs in the input but not in the output. In GOLD, added tokens are mostly function words while missing tokens are typically input concepts that differ from the output lemma. For instance, in Figure 1, *there* and *of* are added tokens while *person* is a missing token. As shown in Table 8, G2S approaches outperform the S2S baseline. G2S-GIN is closest to GOLD with respect to both metrics suggesting that this model is better able to generate novel words to construct the sentence and captures a larger range of concepts from the input AMR graph, covering more information.

Manual Inspection Table 7 shows sentences generated by S2S, Song et al. (2018), G2S-GAT, and the reference sentence. The example shows that our approach correctly verbalises the subject of the embedded clause *"China and ... agreed that terrorism, separatism and extremism$_{SUBJ}$ ... pose major threats to ..."*, while S2S and Song et al. (2018) are fooled by the fact that *agree* frequently takes an infinitival argument which shares its subject (*"China ...$_{SUBJ}$ agreed to threaten / have agreed to be a major threat"*). While this is a single example, it suggests that dual encoding enhances the model ability to take into account the dependencies and the graph structure information, rather than the frequency of n-grams.

6 Conclusion

We have studied the problem of generating text from AMR graphs. We introduced a novel architecture that explicitly encodes two parallel and adjuvant representations of the graph (top-down and bottom-up). We showed that our approach outperforms state-of-the-art results in AMR-to-text generation. We provided an extensive evaluation of our models and demonstrated that they are able to achieve the best performance. In the future, we will consider integrating deep generative graph models to express probabilistic dependencies among AMR nodes and edges.

Acknowledgments

This work has been supported by the German Research Foundation as part of the Research Training Group Adaptive Preparation of Information from Heterogeneous Sources (AIPHES) under grant No. GRK 1994/1.

[6]METEOR score is used as it is a sentence-level metric.

References

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. 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.

Anja Belz, Mike White, Dominic Espinosa, Eric Kow, Deirdre Hogan, and Amanda Stent. 2011. The first surface realisation shared task: Overview and evaluation results. In *Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 217–226, Nancy, France. Association for Computational Linguistics.

Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. 2017. Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42.

Kris Cao and Stephen Clark. 2019. Factorising AMR generation through syntax. 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 2157–2163, Minneapolis, Minnesota. Association for Computational Linguistics.

Thiago Castro Ferreira, Iacer Calixto, Sander Wubben, and Emiel Krahmer. 2017. Linguistic realisation as machine translation: Comparing different MT models for AMR-to-text generation. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 1–10, Santiago de Compostela, Spain. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merrienboer, 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.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tobias Falke, Leonardo F. R. Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. 2019. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2214–2220, Florence, Italy. Association for Computational Linguistics.

Jeffrey Flanigan, Chris Dyer, Noah A. Smith, and Jaime Carbonell. 2016a. CMU at SemEval-2016 task 8: Graph-based AMR parsing with infinite ramp loss. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1202–1206, San Diego, California. Association for Computational Linguistics.

Jeffrey Flanigan, Chris Dyer, Noah A. Smith, and Jaime Carbonell. 2016b. Generation from abstract meaning representation using tree transducers. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 731–739, San Diego, California. Association for Computational Linguistics.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. 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.

Yvette Graham, Timohy Baldwin, Alistair Moffat, and Justin Zobel. 2017. Can machine translation systems be evaluated by the crowd alone. *Natural Language Engineering*, 23(1):3–30.

Zhijiang Guo, Yan Zhang, Zhiyang Teng, and Wei Lu. 2019. Densely connected graph convolutional networks for graph-to-sequence learning. *Transactions*

of the Association for Computational Linguistics, 7:297–312.

Diederick P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diego, CA, USA.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

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, Volume 1 (Long and Short Papers)*, pages 2284–2293, Minneapolis, Minnesota. Association for Computational Linguistics.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural amr: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157, Vancouver, Canada. Association for Computational Linguistics.

Yujia Li, Richard Zemel, Marc Brockschmidt, and Daniel Tarlow. 2016. Gated graph sequence neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, San Juan, Puerto Rico.

Kexin Liao, Logan Lebanoff, and Fei Liu. 2018. Abstract meaning representation for multi-document summarization. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1178–1190, Santa Fe, New Mexico, USA. 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.

Simon Mille, Anja Belz, Bernd Bohnet, Yvette Graham, Emily Pitler, and Leo Wanner. 2018. The first multilingual surface realisation shared task (SR'18): Overview and evaluation results. In *Proceedings of the First Workshop on Multilingual Surface Realisation*, pages 1–12, Melbourne, Australia. Association for Computational Linguistics.

Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. Annotated gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, AKBC-WEKEX '12, pages 95–100, Stroudsburg, PA, USA. Association for Computational Linguistics.

Shashi Narayan and Claire Gardent. 2012. Structure-driven lexicalist generation. In *Proceedings of COLING 2012*, pages 2027–2042, Mumbai, India. The COLING 2012 Organizing Committee.

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 on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Nima Pourdamghani, Kevin Knight, and Ulf Hermjakob. 2016. Generating English from abstract meaning representations. In *Proceedings of the 9th International Natural Language Generation conference*, pages 21–25, Edinburgh, UK. Association for Computational Linguistics.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, pages 593–607.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Stuart M. Shieber, Gertjan van Noord, Fernando C. N. Pereira, and Robert C. Moore. 1990. Semantic-head-driven generation. *Computational Linguistics*, 16(1):30–42.

Linfeng Song, Daniel Gildea, Yue Zhang, Zhiguo Wang, and Jinsong Su. 2019. Semantic neural machine translation using AMR. *Transactions of the Association for Computational Linguistics*, 7:19–31.

Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2018. A graph-to-sequence model for AMR-to-text generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1616–1626, Melbourne, Australia. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

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

Boris Weisfeiler and A.A. Lehman. 1968. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsia*, pages 12–16.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How powerful are graph neural networks? In *International Conference on Learning Representations*, New Orleans, LA, USA.

A Generated Sentences

Table 9 shows three examples with sentences generated by S2S, Song et al. (2018), G2S-GGNN, and the reference sentence (GOLD).

B Human Evaluation Setup

- For each quality evaluation task (meaning similarity and readability), we independently sampled 100 generated sentences for each model.
- We created separate HITs for meaning similarity and readability evaluations. Each HIT contains 10 sentences.
- We paid $0.15 per HIT, employing five workers on each. For qualification, workers were required to have over 1000 approved HITs.
- We applied a quality control step. We removed workers who do not achieve a minimum threshold in sentences with known scores.

GOLD	I don't want to be miserable anymore and the longer he is around the more miserable I will be.
S2S	If he was in longer longer, I don't want to miserable and more miserable.
Song et. al (2018)	I don't want to be miserable anymore, and when he is around longer, I'm a miserable miserable.
G2S-GGNN	I don't want to be miserable anymore, and would be more miserable if he was around longer.

GOLD	Colombia is the source of much of the cocaine and heroin sold in the United States.
S2S	Colombia is a source of cocaine, much of cocaine and heroin sales in the United States.
Song et. al (2018)	Colombia is a source of much of much of cocaine and heroin in the United States.
G2S-GGNN	Colombia is a source of much cocaine and heroin and heroin sold in the United States.

GOLD	Discussions between Lula da Silva and Thabo Mbeki would also address new threats to international security such as terrorism, drugs, illegal weapons trafficking and aids.
S2S	Thabo da Silva has also addressed Thabo Mbeki to discuss new threats such as terrorism, drugs, illegal weapons trafficking and aids in international security.
Song et. al (2018)	Lula da Silva's discussion with Thabo also addressed a new threat against Thabo Mbeki and aids, drugs, illegal weapons and illegal weapons of weapon.
G2S-GGNN	Lula da Silva's discussion with Thabo da Silva also addressed new threat such as terrorism, drugs, illegal weapons trafficking and aids.

Table 9: Examples of generated sentences. GOLD refers to the reference sentence.