Attention Regularized Sequence-to-Sequence Learning for E2E NLG Challenge

Biao Zhang, Jing Yang, Qian Lin and **Jinsong Su**
Xiamen University, Xiamen, China 361005
{zb, jingy, qianl}@stu.xmu.edu.cn, jssu@xmu.edu.cn

Abstract

This paper describes our system used for the end-to-end (E2E) natural language generation (NLG) challenge. The challenge collects a novel dataset for spoken dialogue system in the restaurant domain, which shows more lexical richness and syntactic variation and requires content selection (Novikova et al., 2017). To solve this challenge, we employ the CAEncoder-enhanced sequence-to-sequence learning model (Zhang et al., 2017) and propose an attention regularizer to spread attention weights across input words as well as control the overfitting problem. Without any specific designation, our system yields very promising performance. Particularly, our system achieves a ROUGE-L score of 0.7083, the best result among all submitted primary systems.

1 Task Definition

This task aims at generating an adequate natural language (NL) description for a dialogue act-based meaning representation (MR). An instance is illustrated below[1]:

```
MR:
name[The Eagle],
eatType[coffee shop],
food[French],
priceRange[moderate],
customerRating[3/5],
area[riverside],
kidsFriendly[yes],
near[Burger King]
NL:
``The three star coffee shop, The
Eagle, gives families a mid-priced
dining experience featuring a
variety of wines and cheeses. Find
The Eagle near Burger King.''
```

where the input *MR* consists of several attributes (slots), such as *name, food* or *near*, and their values, and the output *NL* summarizes the main information from the MR into a faithful and fluent natural language.

We formulate this task as a sequence-to-sequence problem. Concretely, we treat the output *NL* as a language sequence, and we flatten the attribute-value structural MR input directly into a key-value sequence as follows:

```
name[The Eagle], eatType[coffee shop], food[French], priceRange[moderate], customerRating[3/5], area[riverside], kidsFriendly[yes], near[Burger King]
```

By doing so, we are able to apply adequate sequence-to-sequence models to transform the MR into its corresponding NL.

2 The Approach

In this paper, we employ the CAEncoder-enhanced sequence-to-sequence learning model[2] (Zhang et al., 2017) as the transformer between MR and NL. Unlike conventional encoder, CAEncoder leverages a two-level hierarchy to jointly summarize the history and future information so as to better model source semantics into the source word representations. Formally, given a source sequence $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$, CAEncoder learns the representation for word x_i recurrently as follows:

$$\overrightarrow{\mathbf{h}}_i^c = \text{GRU}\left(\overrightarrow{\mathbf{h}}_{i-1}^c, \mathbf{E}_{x_i}\right) \quad (1)$$

$$\overleftarrow{\mathbf{h}}_i^a = \text{CAEncoder}\left(\overleftarrow{\mathbf{h}}_{i+1}^a, \mathbf{E}_{x_i}, \overrightarrow{\mathbf{h}}_i^c\right) \quad (2)$$

[1] Example comes from the official website: http://www.macs.hw.ac.uk/InteractionLab/E2E/.

[2] Source code is available at https://github.com/DeepLearnXMU/CAEncoder-NMT

where GRU$(\cdot)$ represents the Gated Recurrent Unit model (Chung et al., 2014), $\mathbf{E}_{x_i} \in \mathbb{R}^{d_w}$ denotes the embedding for word x_i, and $\mathbf{h}_i^* \in \mathbb{R}^{d_h}$ indicates the corresponding hidden representation. We regard the resulted $\overleftarrow{\mathbf{h}}_i^a$ as the final representation for word x_i. In addition, the two-level hierarchy in CAEncoder$(\cdot)$ operates as follows,

$$\tilde{\mathbf{h}}_i = \text{GRU}_{lower}\left(\overleftarrow{\mathbf{h}}_{i+1}^a, \mathbf{E}_{x_i}\right) \quad (3)$$

$$\overleftarrow{\mathbf{h}}_i^a = \text{GRU}_{higher}\left(\tilde{\mathbf{h}}_i, \overrightarrow{\mathbf{h}}_i^c\right) \quad (4)$$

Intuitively, $\overrightarrow{\mathbf{h}}_i^c$ encodes the semantics of history source words, while $\overleftarrow{\mathbf{h}}_{i+1}^a$ captures the future information. The GRU function acts as a bridge to fuse these two kinds of information flow.

Upon the learned source word representations, we stack an attentional recurrent decoder for target sequence decoding. Basically, this is a conditional language model:

$$p(\mathbf{y}|\mathbf{x}) = \prod_{j=1}^{m} p(y_j|\mathbf{x}, \mathbf{y}_{<j}) \quad (5)$$

where $\mathbf{y} = \{y_1, y_2, \dots, y_m\}$ is the ground-truth target sequence, and $\mathbf{y}_{<j}$ denotes the partial sequence from beginning to position j. The probability of target word y_j is computed as follows:

$$p(y_j|\mathbf{x}, \mathbf{y}_{<j}) = \text{softmax}\left(g(\mathbf{E}_{y_{j-1}}, \mathbf{s}_j, \mathbf{c}_j)\right) \quad (6)$$

where $g(\cdot)$ is a feed-forward layer, $\mathbf{E}_{y_{j-1}} \in \mathbb{R}^{d_w}$ is the embedding for word y_{j-1}, $\mathbf{s}_j \in \mathbb{R}^{d_h}$ denotes the decoder hidden state, and $\mathbf{c}_j \in \mathbb{R}^{d_h}$ is the attention vector that summarizes the source-side relevant information.

The attention vector is obtained by the vanilla attention mechanism proposed by Bahdanau et al. (2015):

$$\mathbf{c}_j = \sum_i \alpha_{ji} \overleftarrow{\mathbf{h}}_i^a \quad (7)$$

$$\alpha_{ji} = softmax\left(a(\mathbf{s}_{j-1}, \overleftarrow{\mathbf{h}}_i^a)\right) \quad (8)$$

where attention weights α_{ji} reflects the matching degree between the source word x_i and target word y_j. For unfamiliar readers, we refer to the previous work (Bahdanau et al., 2015; Zhang et al., 2017).

3 Attention Regularizer

The above framework is actually proposed for machine translation task. However, the E2E NLG challenge task differs from it in that the MR input has no linguistic order, and each input attribute and value has its corresponding aligned target output.[3] And in practice, attention weights tend to focus on a few number of input words, resulting in incomplete target output. To alleviate this phenomenon as well as avoid overfitting, we introduce the following attention regularizer (AttReg):

$$\mathcal{L}_{att} = -\sum_{i=1}^{n} \log\left(\sum_{j=1}^{m} \alpha_{ji}\right) \quad (9)$$

With this regularizer, we expect the attention weight α_{ji} can be spread across each input word. In this way, the target output could include all attributes and values without omitting any potential input information.

During optimization, we directly sum both $\mathcal{L}_{att}$ and $\mathcal{L}_{seq2seq}$ as our final objective. We didn't apply any weighting factor to $\mathcal{L}_{att}$ for simplicity, and leave this operation in the future. Notice that $\mathcal{L}_{seq2seq}$ is the vanilla negative log likelihood loss.

4 Experiments

Datasets We used the official training, development and test dataset. We first converted the MR inputs into sequences, and then tokenized both MR and NL using the script "*tokenizer.perl*" in Moses[4]. The resulted source and target word sequences are further split into sub-words via *BPE algorithm*[5] (Sennrich et al., 2016).[6] We limited the sub-word number to be 1000, and shared this vocabulary for both source and target sequence. We didn't apply any delexicalization. We employed the BLEU score produced by the official script[7] as the metric to select the best model parameters according to the performance on development dataset. We performed paired bootstrap sampling (Koehn, 2004) for significance test.

Model Setting We set the dimensionality of word embedding to 620, and that of hidden states to 1000. Sequences longer than 100 were pruned. We initialized all model parameters with an uniform distribution that ranges from -0.08 to 0.08,

[3] For MT, some source words can have no aligned target word.

[4] https://github.com/moses-smt/mosesdecoder

[5] https://github.com/rsennrich/subword-nmt

[6] Notice that we didn't compare the use of sub-word against word. Although the vocabulary in this task is quite limited, we expect the sub-words can deal better with possible unlimited named entities, such as restaurant names.

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

Dataset	Model	BLEU	NIST	ROUGE-L	METEOR	CIDEr
Dev	Baseline	0.6925	8.4781	0.7257	0.4703	2.3987
	Our	0.7157	8.6367	0.7350	0.4660	2.3414
	Our+AttReg	0.7409	8.7964	0.7610	0.4837	2.5506
Test	Baseline	0.6593	8.6094	0.6850	0.4483	2.2338
	Our+AttReg (ensemble)	0.6545	8.1804	0.7083	0.4392	2.1012

Table 1: Performance of baseline and our models. The *Baseline* denotes official result.

and tuned these parameters using Adam optimizer under its default hyperparameters ($\beta_1 = 0.9, \beta_2 = 0.999$). The gradient was clipped when its norm exceeds 5. We used an initial learning rate of 0.0005, and halved it after each epoch. The batch size during training was 80, and we applied dropout on the final prediction layer with a rate of 0.2. For decoding, we used the general beam search algorithm with a beam size of 10. The maximum number of training epoch was set to 5. Notice that we set these hyperparameters mainly by empirical experiences from machine translation tasks (Zhang et al., 2017), rather than intuition or grid search.

Table 1 summarizes the performance of different models. Our final submitted result was an ensemble of two models, which were trained with two different random initializations. On the development, our model equipped with the AttReg significantly outperforms the Baseline. However, our model performs relatively worse than the Baseline on the test set. We contribute this to the mismatch between the development and test set, where, according to the official statement, the test set "is very similar to the development set (e.g. no unknown attributes, values or restaurant names) but the combination of attributes is unique (previously unseen)."

5 Conclusion and Future Work

This paper presents our system submitted to the E2E NLG Challenge. We propose an attention regularizer to control the overfitting problem, and obtain significant improvements over baseline on the development set. However, our final model fails to outperform the Baseline in terms of various evaluation metrics. We believe this is because the style difference between the development and test inputs, and future efforts will be spent on the adaption between these two styles and the generalization of our model.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proc. of ICLR*.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR* .

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proc. of EMNLP*.

Jekaterina Novikova, Ondrej Dusek, and Verena Rieser. 2017. The E2E dataset: New challenges for end-to-end generation. *CoRR* abs/1706.09254. http://arxiv.org/abs/1706.09254.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proc. of ACL*. pages 1715–1725.

B. Zhang, D. Xiong, J. Su, and H. Duan. 2017. A context-aware recurrent encoder for neural machine translation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 25(12):2424–2432. https://doi.org/10.1109/TASLP.2017.2751420.