ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)

https://doi.org/10.17559/TV-20190929153200
Original scientific paper

An RNN Model for Generating Sentences with a Desired Word at a Desired Position

Tianbao SONG, Jingbo SUN, Yinbing ZHANG, Weiming PENG, Jihua SONG

Abstract: Generating sentences with a desired word is useful in many natural language processing tasks. State-of-the-art recurrent neural network (RNN)-based models mainly generate sentences in a left-to-right manner, which does not allow explicit and direct constraints on the words at arbitrary positions in a sentence. To address this issue, we propose a generative model of sentences named Coupled-RNN. We employ two RNN's to generate sentences backwards and forwards respectively starting from a desired word, and inject position embeddings into the model to solve the problem of position information loss. We explore two coupling mechanisms to optimize the reconstruction loss globally. Experimental results demonstrate that Coupled-RNN can generate high quality sentences that contain a desired word at a desired position.

Keywords: desired word; lexically constrained; RNN; sentence generation

1 INTRODUCTION

Sentence generation is a key technique in many natural language processing (NLP) tasks, such as machine translation [1, 2], dialogue generation [3, 4], text summarization [5], and image caption [6]. State-of-the-art models are mainly based on recurrent neural networks (RNN's) that generate sentences in a left-to-right manner, either word-by-word [7, 8] or by first sampling a latent sentence vector [9]. In supervised settings, sentences are usually generated conditioned on task-specific features. In unsupervised settings, the generated sentences are largely randomized and unconstrained. However, in some scenarios, sentences may need to be generated under some constraints, such as including a certain topic or sentiment or containing a desired word.

Generating a sentence with a specific word, which can be seen as lexically constrained sentence generation, is useful in many NLP tasks. For example, for domain adaptation in machine translation, it is sometimes necessary to force a domain terminology to appear in the final translation results [10, 11]. For interactive machine translation, the final translation results may depend not only on automatic translation results but also on user inputs [10, 11]. In dialogue systems, by including a specific word, responses can deliver the information they need to convey, and utterances in a dialogue can remain consistent and informative [12, 13]. For image caption, by forcing the inclusion of selected tag words in the output, out-of-domain images containing novel scenes or objects can be processed [14]. Besides, in the second-language teaching and learning domain that has motivated our model, it is useful to generate example sentences for a specific word to ease the burden on teachers to compile example sentences and help learners better grasp the word.

In this paper, we focus on the task of generating sentences with a desired word. To accomplish this task, there are some challenges to be addressed.

The first challenge is how to guarantee that the desired word can appear in a generated sentence at arbitrary positions. Most previous models can only impose constraints on the first word, and this restricts the ability of the models as well as the form of the sentences. Recently, lexically constrained decoding methods that extend beam search to allow the inclusion of specified words or phrases have been proposed [10, 11, 14]. These methods impose constraints on each time step during inference, rather than considering constraints during model training. Mou et al. [15] proposed a backward and forward (B/F) language model to achieve lexically constrained sentence generation in a more natural way, and it has two variants: syn-B/F and asyn-B/F. Our model, which resembles asyn-B/F, employs two RNN's to generate sentences backwards and forwards respectively starting from a desired word.

The second challenge results from the unfixed positions of the desired words. RNN is very suitable for processing sequential data, and it can use hidden states to process tokens at the corresponding positions of the input sequences. Position is very important information for sequential data, especially for natural language. But owing to the unfixed positions of the desired words, the RNN's in our model cannot utilize the position information of sentences. To deal with this problem, we encode the position information and feed it to the RNN's together with the input tokens of the sentences.

The third challenge relates to the discrete nature of language. Our model uses two RNN's to generate sentences backwards and forwards respectively starting from a desired word. There are correlations between the backward and forward parts of a sentence, but the non-differentiability of discrete RNN's prevents the model from back-propagating gradients to optimize the reconstruction loss of a training sentence globally. To address this issue, we propose two coupling mechanisms: hidden state coupling and weighted output coupling.

Taken together, we propose a model named coupled-RNN to achieve the goal of generating sentences with a desired word. We evaluated our model from the following aspects: language generation quality, the effect of position embedding, and coupling mechanisms. Experimental results demonstrate that Coupled-RNN can generate high quality sentences containing a desired word and even ensure that the word appears at a desired position.

2 RELATED WORK

2.1 RNN-based Sentence Generation Models

Mikolov et al. [7, 8] proposed the RNN-based Language Model, which predicts each token of a sequence conditioned on its previous one together with an evolving hidden state and can model sequences with arbitrary lengths. Sutskever et al. [16] described a character-level long short-term memory RNN that can generate grammatical English sentences. Zhang & Lapata [17] used RNN's to generate Chinese poetry, where an RNN outputs a context vector conditioned on the vectors representing previously generated lines. Afterwards, another RNN outputs the next character conditioned on the context vector together with the encodings of previous characters in the current line.

In more recent literature, RNN's combined with the sequence-to-sequence (Seq2Seq) learning framework [1] has achieved remarkable success and has been used in a wide-range of NLP tasks, such as machine translation [2], dialogue generation [18], and text summarization [19]. Seq2Seq uses an RNN to encode an input sequence to a vector of fixed dimensionality. It then uses another RNN to decode the vector to a target sequence. Bowman et al. [9] proposed an RNN-based variational autoencoder generative model for generic sentence generation that can generate sentences from arbitrary sampled vectors. It utilized the architecture of a variational autoencoder (VAE) [20, 21], which encodes each input data into a continuous hidden space rather than a single point and enables generic generation by decoding the sampled vector from the prior distribution.

2.2 Constrained Sentence Generation Models

Hu et al. [22] proposed a model that combines VAE and attribute discriminators for generating sentences with desired attributes, such as sentiment and tense. The model augments the hidden vector in standard VAE with additional vectors, each of which controls an attribute of sentence, and trains discriminators to measure whether the generated sentences match the specific attributes as well as to drive the decoder to produce better results. The model is effective in controlling the abstract attributes of sentences, but it cannot guarantee that a specific word will appear in the sentences.

Kiddon et al. [12] propose a neural checklist model based on RNN's to generate globally coherent text by tracking what has been said and what still needs to be said from a provided agenda. It was used to generate cooking recipes where titles and ingredients are provided as goals and agenda items, and it was also used to generate responses for hotel and restaurant information systems where query types and facts to be mentioned are provided as goals and agenda items. The model is more suitable for generating long texts, and it may not apply to all kinds of agenda items and goals.

Anderson et al. [14] proposed the constrained beam search algorithm to generalize captioning models to out-of-domain images containing novel scenes or objects. It can enforce lexical constraints expressed by a finite state machine over output sequences. Hokamp & Liu [10] proposed the grid beam search algorithm to allow the inclusion of pre-specified lexical constraints in machine translation. Each word that must appear in the output is a constraint. At each time step, the model can generate text from the model distribution, start new constraints, or continue constraints. To solve time consuming problem of the above two models, Post & Vilar [11] proposed a fast grid beam search algorithm using dynamic beam allocation. All of the above three algorithms impose constraints on models during beam search, and they do not modify model parameters or training data, which is not a natural approach to lexically constrained sentence generation. Besides, the algorithms work in supervised settings, where the input context sentences can give clues to which word the output sentences may contain, which is not applicable for unconditional or generic sentence generation.

A more direct and explicit way for lexically constrained sentence generation was presented by Mou et al. [15]. Their B/F language model, which generates sentences with a specific word has two variants: syn-B/F and asyn-B/F. Experimental results show that asyn-B/F is more effective. In their subsequent work, asyn-B/F was used in dialogue systems to generate replies containing a given word based on Seq2Seq [13].

Our Coupled-RNN model is similar to asyn-B/F. During model training, it splits a training sentence by a randomly selected word into two subsequences, trains an RNN to reconstruct one subsequence backwards starting from the selected word, and then feeds the reconstructed result to another RNN to reconstruct the other subsequence forwards, also starting from the selected word.

Coupled-RNN differs from asyn-B/F mainly in the following aspects. First, because the training sentences are split by words that are selected randomly, asyn-B/F loses the position information of the sentences. In Coupled-RNN, we use position embedding to solve this problem. Second, there are correlations between the two subsequences split from a training sentence, but in asyn-B/F, the generators for the two subsequences are trained separately. This may affect the quality of generated sentences and cause the two parts of a sentence to be inconsistent. In Coupled-RNN, we explore a hidden state coupling mechanism and a weighted output coupling mechanism to train the two RNN's jointly. Experimental results show that the position embedding and coupling mechanisms can improve generation quality of sentences containing a desired word, and can even ensure the word appears at the desired position.

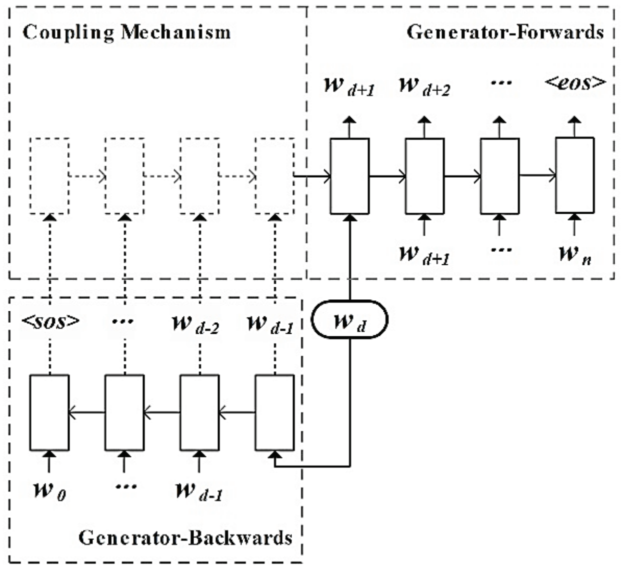

Figure 1 Overall structure of Coupled-RNN. It consists of the Generator-Backwards, Generator-Forwards and Coupling-Mechanism components. Here, w_d in the rounded rectangle represents the desired word and <*sos*> and <*eos*> mark the start and the end of each sentence respectively. Word embeddings and position embeddings are omitted for clarity.

3 COUPLED-RNN MODEL

3.1 Model Overview

Fig. 1 shows the overall structure of Coupled-RNN. It consists of three main components: Generator-Backwards (G_B), Generator-Forwards (G_F), and Coupling Mechanism (CP)

Let $W = \{w_0, ..., w_{d-1}, w_d, w_{d+1}, …, w_n\}$ be a training sentence, where w_n is the *nth wo*rd. We randomly select a word to be the desired word, and use w_d to represent it. w_d splits the sentence into two subsequences:

backward: $W = \{w_0, ..., w_{d-1}, w_d\}$

forward: $W = \{w_d, w_{d+1}, …, w_n\}$

Generator G_B is a gated recurrent unit (GRU)-RNN [23] for generating the backward subsequence, which depicts the following distribution:[1]

$$W_B \sim p_B\left(w_0, ..., w_{d-1} \middle| w_d\right) = \prod_{i=1}^{d} p\left(w_{d-1} \middle| w_{d-i+1}, ..., w_d\right) \quad (1)$$

Generator G_F is another GRU-RNN for generating the forward subsequence conditioned on the output of CP, which depicts the following distribution:

$$\begin{aligned} W_F &\sim p_F\left(w_{d+1}, ..., w_n \middle| w_d, CP(W_B)\right) = \\ &= \prod_{i=d+1}^{n} p\left(w_i \middle| w_{i-1}, ..., w_d, CP(W_B)\right) \end{aligned} \quad (2)$$

where $CP(\cdot)$ represents some sort of processing on the backward subsequence W_B (or the generated backward subsequence), which is done by CP. For example, $CP(\cdot)$ can be a GRU-RNN that inputs the generated backward subsequence W_B and outputs a hidden state, but this may lead to some problems, which is discussed in Subsection 3.3.

The Coupled-RNN is then optimized to minimize the reconstruction error of the training sentences as follows:

$$L\left(\theta_B, \theta_F, \theta_{CP}; W, w_d\right) = -\log p_B\left(W_B\right) - \log p_F\left(W_F \middle| CP\left(W_B\right)\right) \quad (3)$$

where θ_B, θ_F and θ_{CP} denote the parameters of G_B, G_F and CP, respectively.

3.2 Position Embedding

Position is very important information for sequential data, especially for natural language. For a sentence, position information implies its global structure and the dependence between words. In Coupled-RNN, a training sentence is split into two subsequences by a randomly selected word, and then the backward subsequence is fed into G_B and the forward subsequence is fed into G_F Although the RNN itself can capture the position information of the input sentence, the unfixed position of the selected word and the variable length of the subsequences make the two generators unable to utilize the position information effectively. To deal with this problem, we inject position embeddings into the generators together with the input words of the sentence.

Following Vaswani et al. [24], the position embeddings are defined as:

$$\begin{cases} PE\left(pos, 2i\right) = \sin\left(pos / 10000^{2i/emb\_size}\right) \\ PE\left(pos, 2i+1\right) = \cos\left(pos / 10000^{2i/emb\_size}\right) \end{cases} \quad (4)$$

where *pos* represents the position [2], *i* represents the dimension and *emb_size* represents the dimensionality. Position embeddings have the same dimensionality as the word embeddings so that they can be summed.

Position embeddings can take many other forms [25], such as using learned position embeddings or putting position embeddings and word embeddings together to form a joint vector. In Coupled-RNN, we borrow the position embedding method from Vaswani et al. [24], because it can represent both the absolute position information of words and the relative position information between words and can be applied to sentences of variable lengths, which satisfies our requirements.

3.3 Coupling Mechanism

Given a desired word, the most intuitive ways in which G_B and G_F work together to generate a sentence are similar to the approaches of sep-B/F and asyn-B/F in [15]. For the former one, the only connection between G_B and G_F is the desired word. For the latter one, G_B acts on G_F using the generated backward subsequence, but it is impossible to back-propagate gradients from G_F and CP to G_B through the discrete samples, so this is equivalent to training G_B and G_F separately.

Both of the above methods may affect the quality of the generated sentences and cause the generated sentences to be inconsistent and incoherent. We propose two coupling mechanisms to solve this problem.

Hidden State Coupling Mechanism. As shown by the bold arrow in Fig. 2, G_F takes the last hidden state of G_B as input and back-propagate gradients to G_B through the hidden state. Here, $CP(\cdot)$ in Eq. (2) and Eq. (3) is equivalent to the last hidden state of G_B.

This mechanism can be seen as a variant of Seq2Seq, where G_B encodes the backward subsequence into a hidden state and G_F decodes the hidden state to the forward subsequence. However, unlike Seq2Seq, both the encoder and decoder take the desired word as the initial input word, and their outputs constitute the final result together.

[1] During training, the target backward subsequence is $\overline{w}_B = \left\{\langle sos\rangle, w_0, ..., w_{d-1}\right\}$ and the target forward subsequence is $\overline{w}_F = \left\{w_{d+1}, ..., w_n, \langle eos\rangle\right\}$.

[2] For a sentence $W = \left\{\langle sos\rangle, w_0, ..., w_d, ..., w_{n,} \langle eos\rangle\right\}$, the *pos* is $\{0, 1, ..., d+1, ..., n+1, n+2\}$.

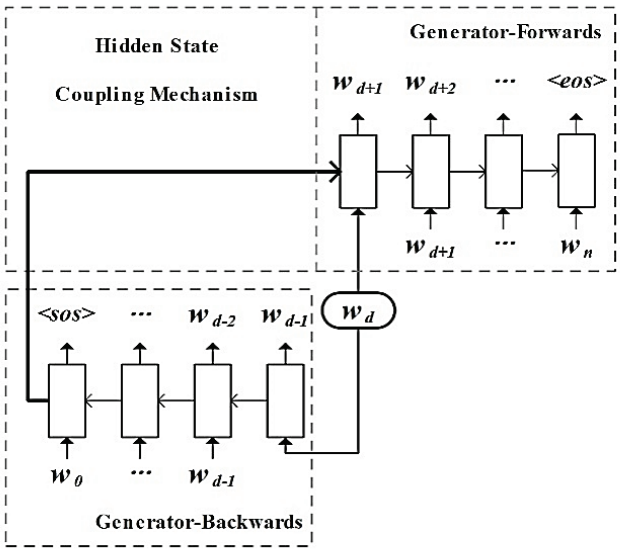

Figure 2 Hidden State Coupling Mechanism. As indicated by the bold arrow, G_F takes the last hidden state of G_B as input.

Weighted Output Coupling Mechanism. The weighted output coupling mechanism is indicated by the bold dashed arrow in Fig. 3. Here, *CP* is a GRU-RNN that takes the weighted output as input and outputs a hidden state to G_F. The calculation of the weighted output is shown in the lower right part of Fig. 3.

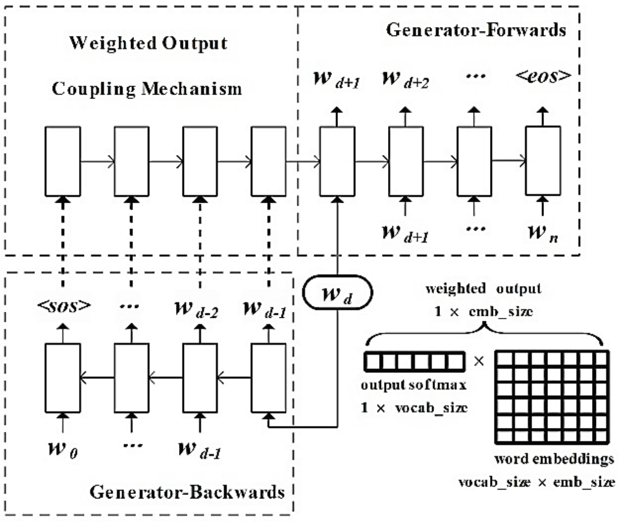

Figure 3 Weighted Output Coupling Mechanism. As indicated by the bold dashed arrow, *CP* takes the weighted output as input and outputs a hidden state to G_F. The calculation of the weighted output is shown in the lower right part of the figure.

Let o_t be the output vector of an RNN unit in G_B at time step *t*. In addition, out_soft_t is the output vector of the softmax function on o_t as follows:

$$out\_soft_t = softmax(o_t / \tau) \tag{5}$$

where $\tau > 0$ is the temperature. The size of out_soft_t is $1 \times vocab\_size$, where *vocab_size* represents the vocabulary size. The generated word w_t in W_{bw} is sampled from the multinomial distribution parameterized by out_soft_t.[3]

If *CP* takes the sampled w_t as input, the discrete samples will prevent the model from back-propagating gradients to optimize the reconstruction loss of a training sentence globally. Instead, we use the weighted output, calculated as follows:

$$out\_weighted_t = outsoft_t \times word\_emb \tag{6}$$

where *word_emb* is a word embedding matrix of size $vocab\_size \times emb\_size$. In this case, $CP(\cdot)$ in Eq. (2) and Eq. (3) is equivalent to the last hidden state of GRU ($out\_weighted_t$).

At the start of training, we set the temperature τ in Eq. (5) to 1. Then, as training progresses, we gradually decrease this temperature to yield peaked distributions, so that $out\_weighted_t$ can approximate to discrete words more precisely.

4 EXPERIMENTS

4.1 Datasets and Setup

We conducted experiments on two datasets. The first one is the Book Corpus [26], which is a collection of 11K books in 16 different genres, e.g., romance, fantasy, and science fiction. The second one is Yelp Review Corpus [27], which contains user ratings and reviews for business activities and is provided by Yelp Inc. for the Yelp Dataset Challenge, Round 13.

For each dataset, we randomly selected 1.5M sentences, split it into train/dev/test sets by the ratios of 80/10/10, and replaced infrequent words (≤ 10) with the token <*unk*>. For the Book Corpus, the resulting vocabulary size is 27,080, and the average sentence length is 14.52/14.49/14.52. For the Yelp review corpus, the resulting vocabulary size is 17,962 and the average sentence length is 16.14/16.13/16.16.

Throughout our experiments, we used the following baselines:

sep-B/F [15]: Starting from the desired word, generate subsequences backward and forward using two RNN's. Then, concatenate the two subsequences to form a complete sentence.

asyn-B/F [15]: Generate the backward subsequence starting from the desired word using an RNN. Then, feed the resulting subsequence to another RNN to generate the backward subsequence. Concatenate the backward and forward subsequences to form a complete sentence.

Both baseline models work in unsupervised settings and can generate sentences containing a desired word. We did not conduct comparative experiments with the models based on beam search [10, 11, 14] and the models for dialogue generation [12, 13], because those are all supervised models and the input context sentence gives a clue to which word the output sentence may contain, which is not applicable in our model.

For our **Coupled-RNN**, we evaluated it under different choices of position embedding and coupling mechanisms:

position embedding (**pos**)
hidden state coupling (**hidden**)
weighted output coupling (**weighted**)
position embedding + hidden state coupling (**pos+hidden**)

[3] The generated words in W_{fw} are sampled in the same way.

position embedding + weighted output coupling (**pos+weighted**)

For all models, we used single-layer GRU-RNN's with a hidden-layer size of 300 and max length of 50. The dimensionality of word embeddings and position embeddings was set to 300. Word embeddings were fixed with GloVe [28]. We optimized the models using Adam [29]. The batch size, threshold for gradient clipping, and learning rate were set to 32, 5 and 0.001, respectively. For the weighted output coupling mechanism, we used the temperature τ in Eq. (5), annealing logistically from 1 to 0 during training. We randomly selected the desired word in each training of each sentence and recorded these words and their positions.

Table 1 Overall structure of Coupled-RNN. It consists of the Generator-Backwards, Generator-Forwards and Coupling-Mechanism components. Here, w_d in the rounded rectangle represents the desired word and <sos> and <eos>mark the start and the end of each sentence respectively. Word embeddings and position embeddings are omitted for clarity.

Models	Book Corpus				Yelp Review Corpus			
	NLL(train)	NLL(dev)	NLL(test)	PPL(test)	NLL(train)	NLL(dev)	NLL(test)	PPL(test)
sep-B/F	61.25	62.26	62.45	14.44	65.57	66.70	66.08	13.63
asyn-B/F	60.92	62.06	62.24	14.40	65.19	66.34	66.41	13.46
hidden	60.21	61.32	61.44	13.90	64.20	65.48	65.57	13.00
weighted	60.19	61.31	61.46	13.92	64.24	65.49	65.57	13.02
pos	59.80	60.68	60.87	13.54	64.24	65.26	65.37	12.94
pos+hidden	59.52	60.55	60.73	13.47	63.75	64.81	64.92	12.61
pos+weighted	59.65	60.66	60.86	13.53	63.87	64.91	64.99	12.70

4.2 Results

Language Modeling. We report the language modeling experimental results for all models in Tab. 1, which are measured in terms of negative log likelihood (NLL) and perplexity (PPL). From Tab.1 we can see that all Coupled-RNN models improved NLL and PPL over sep-B/F and asyn-B/F. All the models with position embedding performed better than their corresponding models without position embedding, demonstrating the consistent effectiveness of position embedding. All the models with hidden state coupling or weighted output coupling performed better than their corresponding models without coupling mechanisms, demonstrating that the two coupling mechanisms can improve the model ability. The model with position embedding and hidden state coupling gained the best result. Compared with the results on the Yelp review corpus, the NLL results on the Book Corpus are better, but the PPL results are worse. Because the average sentence length of the Book Corpus is shorter than Yelp review corpus, PPL was normalized by sentence length, but NLL was not.

Position Accuracy. For the models with position embedding, we evaluated the position accuracies of the generated sentences. We randomly chose a desired word and one of its corresponding positions, which were recorded during training, and used these models to generate sentence under this constraint. We obtained the generated sentences in two ways. One is a greedy way, which means that for every word in the generated sentences we chose the one that has the maximum probability. The other is a sampling-based way, which means that for every word in the generated sentences we sampled one according to the multinomial distribution parameterized by Eq. (5), and τ is set to 0.6. We also evaluated position accuracies of sentences that are generated using randomly selected desired words and positions, where the "desired word-position" combinations may not be encountered during training. For each model and each way of sentence generation, we obtained 10,000 generated sentences. The results are shown in Tab. 3. We can see that all the models had high position accuracy in the sentences generated under the word and position constraints.

Table 2 Position accuracy of sentences generated in greedy and sampling-based way by the models with position embedding. Here, "record" represents that the desired words and positions were selected from the record during training, and "random" represents that the desired words and the positions were selected randomly.

Models	Position accuracy (record) / %		Position accuracy (random) / %	
	Greedy	Sampling	Greedy	Sampling
pos	100	100	96.623	99.442
pos + hidden	100	99.995	99.353	98.975
pos + weighted	100	100	99.870	99.757

Table 3 Percentage of 3/4/5/6-grams that contain the desired word and appear in the test set for different models

Models	3-gram / %	4-gram / %	5-gram / %	6-gram / %
sep-B/F	67.59	31.39	10.53	2.81
asyn-B/F	58.25	32.22	11.11	3.03
hidden	68.89	33.84	11.98	3.36
weighted	69.24	34.07	12.07	3.58
pos	66.78	30.08	9.32	2.53
pos+hidden	66.86	30.59	9.59	2.57
pos+weighted	67.07	30.96	9.61	2.61

Sentence Coherence Measured by n-gram Overlap. To measure whether the two coupling mechanisms can improve the coherence of the generated sentences, we extracted the 3/4/5/6-grams that contain the desired word (but not at the beginning or the end) from 10,000 sentences generated in the sampling-based way by each model. Then, we calculated how many of these n-grams appear in the test set. The results are shown in Tab. 3. All the models with coupling mechanisms improved the percentage of n-gram overlaps compared with their corresponding models without coupling mechanisms. However, the models with position embedding yielded a lower percentage than their corresponding models without position embedding. The reason may be that these models forced the desired word to appear in a designated position, and this may have influenced the coherence of the sentences.

Sentence Length. For each model, we generated 10,000 sentences in the sampling-based way and 10,000 sentences in the greedy way, and then calculated the average sentence length. The results are shown in Tab. 4 and Tab. 5 gives some example sentences generated by different models. The results show that Coupled-RNN models can generate longer sentences, especially the models with the position constraints for desired words. Moreover, they can generate more diverse sentences.

Table 4 Average length of sentences generated by different models

Models	Length
sep-B/F	10.84
asyn-B/F	12.63
hidden	12.81
weighted	12.67
pos	15.93
pos+hidden	15.94
pos+weighted	16.46

Human Evaluation. Although the Coupled-RNN models gained better results than the baseline models in the above experiments and the position embedding and coupling mechanisms have demonstrated their effectiveness, these results are not sufficient to evaluate a model adequately: some form of human evaluation is also important. We randomly selected 1,000 sentences of lengths less than or equal to 12 and 1,000 sentences of lengths greater than 12 from sentences generated in the sampling-based way using sep-B/F, asyn-B/F, and Coupled-RNN with position embedding and hidden state coupling (pos+hidden), each of which was under the "desired word-position" constraints sampled randomly from the training record. Two graduate students with good English education were invited to judge the grammar and plausibility of these sentences. The average results are shown in Tab. 6. For sentences shorter than or equal to 12 words, the results of asyn-B/F and Coupled-RNN of pos+hidden were similar and slightly better than that of sep-B/F. Further, for sentences longer than 12 words, Coupled-RNN pos+hidden gained much better results than the other baseline models.

Table 5 Example sentences generated by different models using the desired word "food".

Models	Sentences
sep-B/F	the ***food*** was delicious. the ***food*** is delicious!
asyn-B/F	the ***food*** is great and the service is great. the ***food*** was good and the service is always excellent.
hidden	the ***food*** was good, but the service was very slow. the ***food*** was great, and the service was very attentive.
weighted	the ***food*** was good, but the service was horrible. the ***food*** was good but nothing special.
pos	price is reasonable for the quality of ***food*** and service. service is good and the ***food*** was amazing but i've never had an issue.
pos+hidden	our waitress was super friendly and i was impressed with the ***food*** ! i've been here 5 times, and the ***food*** is pretty good.
pos+weighted	i was searching for a restaurant with a lot of mexican ***food***. but after the previous reviews, i had to give them 2 stars for the ***food***.

Table 6 Human evaluation results (*C*1: grammatical and plausible, *C*2: grammatical but implausible, *C*3: ungrammatical. Kappa value is 0.801)

Models	Length ≤ 12 / %			Length > 12 / %		
	*C*1	*C*2	*C*3	*C*1	*C*2	*C*3
sep-B/F	68.6	12.8	18.6	30.3	23.0	46.7
asyn-B/F	70.3	15.2	14.5	51.1	22.4	26.5
pos+hidden	70.5	16.1	13.4	55.8	21.7	22.5

Table 7 presents example sentences generated by Coupled-RNN with position embedding and hidden state coupling using the desired word and position.

Word	Position	Sentences
press	3	but the ***press*** is a little more than i've ever imagined. he could ***press*** the buttons on the phone and dialed 911.
	6	then he was trying to ***press*** the button on the door. she sighed and tried to ***press*** her head into the bed.
	9	he took a deep breath and started to ***press*** the buttons. she stood up and pulled me into the ***press***.
friends	3	i have ***friends*** to make me want to be a little better. my old ***friends*** were seated on the edge of the table.
	6	she told them that her ***friends*** had missed the <*unk*>. then i look at his ***friends*** in the middle of the room.
	9	they said they were just talking about their ***friends***. she was standing next to one of her ***friends***.

5 CONCLUSION

In this paper, we propose the Coupled-RNN model for generating sentences with a desired word. Instead of generating sentences in a left-to-right manner, the model generates backwards and forwards starting from the desired word. More importantly, we inject position embedding into the model to solve the position loss problem and propose the hidden state and weighted output coupling mechanisms with the aim of optimizing reconstruction loss globally and generating more coherent and consistent sentences. Coupled-RNN's gained better results than the baseline models in both quantitative evaluation and human evaluation, and it can generate sentences not only with a desired word but also a desired position, which is not possible in the baseline models.

Future work is as follows: first, to explore better metrics to evaluate the semantic coherence of the generated sentences; second, to generate sentences under multiple lexical constraints; third, to impose semantic constraints on the generated sentences. Finally, at present, the generated sentences cannot be directly used in the second-language teaching and learning domain, so it is necessary to import a knowledge base to improve the domain applicability of the model.

Acknowledgements

This work was supported by National Natural Science Foundation of China (Grant No. 61877004) and Natural Science Foundation of the Anhui Higher Education Institutions of China (Grant No. KJ2019A0592). We thank Kimberly Moravec, PhD, from Liwen Bianji, Edanz Editing China (www.liwenbianji.cn/ac), for editing the English text of a draft of this manuscript. We also thank Mo Chen for the constructive advises.

6 REFERENCES

[1] Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in neural information processing* systems, 3104-3112.

[2] Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.

[3] Serban, I. V., Sordoni, A., Lowe, R., Charlin, L., Pineau, J., Courville, A., & Bengio, Y. (2017, February). A hierarchical latent variable encoder-decoder model for generating dialogues. *Thirty-First AAAI Conference on Artificial Intelligence*.

[4] Shao, L., Gouws, S., Britz, D., Goldie, A., Strope, B., & Kurzweil, R. (2017). Generating high-quality and informative conversation responses with sequence-to-sequence models. arXiv preprint arXiv:1701.03185. https://doi.org/10.18653/v1/D17-1235

[5] Miao, Y. & Blunsom, P. (2016). Language as a latent variable: Discrete generative models for sentence compression. arXiv preprint arXiv:1609.07317. https://doi.org/10.18653/v1/D16-1031

[6] Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3156-3164. https://doi.org/10.1109/CVPR.2015.7298935

[7] Mikolov, T., Karafiát, M., Burget, L., Černocký, J., & Khudanpur, S. (2010). Recurrent neural network based language model. *Eleventh annual conference of the international speech communication association*. https://doi.org/10.1109/ICASSP.2011.5947611

[8] Mikolov, T., Kombrink, S., Burget, L., Černocký, J., & Khudanpur, S. (2011, May). Extensions of recurrent neural network language model. *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 5528-5531. https://doi.org/10.1109/ICASSP.2011.5947611

[9] Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R., & Bengio, S. (2015). Generating sentences from a continuous space. arXiv preprint arXiv:1511.06349. https://doi.org/10.18653/v1/K16-1002

[10] Hokamp, C. & Liu, Q. (2017). Lexically constrained decoding for sequence generation using grid beam search. arXiv preprint arXiv:1704.07138. https://doi.org/10.18653/v1/P17-1141

[11] Post, M. & Vilar, D. (2018). Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. arXiv preprint arXiv:1804.06609. https://doi.org/10.18653/v1/N18-1119

[12] Kiddon, C., Zettlemoyer, L., & Choi, Y. (2016). Globally coherent text generation with neural checklist models. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 329-339. https://doi.org/10.18653/v1/D16-1032

[13]Mou, L., Song, Y., Yan, R., Li, G., Zhang, L., & Jin, Z. (2016). Sequence to backward and forward sequences: A content-introducing approach to generative short-text conversation. arXiv preprint arXiv:1607.00970.

[14] Anderson, P., Fernando, B., Johnson, M., & Gould, S. (2016). Guided open vocabulary image captioning with constrained beam search. arXiv preprint arXiv:1612.00576. https://doi.org/10.18653/v1/D17-1098

[15] Mou, L., Yan, R., Li, G., Zhang, L., & Jin, Z. (2015). Backward and forward language modeling for constrained sentence generation. arXiv preprint arXiv:1512.06612.

[16] Sutskever, I., Martens, J., & Hinton, G. E. (2011). Generating text with recurrent neural networks. *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 1017-1024.

[17] Zhang, X. & Lapata, M. (2014). Chinese poetry generation with recurrent neural networks. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 670-680. https://doi.org/10.3115/v1/D14-1074

[18] Wen, T. H., Gasic, M., Mrksic, N., Su, P. H., Vandyke, D., & Young, S. (2015). Semantically conditioned lstm-based natural language generation for spoken dialogue systems. arXiv preprint arXiv:1508.01745. https://doi.org/10.18653/v1/D15-1199

[19] Nallapati, R., Zhou, B., Gulcehre, C., & Xiang, B. (2016). Abstractive text summarization using sequence-to-sequence rnns and beyond. arXiv preprint arXiv:1602.06023. https://doi.org/10.18653/v1/K16-1028

[20] Kingma, D. P. & Welling, M. (2013). Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114.

[21] Rezende, D. J., Mohamed, S., & Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. arXiv preprint arXiv:1401.4082.

[22] Hu, Z., Yang, Z., Liang, X., Salakhutdinov, R., & Xing, E. P. (2017, August). Toward controlled generation of text. *Proceedings of the 34th International Conference on Machine Learning, (70)*, 1587-1596.

[23] Cho, K., Van Merriënboer, B., Bahdanau, D., & Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259. https://doi.org/10.3115/v1/W14-4012

[24] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 5998-6008.

[25] Gehring, J., Auli, M., Grangier, D., Yarats, D., & Dauphin, Y. N. (2017, August). Convolutional sequence to sequence learning. *Proceedings of the 34th International Conference on Machine Learning, (70)*, 1243-1252.

[26] Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., & Fidler, S. (2015). Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *Proceedings of the IEEE international conference on computer vision*, 19-27. https://doi.org/10.1109/ICCV.2015.11

[27] https://www.yelp.com/dataset

[28] Pennington, J., Socher, R., & Manning, C. (2014, October). Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532-1543. https://doi.org/10.3115/v1/D14-1162

[29] Kingma, D. P. & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Contact information:

Tianbao SONG, Master
Beijing Normal University,
No. 19, Xinjiekouwai St, Haidian District, Beijing, 100875, P. R. China
songtianbao@mail.bnu.edu.cn

Jingbo SUN, Bachelor
Beijing Normal University,
No. 19, Xinjiekouwai St, Haidian District, Beijing, 100875, P. R. China
sunjingbo@mail.bnu.edu.cn

Yinbing ZHANG, Master
Beijing Normal University,
No. 19, Xinjiekouwai St, Haidian District, Beijing, 100875, P. R. China
zhangyinbing@mail.bnu.edu.cn

Weiming PENG, PhD
(Corresponding author)
Beijing Normal University,
No. 19, Xinjiekouwai St, Haidian District, Beijing, 100875, P. R. China
pengweiming@mail.bnu.edu.cn

Jihua SONG, PhD, Professor
Beijing Normal University,
No. 19, Xinjiekouwai St, Haidian District, Beijing, 100875, P. R. China
songjh@mail.bnu.edu.cn