Two Birds, One Stone: A Simple, Unified Model for Text Generation from Structured and Unstructured Data

Hamidreza Shahidi,1,2 Ming Li,1,2 and Jimmy Lin1,2

1 David R. Cheriton School of Computer Science, University of Waterloo
2 RSVP.ai

{h24shahi, mli, jimmylin}@uwaterloo.ca

Abstract

A number of researchers have recently questioned the necessity of increasingly complex neural network (NN) architectures. In particular, several recent papers have shown that simpler, properly tuned models are at least competitive across several NLP tasks. In this work, we show that this is also the case for text generation from structured and unstructured data. We consider neural table-to-text generation and neural question generation (NQG) tasks for text generation from structured and unstructured data, respectively. Table-to-text generation aims to generate a description based on a given table, and NQG is the task of generating a question from a given passage where the generated question can be answered by a certain sub-span of the passage using NN models. Experimental results demonstrate that a basic attention-based seq2seq model trained with the exponential moving average technique achieves the state of the art in both tasks. Code is available at https://github.com/h-shahidi/2birds-gen.

1 Introduction

Recent NLP literature can be characterized as increasingly complex neural network architectures that eke out progressively smaller gains over previous models. Following a previous line of research (Melis et al., 2018; Mohammed et al., 2018; Adhikari et al., 2019), we investigate the necessity of such complicated neural architectures. In this work, our focus is on text generation from structured and unstructured data by considering description generation from a table and question generation from a passage and a target answer.

More specifically, the goal of the neural table-to-text generation task is to generate biographies based on Wikipedia infoboxes (structured data). An infobox is a factual table with a number of fields (e.g., name, nationality, and occupation) describing a person. For this task, we use the WikiBio dataset (Lebret et al., 2016) as the benchmark dataset. Figure 1 shows an example of a biographic infobox as well as the target output textual description.

Automatic question generation aims to generate a syntactically correct, semantically meaningful and relevant question from a natural language text and a target answer within it (unstructured data). This is a crucial yet challenging task in NLP that has received growing attention due to its application in improving question answering systems (Duan et al., 2017; Tang et al., 2017, 2018), providing material for educational purposes (Heilman and

Target Output:

Sir Bernard Augustus Keen FRS (5 September 1890 – 5 August 1981) was a British soil scientist and Fellow of University College London.

Figure 1: An example infobox from the WikiBio dataset and the corresponding target output description.

Table 1: A sample (passage, answer, question) triple from the SQuAD dataset.
Smith, 2010), and helping conversational systems to start and continue a conversation (Mostafazadeh et al., 2016). We adopt the widely used SQuAD dataset (Rajpurkar et al., 2016) for this task. Table 1 presents a sample (passage, answer, question) triple from this dataset.

Prior work has made remarkable progress on both of these tasks. However, the proposed models utilize complex neural architectures to capture necessary information from the input(s). In this paper, we question the need for such sophisticated NN models for text generation from inputs comprising structured and unstructured data.

Specifically, we adopt a bi-directional, attention-based seq2seq model (Bahdanau et al., 2015) equipped with a copy mechanism (Gu et al., 2016) for both tasks. We demonstrate that this model, together with the exponential moving average (EMA) technique, achieves the state of the art in both neural table-to-text generation and NQG. Interestingly, our model is able to achieve this result even without using any linguistic features.

Our contributions are two-fold: First, we propose a unified NN model for text generation from structured and unstructured data and show that training this model with the EMA technique leads to the state of the art in neural table-to-text generation as well as NQG. Second, because our model is, in essence, the primary building block of previous models, our results show that some previous papers propose needless complexity, and that gains from these previous complex neural architectures are quite modest. In other words, the state of the art is achieved by careful tuning of simple and well-engineered models, not necessarily by adding more complexity to the model, echoing the sentiments of Lipton and Steinhardt (2018).

2 Related Work

In this section, we first discuss previous work for neural table-to-text generation and then NQG.

2.1 Neural Table-to-Text Generation

Recently, there have been a number of end-to-end trainable NN models for table-to-text generation. Lebret et al. (2016) propose an n-gram statistical language model that incorporates field and position embeddings to represent the structure of a table. However, their model is not effective enough to capture long-range contextual dependencies while generating a description for the table.

To address this issue, Liu et al. (2018) suggest a structure-aware seq2seq model with local and global addressing on the table. While local addressing is realized by content encoding of the model’s encoder and word-level attention, global addressing is accomplished by field encoding using a field-gating LSTM and field-level attention. The field-gating mechanism incorporates field information when updating the cell memory of the LSTM units.

Liu et al. (2019b) utilize a two-level hierarchical encoder with coarse-to-fine attention to model the field-value structure of a table. They also propose three joint tasks (sequence labeling, text auto-encoding, and multi-label classification) as auxiliary supervision to capture accurate semantic representations of the tables.

In this paper, similar to Lebret et al. (2016), we use both content and field information to represent a table by concatenating the field and position embeddings with the word embedding. Unlike Liu et al. (2018), we don’t separate local and global addressing by using specific modules for each, but rather adopt the EMA technique and let the bi-directional model accomplish this implicitly, exploiting the natural advantages of the model.

2.2 Neural Question Generation

Previous NQG models can be classified into rule-based and neural-network-based approaches. Du et al. (2017) propose a seq2seq model that is able to achieve better results than previous rule-based systems without taking the target answer into consideration. Zhou et al. (2017) concatenate answer position indicators with the word embeddings to make the model aware of the target answer. They also use lexical features (e.g., POS and NER tags) to enrich their model’s encoder. In addition, Song et al. (2018) suggest using a multi-perspective context matching algorithm to further leverage information from explicit interactions between the passage and the target answer.

More recently, Kim et al. (2019) use answer-separated seq2seq, which replaces the target answer in the passage with a unique token to avoid using the answer words in the generated question. They also make use of a module called keyword-net to extract critical information from the target answer. Similarly, Liu et al. (2019a) propose using a clue word predictor by adopting graph convolution networks to highlight the imperative aspects of the input passage.
Our model is architecturally more similar to Zhou et al. (2017), but with the following distinctions: (1) we do not use additional lexical features, (2) we utilize the EMA technique during training and use the averaged weights for evaluation, (3) we do not make use of the introduced maxout hidden layer, and (4) we adopt LSTM units instead of GRU units. These distinctions, along with some hyperparameter differences, notably the optimizer and learning rate, have a considerable impact on the experimental results (see Section 5).

3 Model: Seq2Seq with Attention and a Copy Mechanism

In this section, we introduce a simple but effective attention-based seq2seq model for both neural table-to-text generation and NQG. Figure 2 provides an overview of our model.

3.1 Encoder

Our encoder is a bi-directional LSTM (BiLSTM) whose input $x_t$ at time step $t$ is the concatenation of the current word embedding $e_t$ with some additional task-specific features.

For neural table-to-text generation, additional features are field name $f_t$ and position information $p_t$, following Lebret et al. (2016). The position information itself is the concatenation of $p_t^+$, which is the position of the current word in its field when counting from the left, and $p_t^-$, when counting from the right. Considering the word University, in Figure 1, as an example, it is the first word from the left and the third word from the right in the Institutions field. Hence, the structural information of this word would be $\{\text{Institutions}, 1, 3\}$. Thus, the input to the encoder at time step $t$ for this task is $x_t = [e_t; f_t; p_t^+; p_t^-]$, where $[; ;]$ denotes concatenation along the feature dimension.

For NQG, similar to Zhou et al. (2017), we use a single bit $b_t$, indicating whether the $t$th word in the passage belongs to the target answer, as an additional feature. Hence, the input at time step $t$ is $x_t = [e_t; b_t]$. Remarkably, unlike previous work (Song et al., 2018; Kim et al., 2019), we do not use a separate encoder for the target answer to have a unified model for both tasks.

3.2 Attention-Based Decoder

Our decoder is an attention-based LSTM model (Bahdanau et al., 2015). Due to the considerable overlap between input and output words, we use a copy mechanism (Gu et al., 2016) that integrates the attention distribution over the input words with the vocabulary distribution.

3.3 Exponential Moving Average

The exponential moving average (EMA) technique, also referred to as temporal averaging, was initially introduced to be used in optimization algorithms for better generalization performance and reducing noise from stochastic approximation in recent parameter estimates by averaging model parameters (Polyak and Juditsky, 1992; Moulines and Bach, 2011; Kingma and Ba, 2015).

In applying the technique, we maintain two sets of parameters: (1) training parameters $\theta$ that are trained as usual, and (2) evaluation parameters $\tilde{\theta}$ that are an exponentially weighted moving average of the training parameters. The moving average is calculated using the following expression:

$$\tilde{\theta} \leftarrow \beta \times \tilde{\theta} + (1 - \beta) \times \theta \quad (1)$$

where $\beta$ is the decay rate. Previous work (Szegedy et al., 2016; Merity et al., 2018; Adhikari et al., 2019; Liu et al., 2019a) has used this technique for different tasks to produce more stable and accurate results. In Section 5, we show that using this simple technique considerably improves the performance of our model in both of the tasks.

4 Experimental Setup

In this section, we introduce the datasets first, then explain additional implementation details, and finally describe the evaluation metrics.

4.1 Datasets

We use the WikiBio dataset (Lebret et al., 2016) for neural table-to-text generation. This dataset
contains 728,321 articles from English Wikipedia and uses the first sentence of each article as the ground-truth description of the corresponding infobox. The dataset has been divided into training (80%), validation (10%), and test (10%) sets.

For NQG, we use the SQuAD dataset v1.1 (Rajpurkar et al., 2016) in our experiments, containing 536 Wikipedia articles with over 100K question-answer pairs. The test set of the original dataset is not publicly available. Thus, Du et al. (2017) and Zhou et al. (2017) re-divide available data into training, validation, and test sets, which we call split-1 and split-2, respectively. In this paper, we conduct experiments and evaluate our model on both of the data splits.

4.2 Implementation Details

For the sake of reproducibility, we discuss implementation details for achieving the results shown in Tables 2 and 3. We train the model using cross-entropy loss and retain the model that works best on the validation set during training for both tasks. We replace unknown tokens with a word from the input having the highest attention score. In addition, a decay rate of 0.9999 is used for the exponential moving average in both of the tasks.

For the neural table-to-text generation task, we train the model up to 10 epochs with three different seeds and a batch size of 32. We use a single-layer BiLSTM for the encoder and a single-layer LSTM for the decoder and set the dimension of the LSTM hidden states to 500. Optimization is performed using the Adam optimizer with a learning rate of 0.0005 and gradient clipping when its norm exceeds 5. The word, field, and position embeddings are trainable and have a dimension of 400, 50, and 5, respectively. The maximum position number is set to 30. Any higher position number is therefore counted as 30. The most frequent 20,000 words and 1,480 fields in the training set are selected as word vocabulary and field vocabulary, respectively, for both the encoder and the decoder. Ultimately, we conduct greedy search to decode a description for a given input table.

For the NQG task, we use a two-layer BiLSTM for the encoder and a single-layer LSTM for the decoder. We set the dimension of the LSTM hidden states to 350 and 512 for split-1 and split-2, respectively. Optimization is performed using the AdaGrad optimizer with a learning rate of 0.3 and gradient clipping when its norm exceeds 5. The word embeddings are initialized with pre-trained 300-dimensional GloVe embeddings (Pennington et al., 2014), which are frozen during training. We train the model up to 20 epochs with five different seeds and a batch size of 50. We further employ dropout with a probability of 0.1 and 0.3 for data split-1 and split-2, respectively. Moreover, we use the vocabulary set released by Song et al. (2018) for both the encoder and the decoder. During decoding, we perform beam search with a beam size of 20 and a length penalty weight of 1.75.

4.3 Evaluation

Following previous work, we use BLEU-4 (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), ROUGE-4, and ROUGE-L (Lin, 2004) to evaluate the performance of our model. BLEU and METEOR were originally designed to evaluate machine translation systems, and ROUGE was designed to evaluate text summarization systems.

5 Results and Discussion

In this section, we present our experimental results for both neural table-to-text generation and NQG. We report the mean and standard deviation of each metric across multiple seeds to ensure robustness against potentially spurious conclusions (Crane, 2018). In Tables 2 and 3, we compare previous work with our results for NQG and neural table-to-text generation, respectively. All results are copied from the original papers except for Liu et al. (2018) in Table 3, where Repl. refers to scores from experiments that we conducted using the source code released by the authors, and Orig. refers to scores taken from the original paper.

It is noteworthy that a similar version of our model has served as a baseline in previous papers (Liu et al., 2018; Kim et al., 2019; Liu et al., 2019a). However, the distinctions discussed in Section 2, especially the EMA technique, enable our model to achieve the state of the art in all cases but BLEU-4 on the SQuAD split-2, where our score is very competitive; furthermore, Liu et al. (2019a) only report results from a single trial. Our results indicate that a basic seq2seq model is able to effectively learn the underlying distribution of both datasets.

6 Conclusions and Future Work

In this paper, we question the necessity of complex neural architectures for text generation from structured data (neural table-to-text generation) and
unstructured data (NQG). We then propose a simple yet effective seq2seq model trained with the EMA technique. Empirically, our model achieves the state of the art in both of the tasks. Our results highlight the importance of thoroughly exploring simple models before introducing complex neural architectures, so that we can properly attribute the source of performance gains. As a potential direction for future work, it would be interesting to investigate the use of the EMA technique on transformer models as well and conduct similar studies to examine needless architectural complexity in other NLP tasks.

Acknowledgments

This research was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada.

References


