Appendix for Challenges in Data-to-Document Generation

A. Additional Dataset Details

The ROTOWIRE data covers NBA games played between 1/1/2014 and 3/29/2017; some games have multiple summaries. The summaries have been randomly split into training, validation, and test sets consisting of 3398, 727, and 728 summaries, respectively.

The SBNATION data covers NBA games played between 11/3/2006 and 3/26/2017; some games have multiple summaries. The summaries have been randomly split into training, validation, and test sets consisting of 7633, 1635, and 1635 summaries, respectively.

All numbers in the box- and line-scores (but not the summaries) are converted to integers; fractional numbers corresponding to percents are multiplied by 100 to obtain integers in $[0, 100]$. We show the *types* of records in the data in Table 1.

B. Generation Model Details

Encoder For the ROTOWIRE data, a relation r is encoded into $\tilde{\boldsymbol{r}}$ by embedding each of $r.e$, $r.t$, $r.m$ and a "home-or-away" indicator feature in $\mathbb{R}^{600}$, and applying a 1-layer MLP (with ReLU nonlinearity) to map the concatenation of these vectors back into $\mathbb{R}^{600}$. To initialize the decoder LSTMs, we first mean-pool over the $\tilde{\boldsymbol{r}}_j$ by entity (giving one vector per entity), and then linearly transform the concatenation of these pooled entity-representations so that they can initialize the cells and hidden states of a 2-layer LSTM with states also in $\mathbb{R}^{600}$. The SBNATION setup is identical, except all vectors are in $\mathbb{R}^{700}$.

Decoder As mentioned in the body of the paper, we compute two different attention distributions (i.e., using different parameters) at each decoding step. For the Joint Copy model, one attention distribution is not normalized, and is normalized along with all the output-word probabilities.

Within the Conditional Copy model we compute $p(z_t | \hat{y}_{1:t-1}, \boldsymbol{s})$ by mean-pooling the $\tilde{\boldsymbol{r}}_j$, concatenating them with the current (topmost) hidden state of the LSTM, and then feeding this concatenation via a 1-layer ReLU MLP with hidden dimension 600, and with a Sigmoid output layer.

For the reconstruction-loss, we feed blocks (of size at most 100) of the decoder's LSTM hidden states through a (Kim, 2014)-style convolutional model. We use kernels of width 3 and 5, 200 filters, a ReLU nonlinearity, and max-over-time pooling. To create the p_k, these now 400-dimensional features are then mapped via an MLP with a ReLU nonlinearity into 3 separate 200 dimensional vectors corresponding to the predicted relation's entity, value, and type, respectively. These 200 dimensional vectors are then fed through (separate) linear decoders and softmax layers in order to obtain distributions over entities, values, and types. We use $K = 3$ distinct p_k.

Models are trained with SGD, a learning rate of 1 (which is divided by 2 every time validation perplexity fails to decrease), and a batch size of 16. We use dropout (at a rate of 0.5) between LSTM layers and before the linear decoder.

C. Information Extraction Details

Data To form an information extraction dataset, we first sentence-tokenize the gold summary documents $y_{1:T}$ using NLTK (Bird, 2006). We then determine which word-spans $y_{i:j}$ could represent entities (by matching against players, teams, or cities in the database), and which word-spans $y_{k:l}$ could represent numbers (using the open source `text2num` library[1] to convert (strings of) number-words into numbers).[2] We then con-

[1] https://github.com/exogen/text2num

[2] We ignore certain particularly misleading number-words, such as "three-point," where we should not expect a

Player Types	POSN	MIN	PTS	FGM	FGA	FG-PCT	FG3M	FG3A	FG3-PCT
	FTM	FTA	FT-PCT	OREB	DREB	REB	AST	TOV	STL
	BLK	PF	FULL-NAME	NAME1	NAME2	CITY			
Team Types	PTS-QTR1	PTS-QTR2	PTS-QTR3	PTS-QTR4	PTS	FG-PCT	FG3-PCT	FT-PCT	REB
	AST	TOV	WINS	LOSSES	CITY	NAME			

Table 1: Possible Record Types

sider each $y_{i:j}, y_{k:l}$ pair in the same sentence, and if there is a record r in the database such that $r.e = y_{i:j}$ and $r.m = \text{text2num}(y_{k:l})$ we annotate the $y_{i:j}, y_{k:l}$ pair with the label $r.t$; otherwise, we give it a label of ϵ.

Model We predict relations by ensembling 3 convolutional models and 3 bidirectional LSTM (Hochreiter and Schmidhuber, 1997; Graves and Schmidhuber, 2005) models. Each model consumes the words in the sentence, which are embedded in $\mathbb{R}^{200}$, as well as the distances of each word in the sentence from both the entity-word-span and the number-word-spans (as described above), which are each embedded in $\mathbb{R}^{100}$. These vectors are concatenated (into a vector in $\mathbb{R}^{500}$) and fed into either a convolutional model or a bidirectional LSTM model.

The convolutional model uses 600 total filters, with 200 filters for kernels of width 2, 3, and 5, respectively, a ReLU nonlinearity, and max-pooling. These features are then mapped via a 1-layer (ReLU) MLP into $\mathbb{R}^{500}$, which predicts one of the 39 relation types (or ϵ) with a linear decoder layer and softmax.

The bidirectional LSTM model uses a single layer with 500 units in each direction, which are concatenated. The hidden states are max-pooled, and then mapped via a 1-layer (ReLU) MLP into $\mathbb{R}^{700}$, which predicts one of the 39 relation types (or ϵ) with a linear decoder layer and softmax.

References

Steven Bird. 2006. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, pages 69–72. Association for Computational Linguistics.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602–610.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9:1735–1780.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*, pages 1746–1751.

corresponding value of 3 among the records.