A char-based seq2seq submission to the *E2E NLG Challenge*

Shubham Agarwal
Heriot-Watt University [*]
Edinburgh, UK
sa201@hw.ac.uk

Marc Dymetman
NAVER Labs Europe[†]
Grenoble, France
marc.dymetman@naverlabs.com

Éric Gaussier
Université Grenoble Alpes
Grenoble, France
Eric.Gaussier@imag.fr

Abstract

The paper accompanies our submission to the E2E NLG Challenge. This task involves generating human understandable utterances from slot-value pair Meaning Representations (or dialogue acts). Recently, word based neural network approaches (particularly Sequence-to-Sequence based approaches) came into prominence which resorted to a pre- (respectively post-) processing step called *delexicalization* (relexicalization) to handle the rare word problem. In contrast, we first train a character level seq2seq model with attention mechanism which requires no pre/post-processing (delexicalization, tokenization or even lowercasing). The utterances generated by this model are rated excellent in terms of fluency as well as quite reasonable in adequacy, the primary downside being the possible omission of semantic material (slot values). However, in a significant number of cases, a perfect solution can be found in the top-k list. Thus, for further improvement, we explore two different re-ranking approaches to score candidates. During the process, we also introduce a synthetic dataset creation procedure, which opens up new directions for creating artificial datasets for Natural Language Generation.

1 Introduction

Natural Language Generation from Dialogue Acts involves generating human understandable utterances from slot-value pairs in a Meaning Representation (MR). This is a component in Spoken Dialogue Systems, where recent advances in Deep Learning are stimulating interest towards using end-to-end models.

Recurrent Neural Networks with gated cell variants such as LSTMs and GRUs (Hochreiter and Schmidhuber, 1997; Cho et al., 2014) are now extensively used in Natural Language Processing, capitalizing on their ability to model sequential data, where one treats the input text as a sequence of words. This class of neural networks when integrated in a Sequence to Sequence (Cho et al., 2014; Sutskever et al., 2014) framework have produced state-of-art results in Machine Translation (Cho et al., 2014; Sutskever et al., 2014; Bahdanau et al., 2015), Conversational Modeling (Vinyals and Le, 2015), Semantic Parsing (Xiao et al., 2016) and Natural Language Generation (Wen et al., 2015; Mei et al., 2015).

While these models were initially developed to be used at word level in NLP related tasks, there has been a recent interest to use character level sequences, as in Machine Translation (Chung et al., 2016; Zhao and Zhang, 2016; Ling et al., 2016).

Agarwal and Dymetman (2017) describes our initial approach towards the E2E NLG challenge (Novikova et al., 2017). We trained a char2char model with attention mechanism. Moving forward, we employed two different strategies for re-ranking the top predictions to generate the best utterance.

Our contributions in this paper and challenge can, thus, be summarized as:

1. We show how a vanilla character-based sequence-to-sequence model performs successfully on the challenge test dataset in terms of BLEU score, while having a tendency to omit semantic material. However, we found that in significant number of cases a perfect solution ('oracle prediction') can be found in the n-best (or top-k) list obtained using beam search. This opens up the space for application of re-ranking techniques.

2. We propose a novel and very natural data augmentation technique in Natural Language

[*]Work done during internship at Naver Labs (Previously *Xerox Research Centre Europe*.)
[†]Previously *Xerox Research Centre Europe*.

Generation (NLG) with consists in 'editing' the Meaning Representation (MR) and using the original ReFerences (RF). This fabricated dataset will help us in extracting features (to detect omissions), used for re-ranking in the generated candidates (Section 3.2).

3. We introduce two different re-ranking strategies corresponding to our primary and secondary submission (in the challenge), defined in Section 3.3.

2 Related Work

Traditionally, the Natural Language Generation (NLG) component in Spoken Dialogue Systems has been rule-based, involving a two stage pipeline: 'sentence planning' (deciding the overall structure of the sentence) and 'surface realization' (which renders actual utterances using this structure). The resulting utterances using these rule-based systems tend to be rigid, repetitive and limited in scope. Recent approaches in dialogue generation tend to directly learn the utterances from data (Mei et al., 2015; Lampouras and Vlachos, 2016; Dušek and Jurčíček, 2016; Wen et al., 2015).

Most of the RNN-based approaches to Natural Language Generation (NLG) generate the output word-by-word and resort to delexicalization (a process in which they replace named entities (slot values) with special 'placeholders' (Wen et al., 2015)) or copy mechanisms (Gu et al., 2016) to handle rare or unknown words (possibly out-of-vocabulary (OOV) words, even with a large vocabulary), for instance restaurant names or telephone numbers. It can be argued that this delexicalization is unable to account for phenomena such as morphological agreement (gender, numbers) in the generated text (Sharma et al., 2016; Nayak et al., 2017).

One motivation to use char2char models for dialogue generation comes from the above mentioned drawbacks: we do not have to employ any pre- or post-processing (not even tokenization or lowercasing) technique. Goyal et al. (2016) employ a char-based seq2seq model where the input MR is simply represented as a character sequence, and the output is also generated char-by-char; avoiding the rare word problem, as the character vocabulary is very small. Inspired by their work, we experimented with different numbers of layers in the encoder and decoder as well as different beam widths, while using a bi-directional encoder along with an attention mechanism. While they used an additional finite-state mechanism to guide the production of well-formed (and input-motivated) character sequences, our char-based model never produced non-words, contrary to their findings. This can probably be attributed to the nature of the comparatively larger E2E NLG dataset which contains a small number of different entity names (restaurant names, locations) and no addresses and telephone numbers. Our vanilla char2char model uses character embeddings instead of one-hot encodings and a non-null 'length-penalty' (i.e. length normalization (Wu et al., 2016)) when using beam search for inference. As also observed by (Britz et al., 2017), using a non-zero length-penalty significantly improves the decoding results, for the same beam width.

Recently, *tf-seq2seq*[1] was released as a general encoder-decoder framework based on Tensorflow (Abadi et al., 2016), provided along with Britz et al. (2017), with some standard configuration options. We experimented with some of these settings and found they worked well for our model.

This work follows up from Agarwal and Dymetman (2017) and can be considered as a more extensive proposal in the context of the challenge submission. We further explore re-ranking techniques in order to identify the perfect utterance 'oracle prediction'. One of the strategies for re-ranking uses an approach similar to the 'inverted generation' technique of (Chisholm et al., 2017). Konstas et al. (2017) also trained a similar reverse model for parsing and generation. The synthetic data creation technique is also used by Dušek et al. (2017) but as far as we know, our protocol is novel.

3 Model

In the sequel, we will refer to our vanilla char2char model with the term Forward Model.

3.1 Forward Model

We use a Character-based Sequence-to-Sequence (commonly termed as Encoder Decoder Recurrent Neural Networks (RNN)) model (Sutskever et al., 2014; Cho et al., 2014) with attention mechanism (Bahdanau et al., 2015). We feed a sequence of embeddings of the individual characters composing the source Meaning Representation (MR) - seen as a string- to the Encoder RNN and try to

[1]`https://github.com/google/seq2seq`.

predict the character sequence of the corresponding utterances (RF) in the generation stage with the Decoder RNN.

Coupled with the attention mechanism, Sequence to Sequence models have become de-facto standard in generation tasks. The encoder RNN embeds each of the source characters into vectors exploiting the hidden states computed by the RNN. The decoder RNN predicts the next character based on its current hidden state, previous character, and also the "context" vector c_i, computed by the attention model.

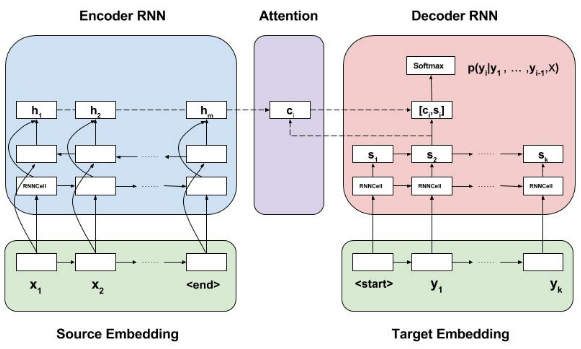

Figure 1: Vanilla Seq2Seq architecture with attention mechanism (source (Britz et al., 2017)). We use character-based sequences for generating utterances.

Figure 1 provides an overview of the framework. While many options are configurable (number of layers, unidirectional *vs* bidirectional encoder, additive *vs* multiplicative attention mechanism, GRU (Cho et al., 2014) *vs* LSTM cells (Hochreiter and Schmidhuber, 1997), etc.), the core architecture is common to all models (Bahdanau et al., 2015; Luong et al., 2015).

While several strategies have been proposed to improve results using Beam Search in Machine Translation (Freitag and Al-Onaizan, 2017), we used the length normalization (aka length penalty) approach Wu et al. (2016) for our task. A heuristically derived length penalty term is added to the scoring function which ranks the probable candidates used to generate the best prediction.

$$lp(Y) = \frac{(5 + |Y|)^\alpha}{(5 + 1)^\alpha} \tag{1}$$

$$s(Y, X) = \log(P(Y|X))/lp(Y) \tag{2}$$

where $\alpha \in (0, 1]$ is the length penalty factor. $\alpha > 0$ encourages longer sequences while a value of $\alpha = 0$ reverts back to traditional beam search. $s(Y, X)$ denotes the scoring function that rank candidate utterances produced by the vanilla model.

3.2 Protocol for synthetic dataset creation

We artificially create a training set for the classifier (defined in Section 3.3.2) to detect errors (primarily omission) in the generated utterances, by a data augmentation technique. Because of the systematic structure of the slots in MR, this gives us freedom to naturally augment data for our use case. To the best of our knowledge, this is the first approach of using data augmentation in this way and opens up new directions to create artificial datasets for Natural Language Generation. We define the procedure first for creating a dataset to detect omission and then we show how a similar approach can be used to create a synthetic dataset to detect additions.

The basic idea assumes that there are no omissions in RF for a given MR. These can be considered as positive pairs when detecting omissions. Now if we artificially add another slot to the original MR and use the same RF for this new MR, naturally the original RF tends to show omission of the added slot.

$$MR_{original} \xrightarrow{\text{+ Added slot}} MR_{new} \tag{3}$$

This is now a two stage procedure:

- Select a slot to add.
- Select a corresponding slot value.

Instead of sampling a particular slot to add, we add all the slots that could be augmented in the MR apart from the currently present slots, one by one. Having chosen the slot type to be added, we add the slot value according to the probability distribution of the slot values for that slot type. The original (MR,RF) pair is assigned a class label of 1 while the new artificial pairs a label of 0, denoting a case of omission. Thus, these triplets (MR, RF, Class Label) allow us to treat this as a classification task.

On the other hand, when we want to create a dataset which would be used for training our model to detect additions, we systematically remove one slot in the original MR to create new MRs. We do not remove the 'name' slot as it was present in all the MRs.

$$MR_{original} \xrightarrow{\text{- Selected slot}} MR_{new} \tag{4}$$

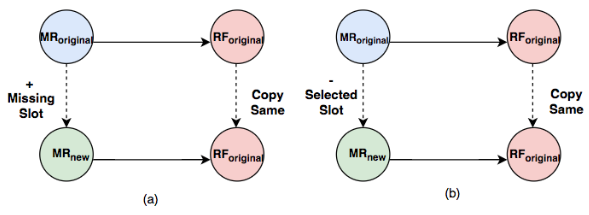

Figure 2: Our data augmentation approach. (a) shows the procedure to create a dataset which can be used to train our model to detect omissions. (b) on the other hand creates a dataset to detect additions.

In both cases, we control the procedure by manipulating MRs instead of the Natural Language RF. This kind of augmented dataset opens up the possibility of using any classifier to detect these kinds of errors.

3.3 Re-ranking Models

In this section, we define two techniques to re-rank the n-best list and these serve as primary and secondary submissions to the challenge.

3.3.1 Reverse Model

We generated a list of top-k predictions (using Beam Search) for each MR in what we call the *forward* phase of the model. In parallel, we trained a *reverse* model which tries to reconstruct the MR given the target RF, similar to the autoencoder model by Chisholm et al. (2017) (See Figure 3). This is guided by an intuition that if our prediction omits some information, the reverse reconstruction of MR would also tend to omit slot-value pairs for the omitted slot values in the prediction. We then score and re-rank the top-k predictions based on a distance metric, namely the edit distance between the original MR and the MR generated by the reverse model, starting from the utterance predicted in the forward direction. Edit distance is commonly used in Natural Language Processing to capture the dissimilarity between two strings, in our case, original MR and reconstructed MR.

To avoid defining the weights to combine edit distance with the log probability of the model, we used a simplified mechanism. At the time of re-ranking, we choose the first output in our n-best list with zero edit distance. If no such prediction can be found, we rely upon the first prediction in our (probabilistically) sorted n-best list. The pipeline for this approach is depicted in Fig. 4.

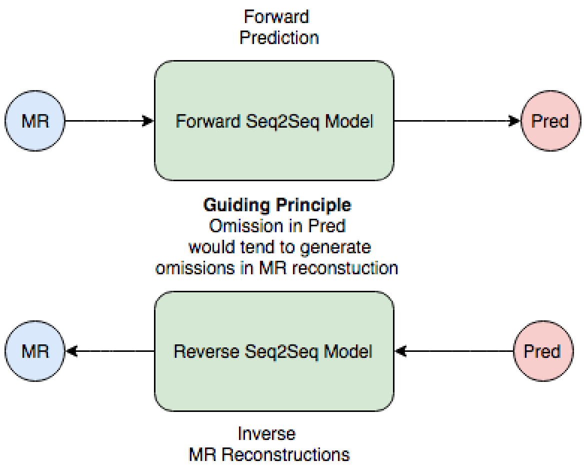
Forward Prediction
MR
Forward Seq2Seq Model
Pred
Guiding Principle
Omission in Pred would tend to generate omissions in MR reconstuction
MR
Reverse Seq2Seq Model
Pred
Inverse MR Reconstructions

Figure 3: In the forward direction we try to model RF using MR. In the reverse direction we construct inverse generations of the MR from RF.

3.3.2 Classifier as a re-ranker

To treat omission (or more generally any kind of *semantic adequacy* mis-representation such as repetition or addition of content) in the predictions as a classification task, we developed a dataset (consisting of triplets) using a protocol defined earlier. However, to train the classifier we relied on hand-crafted features based on string matching in the prediction (with corresponding slot value in the MR) [2]. In total, there were 7 features, corresponding to each slot; as our human evaluation Agarwal and Dymetman (2017) showed no issues while copying restaurant names, we did not have a corresponding feature. To maintain the class balance, we replicated the original (MR,RF) pair (with a class label of 1) for each artificially generated (MR,RF) pair (with a class label of 0; corresponding to omissions).

We used a logistic regression classifier to detect omissions following a similar re-ranking strategy as for the reverse model. For each probable candidate by the forward model, we first extracted these features to have a prediction label by the logistic regression classifier. The first output in our n-best list with a class label 1 is then chosen as the resulting utterance. As a fallback mechanism, we rely on the best prediction by the forward model (similar to the reverse model).

[2]The corresponding string templates used to define these features can be accessed at `https://docs.google.com/spreadsheets/d/1b-lm45TmowfZztk842c6UF59Z8rimPfzrE8aRJVfsIg/edit?usp=sharing`

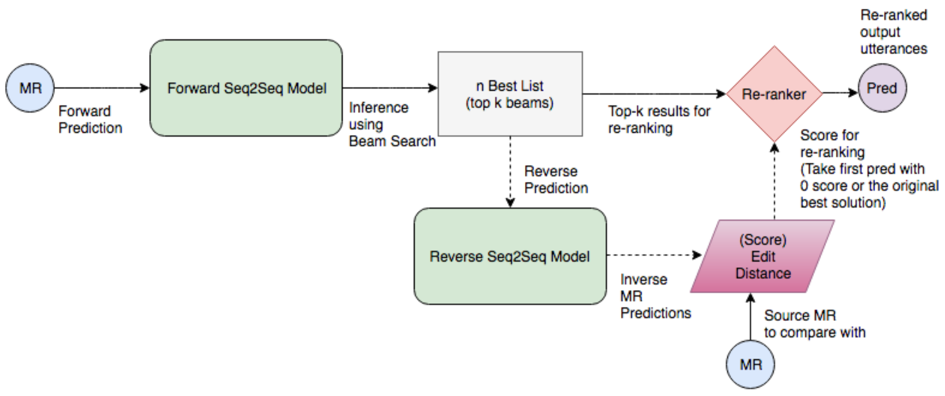
MR
Forward Prediction
Forward Seq2Seq Model
Inference using Beam Search
n Best List (top k beams)
Top-k results for re-ranking
Re-ranker
Re-ranked output utterances
Pred
Score for re-ranking (Take first pred with 0 score or the original best solution)
Reverse Prediction
Reverse Seq2Seq Model
Inverse MR Predictions
(Score) Edit Distance
Source MR to compare with
MR

Figure 4: Illustration of the pipeline for the re-ranking approach (based on inverse reconstructions using reverse model). Apart from Forward and Reverse seq2seq models, we have a re-ranker based on the edit distance of the actual MR and the inverse reconstructed MR.

Thus, the primary submission to the challenge was chosen as the pipeline model with classifier as re-ranker. Our second submission was based on re-ranking using the reverse model while the vanilla forward char2char model was our third submission.

4 Experiments

4.1 Dataset

The updated challenge dataset comprises 50K canonically ordered and systematically structured (MR,RF) pairs. This dataset was collected following the crowdsourcing protocol explained in Novikova et al. (2016). Consisting of 8 different slots (and their respective different values[3]), note that the statistics in the test set differ significantly from the training set (Table 1).

4.2 Implementation and technical details

We used the open source *tf-seq2seq* framework[4], built over TensorFlow (Abadi et al., 2016) and provided along with (Britz et al., 2017), with some standard configuration options.[5] However, to directly use this framework for our character based models, we had to replace non-ascii characters (such as £ and é) with ascii characters not present in our source and target vocabulary ($ and ˆ symbol respectively) as we faced file encoding related issues[6]. All other implementations were also done using Tensorflow and Python.

Agarwal and Dymetman (2017) provide a more detailed analysis of the selection of our parameters for the forward model. We experimented with different numbers of layers in the encoder and decoder as well as different beam widths, while using the bi-directional encoder with an “additive” attention mechanism. In terms of BLEU, our best performing model had the following configuration: encoder 1 layer, decoder 2 layers, GRU cell, beam-width 20, length penalty 1.

5 Evaluation

We followed the same evaluation strategy, as described in Agarwal and Dymetman (2017). We used BLEU as the automatic evaluation metric on the dev set. We defined ‘oracle prediction’ as a generated utterance which could be considered as perfect in both adequacy as well as linguistic quality. Table 2 summarizes our results on the dev set[7].

We chose our primary system to be the re-ranker using the classifier. Table 3 summarizes our ranking among all the 60+ submissions (primary as well as additional) on the test set.

Comparing just the primary systems, our primary model is placed at the 4th place (apart from the baseline); See Table 4 of (Dušek et al., 2018).

Results for human evaluation, as released by the

[3]More detailed statistics of the slot values can be found at `https://docs.google.com/spreadsheets/d/1lNSlwDawoQGeow1n8XD79D5JCW3U4Z8I1qId6rF\5SnI/edit?usp=sharing`

[4]`https://github.com/google/seq2seq.`

[5]Code for processing of the data, conversion to parallel text format as well as our configuration files for the tf-seq2seq model can be found at: `https://github.com/shubhamagarwal92/sigdialSubmission/`

[6]Issue #153 `https://github.com/google/seq2seq/issues/153.`

[7]Much higher scores were obtained with multi-ref BLEU on the updated devset, compared to Agarwal and Dymetman (2017)

Slot	Train Set		Dev Set		Test Set	
	Count	Percent	Count	Percent	Count	Percent
area	24716	59%	420	77%	558	89%
customer rating	28090	67%	481	88%	318	50%
eatType	20111	48%	465	85%	630	100%
familyFriendly	26295	63%	397	73%	572	91%
food	35126	84%	450	82%	546	87%
name	42061	100%	547	100%	630	100%
near	20546	49%	339	62%	618	98%
priceRange	29127	69%	346	63%	480	76%
Total MRs	42061		547		630	

Table 1: Frequency distribution of the slots present in MRs. Name slot is always present in the MR. As we go from train to test set, we find that the statistics of slots change significantly, which helps us better analyse the different observations on dev and test set.

Model	Oracle Predictions	BLEU
Re-ranking using classifier (Primary)	75	0.7052
Re-ranking using reverse (Secondary)	63	0.714
Forward (Third)	54	0.730

Table 2: Oracle (semantically adequate and linguistically correct) Predictions vs BLEU score on a random sample of 100 of the dev set for all our models. We found an inverse trend of the BLEU score compared against the observations by human judges.

Submission	BLEU	Overall Rank
Re-ranking using classifier (Primary)	0.653	18
Re-ranking using reverse (Secondary)	0.666	5
Forward (Third)	0.667	4
Baseline	0.659	10

Table 3: Automatic BLEU evaluations released by organizers on the final challenge submission. We had 3 submissions as described in Section 3. Two of our systems were in the top 5 among all 60+ submissions.

Primary Systems	BLEU
Participant-13	0.6619
Baseline	0.659
Participant-15	0.6561
Participant-1	0.6545
Ours	0.6534

Table 4: Comparing just the top 5 primary systems, we are at 4th position (not counting the baseline)

Metric	TrueSkill	Range	Cluster
Quality	0.048	(8-12)	2
Naturalness	0.105	(4-8)	2

Table 5: Human evaluation was crowd-sourced on the primary system according to the TrueSkill algorithm (Sakaguchi et al., 2014)

challenge organizers, are summarized in Table 5 of (Dušek et al., 2018). They followed the TrueSkill algorithm (Sakaguchi et al., 2014) judging all the primary systems on *Quality* and *Naturalness*. We obtained balance results in terms of both metrics, our system being in the 2nd cluster out of 5 (for both evaluations). On the other hand, most systems ranked high on quality tend to have lower ranks for naturalness and vice versa.

6 Analysis

We found that the presence of an oracle prediction was dependent on the number of slots in the MR. When the number of slots was 7 or 8, the presence of an oracle in the top-20 predictions decreased significantly, as opposed to when the number of slots was less than 7.

However, the most prominent issue was that of omissions, among the utterances produced in first position. There were no additions or non-words. We observed a similar issue of omissions in human references (target for our model) as well. It may be conjectured that the seq2seq model, by “averaging” over many linguistically diverse and sometimes incorrect training examples, was still able to learn what amounts to a reasonable linguistic model for its predictions.

Our two different strategies, thus, improved the semantic adequacy by re-ranking the probable candidates and successfully finding the ‘oracle’ prediction in the top-20 list. However, in terms of automatic evaluation, the BLEU score showed an inverse relationship with adequacy. Nevertheless, we chose our primary system to be the re-ranker with a classifier over the forward model.

We did not find any issues while "copying" the restaurant 'name' or 'near' slots on the dev set. However, on the test set, as the statistics of the data changed in terms of both slots, we found a tendency of the model to generate the more frequent slot values (corresponding to restaurant and near slots) in the training dataset, instead of copying the actual slot value.[8] We also found the same effect when we introduced new slot values (restaurant names) which were completely unseen in the training set. Thus, we feel that our char2char model may show degraded performance when the statistics of the training and test sets diverge significantly.

7 Conclusion

We show how a char2char model can be employed for the task of NLG and show competitive results in this challenge. Our vanilla character based model required minimal effort in terms of any processing of dataset while also producing great diversity in the generated utterances.

Building on top of this previous work (Agarwal and Dymetman, 2017), we propose two simple re-ranking strategies for further improvements. Even though re-ranking methods improve in terms of semantic adequacy, we found a reversal of trend in terms of BLEU. Our synthetic data creation technique could be adapted for augmenting NLG datasets and the classifier-based score could also be used as a reward in a Reinforcement Learning paradigm.

Acknowledgements

We thank Chunyang Xiao, Matthias Gallé and Ioannis Konstas for useful suggestions. We also take this opportunity to thank the E2E challenge organizers for their helpful reviews and comments.

[8]On a superficial evaluation on test set, we found the issues were mainly with the restaurant names 'Cocum', 'Strada' and 'The Phoenix'. These were wrongly interpreted by the model as 'Cotto', 'Zizzi' and 'The Punter' respectively.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *CoRR abs/1603.04467* .

Shubham Agarwal and Marc Dymetman. 2017. A surprisingly effective out-of-the-box char2char model on the e2e nlg challenge dataset. In *Proc. SIGdial*. ACL, Saarbrcken, Germany, pages 158–163.

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

Denny Britz, Anna Goldie, Thang Luong, and Quoc Le. 2017. Massive exploration of neural machine translation architectures. *CoRR abs/1703.03906* .

Andrew Chisholm, Will Radford, and Ben Hachey. 2017. Learning to generate one-sentence biographies from wikidata. *CoRR abs/1702.06235* .

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proc. EMNLP*.

Junyoung Chung, Kyunghyun Cho, and Yoshua Bengio. 2016. A character-level decoder without explicit segmentation for neural machine translation. *CoRR abs/1603.06147* .

Ondřej Dušek and Filip Jurčíček. 2016. Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings. *CoRR abs/1606.05491* .

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2017. Referenceless quality estimation for natural language generation. *arXiv preprint arXiv:1708.01759* .

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

Markus Freitag and Yaser Al-Onaizan. 2017. Beam search strategies for neural machine translation. *CoRR abs/1702.01806* .

Raghav Goyal, Marc Dymetman, and Eric Gaussier. 2016. Natural Language Generation through Character-based RNNs with Finite-State Prior Knowledge. In *Proc. COLING*. Osaka, Japan.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proc. ACL*.

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

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural amr: Sequence-to-sequence models for parsing and generation. *CoRR abs/1704.08381* .

Gerasimos Lampouras and Andreas Vlachos. 2016. Imitation learning for language generation from unaligned data. In *Proc. COLING*. pages 1101–1112.

Wang Ling, Isabel Trancoso, Chris Dyer, and Alan W Black. 2016. Character-based neural machine translation. In *Proc. ICLR*. pages 1–11.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proc. EMNLP*.

Hongyuan Mei, Mohit Bansal, and Matthew R Walter. 2015. What to talk about and how? selective generation using lstms with coarse-to-fine alignment. *CoRR abs/1509.00838* .

Neha Nayak, Dilek Hakkani-Tur, Marilyn Walker, and Larry Heck. 2017. To plan or not to plan? sequence to sequence generation for language generation in dialogue systems.

Jekaterina Novikova, Ondrej Dušek, and Verena Rieser. 2017. The E2E dataset: New challenges for end-to-end generation. In *Proc. SIGdial, ACL*. Saarbrücken, Germany.

Jekaterina Novikova, Oliver Lemon, and Verena Rieser. 2016. Crowd-sourcing NLG Data: Pictures Elicit Better Data. *CoRR abs/1608.00339* .

Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme. 2014. Efficient elicitation of annotations for human evaluation of machine translation. In *Proc. Statistical Machine Translation*. ACL, Baltimore, Maryland, USA, pages 1–11.

Shikhar Sharma, Jing He, Kaheer Suleman, Hannes Schulz, and Philip Bachman. 2016. Natural language generation in dialogue using lexicalized and delexicalized data. *arXiv preprint arXiv:1606.03632* .

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Proc. NIPS*. pages 3104–3112.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *CoRR abs/1506.05869* .

Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In *Proc. EMNLP*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klinger, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR abs/1609.08144* .

Chunyang Xiao, Marc Dymetman, and Claire Gardent. 2016. Sequence-based structured prediction for semantic parsing. *Proc. ACL* .

Shenjian Zhao and Zhihua Zhang. 2016. An efficient character-level neural machine translation. *CoRR abs/1608.04738* .

Appendix: Sample predictions

Slots	Type	Utterance
3	MR	name[Blue Spice], eatType[coffee shop], area[city centre]
	Pred	Blue Spice is a coffee shop located in the city centre.
4	MR	name[Blue Spice], eatType[coffee shop], customer rating[5 out of 5], near[Crowne Plaza Hotel]
	Pred	Blue Spice is a coffee shop near Crowne Plaza Hotel with a customer rating of 5 out of 5.
5	MR	name[The Cricketers], eatType[coffee shop], customer rating[1 out of 5], familyFriendly[yes], near[Avalon]
	Pred	The Cricketers is a children friendly coffee shop near Avalon with a customer rating of 1 out of 5.
6	MR	name[Blue Spice], eatType[pub], food[Chinese], area[city centre], familyFriendly[no], near[Rainbow Vegetarian Café]
	Pred	Blue Spice is a Chinese pub located in the city centre near Rainbow Vegetarian Café. It is not family friendly.
7	MR	name[The Mill], eatType[pub], food[English], priceRange[high], area[riverside], familyFriendly[yes], near[Raja Indian Cuisine]
	Pred	The Mill is a children friendly English pub with a high price range near Raja Indian Cuisine in riverside.
8	MR	name[The Cricketers], eatType[restaurant], food[Chinese], priceRange[£20-25], customer rating[high], area[city centre], familyFriendly[no], near[All Bar One]
	Pred	The Cricketers is a restaurant providing Chinese food in the £20-25 price range. It is located in the city centre near All Bar One. It has a high customer rating and is not kid friendly.

Table 6: Sample predictions. For the first MR of each arity (3 to 8) in the testset, we show the prediction of our primary submission.