Table-to-Text Generation via Row-Aware Hierarchical Encoder

Heng Gong, Xiaocheng Feng, Bing Qin⋆, and Ting Liu

Research Center for Social Computing and Information Retrieval,
Harbin Institute of Technology, Harbin, China
{hgong, xcfeng, qinb, tliu}@ir.hit.edu.cn

Abstract. In this paper, we present a neural model to map structured table into document-scale descriptive texts. Most existing neural network based approaches encode a table record-by-record and generate long summaries by attentional encoder-decoder model, which leads to two problems. (1) portions of the generated texts are incoherent due to the mismatch between the row and corresponding records. (2) a lot of irrelevant information is described in the generated texts due to the incorrect selection of the redundant records. Our approach addresses both problems by modeling the row representation as an intermediate structure of the table. In the encoding phase, we first learn record-level representation via transformer encoder. Afterwards, we obtain each row's representation according to their corresponding records' representation and model row-level dependency via another transformer encoder. In the decoding phase, we first attend to row-level representation to find important rows. Then, we attend to specific records to generate texts. Experiments were conducted on ROTOWIRE, a dataset which aims at producing a document-scale NBA game summary given structured table of game statistics. Our approach improves a strong baseline's BLEU score from 14.19 to 15.65 (+10.29%). Furthermore, three extractive evaluation metrics and human evaluation also show that our model has the ability to select salient records and the generated game summary is more accurate.

Keywords: Table-to-Text Generation · Seq2Seq · Hierarchical Encoder

1 Introduction

We focus on table-to-text generation that maps structured statistical data to document-level natural language texts [19, 8, 22]. In this task, STAT[1] is multi-row multi-column table that consists of multiple records, which would be transformed to long descriptive texts in encoding-decoding process. Datasets are the main driver of progress for statistical approaches in table-to-text [10]. In recent years, Wiseman et al. [22] release ROTOWIRE, a more complex table-to-text

⋆ Corresponding Author

[1] We abbreviate the statistics table as STAT.

Team	PTS	PTS_ QTR1	PTS_ QTR2	WIN	LOSS	…
Raptors	104	28	28	21	8	…
Jazz	98	29	22	18	13	…

Player	PTS	AST	REB	STL	…
Raptors					
DeMar DeRozan	24	1	6	4	…
Kyle Lowry	36	5	4	2	…
Jonas Valanciunas	14	0	7	1	…
Terrence Ross	10	1	1	0	…
Cory Joseph	4	3	1	1	…
Fred VanVleet	N/A	N/A	N/A	N/A	…
…	…	…	…	…	…
Jazz					
Rudy Gobert	13	1	14	1	…
Gordon Hayward	23	5	1	1	…
Trey Lyles	19	1	7	0	…
Shelvin Mack	17	5	4	1	…
Rodney Hood	2	1	0	0	…
Joe Johnson	7	0	0	0	…
Raul Neto	0	0	1	0	…
George Hill	N/A	N/A	N/A	N/A	…
…	…	…	…	…	…

STAT

The Toronto Raptors defeated the Utah Jazz , 104 - 98 … The Raptors were without **DeMar DeRozan** (ankle) , who missed the previous two games with an ankle injury . **Kyle Lowry** led the way for the Raptors with a game - high 36 points , which he supplemented with five assists , four rebounds and two steals . **DeMar DeRozan** followed with 24 points , six rebounds , an assist and four steals . **Jonas Valanciunas** posted a 14 - point , 14 - rebound double - double that also included a block , a steal and a block . Cory Joseph led the bench with 20 points , three rebounds , three assists and a steal . Fred VanVleet paced the reserves with 17 points , six rebounds , an assist and a steal . The Raptors were led by a pair of 24 - point efforts from Rodney Hood and Joe Johnson , with the former adding seven rebounds , an assist and a steal , and the latter supplying seven rebounds , an assist and a steal . George Hill was right behind him with 36 points , five assists , four rebounds and a steal . **Rudy Gobert** was next with a 13 - point , 14 - rebound double - double that also included an assist , a steal and a block . George Hill was next with 17 points , five assists , four rebounds , two steals and a block . Rudy Gobert posted a 13 - point , 14 - rebound double - double that also included an assist , a steal and a block . Raul Neto was productive in a reserve role as well with 17 points , five rebounds , an assist and a steal …

Baseline results

Fig. 1. Generated example on ROTOWIRE by using a strong baseline (conditional copy [22]). Red words mean they are in line with statistics in the table. Blue words mean they contradict table's information or they are not the salient players to be mentioned. Underscored player names indicate that the model correctly select those salient players in line with reference.

generation dataset, which aims at producing a document-scale game news based on plenty of NBA game records. Figure 1 shows an example of parts of a game's statistics and its corresponding computer generated summary. In STAT, each row is a player or team's name and corresponding records, each column is the player or team's attributes, such as points, steals, rebounds, etc. This dataset has an order of magnitude longer target text length with significantly more records than conventional datasets, such as WIKIBIO [8] or WEATHERGOV [11].

In table-to-text task, a lot of neural models [15, 13, 20, 2] are developed, which encode a table record-by-record and generate a long descriptive summary by a decoder with attention mechanism [1, 14]. However, we claim that these two general designs are problematic toward complex statistics. (1) portions of generated texts are incorrect due to the mismatch between the row and the corresponding records. As shown in Figure 1, player *Fred VanVleet* and *George Hill* didn't play in this game, but baseline model wrongly believe they did and use others' scores to describe them. Also, player *DeMar DeRozan* did play in this game, but baseline model thinks he was absent due to injury at the beginning of the texts. Baseline model also use others' information to describe *Jonas Valanciunas*, *Cory Joseph*, etc. (2) a large portion of records in a table are redundant and attention mechanism is insufficient to properly select salient information. Based on extractive evaluation results on reference there are 628 records per game in ROTOWIRE corpus, but only an average of 24.14 records are mentioned in the summary. As shown in Figure 1 and Figure 3, base model fails to include players with impressive performance such as *Gordon Hayward* and *Trey Lyles* in the generated texts while include players like *Cory Joseph*, *Fred VanVleet*, *Rodney Hood*, *Joe Johnson*, *George Hill* and *Raul Neto* who don't appear in the reference.

To address the aforementioned problems, we present a Seq2Seq model with hierarchical encoder that considers the rows of the table as an intermediate representation structure. The approach is partly inspired by the success of structure-aware neural network approaches in summarization [4]. Since records in the table are not sequential, we use transformer encoder [21] to learn record-level representation of the table. Then, we obtain each row's representation considering the corresponding records' representation, modeling dependencies between rows via row-level transformer encoder. The advantage is that our encoding strategy can model the dependencies between players. We conduct experiments on ROTOWIRE. Results show that our approach outperforms existing systems, improving the strong base model's BLEU to 15.65 (+1.46), RG Precision to 93.26 (+18.46) , CS F1% to 38.46 (+5.97) and CO to 18.24 (+2.82).

2 Background

2.1 Task Definition

We model the document-scale data-to-text generation task in an end-to-end fashion. Statistics STAT consists of multiple records $\{r_{1,1}$,..., $r_{i,j}$..., $r_{R,C}\}$ where R is the number of rows and C is the number of columns. Each record is in form of tuple ($row_{i,j}$, $column_{i,j}$, $cell_{i,j}$, $feat_{i,j}$) where $row_{i,j}$ refers to the row's name like player's name or team's name, $column_{i,j}$ refers to the tuple's type like points, $cell_{i,j}$ represents the value of this tuple and $feat_{i,j}$ indicates whether the player or team plays in their home court or not. Given STAT about one game, the model is expected to generate a document-scale summary of this game SUM=$\{w_1$, ..., w_j, ..., $w_N\}$ with w_j being the j^{th} word in the summary. N is the number of words in the summary.

2.2 Base Model

We use Seq2Seq model with attention mechanism [14] and conditional copy mechanism [6] as the base model because it achieves best performance across BLEU and extractive evaluation metrics according to Wiseman et al. [22]. First, given STAT about one game, the encoder utilizes 1-layer MLP to map the representation of each record $r_{i,j} = (emb(row_{i,j}) \circ emb(column_{i,j}) \circ emb(cell_{i,j}) \circ emb(feat_{i,j}))$ into a dense representation $h_{i,j}$ separately. Then, given the representation of STAT $\tilde{H} = \{h_{1,1}, ..., h_{i,j}, ..., h_{R,C}\}$, a LSTM decoder decomposes the probability of choosing each word at time step t into two parts as shown in Equation 1. The binary variable z_t indicates whether the word is copied from table or generated from vocabulary. $z_t = 1$ means the word is copied and $z_t = 0$ indicates the word is generated. In this paper, we use a 1-layer MLP with sigmoid function as its activation function to compute the probability $P(z_t = 1|y_{<t}, S)$. $P_{gen}(y_t|y_{<t}, \tilde{H})$ is computed by the LSTM decoder considering previous generated word and attentional representation of $\tilde{H}$. We reuse the attention weight $\alpha_{t,i',j'}$ as the probability of copying each word in the table. The model is trained to minimize the negative log-likelihood of words in reference in an end-to-end fashion.

$$P(y_t, z_t|y_{<t}, \tilde{H}) = \begin{cases} P(z_t = 1|y_{<t}, \tilde{H}) \sum_{y_t \leftarrow \boldsymbol{r_{i',j'}}} \alpha_{t,i',j'} & z_t = 1 \\ P(z_t = 0|y_{<t}, \tilde{H}) P_{gen}(y_t|y_{<t}, \tilde{H}) & z_t = 0 \end{cases} \tag{1}$$

3 Approach

In this section, we present a row-aware Seq2Seq model with hierarchical encoder. Given a STAT(table) as input, the model output corresponding descriptive texts. We divide the model into following parts. Figure 2 presents a detailed illustration of our model.

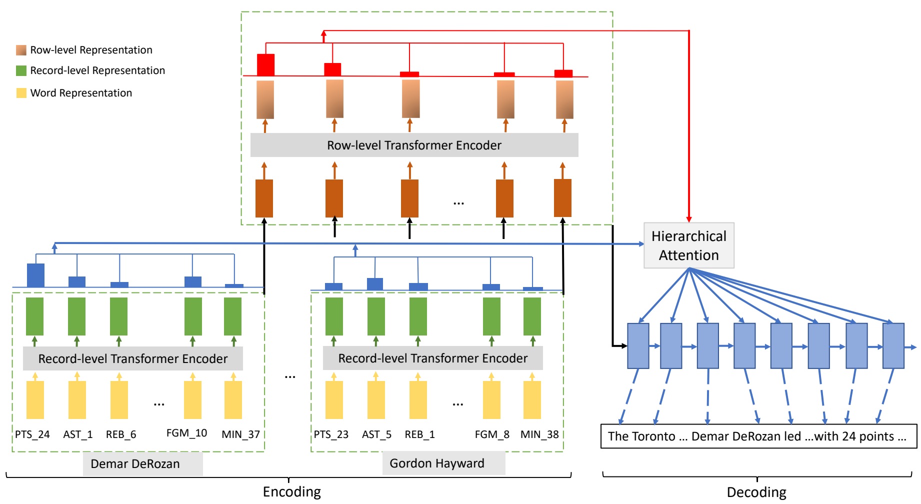

Fig. 2. The architecture of our proposed model. The left part is the hierarchical transformer encoder that encode table at both record-level and row-level. Right part is the LSTM decoder with hierarchical attention that generate words by first attending to rows then attending to each records.

3.1 Hierarchical Encoder

We explore two hierarchical encoder structures to encode game statistics on both record-level and row-level in this paper.

Firstly, we explore hierarchical bi-LSTM encoder structure, which can take other rows or records from the same row into consideration when modeling each row or record. On record level, we use a bi-LSTM to model records of the same row. For example, for row i, we take $r_{i,j}$, which is described in section 2.2, as input for each time step, then obtain records' representation of row i $\{\tilde{h}_{i,1}$,..., $\tilde{h}_{i,j}$,..., $\tilde{h}_{i,C}\}$ by bi-LSTM.

On row level, we concatenate $\overrightarrow{\tilde{h}_{i,C}}$ (C is the number of columns) and $\overleftarrow{\tilde{h}_{i,1}}$ to represent row_i's overall performance h_i. $\overrightarrow{\tilde{h}_{i,T}}$ is the last time step for forward encoder for row i. $\overleftarrow{\tilde{h}_{i,1}}$ is also the last time step for backward encoder for row i. Then we use another bi-LSTM to obtain row-level representation $\tilde{h}_i$.

Secondly, despite the wide use of sequential model to learn representation of table, since the data in the table are not sequential, using sequential model is inadequate. Therefore, we propose to use a hierarchical transformer encoder structure which can model the records representation in the context of other records in a non-sequential way. On record level, we use a transformer encoder [21] to learn the record's representation considering other records in the same row. First, we use 1-layer MLP to obtain record representation $h_{i,j}$ similar to the base model. Then, we use transformer encoder to take records in the same row into consideration. Transformer encoder contains multiple layers. As described in Equation 2 and 3, each layer consists of two components: multi-head self attention and feed-forward network. K represents the layer number of transformer encoder. We use $\tilde{h}_{i,j} = h_{i,j}^K$ from last layer as the record-level representation.

$$t_{i,j}^k = LayerNorm(h_{i,j}^k + MultiHeadSelfAttention(h_{i,j}^k)) \tag{2}$$

$$h_{i,j}^{k+1} = LayerNorm(t_{i,j}^k + FeedForwardNetwork(t_{i,j}^k)) \tag{3}$$

Then, we use mean pooling over $\{\tilde{h}_{i,1}, ..., \tilde{h}_{i,j}, ..., \tilde{h}_{i,T}\}$ to obtain i^{th} row representation h_i. T is the number of records in the i^{th} row. On row level, we use another transformer encoder to model dependency between rows $\{h_1, ..., h_N\}$. N is the number of rows. Then, we can obtain row-level representation $\tilde{h}_i$, which is similar to the record-level transformer encoder.

3.2 Decoder with Hierarchical Attention

Based on our observation, sentences in game summary intend to focus on few player or team. Also, with hierarchical encoder proposed above, we now have both row-level and record-level representation about the table. Therefore, we use hierarchical attention for decoder to first decide which row will be focused on, then attend to relevant records for generating words.

As shown in Equation 4, the decoder takes previous LSTM hidden state, previous generated word and context vector to update the LSTM hidden state. Then, the decoder will first attend to row-level representation in order to identify important row to be mentioned according to Equation 5. Afterwards, the decoder will attend to record-level representation to find relevant records as shown in Equation 6. Please note that the record-level attention weight is normalized among other records in the same row. Then, the decoder will use row-level attention weight as a guidance to re-weight record-level attention weight and obtain context vector c_t as shown in Equation 7. Please note that the re-weighted attention weight sum to 1 across all records in the table.

$$s_t = LSTM(\tilde{s}_{t-1}, c_{t-1}, y_{t-1}) \tag{4}$$

$$\alpha_{t,i} = \frac{exp(score(s_t, \tilde{h}_i))}{\sum_{i'} exp(score(s_t, \tilde{h}_{i'}))} \tag{5}$$

$$\beta_{t,i,j} = \frac{exp(score(s_t, \tilde{h}_{i,j}))}{\sum_{j'} exp(score(s_t, \tilde{h}_{i,j'}))} \tag{6}$$

$$c_t = \sum_i \sum_j \alpha_{t,i} \beta_{t,i,j} \tilde{h}_{i,j} \tag{7}$$

With the context vector c_t that contains relevant records' information, the decoder obtains an attentional hidden state $\tilde{s}_t$ by appling 1-layer MLP on the concatenation of context vector c_t and hidden state s_t as described in Equation 8. W_c is a trainable parameter and [;] denotes vector concatenation. Then we use Equation 9 to replace the base model's $p_{gen}(y_t|y_{<t}, \tilde{H})$ described in Section 2.2. Also, we use $\gamma_{t,i,j} = \alpha_{t,i}\beta_{t,i,j}$ as the probability of copying each word in the table.

$$\tilde{s}_t = tanh(W_c[c_t; s_t]) \tag{8}$$

$$p_{gen}(y_t|y_{<t}, \tilde{H}) = softmax(W_s \tilde{s}_t) \tag{9}$$

3.3 Training

During training, the model is optimized by minimizing the negative log-likelihood given the gold descriptive texts $y^\star$ as described in Equation 10.

$$L = -\sum_{t=1}^{T} logP(y_t^\star | y_{<t}, \tilde{H}) \tag{10}$$

4 Experiments

4.1 Dataset and Evaluation Metrics

We conducted experiments on ROTOWIRE dataset [22]. For each example, it provides 628 records relating to teams and players' statistics and its corresponding long game summary. The average length of game summary is 337.1 and the number of record types is 39. In the end, we use training, validation, test set with 3398, 727, 728 summaries respectively for comparing models' performance.

We follow Wiseman et al. [22] and use BLEU [17] and three extractive evaluation metrics [22] RG, CS and CO for automatic evaluation. The main idea is using an IE (Information Extraction) model to extract record tuples from reference and generated texts. Then, CS (Content Selection) measures model's ability on content selection by comparing records in generated texts with records in reference, CO (Content Ordering) measures ability on content placement by calculating normalized Damerau-Levenshtein Distance [3] between records from generated texts and from reference and RG (Relation Generation) compares summary's records with the table and measures both ability. Furthermore, we also conducted human evaluation study to compare model's ability on content selection and producing coherent texts.

4.2 Implementation Details

Following conditional copy model proposed by Wiseman et al. [22], we use 1-layer encoder to map statistics into R^{600} and use 2-layer LSTM with hidden size of 600 for decoder of the base model. Also, we use input-feeding for attention, following Luong et al. [14]. We set bi-LSTM layer as 1 and hidden size as 600 for both bi-LSTM in hierarchical bi-LSTM encoder. For hierarchical transformer encoder, we set the number of head as 8 and layer as 5 for both transformer encoder. For training, we use Adagrad optimizer [5] with learning rate of 0.15, truncated BPTT (block length 100) and a batch size of 5. During inference, we use beam search to generate texts given tables' information. We set beam size to 5.

4.3 Results

Automatic Evaluation Automatic evaluation results are shown in Table 1. We compare our models with reference. Also we adopt a template system constructed in the same way as the one in [22]. It consists three parts: one introductory sentence generally describe the match which includes the score of both teams and who wins the match, six sentences describe six players statistics in the table who get most scores in the match, ranked by their score from high to low and a conclusion sentence. We defer readers to [22] for more details.

Since conditional copy (CC) model achieves best performance according to Wiseman et al. [22], we use it as base model. We also compare our models with OpAtt [16].

Model	RG P%	RG #	CS P%	CS R%	CS F1%	CO DLD%	BLEU
Reference	94.89	24.14	100.00	100.00	100.00	100.00	100.00
Template	**99.94**	**54.21**	27.02	**58.22**	36.91	15.07	8.58
Conditional Copy [22]	74.80	23.72	29.49	36.18	32.49	15.42	14.19
OpAtt [16]	-	-	-	-	-	-	14.74
Hierarchical bi-LSTM Encoder	89.15	33.81	31.44	45.84	37.30	**18.24**	15.36
Our Model	93.26	30.38	**34.67**	43.17	**38.46**	**18.24**	**15.65**

Table 1. Automatic evaluation results using the updated IE model by [18]. (Our model refers to the hierarchical transformer encoder.)

We compare two hierarchical encoder structure mentioned in approaches, and find that our model (hierarchical transformer encoder) achieves better performance in terms of CS P%, CS F1%. This shows that it has better ability to select important information than hierarchical bi-LSTM encoder. In addition, compared to hierarchical bi-LSTM Encoder, our model achieves higher RG P% and BLEU, which indicates our model can generate more accurate information. Compared with base model and other models on this task, our model also achieves significant improvement on RG, which indicates that our model can generate more high fidelity texts with respect to information in the table. One possible reason is that using hierarchical transformer encoder can capture dependency on both row-level and record-level which help the decoder find relevant records in the table more accurately.

Human Evaluation In this section, we present human evaluation studies, assessing models' ability on choosing salient information and generating coherent texts. We randomly sampled 20 games from test test, shuffled models' produced texts and gave raters $\langle reference, generation\rangle$ pair for review. Raters were asked to identify how many players mentioned in reference was also mentioned in model's output and how many were missed. Also raters were asked to give 1-3 score on model's generated texts. 1 stands for extremely incoherent texts with extensive repetition. 2 stands for moderately coherent texts with little repetition. 3 stands for coherent texts with almost no repetition. Results are presented in Table 2. The results align with the content selection (CS) metrics for automatic evaluation. The higher # include and the lower # miss compared to base model (CC) indicate our model performs better on selecting salient information. Also, our model can produce more coherent texts than base model.

Case Study Figure 3 shows reference and texts generated by the base model and our model. Both models can produce fluent and understandable texts. We can see that base model includes many incorrect claims about player or team's statistics as shown in Figure 1.

Team	PTS	PTS_QTR1	PTS_QTR2	WIN	LOSS	...
Raptors	104	28	28	21	8	...
Jazz	98	29	22	18	13	...

Player	PTS	AST	REB	STL	...
Raptors					
DeMar DeRozan	24	1	6	4	...
Kyle Lowry	36	5	4	2	...
Jonas Valanciunas	14	0	7	1	...
Terrence Ross	10	1	1	0	...
Cory Joseph	4	3	1	1	...
Fred VanVleet	N/A	N/A	N/A	N/A	...
...	...	...	...	...	...
Jazz					
Rudy Gobert	13	1	14	1	...
Gordon Hayward	23	5	1	1	...
Trey Lyles	19	1	7	0	...
Shelvin Mack	17	5	4	1	...
Rodney Hood	2	1	0	0	...
Joe Johnson	7	0	0	0	...
Raul Neto	0	0	1	0	...
George Hill	N/A	N/A	N/A	N/A	...
...	...	...	...	...	...

STAT

On Friday , the Raptors saw a combined 60 points from their All-Star guards . Point guard **Kyle Lowry** had one of his best nights of the season , leading the team with 36 points on ridiculous 15 - of - 20 shooting . **DeMar DeRozan** , meanwhile , scored 24 and recorded four steals on the defensive end . Despite playing just 16 minutes and facing stiff competition in Utah center **Rudy Gobert** , Toronto big man **Jonas Valanciunas** scored 14 points on perfect 5 - of - 5 shooting and had seven rebounds . For Utah , forward **Gordon Hayward** led the way with 23 points . **Hayward** shot 50 percent from the field and had five assists . **Gobert** scored 13 and led the team with 14 rebounds while shooting a solid 5 - of - 7 . Sophomore **Trey Lyles** impressed off the bench , scoring 19 points including four threes .

Reference

The Toronto Raptors defeated the Utah Jazz , 104 - 98 , at Vivint Smart Home Arena on Monday . The Raptors (21 - 8) checked in to Wednesday 's contest looking to snap a five - game losing streak , as they 'd lost four of their last five contests . **DeMar DeRozan** led the way for the Raptors (21 - 8) with 24 points , six rebounds , an assist and four steals . **Kyle Lowry** followed with 36 points , five assists , four rebounds and two steals . **Trey Lyles** was next with a bench - leading 19 points , which he supplemented with seven rebounds and an assist . **Rudy Gobert** turned in a 13 - point , 14 - rebound double - double that also included an assist and a steal . Shelvin Mack led the bench with 17 points , five assists , four rebounds and a steal . Shelvin Mack led the bench with 17 points , five assists , four rebounds and a steal . Shelvin Mack led the bench with 17 points , five assists , four rebounds and a steal . Terrence Ross paced the bench with 10 points , a rebound and an assist . The Jazz remain in second place in the Western Conference 's Northwest Division . They head to Utah to take on the Jazz on Monday .

Our Model

Fig. 3. The generated game summary for *Raptors v.s. Jazz* and its corresponding game statistics. Red words mean it is in line with statistics in the table. Blue words mean it contradicts table's information or it is not the salient player to be mentioned. Underscored player name indicates that the model correctly select this salient player in line with reference.

Model	# include	# miss	coherency
Reference	5.63	0.13	2.95
CC	2.73	3.03	2.73
Our Model	3.78	1.75	2.85

Table 2. Human evaluation results.

It generates many wrong information for *Cory Joseph*, *Rodney Hood*, *Joe Johnson* and *Raul Neto*. It also mistakenly use others' information to describe some of the absent players such as *Fred VanVleet* and *George Hill*. In comparison, most of the statistics information in our model's generated texts are correct. Furthermore, base model mentioned six players who don't appear in reference which indicates they are not salient information.

Instead, our model only includes two players who are not in the reference while still include four important players in the game. This indicates that our model has better ability on content selection than base model.

5 Related Work

Given preprocessed data, the traditional data-to-text systems usually treat the generating task as a pipeline. The first step is to perform document planning, then the traditional models perform microplanning and generating actual sequence of texts via realisation [19]. Recently, neural data-to-text systems perform this task in a end-to-end fashion. Mei et al. [15] propose a pre-selector to generate weather forecast, which strengthens model's content selection ability and obtains considerable improvement over previous models. Some studies focus on transforming Wikipedia infobox into introductory sentences. Sha et al. [20] propose a hybrid attention mechanism to enhance the model's ability on choosing the order of content when generating texts. Liu et al. [13] propose a field-gating encoder focusing on modeling table structure. Meanwhile, Bao et al. [2] develop a table-aware Seq2Seq model. In recent years, a document-scale data-to-text dataset has been introduced, which contains significantly more records and longer target texts by one order of magnitude than previously mentioned datasets. It provides two strong baselines with different copy mechanism to improve model's ability to generate texts with correct record information. There are some studies on this task. Nie et al. [16] introduce pre-executed operation such as minus and argmax in order to help model generate higher fidelity texts. Li et al. [9] decompose the generating process into generating templates at first and then fill in the slots. Meanwhile, Puduppully et al. [18] decompose the process into two stages: selecting a sequence of important records from the table and generating texts according to those important records. However, they treat the table as a set or sequence of records without exploiting the table's multi-row structure. In addition, different from models that decompose data-to-text generation into multiple stages, our model generate texts in one stage.

Some studies encode long input texts in a hierarchical fashion. Cohan et al. [4] propose to use a hierarchical encoder to obtain input's representation on word-level and section-level for single, long-form documents. Also, they propose a discourse-aware decoder that attends to both discourse section and words in the document. Ling et al. [12] also propose to hierarchically encode document with hierarchical attention. Recently, Jain et al. [7] propose a hierarchical encoder and mixed hierarchical attention on data-to-text task.

6 Conclusion

In this work, we develop a hierarchical encoder model that automatically maps the structured table to natural language texts. In detail, we consider the row of the table as an intermediate representation structure to improve the table encoding component. Also these row-level and record-level representations can be used to better focus on important records when generating texts. Experiments were conducted on the ROTOWIRE dataset. Both automatic and human evaluation results show that our neural hierarchical encoder architecture achieves significant improvement over the base model.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments. This work was supported by the National Key R&D Program of China via grant 2018YFB1005103 and National Natural Science Foundation of China (NSFC) via grants 61632011 and 61772156.

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: International Conference on Learning Representations (2015)
2. Bao, J., Tang, D., Duan, N., Yan, Z., Lv, Y., Zhou, M., Zhao, T.: Table-to-text: Describing table region with natural language. In: The Thirty-Second AAAI Conference on Artificial Intelligence. pp. 5020–5027. Association for the Advancement of Artificial Intelligence (2018)
3. Brill, E., Moore, R.C.: An improved error model for noisy channel spelling correction. In: Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics. pp. 286–293. Association for Computational Linguistics (2000)
4. Cohan, A., Dernoncourt, F., Kim, D.S., Bui, T., Kim, S., Chang, W., Goharian, N.: A discourse-aware attention model for abstractive summarization of long documents. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 615–621. Association for Computational Linguistics (2018)
5. Duchi, J.C., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research **12**, 2121–2159 (2010)
6. Gulcehre, C., Ahn, S., Nallapati, R., Zhou, B., Bengio, Y.: Pointing the unknown words. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. pp. 140–149. Association for Computational Linguistics (2016)
7. Jain, P., Laha, A., Sankaranarayanan, K., Nema, P., Khapra, M.M., Shetty, S.: A mixed hierarchical attention based encoder-decoder approach for standard table summarization. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 622–627. Association for Computational Linguistics (2018)
8. Lebret, R., Grangier, D., Auli, M.: Neural text generation from structured data with application to the biography domain. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. pp. 1203–1213. Association for Computational Linguistics (2016)
9. Li, L., Wan, X.: Point precisely: Towards ensuring the precision of data in generated texts using delayed copy mechanism. In: Proceedings of the 27th International Conference on Computational Linguistics. pp. 1044–1055. Association for Computational Linguistics (2018)
10. Liang, P.: Learning executable semantic parsers for natural language understanding. Communications of the ACM **59**(9), 68–76 (2016)
11. Liang, P., Jordan, M., Klein, D.: Learning semantic correspondences with less supervision. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP. pp. 91–99. Association for Computational Linguistics (2009)

12. Ling, J., Rush, A.: Coarse-to-fine attention models for document summarization. In: Proceedings of the Workshop on New Frontiers in Summarization. pp. 33–42. Association for Computational Linguistics (2017)
13. Liu, T., Wang, K., Sha, L., Chang, B., Sui, Z.: Table-to-text generation by structure-aware seq2seq learning. In: The Thirty-Second AAAI Conference on Artificial Intelligence. pp. 4881–4888. Association for the Advancement of Artificial Intelligence (2018)
14. Luong, T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 1412–1421. Association for Computational Linguistics (2015)
15. Mei, H., Bansal, M., Walter, M.R.: What to talk about and how? selective generation using LSTMs with coarse-to-fine alignment. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 720–730. Association for Computational Linguistics (2016)
16. Nie, F., Wang, J., Yao, J.G., Pan, R., Lin, C.Y.: Operation-guided neural networks for high fidelity data-to-text generation. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. pp. 3879–3889. Association for Computational Linguistics (2018)
17. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: Proceedings of 40th Annual Meeting of the Association for Computational Linguistics. pp. 311–318. Association for Computational Linguistics (2002)
18. Puduppully, R., Dong, L., Lapata, M.: Data-to-text generation with content selection and planning. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 6908–6915. Association for the Advancement of Artificial Intelligence (2019)
19. Reiter, E.: An architecture for data-to-text systems. In: Proceedings of the Eleventh European Workshop on Natural Language Generation. pp. 97–104. Association for Computational Linguistics (2007)
20. Sha, L., Mou, L., Liu, T., Poupart, P., Li, S., Chang, B., Sui, Z.: Order-planning neural text generation from structured data. In: The Thirty-Second AAAI Conference on Artificial Intelligence. pp. 5414–5421. Association for the Advancement of Artificial Intelligence (2018)
21. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems. pp. 5998–6008. Curran Associates, Inc., Long Beach, USA (2017)
22. Wiseman, S., Shieber, S., Rush, A.: Challenges in data-to-document generation. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 2253–2263. Association for Computational Linguistics (2017)