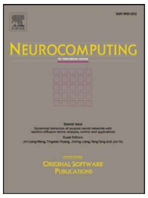
Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom



A topic-driven language model for learning to generate diverse sentences

Ce Gao, Jiangtao Ren*

School of Data and Computer Science, Guangdong Province Key Lab of Computational Science, Sun Yat-sen University, Guangdong 510275, People's Republic of China

ARTICLE INFO

Article history:
Received 26 October 2017
Revised 9 October 2018
Accepted 2 January 2019
Available online 9 January 2019

Communicated by Dr. Y. Chang

Keywords:
Deep learning
Natural language generation
Variational autoencoder

ABSTRACT

Generating diverse sentences under a topic is a meaningful, yet not well-solved task in the field of natural language processing. We present a neural language model for generating diverse sentences conditioned on a given topic distribution. From the perspective of diversity, the proposed model takes the advantages of variational autoencoders with convolutional neural network and long short-term memory architecture. The proposed model is trained end-to-end to learn topic-level Gaussian distributions in the latent space. Then our model decodes the samples of topics obtained from latent space to generate each sentence. Results on Restaurant Dataset and Yahoo! Answers Dataset show that our model outperforms other methods in terms of language model perplexity. Also, our approach can generate a large set of different coherent sentences related to given topics. The diversity of our sentences provides a novel interpretation of topics.

1. Introduction

Nowadays, text-to-text generation methods, which use existing texts as input and automatically produce coherent texts as output, play an increasingly important role in the field of Natural Language Generation (NLG). Current generative applications include text summarization [1], question generation [2], response generation in dialogue systems [3,4] and machine translation [5]. Our work focuses on generating sentences using specific topic information. Topic information can help people understand the potential meaning of sentences better and determine the semantic range to a certain extent. For generative tasks, Lau et al. [6] combine a topic model and a language model to generate sentences under topics. Furthermore, Xing et al. [7] generate the responses of the chatbot based on the topic information using a sequence-to-sequence framework. However, these methods have not taken into account the diversity of generated sentences. The output sentence of the RNN-based decoder of these models is unique if the initial state of the hidden layer is fixed and the word of the maximum probability is taken at each step of the RNN. For example, previous studies can generate only one sentence under a specific topic without using beam search in the decoder [6]. Therefore, our model is proposed to address this problem.

At the same time, generative models, such as generative adversarial networks (GANs) [8] and variational autoencoders (VAEs) [9], have achieved encouraging results in various areas. For example, variational autoencoders have been successfully applied in image generation and sentence generation. Especially for sentence generation, Bowman et al. [10] propose an RNN-based variational autoencoder generative model that can automatically encode the entire source sentences and generate coherent sentences by sampling in the latent space. Variational autoencoders force the initial state of the hidden layer in the RNN-based decoder to become different by sampling to solve the problem of generating diverse sentences, which gives us inspiration. However, simple variational autoencoders can not generate sentences related to a specific topic.

Inspired by the variational autoencoders, our work focuses on how to generate a variety of sentences based on the topic information. By taking the advantages of topic models and the variational autoencoders, we propose a neural variational language model to study the topic-level Gaussian distributions in latent space. Our **T**opic-level **G**aussian distribution **L**anguage **M**odel (henceforth "TGLM") can learn topics and sentence information jointly. Furthermore, we can generate different sentences under the same topic distribution, which is beneficial to real-world applications. For example, in order to enrich the contents of review websites, we can automatically generate comments about the service of a new restaurant that nobody has visited. In the human-machine dialogue system, we can get a topic distribution of an input sentence and return a coherent sentence related to the topics.

* Corresponding author.
E-mail address: issrjt@mail.sysu.edu.cn (J. Ren).

https://doi.org/10.1016/j.neucom.2019.01.002
0925-2312/© 2019 Elsevier B.V. All rights reserved.

Our model takes in word embeddings of a sentence and obtains a sentence vector using a convolutional network. Then the model uses a fully-connected layer to predict the distribution of the topics. We introduce a global topic embedding matrix to represent topics in the TGLM. During training, for each sentence, we only learn the Gaussian distribution of the topic which has the maximum probability in the sentence. More specifically, we put the product of this topic vector and its corresponding probability together with the source sentence into an LSTM to learn the mean and variance. During inference, when we are given a topic distribution, we first sample from each topic latent space sequentially and weight them with the given topic distribution. Then we put the weighted samples into a sequence-to-sequence architecture to generate a novel sentence. The mechanism enables us to generate a large set of varying sentences conditioned on given topics, which improve the interpretability of topics.

We evaluate our approach on a small Restaurant Dataset [11] and a large Yahoo! Answers Dataset [12]. Moreover, we mix the existing sentences with the generated sentences and put them on the website for annotators to score as human evaluation. Our contributions of this work are as follows:

- We propose a neural variational topic model that integrates the advantages of variational autoencoders with the topic information to generate diverse sentences.
- We demonstrate the effectiveness of topic-level Gaussian distribution on the sentence generation task, which can reconstruct sentences better but has not been sufficiently explored before.
- The results of our experiment show that the proposed model outperforms several baseline methods. Also, the results of human evaluation prove that our generative sentences highly resemble the existing sentences.

2. Related work

Traditionally, natural language generation methods rely on hand-crafted rules and templates [13–15]. However, these methods can not scale well to different domains and datasets. In the past few years, the research on encoder-decoder architectures for natural language generation has begun to take off. Specifically, an encoder encodes the source sentence into a vector, and then a decoder learns to generate the target sentence based on the encoding vector. Unlike the traditional sequence-to-sequence structures, the encoder of our model is hierarchical. In the TGLM, we first encode the source sentence into a distribution of topics. Then we encode the weighted topic vector, which has the maximum probability, together with the sentence to learn the Gaussian distribution of the selected topic. For various tasks, the encoders and decoders can use different neural networks. Recurrent neural network performs well when modeling the sequential data such as machine translation [16], response generation [17] and syntactic parsing [18]. In addition, convolutional neural network is appropriate for processing image data, such as video description generation [19] and visual question answering [20]. Moreover, a simple convolutional neural network with one layer of convolution performs remarkably well on sentence classification tasks [21]. Our research has taken the advantages of convolutional neural network and recurrent neural network. In particular, we choose an LSTM [22] instead of a standard RNN in our model.

The definition of our task is also related to the topic model. A starting point can be found in the work of Blei et al. [23], who showed that a word was generated based on a sampled topic in a document. Cao et al. [24] proposed a neural topic model NTM which combined topic model and neural network into a uniform framework. TA-Seq2Seq [7] leveraged the topic information through a joint attention mechanism and a biased generation probability to generate responses. We were inspired by TDLM [6] which jointly trained a topic model and a language model using a neural network. The topic model in TDLM captured the topical information in documents. Parallel to this, the language model in TDLM learned word relations in sentences. In addition, TDLM focused more on the coherence of potential topics. However, in our model we pay more attention to the diversity of the resulting sentences.

Another line of related work is variational autoencoders first introduced by Kingma and Welling [9]. In the VAE, a deterministic representation $\mathbf{z}$ of an input $\mathbf{x}$ is replaced with a posterior distribution $q(\mathbf{z}|\mathbf{x})$ from the encoder. And then the decoder reconstructs the input by sampling $\mathbf{z}$ from the posterior distribution. The prior $p(\mathbf{z})$ is generally chosen to be a standard Gaussian distribution ($\mu = \mathbf{0}, \sigma = \mathbf{1}$) to make sampling easy. The VAE forces the model to decode from every point in the latent space by using a KL divergence between $q(\mathbf{z}|\mathbf{x})$ and $p(\mathbf{z})$. Bowman et al. [10] proposed a variational autoencoder model where both encoder and decoder were LSTM networks for natural language sentences, which could generate coherent and diverse sentences through continuous sampling. The authors also showed applying VAEs to generate sentences was challenging as straightforward implementations failed to generate diverse sentences and fell back to a language model. Two tricks were proposed to address this problem: (1) add a variable weight to the KL loss in cost function, which gradually increased from 0 to 1 during the training to make the model encode more information in $\mathbf{z}$. (2) apply dropout to the inputs of the decoder to weaken the decoder. We also perform experiment with the tricks in our model. Jain et al. [25] combined the advantages of VAEs and LSTM cells to generate a diverse set of questions given a single input image, which was very enlightening. This model generated image-related questions by learning the sentence-level Gaussian distribution in the latent space. However, most of the generated sentences are simple and difficult to cover multiple topics. Our model addresses this problem by learning the topic-level Gaussian distributions, making the resulting sentences complicated based on multiple topics.

3. Modelling approach

In this section, we introduce a neural variational topic model to generate diverse topic-related sentences by learning the Gaussian distributions of the topic vectors. The architecture of the proposed TGLM is shown in Fig. 1.

There are three components in TGLM. In the first part, we use a convolution neural network to obtain a sentence vector and feed the sentence vector into a fully connected layer to get a topic distribution of the sentence. Given the topic distribution, we introduce a global topic embedding matrix to compute each weighted topic representation of the source sentence. The weighted topic vectors of the source sentence will be used in the second part. In the second part, we select the weighted topic vector which has the maximum topic probability and then put it together with the sentence into a standard LSTM model to study the Gaussian distribution for this topic in the latent space. Marrying each sample from Gaussian distributions and the topic distribution, using a sequence-to-sequence framework, the decoder reconstructs the source sentence by sampling in the third part.

3.1. Topic encoder

In the first part of TGLM, a sentence of length n is represented as a vector concatenated with n words. Each of the n words is the e-dimensional word vector $\mathbf{x}_i \in \mathbb{R}^e$ representing the ith word in the sentence:

$$\mathbf{x}_{1:n} = \mathbf{x}_1 \oplus \mathbf{x}_2 \oplus \ldots \oplus \mathbf{x}_n, \tag{1}$$

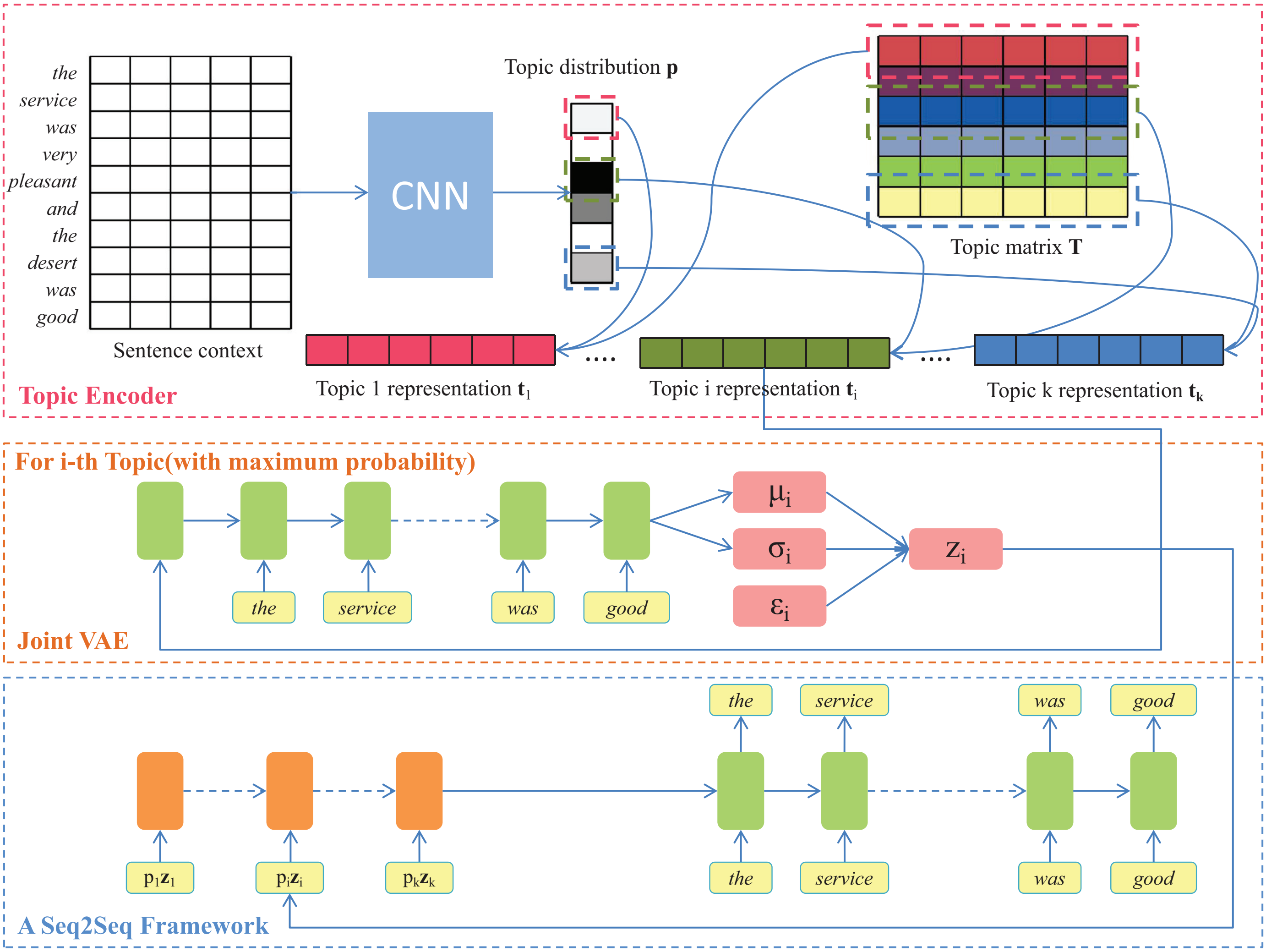


Fig. 1. Structure of TGLM. Our model consists of three main parts. The scope in red dotted line denotes the topic encoder. The scope in yellow dotted line denotes the variational autoencoders. The scope of a sequence-to-sequence framework is in blue dotted line (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

where ⊕ denotes the concatenation operator. Each e-dimensional word embedding vector $\mathbf{x}_i \in \mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_n)$ is selected from the matrix $\mathbf{W}_e \in \mathbb{R}^{e \times v}$ which is pre-trained, where v denotes the total number of words in the vocabulary. Let $\mathbf{w}_v \in \mathbb{R}^{je}$ be a filter which we apply to a window of j words to produce a feature. For example, a feature c_i is generated from a window of words $\mathbf{x}_{i:i+j-1}$ by:

$$c_i = f(\mathbf{w}_v \cdot \mathbf{x}_{i:i+j-1} + b_v) \tag{2}$$

Here b_v is a bias term and f is a non-linear function. This filter is applied to each j-gram in a sentence to obtain a feature map sequence $\mathbf{c} = \left[c_1, c_2, \dots, c_{n-j+1}\right]$. We apply a max-overtime pooling operation [26] over the feature map to capture the most salient feature:

$$s = \max_i c_i \tag{3}$$

In the TGLM, the number of filters is a. Then we get the vector representation $\mathbf{s} \in \mathbb{R}^a$ of the source sentence. Then we take the vector into a fully connected softmax layer whose output is the probability distribution over topics:

$$\mathbf{p} = softmax(\mathbf{W}_s \mathbf{s}) \tag{4}$$

where $\mathbf{W}_s \in \mathbb{R}^{k \times a}$ and k is the number of topics. Given the topic distribution $\mathbf{p} = (p_1, \dots, p_k)$ and a global topic matrix $\mathbf{T} = (\mathbf{T}_1, \dots, \mathbf{T}_k)$, each weighted topic vector $\mathbf{t}_i$ of the source sentence is computed as follows:

$$\mathbf{t}_i = p_i \mathbf{T}_i, \tag{5}$$

where $\mathbf{T} \in \mathbb{R}^{k \times b}$ and b is the dimension of the topic vectors. We consider the weighted topic vectors as the components of the sentence topic representation. The weighted topic vector of the maximum probability reveals the main potential meaning of this sentence. We will use this weighted topic vector in the second part of the TGLM. While the other weighted topic vectors are just modifying the main topic. The modification can diversify the sentences of the same topic.

3.2. Learning using VAE

We encode the given sentence and the weighted topic vector of the maximum probability into a latent representation in the second part. Specifically, we have multiple LSTM encoders to learn k Gaussian distributions. When the topic with the maximum probability of the source sentence is the ith topic, the TGLM will choose the ith LSTM encoder to learn the mean and the variance of the ith Gaussian distribution $q(\mathbf{z}_i|\mathbf{x})$. The LSTM unit first maps the weighted topic vector $\mathbf{t}_i$ linearly into its h dimensional latent space using the matrix $\mathbf{W}_t \in \mathbb{R}^{h \times b}$. Then the LSTM unit projects the word embedding $\mathbf{x}_i \in \mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ linearly into the h dimensional latent space through the matrix $\mathbf{W}_w \in \mathbb{R}^{h \times e}$. We put the sequence into the LSTM layer and then obtain the final hidden state $\mathbf{h}_T \in \mathbb{R}^h$ from the last step. In order to obtain the mean $\boldsymbol{\mu}_i$ and the log variance $\log\left(\sigma_i^2\right)$ of an m-variate Gaussian distribution $(q(\mathbf{z}_i|\mathbf{x}))$, we employ two linear transformations to the last state:

$$\mu_i = \mathbf{W}_\mu \mathbf{h}_T \tag{6}$$

$$\log\left(\sigma_i^2\right) = \mathbf{W}_\sigma \mathbf{h}_T, \tag{7}$$

where $\mathbf{W}_\mu \in \mathbb{R}^{m \times h}$ and $\mathbf{W}_\sigma \in \mathbb{R}^{m \times h}$. During training, we use a KL loss to let $q(\mathbf{z}_i|\mathbf{x})$ approximate a prior $p(\mathbf{z}_i)$ which is generally a standard Gaussian $(p(\mathbf{z}_i) = \mathcal{N}(0, 1))$.

3.3. A sequence-to-sequence framework

In the third part of TGLM, we use a sequence to sequence framework to reconstruct the source sentence. Although in the second part of the TGLM we only study the Gaussian distribution of the topic which has the maximum probability, we use all topic information to decode the sentence in the third part. In the encoding process, we first sample each topic representation from each Gaussian distribution and obtain a sample sequence $\mathbf{z} = (\mathbf{z}_1, \ldots, \mathbf{z}_k)$. Then we compute each weighted sample as follows:

$$\bar{\mathbf{z}}_i = p_i \mathbf{z}_i \tag{8}$$

We input a sentence $\bar{\mathbf{z}} = (\bar{\mathbf{z}}_1, \ldots, \bar{\mathbf{z}}_i, \ldots, \bar{\mathbf{z}}_k)$ to the LSTM layer and obtain the last hidden state as sampled representation of the sentence. The randomly drawn m-variate $\mathbf{z}_i \sim \mathcal{N}(0, 1)$ is shifted and scaled by the mean $\boldsymbol{\mu}_i$ and the variance σ_i^2 during training. It is worth noting that we use “reparameterization trick” [9] here. Specifically, we can sample easily by $\mathbf{z}_i = \mu_i + \sigma_i \cdot \varepsilon_i$, where $\varepsilon_i = \left(\varepsilon_i^1, \ldots, \varepsilon_i^m\right)$ and $\varepsilon_i^j \sim \mathcal{N}(0, 1)$. We also use other methods to merge the weighted samples (sum or concatenation), but experimental results show that using the LSTM encoder works best. In the decoding process, the hidden state from encoder is used to initialize the first time step in decoder. We provide the start symbol < go >, word embeddings and end symbol < eos > as the input to the succeeding LSTM units of the decoder. Analogous to the second part, those inputs are transformed linearly into h dimensional latent space. In each time step t, the LSTM hidden state $\mathbf{h}_t$ is used for word prediction through a dense layer with linear transformation and softmax output. The model is optimised using standard categorical cross-entropy loss.

At test time, given a topic distribution p, we first sample z_i from each Gaussian distribution and compute each weighted sample by the topic distribution. Then, we get a sequence-to-sequence model that is able to generate sentences given each weighted sample vector. We generate different sentences under the same topic distribution by sampling from each topic latent space sequentially. The input of the weighted sample of the main topic determines the central semantic range of the target sentence. The inputs of other weighted samples add additional meanings to the sentence and increase the diversity.

3.4. Training and regularization

We treat the three parts in TGLM as subtasks in a multi-task learning. The objective function breaks into three terms: the topic loss which is defined as the cross-entropy error between the true topic distribution and predicted topic distribution, the KL divergence of the posterior from the prior and the likelihood of the generated sentences under the posterior (expressed as cross entropy). The true topic distribution means the probability of the topic with ground truth being 1 and others being 0.

Table 1
Statistics of datasets.

Dataset	Classes	Train	Test
Restaurant	5	3,041	800
Yahoo! Answers	10	1,400,000	50,000

Straightforward implementations of our models fail to generate diverse sentences. During training our models consistently set $\mathbf{q}(\mathbf{z}_i|\mathbf{x})$ equal to the prior $p(\mathbf{z}_i)$, making the KL divergence term of the objective function to zero. It makes our model fall back to a standard language model. To overcome this problem we use KL cost annealing and dropout words [10] to weaken the decoder. For KL cost annealing, we add a variable weight to the KL loss in cost function to control more information is encoded into $\mathbf{z}_i$. At the beginning of training we set the weight to zero. After that, we gradually increase the weight to 1 in the training progresses, making the weighted loss function equal to the true loss function. For dropout words, we randomly replace some word tokens with the generic unknown word token < unk >. Especially we set the keep rate $r = 0.5$ to force the model to rely on the weighted latent samples to make good predictions.

4. Experiments

In this section, we first compare our proposed method with several baseline methods by applying them to two different datasets. The quality of the generated sentences is measured with the language model perplexity. Then we compare the generated sentences with the exiting sentences through manual judgment. Finally we discuss the diversity of the generated sentences. We will introduce the datasets and settings before showing our results.

4.1. Datasets

To demonstrate the effectiveness of our proposed model, we perform experiments on the datasets of two different sizes: Restaurant dataset and Yahoo! Answers dataset. The statistics are summarized in Table 1. The Restaurant dataset contains 3841 English sentences extracted from restaurant customer reviews. Yahoo! Answers dataset is obtained through the Yahoo! Webscope program, which is annotated 10 largest main categories. Each class contains 140,000 training samples and 5,000 testing samples. Each sentence in Yahoo! Answers dataset has only one topic. But each online review in Restaurant dataset may cover multiple topics about a restaurant.

4.2. Settings

We use NLTK [27] to tokenize the sentences in datasets, lowercase all word tokens, and keep words that appear more than 10 times in our vocabulary. We initialize our word embeddings with publicly available 300-dimensional Glove vectors[1] [28]. The dimensions of the mean and the log variance are set to 20. The dimension of the topic vectors is set to 64. Moreover, we use a recurrent neural network with 1 LSTM layer and 512 hidden units to generate sentences. All the parameters are randomly initialized by sampling from a uniform distribution $[-0.01, 0.01]$. The batch size and base learning rate of Adam are set to 64 and 0.001. After 10 epochs, the learning rate is decreased by a factor of 0.97 at the end of each epoch [29]. The preprocessed dataset statistics are presented in Table 2.

[1] Pre-trained word vectors of Glove can be obtained from https://nlp.stanford.edu/projects/glove/

Table 2
Model hyper-parameters.

Hyper-parameter	Value	Description
s	32	Sequence length for sentences
n_{batch}	64	Minibatch size
n_{layer}	1	Number of LSTM layers
n_{hidden}	512	LSTM hidden size
n_{epoch}	16	Number of training epochs
k	5,10	Number of topics
e	300	Word embedding size
m	20	Mean and log variance vector size
b	64	Topic vector size
j	2	Convolutional filter width
a	20	Number of features for convolutional filter
l	0.001	Initial learning rate
$p_{dropout}$	0.5	Dropout keep probability

Table 3
Language model perplexity performance of all models on Restaurant and Yahoo! Answers datasets.

Model	Restaurant		Yahoo! Answers	
	Standard	Inputless decoder	Standard	Inputless decoder
LSTM	105.39	312.76	76.62	>500
VAE	99.13	123.57	62.26	108.56
LSTM + LDA	96.68	263.24	60.04	>500
TDLM	89.22	218.54	59.63	>500
TGLM	**88.69**	**106.72**	**57.07**	**93.84**

4.3. Evaluation

4.3.1. Automatic evaluation

The language model perplexity score is used for automatic evaluation, which is a measurement of how well a probability model predicts a sentence. We describe the comparison methods as follows:

- LSTM: An LSTM language model which uses the same hyper-parameters of our model.
- VAE: A language model [10] that incorporates a variational autoencoder to generate natural language sentences. It can generate coherent and diverse sentences through purely continuous sampling. Again we set the same hyper-parameters. We use KL cost annealing and dropout words to weaken the decoder.
- LSTM+LDA: An LSTM language model that combines LDA topic information. First, we train an LDA model [23] to learn 5/10 topics for Restaurant and Yahoo! Answers datasets and obtain the LDA topic distribution $\mathbf{q}_{LDA}$ for each sentence. And then the LSTM combines the distribution $\mathbf{q}_{LDA}$ by concatenating it with the output hidden state to predict the next word (i.e. $\mathbf{h}_t^{'} = \mathbf{h}_t \oplus \mathbf{q}_{LDA}$).
- TDLM: A topically driven neural language model [6] that has two components: a language model and a topic model. The topic model in TDLM learns the topic distribution using a convolution neural network. Based on the topic distribution and a topic matrix, the language model in TDLM generates sentences using a standard LSTM. TDLM can also generate sentences given a topic distribution like our model. All hyper-parameters are the same with TGLM.

As shown in Table 3, we present language model perplexity performance on the test datasets. In the standard setting, our model outperforms other methods on the small Restaurant dataset and the large Yahoo! Answers dataset. In addition, to demonstrate the ability of the topic-level Gaussian distributions to encode the full content of sentences, we also provide inputless decoders corresponding to a word dropout keep rate of 0.5. The strong performance of our model suggests that combining topic information and variational autoencoders benefits language modelling.

Table 4
Human evaluation of our generated sentences and the existing sentences. S1 represents the score of the coherence. S2 represents the score of the relevance of the sentence to the given topic. Scoring values are as follows: best = 5, better = 4, ok = 3, bad = 2, worst = 1.

Domain	Existing sentences		TGLM	
	S1	S2	S1	S2
Restaurant	**3.36**	**3.60**	3.18	3.55
Yahoo! Answers	**3.72**	3.76	3.67	**3.97**

4.3.2. Human evaluation

We conduct human evaluation using a crowdsourcing service. We ask the annotators to score 750 real sentences and 750 generated sentences on our website. They are required to score for: (1) the coherence of the sentence; (2) the relevance of the sentence to the given topic. The results of scoring are shown in Table 4. The generated sentences highly resemble the existing sentences. It proves that our proposed model is competitive on the Restaurant and Yahoo! Answers datasets.

4.4. Analysis of diversity

One of the advantages of our model is the ability to generate diverse sentences given a topic distribution. Combining the variational autoencoders allows our model to generate sentences from latent space by sampling. While a non-variational RNN language model does not have a way to perform this. In order to prove that the representations fill up the latent space, we set the sample $z_i = \left(z_i^1, \ldots, z_i^m\right)$ to different values to generate diverse sentences. Due to limitations of space, we sample 9 points for each topic. The range of z_i^j is $[-1, 1]$ and the step size is 0.25.

In Table 5 we can see the sentences which are generated by model TGLM with topic "food" and topic "health". The generated sentences highlight the content of the given topics, providing different aspects of the topics. Meanwhile, our sentences increase the interpretability of topics and prove that our model performs the ability of diversity.

For the quantitative evaluation of the diversity, we design a metric called diversity score to evaluate diversity in the generated sentences. The metric is defined as follows:

$$diversity\,score = \frac{Unique\,sentences\,which\,were\,never\,seen\,in\,training\,set}{Total\,unique\,sentences\,for\,the\,topic} \tag{9}$$

For the z sampling mechanism of $\mathcal{N}(0, 1)$ using 500 points, we obtained 26.24% diversity score for the Restaurant Dataset and 32.89% diversity score for the Yahoo! Answers Dataset.

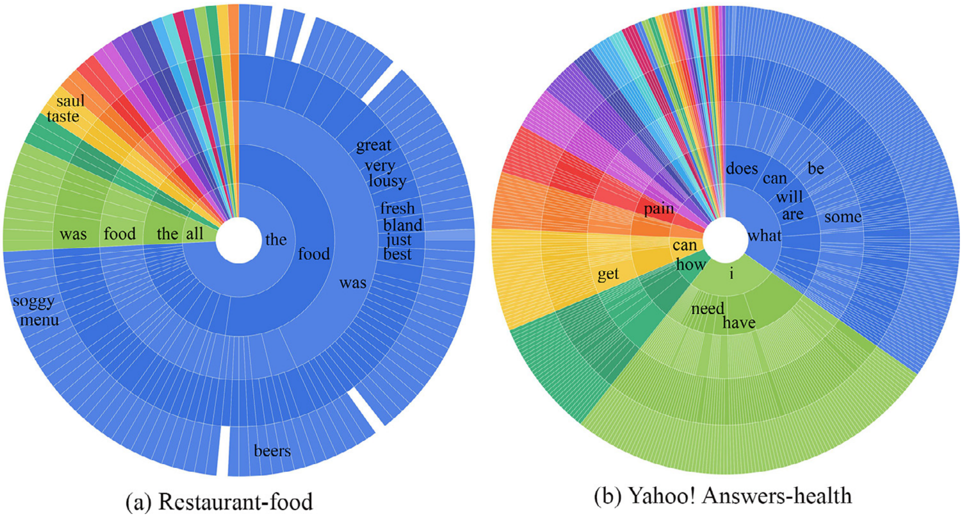
Furthermore, in order to illustrate the diversity of the resulting sentences, we use the sunburst plots shown in Fig. 2 for Restaurant and Yahoo! Answers datasets. Although we only use the first five words to make sunburst plots, we still observe the diversity of our sentences.

4.5. Generated examples

In Table 5 we have shown the successful cases of our model related to a single topic. Also, a range of relative sentences are generated by our model with multiple topics. We present three randomly generated sentences under each topic distribution from model TGLM in Table 6. The unknown token $< \mathrm{unk} >$ is removed from the vocabulary of decoder in the generation process. Moreover, the generation process terminates when a predefined maximum sentence length is reached or an end token $< \mathrm{eos} >$ appears.

Table 5
Results for generating sentences by sampling in latent space. The first block lists the results of food-topic sentences on Restaurant dataset. The second block lists the results of health-topic sentences on Yahoo! Answers dataset.

z_i^j	Generated sentences
−1	The food was good.
−0.75	The rice was poor quality and was cooked hard.
−0.5	The entree was ok and the dessert was not.
−0.25	The pizza was delivered cold and the cheese was not even melted.
0	While the food was excellent, it was not cheap though not extremely expensive either.
0.25	The food was delicious i had a halibut special my husband had steak.
0.5	The food was lousy too sweet or too salty and the portions tiny.
0.75	The food was really good, i had the soup and it was one of the best ever.
1	The food was average or above, including some surprising tasty dishes.
−1	What is a good way to cure my stomach swollen?
−0.75	i really have pain in my back wrist.
−0.5	What is a good natural remedy for bacterial infection?
−0.25	What is a name for a varicose veins after bypass?
0	i also have some redness on my legs.
0.25	How do i treat for my acne painful?
0.5	What is causing breast pain on my girlfriend?
0.75	How do i stop swelling pain in nodules?
1	What should be the best way to get rid of acne pimples on my face?



(a) Restaurant-food

(b) Yahoo! Answers-health

Fig. 2. Sunburst plots for diversity. The *i*th ring represents the frequency distribution over the vocabulary for *i*th word in the generated sentence. We only use the first five words to make the sunburst plots for readability.

Table 6
Examples of sentences generated by TGLM trained on Restaurant dataset and Yahoo! Answers dataset.

Topic	Generated sentences
0.5 service	The food was good and the service classy attentive without being overbearing.
+	While we enjoyed the food we were highly disappointed by the poor service, waiter was not quite competent and slow service and lack of remorse.
0.5 food	From the terrible service and the bland food not to mention the unaccommodating managers the overall experience was horrible.
0.5 food	In addition the food is very good and the prices are reasonable.
+	The prix fixe menu is worth every penny and you get more than enough both the quantity and quality.
0.5 price	Even though its seafood is good, the prices are too high.
0.5 health	How do i get spyware infections?
+	How do i prevent computer viruses?
0.5 computers	Why do i get canker disk?
0.2 society	Are christians more tolerant on the muslims on the immigration?
+	Why do christians oppose more wars from iraq ?
0.8 politics	What are your opinions about all human rights issues worship?

As shown in Table 6, our model trained on Restaurant dataset performs well. For instance, “the food was good and the service classy attentive without being overbearing” refers to two aspects of both food and service, and it corresponds to the given topic distribution. Each sentence in Restaurant dataset may cover multiple topics about a restaurant. It is reasonable to generate sentences that correspond to multiple topics. However, every sentence in Yahoo! Answers dataset has only one topic. The topic loss in the loss function forces the topic distribution $\mathbf{p}$ to approximate to a one-hot vector on Yahoo! Answers dataset. It means that the sentences related to multiple topics do not appear in the training process. But our model still generates coherent sentences as we expected. For example, the words “spyware infection” and “computer viruses” combine two irrelevant topics of “health” and “computers”, which have not appeared in the training set. The results show our framework can generate diverse sentences for a single topic or multiple topics.

We also illustrate a failing sentence in Table 6. The word “canker disk” is a meaningless word which simply combines two high-frequency words of topic “health” and topic “computers”. How to avoid this kind of meaningless words is the problem we have to solve in our future work.

5. Conclusion

In this paper, we address the following problem for sentence generation: how can we generate coherent and diverse sentences conditioned on topics? We propose a novel neural language model which integrates the advantages of variational autoencoders with topic information. The experimental results demonstrate that the proposed method outperforms a variety of baselines. Moreover, we envision our framework being applicable in domains such as review generation and chatbot. There are still potentials for the future researches: (1) avoid the meaningless words in generated sentences; (2) integrate attention mechanism into our framework to generate diverse sentences with a certain word under given topics.

Acknowledgments

This research is partially supported by National Natural Science Foundation of China (No. U1711263).

References

[1] P. Li, Z. Wang, W. Lam, Z. Ren, L. Bing, Salience estimation via variational auto-encoders for multi-document summarization, in: Proceedings of the AAAI, 2017, pp. 3497–3503.

[2] M. Ren, R. Kiros, R. Zemel, Exploring models and data for image question answering, in: Proceedings of the Advances in Neural Information Processing Systems, 2015, pp. 2953–2961.

[3] I.V. Serban, T. Klinger, G. Tesauro, K. Talamadupula, B. Zhou, Y. Bengio, A.C. Courville, Multiresolution recurrent neural networks: An application to dialogue response generation, in: Proceedings of the AAAI, 2017, pp. 3288–3294.

[4] I.V. Serban, A. Sordoni, Y. Bengio, A.C. Courville, J. Pineau, Building end-to-end dialogue systems using generative hierarchical neural network models, in: Proceedings of the AAAI, 2016, pp. 3776–3784.

[5] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, Comput. Sci. (2014).

[6] J.H. Lau, T. Baldwin, T. Cohn, Topically driven neural language model, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 1, 2017, pp. 355–365.

[7] C. Xing, W. Wu, Y. Wu, J. Liu, Y. Huang, M. Zhou, W.-Y. Ma, Topic aware neural response generation., in: Proceedings of the AAAI, 2017, pp. 3351–3357.

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Proceedings of the Advances in Neural Information Processing Systems, 2014, pp. 2672–2680.

[9] D.P. Kingma, M. Welling, Auto-encoding variational bayes, 1050 (2014) 1. arXiv:1704.08012.

[10] S.R. Bowman, L. Vilnis, O. Vinyals, A.M. Dai, R. Józefowicz, S. Bengio, Generating sentences from a continuous space, in: Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11–12, 2016, 2016, pp. 10–21.

[11] M. Pontiki, D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos, S. Manandhar, Semeval-2014 task 4: Aspect based sentiment analysis, Proceedings of the International Workshop on Semantic Evaluation at (2014) 27–35.

[12] X. Zhang, J. Zhao, Y. LeCun, Character-level convolutional networks for text classification, in: Proceedings of the Advances in Neural Information Processing Systems, 2015, pp. 649–657.

[13] R. Dale, S. Geldof, J.-P. Prost, Using natural language generation in automatic route, J. Res. Pract. Inf. Technol. 37 (1) (2005) 89.

[14] E. Reiter, R. Dale, Building Natural Language Generation Systems, Cambridge University Press, 2000.

[15] A. Ratnaparkhi, Trainable methods for surface natural language generation, in: Proceedings of the 1st North American chapter of the Association for Computational Linguistics Conference, Association for Computational Linguistics, 2000, pp. 194–201.

[16] M. Wang, Z. Lu, H. Li, Q. Liu, Memory-enhanced decoder for neural machine translation, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2016, pp. 278–286.

[17] L. Shang, Z. Lu, H. Li, Neural responding machine for short-text conversation, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), 1, 2015, pp. 1577–1586.

[18] O. Vinyals, Ł. Kaiser, T. Koo, S. Petrov, I. Sutskever, G. Hinton, Grammar as a foreign language, in: Proceedings of the Advances in Neural Information Processing Systems, 2015, pp. 2773–2781.

[19] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, T. Darrell, Long-term recurrent convolutional networks for visual recognition and description, in: proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

[20] H. Xu, K. Saenko, Ask, attend and answer: exploring question-guided spatial attention for visual question answering, in: Proceedings of the European Conference on Computer Vision, Springer, 2016, pp. 451–466.

[21] Y. Kim, Convolutional neural networks for sentence classification, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, 2014, pp. 1746–1751.

[22] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.

[23] D.M. Blei, A.Y. Ng, M.I. Jordan, Latent dirichlet allocation, J. Mach. Learn. Res. 3 (Jan) (2003) 993–1022.

[24] Z. Cao, S. Li, Y. Liu, W. Li, H. Ji, A novel neural topic model and its supervised extension, in: Proceedings of the AAAI, 2015, pp. 2210–2216.

[25] U. Jain, Z. Zhang, A. Schwing, Creativity: generating diverse questions using variational autoencoders, in: Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on, 2017, pp. 5415–5424.

[26] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, P. Kuksa, Natural language processing (almost) from scratch, J. Mach. Learn. Res. 12 (Aug) (2011) 2493–2537.

[27] S. Bird, E. Klein, E. Loper, Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit, “O”Reilly Media, Inc.”, 2009.

[28] J. Pennington, R. Socher, C.D. Manning, Glove: global vectors for word representation., in: Proceedings of the EMNLP, 14, 2014, pp. 1532–1543.

[29] A. Karpathy, J. Johnson, L. Fei-Fei, in: Visualizing and understanding recurrent networks, 2015. arXiv: 1506.02078

Ce Gao received the B.Eng. degree from Hunan University, Changsha, China. He is currently pursuing the M.Eng. degree from School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China. His current research interests include machine learning and data mining.

Jiangtao Ren is currently an Associate Professor at School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China. He received the Ph.D. degree from Department of Automation, Tsinghua University, Beijing, China, in 2003. He has published more than 20 papers in international conferences including the ICML, KDD, ICDM, ECML/PKDD, WWW, SDM, ICME. His current research interests include pattern recognition, machine learning and data mining.