Generating Diverse Descriptions from Semantic Graphs

Jiuzhou Han **Daniel Beck** **Trevor Cohn**
School of Computing and Information Systems
The University of Melbourne, Australia
jiuzhouh@foxmail.com, {d.beck,trevor.cohn}@unimelb.edu.au

Abstract

Text generation from semantic graphs is traditionally performed with deterministic methods, which generate a unique description given an input graph. However, the generation problem admits a range of acceptable textual outputs, exhibiting lexical, syntactic and semantic variation. To address this disconnect, we present two main contributions. First, we propose a stochastic graph-to-text model, incorporating a latent variable in an encoder-decoder model, and its use in an ensemble. Second, to assess the diversity of the generated sentences, we propose a new automatic evaluation metric which jointly evaluates output diversity and quality in a multi-reference setting. We evaluate the models on WebNLG datasets in English and Russian, and show an ensemble of stochastic models produces diverse sets of generated sentences, while retaining similar quality to state-of-the-art models.

1 Introduction

Semantic graphs are an integral part of knowledge bases that integrate and store information in a structured and machine-accessible way (van Harmelen et al., 2008). They are usually limited to specific domains, describing concepts, entities and their relationships in the real world. Generating descriptions from semantic graphs is an important application of Natural Language Generation (NLG) and can be framed in a *graph-to-text* transduction approach.

In recent years, approaches to graph-to-text generation can be broadly categorised into two groups. The first uses a sequence-to-sequence model (Trisedya et al., 2018; Konstas et al., 2017; Ferreira et al., 2019): the key step in this approach is to linearise the input graph to a sequence. Sequence-to-sequence models have been proved to be effective for tasks like question answering (Yin et al., 2016), text summarisation (Nallapati et al., 2016), and constituency parsing (Vinyals et al., 2015). However, when dealing with graph inputs, this method does not take full advantage of the graph structure. Another approach is to handle the graph directly, using a graph-to-sequence model (Ribeiro et al., 2020; Beck et al., 2018; Zhao et al., 2020). This approach has been recently widely adopted as it shows better performance for generating text from graphs (Xu et al., 2018).

The models used in previous work are *deterministic*: given the same input graph, they will always generate the same text (assuming a deterministic decoding algorithm is used). However, it is widely known that many graphs admit multiple valid descriptions. This is evidenced by the presence of *multiple references* in datasets such as WebNLG (Gardent et al., 2017a,b) and it is a common phenomenon in other generation tasks such as machine translation and image captioning. In this work, we propose to use models that generate *sets of descriptions* instead of a single one. In particular, we develop *stochastic* models with latent variables that capture *diversity* aspects of semantic graph descriptions, such as lexical and syntactic variability. We also propose a novel evaluation methodology that combines quality and diversity into a single score, in order to address caveats of previously proposed diversity metrics. Our findings show that stochastic models perform favourably when generating sets of descriptions, without sacrificing the quality of state-of-the-art architectures.

2 Related Work

Graph-to-sequence Models Standard graph-to-sequence models have two main components: a graph encoder and a sequence decoder. The encoder learns the hidden representation of the input graph and the decoder generates text based on this representation. Different graph-to-sequence mod-

Proceedings of the 14th International Conference on Natural Language Generation (INLG), pages 1–11,
Aberdeen, Scotland, UK, 20-24 September 2021.

els vary mainly in the graph encoders.

Marcheggiani and Perez-Beltrachini (2018) proposed an encoder based on Graph Convolutional Networks (Kipf and Welling, 2017, GCNs), which directly exploit the input structure. Similar to Convolutional Neural Networks (LeCun et al., 1998), GCN layers can be stacked, resulting in representations that take into account non-adjacent, long-distance neighbours. Beck et al. (2018) used Gated Graph Neural Networks (Li et al., 2016) by extending networks on graph architectures with gating mechanisms, similar to Gated Recurrent Units (Cho et al., 2014, GRUs). Koncel-Kedziorski et al. (2019) proposed Graph Transformer Encoder by extending Transformers (Vaswani et al., 2017) to graph-structured inputs, based on the Graph Attention Network (Velickovic et al., 2017, GAT) architecture. This graph encoder generates node embeddings by attending over its neighbours through a self-attention strategy. Ribeiro et al. (2020) propose new models to encode an input graph with both global and local node contexts. To combine these two node representations together, they make a comparison between a cascaded architecture and a parallel architecture.

Latent Variable Models Within neural networks, a standard approach for generative models with latent variables is the Variational Autoencoder (VAE) (Kingma and Welling, 2014). The generative process is represented as: $p_\theta(x, z) = p_\theta(x \mid z)p_\theta(z)$, where $p_\theta(z)$ is the prior from which the latent variable is drawn, $p_\theta(x \mid z)$ is the likelihood of data point x conditioned on the latent variable z, typically calculated using a deep non-linear neural network, and θ denotes the model parameters.

Bowman et al. (2016) proposed a pioneering variational autoencoder for text generation to explicitly learn the global features using a continuous latent variable. They adapt the VAE to text data using an LSTM (Hochreiter and Schmidhuber, 1997) for both the encoder and the decoder, using a Gaussian prior to build a sequence autoencoder. This architecture can be extended to *conditional* tasks (when there is an input guiding the generation). Zhang et al. (2016) proposed an end-to-end variational model for Neural Machine Translation (NMT), using a continuous latent variable to capture the semantics in source sentences and guide the translation process. Schulz et al. (2018) proposed a more expressive word-level machine translation model incorporating a chain of latent variables, modelling lexical and syntactic variation in parallel corpora.

Diversity in Neural Networks and Generation Variational latent variable models are commonly employed when there is a need for generating diverse outputs. This is achieved by sampling from the latent variable every time a new output is required. One can also use a standard deterministic model and sample from the decoder distributions instead but this tends to decrease the quality of the generated outputs. Here we review a few common techniques to address this issue.

Dropout (Srivastava et al., 2014) is a regularisation method used to prevent overfitting in neural networks. At training time, it masks random parameters in the network at every iteration. Dropout can also be employed in the testing phase, during generation. This idea was first proposed by Gal and Ghahramani (2016) and it is also called Monte Carlo (MC) dropout. Because MC dropout disables neurons randomly, the network will have different outputs every generation, which can make a deterministic model generate different outputs.

Another technique to generate diverse outputs is *ensemble* learning. Typically, they are employed to prevent overfitting but they can also be used to generate diverse outputs. The idea is for each individual model in the ensemble to generate its own output. This approach can be very useful as each model tends to provide different optimal solutions in the network parameter space. This property has shown to benefit uncertainty estimation in deep learning (Lakshminarayanan et al., 2017). It can also be used both with deterministic and stochastic models, a property we exploit in our experiments.

3 Stochastic Graph-to-Sequence Model

In this section we introduce the proposed approach to generate diverse descriptions from semantic graphs. We start from the state-of-the-art model of Ribeiro et al. (2020), which is a deterministic graph-to-sequence architecture. Then we incorporate a latent variable and a variational training procedure to this model, in order to turn the model stochastic. This latent variable aims at capturing linguistic variations in the descriptions and is responsible for increasing the diversity at generation time. The architecture is shown in Figure 1.

3.1 Graph Encoder and Text Decoder

The encoder is similar to Ribeiro et al. (2020), consisting of a *global* and a *local* subencoder. The

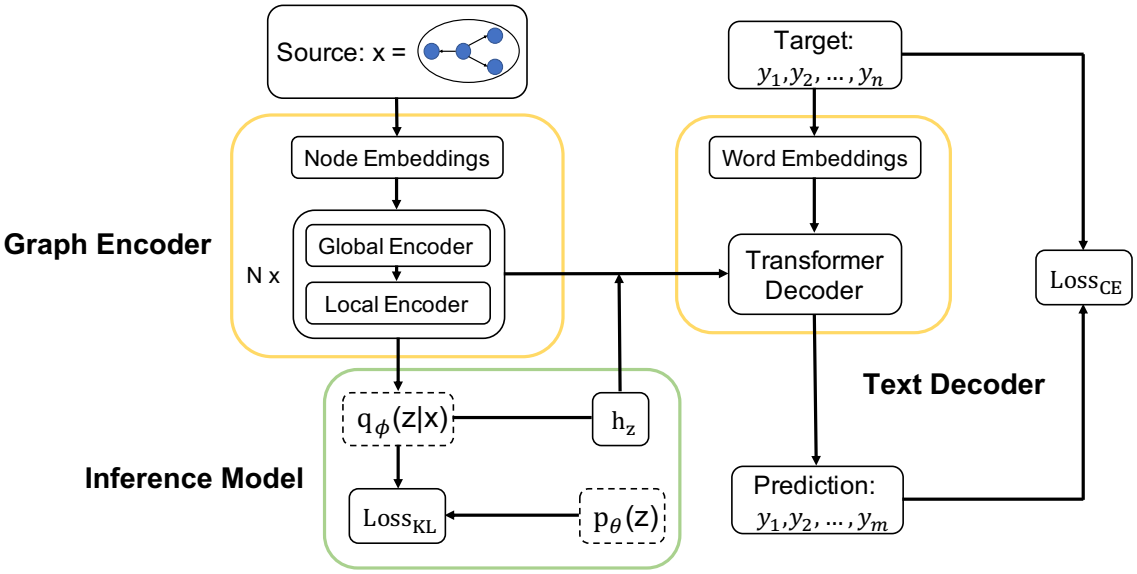


Figure 1: Proposed stochastic graph-to-sequence model architecture.

global encoder considers a wide range of contexts but it ignores the graph topology by considering each node as if it were connected to all the other nodes in the graph. The local encoder learns the hidden representation of each node on the basis of its neighbour nodes, which exploits the graph structure effectively. Combining both global and local node aggregations, this encoder can learn better contextualised node embeddings. The global encoding strategy is mainly based on the Transformer architecture (Vaswani et al., 2017), using a self-attention mechanism to calculate node representations of all nodes in the graph. The local encoding strategy adopts a modified version of Graph Attention Network (Velickovic et al., 2017) by adding relational weights to calculate the local node representations.

The decoder is also based on a *transformer* architecture. In our model, the input of the decoder is the contextualised node embeddings h_x concatenated with the hidden state of the latent variable h_z, which can be represented as $[h_x; h_z]$. Following Ribeiro et al. (2020), we also use beam search with length penalty (Wu et al., 2016) to encourage the model to generate longer sentences.

3.2 Inference Model

Here is where we introduce a latent Gaussian variable z, which together with the input graph x, guides the generation process. With this, the conditional probability of sentence y given x is

$$p(y|x) = \int_z p(y|z,x)p(z|x)dz.$$

The posterior inference in this model is intractable. Following previous work (Bowman et al., 2016; Kingma and Welling, 2014), we employ neural networks to fit the posterior distribution, to make the inference tractable. We regard the posterior as a diagonal Gaussian $\mathcal{N}\left(\mu, \text{diag}\left(\sigma^2\right)\right)$. The mean μ and variance σ^2 are parameterised with feed-forward neural networks (FFNNs), using the reparametrisation trick (Bowman et al., 2016; Kingma and Welling, 2014) of the Gaussian variables. It reparameterises the latent variable z as a function of mean μ and variance σ:

$$z = \mu + \sigma \odot \epsilon \quad \epsilon \sim \mathcal{N}(0, \text{I}),$$

where ϵ is a standard Gaussian variable which plays the role of introducing noises, and $\odot$ denotes element-wise multiplication. The reparametrisation trick enables back-propagation in optimisation process with Stochastic Gradient Descent (SGD). Then we transform the latent variable z into its hidden state h_z through another FFNN.

The training objective encourages the model to keep its posterior distributions $q(z \mid \mathbf{x})$ close to a prior $p(z)$ that is a standard Gaussian $\mathcal{N}(\mu = 0, \sigma = 1)$. The loss function of the stochastic conditional model can be defined as

$$\mathcal{L}(\phi, \theta; \mathbf{x}, \mathbf{y}) = -\mathbb{E}_{z \sim q_\phi(z|\mathbf{x})}\left[\log p_\theta\left(\mathbf{y} \mid z, \mathbf{x}\right)\right] + \text{KL}\left(q_\phi\left(z \mid \mathbf{x}\right) \| p(z)\right).$$

The first term is the expected negative log-likelihood of data which is called reconstruction loss or cross-entropy loss. It forces the model to learn to reconstruct the data. The second term is the KL divergence which acts as a regulariser. By minimising the KL term, we want to make the approximate posterior stay close to the prior. We use SGD to optimise the loss function.

3.3 Optimisation

As shown above, the stochastic model objective comprises two terms reconstruction and KL regularisation. The KL divergence term will be non-zero and the cross-entropy term will be relatively small if the model encodes task-relevant information in the latent variable z. A difficulty of training is that the KL term tends to zero, causing the model to ignore z. This makes the model deterministic. This phenomenon is also known as the *KL collapse* or *KL vanishing problem* (Lucas et al., 2019). We adopt the *KL Threshold* method (Pagnoni et al., 2018) to alleviate this issue. In this approach, we introduce a threshold ζ into the loss function to control the KL term. A large KL term means the latent variable learns much information. By setting a threshold, we can force the model to take at least a fixed KL regularisation cost. In our experiments, we set the threshold ζ as 10. The new loss function can be represented as

$$\mathcal{L}(\phi, \theta; \mathbf{x}, \mathbf{y}) = -\mathbb{E}_{z \sim q_\phi(z|\mathbf{x})} \left[\log p_\theta \left(\mathbf{y} \mid z, \mathbf{x}\right)\right] + \max\left(\mathrm{KL}\left(q_\phi\left(z \mid \mathbf{x}\right) \| p(z)\right), \zeta\right).$$

4 Joint Evaluation of Diversity and Quality

Addressing diversity in language generation is a recent topic that attracted attention in particular in image captioning. This led to the development of metrics that aim at measuring the diversity of a set of sentences, such as Self-BLEU (Zhu et al., 2018). However, these metrics are based only on the generated output space, ignoring the references in the gold standard. This lead to spurious measurements, such as unconditional language models having excellent performance according to these metrics, even though they have no practical use as they ignore the input.

To address these caveats, we propose a new evaluation procedure that assesses diversity and quality *jointly*. Our key insight (and assumption) is based on using the reference set as a gold standard for both aspects. Given a graph, the set of references acts as the "end goal", containing high-quality descriptions with sufficient levels of diversity.[1] We call this procedure **Multi-Score (MS)**.

The idea behind Multi-Score is shown pictorially in Figure 2. In this example, we have a single instance with three references and three predicted descriptions generated by a model. Given a sentence-level quality metric we can calculate it among *all possible pairs* between each prediction and reference, obtaining a weighted bipartite graph. We then solve the respective *maximum matching problem* for this bipartite graph and take the average weight of the edges corresponding to the optimal matching. We show the full procedure to calculate Multi-Score in Algorithm 1.

Algorithm 1 Multi-Score procedure

```
function MULTI-SCORE(o: outputs, r: references, M: sentence-level metric)
    G ← 0                                   ▷ initialise graph
    for i ← 0 to len(o) do                  ▷ fill graph
        for j ← 0 to len(r) do
            G(i, j) ← M(o[i], r[j])
    match ← MAXMATCH(G)                     ▷ stores edges
    score ← 0
    for edge ∈ match do
        score ← score + edge.weight
    return score / len(match)
                                            ▷ returns average weight
```

For the example in Figure 2, the optimal matching (shown in red) matches prediction 1 with output 2, prediction 2 with output 3 and prediction 3 with output 1. From this, the resulting Multi-Score is: $(56+50+58)/3 = \mathbf{54.67}$. The matching problem MAXMATCH can be solved using the Hungarian Algorithm (Kuhn, 2010) in $O(n^3)$ time, where n is the number of nodes in the bipartite graph. This makes the procedure efficient for reference set sizes found in standard datasets.

As a metric, Multi-Score has a number of desirable properties:

- As long as the sentence-level metric has an upper bound (which is the case of most standard automatic evaluation metrics), if the set of predictions is exactly equal to the references, then MS will give the maximum score.

[1] We discuss limitations of this assumption in Section 7.

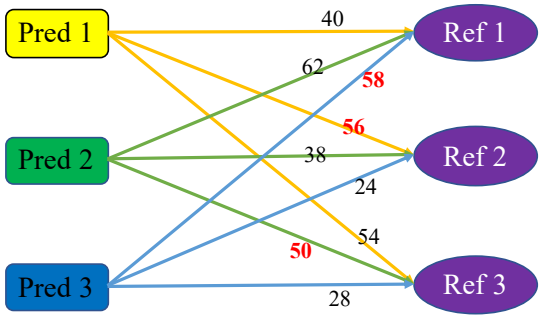


Figure 2: An example of calculating Multi-Score. The three "Pred" nodes on the left side represent three predicted descriptions while the three "Ref" nodes on the right side represent three references. The weight of each edge corresponds to the sentence-level quality score of this prediction-reference pair. The highlighted scores are the ones corresponding to the maximal matching, which are then used to calculate the MS metric. Other scores are ignored.

- If the outputs are diverse but unrelated to the references (as in an unconditional LM), MS will penalise the output because the underlying quality values will be low.
- If the outputs are high-quality but not diverse (typical of an n-best list in a deterministic model), MS will penalise the output due to the assignment constraint. One of the outputs will have a high-quality value but the others will have a low-quality value because they will be forced to match other references.
- Finally, MS can be used with any sentence-level quality metric, making it easily adaptable to any developments in better quality metrics, as well as other generation tasks.

5 Experimental Settings

5.1 Dataset

We evaluate the models using datasets from the WebNLG shared tasks (Gardent et al., 2017a,b). The data is composed of data-text pairs where the data is a set of RDF triples extracted from DBpedia and the text is the verbalisation of these triples. For each graph, there may be multiple descriptions. In our experiments, we assume a reference set of size 3 for each input, as most graphs in both datasets have three reference descriptions.

English WebNLG 2017 This dataset contains 18102 training, 872 development and 971 test data-text pairs. Entities are classified into 15 distinct categories (Astronaut, University, Monument, Building, ComicsCharacter, Food, Airport, SportsTeam, WrittenWork, Athlete, Artist, City, MeanOfTransportation, CelestialBody, Politician).

Russian WebNLG 2020 The Russian dataset comprises 16571 training, 790 development and 1102 test data-text pairs. This dataset has 9 distinct categories (Airport, Astronaut, Building, CelestialBody, ComicsCharacter, Food, Monument, SportsTeam, and University).

5.2 Preprocessing

Levi Graph Transformation To decrease the number of parameters and avoid parameter explosion, we follow previous work and use a Levi Graph Transformation (Ribeiro et al., 2020; Beck et al., 2018). This transformation creates new relation nodes from relational edges between entities, which explicitly represents the relations between an original node and its neighbour edges.

Byte Pair Encoding Following previous work (Ribeiro et al., 2020), we employ Byte Pair Encoding (BPE) to split entity words into frequent characters or character sequences which are subword units. After the BPE operations, some nodes in the graph are split to subwords. Likewise, we also split the target descriptions using BPE.

5.3 Models

All models are able to generate sets of descriptions: we generate three sentences per graph as this matches the number of available references. For the proposed stochastic models, we generate each sentence by sampling a new value for the latent variable. For the deterministic models, we use different decoding strategies to generate these sets.

Top-3 Beam Search Beam Search is the standard algorithm to obtain a sentence from deterministic models, by selecting the output with (approximate) highest probability. In Top-3 Beam Search, we choose the top-3 generated sentences from the final candidate list.

Total Random Sampling Random Sampling (Ippolito et al., 2019) generates a sentence from left to right sampling the next token from all possible candidates until the end-of-sequence symbol is generated. Because each token is *sampled* from

the distribution over next tokens given the previous ones, this method generates different outputs each time it generates a new description.

Top-3 Random Sampling In this approach, we still use Random Sampling but modify it slightly while generating the next token. Instead of sampling the next token from all possible candidates, the model samples the next token from the top-3 most likely candidates (Ippolito et al., 2019).

MC Dropout We employ MC dropout to the deterministic model and keep the dropout rate in the testing phase and training phase the same. It disables neurons randomly at decoding time, resulting in different outputs at each generation.

Ensemble Finally, we create an ensemble of three independently-trained deterministic models, whereby we select the most likely sentence from each model using Beam Search. These sentences then form the output set from the ensemble. Since this is a general strategy, *we also apply it to the stochastic model* as another point of comparison in our experiments.

6 Results

We assess each model on the test set of English and Russian datasets respectively and report the quality and diversity results. The quality evaluation scores (BLEU: Papineni et al. (2002), CHRF++: Popovic (2017)) are calculated based on the average score of the three outputs. We report the original BLEU and CHRF++ score to show the quality of the generated sentences from each model. The diversity evaluation scores (Self-BLEU, Multi-Score) are computed using the three outputs. As we describe in Section 4, our proposed diversity evaluation metrics require a sentence-level quality evaluation metric to compute the score of two sentences. We adopt sentence-level BLEU and CHRF++ and refer to their corresponding Multi-Score versions as MS-BLEU and MS-CHRF.

Table 1 shows the quality results on both English and Russian datasets. As expected, the two random sampling methods do not show good quality performance. For English data, our stochastic models perform on par with previous work and have comparable quality with deterministic models. The trends for English and Russian data are similar but Russian has lower scores in general.

The diversity scores of these two datasets are shown in Table 2. *Total random sampling* has the lowest Self-BLEU on two datasets, as expected, but it also has the worst quality. On the other hand, with our new metrics, the stochastic ensemble model gives the best results on both English and Russian datasets, showing high diversity without compromising quality.

6.1 Error Analysis

To further assess the quality of the generated sentences from each model, we perform a manual error analysis in a subset of the English test data. We randomly selected five input graphs, generating 15 sentences for each model (as we generate 3 sentences for each graph). Given we analysed five models, this gives a total of 75 sentences for our analysis. We observed three common mistakes from the outputs:

- **Syntax/Spelling Mistake:** There are grammar mistakes or spelling mistakes.
- **Lack of Information:** The information in the graph is not fully realised in the description.
- **Information Redundancy:** Some information in the sentence is repeated.

We calculate the rates of each model making different types of mistakes and report the results in Table 3. The results show that total random sampling makes the most mistakes among all models and most of them are syntax or spelling mistakes. *Top-3 random sampling* and *MC dropout* make the same percentage of total mistakes. The former makes almost half of the total information redundancy mistakes while the latter makes the most lack of information mistakes. Top-3 beam search makes fewer mistakes than the other three models and it does not make information redundancy mistakes in our evaluated test cases.

As for ensemble-based models, both deterministic and stochastic ensembles make the fewest total mistakes among all models. This is in line with the results obtained from automatic quality metrics. In particular, the deterministic ensemble does not make any syntax or spelling mistakes in the evaluated test cases. The stochastic ensemble also shows good performance with regard to the quality of the generated sentences, which has a low error rate for all types of mistakes.

In general, the diverse outputs generated by our proposed model tend to have comparable quality to the outputs from the best baseline model. However,

	English		Russian	
	BLEU↑	**CHRF++↑**	**BLEU↑**	**CHRF++↑**
Deterministic Models				
Top-3 beam search	62.69	74.48	52.50	64.76
Total random sampling	49.01	66.35	40.62	57.06
Top-3 random sampling	56.62	71.16	46.91	61.45
MC dropout	59.10	71.57	47.97	61.41
Ensemble	**63.31**	**74.52**	**53.60**	**65.30**
Stochastic Models				
Single model	62.81	74.12	52.45	64.43
Ensemble	62.88	74.25	52.60	64.38
Previous Work				
Melbourne (Gardent et al., 2017b)	54.52	70.72	-	-
Adapt (Gardent et al., 2017b)	60.59	76.01	-	-
CGE-LW (Ribeiro et al., 2020)	63.69	76.66	-	-

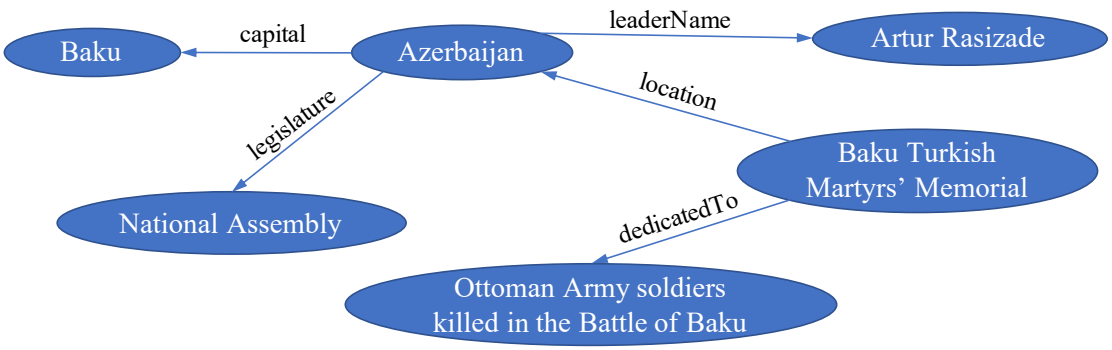
Table 1: Quality evaluation results on the test sets of both English and Russian datasets. Note that models without declaring decoding strategy use Beam Search. For reference, we also report results from previous work in the English dataset. Boldface shows the best result for a column, and arrows indicate the direction of improvement, i.e., ↑: higher is better.

	English			Russian		
	Self-B↓	**MS-B↑**	**MS-C↑**	**Self-B↓**	**MS-B↑**	**MS-C↑**
Deterministic Models						
Top-3 beam search	86.72	46.65	71.45	76.50	38.23	61.58
Total random sampling	**56.48**	40.47	67.00	**52.30**	31.37	56.30
Top-3 random sampling	64.66	45.15	70.40	60.31	35.61	59.95
MC dropout	68.70	46.90	70.87	61.59	36.14	59.37
Ensemble	81.31	47.32	71.52	75.70	38.50	61.71
Stochastic Models						
Single model	97.30	43.25	69.45	97.62	33.53	58.40
Ensemble	77.85	**47.61**	**71.95**	73.50	**38.86**	**61.95**

Table 2: Diversity evaluation results on the test sets of both English and Russian datasets. **Self-B** refers to Self-BLEU while **MS-B** and **MS-C** refer to the proposed Multi-Score metric using sentence-level BLEU and CHRF++ as the underlying quality metric. Note that models without declaring decoding strategy use beam search decoding.

Models	**Syntax/Spelling Mistake**	**Lack of Information**	**Information Redundancy**	**Average**
Deterministic Models				
Total random sampling	0.54	0.18	0.20	0.33
Top-3 random sampling	0.18	0.14	0.49	0.22
MC dropout	0.18	0.32	0.20	0.22
Top-3 beam search	0.07	0.14	0.00	0.09
Ensemble	0.00	0.09	0.03	0.06
Stochastic Models				
Ensemble	0.03	0.13	0.08	0.08

Table 3: Error analysis results, showing the rates of mistakes for each model.



DM (MC dropout) 1: The Baku Turkish Martyrs' Memorial, which is dedicated to the Ottoman Army soldiers killed in the battle of Baku, is found in Azerbaijan. The capital of Azerbaijan is Baku and the leader is Artur Rasizade. (missing: legislature information)
DM (MC dropout) 2: The Baku Turkish Martyrs' Memorial, which is dedicated to the Ottoman Army soldiers killed in the battle of Baku, is dedicated to the Ottoman Army soldiers killed in the country is led by Artur Rasizade. (missing: legislature information)
DM (MC dropout) 3: The Baku Turkish Martyrs' Memorial is dedicated to the Ottoman Army soldiers killed in the battle of Baku. It is dedicated to the Ottoman Army soldiers killed in the battle of Baku, the leader of Azerbaijan is Artur Rasizade. (missing: legislature information)

SM (Ensemble) 1: The Baku Turkish Martyrs' Memorial is dedicated to the Ottoman Army soldiers killed in the battle of Baku. It **is located in** Azerbaijan **whose capital is** Baku and its leader is Artur Rasizade. **The legislature is** the National Assembly.
SM (Ensemble) 2: Baku **is the capital of** Azerbaijan **where the legislature is** the National Assembly and the leader is Artur Rasizade. The country **is the location of** the Baku Turkish Martyrs Memorial **which is dedicated to** the Ottoman Army soldiers killed in the battle of Baku.
SM (Ensemble) 3: The Baku Turkish Martyrs' Memorial **is dedicated to** the Ottoman Army soldiers killed in the battle of Baku. It is located in Azerbaijan whose capital is Baku and its leader is Artur Rasizade**, and its legislature is** the National Assembly.

Table 4: A WebNLG input graph and the outputs from a Deterministic Model (MC dropout) and a Stochastic Model (Ensemble). Highlighted segments indicate mistakes: red, dotted lines represent Syntax/Spelling mistakes, blue, solid lines corresponds to Lack of Information, and orange, dashed lines represent Information Redundancy. **Bold** segments show examples of syntactic variations.

lack of information still remains a challenge for some instances in this setting. Addressing this problem is an avenue that we leave for future work.

6.2 Case Study

Table 4 shows an instance of a semantic graph from which we collect three outputs from a deterministic model (MC dropout) and a stochastic model (Ensemble). The outputs from MC dropout contain three types of mistakes and have low diversity. While there is no mistake in the outputs of the stochastic model, and the boldface illustrates syntactic variation.

7 Conclusion and Future Work

In this work, we first propose stochastic graph-to-text models to generate diverse sentences from semantic graphs. This was implemented through latent variable models that aim to capture linguistic variation and ensembling techniques. Furthermore, to solve the limitation of the existing diversity evaluation metrics, we also propose Multi-Score, a new automatic evaluation metric assessing diversity and quality jointly. It provides a general and effective way to assess the diversity of generated sentences for any text generation task. We perform experiments on English and Russian datasets and results demonstrate the generated sentences from the stochastic ensemble have both high diversity and high quality.

Since Multi-Score is based on using the reference set as the gold standard, it has a limitation that the variety of the reference sentences can largely influence the metric. Datasets containing reference sentences with higher quality and diversity will likely generate a more accurate Multi-Score for the predicted sentences. In other words, Multi-Score evaluates diversity *implicitly* through the references, as opposed to *explicit* judgements of diversity. However, explicit human evaluation requires a formal definition of diversity which is difficult to establish (as compared to quality judgements, for

instance). Nevertheless, addressing this challenge could provide a pathway to reduce the need for multiple references in evaluating diversity.

To the best of our knowledge this is the first work that incorporates a latent variable within a graph-to-sequence model. This in turn leads to many promising research avenues to explore in future work. Our analysis showed that the latent variable mostly helps in syntactic variation but less in other aspects such as semantics. Analysing the behaviour of the latent variable when modelling linguistic information is an important avenue that will enhance the understanding of stochastic models.

References

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, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 273–283. Association for Computational Linguistics.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 10–21. ACL.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, 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 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734. ACL.

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 2019, Hong Kong, China, November 3-7, 2019*, pages 552–562. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1050–1059. JMLR.org.

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, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 179–188. 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, INLG 2017, Santiago de Compostela, Spain, September 4-7, 2017*, pages 124–133. Association for Computational Linguistics.

Frank van Harmelen, Vladimir Lifschitz, and Bruce W. Porter, editors. 2008. *Handbook of Knowledge Representation*, volume 3 of *Foundations of Artificial Intelligence*. Elsevier.

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

Daphne Ippolito, Reno Kriz, João Sedoc, Maria Kustikova, and Chris Callison-Burch. 2019. Comparison of diverse decoding methods from conditional language models. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3752–3762. Association for Computational Linguistics.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2284–2293. 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, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 146–157. Association for Computational Linguistics.

Harold W. Kuhn. 2010. The hungarian method for the assignment problem. In Michael Jünger, Thomas M. Liebling, Denis Naddef, George L. Nemhauser, William R. Pulleyblank, Gerhard Reinelt, Giovanni Rinaldi, and Laurence A. Wolsey, editors, *50 Years of Integer Programming 1958-2008 - From the Early Years to the State-of-the-Art*, pages 29–47. Springer.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6402–6413.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. 2016. Gated graph sequence neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

James Lucas, George Tucker, Roger B. Grosse, and Mohammad Norouzi. 2019. Understanding posterior collapse in generative latent variable models. In *Deep Generative Models for Highly Structured Data, ICLR 2019 Workshop, New Orleans, Louisiana, United States, May 6, 2019*. OpenReview.net.

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, Tilburg University, The Netherlands, November 5-8, 2018*, pages 1–9. Association for Computational Linguistics.

Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çaglar Gülçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 280–290. ACL.

Artidoro Pagnoni, Kevin Liu, and Shangyan Li. 2018. Conditional variational autoencoder for neural machine translation. *CoRR*, abs/1812.04405.

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, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Maja Popovic. 2017. chrf++: words helping character n-grams. In *Proceedings of the Second Conference on Machine Translation, WMT 2017, Copenhagen, Denmark, September 7-8, 2017*, pages 612–618. 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. *Trans. Assoc. Comput. Linguistics*, 8:589–604.

Philip Schulz, Wilker Aziz, and Trevor Cohn. 2018. A stochastic decoder for neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 1243–1252. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

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, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 1627–1637. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2017. Graph attention networks. *CoRR*, abs/1710.10903.

Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey E. Hinton. 2015. Grammar as a foreign language. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2773–2781.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws,

Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Kun Xu, Lingfei Wu, Zhiguo Wang, Yansong Feng, and Vadim Sheinin. 2018. Graph2seq: Graph to sequence learning with attention-based neural networks. *CoRR*, abs/1804.00823.

Jun Yin, Xin Jiang, Zhengdong Lu, Lifeng Shang, Hang Li, and Xiaoming Li. 2016. Neural generative question answering. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 2972–2978. IJCAI/AAAI Press.

Biao Zhang, Deyi Xiong, Jinsong Su, Hong Duan, and Min Zhang. 2016. Variational neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 521–530. The Association for Computational Linguistics.

Chao Zhao, Marilyn A. Walker, and Snigdha Chaturvedi. 2020. Bridging the structural gap between encoding and decoding for data-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2481–2491. Association for Computational Linguistics.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 1097–1100. ACM.