Learning to Select, Track, and Generate for Data-to-Text

Hayate Iso[†], Yui Uehara^{††}, Tatsuya Ishigaki^{†††,††}, Hiroshi Noji^{††}, Eiji Aramaki^{†,††},
Ichiro Kobayashi^{††††,††}, Yusuke Miyao^{††††,††}, Naoaki Okazaki^{†††,††} and
Hiroya Takamura^{†††,††}

We propose a data-to-text generation model with two modules, one for tracking and the other for text generation. Our tracking module selects and keeps track of salient information and remembers which record has been mentioned. Our generation module generates a summary conditioned on the state of tracking module. In addition, we also explore the effectiveness of the writer information for generation. Experimental results show that our model outperforms existing models in all evaluation metrics even without writer information. Incorporating writer information further improves the performance, contributing to content planning and surface realization.

Key Words: *Natural Language Generation, Data-to-Text Generation, Entity Tracking*

1 Introduction

Advances in sensor and data storage technologies have rapidly increased the amount of data produced in various fields such as weather, finance, and sports. To address the information overload caused by the massive amounts of data, data-to-text generation technology, which expresses the contents of data in natural language, is becoming more important (Barzilay and Lapata 2005). Recently, neural methods can generate high-quality short summaries especially from small pieces of data (Liu, Wang, Sha, Chang, and Sui 2018).

Despite this success, it remains challenging to generate a high-quality long summary from data (Wiseman, Shieber, and Rush 2017). One of the difficulties is that the input data is too large for a naive model to find its salient part, i.e., to determine which part of the data should be mentioned. In addition, the salient part moves as the summary explains the data. For example, when generating a summary of a basketball game (Table 1 (b)) from the box score (Table 1 (a)), the input contains numerous data records about the game: e.g., *Jordan Clarkson scored 18*

[†] Nara Institute of Science and Technology

^{††} Artificial Intelligence Research Center, AIST

^{†††} Tokyo Institute of Technology

^{††††} Ochanomizu University

^{†††††} The University of Tokyo

Table 1 Example of input and output data: task defines box score (1a) used for input and summary document of game (1b) used as output

TEAM	H/V	WIN	LOSS	PTS	REB	AST	FG_PCT	FG3_PCT	...
KNICKS	H	16	19	104	46	26	45	46	...
BUCKS	V	18	16	105	42	20	47	32	...

PLAYER	H/V	PTS	REB	AST	BLK	STL	MIN	CITY
CARMELO ANTHONY	H	30	11	7	0	2	37	NEW YORK
DERRICK ROSE	H	15	3	4	0	1	33	NEW YORK
COURTNEY LEE	H	11	2	3	1	1	38	NEW YORK
GIANNIS ANTETOKOUNMPO	V	27	13	4	3	1	39	MILWAUKEE
GREG MONROE	V	18	9	4	1	3	31	MILWAUKEE
JABARI PARKER	V	15	4	3	0	1	37	MILWAUKEE
MALCOLM BROGDON	V	12	6	8	0	0	38	MILWAUKEE
MIRZA TELETOMIC	V	13	1	0	0	0	21	MILWAUKEE
JOHN HENSON	V	2	2	0	0	0	14	MILWAUKEE
...	...	...	...	...	...	...	...	...

(a) Box score: Top contingency table shows number of wins and losses and summary of each game. Bottom table shows statistics of each player such as points scored (PLAYER's PTS), and total rebounds (PLAYER's REB).

The **Milwaukee Bucks** defeated the **New York Knicks**, **105-104**, at Madison Square Garden on Wednesday. The **Knicks (16-19)** checked in to Wednesday's contest looking to snap a five-game losing streak and heading into the fourth quarter, they looked like they were well on their way to that goal. ... **Antetokounmpo** led the Bucks with **27** points, **13** rebounds, **four** assists, a steal and **three** blocks, his second consecutive double-double. **Greg Monroe** actually checked in as the second-leading scorer and did so in his customary bench role, posting **18** points, along with **nine** boards, **four** assists, **three** steals and a block. **Jabari Parker** contributed **15** points, **four** rebounds, **three** assists and a steal. **Malcolm Brogdon** went for **12** points, **eight** assists and **six** rebounds. **Mirza Teletovic** was productive in a reserve role as well, generating **13** points and a rebound. ... **Courtney Lee** checked in with **11** points, **three** assists, **two** rebounds, a steal and a block. ... The Bucks and Knicks face off once again in the second game of the home-and-home series, with the meeting taking place Friday night in Milwaukee.

(b) NBA basketball game summary: Each summary consists of game victory or defeat of the game and highlights of valuable players.

Extracted entities are shown in bold face. Extracted values are shown in green.

points. Existing models often refer to the same data record multiple times (Puduppully, Dong, and Lapata 2019a). However, existing models may mention an incorrect data record, e.g., *Kawhi Leonard added 19 points*: the summary should mention *LaMarcus Aldridge*, who scored 19 points. Thus, we need a model that finds salient parts, tracks transitions of salient parts, and expresses information faithful to the input.

In this paper, we propose a novel data-to-text generation model with two modules, one for saliency tracking and another for text generation. The tracking module keeps track of saliency in the input data so as when the module detects a saliency transition, the tracking module selects a new data record¹ and updates the state of the tracking module. The text generation module generates a document conditioned on the current tracking state. The model proposed here could be considered to be a neural network-based variant of the end-to-end data-to-text generation model that selects the input information and keeps decision histories while generating the report

¹ We use 'data record' and 'relation' interchangeably.

(Angeli, Liang, and Klein 2010). The major difference from the traditional model is to make use of the recurrent neural network. Our model employs the recurrent neural network-based saliency tracking module for maintaining the decision histories as continuous vectors instead of explicitly keeping all decision history. In addition, some writer-specific patterns and characteristics i.e., how data records are selected to be mentioned; and how data records are expressed as text, e.g., the order of data records and the word usages. In addition, writer information was added to the model.

The experimental results demonstrate that, even without writer information, this model achieves the best performance among all previous models across all evaluation metrics with a 91.98% precision of relation generation, 43.31 F1 score of content selection, 21.56% normalized Damerau-Levenshtein Distance (DLD) of content ordering, and 15.74 of BLEU score. It was also confirmed that adding writer information further improves the performance.

2 Related Work

2.1 Data-to-Text Generation

Data-to-text generation is a task that generates descriptions from structured or non-structured data including sports commentary (Tanaka-Ishii, Hasida, and Noda 1998; Chen and Mooney 2008; Taniguchi, Feng, Takamura, and Okumura 2019), weather forecasts (Liang, Jordan, and Klein 2009; Mei, Bansal, and Walter 2016), biographical text from the “infobox” in Wikipedia (Lebret, Grangier, and Auli 2016; Sha, Mou, Liu, Poupart, Li, Chang, and Sui 2018; Liu et al. 2018) and market comments on stock prices (Murakami, Watanabe, Miyazawa, Goshima, Yanase, Takamura, and Miyao 2017; Aoki, Miyazawa, Ishigaki, Goshima, Aoki, Kobayashi, Takamura, and Miyao 2018).

Neural generation methods have become the mainstream approach for data-to-text generation. The encoder-decoder framework (Cho, van Merriënboer, Gulcehre, Bahdanau, Bougares, Schwenk, and Bengio 2014; Sutskever, Vinyals, and Le 2014) with the attention (Bahdanau, Cho, and Bengio 2015; Luong, Pham, and Manning 2015) and copy mechanism (Gu, Lu, Li, and Li 2016; Gulcehre, Ahn, Nallapati, Zhou, and Bengio 2016) has been successfully applied to data-to-text tasks. However, neural generation methods sometimes yield fluent but inadequate descriptions (Tu, Liu, Lu, Liu, and Li 2017). In data-to-text generation, descriptions inconsistent with the input data are highly problematic.

Recently, Wiseman et al. (2017) introduced the ROTOWIRE dataset, which contains multi-sentence summaries of basketball games with box-score (Table 1). This dataset requires the

selection of a salient subset of data records for generating descriptions. They also proposed automatic evaluation metrics for measuring the informativeness of generated summaries. Their dataset and metrics have also been used by other researchers (Nie, Wang, Yao, Pan, and Lin 2018; Li and Wan 2018; Puduppully et al. 2019a; Puduppully, Dong, and Lapata 2019b; Rebuffel, Soulier, Scoutheeten, and Gallinari 2020).

Puduppully et al. (2019a) proposed a two-stage method that first predicts the sequence of data records to be mentioned and then generates a summary conditioned on the predicted sequences. Their idea is similar to the one proposed here in that the both consider a sequence of data records as content planning. However, the model proposed here differs from theirs in this one uses a recurrent neural network for saliency tracking, and that the decoder dynamically chooses a data record to be mentioned without fixing a sequence of data records.

Puduppully et al. (2019b) simultaneously proposed an entity-centric method for data-to-text generation tasks, which keeps track of each entity’s state at every time step. The advantage of this approach is that the model does not require a random variable annotation. In contrast to their approach, our model has intermediate discrete actions that should make it easier to interpret the generation behavior of the model. In addition, our model shows empirically better performance on both ROTOWIRE-MODIFIED and ROTOWIRE-FG, especially for the content selection performance.

2.2 Memory Modules

The memory network can be used to maintain and update representations of salient information (Weston, Chopra, and Bordes 2015; Sukhbaatar, Szlam, Weston, and Fergus 2015; Graves, Wayne, Reynolds, Harley, Danihelka, Grabska-Barwińska, Colmenarejo, Grefenstette, Ramalho, Agapiou, et al. 2016; Henaff, Weston, Szlam, Bordes, and LeCun 2017). This module is often used in natural language understanding to keep track of the entity state (Kobayashi, Tian, Okazaki, and Inui 2016; Hoang, Wiseman, and Rush 2018; Bosselut, Levy, Holtzman, Ennis, Fox, and Choi 2018).

Recently, entity tracking has become popular when trying to generate coherent text (Kiddon, Zettlemoyer, and Choi 2016; Ji, Tan, Martschat, Choi, and Smith 2017; Yang, Blunsom, Dyer, and Ling 2017; Clark, Ji, and Smith 2018). Kiddon et al. (2016) proposed a neural checklist model that updates predefined item states. Furthermore, Ji et al. (2017) proposed an entity representation for the language model that updated entity tracking states when the entity was introduced. Notably, their method selects the salient entity state whereas our model extends this entity tracking module for data-to-text generation tasks. The entity tracking module selects the

salient entity and appropriate attribute in each timestep, updates their states, and generates coherent summaries from the selected data record.

3 Data

Through careful examination, it was found that in the original dataset ROTOWIRE, some NBA games had two documents, one of which was sometimes in the training data and the other was in the test or validation data. Such documents are similar to each other, though not identical. To make this dataset more reliable as an experimental dataset, a new version was created.

The script provided by Wiseman et al. (2017) was run, which was used for crawling the ROTOWIRE website for NBA game summaries. The script collected approximately 78% of the documents in the original dataset; the remaining documents disappeared. The box-scores associated with the collected documents were also collected. It was observed that some of the box-scores were modified compared to the original ROTOWIRE dataset.

The dataset contains 3,738 instances (i.e., pairs of a document and box-scores). However, the four shortest documents were not summaries; they were, for example, an announcement about the postponement of a match. Thus 4 instances like these were deleted and the set was left with 3,734 instances. We followed the dataset split by Wiseman et al. (2017) to split our dataset into training, development, and test data. Finally, the sizes of our training, development, test dataset are respectively 2705, 532, and 497. On average, each summary has 384 tokens and 644 data records. Each match had only one summary in the dataset. Information on the writer for each document was also collected and the dataset contained 31 different writers. The most prolific writer in the dataset wrote 607 documents. There were also writers who wrote less than ten documents; on average, each writer wrote 117 documents. The statistics of the number of relations and the length of summary for each writer were also calculated. As shown in Figure 1, each writer had a remarkable tendency for the number of retrieved relations and the length of the summary. This new dataset is called ROTOWIRE-MODIFIED.²

4 Saliency-Aware Text Generation

At the core of this model is a neural language model with a hidden state $\mathbf{h}^{\text{LM}} \in \mathbb{R}^h$ to generate a summary $y_{1:T} = (y_1, \dots, y_T)$ given a set of data records $\mathbf{x}$, where h is the size of hidden states.

² For information about the dataset, please follow this link: <https://github.com/aistairc/rotowire-modified>

Fig. 1 Bar chart for the number of summaries for each writer, and boxplots for the number of relations and the length of summary for each writer. Horizontal axes are sorted by the median number of relations retrieved by the information extractor.

This model has another hidden state $\mathbf{h}^{\text{ENT}} \in \mathbb{R}^h$, which is used to remember the data records that have been referred to. $\mathbf{h}^{\text{ENT}}$ is also used to update $\mathbf{h}^{\text{LM}}$, meaning that the referred data records affect the text generation.

Our model decides whether to refer to $\mathbf{x}$, which data record $r \in \mathbf{x}$ to be mentioned, and how to express a number. The selected data record is used to update $\mathbf{h}^{\text{ENT}}$. Formally, we use the four variables:

- (1) Z_t : binary variable that determines whether the model refers to input $\mathbf{x}$ at time step t ($Z_t = 1$).
- (2) E_t : At each time step t , this variable indicates the salient entity (e.g., HAWKS, LEBRON JAMES).
- (3) A_t : At each time step t , this variable indicates the salient attribute to be mentioned (e.g., PTS).

Table 2 A running example of our model’s generation process

t	199	200	201	202	203	204	205	206	207	208	209
Y_t	Jabari	Parker	contributed	15	points	,	four	rebounds	,	three	assists
Z_t	1	1	0	1	0	0	1	0	0	1	0
E_t	JABARI PARKER	JABARI PARKER	—	JABARI PARKER	—	—	JABARI PARKER	—	—	JABARI PARKER	—
A_t	FIRST NAME	LAST NAME	—	PLAYER PTS	—	—	PLAYER REB	—	—	PLAYER AST	—
N_t	—	—	—	0	—	—	1	—	—	1	—

At every time step t , the model predicts each random variable. The model firstly determines whether to refer to data records ($Z_t = 1$) or not ($Z_t = 0$). If random variable $Z_t = 1$, model selects entity E_t , its attribute A_t and binary variables N_t if needed. For example, at $t = 202$, the model predicts random variable $Z_{202} = 1$ and then selects the entity **Jabari Parker** and its attribute **Player Pts**. Given these values, the model outputs token **15** from selected data record.

- (4) N_t : If attribute A_t of the salient entity E_t is a numeric attribute, this variable determines if a value in the data records should be output in Arabic numerals (e.g., 50) ($N_t = 1$) or in English words (e.g., five) ($N_t = 0$).

To keep track of the salient entity, our model predicts these random variables at each time step t through its summary generation process. A running example of our model is shown in Table 2 and full algorithm is described in Algorithm 1. For a concise description, we omit the condition for each probability notation. <SoD> and <EoD> represent “start of the document” and “end of the document,” respectively.

In the following subsections, we explain how to initialize the model, predict these random variables, and generate a summary. The illustration of our proposed model is shown in Figure 2.

Before explaining our method, we describe our notation. Let $\mathcal{E}$ and $\mathcal{A}$ denote the sets of entities and attributes, respectively. Each record $r \in \mathbf{x}$ consists of entity $e \in \mathcal{E}$, attribute $a \in \mathcal{A}$, and its value $\mathbf{x}[e, a]$, and is therefore represented as $r = (e, a, \mathbf{x}[e, a])$. For example, the box-score in Table 1 has a record r such that $e = \text{ANTHONY DAVIS}$, $a = \text{PTS}$, and $\mathbf{x}[e, a] = 20$.

4.1 Initialization

Let $\mathbf{r} \in \mathbb{R}^d$ denote the embedding of data record $r \in \mathbf{x}$, where d is the dimension of embeddings. Let $\bar{\mathbf{e}}$ denote the embedding of entity e . Note that $\bar{\mathbf{e}}$ depends on the set of data records, i.e., it depends on the game. We also use $\mathbf{e}$ for static embedding of entity e , which, on the other hand, does not depend on the game.

Given the embedding of entity $\mathbf{e}$, attribute $\mathbf{a}$, and its value $\mathbf{v}$, we use the concatenation layer to combine the information from these vectors to produce the embedding of each data record

Algorithm 1: Generation process

Input: Data records $\mathbf{s}$,
Annotations $Z_{1:T}, E_{1:T}, A_{1:T}, N_{1:T}, \mathbf{h}_0^{\text{LM}}, \mathbf{h}_0^{\text{ENT}}$

```

1 Initialize  $\{\mathbf{r}_{e,a,v}\}_{r \in \mathbf{x}}, \{\bar{\mathbf{e}}\}_{e \in \mathcal{E}}, \mathbf{h}_0^{\text{LM}}, \mathbf{h}_0^{\text{ENT}}$ 
2  $t \leftarrow 0$ 
3  $e_t, y_t \leftarrow \text{NONE}, < \text{SoD} >$ 
4 while  $y_t \neq < \text{EoD} >$  do
5    $t \leftarrow t + 1$ 
6   if  $p(Z_t = 1) \geq 0.5$  then
7     /* Select the entity */
8      $e_t \leftarrow \arg \max p(E_t = e'_t)$ 
9     if  $e_t \notin \mathcal{E}_{t-1}$  then
10       /* If  $e_t$  is a new entity */
11        $\mathbf{h}_t^{\text{ENT}'} \leftarrow \text{GRU}^E(\bar{\mathbf{e}}_t, \mathbf{h}_{t-1}^{\text{ENT}})$ 
12        $\mathcal{E}_t \leftarrow \mathcal{E}_{t-1} \cup \{e_t\}$ 
13     else if  $e_t \neq e_{t-1}$  then
14       /* If  $e_t$  has been observed before, but is different from the previous one. */
15        $\mathbf{h}_t^{\text{ENT}'} \leftarrow \text{GRU}^E(\mathbf{W}^S \mathbf{h}_s^{\text{ENT}}, \mathbf{h}_{t-1}^{\text{ENT}})$ ,
16       where  $s = \max\{s : s \leq t-1, e = e_s\}$ 
17     else
18        $\mathbf{h}_t^{\text{ENT}'} \leftarrow \mathbf{h}_{t-1}^{\text{ENT}}$ 
19     /* Select an attribute for the entity,  $e_t$ . */
20      $a_t \leftarrow \arg \max p(A_t = a'_t)$ 
21      $\mathbf{h}_t^{\text{ENT}} \leftarrow \text{GRU}^A(\mathbf{r}_{e_t, a_t, \mathbf{x}[e_t, a_t]}, \mathbf{h}_t^{\text{ENT}'})$ 
22     if  $a_t$  is a number attribute then
23       if  $p(N_t = 1) \geq 0.5$  then
24          $y_t \leftarrow \text{Arabic numeral of } \mathbf{x}[e_t, a_t]$ 
25       else
26          $y_t \leftarrow \mathbf{x}[e_t, a_t]$ 
27       end
28     else
29        $y_t \leftarrow \mathbf{x}[e_t, a_t]$ 
30   end
31    $e_t, a_t, \mathbf{h}_t^{\text{ENT}} \leftarrow e_{t-1}, a_{t-1}, \mathbf{h}_{t-1}^{\text{ENT}}$ 
32 end
33  $\mathbf{h}_t' \leftarrow \tanh(\mathbf{W}^H(\mathbf{h}_{t-1}^{\text{LM}} \oplus \mathbf{h}_t^{\text{ENT}}))$ 
34 if  $p(Z_t = 0) > 0.5$  then
35    $y_t \leftarrow \arg \max p(Y_t)$ 
36    $\mathbf{h}_t^{\text{LM}} \leftarrow \text{LSTM}(\mathbf{y}_t \oplus \mathbf{h}_t', \mathbf{h}_{t-1}^{\text{LM}})$ 
37 if  $y_t$  is “.” then
38    $\mathbf{h}_t^{\text{ENT}} \leftarrow \text{GRU}^A(\mathbf{v}_{\text{REFRESH}}, \mathbf{h}_t^{\text{ENT}})$ 
39 end
40 return  $y_{1:t-1}$ ;

```

(e, a, v) , denoted as $\mathbf{r}_{e,a,v}$ as follows:

$$\mathbf{r}_{e,a,v} = \tanh\left(\mathbf{e} + \mathbf{a} + \mathbf{v} + \mathbf{b}^{\text{R}}\right), \quad (1)$$

where $\mathbf{b}^{\text{R}} \in \mathbb{R}^d$ denotes a bias vector.³

We obtain a game-dependent entity embedding $\bar{\mathbf{e}}$ in the set of data records $\mathbf{x}$, by summing all the data-record embeddings transformed by a matrix:

³ We also concatenate the embedding vectors that represents whether the entity is in home or away team.

Fig. 2 Illustrations of generation process. It consists of hidden states of language modeling, entity tracking, and dynamic entity representations as external memory.

$$\bar{e} = \tanh \left(\sum_{a \in \mathcal{A}} \mathbf{W}_a^A \mathbf{r}_{e,a,\mathbf{x}[e,a]} + \mathbf{b}_a^A \right), \quad (2)$$

where $\mathbf{W}_a^A \in \mathbb{R}^{d \times d}$ is a weight matrix, and $\mathbf{b}_a^A \in \mathbb{R}^d$ is a bias vector for attribute a , respectively. Since $\bar{e}$ depends on the game as above, $\bar{e}$ is supposed to represent how entity e played in the game.

To initialize the hidden state of each module, we use embeddings of $\langle \text{SoD} \rangle$ for $\mathbf{h}^{\text{LM}}$ and averaged embeddings of $\bar{e}$ for $\mathbf{h}^{\text{ENT}}$.

4.2 Saliency Transition

Generally, the saliency of text changes during text generation. In our work, we suppose that the saliency is represented as the entity and its attribute being talked about. We therefore propose a model that refers to a data record at each timepoint, and transits to another as text goes.

To determine whether to transit to another data record or not at time t , the model calculates the following probability:

$$p(Z_t = 1 \mid \mathbf{h}_{t-1}^{\text{LM}}, \mathbf{h}_{t-1}^{\text{ENT}}) = \sigma(\mathbf{w}_z^\top (\mathbf{h}_{t-1}^{\text{LM}} \oplus \mathbf{h}_{t-1}^{\text{ENT}}) + b_z), \quad (3)$$

where $\mathbf{w}_z \in \mathbb{R}^{2d}$ is a weight vector, $\oplus$ indicates the concatenation of vectors, and $\sigma(\cdot)$ is the sigmoid function. If $p(Z_t = 1 \mid \mathbf{h}_{t-1}^{\text{LM}}, \mathbf{h}_{t-1}^{\text{ENT}})$ is high, the model transits to another data record.

When the model decides to transit to another, the model then determines which entity and

attribute to refer to, and generates the next word (Section 4.3). On the other hand, if the model decides not to transit to another, the model generates the next word without updating the tracking states $\mathbf{h}_t^{\text{ENT}} = \mathbf{h}_{t-1}^{\text{ENT}}$ (Section 4.4).

4.3 Selection and Tracking

When the model refers to a data record ($Z_t = 1$), it selects an entity and its attribute. It also tracks the saliency by putting the information about the selected entity and attribute into the memory vector $\mathbf{h}^{\text{ENT}}$. The model begins to select the subject entity and update the memory states if the subject entity will change.

Specifically, the model first calculates the probability of selecting an entity e_t that is referred at time step t :

$$p(E_t = e \mid \mathbf{h}_{t-1}^{\text{LM}}, \mathbf{h}_{t-1}^{\text{ENT}}) \propto \begin{cases} \exp(\mathbf{h}_s^{\text{ENT}\top} \mathbf{W}^{\text{OLD}} \mathbf{h}_{t-1}^{\text{LM}}) & \text{if } e \in \mathcal{E}_{t-1} \\ \exp(\bar{\mathbf{e}}^\top \mathbf{W}^{\text{NEW}} \mathbf{h}_{t-1}^{\text{LM}}) & \text{otherwise} \end{cases}, \quad (4)$$

where $\mathbf{W}^{\text{OLD}} \in \mathbb{R}^{h \times h}$ and $\mathbf{W}^{\text{NEW}} \in \mathbb{R}^{d \times h}$ are weight matrices, $\mathcal{E}_{t-1}$ is the set of entities that have already been referred to by time step t , and s is defined as $s = \max\{s : s \leq t-1, e = e_s\}$, which indicates the time step when this entity was last mentioned.

The model selects the most probable entity as the next salient entity and updates the set of entities that appeared ($\mathcal{E}_t = \mathcal{E}_{t-1} \cup \{e_t\}$).

If the salient entity changes ($e_t \neq e_{t-1}$), the model updates the hidden state of the tracking model $\mathbf{h}^{\text{ENT}}$ with a recurrent neural network with a gated recurrent unit (GRU) (Chung, Gulcehre, Cho, and Bengio 2014):

$$\mathbf{h}_t^{\text{ENT}'} = \begin{cases} \mathbf{h}_{t-1}^{\text{ENT}} & \text{if } e_t = e_{t-1} \\ \text{GRU}^E(\bar{\mathbf{e}}, \mathbf{h}_{t-1}^{\text{ENT}}) & \text{else if } e_t \notin \mathcal{E}_{t-1} \\ \text{GRU}^E(\mathbf{W}^{\text{S}} \mathbf{h}^{\text{ENT}} + \mathbf{b}^{\text{S}}, \mathbf{h}_{t-1}^{\text{ENT}}) & \text{otherwise,} \end{cases} \quad (5)$$

where $\mathbf{h}_t^{\text{ENT}'}$ is the intermediate entity tracker representation, $\mathbf{W}^{\text{S}} \in \mathbb{R}^{d \times h}$ is a weight matrix, and $\mathbf{b}^{\text{S}} \in \mathbb{R}^d$ is a bias vector. Note that if the selected entity at time step t , e_t , is identical to the previously selected entity e_{t-1} , the hidden state of the tracking model is not updated. If the selected entity e_t is new ($e_t \notin \mathcal{E}_{t-1}$), the hidden state of the tracking model is updated with the embedding $\bar{\mathbf{e}}$ of entity e_t as input. In contrast, if entity e_t has already appeared in the past ($e_t \in \mathcal{E}_{t-1}$) but is not identical to the previous one ($e_t \neq e_{t-1}$), we use $\mathbf{h}_s^{\text{ENT}}$ (i.e., the memory

state when this entity last appeared) to fully exploit the local history of this entity. In other words, the entity tracker contains the *memory states* of all the selected entities, instead of having explicitly separate memory modules for the entities as in the model proposed by Puduppully et al. (2019b).

Given the updated hidden state of the tracking model $\mathbf{h}_t^{\text{ENT}'}$, we next select the attribute of the salient entity at time t , a_t , by the following probability:

$$\begin{aligned} & p(A_t = a \mid e_t, \mathbf{h}_{t-1}^{\text{LM}}, \mathbf{h}_t^{\text{ENT}'}) \\ & \propto \exp \left(\mathbf{r}_{e_t, a, \mathbf{x}[e_t, a]}^\top \mathbf{W}^{\text{ATTR}} (\mathbf{h}_{t-1}^{\text{LM}} \oplus \mathbf{h}_t^{\text{ENT}'}) \right), \end{aligned} \quad (6)$$

where $\mathbf{W}^{\text{ATTR}} \in \mathbb{R}^{d \times 2h}$ is a weight matrix. After selecting a_t , i.e., the most probable attribute of the salient entity, the tracking model updates the memory state $\mathbf{h}_t^{\text{ENT}}$ with the embedding of the data record $\mathbf{r}_{e_t, a_t, \mathbf{x}[e_t, a_t]}$ introduced in Section 4.1:

$$\mathbf{h}_t^{\text{ENT}} = \text{GRU}^{\text{A}}(\mathbf{r}_{e_t, a_t, \mathbf{x}[e_t, a_t]}, \mathbf{h}_t^{\text{ENT}'}). \quad (7)$$

Note that we used different GRU units (GRU^{E} and GRU^{A}) for updating the tracking model because they accept the different type of embeddings for the input.

Once the entity is referred to the model, we use the hidden states of lastly mentioned time step for each entity as the hidden states of the entity.

4.4 Summary Generation

Given the two hidden states, one for language model $\mathbf{h}_{t-1}^{\text{LM}}$ and the other for tracking model $\mathbf{h}_t^{\text{ENT}}$, the model generates the next word y_t . We also incorporate a copy mechanism that copies the value of the salient data record $\mathbf{x}[e_t, a_t]$.

If the model refers to a data record ($Z_t = 1$), it directly copies the value of the data record $\mathbf{x}[e_t, a_t]$. However, the values of numerical attributes can be expressed in at least two different manners: Arabic numerals (e.g., *14*) and English words (e.g., *fourteen*). One simple solution would be to follow the basic English rule that uses English words for single digits and Arabic numerals for the others. However, in the real descriptions, we found many irregular usages that do not conform to the basic English rule above. Thus, we decide which one to use by the following probability:

$$p(N_t = 1 \mid \mathbf{h}_{t-1}^{\text{LM}}, \mathbf{h}_t^{\text{ENT}}) = \sigma(\mathbf{w}^{\text{N}\top} (\mathbf{h}_{t-1}^{\text{LM}} \oplus \mathbf{h}_t^{\text{ENT}}) + b^{\text{N}}), \quad (8)$$

where $\mathbf{w}^N \in \mathbb{R}^{2h}$ is a weight vector. The model then updates the hidden states of the language model:

$$\mathbf{h}'_t = \tanh \left(\mathbf{W}^H (\mathbf{h}_{t-1}^{\text{LM}} \oplus \mathbf{h}_t^{\text{ENT}}) + \mathbf{b}^H \right), \quad (9)$$

where $\mathbf{W}^H \in \mathbb{R}^{h \times 2h}$ is a weight matrix, and $\mathbf{b}^H \in \mathbb{R}^h$ is a bias vector.

If the model does not refer to a data record ($Z_t = 0$), it predicts the next word y_t via a probability over words conditioned on the context vector $\mathbf{h}'_t$:

$$p(Y_t | \mathbf{h}'_t) = \text{softmax}(\mathbf{W}^Y \mathbf{h}'_t), \quad (10)$$

where $\mathbf{W}^Y \in \mathbb{R}^{|\mathcal{V}| \times h}$ is a weight matrix and $|\mathcal{V}|$ is the size of the vocabulary.

Subsequently, the hidden state of language model $\mathbf{h}^{\text{LM}}$ is updated:

$$\mathbf{h}_t^{\text{LM}} = \text{LSTM}(\mathbf{y}_t \oplus \mathbf{h}'_t, \mathbf{h}_{t-1}^{\text{LM}}), \quad (11)$$

where $\mathbf{y}_t \in \mathbb{R}^d$ is the embedding of the word generated at time step t .⁴

4.5 Incorporating Writer Information

For the document-level data-to-text generation task, we hypothesize that each writer would select the different entity information from the table and describe the summary in their writing style. To test our hypothesis, we also incorporate the information about the writer into our model to study how external style information affects the generation. Specifically, instead of using Equation (9), we concatenate the embedding $\mathbf{w}$ of a writer to $\mathbf{h}_{t-1}^{\text{LM}} \oplus \mathbf{h}_t^{\text{ENT}}$ to construct context vector $\mathbf{h}'_t$:

$$\mathbf{h}'_t = \tanh \left(\mathbf{W}'^H (\mathbf{h}_{t-1}^{\text{LM}} \oplus \mathbf{h}_t^{\text{ENT}}) + \mathbf{w} + \mathbf{b}'^H \right), \quad (12)$$

Since this new context vector $\mathbf{h}'_t$ is used for calculating the probability over words in Equation (10), the writer information will directly affect word generation, which is regarded as surface realization in terms of traditional text generation. Simultaneously, context vector $\mathbf{h}'_t$ enhanced with the writer information is used to obtain $\mathbf{h}_t^{\text{LM}}$, which is the hidden state of the language model and is further used to select the salient entity and attribute, as mentioned in Sections 4.2 and 4.3. Therefore, in our model, the writer information affects both surface realization and content planning.

⁴ In our initial experiment, we observed a word repetition problem when the tracking model is not updated during generating each sentence. To avoid this problem, we also update the tracking model with special trainable vectors $\mathbf{v}_{\text{REFRESH}}$ to refresh these states after our model generates a period: $\mathbf{h}_t^{\text{ENT}} = \text{GRU}^A(\mathbf{v}_{\text{REFRESH}}, \mathbf{h}_t^{\text{ENT}})$

4.6 Learning Objective

We apply fully supervised training that maximizes the following log-likelihood:

$$\begin{aligned}
& \log p(Y_{1:T}, Z_{1:T}, E_{1:T}, A_{1:T}, N_{1:T} \mid \mathbf{x}) \\
&= \sum_{t=1}^T \log p(Z_t = z_t \mid \mathbf{h}_{t-1}^{\text{LM}}, \mathbf{h}_{t-1}^{\text{ENT}}) \\
&\quad + \sum_{t:Z_t=1} \log p(E_t = e_t \mid \mathbf{h}_{t-1}^{\text{LM}}, \mathbf{h}_{t-1}^{\text{ENT}}) \\
&\quad + \sum_{t:Z_t=1} \log p(A_t = a_t \mid e_t, \mathbf{h}_{t-1}^{\text{LM}}, \mathbf{h}_t^{\text{ENT}'}) \\
&\quad + \sum_{t:Z_t=1, a_t \text{ is num_attr}} \log p(N_t = n_t \mid \mathbf{h}_{t-1}^{\text{LM}}, \mathbf{h}_t^{\text{ENT}}) \\
&\quad + \sum_{t:Z_t=0} \log p(Y_t = y_t \mid \mathbf{h}_t')
\end{aligned}$$

5 Experiments

5.1 Experimental Settings

We used ROTOWIRE-MODIFIED as the dataset for our experiments, which we explained in Section 3. The training, development, and test data respectively contained 2,705, 532, and 497 games.

We also used recently introduced dataset, ROTOWIRE-FG (Wang 2019), which is enlarged, enriched, and purified dataset from the original ROTOWIRE dataset. The biggest difference with the ROTOWIRE-MODIFIED is whether to include the unsupported sentence by the box-scores. In the ROTOWIRE-FG dataset, they convert all of the numbers written in English words into the Arabic numbers. Thus, we don't have to introduce the random variable N_t to determine the surface realization of the selected number for this dataset. We used the official split for training, development, and test data respectively contained games.

Since we take a supervised training approach, we need the annotations of the random variables (i.e., Z_t , E_t , A_t , and N_t) in the training data, as shown in Table 2. Instead of simple lexical matching with $r \in \mathbf{x}$, which is prone to errors in the annotation, we use the information extraction systems. For ROTOWIRE-MODIFIED dataset, we used the information extraction model provided by Wiseman et al. (2017) and for ROTOWIRE-FG dataset, we used the hard-coded information retrieval system provided by Wang (2019). These annotation performances directly affect the model performance. Although the model-based system is trained on noisy rule-based annotations,

we conjecture that it is more robust to errors because it is trained to minimize the marginalized loss function for ambiguous relations. However, the model only considers sentence-level relation extractions so that the model always fails to extract relations across the sentence.

We set the dimensions of the embeddings to 128, and those of the hidden state of RNN to 512 and all of parameters are initialized with the Xavier initialization (Glorot and Bengio 2010). We set the maximum number of epochs to 30, and choose the model with the highest BLEU score on the development data. The initial learning rate is $2e-3$ and AMSGrad is also used for automatically adjusting the learning rate (Reddi, Kale, and Kumar 2018). Our implementation uses DyNet (Neubig, Dyer, Goldberg, Matthews, Ammar, Anastasopoulos, Ballesteros, Chiang, Clothiaux, Cohn, et al. 2017).

5.2 Models to be Compared

We compare our model⁵ against the following baseline models.

- ED-CC: the model consists of an attention-based encoder-decoder model with conditional copy mechanism used by Wiseman et al. (2017).
- HIERARCHICAL: the model consists of the encoder with hierarchical attention proposed by Rebuffel et al. (2020)
- NCP: the model first predicts the sequence of data records and then generates a summary conditioned on the predicted sequences proposed by Puduppully et al. (2019a).
- ENT: the model is simultaneously proposed with ours, which extends the encoder-decoder model with memory modules proposed by Puduppully et al. (2019b).

ED-CC model refers to all data records every timestep, while NCP model refers to a subset of all data records, which is predicted in the first stage. Unlike these models, our model uses one memory vector $\mathbf{h}_t^{\text{ENT}}$ that tracks the history of the data records, during generation. We retrained the baselines on our new dataset. We also present the performance of GOLD and TEMPLATES summaries. The GOLD summary is exactly identical with the reference summary and each TEMPLATES summary is generated in the same manner as Wiseman et al. (2017).

In the latter half of our experiments, we examine the effect of adding information about writers. In addition to our model enhanced with writer information, we also add writer information to the NCP model by Puduppully et al. (2019a). Their method consists of two stages corresponding to content planning and surface realization. Therefore, by incorporating writer information to each of the two stages, we can clearly see which part of the model to which the writer information

⁵ Our code is available from <https://github.com/aistairc/sports-reporter>

contributes to. For NCP model, we attach the writer information in the following three ways:

- (1) concatenating writer embedding $\mathbf{w}$ with the input vector for LSTM in the content planning decoder (stage 1);
- (2) concatenating writer embedding $\mathbf{w}$ with the input vector for LSTM in the text generator (stage 2);
- (3) using both 1 and 2 above.

For more details about each decoding stage, readers can refer to Puduppully et al. (2019a).

5.3 Evaluation Metrics

As evaluation metrics, we use BLEU score (Papineni, Roukos, Ward, and Zhu 2002) and the extractive metrics proposed by Wiseman et al. (2017), i.e., relation generation (RG), content selection (CS), and content ordering (CO) as evaluation metrics. The extractive metrics measure how well the relations extracted from the generated summary match the correct relations:⁶

- RG: the ratio of the correct relations out of all the extracted relations, where correct relations are relations found in the input data records $\mathbf{x}$. The average number of extracted relations is also reported.
- CS: precision and recall of the relations extracted from the generated summary against those from the reference summary.
- CO: edit distance measured with normalized Damerau-Levenshtein Distance (DLD) between the sequences of relations extracted from the generated and reference summary.

6 Results and Discussions

We first focus on the quality of tracking model and entity representation in Sections 6.1 to 6.4, where we use the model without writer information. We examine the effect of writer information in Section 6.5.

6.1 Saliency Tracking-Based Model

As shown in Table 3, our model outperforms all baselines across all evaluation metrics on ROTOWIRE-MODIFIED. While the most similar model, ENT, shows the similar performance, our model performs noticeable improvement on F1 score of the content selection.

⁶ The model for extracting relation tuples was trained on tuples made from the entity (e.g., team name, city name, player name) and attribute value (e.g., “Lakers”, “92”) extracted from the summaries, and the corresponding attributes (e.g., “TEAM NAME”, “PTS”) found in the box- or line-score. The precision and the recall of this extraction model are respectively 93.4% and 75.0% in the test data.

Table 3 Experimental result on the ROTOWire-MODIFIED

Method	RG		CS			CO	BLEU
	#	P%	P%	R%	F1%	DLD%	
GOLD	27.36	93.42	100.	100.	100.	100.	100.
TEMPLATES	54.63	100.	31.01	58.85	40.61	17.50	8.43
ED-CC	20.28	61.76	27.20	29.76	28.42	15.88	15.26
HIERARCHICAL	26.34	78.74	35.74	39.04	37.32	20.36	15.40
NCP	34.05	82.55	32.30	43.74	37.16	16.67	14.82
ENT	32.41	91.13	37.05	43.06	39.83	20.62	15.23
PROPOSED	31.66	91.98	40.44	46.63	43.31	21.56	15.74

Each metric evaluates whether important information (CS) is described accurately (RG) and in correct order (CO).

Table 4 Experimental result on the ROTOWire-FG

Method	RG		CS			CO	BLEU
	#	P%	P%	R%	F1%	DLD%	
TEMPLATES	51.80	98.89	23.98	43.96	31.03	10.25	12.09
ED-CC	30.28	82.16	35.84	38.40	37.08	18.45	20.80
ENT	35.69	93.72	39.04	49.29	43.57	17.50	21.23
NCP	35.99	94.21	43.31	55.15	48.52	23.46	23.86
NCP+ TR	37.49	95.70	42.90	56.91	48.92	24.47	24.41
PROPOSED	40.74	98.36	43.38	62.53	51.22	20.64	24.23

Results of baselines are taken from Wang (2019). TR stands for table reconstruction.

We also show the result on the ROTOWire-FG dataset in Table 4. Our model outperforms the relation generation and the content selection scores and shows the second-highest performance on the BLEU score. The NCP+TR model, however, clearly shows the best performance on the content ordering and BLEU score.

These two experimental results show the following properties. (1) The entity-centric models, which are the ENT model and our proposed model, perform almost always better than the baselines under the noisy data-to-text setting (ROTOWire-MODIFIED). (2) The explicit annotation, where the NCP model and our proposed model have, is particularly useful under the clean data-to-text setting (ROTOWire-FG). In other words, our proposed model could be seen as picking the best characteristics of both the ENT and the NCP models.

6.2 Qualitative Analysis of Entity Embedding

Our model has the game-dependent entity embedding $\bar{e}$, which depends on the box score for each game in addition to static entity embedding e . Now we analyze the difference of these two

types of embeddings.

We present a two-dimensional visualizations of both embeddings produced using PCA (Pearson 1901). As shown in Figure 3, which is the visualization of static entity embedding $\mathbf{e}$, the top-ranked players are closely located.

We also present the visualizations of game-dependent entity embeddings $\bar{\mathbf{e}}$ in Figure 4. Al-

Fig. 3 Illustrations of static entity embeddings $\mathbf{e}$. Players with colored letters are listed in the ranking top 100 players for the 2016-17 NBA season. Only *LeBron James* is in red and the other players in top 100 are in blue. Top-ranked players have similar representations of $\mathbf{e}$.

Fig. 4 Illustrations of game-dependent entity embedding $\bar{\mathbf{e}}$. Both left and right figures are for *Cleveland Cavaliers* vs. *Detroit Pistons*, on different dates. LeBron James is in red letters. Entities with orange symbols appeared only in the reference summary. Entities with blue symbols appeared only in the generated summary. Entities with green symbols appeared in both the reference and the generated summary. The others are with red symbols. $\square$ represents player who scored in the double digits, and $\diamond$ represents player who recorded double-double. Players with $\triangle$ did not participate in the game. $\circ$ represents other players.

though we did not carry out feature engineering specific to the NBA (e.g., whether a player scored double digits or not)⁷ for representing the game-dependent entity embedding $\bar{e}$, the embeddings of the players who performed well for each game have similar representations. In addition, the change in embeddings of the same player was observed depending on the box-scores for each game. For instance, *LeBron James* recorded a double-double in a game on April 22, 2016. For this game, his embedding is located close to the embedding of *Kevin Love*, who also scored a double-double. However, he did not participate in the game on December 26, 2016. His embedding for this game became closer to those of other players who also did not participate.

6.3 Duplicate Ratios of Extracted Relations

As Puduppully et al. (2019a) pointed out, a generated summary may mention the same relation multiple times. Such duplicated relations are not favorable in terms of the brevity of text.

Figure 5 shows the ratios of the generated summaries with duplicate mentions of relations in the development data. While the models by Wiseman et al. (2017) and Puduppully et al. (2019a) respectively showed 36.0% and 15.8% as duplicate ratios, our model exhibited 4.2%. This suggests that our model dramatically suppressed generation of redundant relations. We speculate that the tracking model successfully memorized which input records have been selected in $\mathbf{h}_s^{\text{ENT}}$.

Fig. 5 Ratios of generated summaries with duplicate mention of relations. Each label represents number of duplicated relations within each document. While Wiseman et al. (2017)’s model exhibited 36.0% duplication and Puduppully et al. (2019a)’s model exhibited 15.8%, our model exhibited only 4.2%.

⁷ In the NBA, a player who accumulates a *double-digit* score in one of five categories (points, rebounds, assists, steals, and blocked shots) in a game, is regarded as a good player. If a player had a double in two of those five categories, it is referred to as *double-double*.

6.4 Qualitative Analysis of Output Examples

Figure 6 shows the generated examples from validation inputs with Puduppully et al. (2019a)’s model and our model. Whereas both generations seem to be fluent, the summary of Puduppully et al. (2019a)’s model includes erroneous relations colored in orange.

Specifically, the description about DERRICK ROSE’s relations, “15 points, four assists, three rounds and one steal in 33 minutes.”, is also used for other entities (e.g., JOHN HENSON and WILLY HERNAGOMEZ). This is because Puduppully et al. (2019a)’s model has no tracking module unlike our model, which mitigates redundant references and therefore rarely contains erroneous relations.

However, when complicated expressions such as parallel structures are used our model also generates erroneous relations as illustrated by the underlined sentences describing the two players who scored the same points. For example, “11-point efforts” is correct for COURTNEY LEE but not for DERRICK ROSE. As a future study, it is necessary to develop a method that can handle such complicated relations.

The **Milwaukee Bucks** defeated the **New York Knicks**, **105-104**, at Madison Square Garden on Wednesday evening. The **Bucks (18-16)** have been one of the hottest teams in the league, having won five of their last six games, and they have now won six of their last eight games. The **Knicks (16-19)** have now won six of their last six games, as they continue to battle for the eighth and final playoff spot in the Eastern Conference. **Giannis Antetokounmpo** led the way for Milwaukee, as he tallied **27** points, **13** rebounds, **four** assists, **three** blocked shots and **one** steal, in **39** minutes. **Jabari Parker** added **15** points, **four** rebounds, **three** assists, **one** steal and **one** block, and **6-of-8** from long range. **John Henson** added **two** points, **two** rebounds, **one** assist, **three** steals and **one** block. **John Henson** was the only other player to score in double digits for the Knicks, with **15** points, **four** assists, **three** rebounds and **one** steal, in **33** minutes. The Bucks were led by **Derrick Rose**, who tallied **15** points, **four** assists, **three** rebounds and **one** steal in **33** minutes. **Willy Hernangomez** started in place of Porzingis and finished with **15** points, **four** assists, **three** rebounds and **one** steal in **33** minutes. **Willy Hernangomez** started in place of Jose Calderon (knee) and responded with **one** rebound and **one** block. The Knicks were led by their starting backcourt of **Carmelo Anthony** and **Carmelo Anthony**, but combined for just **13** points on 5-of-16 shooting. The Bucks next head to Philadelphia to take on the Sixers on Friday night, while the Knicks remain home to face the Los Angeles Clippers on Wednesday.

(a) Puduppully et al. (2019a)

The **Milwaukee Bucks** defeated the **New York Knicks**, **105-104**, at Madison Square Garden on Saturday. The **Bucks (18-16)** checked in to Saturday’s contest with a well, outscoring the **Knicks (16-19)** by a margin of 39-19 in the first quarter. However, New York by just a 25-foot lead at the end of the first quarter, the Bucks were able to pull away, as they outscored the Knicks by a 59-46 margin into the second. 45 points in the third quarter to seal the win for New York with the rest of the starters to seal the win. The Knicks were led by **Giannis Antetokounmpo**, who tallied a game-high **27** points, to go along with **13** rebounds, **four** assists, **three** blocks and a steal. The game was a crucial night for the Bucks’ starting five, as the duo was the most effective shooters, as they posted Milwaukee to go on a pair of low low-wise (Carmelo Anthony) and Malcolm Brogdon. **Anthony** added **11** rebounds, **seven** assists and **two** steals to his team-high scoring total. **Jabari Parker** was right behind him with **15** points, **four** rebounds, **three** assists and a block. **Greg Monroe** was next with a bench-leading **18** points, along with **nine** rebounds, **four** assists and **three** steals. **Brogdon** posted **12** points, **eight** assists, **six** rebounds and a steal. **Derrick Rose** and **Courtney Lee** were next with a pair of **{11 / 11}** -point efforts. **Rose** also supplied **four** assists and **three** rebounds, while **Lee** complemented his scoring with **three** assists, a rebound and a steal. **John Henson** and **Mirza Teletovic** were next with a pair of **{two / two}** -point efforts. **Teletovic** also registered **13** points, and he added a rebound and an assist. **Jason Terry** supplied **eight** points, **three** rebounds and a pair of steals. The Cavs remain in last place in the Eastern Conference’s Atlantic Division. They now head home to face the Toronto Raptors on Saturday night.

(b) Our model

Fig. 6 Example summaries generated with Puduppully et al. (2019a)’s model (left) and our model (right). Names in bold face are salient entities. Blue numbers are correct relations derived from input data records but are not observed in reference summary. Orange numbers are incorrect relations. Green numbers are correct relations mentioned in reference summary.

(a) Transition of probabilities of entity $p(E_t = e)$. (b) Transition of probabilities of attribute $p(A_t = a)$.**Fig. 7** Probability sequences of selecting entities and attributes.

We also show the sequences of probabilities of selecting an entity $p(E_t = e_t)$ and an attribute $p(A_t = a_t)$ in Figure 7. The model first refers to overall information of the game, and then starts to look for salient players in the game. Once the model selects a player, it consistently selects the same player until it finishes mentioning the important information of this player. After finishing mentioning them, the model almost never refers to that entity. This suggests that the model successfully learns to guide the consistent entity reference, and mitigate the saliency of the entity mentioned so far.

6.5 Use of Writer Information

We first look at the results of an extension of Puduppully et al. (2019a)’s model with writer information w in Table 5. By adding w to content planning (stage 1), the method obtained improvements in CS (37.16 to 45.58), CO (16.67 to 21.26), and BLEU score (14.82 to 18.01). By adding w to the component for surface realization (stage 2), the method obtained an improvement in BLEU score (14.82 to 16.57), while the effects on the other metrics were not very significant. By adding w to both stages, the method scored the highest BLEU, while the other metrics were

Table 5 Effects of writer information

Method	RG		CS			CO	BLEU
	#	P%	P%	R%	F1%	DLD%	
NCP	34.05	82.55	32.30	43.74	37.16	16.67	14.82
+ w in stage 1	30.26	85.54	42.33	49.38	45.58	21.26	18.01
+ w in stage 2	32.42	83.35	33.28	42.92	37.49	16.73	16.57
+ w in stage 1 & 2	28.16	84.09	43.63	47.75	45.60	21.96	18.57
PROPOSED	31.66	91.98	40.44	46.63	43.31	21.56	15.74
+ w	29.44	93.32	51.76	55.21	53.42	24.97	20.62

w indicates that WRITER embeddings are used. Numbers in bold are the largest among the variants of each method.

not very different from those obtained by adding w to stage 1. This result suggests that writer information contributes to both content planning and surface realization when it is properly used, and improvements of content planning lead to much better performance in surface realization.

Our model showed improvements in all metrics and showed the best performance by incorporating writer information w . As discussed in Section 4.5, w is supposed to affect both content planning and surface realization. Our experimental result is consistent with the discussion.

We show generated examples of the PROPOSED model with and without writer information in Table 6. The summary generated by the PROPOSED model with writer information shares more similar vocabulary usage with the GOLD summary than the summary generated by the PROPOSED model without writer information.

In addition to the text style, the writer information is useful for choosing similar relations with the GOLD summary. Although the PROPOSED model without writer information mentions a bigger number of relations by generating longer summary, the PROPOSED model with writer information mentions a similar number of relations with the GOLD summary. Specifically, the GOLD summary contains the 31 relations and summaries generated by the PROPOSED model with or without writer information contain 46, 34 relations, and shares 20, 24 correct relations with relations in the GOLD summary respectively.

7 Conclusion

In this research, we proposed a new data-to-text model that produces a summary text while tracking the salient information. As a result, our model outperformed the existing models in all evaluation measures on the ROTOWIRE-MODIFIED dataset. We also explored the effects of

Table 6 Example summaries and their BLEU scores on the sample of the development set

	SUMMARY	BLEU
GOLD	<p>The Denver Nuggets defeated the host Orlando Magic, 121-113, at Amway Center on Saturday. In a game between two struggling teams, Orlando could n't quite keep up in their fourth game in five nights. That was clearly evident late, as the Nuggets ended the game on a 17-5 run to pull off the comeback. While the defense struggled all night, Denver held Orlando to just 18 points in the fourth quarter, while scoring 29 points on their end. The Nuggets offense was on point, as they shot 57 percent from the field and recorded 25 assists in the victory. Rebounding was key as well, with Denver winning that battle, 44-31. Denver (9-15) rode one of their best offensive performances here to an impressive road win. While no player on the Nuggets had more than 21 points, they had six players score at least 14. Danilo Gallinari led the team in scoring, tallying 21 points, six rebounds and three assists. Jusuf Nurkic scored 17 points and grabbed eight boards on 8-of-11 shooting in just 20 minutes of action. Kenneth Faried was huge off the bench, dropping 19 points and 10 rebounds on 9-of-10 from the field. Nikola Jokic and Juancho Hernangomez each scored {14/14} a piece off the bench. On deck for the Nuggets is a road matchup against the Dallas Mavericks on Monday.. Orlando (10-15) had to play this game without Nikola Vucevic and it clearly hurt their defense. Evan Fournier did all he could against his former team, as he led the Magic with 24 points, four rebounds and six assists. Serge Ibaka was n't far behind, accumulating 22 points, four rebounds and four blocks in the defeat. Jeff Green was questionable coming in to this game with an ankle injury, but provided 17 points off the bench. Orlando 's next game is on Tuesday in a road matchup against the Atlanta Hawks.</p>	—
PROPOSED	<p>The Denver Nuggets defeated the host Orlando Magic, 121-113, at Amway Center on Wednesday. The Nuggets (9-15) checked in to Friday with only seven road wins in 23 tries per game and had a road win behind the Magic (10-15). However, their second-highest scorer in Kenneth Faried 's head-high 21 combined attempts from the T-Wolves. He went off to grab a win for two of the Nuggets ' points in the victory, despite shooting 57 percent from the field and 42 percent from three-point range. They did turn up on the defensive side of the ball, turning the ball over two times and a pair of rebounds of an impressive 19 points. Faried supplemented his total double-double on the night, tallying 10 boards and dishing out one assist. Danilo Gallinari and Will Barton were next with a pair of {16/16}-point efforts. Gallinari added six rebounds, and three assists, while the backcourt was also a perfect night for the starting shooting night. Nikola Jokic continued his strong play in reserve roles, supplying 14 points, six rebounds, six assists and a block. Jusuf Nurkic continued his stretch of strong play down low, posting 17 points, eight rebounds, one block and one block. Evan Fournier continued his recent stretch of strong play with 24 points, four rebounds, six assists and one steal. Serge Ibaka totaled 22 points, four rebounds, one steal and four blocks. Aaron Gordon provided 11 points, six boards, three assists and one steal. Bismack Biyombo tallied 12 points, six rebounds, four assists and one block. Barton added four rebounds and a block, while Juan Hernangomez and Jeff Green supplied matching {14/14}-point efforts, with the former also hauling in four boards and recording zero boards, and the latter posting four rebounds and one assist. The Nuggets head back home to tangle with the Philadelphia 76ers on Wednesday evening, while the Magic remain home to take on the Magic on Friday evening as well.</p>	22.03
PROPOSED + <i>w</i>	<p>The Denver Nuggets defeated the host Orlando Magic, 121-113, at Amway Center on Wednesday. The Nuggets were expected to win this game easily, but they were n't able to prevail with a win. In fact, the Nuggets led by 10 after the first quarter, outscoring Orlando, 33-14. Defense was key for the Nuggets, as they held the Magic to 49 percent from the field and 5-of-28 from three-point range. Los Angeles were able to pull out a huge rebound, as they held the Magic to just 18 points in the final quarter. The Nuggets (9-15) have been one of the best teams in the league right now and they simply could n't get enough of a rhythm to win. Danilo Gallinari led the team in scoring, as he tallied 21 points, six rebounds and three assists on 5-of-8 shooting. Kenneth Faried was second on the team, as he accrued 19 points and 10 rebounds on 9-of-10 shooting off the bench. Jusuf Nurkic was second on the team, with his 17 points and eight rebounds on 8-of-11 shooting. Juan Hernangomez was a nice spark off the bench, as he amassed 14 points, four rebounds and two assists. On deck for Denver is a home matchup against the Los Angeles Lakers on Wednesday. The Magic (10-15) have been one of the surprises of the season and a comeback performance from their worst team. Evan Fournier had his best game of the season, as he tallied 24 points and four rebounds on 9-of-19 shooting. Serge Ibaka was second on the team, as he accrued 22 points, four rebounds and four blocks. Aaron Gordon was a nice spark off the bench, providing 11 points, six rebounds and three assists. Orlando will look to keep rolling on Monday in a home matchup against the Cleveland Cavaliers.</p>	30.58

Blue numbers are correct relations derived from input data records but are not observed in reference summary. Orange numbers are incorrect relations. Green numbers are correct relations mentioned in reference summary.

incorporating writer information to data-to-text models. With writer information, our model successfully generated highest quality summaries that scored 20.62 points of BLEU score.

Acknowledgement

This paper is an extension of our manuscript presented by the 57th Annual Meeting of the Association for Computational Linguistics (Iso, Uehara, Ishigaki, Noji, Aramaki, Kobayashi, Miyao, Okazaki, and Takamura 2019) with more detailed analysis and experiments on new dataset. This paper is based on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO), JST PRESTO (Grant Number JPMJPR1655), and AIST-Tokyo Tech Real World Big-Data Computation Open Innovation Laboratory (RWBC-OIL). The authors would like to thank Enago (<https://www.enago.jp/>) for the English language review.

References

- Angeli, G., Liang, P., and Klein, D. (2010). “A Simple Domain-Independent Probabilistic Approach to Generation.” In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 502–512, Cambridge, MA. Association for Computational Linguistics.
- Aoki, T., Miyazawa, A., Ishigaki, T., Goshima, K., Aoki, K., Kobayashi, I., Takamura, H., and Miyao, Y. (2018). “Generating Market Comments Referring to External Resources.” In *Proceedings of the 11th International Conference on Natural Language Generation*, pp. 135–139.
- Bahdanau, D., Cho, K., and Bengio, Y. (2015). “Neural Machine Translation by Jointly Learning to Align and Translate.” In *Proceedings of the Third International Conference on Learning Representations*.
- Barzilay, R. and Lapata, M. (2005). “Collective Content Selection for Concept-to-Text Generation.” In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pp. 331–338.
- Bosselut, A., Levy, O., Holtzman, A., Ennis, C., Fox, D., and Choi, Y. (2018). “Simulating Action Dynamics with Neural Process Networks.” In *Proceedings of the 6th International Conference on Learning Representations*.
- Chen, D. L. and Mooney, R. J. (2008). “Learning to sportscast: a test of grounded language acquisition.” In *Proceedings of the 25th international conference on Machine learning*,

pp. 128–135.

- Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation.” In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1724–1734.
- Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling.” *arXiv preprint arXiv:1412.3555*.
- Clark, E., Ji, Y., and Smith, N. A. (2018). “Neural Text Generation in Stories Using Entity Representations as Context.” In *Proceedings of the 16th Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2250–2260.
- Glorot, X. and Bengio, Y. (2010). “Understanding the Difficulty of Training Deep Feedforward Neural Networks.” In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, pp. 249–256.
- Graves, A., Wayne, G., Reynolds, M., Harley, T., Danihelka, I., Grabska-Barwińska, A., Colmenarejo, S. G., Grefenstette, E., Ramalho, T., Agapiou, J., et al. (2016). “Hybrid Computing using a Neural Network with Dynamic External Memory.” *Nature*, **538** (7626), p. 471.
- Gu, J., Lu, Z., Li, H., and Li, V. O. (2016). “Incorporating Copying Mechanism in Sequence-to-Sequence Learning.” In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Vol. 1, pp. 1631–1640.
- Gulcehre, C., Ahn, S., Nallapati, R., Zhou, B., and Bengio, Y. (2016). “Pointing the Unknown Words.” In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pp. 140–149.
- Henaff, M., Weston, J., Szlam, A., Bordes, A., and LeCun, Y. (2017). “Tracking the World State with Recurrent Entity Networks.” In *Proceedings of the 5th International Conference on Learning Representations*.
- Hoang, L., Wiseman, S., and Rush, A. (2018). “Entity Tracking Improves Cloze-style Reading Comprehension.” In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1049–1055.
- Iso, H., Uehara, Y., Ishigaki, T., Noji, H., Aramaki, E., Kobayashi, I., Miyao, Y., Okazaki, N., and Takamura, H. (2019). “Learning to Select, Track, and Generate for Data-to-Text.” In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 2102–2113, Florence, Italy.
- Ji, Y., Tan, C., Martschat, S., Choi, Y., and Smith, N. A. (2017). “Dynamic Entity Represen-

- tations in Neural Language Models.” In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1830–1839.
- Kiddon, C., Zettlemoyer, L., and Choi, Y. (2016). “Globally coherent text generation with neural checklist models.” In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 329–339.
- Kobayashi, S., Tian, R., Okazaki, N., and Inui, K. (2016). “Dynamic Entity Representation with Max-pooling Improves Machine Reading.” In *Proceedings of the 15th Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 850–855.
- Lebret, R., Grangier, D., and Auli, M. (2016). “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.
- Li, L. and Wan, X. (2018). “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.
- Liang, P., Jordan, M. I., and Klein, D. (2009). “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.
- Liu, T., Wang, K., Sha, L., Chang, B., and Sui, Z. (2018). “Table-to-text Generation by Structure-aware Seq2seq Learning.” In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*.
- Luong, T., Pham, H., and Manning, C. D. (2015). “Effective Approaches to Attention-based Neural Machine Translation.” In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1412–1421.
- Mei, H., Bansal, M., and Walter, M. R. (2016). “What to Talk about and How? Selective Generation using LSTMs with Coarse-to-Fine Alignment.” In *Proceedings of the 15th Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 720–730.
- Murakami, S., Watanabe, A., Miyazawa, A., Goshima, K., Yanase, T., Takamura, H., and Miyao, Y. (2017). “Learning to Generate Market Comments from Stock Prices.” In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pp. 1374–1384.
- Neubig, G., Dyer, C., Goldberg, Y., Matthews, A., Ammar, W., Anastasopoulos, A., Ballesteros, M., Chiang, D., Clothiaux, D., Cohn, T., et al. (2017). “DyNet: The Dynamic Neural

- Network Toolkit.” *arXiv preprint arXiv:1701.03980*.
- Nie, F., Wang, J., Yao, J.-G., Pan, R., and Lin, C.-Y. (2018). “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.
- Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). “BLEU: A Method for Automatic Evaluation of Machine Translation.” In *Proceedings of the 40th annual meeting on association for computational linguistics*, pp. 311–318.
- Pearson, K. (1901). “On Lines and Planes of Closest Fit to Systems of Points in Space.” *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, **2** (11), pp. 559–572.
- Puduppully, R., Dong, L., and Lapata, M. (2019a). “Data-to-Text Generation with Content Selection and Planning.” In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pp. 6908–6915.
- Puduppully, R., Dong, L., and Lapata, M. (2019b). “Data-to-text Generation with Entity Modeling.” In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2023–2035. Association for Computational Linguistics.
- Rebuffel, C., Soulier, L., Scoutheeten, G., and Gallinari, P. (2020). “A Hierarchical Model for Data-to-Text Generation.” In *European Conference on Information Retrieval*, pp. 65–80. Springer.
- Reddi, S. J., Kale, S., and Kumar, S. (2018). “On the Convergence of Adam and Beyond.” In *Proceedings of the 6th International Conference on Learning Representations*.
- Sha, L., Mou, L., Liu, T., Poupart, P., Li, S., Chang, B., and Sui, Z. (2018). “Order-planning Neural Text Generation from Structured Data.” In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pp. 5414–5421.
- Sukhbaatar, S., Szlam, A., Weston, J., and Fergus, R. (2015). “End-to-end Memory Networks.” In *Advances in Neural Information Processing Systems*, pp. 2440–2448.
- Sutskever, I., Vinyals, O., and Le, Q. V. (2014). “Sequence to Sequence Learning with Neural Networks.” In *Advances in Neural Information Processing Systems*, pp. 3104–3112.
- Tanaka-Ishii, K., Hasida, K., and Noda, I. (1998). “Reactive Content Selection in the Generation of Real-time Soccer Commentary.” In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, pp. 1282–1288.
- Taniguchi, Y., Feng, Y., Takamura, H., and Okumura, M. (2019). “Generating Live Soccer-Match Commentary from Play Data.” In *Proceedings of the 33rd AAAI Conference on Artificial*

Intelligence, pp. 7096–7103.

- Tu, Z., Liu, Y., Lu, Z., Liu, X., and Li, H. (2017). “Context Gates for Neural Machine Translation.” *Transactions of the Association for Computational Linguistics*, **5**, pp. 87–99.
- Wang, H. (2019). “Revisiting Challenges in Data-to-Text Generation with Fact Grounding.” In *Proceedings of the 12th International Conference on Natural Language Generation*, pp. 311–322, Tokyo, Japan. Association for Computational Linguistics.
- Weston, J., Chopra, S., and Bordes, A. (2015). “Memory Networks.” In *Proceedings of the 3rd International Conference on Learning Representations*.
- Wiseman, S., Shieber, S., and Rush, A. (2017). “Challenges in Data-to-Document Generation.” In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2253–2263.
- Yang, Z., Blunsom, P., Dyer, C., and Ling, W. (2017). “Reference-Aware Language Models.” In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1850–1859.

Hayate Iso: received his B.S. degree from Osaka University in 2016 in Mathematical Science and M.S. degree from Nara Institute of Science and Technology (NAIST) in 2017, in information science. He is currently a Ph.D. student at NAIST. His research interests include natural language generation and social media analysis.

Yui Uehara: received her M.S. degree from Japan Advanced Institute of Science and Technology in 2019. Her current research interests are computational music research and natural language processing.

Tatsuya Ishigaki: received his Ph.D. degree from Tokyo Institute of Technology, and is currently a postdoctoral researcher at Artificial Intelligence Research Center, AIST.

Hiroshi Noji: received his Ph.D. from the Graduate University for Advanced Studies. He was an assistant professor at Nara Institute of Science and Technology from 2016 to 2018, and is currently a researcher at Artificial Intelligence Research Center, AIST. His current research interests are syntactic parsing as well as modeling of human language processing such as language acquisition.

Eiji Aramaki: received his Ph.D. from the University of Tokyo in 2005, and is currently an associate professor at Nara Institute of Science and Technology.

His current research interest is in natural language processing for the medical field.

Ichiro Kobayashi: received his Ph.D. from Tokyo Institute of Technology, and is currently a professor at Ochanomizu University. His current research interests are in language processing and intelligent information processing.

Yusuke Miyao: received his Ph.D. from the University of Tokyo in 2006, and is currently a professor at the University of Tokyo. His current research interests are syntactic parsing and its applications.

Naoaki Okazaki: received his Ph.D. from the University of Tokyo in 2007, and is currently a professor at Tokyo Institute of Technology. His current research interests are natural language processing, text mining, and machine learning.

Hiroya Takamura: received his Ph.D. from Nara Institute of Science and Technology, and is currently a professor at Tokyo Institute of Technology and a team leader at Knowledge and Information Research Team of Advanced Industrial Science and Technology. His current research interest is computational linguistics.

(Received February 1, 2020)

(Revised May 1, 2020)

(Accepted June 17, 2020)