Technical Report for E2E NLG Challenge

Heng Gong
Harbin Institute of Technology
gong@hit.edu.cn

Abstract

This paper describes the primary system submitted by the author to the E2E NLG Challenge on the E2E Dataset (Novikova et al. (2017)).Based on the baseline system called TGen (Dušek and Jurcicek (2016)), the primary system uses REINFORCE to utilize multiple reference for single Meaning Representation during training, while the baseline model treated them as individual training instances.

1 Introduction

Natural Language Generation is an essential part in many applications such as conversational agent, automatic sport report generation, etc. E2E NLG Challenge focuses on generating natural language based on some attributes in restaurant domain with crowd-sourced 50k instances. (Novikova et al. (2016)). Each instance consists of attributes and multiple reference in language for them. Given a set of meaning representations $\{type_1[value_1], type_2[value_2], ..., type_T[value_T]\}$, the system should output a sequence of words $\{word_1, word_2, ..., word_S\}$. For example, given "name[The Punter], food[Indian], priceRange[cheap]", a description of this restaurant like "The Punter provides Indian food in the cheap price range." should be generated.

TGen (Dušek and Jurcicek (2016)), a Seq2Seq model with attention, beam search and reranker during testing has achieved impressive performance on this dataset. It treats multiple reference to a set of attributes as different instances for training, while evaluates generated description against multiple references using automatic metrics for testing. We argue that forcing the model to learn to generate descriptions similar to multiple references for the same input may weaken the model's ability to generate diversified description, and thus generate more general expressions. With that in mind, we use REINFORCE to fine-tune model trained by TGen (Dušek and Jurcicek (2016)) with reward as BLEU between the generated sentence and multiple reference.

In the following sections, we first describe the base model and the REINFORCE to fine-tune on them. Then we will describe some details of the model implementation, automatic metrics result on dev set and test set and human evaluation result provided by E2E NLG Challenge Comittee.

2 Method

2.1 Base Model

We use TGen (Dušek and Jurcicek (2016)) and the code that has been release on GitHub [1]. It is a Seq2Seq model with attention (Bahdanau et al. (2015)) on decoding side and beam search and a re-ranker for testing.

Input: The attributes are represented as a set of "Act type, slot, value" such as "inform, eattype, restaurant". For Encoder, Multiple attribute representations are concatenated as a sequence such as $\{inform, eattype, restaurant, inform, area, citycentre, ...\}$.

Encoder: For Encoder, base model uses a Recurrent Neural Network with Long Short-term Memory Hochreiter and Schmidhuber (1997) (LSTM) as its cell Gers et al. (2002) to model the input attributes $\{a_1, a_2, ..., a_t\}$ into corresponding fix-sized hidden state $\{h_1, h_2, ..., h_t\}$. It uses $h_t = RNN(h_{t-1}, a_t)$ to update the hidden state for the current input. The hidden state for the last input is used to initialize the hidden state for decoder, and the set of all hidden states in encoder are used for attention mechanism.

Decoder: For Decoder, base model uses another RNN with LSTM as its cell with different

[1] https://github.com/UFAL-DSG/tgen

parameters to generate description given the attributes. It uses $s_t = RNN(s_{t-1}, W_d(c_t \circ y_{t-1}))$ to update its hidden state. y_{t-1} is word that is generated in previous time-step, c_t is a context vector obtained by attention mechanism, which is computed by $c_t = \sum \alpha_{ti} h_i$. The weight for each encoding hidden state are computed using single hidden layer feed-forward network with tanh as its activation function.

Finally, The probability for each token in vocabulary are calculated as follows:

$$P(y_t \mid y_1, ..., y_{t-1}, x) = softmax(W_s(s_t \circ c_t))$$

Re-ranker:Base model also uses a re-ranker to rerank top k outputs from beam search in favor of outputs that match information from attributes better. It uses another RNN, structural identical to the encoder, to model the generated description as follows:

$$g_t = RNN(g_{t-1}, z_t)$$

Then, it uses multiple binary classifiers on the last hidden state for all possible slot, value combinations seen in training data. Each classifier uses sigmoid function. Then uses classifiers to generate 1-hot style vector for generated description. Also, a one-hot style vector is generated for meaning representation. Then the hamming distance between those two vectors is used as the penalty, and weighted penalties are subtracted from the log probability of the corresponding generated description.

2.2 REINFORCE

We use REINFORCE to utilize multiple reference for one instance during training. First, we train TGen (Dušek and Jurcicek (2016)) model on the training dataset to obtain a good model for generating description. Then we use REINFORCE to finetune on the obtained model. For each instance in the dataset, we use the current model to generate a description, and define the reward for this description as follows:

$$R(d_i) = BLEU(d_i, multi - references_i)$$

Then, we update each parameter of the model in terms of the generated description as follows:

$$\theta \leftarrow \theta + \alpha \nabla_\theta \sum log(p_\theta(v_i \mid v_{1:(i-1)}))R(v_{1:T})$$

α represents the learning rate for fine-tuing and θ represents each parameter in the model, v_i represents the i^{th} word in the generated description, and the log probability are produced by the decoder. The length of this particular description is T.

Model	BLEU	NIST	METEOR
Base Model	0.6925	8.4781	0.4703
Our Model	0.6643	8.3886	0.4528

Table 1: Dev Result for BLEU, NIST, METEOR

Model	ROUGE_L	CIDEr
Base Model	0.7257	2.3987
Our Model	0.6917	2.3517

Table 2: Dev Result for ROUGE_L, CIDEr

3 Result

3.1 Model Implementation

We follows the model configuration provided on GitHub for base model. [2] For fine-tuning using REINFORCE, the learning rate is reduced to 0.00005, while learning rate is 0.0005 for training base model. For fine-tuing stage of REINFORCE, the batch size is 1. For REINFORCE, we group multiple references according to their corresponding meaning representation for calculating the reward.

3.2 Automatic Evaluation

We evaluate the generated description on dev set using evaluating scripts on GitHub [3].It provides 5 automatic metrics results including BLEU (Papineni et al. (2002)), NIST(Doddington (2002)), METEOR (Lavie and Agarwal (2007)), ROUGE_L (Lin (2004)), and CIDEr(Vedantam et al. (2015)).Results for base model on dev set are extracted from the E2E Challenge Website [4]. Results for both model on test set are also gathered from the E2E Challenge Website. Table 1 shows both model's performance on dev set with respect to BLEU, NIST, and METEOR. Table 2 shows the results on dev set with respect to ROUGE_L and CIDEr. Table 3 shows the result on test set with respect to BLEU, NIST, and METEOR, while Table 4 shows the result on test set with respect to ROUGE_L and CIDEr.

3.3 Human Evaluation

Human evaluation results are reported on E2E NLG Challenge website(Dušek et al. (2018)). According to the website, they use CrowdFlower to

[2]https://github.com/UFAL-DSG/tgen/tree/master/e2e-challenge

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

[4]http://www.macs.hw.ac.uk/InteractionLab/E2E/

Model	BLEU	NIST	METEOR
Base Model	0.6593	8.6094	0.4483
Our Model	0.6422	8.3453	0.4469

Table 3: Test Result for BLEU, NIST, METEOR

Model	ROUGE_L	CIDEr
Base Model	0.6850	2.2338
Our Model	0.6645	2.2721

Table 4: Test Result for ROUGE_L, CIDEr

Model	TrueSkill	Range
Base Model	0.209	(2-5)
Our Model	0.228	(2-4)

Table 5: Quality Result

Model	TrueSkill	Range
Base Model	0.103	(5-7)
Our Model	0.054	(7-11)

Table 6: Naturalness Result

conduct the evaluation. Evaluators are presented with five randomly selected outputs from different systems for the same meaning representation, and asked to rank them from the best to worst, ties accepted.They conducted two types of human evaluation for quality and naturalness respectively. As for quality, they evaluate overall quality of the utterance with respect to grammars, fluency, adequacy, etc. As for naturalness, they measure how likely the result is produced by native speaker. The final result are then produced by TrueSkill algorithms (Sakaguchi et al. (2014)) and clustered. Table 5 shows the human evaluation result with respect to quality, and Table 6 shows the results in terms of naturalness. In terms of both quality and naturalness, both models are clustered into the same group.

3.4 Conclusion

In this report, we briefly introduce the task and the form of dataset. Then we introduce the base model, and explain how to fine-tune on the base model. In the end, we report results on both dev and test set, mostly provided by the E2E NLG Challenge.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate .

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*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, HLT '02, pages 138–145. http://dl.acm.org/citation.cfm?id=1289189.1289273.

Ondřej Dušek and Filip Jurcicek. 2016. Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, pages 45–51. https://doi.org/10.18653/v1/P16-2008.

Ondrej Dušek, Jekaterina Novikova, and Verena Rieser. 2018. Findings of the E2E NLG challenge. In *(in prep.)*.

Felix A. Gers, Nicol N. Schraudolph, and Jürgen Schmidhuber. 2002. Learning precise timing with LSTM recurrent networks. *Journal of Machine Learning Research* 3:115–143. http://www.jmlr.org/papers/v3/gers02a.html.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Alon Lavie and Abhaya Agarwal. 2007. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*. Association for Computational Linguistics, Stroudsburg, PA, USA, StatMT '07, pages 228–231. http://dl.acm.org/citation.cfm?id=1626355.1626389.

Chin-Yew Lin. 2004. Rouge: a package for automatic evaluation of summaries. https://www.microsoft.com/en-us/research/publication/rouge-a-package-for-automatic-evaluation-of-su

Jekaterina Novikova, Ondrej Dušek, and Verena Rieser. 2017. The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. Saarbrücken, Germany. ArXiv:1706.09254. https://arxiv.org/abs/1706.09254.

Jekaterina Novikova, Oliver Lemon, and Verena Rieser. 2016. Crowd-sourcing nlg data: Pictures elicit better data. https://doi.org/10.18653/v1/W16-6644.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '02, pages 311–318. https://doi.org/10.3115/1073083.1073135.

Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme. 2014. Efficient elicitation of annotations for human evaluation of machine translation. https://doi.org/10.3115/v1/W14-3301.

R. Vedantam, C. L. Zitnick, and D. Parikh. 2015. Cider: Consensus-based image description evaluation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. pages 4566–4575. https://doi.org/10.1109/CVPR.2015.7299087.