Sheffield at E2E: structured prediction approaches to end-to-end language generation.

Mingjie Chen **Gerasimos Lampouras** **Andreas Vlachos**
Department of Computer Science, University of Sheffield, UK
{mchen33, g.lampouras, a.vlachos}@sheffield.ac.uk

Abstract

We describe the two systems, and their variations, that were submitted by the University of Sheffield to the E2E NLG challenge. Our systems consist of different approaches to structured prediction for end-to-end language generation. Our first submitted system employs imitation learning for structured prediction to explore the large search space without explicitly enumerating it. Our second submitted system uses encoder-decoder architectures to generate sequences of words. Our submitted runs for each system achieved BLEU scores of 0.60 and 0.54 respectively. On human evaluation our imitation learning model were placed in the 2nd best quality and 3rd best naturalness clusters according to Trueskill scores, while our encoder-decoder model was the best performing system on naturalness but on quality it was placed in the 5th best cluster.

1 Introduction

Concept-to-text natural language generation (NLG) is the task of expressing the components (attributes and values) of a meaning representation (MR) as a fluent natural language (NL) text.

Recently, many ML-based approaches have emerged that aim to generate natural language from input MRs while incorporating an end-to-end design, i.e. without having intermediary stages disjoint from each other, as was traditional in NLG models (e.g. separate content selection, or aggregation modules). An advantage of these approaches is that by bypassing intermediary stages, they also bypass the need for expensive stage-specific training resources, e.g. manually annotated alignments between an MR's components and NL sentences' words. This also helps bigger and more complex datasets to be constucted.

Wen et al. (2015) recently introduced an end-to-end approach to NLG, using a Long Short-term Memory (LSTM) network to learn from unaligned data and jointly address sentence planning and surface realization. To keep track of the MR while generating words, they augmented each cell of the LSTM with a gate that conditions it on the MR. Dušek and Jurcicek (2016) proposed a sequence-to-sequence architecture with attention, and applied it to generating linearized deep syntax trees which are in turn converted to sentences. They applied the same model to also directly generate sentences word by word. The final sentence is produced by decoding with beam search, and subsequently reranking the outputs to ensure maximum coverage over the MR's components. Mei et al. (2016) proposed an encoder-aligner-decoder model to perform content selection and surface realization. Their work employed bidirectional LSTM-RNN models and a coarse-to-fine aligner.

Our submissions to the E2E NLG challenge (Novikova et al., 2017) consist of two systems and their variations. The first system applies imitation learning to structured prediction NLG models, and is based on previous work by Lampouras and Vlachos (2016). The submitted run for this system achieved a BLEU score of 0.60 and was placed in the 2nd best quality and 3rd best naturalness clusters according to Trueskill scores based on a series of human evaluations (Dušek et al., 2018). Our second system employs an encoder-decoder architecture with attention and attempts to directly generate an output sequence of words. It achieved a BLEU score of 0.54 and was placed in the 5th best quality and top best naturalness Trueskill clusters, i.e. it was the highest system on naturalness.

2 Dataset pre-preprocessing

The NLG input in the E2E dataset is a meaning representation (MR), which consists of an unordered set of attributes and corresponding values; the out-

```
name = "Midsummer House"
food = Italian
priceRange = high
customer rating = average
near = All Bar One
```

Reference:
There is an Italian place, Midsummer House, situated near All Bar One with average customer rating and high pricing.

Reference with replaced verbatim values:
There is an Italian place, X-name, situated near X-near with average customer rating and high pricing.

Figure 1: Sample MR and corresponding NL reference from the E2E dataset.

put is a NL sentence, and the dataset provides multiple possible NL references per MR. Given that there is at least one available NL reference in the dataset that fully expresses all the available attributes in the corresponding MR, we aim to fluently express all available attributes and values in our models' output. Figure 1 shows a sample from the E2E dataset (Novikova et al., 2017).

Each attribute has a single value, and each attribute corresponds to a specific value data type, e.g. `familyFriendly` takes boolean values, and `food` takes values from a closed set dictionary of different food types. The attributes `name` and `near` are the only ones that may take any string value, but their values are guaranteed to appear verbatim in the NL reference. To minimize noise in the training signal, we preprocess all MR-NL pairs and replace the values of `name` and `near` with variables ("X-") in the MRs and NLs (see Figure 1).

3 Imitation learning system

This section describes the first system and its variations (namely LOLS-NLG) we submitted to the E2E challenge. LOLS-NLG employ imitation learning to train structured prediction NLG models, and is based on previous work by Lampouras and Vlachos (2016). In this paper, we will briefly describe the system, focusing on adjustments pertaining to this task, but we refer the reader to the original article for a more detailed overview.

To begin, we formulate the generation of a NL sentence from a MR as a sequence of two types of actions, content prediction actions a_c and word prediction actions a_w (see Figure 2). To generate an NL sentence, each content prediction action selects which attribute c (and corresponding value) should be expressed next, in effect ordering the set of available attributes in the MR. Once the content prediction action sequence is completed, for each selected attribute c, we generate a sequence of words chosen from its corresponding dictionary D_c. This dictionary consists of all the words that we have observed to co-occur with attribute c in the training data. From the action sequence produced, it is straightforward to derive the final sentence by keeping the word prediction actions; the content prediction actions are discarded.

The aforementioned content and word actions are generated from trained classifiers; we use adaptive regularization of weight vectors (AROW) classifiers (Crammer et al., 2013). To train them we employ the imitation learning Locally Optimal Learning to Search (LOLS) framework (Chang et al., 2015). LOLS explores the search space of possible action sequences by considering alternative action trajectories at each time-step, while avoiding explicitly enumerating all possible trajectories. Additionally, imitation learning allows us to train classifiers using non-decomposable loss functions (e.g. BLEU in the case of NLG) by requiring only evaluation of the complete output NL sentence instead of distinct content or word actions.

3.1 Expert policy

During training, an expert policy (also referred to as dynamic oracle) is required to determine which content or word action would be optimal given the current state of generation (i.e. which words have been predicted already) and the gold standard NL references corresponding to that instance's MR.

If a word action is to be predicted by the expert policy we locate the word that makes the sequence (ignoring content actions) best match (i.e. minimize the loss function to) the NL reference. To determine an optimal content action, the expert policy needs to determine their order in the NL reference. To estimate this, we match the attribute values with unaligned word n-grams in the reference, based on their Levenshtein distance. Specifically, for boolean attributes, we match the attribute name (e.g. `family_friendly=yes` to "it is a family restaurant") instead of the value.

As the loss function for training, we use the harmonic mean of BLEU-4, ROUGE-4 and ERR(%). ERR(%) tries to estimate how many attribute values in the MR are not expressed in the NL generated sentence; it considers the word actions exclusively. Content actions, which are ignored by the loss function as they are not part of the sentence, are evaluated based on their impact on word actions that follow.

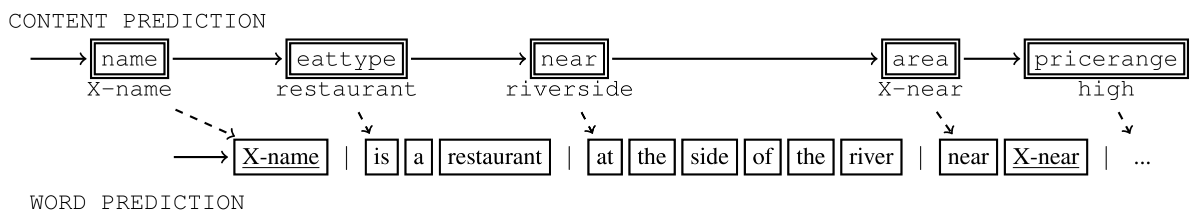

Figure 2: Example of the NLG process with content and word prediction action sequences.

3.2 Reducing the action space

Each MR in the E2E dataset has multiple references, which while beneficial since they capture lexical variation for the same meaning, also pose a challenge to model learning as they provide additional, and possibly infrequent, actions and ambiguous training signal. We experimented with reducing the available NL references per MR during training, to a single sentence, with the goal of reducing model complexity and training times. This limits the lexical variety of the output, but we do not consider variety as a goal of the model. Under this configuration, for each MR we only kept the reference whose words had the highest average frequency in the training set, among all available.

To further reduce the computational complexity introduced by the number of word actions in the E2E dataset, we modified LOLS to use targeted exploration (Goodman et al., 2016) during training. This reduces the number of alternative action trajectories that LOLS explores during training, to those considered optimal by the expert policy, and the ones that are highly scored by the classifier.

4 Encoder-decoder systems

In this section, we describe the second system, and variations, (namely ENC-DEC-NLG) that we submitted to the E2E challenge. ENC-DEC-NLG use encoder-decoder architectures to generate an output sequence of words, given a particular input MR. In contrast to LOLS-NLG, ENC-DEC-NLG only generates words and no content actions. We present two variations of this system, one based on an LSTM2LSTM architecture and another based on a CNN2LSTM architecture.

4.1 LSTM2LSTM architecture

The first variation of our ENC-DEC-NLG system, employs Long Short-term Memory (LSTM) recurrent networks (Hochreiter and Schmidhuber, 1997) for both its encoder and decoder, with an added attention mechanism (Bahdanau et al., 2014) over the input sequence. The architecture we employ is similar to the one used by Dušek and Jurcicek (2016), with the exception that this system does not perform any reranking on the outputs of the beam search after decoding.

We start by converting the input MR into a sequence of attributes (a_i) and corresponding values (v_i): $x = \{a_1, v_1, ..., a_n, v_n\}$. We ensure that the order and position in the input sequence remains consistent. For training we use the Adam (Kingma and Ba, 2014) optimizer while employing cross entropy to calculate the loss. To obtain the final sentence we use beam search (with a beam size of 3) over the decoder's output (softmax) layer.

4.2 CNN2LSTM architecture

As discussed in the previous section, in the LSTM2LSTM architecture, we ensure a consistent order on how the attributes and values of the MR are input in the encoder. However, this may bias the model towards a particular attribute order in the output sentence. To minimize this bias, we also experiment with a second variation on our ENC-DEC-NLG system, where the input sequence $x = \{a_1, v_1, ..., a_n, v_n\}$ is encoded with a convolutional neural network (CNN) (Lecun et al., 1998), instead of an LSTM encoder.

The CNN encoder is defined as follows:

$$h = conv2d\,(x_i) \tag{1}$$

$$e = maxpooling\,(relu\,(h + b)) \tag{2}$$

where x_i is the i-th component of the input sequence (it may correspond to an attribute or value), b is a bias, and e is the CNN encoder's output.

In our CNN2LSTM variant, we introduce a hierarchical attention network to the LSTM decoder, described by the following equations, and partly inspired by the work of Yang et al. (2016). As described by the equations, the attention considers both the encoder's input and output, due to the latter's convolutional nature (Gehring et al., 2017).

$$a_i = \sum_j (v_i \cdot (h_t + conv2d(x_i))) \tag{3}$$

$$z_{t,i} = a_i \cdot x_i \tag{4}$$

$$z_t = concat\left(z_{t,0} \dots z_{t,|x|}\right) \tag{5}$$

where a_i are weights calculated by the current hidden state h_t and the convolutional output, and v_i are the latent variables. The output of the attention model is the concatenation of the $z_{t,i}$ vectors.

Furthermore, an additional memory cell is used in an attempt to track how attention over the input should change throughout the generation. The attention memory cell is initialized randomly, and is updated at each time-step according to the current attention vector z_i and weights w_1 and w_2.

$$c_t = tanh\,(w_1 \cdot c_{t-1} + w_2 \cdot z_t) \tag{6}$$

Finally, at each time-step, the decoder's output layer considers the hidden state h_t, attention z_t, and attention memory cell c_t, to estimate the probability distribution over all possible words.

$$o = softmax\,(tanh\,(W \cdot (c_t + h_t + z_t) + b)) \tag{7}$$

We should note that the CNN2LSTM model does not use beam search during decoding.

5 Experiments and results

We examined a number of different configurations in our experiments; the BLEU score results from the development set are presented in Table 1. For the LOLS-NLG model we tried configurations that use all available NL references per MR during training (*multi*) or instead use only one (*single*) reference as detailed in section 3.2. The parameter e denotes the number of epochs imitation learning was allowed to run over the training data; $e = 0$ effectively means that no imitation learning was applied to the classifiers. We observe that imitation learning had a consistent but small effect on the results. The differences between *multi* and *single* configurations were also small but we argue that the *single* configuration allows further training of the models (e.g. additional imitation learning epochs) to be computationally feasible.

The automatic results of the two ENC-DEC-NLG system variations, LSTM2LSTM and CNN2LSTM, are also included in Table 1. They seem to outperform the LOLS-NLG variations, but again the differences are not significant. The ENC-DEC-NLG systems always used all available references, and were not examined under the *single* configuration.

System configuration	BLEU
LOLS (*multi*, e=0)	0.68
LOLS (*multi*, e=2)	0.71
LOLS (*single*, e=0)	0.68
LOLS (*single*, e=3)	0.70
LSTM2LSTM	0.71
CNN2LSTM	0.72

Table 1: Automatic evaluation on development set.

A small-scale internal human evaluation was conducted to assess the comparative performance of the LOLS-NLG (*multi*, e=2), LSTM2LSTM, and CNN2LSTM systems. We also included the output of the E2E NLG challenge's baseline model TGen (Dušek and Jurcicek, 2016) to this comparison.

We examined generated sentences from all aforementioned systems for a subset of 60 randomly selected MRs of the development set. Each generated sentence was examined independently of the other systems' output, and scored in terms of naturalness (i.e. how grammatical and fluent the text is) and informativeness (i.e. does the text express all the information in the MR); a Likert scale from 1 to 6 was used. The results are shown in Table 2 and we can observe that the ENC-DEC models tend to be more fluent, but the LOLS-NLG model expresses all available information. However, we would like to stress that this evaluation provides only an indication of comparative quality, as the examined subset was small, and the observed differences were not significant.

System	Informative	Natural
TGen	4.87	4.87
LOLS (*multi*, e=2)	4.97	4.72
LSTM2LSTM	4.77	4.79
CNN2LSTM	4.61	4.82

Table 2: Human evaluation on development set.

Tables 3 and 4 detail the automatic and human evaluation scores achieved by our submitted runs on the challenge's test data, as they are reported by Dušek et al. (2018). The automatic metric results suggest that the LOLS-NLG variations perform bet-

System configuration	BLEU	NIST	METEOR	ROUGE_L	CIDEr
LOLS (*multi*, e=2)	0.6015	8.3075	0.4405	0.6778	2.1775
LOLS (*multi*, e=0)	0.6233	8.1751	0.4378	0.6887	2.284
LOLS (*single*, e=0)	0.569	8.0382	0.4202	0.6348	2.0956
LOLS (*single*, e=3)	0.5799	7.9163	0.431	0.667	2.0691
LSTM2LSTM	0.5436	5.7462	0.3561	0.6152	1.413
CNN2LSTM	0.5356	7.8373	0.3831	0.5513	1.582

Table 3: Automatic evaluation on test set.

	Quality		Naturalness	
System configuration	Trueskill	Cluster	Trueskill	Cluster
LOLS (multiRef, e=2)	-0.012	2	-0.077	3
LSTM2LSTM	-0.457	5	0.211	1

Table 4: Human evaluation on test set.

ter than the ENC-DEC-NLG variations, and that the *multi* configuration performs better than the *single*. The effect of imitation learning is less pronounced than what was observed in the development data.

As pertains the human evaluation of the submitted runs (Table 4), again LSTM2LSTM outperforms LOLS-NLG in terms of fluency, but suffers in terms of quality. Naturalness here is defined in the same way as in our internal human evaluation, but quality is defined as combination of "grammatical correctness, fluency, adequacy and other important factors". In contrast, our informativeness metric only measured whether the sentences contained all the information present in the corresponding MR.

All systems submitted in the challenge were clustered by the organizers using bootstrap resampling ($p \leq 0.05$). The LSTM2LSTM variation was the highest ranked among all participants in terms of naturalness, while the LOLS-NLG was assigned to the second best cluster in terms of quality.

6 Conclusion

We proposed two systems, and their variations; the first employs imitation learning over structured prediction models, while the second uses encoder-decoder architectures to generate the output sequence of words. Our results suggest that encoder-decoder architectures lead to the best results in regards to fluency and naturalness, but suffer in terms of expressing all the available information in the MR. On the other hand, we observed that the imitation learning-based models tend to exhaustively express all the attributes and values in the MR, but their output is less fluent in comparison.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR* abs/1409.0473. http://arxiv.org/abs/1409.0473.

Kai-Wei Chang, Akshay Krishnamurthy, Alekh Agarwal, Hal Daumé III, and John Langford. 2015. Learning to search better than your teacher. In *Proceedings of the International Conference on Machine Learning (ICML)*. http://hal3.name/docs/#daume15lols.

Koby Crammer, Alex Kulesza, and Mark Dredze. 2013. Adaptive regularization of weight vectors. *Machine Learning* 91:155–187.

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.)*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122* .

James Goodman, Andreas Vlachos, and Jason Naradowsky. 2016. Noise reduction and targeted exploration in imitation learning for abstract meaning representation parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Berlin, Germany.

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

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR* abs/1412.6980. http://arxiv.org/abs/1412.6980.

Gerasimos Lampouras and Andreas Vlachos. 2016. Imitation learning for language generation from unaligned data. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics*. Osaka, Japan, pages 1101–1112.

Yann Lecun, Lon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*. pages 2278–2324.

Hongyuan Mei, Mohit Bansal, and Matthew R. Walter. 2016. What to talk about and how? selective generation using lstms with coarse-to-fine alignment. *Proceedings of NAACL* .

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.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J Smola, and Eduard H Hovy. 2016. Hierarchical attention networks for document classification. In *HLT-NAACL*. pages 1480–1489.