A Context-aware Natural Language Generator for Dialogue Systems

Ondřej Dušek and **Filip Jurčíček**
Charles University in Prague, Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
Malostranské náměstí 25, CZ-11800 Prague, Czech Republic
{odusek,jurcicek}@ufal.mff.cuni.cz

Abstract

We present a novel natural language generation system for spoken dialogue systems capable of entraining (adapting) to users' way of speaking, providing contextually appropriate responses. The generator is based on recurrent neural networks and the sequence-to-sequence approach. It is fully trainable from data which include preceding context along with responses to be generated. We show that the context-aware generator yields significant improvements over the baseline in both automatic metrics and a human pairwise preference test.

1 Introduction

In a conversation, speakers are influenced by previous utterances of their counterparts and tend to adapt (align, entrain) their way of speaking to each other, reusing lexical items as well as syntactic structure (Reitter et al., 2006). Entrainment occurs naturally and subconsciously, facilitates successful conversations (Friedberg et al., 2012; Nenkova et al., 2008), and forms a natural source of variation in dialogues. In spoken dialogue systems (SDS), users were reported to entrain to system prompts (Parent and Eskenazi, 2010).

The function of natural language generation (NLG) components in task-oriented SDS typically is to produce a natural language sentence from a *dialogue act* (DA) (Young et al., 2010) representing an action, such as *inform* or *request*, along with one or more attributes (*slots*) and their values (see Fig. 1). NLG is an important component of SDS which has a great impact on the perceived naturalness of the system; its quality can also influence the overall task success (Stoyanchev and Stent, 2009; Lopes et al., 2013). However, typical

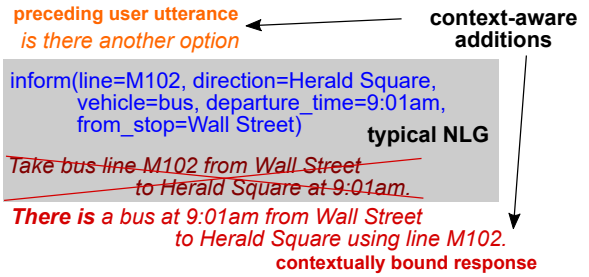

Figure 1: An example of NLG input and output, with context-aware additions.

NLG systems in SDS only take the input DA into account and have no way of adapting to the user's way of speaking. To avoid repetition and add variation into the outputs, they typically alternate between a handful of preset variants (Jurčíček et al., 2014) or use overgeneration and random sampling from a k-best list of outputs (Wen et al., 2015b). There have been several attempts at introducing entrainment into NLG in SDS, but they are limited to rule-based systems (see Section 4).

We present a novel, fully trainable context-aware NLG system for SDS that is able to entrain to the user and provides naturally variable outputs because generation is conditioned not only on the input DA, but also on the preceding user utterance (see Fig. 1). Our system is an extension of Dušek and Jurčíček (2016b)'s generator based on sequence-to-sequence (seq2seq) models with attention (Bahdanau et al., 2015). It is, to our knowledge, the first fully trainable entrainment-enabled NLG system for SDS. We also present our first results on the dataset of Dušek and Jurčíček (2016a), which includes the preceding user utterance along with each data instance (i.e., pair of input meaning representation and output sentence), and we show that our context-aware system outperforms the baseline in both automatic metrics and a human pairwise preference test.

In the following, we first present the architecture of our generator (see Section 2), then give an account of our experiments in Section 3. We include a brief survey of related work in Section 4. Section 5 contains concluding remarks and plans for future work.

2 Our generator

Our seq2seq generator is an improved version of Dušek and Jurčíček (2016b)'s generator, which itself is based on the seq2seq model with attention (Bahdanau et al., 2015, see Fig. 2) as implemented in the TensorFlow framework (Abadi et al., 2015).[1] We first describe the base model in Section 2.1, then list our context-aware improvements in Section 2.2.

2.1 Baseline Seq2seq NLG with Attention

The generation has two stages: The first, *encoder stage* uses a recurrent neural network (RNN) composed of long-short-term memory (LSTM) cells (Hochreiter and Schmidhuber, 1997; Graves, 2013) to encode a sequence of input tokens[2] $\mathbf{x} = \{x_1, \ldots, x_n\}$ into a sequence of hidden states $\mathbf{h} = \{h_1, \ldots, h_n\}$:

$$h_t = \text{lstm}(x_t, h_{t-1}) \quad (1)$$

The second, *decoder stage* then uses the hidden states $\mathbf{h}$ to generate the output sequence $\mathbf{y} = \{y_1, \ldots, y_m\}$. Its main component is a second LSTM-based RNN, which works over its own internal state s_t and the previous output token y_{t-1}:

$$s_t = \text{lstm}((y_{t-1} \circ c_t)W_S, s_{t-1}) \quad (2)$$

It is initialized by the last hidden encoder state ($s_0 = h_n$) and a special starting symbol. The generated output token y_t is selected from a softmax distribution:

$$p(y_t|y_{t-1} \ldots, \mathbf{x}) = \text{softmax}((s_t \circ c_t)W_Y) \quad (3)$$

In (2) and (3), c_t represents the *attention model* – a sum over all encoder hidden states, weighted by a feed-forward network with one $\tanh$ hidden layer; W_S and W_Y are linear projection matrices and "$\circ$" denotes concatenation.

DAs are represented as sequences on the encoder input: a triple of the structure "DA type, slot, value" is created for each slot in the DA and the triples are concatenated (see Fig. 2). [3] The generator supports greedy decoding as well as beam search which keeps track of top k most probable output sequences at each time step (Sutskever et al., 2014; Bahdanau et al., 2015).

The generator further features a simple content classification reranker to penalize irrelevant or missing information on the output. It uses an LSTM-based RNN to encode the generator outputs token-by-token into a fixed-size vector. This is then fed to a sigmoid classification layer that outputs a 1-hot vector indicating the presence of all possible DA types, slots, and values. The vectors for all k-best generator outputs are then compared to the input DA and the number of missing and irrelevant elements is used to rerank them.

2.2 Making the Generator Context-aware

We implemented three different modifications to our generator that make its output dependent on the preceding context: [4]

Prepending context. The preceding user utterance is simply prepended to the DA and fed into the encoder (see Fig. 2). The dictionary for context utterances is distinct from the DA tokens dictionary.

Context encoder. We add another, separate encoder for the context utterances. The hidden states of both encoders are concatenated, and the decoder then works with double-sized vectors both on the input and in the attention model (see Fig. 2).

n-gram match reranker. We added a second reranker for the k-best outputs of the generator that promotes outputs that have a word or phrase overlap with the context utterance. We use geometric mean of modified n-gram precisions (with $n \in \{1, 2\}$) as a measure of context overlap, i.e., BLEU-2 (Papineni et al., 2002) without brevity penalty. The log probability l of an output sequence on the generator k-best list is updated as follows:

$$l = l + w \cdot \sqrt{p_1 p_2} \quad (4)$$

[1] See (Dušek and Jurčíček, 2016b) and (Bahdanau et al., 2015) for a more formal description of the base model.

[2] Embeddings are used (Bengio et al., 2003), i.e., x_t and y_t are vector representations of the input and output tokens.

[3] While the sequence encoding may not necessarily be the best way to obtain a vector representation of DA, it was shown to work well (Dušek and Jurčíček, 2016b).

[4] For simplicity, we kept close to the basic seq2seq architecture of the generator; other possibilities for encoding the context, such as convolution and/or max-pooling, are possible.

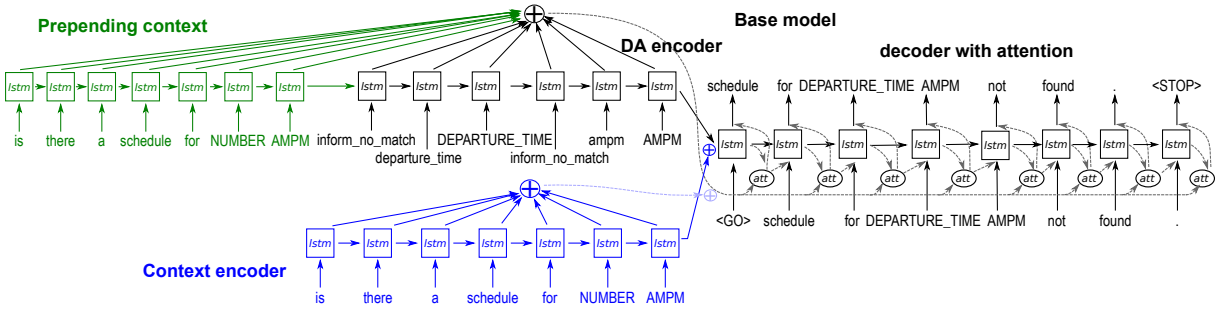

Figure 2: The base Seq2seq generator (black) with our improvements: prepending context (green) and separate context encoder (blue).

Setup	BLEU	NIST
Baseline (context not used)	66.41	7.037
n-gram match reranker	68.68	7.577
Prepending context	63.87	6.456
+ n-gram match reranker	69.26	7.772
Context encoder	63.08	6.818
+ n-gram match reranker	69.17	7.596

Table 1: BLEU and NIST scores of different generator setups on the test data.

In (4), p_1 and p_2 are modified unigram and bigram precisions of the output sequence against the context, and w is a preset weight. We believe that any reasonable measure of contextual match would be viable here, and we opted for modified n-gram precisions because of simple computation, well-defined range, and the relation to the de facto standard BLEU metric.[5] We only use unigrams and bigrams to promote especially the reuse of single words or short phrases.

In addition, we combine the n-gram match reranker with both of the two former approaches.

We used gold-standard transcriptions of the immediately preceding user utterance in our experiments in order to test the context-aware capabilities of our system in a stand-alone setting; in a live SDS, 1-best speech recognition hypotheses and longer user utterance history can be used with no modifications to the architecture.

3 Experiments

We experiment on the publicly available dataset of Dušek and Jurčíček (2016a)[6] for NLG in the public transport information domain, which includes preceding context along with each pair of input DA and target natural language sentence. It contains over 5,500 utterances, i.e., three paraphrases for each of the over 1,800 combinations of input DA and context user utterance. The data concern bus and subway connections on Manhattan, and comprise four DA types (*iconfirm*, *inform*, *inform_no_match*, *request*). They are delexicalized for generation to avoid sparsity, i.e., stop names, vehicles, times, etc., are replaced by placeholders (Wen et al., 2015a). We applied a 3:1:1 split of the set into training, development, and test data. We use the three paraphrases as separate instances in training data, but they serve as three references for a single generated output in validation and evaluation.

We test the three context-aware setups described in Section 2.2 and their combinations, and we compare them against the baseline non-context-aware seq2seq generator. Same as Dušek and Jurčíček (2016b), we train the seq2seq models by minimizing cross-entropy on the training set using the Adam optimizer (Kingma and Ba, 2015), and we measure BLEU on the development set after each pass over the training data, selecting the best-performing parameters.[7] The content classification reranker is trained in a similar fashion, measuring misclassification on both training and development set after each pass.[8] We use 5 different random initializations of the networks and

[5] We do not use brevity penalty as we do not want to demote shorter output sequences. However, adding it to the formula in our preliminary experiments yielded similar results to the ones presented here.

[6] The dataset is released at `http://hdl.handle.net/11234/1-1675`; we used a more recent version from GitHub (`https://github.com/UFAL-DSG/alex_context_nlg_dataset`), which contains several small fixes.

[7] Based on our preliminary experiments on development data, we use embedding size 50, LSTM cell size 128, learning rate 0.0005, and batch size 20. Training is run for at least 50 and up to 1000 passes, with early stopping if the top 10 validation BLEU scores do not change for 100 passes.

[8] We use the same settings except for the number of passes over the training data, which is at least 20 and 100 at most. For validation, development set is given 10 times more importance than the training set.

average the results.

Decoding is run with a beam size of 20 and the penalty weight for content classification reranker set to 100. We set the n-gram match reranker weight based on experiments on development data.[9]

3.1 Evaluation Using Automatic Metrics

Table 1 lists our results on the test data in terms of the BLEU and NIST metrics (Papineni et al., 2002; Doddington, 2002). We can see that while the n-gram match reranker brings a BLEU score improvement, using context prepending or separate encoder results in scores lower than the baseline.[10] However, using the n-gram match reranker together with context prepending or separate encoder brings significant improvements of about 2.8 BLEU points in both cases, better than using the n-gram match reranker alone.[11] We believe that adding the context information into the decoder does increase the chances of contextually appropriate outputs appearing on the decoder k-best lists, but it also introduces a lot more uncertainty and therefore, the appropriate outputs may not end on top of the list based on decoder scores alone. The n-gram match reranker is then able to promote the relevant outputs to the top of the k-best list. However, if the generator itself does not have access to context information, the n-gram match reranker has a smaller effect as contextually appropriate outputs may not appear on the k-best lists at all. A closer look at the generated outputs confirms that entrainment is present in sentences generated by the context-aware setups (see Fig. 2).

In addition to BLEU and NIST scores, we measured the slot error rate ERR (Wen et al., 2015b), i.e., the proportion of missing or superfluous slot placeholders in the delexicalized generated outputs. For all our setups, ERR stayed around 3%.

3.2 Human Evaluation

We evaluated the best-performing setting based on BLEU/NIST scores, i.e., prepending context with n-gram match reranker, in a blind pairwise preference test with untrained judges recruited on the CrowdFlower crowdsourcing platform.[12] The judges were given the context and the system output for the baseline and the context-aware system, and they were asked to pick the variant that sounds more natural. We used a random sample of 1,000 pairs of different system outputs over all 5 random initializations of the networks, and collected 3 judgments for each of them. The judges preferred the context-aware system output in 52.5% cases, significantly more than the baseline.[13]

We examined the judgments in more detail and found three probable causes for the rather small difference between the setups. First, both setups' outputs fit the context relatively well in many cases and the judges tend to prefer the overall more frequent variant (e.g., for the context "starting from Park Place", the output "Where do you want to go?" is preferred over "Where are you going to?"). Second, the context-aware setup often selects a shorter response that fits the context well (e.g., "Is there an option at 10:00 am?" is confirmed simply with "At 10:00 am."), but the judges seem to prefer the more eloquent variant. And third, both setups occasionally produce non-fluent outputs, which introduces a certain amount of noise.

4 Related Work

Our system is an evolutionary improvement over the LSTM seq2seq system of Dušek and Jurčíček (2016b) and as such, it is most related in terms of architecture to other recent RNN-based approaches to NLG, which are not context-aware: RNN generation with a convolutional reranker by Wen et al. (2015a) and an improved LSTM-based version (Wen et al., 2015b), as well as the LSTM encoder-aligner-decoder NLG system of Mei et al. (2015). The recent end-to-end trainable SDS of Wen et al. (2016) does have an implicit access to previous context, but the authors do not focus on its influence on the generated responses.

There have been several attempts at modelling entrainment in dialogue (Brockmann et al., 2005; Reitter et al., 2006; Buschmeier et al., 2010) and even successful implementations of entrainment models in NLG systems for SDS, where entrainment caused an increase in perceived naturalness of the system responses (Hu et al., 2014) or increased naturalness and task success (Lopes et al.,

[9] w is set to 5 when the n-gram match reranker is run by itself or combined with the separate encoder, 10 if combined with prepending context.

[10] In our experiments on development data, all three methods brought a mild BLEU improvement.

[11] Statistical significance at 99% level has been assessed using pairwise bootstrap resampling (Koehn, 2004).

[12] `http://crowdflower.com`

[13] The result is statistically significant at 99% level according to the pairwise bootstrap resampling test.

Context	is there a later option
Input DA	iconfirm(alternative=next)
Baseline	Next connection.
n-gram match reranker	You want a later connection.
Prepending context + n-gram match reranker	You want a later connection.
Context encoder + n-gram match reranker	You want a later option.
Context	i need to find a bus connection
Input DA	inform_no_match(vehicle=bus)
Baseline	No bus found, sorry.
n-gram match reranker	I did not find a bus route.
Prepending context + n-gram match reranker	I'm sorry, I cannot find a bus connection.
Context encoder + n-gram match reranker	I'm sorry, I cannot find a bus connection.
Context	i rather take the bus
Input DA	inform(vehicle=bus, departure_time=8:01am, direction=Cathedral Parkway, from_stop=Bowling Green, line=M15)
Baseline	At 8:01am by bus line M15 from Bowling Green to Cathedral Parkway.
n-gram match reranker	At 8:01am by bus line M15 from Bowling Green to Cathedral Parkway.
Prepending context + n-gram match reranker	You can take the M15 bus from Bowling Green to Cathedral Parkway at 8:01am.
Context encoder + n-gram match reranker	At 8:01am by bus line M15 from Bowling Green to Cathedral Parkway.

Table 2: Example outputs of the different setups of our generator (with entrainment highlighted)

2013; Lopes et al., 2015). However, all of the previous approaches are completely or partially rule-based. Most of them attempt to model entrainment explicitly, focus on specific entrainment phenomena only, and/or require manually selected lists of variant expressions, while our system learns synonyms and entrainment rules implicitly from the corpus. A direct comparison with previous entrainment-capable NLG systems for SDS is not possible in our stand-alone setting since their rules involve the history of the whole dialogue whereas we focus on the preceding utterance in our experiments.

5 Conclusions and Further Work

We presented an improvement to our natural language generator based on the sequence-to-sequence approach (Dušek and Jurčíček, 2016b), allowing it to exploit preceding context user utterances to adapt (entrain) to the user's way of speaking and provide more contextually accurate and less repetitive responses. We used two different ways of feeding previous context into the generator and a reranker based on n-gram match against the context. Evaluation on our context-aware dataset (Dušek and Jurčíček, 2016a) showed a significant BLEU score improvement for the combination of the two approaches, which was confirmed in a subsequent human pairwise preference test. Our generator is available on GitHub at the following URL:

`https://github.com/UFAL-DSG/tgen`

In future work, we plan on improving the n-gram matching metric to allow fuzzy matching (e.g., capturing different forms of the same word), experimenting with more ways of incorporating context into the generator, controlling the output eloquence and fluency, and most importantly, evaluating our generator in a live dialogue system. We also intend to evaluate the generator with automatic speech recognition hypotheses as context and modify it to allow n-best hypotheses as contexts. Using our system in a live SDS will also allow a comparison against previous handcrafted entrainment-capable NLG systems.

Acknowledgments

This work was funded by the Ministry of Education, Youth and Sports of the Czech Republic under the grant agreement LK11221 and core research funding, SVV project 260 333, and GAUK grant 2058214 of Charles University in Prague. It used language resources stored and distributed by the LINDAT/CLARIN project of the Ministry of Education, Youth and Sports of the Czech Republic (project LM2015071). The authors would like to thank Ondřej Plátek and Miroslav Vodolán for helpful comments.

References

M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

D. Bahdanau, K. Cho, and Y. Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*. arXiv:1409.0473.

Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

C. Brockmann, A. Isard, J. Oberlander, and M. White. 2005. Modelling alignment for affective dialogue. In *Workshop on Adapting the Interaction Style to Affective Factors at the 10th International Conference on User Modeling*.

H. Buschmeier, K. Bergmann, and S. Kopp. 2010. Modelling and evaluation of lexical and syntactic alignment with a priming-based microplanner. In *Empirical Methods in Natural Language Generation*, number 5790 in Lecture Notes in Computer Science, pages 85–104. Springer.

G. 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*, pages 138–145.

O. Dušek and F. Jurčíček. 2016a. A context-aware natural language generation dataset for dialogue systems. In *Workshop on Collecting and Generating Resources for Chatbots and Conversational Agents - Development and Evaluation*, pages 6–9.

O. Dušek and F. Jurčíček. 2016b. Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings. *arXiv:1606.05491*. To appear in Proceedings of ACL.

H. Friedberg, D. Litman, and S. B. F. Paletz. 2012. Lexical entrainment and success in student engineering groups. In *Proc. of SLT*, pages 404–409.

A. Graves. 2013. Generating sequences with recurrent neural networks. *arXiv:1308.0850*.

S. Hochreiter and J. Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Z. Hu, G. Halberg, C. Jimenez, and M. Walker. 2014. Entrainment in pedestrian direction giving: How many kinds of entrainment. In *Proc. of IWSDS*, pages 90–101.

F. Jurčíček, O. Dušek, O. Plátek, and L. Žilka. 2014. Alex: A statistical dialogue systems framework. In *Proc. of Text, Speech and Dialogue*, pages 587–594.

D. Kingma and J. Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*. arXiv:1412.6980.

P. Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP*, pages 388–395.

J. Lopes, M. Eskenazi, and I. Trancoso. 2013. Automated two-way entrainment to improve spoken dialog system performance. In *Proc. of ICASSP*, pages 8372–8376.

J. Lopes, M. Eskenazi, and I. Trancoso. 2015. From rule-based to data-driven lexical entrainment models in spoken dialog systems. *Computer Speech & Language*, 31(1):87–112.

H. Mei, M. Bansal, and M. R. Walter. 2015. What to talk about and how? selective generation using LSTMs with coarse-to-fine alignment. *arXiv:1509.00838*.

A. Nenkova, A. Gravano, and J. Hirschberg. 2008. High frequency word entrainment in spoken dialogue. In *Proc. of ACL-HLT*, pages 169–172.

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proc. of ACL*, pages 311–318.

G. Parent and M. Eskenazi. 2010. Lexical entrainment of real users in the Let's Go spoken dialog system. In *Proc. of Interspeech*, pages 3018–3021.

D. Reitter, F. Keller, and J. D. Moore. 2006. Computational modelling of structural priming in dialogue. In *Proc. of NAACL-HLT: Short Papers*, pages 121–124.

S. Stoyanchev and A. Stent. 2009. Lexical and syntactic priming and their impact in deployed spoken dialog systems. In *Proc. of NAACL-HLT*, pages 189–192.

I. Sutskever, O. Vinyals, and Q. VV Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112. arXiv:1409.3215.

T.-H. Wen, M. Gasic, D. Kim, N. Mrksic, P.-H. Su, D. Vandyke, and S. Young. 2015a. Stochastic language generation in dialogue using recurrent neural networks with convolutional sentence reranking. In *Proc. of SIGDIAL*, pages 275–284.

T.-H. Wen, M. Gasic, N. Mrkšić, P.-H. Su, D. Vandyke, and S. Young. 2015b. Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In *Proc. of EMNLP*, pages 1711–1721.

T.-H. Wen, M. Gašić, N. Mrkšić, L. M. Rojas-Barahona, P.-H. Su, S. Ultes, D. Vandyke, and S. Young. 2016. A network-based end-to-end trainable task-oriented dialogue system. *arXiv:1604.04562*.

S. Young, M. Gašić, S. Keizer, F. Mairesse, J. Schatzmann, B. Thomson, and K. Yu. 2010. The hidden information state model: A practical framework for POMDP-based spoken dialogue management. *Computer Speech & Language*, 24(2):150–174.