Natural Language Engineering (2023), 1–26
doi:10.1017/S1351324923000207

CAMBRIDGE UNIVERSITY PRESS

ARTICLE

Improving semantic coverage of data-to-text generation model using dynamic memory networks

Elham Seifossadat and Hossein Sameti

Department of Computer Engineering, Sharif University of Technology, Tehran, Iran
Corresponding author: Hossein Sameti; E-mail: Sameti@sharif.edu

(Received 6 August 2021; revised 2 April 2023; accepted 17 April 2023)

Abstract

This paper proposes a sequence-to-sequence model for data-to-text generation, called DM-NLG, to generate a natural language text from structured nonlinguistic input. Specifically, by adding a dynamic memory module to the attention-based sequence-to-sequence model, it can store the information that leads to generate previous output words and use it to generate the next word. In this way, the decoder part of the model is aware of all previous decisions, and as a result, the generation of duplicate words or incomplete semantic concepts is prevented. To improve the generated sentences quality by the DM-NLG decoder, a postprocessing step is performed using the pretrained language models. To prove the effectiveness of the DM-NLG model, we performed experiments on five different datasets and observed that our proposed model is able to reduce the slot error rate rate by 50% and improve the BLEU by 10%, compared to the state-of-the-art models.

Keywords: Data-to-text generation; Natural language generation; Dynamic memory network; Recurrent neural network; Attention mechanism; Pre-trained language model; Dialogue system

1. Introduction

Data-to-text generation (D2T) is a subfield of natural language generation (NLG) that converts nonlinguistic, structured, formal, and abstract meaning representation (MR) to a natural language sentence and is used for the generation of task-oriented dialog, question answering, machine translation, selective generation systems, and search engines. For a given MR, which is a set of meaning labels in the form of tables or dialog acts, a generator must generate readable, fluent, adequate, and diverse sentences that express all the required information contained in the MR. Early models of D2T were based on manual rules and templates (Knight and Hatzivassiloglou, 1995; Langkilde and Knight, 1998; Ringger *et al.*, 2004). Although these systems produce high-quality text, they are domain-specific so they cannot be used for other domains, are expensive to build, and have limitations in generating diverse sentences. Another trend in data-to-text approaches is corpus-based methods which learn the mapping between MR and sentences in training data (Duboue and Mckeown, 2003; Barzilay and Lee, 2004; Barzilay and Lapata, 2005; Soricut and Marcu, 2006; Wong and Mooney, 2007; Belz, 2008; Liang, Michael, and Klein, 2009; Lu, Ng, and Lee, 2009; Angeli, Liang, and Klein, 2010; Kim and Mooney, 2010; Lu and Ng, 2011; Konstas and Lapata, 2012, Gyawali, 2016; Gyawali, 2016). These systems are relatively inexpensive and more generalizable to different domains. However, due to the lack of a mechanism to control the concept of sentences, their output is not always coherent and fluent.

Recently, systems designed to generate text are generally based on recurrent neural networks (RNN). The RNN-based D2T systems, which have an encoder–decoder structure, have been able

to achieve significant results in generating descriptive text from nonlinguistic data (Wen *et al.*, 2015a, 2015b, 2015c, 2016; Lebret, Grangier, and Auli, 2016; Mei, Bansal, and Walter, 2016; Nayak *et al.*, 2017; Riou *et al.*, 2017; Tran, Nguyen, and Nguyen, 2017; Tran and Nguyen, 2017; Lee, Krahmer, and Wubben, 2018; Liu *et al.*, 2018; Deriu and Cieliebak, 2018; Sha *et al.*, 2018; Tran and Nguyen, 2019; Qader, Portet, and Labbe, 2019; Shen *et al.*, 2020). However, these systems also have limitations. One of their most important problems is the inability to express all the necessary input information in the output text (Tran and Nguyen, 2019). The input MR may include meaning labels with a binary value (*true*, *false*, *yes*, *no*) or a value that cannot be expressed directly in words (*none*, *don't-care*) and can only be identified by their concept. For example, consider a meaning label that indicates whether a hotel allows dogs to enter or not, whose value can be *yes*, *no*, *none*, and *don't-care*. *none* means that information about allowing dogs in hotels is not available. This concept is expressed in sentences by the phrases such as *dogs are allowed* (similar when its value is *yes*), *dogs are not allowed* (similar when its value is *no*), or *I do not know if it allows dogs*. In these cases, the model should learn the relationship between these values and the words in the text and also how to express its meaning in the output. Another problem, which is related to the nature of RNN networks, is the inability to retain information about the distant past. Although LSTM is able to retain this information to some extent, the information that leads to the generation of the first words of the text is still forgotten as the output text lengthens. This causes problems like duplicate generated words, missing, and redundant meaning labels to be expressed in the output text. So far, many models have been tried to solve these deficiencies, among which models with encoder-attention-decoder structure have achieved relative but not complete success (Wen *et al.*, 2015a, 2015b, 2015c, 2016; Dusek and Jurcicek, 2016; Tran *et al.*, 2017; Tran and Nguyen, 2017, Qader *et al.*, 2019; Shen *et al.*, 2020; Shen *et al.*, 2020).

In this paper, we have enabled the basic attention-based sequence-to-sequence model to store information leading to the generation of previous words and generate new words not only based on the current state but also on previous information. For this purpose, we use a dynamic memory network (DMN) (Kumar *et al.*, 2016). The DMN, which is introduced in 2016 for the Question and Answer (Q&A) task, has two modules for reading and writing; at each step, the memory content is updated and rewritten based on its previous value and query, and the output tokens are generated based on the content read from memory. Using this type of memory along with the sequence-to-sequence model, each output word is generated not only based on the current state but also the information and history of decisions that led to the generation of previous words. In addition, since generating words in the proposed model, unlike previous models, are done by two different levels of attention between encoder and decoder, as well as decoder and memory, the meanings of the words and their relation to the input MR are well learned and as a result, all meaning labels with binary or special values will be expressed in the output. Improving the sentence structure is another goal that was achieved by applying a postprocessing step on the proposed model outputs by using pretrained language models.

We performed experiments to evaluate the performance of the proposed model and compare it with the baseline models. These experiments are performed on datasets and evaluation metrics used by RNN-based, autoencoder-based, and transformer-based baseline models to evaluate the effect of memory usage compared to the case where there is no memory. The results of the experiments show clear improvements in the quality of the sentences, both grammatically and semantically compared to the previous models. The main contributions of our work are summarized below:

- To reduce the flaws in the semantics of sentences generated by the attention-based sequence-to-sequence model, we proposed to use a dynamic memory module for storing the previous content vectors and decoder hidden states and then use its contents to generate new tokens.
- To make the dynamic memory more compatible with the input MRs, we proposed two different structures of only one-slot cell and multislot cells.

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

The rest of the paper is organized as follows: Section 2 presents related works. Section 3 presents the proposed DM-NLG model. Datasets, experimental setups, and evaluation metrics are described in Section 4. The resulting analysis and case study are presented in Section 5. The paper is concluded in Section 6.

2. Related works

RNN-based approaches have recently demonstrated promising performance in the D2T area. In the weather forecasting and sports domain, Mei *et al.* (2016) proposed an attention-based encoder–decoder model that used both local and global attention for content selection and generating sentences.

Wen *et al.* (2015c, 2016) created a dataset containing dialog acts of four different domains: finding a restaurant, finding a hotel, buying a laptop, and buying a TV. Then, they used LSTM to generate text word by word, while checking if all information is expressed in the output by using a heuristic gate. For each input dialog act, they over-generate a number of sentences, then all generated sentences are re-ranked with the CNN ranking method and those sentences that have higher ranks are chosen as output (Wen *et al.*, 2015c). The reported results of this model show that despite the use of the control gate, a large percentage of meaning labels are still not expressed in the output. To improve this model, Wen *et al.* (2015b) used a semantically conditioned LSTM instead of a heuristic control gate. In this model, the encoded input dialog act is updated after the generation of each word to reduce the impact of the expressed meaning labels for generating the next words. They also used a backward LSTMs re-ranking method to rank the generated sentences. They also compared the results obtained from these models with an Encoder-attention-Decoder structure (Wen *et al.*, 2015a). These results show that their proposed models for simple domains such as Restaurant and Hotel have been more successful than Laptop and TV, but still, some meaning labels remain unexpressed. After that, Riou *et al.* (2017) proposed Combined-Context LSTM which was a combination of the approaches of Wen *et al.* (2015b) and Wen *et al.* (2016). This hybrid model simultaneously controlled unexpressed slots and slots that should be considered by the decoder at each time step. Although this model was able to use more meaning label information to generate sentences than its base models, the quality of their output sentences decreased. As the next attempt to improve the Wen *et al.* (2015b) results, Tran *et al.* (2017) suggested using an attention mechanism to represent dialog acts and then refining the input token based on this representation before sending it to the decoder. The results showed that their model was able to improve the BLEU metric by 0.01 by preventing the repetition of words or semantic defects. Further, they added a control gate on the output information of the decoder to control semantic information (Tran and Nguyen, 2019). Finally, their model was able to reduce the slot error rate (SER) score value due to repetition or semantic defect of sentences by 40–50% while improving the value of BLEU by 0.01–0.02. Dusek and Jurcicek (2016) used a short version of the Restaurant dialog acts dataset (Wen *et al.*, 2015c), called BAGEL. They proposed a sequence-to-sequence model to generate sentences. Moreover, they re-ranked the n-best generated sentences and penalized those with repetitive words or semantic defects.

Lebret *et al.* (2016) generated a dataset containing Wikipedia biographies and their fact tables, called Wikibio. Then they used a neural feed-forward language model conditioned on structured data to generate the first sentence of each biography. Moreover, for sample-specific words like the name of persons, they used a copy mechanism that chooses these words from the input database to be expressed in the output sentence. Working on the same dataset, Liu *et al.* (2018) used a sequence-to-sequence structure to generate descriptive sentences from tables. The table information was encoded in the form of a field and value pairs, so their information would be present in the table representation. Also, in the decoding phase, two attention mechanisms were used, one at the word level and the other at the field level, to model the semantic relationship between the

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

table and the generated text. After that, Sha *et al.* (2018) extended this approach to learning the order of meaning labels in the corresponding text by adding a linked matrix, so their model was able to generate a more fluent output.

In the case of cross-domain dialog generation, Tseng *et al.* (2018), used a semantically conditioned variational autoencoder architecture. This model at first encoded the input MRs and their corresponding sentences to a latent variable. Then conditioning on this latent variable, the output sentences for a given MR were generated. Tran and Nguyen (2018a) proposed a variational neural language generator that was trained adversarially by first being trained on a source domain data and then being fine-tuned on a target domain under the guidance of text similarity and domain critics. Furthermore, to deal with the low-resource domain, they integrated a variational inference into an encoder–decoder generator (Tran and Nguyen, 2018b).

The E2E dataset, a large dataset in the restaurant domain, is produced in 2017 by Novikova *et al.* (2017) for use in the E2E challenge (Dusek, Novikova, and Rieser, 2018). Most of the submissions in that challenge were End-to-End sequence-to-sequence models (Juraska *et al.*, 2018; Gehrmann, Dai, and Elder, 2018; Zhang *et al.*, 2018; Gong, 2018; Deriu and Cieliebak, 2018) who were able to get a better result than the baseline approach by Dusek and Jurcicek (2016). Working on the same dataset, Qader *et al.* (2019) proposed a combination of an NLG and an NLU sequence-to-sequence models in the form of an autoencoder structure that can learn from annotated and nonannotated data. Their model was able to achieve a result close to the winner of the E2E challenge (Juraska *et al.*, 2018). Also, Shen *et al.* (2020) proposed to automatically extract the segmental structures of texts and learn to align them with their data correspondences. More precisely, at first the input records are encoded. Then at each time step, the decoder generates tokens based on the attention weights of the input records and previously expressed records in the output sentence. For reducing hallucination, they used a constraint that all records must be used only once.

Recently, pretrained language models like GPT-2 (Radford *et al.*, 2019) are used for data-to-text in the few-shot or zero-shot settings. As for the E2E dataset (Dusek *et al.*, 2018), Kasner and Dušek (2020) propose a few-shot approach for D2T based on iterative text editing by using LASERTAGGER (Malmi *et al.*, 2019), a sequence tagging model based on the Transformer (Vaswani *et al.*, 2017) architecture with the BERT (Devlin *et al.*, 2019) pretrained language model as the encoder and the pretrained language model GPT-2. This model first transforms data items to text using trivial templates, and then iteratively improves the resulting text by a neural model trained for the sentence fusion task. The output of the model then is filtered by a simple heuristic and re-ranked with GPT-2 language model. Peng *et al.* (2020) introduce a model-based GPT-2 called semantically conditioned generative pretraining. This model is first pretrained on a large amount of publicly available dialog datasets and then fine-tuned on the target D2T dataset with few training instances. Harkous *et al.* (2020) introduce the DATATUNER model, an end-to-end, domain-independent data-to-text system that used GPT-2 pretrained language model and a weakly supervised semantic fidelity classifier to detect and avoid generation errors such as hallucination and omission. Chen *et al.* (2020) proposed a few-shot learning approach that used GPT-2 with a copy mechanism. Kale and Rastogi (2020) used templates to improve the semantic correctness of the generated responses. In a zero-shot setting, at first, their model generates semantically correct but possibly incoherent responses based on the slots, with the constraint of templates, then by using T5 pretrained Text-to-Text model (Raffel *et al.*, 2020) as reorganizer, the generated utterances are transformed into coherent ones. Chang *et al.* (2021) use the data augmentation methods to improve few-shot text generation results using GPT-2 language model. Lee (2021) presents a simple one-stage approach to generating sentences from MRs using GPT-2 language model. Juraska and Walker (2021) proposed SEA-GUIDE, a semantic attention-guided decoding method for reducing semantic errors in the output text and applied it on fine-tuned T5 and BART (Lewis *et al.*, 2020). This decoding method extracts interpretable information from cross-attention between encoder and decoder to infer which meaning labels are mentioned in the generated text.

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

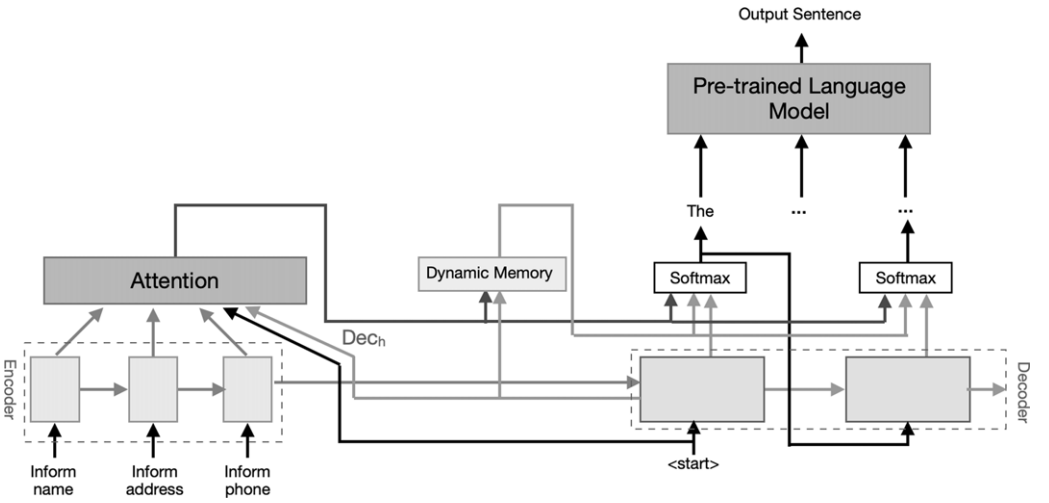

Figure 1. The block diagram of our proposed DM-NLG model. Here, the input of the encoder is an example of the restaurant dataset MRs.

The use of memory as an entity modifying (Puduppully, Dong, and Lapata, 2019) or entity tracker (Iso *et al.*, 2019) in the D2T task has already been suggested. In the model proposed by Puduppully *et al.* (2019), after each output token is generated by the decoder, the MR vectors of the input entities obtained by the encoder are updated using the memory module, and these updated vectors are used to select the next entity to describe in the output text. The proposed model by Iso *et al.* (2019) uses a memory module along with the decoder, that in each time step selects the appropriate entity to be expressed in the output text, updates the encoder's hidden state of the selected entity, and generates output tokens for describing it. However, our proposal differs from theirs in that, ours uses memory to help the decoder store the information it used to generate previous tokens. In this way, each output word is generated not only based on the current decoder state but also the information and history of decisions that led to the generation of previous words.

3. DM-NLG model

The proposed text generator in this paper is based on the sequence-to-sequence encoder–decoder architecture. The encoder part of the model, which is made by RNN, gets MR=$\{x_0, x_1, \ldots, x_L\}$ as input and converts it to hidden vectors. The hidden vectors of the encoder are used by the attention mechanism to produce the context vector and the final hidden state is used for initializing the hidden state of the decoder. Since the text generator must be able to express all the necessary input information in the generated output text, we propose to use dynamic memory alongside the sequence-to-sequence model. In this new model, which we named *DM-NLG*, the output tokens $\{w_0, w_1, \ldots, w_N\}$ are generated by the decoder based on the content of the memory cell. After generating each token, the content of the memory cell is updated. Finally, as a postprocessing stage, the generated sentences are given to a fine-tuned pretrained language model to improve their quality. The general outline of this model is shown in Figure 1.

3.1. Encoder

The encoder, which is a stack of GRU units, takes the input MR. At each step of the recurrence, a meaning label x_l of the input and hidden state s_{l-1} of the previous state is given to the GRU unit and a new hidden state s_l is generated. This process continues until all the meaning labels of

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

the input MR have been processed and their information stored inside the final hidden state. The formulas for this process are as follows:

$$\begin{aligned} r &= \sigma\left(W_r\, x_l + U_r\, s_{l-1}\right) \\ z &= \sigma\left(W_z\, x_l + U_z\, s_{l-1}\right) \\ \hat{s} &= \tanh\left(W_s\, x_l + U_s\left(r \odot s_{l-1}\right)\right) \\ s_l &= r \odot s_{l-1} + (1-r) \odot \hat{s} \end{aligned} \tag{1}$$

where W_r, W_z, W_s, U_r, U_z, and U_s are weight parameters that are learned during the model training process and $\odot$ denote for the element-wise product.

3.2. Decoder

The final hidden state of the encoder is used as the initial value for the hidden state of the decoder. The decoder, consisting of GRU units, at each step gets the generated token w_{t-1} and hidden state h_{t-1} of the previous step and generates the new hidden state h_t. The generated hidden state is used to compute a probability discrete distribution over the vocabulary and then to predict the next word:

$$\begin{aligned} h_0 &= s_L \\ r &= \sigma\left(W_r\, w_{t-1} + U_r\, h_{t-1}\right) \\ z &= \sigma\left(W_z / w_{t-1} + U_z\, h_{t-1}\right) \\ \hat{h} &= \tanh\left(W_h\, w_{t-1} + U_h\left(r \odot h_{t-1}\right)\right) \\ h_t &= r \odot h_{t-1} + (1-r) \odot \hat{h} \end{aligned} \tag{2}$$

where h_0 is the initial values of the decoder's hidden state and W_r, W_z, W_s, U_r, U_z, and U_s are weight parameters that are learned during the model training process.

3.3. Attention mechanism

The output of an NLG system is a descriptive text embracing the input MR; therefore, there should be some alignment between each output text token and input labels. This alignment is modeled through the attention mechanism. Hence, at each step t, the degree of relevancy of the hidden state h_t of the decoder to all encoder hidden states as well as previously generated token w_{t-1} is measured, and based on that a score is given to each hidden state of meaning labels. These scores indicate which input labels are more responsible for generating token w_t and are calculated as follows:

$$\begin{aligned} \text{score} &= V^T \tanh\left(W_a\, s_l + U_a\, h_t + Z_a\, w_{t-1}\right) \\ \alpha_{lt} &= \frac{\exp(\text{score}(s_l, h_t))}{\sum_j \exp\left(\text{score}(s_j, h_t)\right)} \\ c_t &= \sum_l \alpha_{lt} s_l \end{aligned} \tag{3}$$

where V, W_a, U_a, and Z_a are learning weight parameters. The context vector c_t, which is the weighted sum of the encoder hidden states, denotes the contribution of the input meaning labels in the production of the w_t.

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

3.4. Dynamic memory

As mentioned before in Equations (2) and (3), both phases of generating words by the decoder and calculating attention weights by the attention mechanism are performed based on the last generated word embedding vector (w_{t-1}) and the last hidden vector of the decoder (h_t). Therefore, the only information from the past steps that affect the attention weights and word generation at current step t is the content of the h_t vector, which includes a compact representation of the sequence of words produced from step 0 to $t-1$. On the other hand, as the generated word sequence lengthens, the weights of the initial generated words become less than the recently generated words in this representation. As a result, the information that leads to the generation of the words at the beginning of the sentence is lost, which results in the text generator to produce repetitive words or not to express part of the input meaning labels in the output text. Therefore, we propose to use dynamic memory to store the history of previous information. In this way, to generate the output word w_t, the content of memory that is initialized with encoder outputs, is first updated and re-written based on hidden state h_t. The information in the updated memory is then read by the attention mechanism and used to generate the output word. In this way, all the past information that leads to generating words is always available to the decoder.

In this work, two types of dynamic memories with different sizes and initialization values are proposed. In the first case, the memory M^t at time step t consists of a slot m_0^t that is initialized by the last encoder's hidden state s_L, so contains the compressed information of the input MR. In the second case, the number of memory slots M^t at time step t is equal to the input meaning labels, $M^t = \left\{m_0^t, m_1^t, \ldots, m_L^t\right\}$. These slots are initialized by their hidden states generated by the encoder, $\{s_0, s_1, \ldots, s_L\}$.

3.4.1. Memory writing module

As mentioned before, at each step t, the hidden vector h_{t-1} and the previously generated word w_{t-1} are given to the decoder to generate the hidden vector h_t. After that, this vector with the context vector generated by the attention mechanism is given to the writing module. The writing module, which is made of GRU units, updates, and rewrites the memory as follows:

$$\begin{aligned} f &= \sigma(W_f\, h_t + V_f\, c_t + U_f\, M^t) \\ \hat{m} &= \tanh\left(W_m\, [h_t \oplus c_t] + U_m(f\ \odot M^t)\right) \\ M^{t+1} &= (1-f) \odot M^t\ + f \odot \hat{m} \end{aligned} \tag{4}$$

where W_f, W_m, V_f, U_f, and U_m are learning weight parameters and $\oplus$ is the concatenation operation. If memory M^t has multiple slots, the writing module operates recursively on $\left\{m_0^t, m_1^t, \ldots, m_L^t\right\}$; Otherwise, it will act as a single GRU unit on slot m_0^t.

3.4.2. Memory reading module

At time step t and after the memory M^t is updated and rewritten in M^{t+1}, its information is read by the reading module and used to generate the next word w_{t+1}. The process of reading memory is done by the attention mechanism as follows:

$$\begin{aligned} \text{score} &= V_r^T \tanh(W_r\, h_t + U_r\, M^{t+1}) \\ \alpha_t &= \text{softmax}(\text{score}) \\ \beta_t &= \sum_t \alpha_t M^{t+1} \end{aligned} \tag{5}$$

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

where V_r, W_r, and U_r are learning weight parameters. The output distribution of each token then is defined by using a softmax function as follows:

$$\begin{aligned} &p(w_{t+1}|w_t, w_{t-1}, \ldots, w_0) = \text{softmax}(W_o H) \\ &H = [w_t \oplus h_t \oplus \beta_t] \end{aligned} \tag{6}$$

where W_o is the learning weight parameter.

3.5. Training

The model is trained to minimize the negative log-likelihood cost function based on back propagation. Accordingly, $J(\theta)$ formulated as follows:

$$J(\theta) = -\sum_l \hat{y}_l \log p_l \tag{7}$$

where $\hat{y}_l$ is the actual word distribution, p_l is the predicted word distribution. For training, the ground truth token for the previous time step is used as the input of the next step; but for inference, a simple beam search is used to generate output text.

3.6. Postprocessing

At this stage, a pretrained language model that autoregressively generates text, such as GPT-2 or Transformer-XL (Dai *et al.*, 2019), is used to improve the structural quality of the sentences produced by the decoder. For this purpose, first, the equivalent sentences for the MRs of the training data are generated by the DM-NLG model. These sentences are then used along with the ground truths as an input and output pair to fine-tune the pretrained language models, in the form of `<BOS> generated text <SEP> ground truths text <EOS>`. Finally, each generated sentence at the inference time is given to the fine-tuned model as a language model, in the form of `<BOS> generated text <SEP>` to generate the equivalent sentence as a continuation of the input. The impact of using each of these pretrained models in improving the quality of output sentences is discussed in the next section.

4. Experiments

To evaluate the performance of the proposed model, we conducted experiments. The used datasets, experiment settings, evaluation metrics, results of the experiments, and their analysis are described below.

4.1. Datasets

We used five publicly available datasets to examine the quality of the proposed models: finding a hotel, finding a restaurant, buying a TV, buying a laptop published by Wen *et al.* (2015c), the E2E challenge dataset provided by Dusek *et al.* (2018) and Personage dataset published by Oraby *et al.* (2018) which provides multiple reference outputs with various personality type (agreeable, disagreeable, conscientious, unconscientious, and extrovert) for each E2E MR. These datasets contain a set of scenarios, each containing an MR and an equivalent sentence. Details of these datasets are given in Table 1. In preprocessing, the value of meaning labels that appear exactly in the text (such as the name, address, phone, postal code, and etc.) is replaced with a specific token both in the text and in the input MR. In this way, while reducing the size of the vocabulary, the information that influences the process of generating text becomes more compact. To perform postprocessing, these values were put back into place.

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

Table 1. Datasets statistics.

	Restaurant	Hotel	Laptop	TV	E2E	Personage
#Train	3114	3223	7944	4221	33,525	88,855
#Validation	1039	1075	2649	1407	4299	-
#Test	1039	1074	2649	1407	4693	1390
#Meaning labels	22	22	33	29	8	9

4.2. Experimental setups

The proposed models are implemented using the TensorFlow library and trained on Google Co-laboratory with one Tesla P100-PCIE-16 GB GPU. The hidden layer size and embedding dimensions are set to 80, the batch size is set to 100, and the generators were trained with a 70% of keep dropout rate. The models were initialized with pretrained word vectors GloVe (Pennington, Socher, and Manning, 2014) and optimized using Adam optimizer with a learning rate of 1e-3. This process is terminated by early stopping based on the validation loss. Moreover, for every five epochs, L2-regularization with $\lambda = 1e-5$ is added to lose values. In the inference phase, beam search with width 10 is used and for each MR, 20 candidate sentences are over-generated, then the five top sentences are selected based on their negative log-likelihood values.[a] For fine-tuning GPT-2 and Transformer XL as a postprocessing module, we used the *gpt2* and *transfo-xl-wt103* pre-trained models in the Hugging Face library (Wolf *et al.*, 2020) and trained them for five epochs with a learning rate of 5e-5 and batch size 16. The output sentences are then regenerated with beam width 10 and temperature 0.9 using fine-tuned models.

To compare the performance of the proposed DM-NLG model with the pre-trained encoder–decoder models, we fine-tuned the T5 and BART models on all five datasets. For this purpose, the *t5-base* and *bart-base* models available in the Hugging Face Library were used and trained for five epochs with a learning rate of 5e-5 and batch size 16. The output sentences are then generated using these fine-tuned models with a beam width of 10.

4.3. Evaluation metrics

Performance of the proposed model on the Restaurant, Hotel, TV, and Laptop dataset was evaluated using BLEU-4 (Papineni *et al.*, 2002), BERTScore (Zhang *et al.*, 2020), and SER metrics. SER is defined as

$$\text{SER} = \frac{p+q}{N} \tag{8}$$

where N is the number of slots in the MR, p, and q are the number of missing and redundant slots in the generated sentence (Wen *et al.*, 2015c). For the E2E and Personage datasets, BLEU-4, NIST (Martin and Przybocki, 2000), METEOR (Lavie, Sagae, and Jayaraman, 2004), ROUGE-L (Lin, 2004), SER, and BERTScore metrics are used.[b] For evaluating the diversity of generated text, Shannon Entropy is used as follows:

$$H = -\sum_{x \in S} \frac{\text{freq}}{\text{total}} * \log_2 \left(\frac{\text{freq}}{\text{total}} \right) \tag{9}$$

where S is the set of unique words in all generated sentences by the model, *freq* is the number of times that a word occurs in all generated sentences, and *total* is the number of words in all reference sentences (Oraby *et al.*, 2018).

[a] Code and data will be published in https://github.com/seifossadat/DM-NLG

[b] https://github.com/tuetschek/e2e-metrics

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

We also ran human evaluation to evaluate the faithfulness (How many of the semantic units in the given sentence can be found/recognized in the given MR), coverage (How many of the given MRs slots values can be found/recognized in the given sentence), and fluency (whether the given sentence is clear, natural, grammatically correct, and understandable) of the generated sentences by our proposed model. For fluency, we asked the judges to evaluate the given sentence and then give it a score: 1 (with many errors and hardly understandable), 2 (with a few errors, but mostly understandable), and 3 (clear, natural, grammatically correct, completely understandable). To do this, 50 test MRs are selected randomly from each dataset. Also, as judges, 20 native English speakers (Fleiss's $\kappa = 0.51$, Krippen-dorff's $\alpha = 0.59$) are used. To avoid all bias, judges are shown one of the randomly selected MRs at a time, together with its gold sentence and the corresponding generated sentence by our model. Judges were not aware of which sentences were generated by our model.

4.4. Baselines

We compared our proposed models against strong baselines including:

- *Gating-based models*. HLSTM (Wen *et al.*, 2015c) which uses a heuristic gate to ensure that all information was accurately expressed in the generated text; SCLSTM (Wen *et al.*, 2015b) which uses a sigmoid control gate to keep track of meaning labels in generating text; SRGRU-Context, ESRGRU-MUL, ESRGRU-INNER, and RALSTM (Tran *et al.*, 2017; Tran and Nguyen, 2017, 2019) which use different attention mechanisms for representing the input MR and then refining the input token based on this representation using a sigmoid gate.
- *Attention-based models*. ENCDEC (Wen *et al.*, 2015a) which applies the attention mechanism to an LSTM encoder–decoder; CONTEXT (Oraby *et al.*, 2018) which uses a simple sequence-to-sequence with attention model; Fine-Control (Harrison *et al.*, 2019) which uses sequence-to-sequence with attention model and incorporates side constraint information into the generation process by adding additional inputs to the LSTM decoder; Seg&Align (Shen *et al.*, 2020) which uses LSTM encoder-attention-decoder to first segments the text into small fragments and then learns the alignment between data and text segments.
- *Autoencoders*. SCVAE (Tseng *et al.*, 2018) which uses Conditional Variational autoencoder architecture to generate dialog sentences from semantic representations; VRALSTM and VNLG (Tran and Nguyen, 2018a, 2018b) which use Variational encoder–decoder based models designed to deal with low in-domain data; (Qader *et al.*, 2019) work which uses two sequence-to-sequence models for understanding and generating learned jointly to compensate for the lack of annotation data.
- *Transformers*. Chen *et al.* (2020) which use the GPT-2 as generator and enabled it to switch between copying input and generating tokens; Chang *et al.* (2021) which use data augmentation methods with GPT-2 language model; Lee (2021) which uses GPT-2 for generating sentences given the input MRs; SEA-GUIDE-T5-small, SEA-GUIDE-T5-based and SEA-GUIDE-BART-based (Juraska and Walker, 2021) which use a semantic attention-guided decoding method along beam search to reduce semantic errors in the output text.

5. Results and analysis

Tables 2[c], 3, 4[c], and 5[c] contain the results of our proposed DM-NLG model, with and without postprocessing, along with the aforementioned baseline models on the Restaurant, Hotel,

[c]We performed five runs with different random initializations of the network and reported the average of resulted BLEU and SER values. The evaluation metrics scores for baseline systems are the values reported by authors.

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

Table 2. Performance of the baseline and the proposed DM-NLG model on Restaurant, Hotel, TV, and Laptop datasets in terms of BLEU and SER.

	Restaurant		Hotel		TV		Laptop	
Model	BLEU	SER (%)	BLEU	SER (%)	BLEU	SER (%)	BLEU	SER (%)
SCLSTM	75.25	0.38	84.82	3.07	52.65	2.31	51.16	0.79
HLSTM	74.66	0.74	85.04	2.67	52.50	2.50	51.34	1.10
ENCDEC	73.98	2.78	85.49	4.69	51.82	3.18	51.08	4.04
SRGRU-Context	76.34	0.94	87.76	0.98	53.11	1.33	51.19	1.19
ESRGRU-MUL	76.49	1.01	88.99	0.53	53.21	0.90	52.23	1.10
ESRGRU-INNER	76.56	0.76	89.67	0.94	53.30	0.90	52.36	0.90
RALSTM	77.89	0.16	89.81	0.43	54.06	0.63	52.52	0.42
SCVAE	54.00	0.28	65.20	0.15	47.80	0.28	44.20	0.18
VRALSTM	77.09	0.36	88.51	0.57	53.56	0.73	52.10	0.59
VNLG	77.09	0.36	88.51	0.57	53.56	0.73	52.10	0.59
DM-NLG without postprocessing								
without memory	75.10	0.20	85.16	0.90	52.40	1.30	50.87	1.75
with one-slot memory	78.12	0.09	89.90	0.13	54.05	0.37	53.08	0.42
with multislot memory	79.11	**0.03***	90.81	**0.07***	55.18	**0.24***	54.20	**0.16***
DM-NLG with postprocessing: GPT-2								
without memory	75.31	0.20	85.50	0.90	53.10	1.30	51.00	1.75
with one-slot memory	78.91	0.09	90.52	0.13	54.60	0.37	53.71	0.42
with multislot memory	**79.62****	**0.03***	**91.13****	**0.07***	**55.72***	**0.24***	**54.91***	**0.16***
DM-NLG with postprocessing: Transformer-XL								
without memory	75.10	0.20	85.00	0.90	52.00	1.30	50.87	1.75
with one-slot memory	78.09	0.09	89.72	0.13	53.93	0.37	52.73	0.42
with multislot memory	78.93	**0.03***	89.91	**0.07***	54.65	**0.24***	53.09	**0.16***

Measured by CIDEr and BLEU-4 (B4) scores. * and ** indicate statistically significant best results at $p \leq 0.05$ and $p \leq 0.01$, respectively, for a paired *t*-test evaluation. The best and second-best models are highlighted in bold and underline face, respectively.

TV, Laptop, E2E, and Personage datasets. As can be seen, by adding memory to the attentional sequence-to-sequence model, SER is significantly decreased and BERTScore increased compared with attentional encoder–decoder autoencoder-based and transformer-based models indicating the improvement of the semantic and structural quality of the generated sentences by our proposed model. However, despite the fact that our proposed model has been able to get a high BERTScore and surpass all the basic models in terms of semantics and structure, its BLEU values are either less or slightly better than the baselines, except for the Personage dataset which baseline models were small and simple compared to our proposed model. This difference is due to the fact that n-gram-based metrics compare the hypothesis with a reference sentence based on its phrases, words, syntax, and length similarity. As a result, if the hypothesis sentence is completely correct in terms of semantics and structure but not similar in terms of used words and phrases, it gets a lower BLEU score.

Based on Tables 2–5, using the GPT model as a booster of the quality of the generated sentences leads to a slight improvement in the value of the BERTScore evaluation metric. However,

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

Table 3. Performance of the proposed DM-NLG model on Restaurant, Hotel, TV, and Laptop datasets in terms of BERTScore.

Model	Restaurant	Hotel	TV	Laptop
DM-NLG without postprocessing				
without memory	0.890	0.942	0.865	0.830
with one-slot memory	0.919	0.960	0.885	0.870
with multislot memory	0.932	0.971	0.911	0.900
DM-NLG with postprocessing: GPT-2				
without memory	0.910	0.950	0.870	0.850
with one-slot memory	0.915	0.967	0.890	0.872
with multislot memory	0.948	0.972	0.915	0.910
DM-NLG with postprocessing: Transformer-XL				
without memory	0.875	0.925	0.823	0.808
with one-slot memory	0.904	0.940	0.880	0.847
with multislot memory	0.920	0.945	0.895	0.890

the Transformer-XL model reduces the quality of the output sentences, as it focuses more on learning the long dependencies between words and sentence segments, and given the number of model parameters (approximately twice as many as the GPT-2) and length of training sentences (30–40 tokens), the model tends to shorten sentences by expressing existing meaning labels only. It should be noted that since the generated sentences by DM-NLG are used as training data to fine-tune the GPT and the Transformer-XL models if there is any semantic error due to missing or redundant meaning labels expressed in these sentences, this error will also appear in the output sentences of the GPT and Transformer-XL and does not change the SER. Examples of the sentences generated by the decoder part of the DM-NLG and then regenerated by GPT and Transformer-XL are shown in Table 6. Moreover, the performance of the DM-NLG model without adding memory is close to other attention-based encoder–decoder models; but with the addition of memory to the model, the BERTScore metric improves and the SER decreases significantly. As mentioned earlier, the DM-NLG model with one-slot memory is initialized with the final hidden state of the encoder, which contains a compressed representation of the whole input MR. As a result, the decoder is aware of the history of previous alignments when generating the hidden state for the next token, resulting in a further reduction in the SER and improvement in BERTScore. When the information of each input meaning label is stored in different memory slots, the history of previous alignments is kept separately. Hence, the weights are more evenly distributed between the labels when reading memory, resulting in further reduction of SER and BERTScore improvement compared to the case where there is only one slot.

Tables 7–9 show the results obtained by comparing the performance of the transformer-based encoder–decoder models with the proposed DM-NLG model without the postprocessing step. For this purpose, we fine-tuned the T5-base and BART-base models on the training dataset and examined the quality of sentences generated by these models by automated evaluation metrics. As can be seen, for the proposed model, the values of SER are less and the values of BERTScore are higher than the fine-tuned models indicating that the proposed model is well able to control the flow of input MRs information at the output text. This is why, although transformer-based models are trained on large volumes of text and then fine-tuned on training data, the values obtained for n-gram-based evaluation metrics are also close to each other. Therefore, it can be concluded that

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

Table 4. Performance of the baseline and the proposed DM-NLG model on E2E dataset in terms of BLEU, NIST, METEOR, ROUGE-L, SER, and BERTScore.

Model	BLEU	METEOR	NIST	ROUGE-L	SER	BERTScore F1
Qader *et al.* (2019)	64.00	0.4500	-	0.6700	-	-
Shen *et al.* (2020)	65.10	0.4550	-	0.6820	-	-
Chen *et al.* (2020)	63.72	0.4025	7.76	0.6623	-	-
Chang *et al.* (2021)	**68.88**	**0.4853**	**8.89**	**0.7212**	-	-
Lee (2021)	67.94	0.4579	8.64	0.6998	-	0.942
SEA-GUIDE-T5-small (Juraska and Walker, 2021)	67.50	0.4530	-	0.6900	0.04	-
SEA-GUIDE-T5-based (Juraska and Walker, 2021)	68.20	0.4540	-	0.6910	0.05	-
SEA-GUIDE-BART-based (Juraska and Walker, 2021)	68.00	0.4530	-	0.6950	0.08	-
DM-NLG without postprocessing						
without memory	64.13	0.4373	8.02	0.6659	0.11	0.918
with one-slot memory	66.32	0.4503	8.70	0.6803	0.07	0.930
with multislot memory	66.70	0.4562	8.72	0.6911	**0.03***	0.951
DM-NLG with postprocessing: GPT-2						
without memory	65.72	0.4403	8.08	0.6691	0.11	0.909
with one-slot memory	67.52	0.4613	8.79	0.6923	0.07	0.943
with multislot memory	68.59	0.4817	8.82	0.7134	**0.03***	**0.960***
DM-NLG with postprocessing: Transformer-XL						
without memory	64.10	0.4373	8.02	0.6658	0.11	0.853
with one-slot memory	64.31	0.4393	8.19	0.6680	0.07	0.882
with multislot memory	64.94	0.4544	8.25	0.6719	**0.03***	0.908

The best and second-best models are highlighted in **bold** and underline face, respectively. * indicates statistically significant best results at $p \leq 0.05$, for a paired *t*-test evaluation.

the proposed model has a good ability to generate text from structured data, especially despite having a smaller number of trainable parameters.

For all datasets, we evaluated the amount of variation in the training set, and also in the output generated text by the proposed model and fine-tuned BART and T5 using entropy (Equation 9), where the larger entropy value indicates a larger amount of linguistic variation in the generated outputs. The results are shown in Table 10. It is noticed that the highest entropy value is related to the training sets, and none of the models are able to match their variability. However, the small difference between the entropy value of the generated and training sentences for the different datasets shows the ability of the proposed model to produce texts with appropriate variations.

To evaluate the performance of the proposed model on unseen data, we designed an experiment in which the DM-NLG model is trained on one data set and tested on another. Due to the shared meaning labels between the Hotel and Restaurant datasets, TV, and Laptop datasets, as well as E2E and Personage datasets, we performed this test individually on these sets of data, so that in each set one was used as training data and the other as test data. The performance comparisons are shown in Table 11. As observed, for all sets of data, the BLEU and BERTScore decreased and SER is increased, respectively. These differences are larger when the Hotel and Laptop domains are used

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

Table 5. Performance of the baseline and the proposed DM-NLG model on Personage dataset in terms of BLEU, NIST, METEOR, ROUGE-L, SER, and BERTScore.

Model	BLEU	METEOR	NIST	ROUGE-L	SER	BERTScore F1
Oraby *et al.* (2018)	37.66	0.3964	5.3437	0.5255	-	-
Harrison *et al.* (2019)	55.98	-	-	-	0.014	-
DM-NLG without postprocessing						
without memory	85.89	0.4391	5.5100	0.5227	0.017	0.920
with one-slot memory	88.17	0.4452	5.5700	0.5263	0.004	0.943
with multislot memory	91.84	0.4500	5.6130	0.5290	**0.000**	0.956
DM-NLG with postprocessing: GPT-2						
without memory	85.95	0.4410	5.5230	0.5230	0.017	0.917
with one-slot memory	88.63	0.4471	5.6000	0.5281	0.004	0.944
with multislot memory	**92.87***	**0.4607***	**5.6300***	**0.5430***	**0.000***	**0.974***
DM-NLG with post-processing: Transformer-XL						
without memory	85.80	0.4391	5.5100	0.5220	0.017	0.893
with one-slot memory	86.03	0.4397	5.5190	0.5220	0.004	0.918
with multislot memory	86.73	0.4400	5.5231	0.5234	**0.000***	0.931

* indicates statistically significant best results at $p \leq 0.01$, for a paired t-test evaluation. The best and second-best models are highlighted in bold and underline face, respectively.

Table 6. Examples of generated sentences by DM-NLG with multislot memory and improved by GPT and Transformer-XL for given MRs from the Restaurant and Hotel datasets.

MR	inform-no-match(food=’gluten free’;pricerange=’cheap’)
Reference	unfortunately there is no cheap restaurant that serve -s gluten free food
multislot memory	i’m sorry i did not find any restaurant -s that serve gluten free food and are cheap
refined by GPT	i’m sorry there is no cheap restaurant that serve-s gluten free food
refined by Transformer-XL	there is no cheap restaurant that serve -s gluten free
MR	inform-count(type=’hotel’;count=’182’;pricerange=dont-care)
Reference	there are 182 hotel -s if you do not care about the price
multislot memory	there are 182 hotel -s available if you do not care about price range
refined by GPT	there are 182 hotel -s if you do not care about the price range
refined by Transformer-XL	there are 182 hotel -s if you do not care price range

as unseen datasets due to the different distributions of unshared meaning labels in the test scenarios of each one. Both E2E and Personage datasets have the same meaning labels. The Personage dataset only has one additional personality tag that specifies the style of the sentences. When E2E and Personage are used as training and test sets, respectively, the trained model does not have the ability to mimic the personality style of sentences. As a result, the value of the BLEU and BERTScore is greatly reduced although SER has not changed much. Conversely, when Personage is used as training data, the model can generate descriptive sentences well for E2E datasets input

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

Table 7. Evaluation results on Restaurant, Hotel, TV and Laptop datasets, for proposed DM-NLG model, without postprocessing, compared to pretrained encoder–decoder transformer-based models, in terms of BLEU (B), SER, and BERTScore-F1 (BS).

	Restaurant			Hotel			TV			Laptop		
Model	B	SER	BS	B	SER	BS	B	SER	BS	B	SER	BS
DM-NLG												
without memory	75.10	0.20	0.890	85.16	0.90	0.942	52.40	1.30	0.865	50.87	1.75	0.830
one-slot memory	78.12	0.09	0.919	89.90	0.13	0.960	54.05	0.37	0.885	53.08	0.42	0.870
multislot memory	**79.11****	**0.03***	**0.932***	**90.81***	**0.07***	**0.971***	**55.18***	**0.24***	**0.911***	**54.20****	**0.16***	**0.900***
T5-base	76.00	1.10	0.830	87.45	0.37	0.905	52.00	1.17	0.861	50.36	0.97	0.856
BART-base	78.80	0.96	0.857	88.50	0.30	0.922	53.10	1.15	0.892	51.33	0.40	0.882

* and ** indicate statistically significant best results at $p \leq 0.05$ and $p \leq 0.01$, respectively, for a paired t-test evaluation. The best and second-best models are highlighted in bold and underline face, respectively.

Table 8. Evaluation results on E2E dataset, for proposed DM-NLG model, without postprocessing, compared to pretrained encoder–decoder transformer-based models, in terms of BLEU, NIST, METEOR, ROUGE-L, SER, and BERTScore-F1.

Model	BLEU	METEOR	NIST	ROUGE-L	SER	BERTScore-F1
DM-NLG						
without memory	64.13	0.4373	8.02	0.6659	0.11	0.918
with one-slot memory	66.32	0.4503	8.70	0.6803	0.07	0.930
with multislot memory	66.70	**0.4562***	**8.72**	0.6911	**0.03***	**0.951***
T5-base	66.59	0.4530	8.70	0.6870	0.15	0.907
BART-base	**67.00***	0.4550	**8.72***	**0.6915***	0.10	0.920

* indicates statistically significant best results at $p \leq 0.05$, for a paired t-test evaluation. The best and second-best models are highlighted in bold and underline face, respectively.

Table 9. Evaluation results on Personage dataset, for proposed DM-NLG model, without postprocessing, compared to pretrained encoder–decoder transformer-based models, in terms of BLEU, NIST, METEOR, ROUGE-L, SER, and BERTScore-F1.

Model	BLEU	METEOR	NIST	ROUGE-L	SER	BERTScore-F1
DM-NLG						
without memory	85.89	0.4391	5.5100	0.5227	0.017	0.920
with one-slot memory	88.17	0.4452	5.5700	0.5263	0.004	0.943
with multislot memory	**91.84***	0.4500	**5.6130***	**0.5290***	**0.000***	**0.956***
T5-base	88.16	0.4481	5.5420	0.5180	0.070	0.890
BART-base	90.03	**0.4497***	5.5900	0.5213	0.050	0.915

* indicates statistically significant best results at $p \leq 0.05$, for a paired t-test evaluation. The best and second-best models are highlighted in bold and underline face, respectively.

MRs, and therefore, the BLEU and BERTScore decrease slightly. These results prove that our proposed model also can adapt to a new, unseen dataset.

As subjective evaluations, we compared the output generated by our best-performing model, DM-NLG multislot along with postprocessing, with its gold reference. Table 12 shows the scores of each human evaluation metric for each dataset. The sentences generated by our model received

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

Table 10. Performance of our proposed model on six datasets in terms of Shannon Text Entropy.

Model	Restaurant	Hotel	TV	Laptop	E2E	Personage
References of training set	6.99	7.02	7.99	8.53	8.26	6.31
DM-NLG one-slot memory	5.65	5.35	6.70	6.90	7.32	5.26
DM-NLG multislot memory	6.45	6.33	7.48	8.05	7.92	6.01
T5-base	5.73	6.30	6.10	7.01	6.81	5.19
BART-base	5.90	6.15	6.91	7.26	7.65	5.73

Table 11. Performance of proposed models *without postprocessing* on seen and unseen data. in terms of BLEU, SER, and BERTScore.

Dataset Pair	Experiment Setup	Model	BLEU	SER	BERTScore-F1
Restaurant & Hotel	Seen: Restaurant	DM-NLG one-slot	78.12	0.090	0.919
		DM-NLG multislot	79.11	0.030	0.932
	Unseen: Restaurant	DM-NLG one-slot	77.01	0.100	0.891
		DM-NLG multislot	77.19	0.098	0.905
	Seen: Hotel	DM-NLG one-slot	89.90	0.130	0.960
		DM-NLG multislot	90.81	0.070	0.971
	Unseen: Hotel	DM-NLG one-slot	80.71	0.430	0.865
		DM-NLG multislot	83.02	0.230	0.877
TV & Laptop	Seen: TV	DM-NLG one-slot	54.05	0.370	0.885
		DM-NLG multislot	55.18	0.240	0.911
	Unseen: TV	DM-NLG one-slot	51.00	1.810	0.835
		DM-NLG multislot	54.05	1.370	0.852
	Seen: Laptop	DM-NLG one-slot	53.08	0.420	0.870
		DM-NLG multislot	54.20	0.170	0.900
	Unseen: Laptop	DM-NLG one-slot	51.25	2.310	0.811
		DM-NLG multislot	51.40	2.170	0.829
E2E & Personage	Seen: E2E	DM-NLG one-slot	66.32	0.070	0.930
		DM-NLG multislot	66.70	0.030	0.951
	Unseen: E2E	DM-NLG one-slot	65.17	0.090	0.924
		DM-NLG multislot	66.10	0.052	0.947
	Seen: Personage	DM-NLG one-slot	88.17	0.004	0.943
		DM-NLG multislot	91.84	0.000	0.956
	Unseen: Personage	DM-NLG one-slot	59.16	0.080	0.841
		DM-NLG multislot	60.15	0.002	0.852

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

Table 12. Results of Human Evaluations on five used datasets in terms of Faithfulness, Coverage, and Fluency (rating out of 3).

Dataset	Model	Faithfulness	Coverage	Fluency
Restaurant	Reference	100.0%	96.0%	2.71
	DM-NLG multislot	100.0%	95.2%	2.68
Hotel	Reference	100.0%	95.4%	2.89
	DM-NLG multislot	100.0%	95.4%	2.90**
TV	Reference	100.0%	94.8%	2.74
	DM-NLG multislot	100.0%	92.0 %	2.70
Laptop	Reference	100.0%	93.7%	2.81
	DM-NLG multislot	100.0%	94.0*%	2.75
E2E	Reference	95.1%	99.1%	2.85
	DM-NLG multislot	100.0*%	98.9%	2.81
Personage	Reference	100.0%	100.0%	2.95
	DM-NLG multislot	100.0%	98.6%	2.89

References are the golden truth sentences in each dataset. * and ** indicate statistically significant best results at $p \leq 0.05$, and $p \leq 0.01$, for a paired t-test and ANOVA evaluation, respectively.

Table 13. Comparison of the generated sentences from the Laptop dataset for the DM-NLG model *without postprocessing* and baselines.

MR	?compare(name='satellite tartarus 56'; platform='windows 8'; dimension='15.3 inch'; name='satellite achelous 45'; platform='windows 8.1'; dimension='15.5 inch')
Reference	the satellite tartarus 56 has a 15.3 inch dimension and uses windows 8 whereas satellite achelous 45 has a 15.5 inch dimension and a windows 8.1 platform which one do you prefer
Wen *et al.* (2015a)	the satellite tartarus 56 is 15.3 inch the satellite achelous 45 operates on windows 8 and has a 15.5 inch screen which one do you prefer platform = windows 8.1
Wen *et al.* (2015b)	the satellite tartarus 56 operates on windows 8 and has a 15.3 inch display and is 15.5 inch -s which one do you prefer name = satellite achelous 45, platform = windows 8.1
Wen *et al.* (2015c)	the satellite tartarus 56 is 15.3 inch -s and operates on windows 8 and the satellite achelous 45 has a 15.5 inch display which one do you want platform = windows 8.1
Tran *et al.* (2017)	the satellite tartarus 56 has a 15.3 inch dimension the satellite achelous 45 is 15.5 inch -s which one do you want platform = windows 8, platform = windows 8.1
Tran and Nguyen (2019)	the satellite tartarus 56 is 15.3 inch -s and runs on windows 8 the satellite achelous 45 is 15.5 inch -s which one do you prefer platform = windows 8.1
T5	the satellite tartarus 56 has a slot-inform-design and 15.3 inch -s and windows 8 and satellite achelous 45 has a slot-inform-design and 15.5 inch -s and windows 8.1 which one do you want to choose between the two? slot-inform-design
BART	the satellite tartarus 56 is 15.3 inch -s with windows 8 versus satellite achelous 45 is 15.5 inch -s with windows 8.1 which one
One-slot memory	satellite tartarus 56 has a 15.3 inch -s display and operates on windows 8 also the satellite achelous 45 which is 15.5 inch -s operates on windows 8.1 which one do you prefer
Multi-slot memory	the satellite tartarus 56 is 15.3 inch -s display and runs windows 8 while the satellite achelous 45 is 15.5 inch -s and operates on windows 8.1 which one do you like more

The missed and redundant meaning labels for each generated sentence are shown in red and orange, respectively.

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

Table 14. Comparison of the generated sentences from the E2E dataset for the DM-NLG model *without postprocessing* and baselines.

MR	name[The Punter], eatType[restaurant], food[Indian], priceRange[moderate], customer rating[1 out of 5], area[city center], familyFriendly[no], near[Express by Holiday Inn]
Reference	The Punter is a restaurant providing Indian food in the moderate price range. It is located in the city center. It is near Express by Holiday Inn. Its customer rating is 1 out of 5. familyFriendly[no]
Juraska *et al.* (2018)	The Punter is a moderately priced Indian restaurant in the city center near Express by Holiday Inn. It is not kid friendly and has a customer rating of 1 out of 5.
Gehrmann *et al.* (2018)	The Punter is a restaurant providing Indian food in the moderate price range. It is located in the city center and is near Express by Holiday Inn. Its customer rating is 1 out of 5. familyFriendly[no]
Zhang *et al.* (2018)	The Punter is a restaurant providing moderately Indian food. It is located in the city center near Express by Holiday Inn. Its customer rating is 1 out of 5. familyFriendly[no]
Gong (2018)	The Punter is a Indian restaurant in the city center near Express by Holiday Inn. It is not kids friendly and has a moderate price range and a customer rating of 1 out of 5.
Qader *et al.* (2019)	The punter is a restaurant serving moderate Indian food. It is located in the city center near Express by Holiday Inn. Its customer rating is 1 out of 5. familyFriendly[no]
T5	The Punter is a family friendly restaurant, is located near Express by Holiday Inn. It has a customer rating of 1 out of 5. familyFriendly[no], food[Indian], area[city center], familyFriendly[yes]
BART	The Punter is a moderate Indian restaurant with a customer rating of 1 out of 5. It is located in the city center near Express by Holiday Inn. family friendly[no]
One-slot memory	The punter is a restaurant providing Indian food in the moderate price range. It is located in the city center near Express by Holiday Inn. Its customer rating is 1 out of 5 and is not family friendly.
Multislot memory	The Punter is a moderately priced restaurant providing Indian food. It is located in the city center near Express by Holiday Inn. Its customer rating is 1 out of 5 and is not family friendly.

The missed and redundant meaning labels for each generated sentence are shown in red and orange, respectively.

high scores from the judges for all human evaluation metrics. It should be noted that, in the E2E dataset, there is a slot called food, which specifies what kind of food is served in the restaurant (*Chinese, English, French, Indian, Italian, Japanese, and Fast Food*). But for some MRs, instead of expressing types of foods, names of foods, like *"wines and cheeses," "cheeses, fruits and wines," "snacks, wine and coffee," "fish and chips*," and *"pasta,"* are mentioned in the reference sentences. Considering that the nationality of the food cannot be determined with certainty based on its name, the judges have considered such cases as a missing slot. For this reason, this data set has not received a full score for the coverage and faithfulness factors. Moreover, the scores of fluency of the proposed model are lower compared to the reference sentences (about 0.1 units), which is an acceptable difference considering that the reference sentences are written by humans. Examples of input MRs and sentences generated by our proposed model are given in Tables 13–15.

6. Case study and visualization

As an intuitive way to show the performance of the DM-NLG model, Tables 13–15 show samples of generated texts for certain MRs from the Laptop, E2E, and Personage datasets by our model without doing postprocessing. The input MR selected from the Laptop domain relates to the comparison of two different models of laptops, so it includes duplicate labels. The model should be able to express all these labels in the output sentence in a way that the concept of comparison is clearly recognizable in the generated sentence. As shown in Table 13, considering the concept

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

Table 15. Comparison of the generated sentences from the Personage dataset for the DM-NLG multislot model *without postprocessing* and baselines.

MR	name[nameVariable], eatType[restaurant], food[English], priceRange[high], customerRating[average], area[city center], familyFriendly[yes], near[nearVariable]
Personality: **Agreeable**	**Reference**: Did you say nameVariable? I see, well it is a restaurant with an average rating, also nameVariable is near nearVariable kid friendly in city center, an English place and quite expensive, okay?
	Harrison *et al.* (2019): You want to know more about nameVariable? Yeah, ok it has an average rating, it is a restaurant and it is an English restaurant in city center, quite expensive near nearVariable and family friendly. [OK]
	T5: let's see what we can find on nameVariable. i see, it is an English restaurant near nearVariable in city center. it isn't rather kid friendly, also it's a restaurant. priceRange[high], customerRating[average], familyFriendly[yes], familyFriendly[no], eatType[restaurant]
	BART: let's see what we can find on nameVariable. i see, it is an English restaurant near nearVariable in city center with a rather average rating also it isn't family friendly, you see? priceRange[high], familyFriendly[yes], familyFriendly[no]
	DM-NLG multi-slot: let's see what we can find on nameVariable. it is an English restaurant with a average rating city center near nearVariable, quite expensive and kid friendly.
Personality: **Disagreeable**	**Reference**: Damn expensive nameVariable is it's near nearVariable, also it is a restaurant. . . It is in city center. it's an English restaurant. nameVariable is family friendly. It has like, an average rating.
	Harrison *et al.* (2019): Oh god I mean, I thought everybody knew that nameVariable is expensive with an average rating, it's near nearVariable, it is an English place, it is a restaurant and nameVariable is in city center, also it is family friendly. [OK]
	T5: oh god i mean, i don't know. mmhm. . . nameVariable it's an English place in city center, it has an average rating, nameVariable it isn't kid friendly. it is a restaurant. priceRange[high], familyFriendly[yes], familyFriendly[no]
	BART: oh god i mean, i don't know. mmhm. . . nameVariable it's expensive in city center, it is an English place, nameVariable is kid friendly, also it has an average rating. it is near nearVariable.
	DM-NLG multi-slot: nameVariable is a restaurant with an average rating. nameVariable is in city center. it is an English place and high priced near nearVariable and it is kid friendly. oh god come on.
Personality: **Conscientious**	**Reference**: Let's see, nameVariable. . . Well, i see, I suppose it is a restaurant with an average rating, also it is expensive in city center, and it is an English restaurant and kid friendly near nearVariable.
	Harrison *et al.* (2019): I think that nameVariable is a restaurant, it has an average rating and it is somewhat expensive in city center and an English restaurant near nearVariable. It is rather kid friendly.
	T5: let's see what we can find on nameVariable. i see, it is an English restaurant near nearVariable in city center. it is kid friendly. nameVariable it has an average rating. priceRange[high]
	BART: let's see what we can find on nameVariable. i see, it seems to me that it is an English restaurant near nearVariable in city center, expensive with an average rating, it is kid friendly and it is a restaurant. eatType[restaurant]
	DM-NLG multi-slot: let's see, nameVariable. . . i see, well it is a restaurant with an average rating, also it is expensive in city center, kid friendly near nearVariable and an English place.
Personality: **Unconscientious**	**Reference**: Oh gosh yeah, I am not sure. nameVariable has an average rating, nameVariable is an English place, expensive and family friendly near nearVariable, it is a restaurant and it is kind of in city center.

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

Table 15. Continued.

	Harrison *et al.* (2019): Yeah, I don't know. Mmhm. . . nameVariable is a restaurant, nameVariable is expensive, it's kind of in city center, it is an English place and nameVariable has an average rating. It is kind of near nearVariable.
	T5: oh god ohhh gosh i don't know. mmhm . . . nameVariable i mean it is an English restaurant in city center, it has an average rating, it is expensive, and it is kid friendly.
	BART: oh god ohhh gosh i don't know. mmhm . . . nameVariable I mean it is an English place in city center, expensive, it is kid friendly with an average rating, it is a restaurant.
	DM-NLG multi-slot: oh gosh i don't know. nameVariable is a retaurant with an average rating, also it is damn kid friendly in city center, expensive near nearVariable and an English place, i mean.
Personality: **Extravert**	**Reference**:nameVariable has an average rating, it is expensive, also it's an English place and family friendly, buddy, it is a restaurant and it is near nearVariable in city center, you know!
	Harrison *et al.* (2019): nameVariable is a restaurant, you know, it is an English place, family friendly in city center and expensive near nearVariable and nameVariable has an average rating friend!
	T5: Oh god i mean, i thought everybody knew that it's expensive, it is an English restaurant in city center and it has an average rating. It is kid friendly, you know! name[nameVariable]
	BART: oh god i mean, i thought everybody knew that nameVariable it's an English place in city center, it has an average rating, expensive, and kid friendly. It is a restaurant.
	DM-NLG multi-slot: nameVariable is a restaurant with an average rating, also nameVariable is expensive in city center, it is an English place and it is near nearVariable, also it is kid friendly. buddy, you know!

The missed, redundant, and duplicated meaning labels for each generated sentence are shown in red, orange, and blue, receptively.

of comparison, it can be said that text generated by our proposed model is close to the reference text by producing words such as "also" and "while." The input MR selected from the E2E dataset has a label with a binary value, so the model should be able to express this concept in the output sentence. The output generated by our model successfully expressed all labels. Finally, the input MR selected from the Personage dataset also has a label with a binary value, but the main task is generating sentences that describe the input MRs with different types of personality. As can be seen in Table 15, our proposed model is able to produce the equivalent text of each personality label correctly.

The heat map in Figure 2 shows the average value of the forget gate of the memory writing module (Equation 4) and attention weights (Equation 3) for a sentence from the Laptop dataset generated by the DM-NLG one-slot model. The horizontal and vertical axes show the generated sentence and the input MR, respectively. The darker color reveals a higher value while the lighter part has a lower value. Figure 2 shows that, at the beginning of the word generation process based on the input MR, the value of the forgetting gate is smaller, meaning that the contents of the memory are changing. As it gets closer to the end of the sentence while expressing all the meaning labels in it, the value of the forget gate becomes larger, meaning that the contents of the memory no longer change. The attention weights assigned to the input labels after generating each word in the output also reflect the desirable effect of memory usage.

The average value of the forgetting gate of the memory writing module, attention weights of the memory reading module (Equation 5), and attention weights for a sentence from the Laptop dataset generated by the DM-NLG multislot model are shown in Figure 3. Here too, the horizontal and vertical axes show the generated sentence and the input MR, respectively. In this model, for each memory cell, the value of the forget gate of writing modules changes according to its corresponding meaning labels expressed in the text. As can be seen in Figure 3a, the contents of memory slots corresponding to the *name*, *type,* and *price range* labels that appear at the beginning and middle of the text change more than the *drive* and *platform* labels that are expressed at the

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

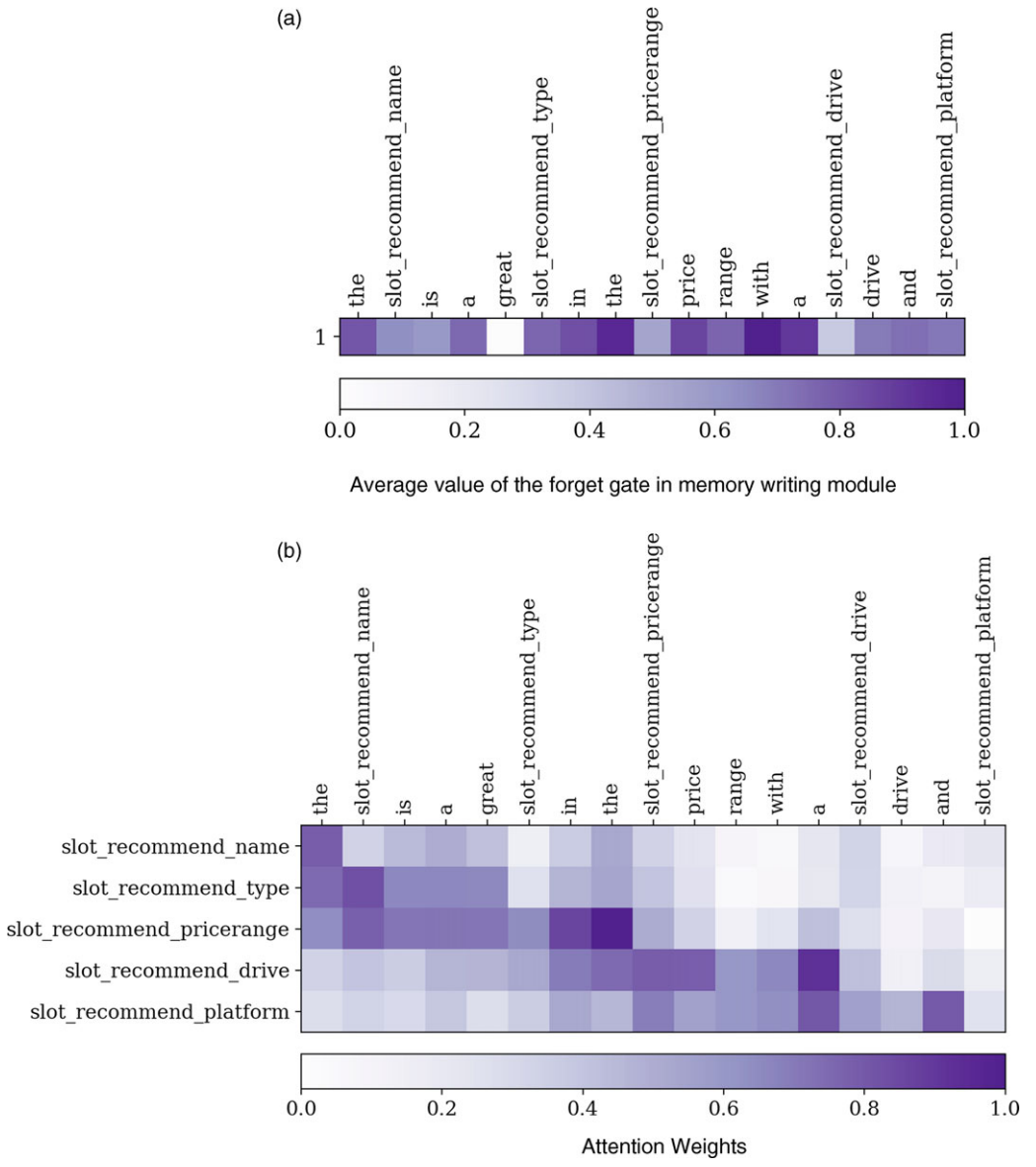

Figure 2. The change of the forget gate value of memory writing module (a) and the attention weights (b) for DM-NLG one-slot model after generating each word of the generated text for a given MR from the Laptop dataset.

end of the sentence. Also, the uniform value of the forgetting gate from the middle of the sentence indicates that the change in the content of memory slots is balanced based on their previous step value and the new information. Furthermore, as observed in Figure 3b, at the beginning of the output generation, the weights that the memory reading module assigns to each slot have a nonuniform distribution, which indicates that some memory slots have more attention weights. But in the end, when the memory changes very slightly, these weights are evenly distributed, indicating that the contents of each memory slot corresponding to each meaning label are read in a balanced way. Weights calculated to the attention mechanism also show the same favorable effect on memory usage.

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

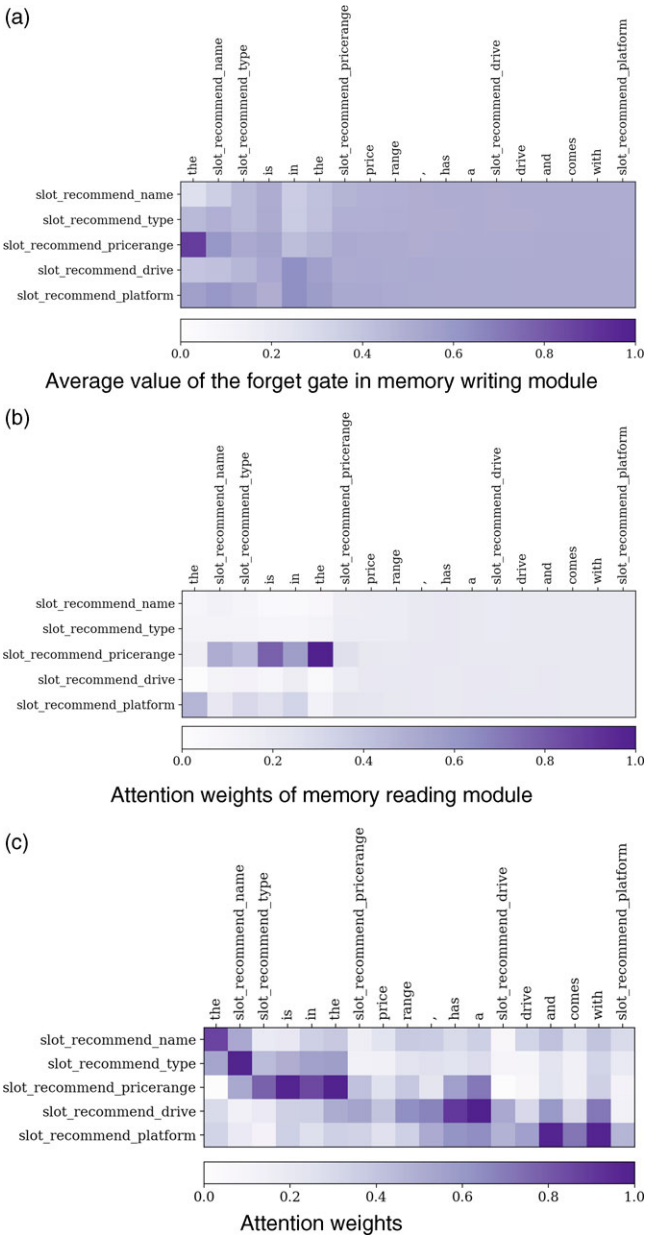

Figure 3. The change of the forget gate value in memory writing module (a), attention weights of memory reading module (b), and the attention weights (c) for DM-NLG multislot model after generating each word of the generated text for a given MR from the Laptop dataset.

7. Conclusion

In this paper, we presented our DM-NLG model to improve the performance of basic models based on the sequence-to-sequence structure in order to fully express the input information at the output. This model uses a dynamic memory that is initialized with the encoded input MR. The content of memory is rewritten and read based on the decoder's hidden state of each generated word; in this way, the memory at any time step contains the history of information used to

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

produce previous words, and this will prevent the generation of sentences with duplicate words or incomplete information. In this work, we considered two types of memories. In the first case, the memory has only one slot, which is initialized by the last encoder mode vector. In the second case, the number of memory slots is equal to the number of input meaning labels, and each slot is initialized by the hidden state of the encoder for its equivalent label. To improve the generated sentences quality by the decoder of DM-NLG, we used the pretrained language models which are first fine-tuned using training sentences and the generated training sentences by the DM-NLG model. Then each generated sentence for test data is regenerated by these fine-tuned models. To evaluate the performance of the proposed model versus the baseline models, we performed experiments on known datasets in the field of D2T. The subjective and objective metrics used for evaluation confirmed the superior performance of our proposed model. Also to compare the performance of the model in the case where we do not use memory versus the case where we use memory with one or more slots, we performed an experiment and its result showed that the quality of generated text will be better when multislot memory is used.

References

Angeli G., Liang P. and Klein D. (2010). A simple domain independent probabilistic approach to generation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, Cambridge, MA, pp. 502–512.

Barzilay R. and Lapata M. (2005). Collective content selection for concept-to-text generation. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, Vancouver, BC, Canada, pp. 331–338.

Barzilay R. and Lee L. (2004). Catching the drift: Probabilistic content models, with applications to generation and summarization. In *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, Seattle, WA, pp. 113–120.

Belz A. (2008). Automatic generation of weather forecast texts using comprehensive probabilistic generation-space models. *Natural Language Engineering* **14**(4), 431–455.

Chang E., Shen X., Zhu D., Demberg V. and Su H. (2021). Neural data-to-text generation with LM-based text augmentation, arXiv preprint arXiv:2102.03556.

Chen Z., Eavani H., Liu Y. and Wang W. Y. (2020). Few-shot nlg with pre-trained language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 183–190, Association for Computational Linguistics (Online).

Dai Z., Yang Z., Yang Y., Carbonell Z., Le Q. and Salakhutdinov R. (2019). Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, pp. 2978–2988.

Deriu J. and Cieliebak M. (2018). End-to-end trainable system for enhancing diversity in natural language generation. *E2E NLG Challenge System Descriptions*, Switzerland: Zurich University of Applied Sciences.

Devlin J., Chang M., Lee K. and Toutanova K. (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*, Minneapolis, MN, pp. 4171–4186.

Duboue P. A. and Mckeown R. K. (2003). Statistical acquisition of content selection rules for natural language generation. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, Sapporo, Japan, pp. 121–128.

Dusek O. and Jurcicek F. (2016). Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Berlin, Germany, pp. 45–51.

Dusek O., Novikova J. and Rieser V. (2018). Findings of the E2E NLG challenge. In *Proceedings of the 11th International Conference on Natural Language Generation*. The Netherlands: Tilburg University, pp. 322–328.

Gehrmann S., Dai F. Z. and Elder H. (2018). End-to-end content and plan selection for data-to-text generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, The Netherlands: Tilburg University, pp. 45–56.

Gong H. (2018). Technical report for E2E NLG challenge, *E2E NLG Challenge System Descriptions*, Switzerland: Zurich University of Applied Sciences.

Gyawali B. (2016). *Surface Realization from Knowledge Bases*, PhD Dissertation. University de Lorraine, Lorraine, France.

Harkous H., Groves I. and Saffari A. (2020). Have your text and use it too! End-to-end neural data-to-text generation with semantic fidelity. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 2410–2424.

Harrison V., Reed L., Oraby S. and Walker M. (2019). Maximizing stylistic control and semantic accuracy in NLG: Personality variation and discourse contrast. In *Proceedings of the 1st Workshop on Discourse Structure in Neural NLG*, Tokyo, Japan, pp. 1–12.

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

Iso H., Uehara Y., Ishigaki T., Noji H., Aramaki E., Kobayashi I., Miyao Y., Okazaki N. and Takamura H. (2019). Learning to select, track, and generate for data-to-text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, pp. 2102–2113.

Juraska J., Karagiannis P., Bowden K. and Walker M. (2018). A deep ensemble model with slot alignment for sequence-to-sequence natural language generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, New Orleans, LA, pp. 152–162.

Juraska J. and Walker M. (2021). Attention is indeed all you need: Semantically attention-guided decoding for data-to-text NLG. In *Proceedings of the 14th International Conference on Natural Language Generation*, Scotland, UK, pp. 416–431.

Kale M. and Rastogi A. (2020). Template guided text generation for task-oriented dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pp. 6505–6520, Association for Computational Linguistics (Online).

Kasner Z. and Dušek O. (2020). Data-to-text generation with iterative text editing. In *Proceedings of the 13th International Conference on Natural Language Generation*, Dublin, Ireland, pp. 60–67.

Kim J. H. and Mooney R. J. (2010). Generative alignment and semantic parsing for learning from ambiguous supervision, *Coling 2010: Posters*. Beijing, China, pp. 543–551.

Knight K. and Hatzivassiloglou V. (1995). Two-level, many paths generation. In *33rd Annual Meeting of the Association for Computational Linguistics*, Cambridge, MA, USA, pp. 252–260.

Konstas I. and Lapata M. (2012). Unsupervised concept-to-text generation with hypergraphs. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Montréal, Canada, pp. 752–761.

Konstas I. and Lapata M. (2013). Inducing document plans for concept-to-text generation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Seattle, WA, USA, pp. 1503–1514.

Kumar A., Irsoy O., Ondruska P., Iyyer M., Bradbury J., Gulrajani I., Zhong V., Paulus R. and Socher R. (2016). Ask me anything: Dynamic memory networks for natural language processing. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning*, New York, USA, pp. 1378–1387.

Langkilde I. and Knight K. (1998). Generation that exploits corpus-based statistical knowledge. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Montreal, QC, Canada, pp. 704–710.

Lavie A., Sagae K. and Jayaraman S. (2004). The significance of recall in automatic metrics for mt evaluation. In *Proceedings of the Machine Translation: From Real Users to Research*, Berlin, Heidelberg, pp. 134–143.

Lebret R., Grangier D. and Auli M. (2016). Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, TX, pp. 1203–1213.

Lee C., Krahmer E. and Wubben S. (2018). Automated learning of templates for data-to-text generation: Comparing rule-based, statistical and neural methods. In *Proceedings of the 11th International Conference on Natural Language Generation*, The Netherlands: Tilburg University, pp. 35–45.

Lee J. (2021). Transforming multi-conditioned generation from meaning representation, arXiv preprint arXiv:2101.04257.

Lewis M., Liu Y., Goyal N., Ghazvininejad M., Mohamed A., Levy O., Stoyanov V. and Zettlemoyer L. (2020). Bart: Denoising sequence-to-sequence pretraining for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7871–7880. Association for Computational Linguistics (Online).

Liang P., Michael J. I. and Klein D. (2009). Learning semantic correspondences with less supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, Suntec, Singapore, pp. 91–99.

Lin C. W. (2004). Rouge: A package for automatic evaluation of summaries, *Association for Computational Linguistics*. Barcelona, Spain, pp. 74–81.

Liu T., Wang K., Sha L., Chang B. and Sui Z. (2018). Table-to-text generation by structure aware seq2seq learning. In *The Thirty-Second AAAI Conferenceon Artificial Intelligence (AAAI-18)*, New Orleans, LA, USA: Hilton New Orleans Riverside, pp. 4881–4888.

Lu W. and Ng H. T. (2011). A probabilistic forest-to-string model for language generation from typed lambda calculus expressions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, Edinburgh, Scotland, UK, pp. 1611–1622.

Lu W., Ng H. T. and Lee W. S. (2009). Natural language generation with tree conditional random fields. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Singapore, pp. 400–409.

Malmi E., Krause S., Rothe S., Mirylenka D. and Severyn A. (2019). Encode, tag, realize: High-precision text editing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, Hong Kong, China, pp. 5054–5065.

Martin A. and Przybocki M. (2000). The nist 1999 speaker recognition evaluation-an overview. *Digital Signal Processing* **10**(1-3), 1–18.

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

Mei H. Y., **Bansal M. and Walter M.** (2016). What to talk about and how? Selective generation using LSTMs with coarse-to-fine alignment. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, San Diego, CA, pp. 720–730.

Nayak N., **Hakkani-Tur D.**, **Walker M. and Heck L.** (2017). To plan or not to plan? Discourse planning in slot-value informed sequence to sequence models for language generation. In *Conference of the International Speech Communication Association*, Stockholm, Sweden, pp. 3339–3343.

Novikova J., **Dusek O. and Rieser V.** (2017). The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, Saarbrücken, Germany, pp. 201–206.

Oraby O., **Reed L.**, **Tandon S.**, **S. T. S.**, **Lukin S. and Walker M.** (2018). Controlling personality-based stylistic variation with neural natural language generators. In *Proceedings of the 19th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, Melbourne, Australia, pp. 180–190.

Papineni K., **Roukos S.**, **Ward T. and Zhu W. J.** (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, PA, USA, pp. 311–318.

Peng B., **Zhu C.**, **Li C.**, **Li X.**, **Li J.**, **Zeng M. and Gao J.** (2020). Few-shot natural language generation for task-oriented dialog, arXiv preprint arXiv:2002.12328.

Pennington J., **Socher R. and Manning C. D.** (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, Doha, Qatar, pp. 1532–1543.

Puduppully R., **Dong L. and Lapata M.** (2019). Data-to-text generation with entity modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, pp. 2023–2035.

Qader R., **Portet F. and Labbe C.** (2019). Semi-supervised neural text generation by joint learning of natural language generation and natural language understanding models. In *Proceedings of the 1st Workshop on Discourse Structure in Neural NLG*, Tokyo, Japan, pp. 552–562.

Radford A., **Wu J.**, **Child R.**, **Luan D.**, **Amodei D. and Sutskever I.** (2019). Language models are unsupervised multitask learners. *OpenAI* **1**(8), 9, Technical report.

Raffel C., **Shazeer N.**, **Roberts A.**, **Lee K.**, **Narang S.**, **Matena M.**, **Zhou Y.**, **Li W. and Liu P.** (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research* **21**, 1–67.

Ringger E., **Gamon M.**, **Moore R. C.**, **Rojas D.**, **Smets M. and Corston O. S.** (2004). Linguistically informed statistical models of constituent structure for ordering in sentence realization. In *Proceedings of the 20th International Conference on Computational Linguistics*, Geneva, Switzerland, pp. 673–679.

Riou M., **Jabaian B.**, **Huet S. and Lefèvre F.** (2017). Online adaptation of an attention-based neural network for natural language generation. In *Conference of the International Speech Communication Association*, Stockholm, Sweden, pp. 3344–3348.

Sha L., **Mou L.**, **Liu T.**, **Poupart P.**, **Li S.**, **Chang B. and Sui Z.** (2018). Order-planning neural text generation from structured data. In *The Thirty-Second AAAI Conferenceon Artificial Intelligence (AAAI-18)*, New Orleans, LA, USA: Hilton New Orleans Riverside, pp. 5414–5421.

Shen X., **Chang E.**, **Su H.**, **Zhou J. and Klakow D.** (2020). Neural data-to-text generation via jointly learning the segmentation and correspondence. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7155–7165. Association for Computational Linguistics (Online).

Soricut R. and Marcu D. (2006). Stochastic language generation using WIDL-expressions and its application in machine translation and summarization. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, Sydney, Australia, pp. 1105–1112.

Tseng B., **Kreyssig F.**, **Budzianowski P.**, **Casanueva I.**, **Wu Y.**, **Ultes S. and Gasic M.** (2018). Variational cross-domain natural language generation for spoken dialogue systems. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, Melbourne, Australia, pp. 338–343.

Tran V. K. and Nguyen L. M. (2017). Natural language generation for spoken dialogue system using RNN encoder-decoder networks. In *Proceedings of the 21st Conference on Computational Natural Language Learning*, Vancouver, Canada, pp. 442–451.

Tran V. K. and Nguyen L. M. (2018a). Adversarial domain adaptation for variational neural language generation in dialogue systems. In *Proceedings of the 27th International Conference on Computational Linguistics*, Santa Fe, NM, USA, pp. 1205–1217.

Tran V. K. and Nguyen L. M. (2019). Gating mechanism based natural language generation for spoken dialogue systems. *Nerocomputing* **325**, 48–58.

Tran V. K., **Nguyen V. T. and Nguyen L. M.** (2017). Enhanced semantic refinement gate for RNN-based neural language generator. In *9th International Conference on Knowledge and Systems Engineering*, Hue, Vietnam, pp. 172–178.

Tran V. K. and Nguyen L. M. (2018b). Dual latent variable model for low-resource natural language generation in dialogue systems. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, Brussels, Belgium, pp. 21–30.

Vaswani A., **Shazeer N.**, **Parmar N.**, **Uszkoreit J.**, **Jones L.**, **Gomez A.**, **Kaiser L. and Polosukhin I.** (2017). Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, New York, USA, pp. 6000–6010.

https://doi.org/10.1017/S1351324923000207 Published online by Cambridge University Press

Wen T. H., **Gasic M.**, **Kim D.**, **Mrksic N.**, **Su P. H.**, **Vandyke D. and Young S.** (2015a). Stochastic language generation in dialogue using recurrent neural networks with convolutional sentence reranking. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, Prague, Czech Republic, pp. 275–284.

Wen T. H., **Gasic M.**, **Mrksic N.**, **Rojas Barahona L. M.**, **Su P. H.**, **Vandyke D. and Young S.** (2015b). Toward multi-domain language generation using recurrent neural networks. *NIPS Workshop on Machine Learning for Spoken Language Understanding and Interaction*. Hanoi, Vietnam.

Wen T. H., **Gasic M.**, **Mrksic N.**, **Su P. H.**, **Vandyke D. and Young S.** (2015c). Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, pp. 1711–1721.

Wen T. H., **Gasic M.**, **Mrksic N.**, **Rojas Barahona L. M.**, **Su P. H.**, **Vandyke D. and Young S.** (2016). Multi-domain neural network language generation for spoken dialogue systems. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, San Diego, CA, pp. 120–129.

Wolf T., **Debut L.**, **Sanh V.**, **Chaumond J.**, **Delangue C.**, **Moi A.**, **Cistac P.**, **Rault T.**, **Louf R.**, **Funtowicz M.**, **Davison J.**, **Shleifer S.**, **Platen P.**, **Ma C.**, **Jernite Y.**, **Plu J.**, **Xu C.**, **Le Scao T.**, **Gugger S.**, **Drame M.**, **Lhoest Q. and Rush A. M.** (2020). Huggingface's transformers: State-of-the-art natural language processing, *Association for Computational Linguistics*, pp. 38–45, Association for Computational Linguistics (Online).

Wong Y. W. and Mooney R. J. (2007). Generation by inverting a semantic parser that uses statistical machine translation. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, Rochester, NY, pp. 172–179.

Zhang B., **Yang J.**, **Lin Q. and Su J.** (2018). Attention regularized sequence-to-sequence learning for E2E NLG challenge, *E2E NLG Challenge System Descriptions*, Switzerland: Zurich University of Applied Sciences.

Zhang T., **Kishore V.**, **Wu F.**, **Weinberger Q. K. and Artzi Y.** (2020). Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*, Addis Ababa, Ethiopia: OpenReview.net.

Cite this article: Seifossadat E and Sameti H. Improving semantic coverage of data-to-text generation model using dynamic memory networks. *Natural Language Engineering* https://doi.org/10.1017/S1351324923000207