Dynamic Topic Tracker for KB-to-Text Generation[*]

Zihao Fu[1], Lidong Bing[2], Wai Lam[1], Shoaib Jameel[3]
[1]Department of Systems Engineering and Engineering Management,
The Chinese University of Hong Kong
[2]DAMO Academy, Alibaba Group
[3]School of Computer Science and Electronic Engineering, University of Essex
zhfu@se.cuhk.edu.hk; l.bing@alibaba-inc.com;
wlam@se.cuhk.edu.hk; shoaib.jameel@essex.ac.uk

Abstract

Recently, many KB-to-text generation tasks have been proposed to bridge the gap between knowledge bases and natural language by directly converting a group of knowledge base triples into human-readable sentences. However, most of the existing models suffer from the off-topic problem, namely, the models are prone to generate some unrelated clauses that are somehow involved with certain input terms regardless of the given input data. This problem seriously degrades the quality of the generation results. In this paper, we propose a novel dynamic topic tracker for solving this problem. Different from existing models, our proposed model learns a global hidden representation for topics and recognizes the corresponding topic during each generation step. The recognized topic is used as additional information to guide the generation process and thus alleviates the off-topic problem. The experimental results show that our proposed model can enhance the performance of sentence generation and the off-topic problem is significantly mitigated.

1 Introduction

In recent years, many knowledge bases (KBs) have been built to incorporate different kinds of human knowledge into a structured triple representation such as Freebase (Bollacker et al., 2008), DBpedia (Auer et al., 2007), YAGO (Suchanek et al., 2007) and Wikidata (Vrandečić and Krötzsch, 2014). Many tasks including question answering and recommendation systems have benefited from KBs (Wang et al., 2017) as external knowledge sources to improve the results. Though KBs have achieved great success in supporting and improving various text mining tasks, they are still incomprehensible to humans due to the over-rigid structured format. Reading a bunch of triples always annoys people since the form is not easily understandable especially to people who have never heard about KBs. In order to address this problem, recently some researchers have proposed the KB-to-text generation task (Lebret et al., 2016; Gardent et al., 2017a; Gardent et al., 2017b) to bridge the gap between KBs and natural language. This KB-to-text generation problem aims at directly converting a group of KBs triples into human-readable sentences. For example, given a triple group (⟨Bill Gates, BirthPlace, Seattle ⟩, ⟨Bill Gates, FounderOf, Microsoft ⟩), the goal is to generate a comprehensible sentence such as "*Bill Gates, the founder of Microsoft, was born in Seattle.*"

Some works employ the techniques in the text generation area (Gatt and Krahmer, 2018) to tackle the KB-to-text problem. Though these models have achieved some success, there are still quite many limitations. One major drawback of existing models is that most of them suffer from the off-topic problem. Consider the example given in Figure 1, the topic of the target sentence is expected to change from "person" to "company" in the generation process. However, a model is prone to generate unrelated off-topic clauses like "*Bill is a commonly used name in the USA.*" which is not consistent with the given input data and we recognize this phenomenon as the off-topic problem. This is because during the training

[*] The work described in this paper is substantially supported by a grant from the Research Grant Council of the Hong Kong Special Administrative Region, China (Project Codes: 14204418).

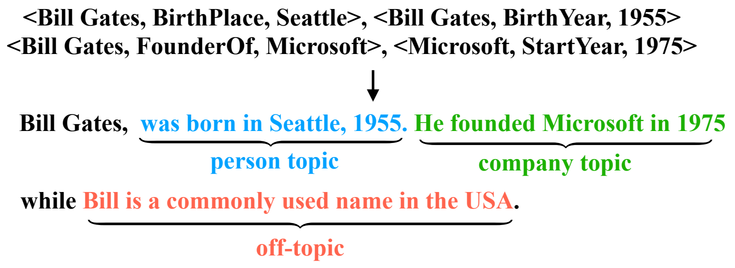
<Bill Gates, BirthPlace, Seattle>, <Bill Gates, BirthYear, 1955>
<Bill Gates, FounderOf, Microsoft>, <Microsoft, StartYear, 1975>
Bill Gates, was born in Seattle, 1955. He founded Microsoft in 1975
person topic
company topic
while Bill is a commonly used name in the USA.
off-topic

Figure 1: The off-topic problem. During the generation process, the given data ranges from the topic of person topic to the topic of company. However, the models are prone to generate off-topic sentences just because the models associate this kind of information with the word "Bill".

stage, the models associate this kind of information with some input words like "Bill". In the testing or operational stage, when these words occur in the given data, the models are prone to generate off-topic sentences related to these words regardless of the given data.

To solve the off-topic problem, we propose to utilize the topic information as a piece of clue in the sentence generation process. Unfortunately, the corresponding topic information is not available and there is no existing dataset containing the topic annotations. Therefore, it is difficult to adopt supervised learning approaches for detecting topics. Moreover, it is even more expensive to annotate the dynamic change of topic information in one sentence, as exemplified by the topic changes from "person" to "company" in Figure 1. Therefore, we investigate the task of automatically detecting the hidden topic information and incorporating such information for the generation of sentences. Many works have been proposed to utilize the static topic information to improve the generation performance. Chen et al. (2016) and Ou et al. (2018) propose to represent the topic for each sentence as a learnable vector. The topic is predicted by the input sentence and is used to enhance the generating phase. Xing et al. (2017) and Zhang et al. (2016) detect the topic representation by applying a pre-trained LDA model on the input sequence. Moreover, Choudhary et al. (2017) and Ou et al. (2018) predict the topic representation directly from the input sequence using Recurrent Neural Networks (RNN). All the above methods make an assumption that during generation the topic does not change so as to make the problem tractable, which scarifies the advantage of modeling the dynamic nature of topic information.

We propose a novel Dynamic Topic Tracker (DTT) neural model to tackle the problem. Different from existing models, our proposed DTT model learns how the target sentence topic dynamically evolves and how to use the topic information to guide the generation process simultaneously. Specifically, our DTT model is a neural model composed of four parts, namely, the state tracker, the topic attention, the global topic bank, and the topic memory. The state tracker captures the decoder state for each generation step. The topic attention uses the captured decoder state to focus on the input hidden representation to get a local topic state. The topic bank learns a global hidden topic representation and it calculates the most suitable local topic representation for each local topic state. The topic memory is used to memorize the previous local topic representation and computes the dynamic topic state for each generation step to guide the target sentence generation procedure.

2 Related Work

Recently, various data-to-text tasks have been proposed handling different kinds of data. Gardent et al. (2017a; Gardent et al. (2017b) construct the WebNLG dataset which aims at generating text descriptions based on DBpedia (Auer et al., 2007) triples. Lebret et al. (2016) and Chisholm et al. (2017) propose to generate a person's biography based on Wikipedia's infobox. Fu et al. (2020a) build the WikiEvent dataset aiming at generating text based on an event chain. Novikova et al. (2017) generate restaurant reviews based on the information of restaurant attributes. Wiseman et al. (2017) generate basketball match descriptions based on the game records. Moreover, Fu et al. (2020c) propose to directly train the model on partially-aligned data called WITA while Fu et al. (2020b) propose to train a model based on purely unaligned data unsupervised with a dual learning framework. All of the above problems aim at converting some formatted data into natural language texts facilitating more understandability.

Some models have proposed to solve the KB-to-text problem by utilizing various information of the KBs. Chisholm et al. (2017) propose to directly rank the triples by relation frequency and flatten the triples to pure text. The flattened text is used as the input for a sequence-to-sequence model to generate the output text. Vougiouklis et al. (2018) propose to use a triple encoder to encode each triple into a hidden vector. The decoder input is constructed by simply concatenating all of the hidden vectors. Trisedya et al. (2018) propose a GTR-LSTM model to encode not only the triple information, but also the structure information of the entity graph into hidden semantic space. Jain et al. (2018) exploit a mixed hierarchical attention based encoder-decoder model to leverage the structure and content information. Shimorina and Gardent (2018) propose to use delexicalization and copy mechanism to enhance the performance of the sequence-to-sequence framework. Konstas and Lapata (2013) and Wiseman et al. (2018) propose to use template based methods to generate the text by using the extracted template information in the training set. Cheng et al. (2020) propose to generate text description for entities by utilizing the knowledge distilled from the existing knowledge base. However, none of the above works consider the topic information in the KB-to-text generation process and thus not directly comparable to our work proposed in this paper.

Some works in text generation (Gatt and Krahmer, 2018) have been proposed to incorporate the topic information to help generate the text. These ideas can be adopted in KB-to-text generation. Tars and Fishel (2018) and Johnson et al. (2017) add an extra topic tag into the source sentence for incorporating the topic information into the model. The whole model is built based on the sequence-to-sequence (Sutskever et al., 2014; Cho et al., 2014; Klein et al., 2017) framework with standard attention (Bahdanau et al., 2014; Luong et al., 2015). Mikolov and Zweig (2012) as well as Liu et al. (2015) propose to use the topic information as extra features to enhance the performance of the language model and word embedding. Chen et al. (2016) and Ou et al. (2018) use the same idea to utilize the topic feature to enhance the generation of the text. However, all these methods assume that the topic information is known in advance.

Some methods investigate the problem setting that the topic information is not given and needs to be detected. For example, the topic information can be detected from Latent Dirichlet Allocation (LDA). Zhang et al. (2016; Dziri et al. (2018; Wang et al. (2019) detect the topic distribution of words via topic model to enhance the translation procedure. Xing et al. (2017) propose a TA-Seq2Seq framework which uses the word topic information from LDA to generate the responses in chatbot dialog systems. Moreover, some researchers propose to directly detect the topic vector from the input sentences in the sequence-to-sequence framework. For example, Choudhary et al. (2017) propose to train a classifier to predict the topic of the source sentence and use it to help generate the dialog response. Ou et al. (2018) also propose to predict the topic vector directly from the input sequence. Dathathri et al. (2020) propose to use the topic information as a reward function. However, none of the existing works can capture the dynamic topic information suitable for the KB-to-text generation problem.

3 Method

3.1 Our Framework

The KB-to-text generation task aims to generate one or more sentences based on a given set of triples. For example, given a triple group $\{\langle$ Bill Gates, BirthPlace, Seattle$\rangle$, $\langle$Bill Gates, FounderOf, Microsoft $\rangle\}$ as input, we aim at generating a sentence such as "Bill Gates, the founder of Microsoft, was born in Seattle.". Formally, the input is a set of triples which can be denoted as $\{\langle h_1, r_1, t_1\rangle, \langle h_2, r_2, t_2\rangle, \cdots, \langle h_{\tilde{n}}, r_{\tilde{n}}, t_{\tilde{n}}\rangle\}$, in which h_i, r_i, t_i stands for the ith head, relation and tail entity respectively. $\tilde{n}$ is the number of the triples. The goal is to maximize the conditional probability of the generated text $(s_1, s_2, \cdots, s_m)$ given such input in the training set. We denote $k_i = \langle h_i, r_i, t_i\rangle$, The problem can be expressed as:

$$\max_{\theta} p_{\theta}(s_1, s_2, \cdots, s_m | \{k_1, k_2, \cdots, k_{\tilde{n}}\}),$$

in which θ denotes all the parameters in the model and m is the length of the generated text.

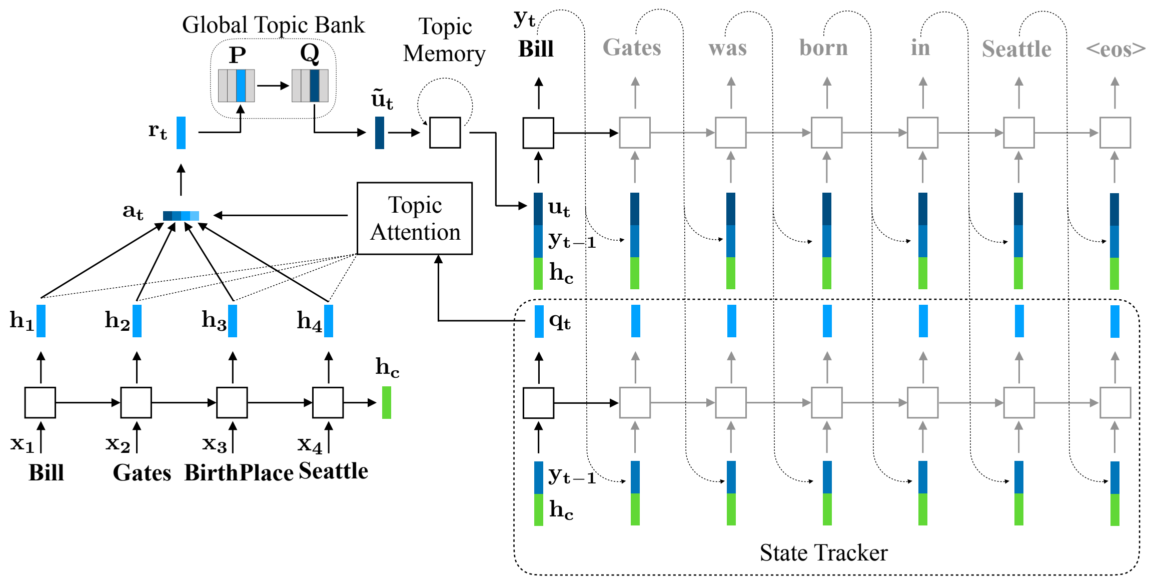

Figure 2: Overview of our framework. Due to the limited space, we omit the traditional attention layer. This figure shows the first time step of the decoding process.

Our proposed framework is shown in Figure 2. It is built on top of a sequence-to-sequence neural structure. The encoder is similar to the standard sequence-to-sequence model while the decoder is equipped with the novel Dynamic Topic Tracker (DTT) model.

Following the idea of (Chisholm et al., 2017), we construct the input by listing all words in the triple elements one after another (i.e. $[h_1, r_1, t_1, h_2, r_2, t_2, \cdots, h_{\tilde{n}}, r_{\tilde{n}}, t_{\tilde{n}}]$) to construct a sequence which is denoted as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$, $\mathbf{X} \in \mathbb{R}^{d \times n}$ in which d is the embedding size and n is the length of all input words. $\mathbf{x}_i$ is the embedding for each word. Afterwards, the sequence is encoded by an encoder (left bottom part of Figure 2) into a hidden context vector $\mathbf{h}_c$. We use a stacked LSTM layer as the encoder:

$$\mathbf{H} = \text{LSTM}(\mathbf{X}),$$

where $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_n]$, $\mathbf{H} \in \mathbb{R}^{h \times n}$ is the output hidden matrix of the input sequence in which each column is the hidden vector representation of an input word. h is the size of the output hidden vectors. $\mathbf{h}_c = \mathbf{h}_n \in \mathbb{R}^h$ is the last element of $\mathbf{H}$ which can be regarded as a hidden representation of the full input sequence.

We design a novel decoder (depicted on the right hand side of Figure 2) which exploits the dynamic topic information for each generation step. This decoder generates the output sentence based on the hidden context vector $\mathbf{h}_c$ and the last time step's output vector $\mathbf{y}_{t-1} \in \mathbb{R}^d$. Therefore, the input and the output are similar to the standard decoder. However, different from traditional decoders, our proposed new decoder contains the DTT model which is capable of detecting the dynamic topics and incorporating these topic vectors into the generation process. More description for DTT will be presented in the next sub-section. The new decoding procedure can be expressed as:

$$\begin{aligned}
\mathbf{u_t} &= \text{DTT}([\mathbf{h}_c; \mathbf{y}_{t-1}]) \\
\mathbf{z}_t &= \text{LSTM}([\mathbf{h}_c; \mathbf{y}_{t-1}; \mathbf{u}_t]) \\
\mathbf{y}_t &= \text{Attn}(\mathbf{z}_t, \mathbf{H}),
\end{aligned}$$

in which $[\mathbf{h}_c; \mathbf{y}_{t-1}] \in \mathbb{R}^{h+d}$ is the concatenation of $\mathbf{h}_c$ and $\mathbf{y}_{t-1}$. The DTT model detects the topic vector $\mathbf{u}_t$ for the current generation step based on this input. $[\mathbf{h}_c; \mathbf{y}_{t-1}; \mathbf{u}_t] \in \mathbb{R}^{h+d+u}$ is the concatenation of the three vectors. It is used as the input vector for the following decoder layer. u is the size of the dynamic topic state vector $\mathbf{u}_t$. Attn is a commonly used attention layer for the decoder which is similar with (Luong et al., 2015).

3.2 Dynamic Topic Tracker (DTT)

Our proposed DTT aims at capturing the dynamic topic information in each generation step. It represents each topic by a vector which will be learned during training. When given a new set of triples, the model will automatically find the most suitable topic vector for each generation step and use it to guide the prediction of the output words. It contains four components namely the state tracker, the topic attention component, the global topic bank, and the topic memory. The state tracker obtains a hidden representation of the current decoder step's state. The topic attention component utilizes this state to get a local topic state vector. This vector is then fed into the global topic bank to get the local topic representation of the current local topic state. Afterwards, the local topic representation is sent to the topic memory which get a dynamic topic state based on the existing previous states. The details of each component are described in the following sub-sections.

3.2.1 State Tracker

The state tracker is used to capture the status of the current generation step. It is composed of several stacked LSTM layers and will output a state vector. The input sentence hidden vector $\mathbf{h}_c$ and the output of the last time step $\mathbf{y}_{t-1}$ are concatenated and fed to a stacked LSTM layer which can be expressed as:

$$\mathbf{q}_t = \text{LSTM}([\mathbf{h}_c; \mathbf{y}_{t-1}]),$$

in which $\mathbf{q}_t \in \mathbb{R}^h$ is the hidden representation of the current state and is used to calculate the topic representation by the following topic attention component. The state tracker is similar to a decoder. However, there is no dropout layer for the output and the state tracker is trained to capture the state of the current generation step rather than directly predicting the output representation.

3.2.2 Topic Attention

The topic attention uses the current state $\mathbf{q}_t$ of the state tracker to calculate the attention of each input sequence's hidden representation $[\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_n]$. A relevance score is calculated as:

$$\begin{aligned}\tilde{\mathbf{q}}_t &= \mathbf{W}_q \mathbf{q}_t + \mathbf{b}_q \\ \mathbf{c}_t &= \mathbf{H}^T \tilde{\mathbf{q}}_t,\end{aligned}$$

in which $\tilde{\mathbf{q}}_t \in \mathbb{R}^h$ is the transformed state vector and $\mathbf{W}_q \in \mathbb{R}^{h \times h}$, $\mathbf{b}_q \in \mathbb{R}^h$ is the transformation matrix and the corresponding bias vector. $\mathbf{c}_t \in \mathbb{R}^n$ is calculated by a simple vector inner product between $\tilde{\mathbf{q}}_t$ and each $\mathbf{h}_i$. Each element in $\mathbf{c}_t$ is the similarity score for each $\mathbf{h}_i$. Afterwards, the score vector $\mathbf{c}_t$ is sent to a softmax layer to calculate the normalized attention to each $\mathbf{h}_i$:

$$\mathbf{a}_t[i] = \frac{e^{\mathbf{c}_t[i]}}{\sum_{j=1}^{n} e^{\mathbf{c}_t[j]}},$$

in which $\mathbf{a}_t$ is the normalized attention and $\mathbf{a}_t[i]$ is the ith element of $\mathbf{a}_t$. Finally, the topic state is calculated by a weighted sum of each $\mathbf{h}_i$ by the attention vector $\mathbf{a}_t$:

$$\mathbf{r}_t = \mathbf{H}\mathbf{a}_t.$$

$\mathbf{r}_t$ is the local topic state vector indicating the topic state of the current decoder. The topic attention has a similar structure with traditional attention. The difference is that the output $\mathbf{r}_t$ will be used to find a new topic vector in the topic bank to make the topic representation more general for each kind of sentence.

3.2.3 Global Topic Bank

The global topic bank acts as a database to store the trained hidden topic representation which is used by all sentences. Note that different from the traditional topic representation which is a word distribution (Blei et al., 2003), our topics are represented by dense vectors. It consists of two matrices of the same size, namely $\mathbf{P}, \mathbf{Q} \in \mathbb{R}^{u \times l}$, in which u is the size of the topic vector and l is the number of the topic vectors. l is a hyperparameter that can be assigned by the user. When $\mathbf{r}_t$ is calculated, it will be used to calculate a similarity score for each topic by a simple vector inner product. The calculation is as follows:

$$\tilde{\mathbf{w}}_t = \mathbf{P}\mathbf{r}_t,$$

in which $\tilde{\mathbf{w}}_t \in \mathbb{R}^l$ is a score vector indicating the similarity score of each topic to $\mathbf{r}_t$. It will also be normalized with a softmax function to get the probability of each topic:

$$\mathbf{w}_t[i] = \frac{e^{\tilde{\mathbf{w}}_t[i]}}{\sum_{j=1}^{l} e^{\tilde{\mathbf{w}}_t[j]}}.$$

The matrix $\mathbf{P}$ can be regarded as a projection matrix which spans in the topic state semantic space. After the projection on $\mathbf{P}$, the information in $\mathbf{r}_t$ that is irrelevant to the topic will be eliminated. The topic is represented by the combination coefficients of topics instead of a simple vector. Afterwards, the topic representation is obtained by the weighted sum of each topic representation with the probability vector $\mathbf{w}_t$:

$$\tilde{\mathbf{u}}_t = \mathbf{Q}\mathbf{w}_t,$$

in which $\tilde{\mathbf{u}}_t$ is the local topic representation. It should be noted that here we use another matrix $\mathbf{Q}$ to get the final topic representation. The reason is that $\mathbf{P}$ and $\mathbf{Q}$ are the representation of topics in different semantic spaces. Specifically, $\mathbf{P}$ represents topics in the topic state space for the state tracker while $\mathbf{Q}$ represents topics in the topic representation space for the generation process.

3.2.4 Topic Memory

Since $\tilde{\mathbf{u}}_t$ is calculated only by the current decoder state, it may lack some historical information. For some words that have no obvious topic information such as "the", "have", it is necessary to refer to the topic of the last step. Therefore, we design a topic memory component to help keep the state of the history topic and it can be used to help build the current topic information. We use a simple RNN to help memorize the history information. The topic memory can be expressed as:

$$\mathbf{u}_t, \tilde{\mathbf{h}}_t = \text{RNN}(\tilde{\mathbf{u}}_t, \tilde{\mathbf{h}}_{t-1}),$$

in which $\mathbf{u}_t$ is the dynamic topic state vector and $\tilde{\mathbf{h}}_t$ is the hidden state of the RNN. The history topic information is stored in $\tilde{\mathbf{h}}_t$ and will be passed to help the next step's generation.

4 Experiments

4.1 Dataset

We evaluate our framework on the release v2 of KB-to-text generation dataset WebNLG (Gardent et al., 2017a) [1]. WebNLG dataset contains KB triples and their associated sentences. The KB triples are sampled from DBPedia, while the corresponding text is written by crowdsourcing. Each sentence is also given a domain tag (e.g. Building, City and etc.). The statistics of the dataset is shown in Table 1.

	Train	Dev	Test
Size	34,352	4,316	4,224

Table 1: Statistics of WebNLG dataset.

4.2 Comparison Models

We compare our DTT with several topic based models. Our main focus will be on those models that can detect topic information (e.g., LDA-S2S and T2S) and consider dynamic topics such as DLDA-S2S. We also compare with models that use additional given topic information (e.g., TopicTag and TopicFeature).

[1] `https://gitlab.com/shimorina/webnlg-dataset`

	BLEU	ROUGE$_L$	NIST	METEOR	CIDEr
S2S	0.5444(1.2e-2)	0.6608(9.9e-3)	10.04(2.0e-1)	0.3990(7.0e-3)	3.525(9.4e-2)
TopicTag	0.5558(1.0e-2)	0.6732(8.1e-3)	10.18(1.6e-1)	0.4021(5.8e-3)	3.579(1.0e-1)
TopicFeature	0.5501(**4.9e-3**)	0.6667(6.5e-3)	10.09(**6.8e-2**)	0.3989(2.9e-3)	3.527(6.6e-2)
LDA-S2S	0.5591(1.3e-2)	0.6689(1.2e-2)	10.22(2.1e-1)	0.4046(8.2e-3)	3.571(1.3e-1)
DLDA-S2S	0.5650(9.2e-3)	0.6809(5.3e-3)	10.20(1.2e-1)	0.3998(4.6e-3)	3.556(7.9e-2)
T2S	0.5592(8.8e-3)	0.6710(9.2e-3)	10.19(1.1e-1)	0.4034(4.8e-3)	3.552(7.1e-2)
DTT	**0.5704**(5.9e-3)	**0.6823**(**3.5e-3**)	**10.39**(7.6e-2)	**0.4116**(**2.2e-3**)	**3.703**(**3.3e-2**)
DTT w/o memory	0.5680(7.3e-3)	0.6801(4.6e-3)	10.35(9.0e-2)	0.4099(3.9e-3)	3.672(7.3e-2)

Table 2: Main results. The parentheses give the standard deviations. The best score and the smallest variance are marked in bold font.

We implement all the baseline models to make them more comparable with each other. The comparison models are as follows:

S2S follows the model proposed by Shimorina and Gardent (2018) which uses a standard sequence-to-sequence model with attention.

TopicTag: follows the model proposed by Johnson et al. (2017; Tars and Fishel (2018). It utilizes additional information namely the domain tag provided by the dataset for each sentence to serve as the topic information. Precisely, the domain tag of each sentence is added as a new word at the end of each sentence similar to Johnson et al. (2017; Tars and Fishel (2018). The modified sentences are then fed into the S2S model.

TopicFeature follows the model proposed by (Chen et al., 2016; Ou et al., 2018), this model learns a vector representation for each domain tag rather than just adding it as an additional word. For each domain, we denote it as a one-hot vector and the one-hot vector is fed into a feedforward layer to get the topic membership vector, which is then used as the extra feature to predict the decoder output.

LDA-S2S follows the model proposed by Zhang et al. (2016; Xing et al. (2017), we first train an LDA model on the source sentences of the training dataset. Then, the sentence topic is calculated by averaging the topic distribution on each word and it is used as an extra context feature in the decoder similar to the TopicFeature model.

DLDA-S2S follows the model proposed by Mikolov and Zweig (2012; Dziri et al. (2018), we dynamically calculate the topic distribution for each word by summing the source word vectors weighted by the attention. The word topic distribution is calculated similarly to LDA-S2S.

T2S follows the model proposed by Choudhary et al. (2017; Ou et al. (2018). The topic distribution is predicted based on the input sentence hidden vector $\mathbf{h}_c$. $\mathbf{h}_c$ is fed into several linear layers to get the fixed topic representation. Then the fixed topic representation is used as a new context feature in each step of the decoder.

4.3 Experiment Setup

We implement our model based on OpenNMT-py[2], a Python port of OpenNMT (Klein et al., 2017). All the hyper-parameters are tuned on the dev set with grid search. We follow the baseline model's default settings (Gardent et al., 2017a; Gardent et al., 2017b), in which the word embedding size is 500. The size of LSTM hidden vector states is set to 500. We use two layers of LSTMs for both encoder and decoder. For the last layer of the stacked LSTM, we add a dropout layer with a ratio of 0.3. We use SGD as our optimizer with the initial learning rate of 1.0 and the decay rate of 0.5. The commonly-used attention (Luong et al., 2015) is added to all models. When generating a new sentence, we use beam search with beam size 5 which is a traditional setting for generation tasks. We tune the number of topics of our model and comparison models using the dev set. In the TopicFeature model, the number of topics is set to 20. In LDA-S2S, we train the LDA model with the Python based LDA package[3] and

[2]`https://github.com/OpenNMT/OpenNMT-py`

[3]`https://pypi.org/project/lda/`

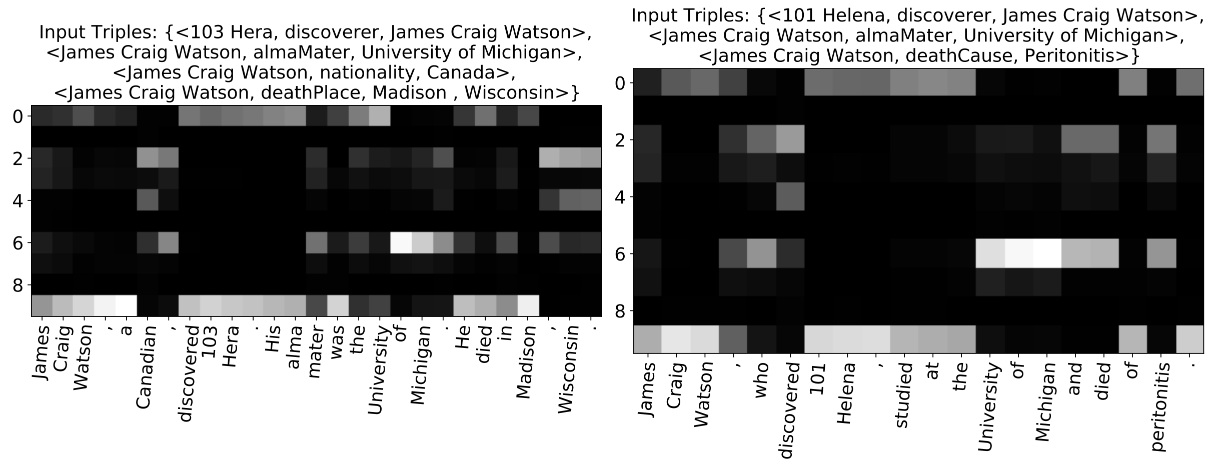

Figure 3: Topic distribution at each step. White stands for higher probability while black stands for lower.

the number of topics is set to 100. In the T2S model, we set the number of topics to 200. In the DTT model, we set the number of topics to 500. We evaluate all the models with the same evaluation script [4]. Several metrics are evaluated, including BLEU (Papineni et al., 2002), ROUGE$_L$ (Lin, 2004), NIST (Doddington, 2002), METEOR (Banerjee and Lavie, 2005) and CIDEr (Vedantam et al., 2015). Since some metrics are sensitive to randomness, we run each model for 5 times and report the median score with the standard deviation.

4.4 Results

The experimental results are shown in Table 2. We can observe that our DTT model outperforms all comparison models significantly and consistently. It illustrates that the DTT model can capture the dynamic topic information to mitigate the off-topic problem and thus improves the overall generation performance. Besides, our DTT model not only improves performance but also improves the stability of the performance. It can be observed that the standard deviation is almost reduced to half of that in the S2S model and is the smallest in most of the metrics. These results show that our proposed DTT model is more robust against the randomness when generating sentences by incorporating the dynamic topic information.

The T2S model's results show that simply using several linear layers to learn topic information performs no worse than models with annotated tags or pre-trained topic allocation. It illustrates that pure neural models can learn reasonable topic representation. The T2S model performs not as good as our DTT model. The main reason is that it predicts the topic directly by the input sequence and the topic is fixed during the whole generating process. The result of DLDA-S2S with dynamic topic vectors is also better than its counterpart with static topics, i.e. LDA-S2S. All these results show that capturing the dynamic topic can provide more suitable information for text generation.

The TopicTag model and the TopicFeature model outperform the S2S model. However, they fail exceeding other models since the domain tags only give very general and insufficient topic information for each sentence. Therefore, even learning topic information from scratch outperforms these models using domain tags.

We conduct an ablation experiment by investigating our model without the previous topic information (denoted as DTT w/o memory). The performance decreases slightly. It indicates that the topic memory can utilize the historical topic information for learning a better representation for the current topic. Without the topic memory, for those words without explicit topic information, the decoder loses the record of what the current topic is. Besides, the standard deviation increases slightly in all metrics showing that the topic memory also makes the model more robust.

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

Input Triple	S2S Output	DTT Output
⟨Sumatra, ethnicGroup, Malays (ethnic group)⟩	Asam pedas is from the Sumatra .	The Minangkabau people are an ethnic group of Sumatra .
⟨109 Felicitas, discoverer, Christian Heinrich Friedrich Peters⟩	Aleksandra Kovač ' s musical genre is the House musician .	The celestial body known as 109 Felicitas was discovered by Christian Burns .
⟨Live Nation Entertainment, location, Beverly Hills California⟩	The owner of the government of Aarhus is The location of government .	The 3Arena is located in North Wall , California and it is owned by Live Nation Entertainment .
⟨Castle novel, language, English language⟩	The novel Owen Glendower is a notable team .	The official language in Poland is the English language .

Table 3: Case study.

4.5 Topic Evolving Analysis

To show that our DTT model does capture the evolving of the topics, we sample two sets of similar triples and generate the corresponding text to illustrate the evolving procedure. We set the topic number to 10 and retrain the model for the sake of easier illustration. We record the topic distribution for each generation step and observe how the topic distribution is changing. The result is shown in Figure 3. At each step, our DTT model predicts one of the topics with a very high probability while other topics only have relatively low probability. This observation complies with our intuition of topics. When we talk about some facts within one sentence, the sentence may contain several topics, but at one time, only one topic is dominating. The dominating topic changes during the generation procedure which illustrates that our model captures the changes of the hidden topics when generating from one triple to another. There are some major topics in each group of triples. For example, the sentences generated in the figure mainly talk about a person. The main topic for both of them is Topic 9. It can also be observed that some topics are allocated to the same place in the two sentences. For example, Topic 6 captures the education background while Topic 9 captures the discovery event.

4.6 Case Study

In order to give an intuitive illustration of the off-topic problem and show how this problem is alleviated by our model, we sampled some challenging cases handled by the S2S model together with the result produced by our DTT model. These challenging cases perform not well in both models. Nevertheless, the sentences generated by the DTT model seem much better. The result is shown in Table 3. All these triples are very challenging to handle. Therefore, both models cannot generate the sentence perfectly. However, since the DTT model are guided by the topic information, all the topics of the generated sentences are reasonable, though some generated terms are not quite correct. The S2S model is prone to be misled by some keywords and generates unrelated sentences. For example, consider the triple ⟨ Sumatra, ethnicGroup, Malays (ethnic group) ⟩, the S2S model is misled by the beginning tag "Sumatra" and generates a sentence that is totally unrelated to the triple. This result may be caused by the fact that the training set contains many examples related to "Sumatra" and "Asam pedas". In contrast, our DTT model is guided by the detected topic information of each decoding step making it more likely to predict the correct sentence. Such topic information also makes it more robust thus the decoder is less likely to generate sentences randomly. This observation to some extent explains why the performance of the DTT model only has half of the standard deviation of that in the S2S model.

5 Conclusions and Future Work

In this paper, we recognize the off-topic problem in the KB-to-text problem. We consider to utilize the dynamic topic information to alleviate this problem and improve the generation performance. To achieve this, we propose a DTT model which can learn the hidden representation of the topic information. During the sentence generating process, it can utilize the learned topic information in the sequence-to-sequence framework for enhancing the generation process. More importantly, the topic information is dynamic for each step in the generation process, and thus enables stronger capability than existing works. Experimental results on a benchmark dataset show that our model can effectively capture the dynamic topic information at each step in the decoder.

Despite the promising result our model has achieved, there are some remaining challenges: (1) The model stacks too many LSTM layers which leads the gradient hard to back-propagate if more layers are going to be added. Some new technologies such as the Transformer or gated CNN can be used to tackle this problem. (2) The topic representation only contains information from our training set. Nevertheless, the novel architecture makes it possible to use any set of the topic vectors which can be pre-trained on a larger unannotated dataset.

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference*, pages 722–735.

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

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

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 1247–1250.

Wenhu Chen, Evgeny Matusov, Shahram Khadivi, and Jan-Thorsten Peter. 2016. Guided alignment training for topic-aware neural machine translation. *The 25th Conference of The Association for Machine Translation in the Americas*, page 121.

Liying Cheng, Dekun Wu, Lidong Bing, Yan Zhang, Zhanming Jie, Wei Lu, and Luo Si. 2020. Ent-desc: Entity description generation by exploringknowledge graph. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Andrew Chisholm, Will Radford, and Ben Hachey. 2017. Learning to generate one-sentence biographies from wikidata. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 633–642.

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

Sajal Choudhary, Prerna Srivastava, Lyle Ungar, and João Sedoc. 2017. Domain aware neural dialog system. *arXiv preprint arXiv:1708.00897*.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the Second International Conference on Human Language Technology Research*, pages 138–145.

Nouha Dziri, Ehsan Kamalloo, Kory W Mathewson, and Osmar Zaiane. 2018. Augmenting neural response generation with context-aware topical attention. *arXiv preprint arXiv:1811.01063*.

Zihao Fu, Lidong Bing, and Wai Lam. 2020a. Open domain event text generation. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 7748–7755.

Zihao Fu, Bei Shi, Lidong Bing, and Wai Lam. 2020b. Unsupervised kb-to-text generation with auxiliary triple extraction using dual learning. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*.

Zihao Fu, Bei Shi, Wai Lam, Lidong Bing, and Zhiyuan Liu. 2020c. Partially-aligned data-to-text generation with distant supervision. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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

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

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170.

Parag Jain, Anirban Laha, Karthik Sankaranarayanan, Preksha Nema, Mitesh M Khapra, and Shreyas Shetty. 2018. A mixed hierarchical attention based encoder-decoder approach for standard table summarization. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 622–627.

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association of Computational Linguistics*, 5(1):339–351.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, System Demonstrations*, pages 67–72.

Ioannis Konstas and Mirella Lapata. 2013. A global model for concept-to-text generation. *Journal of Artificial Intelligence Research*, 48:305–346.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1203–1213.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Workshop on Text Summarization Branches Out*.

Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. 2015. Topical word embeddings. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2418–2424.

Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1412–1421.

Tomas Mikolov and Geoffrey Zweig. 2012. Context dependent recurrent neural network language model. In *2012 IEEE Spoken Language Technology Workshop (SLT)*, pages 234–239.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The e2e dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206.

Wenjie Ou, Chaotao Chen, and Jiangtao Ren. 2018. T2s: An encoder-decoder model for topic-based natural language generation. In *International Conference on Applications of Natural Language to Information Systems*, pages 143–151.

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

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

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, pages 697–706.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112.

Sander Tars and Mark Fishel. 2018. Multi-domain neural machine translation. *arXiv preprint arXiv:1805.02282*.

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

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 4566–4575.

Pavlos Vougiouklis, Hady Elsahar, Lucie-Aimée Kaffee, Christophe Gravier, Frederique Laforest, Jonathon Hare, and Elena Simperl. 2018. Neural wikipedian: Generating textual summaries from knowledge base triples. *Journal of Web Semantics*, 52-53:1–15.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge & Data Engineering*, 29(12):2724–2743.

Yue Wang, Jing Li, Hou Pong Chan, Irwin King, Michael R Lyu, and Shuming Shi. 2019. Topic-aware neural keyphrase generation for social media language. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2516–2526.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2253–2263.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2018. Learning neural templates for text generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3174–3187.

Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. 2017. Topic aware neural response generation. In *Thirty-First AAAI Conference on Artificial Intelligence*, pages 3351–3357.

Jian Zhang, Liangyou Li, Andy Way, and Qun Liu. 2016. Topic-informed neural machine translation. In *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers (COLING)*, pages 1807–1817.