Data-to-text Generation with Entity Modeling

Ratish Puduppully and Li Dong and Mirella Lapata
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh EH8 9AB
r.puduppully@sms.ed.ac.uk li.dong@ed.ac.uk mlap@inf.ed.ac.uk

Abstract

Recent approaches to data-to-text generation have shown great promise thanks to the use of large-scale datasets and the application of neural network architectures which are trained end-to-end. These models rely on representation learning to select content appropriately, structure it coherently, and verbalize it grammatically, treating entities as nothing more than vocabulary tokens. In this work we propose an entity-centric neural architecture for data-to-text generation. Our model creates entity-specific representations which are dynamically updated. Text is generated conditioned on the data input and entity memory representations using hierarchical attention at each time step. We present experiments on the ROTOWIRE benchmark and a (five times larger) new dataset on the baseball domain which we create. Our results show that the proposed model outperforms competitive baselines in automatic and human evaluation.

1 Introduction

Data-to-text generation is the task of generating textual output from non-linguistic input (Reiter and Dale, 1997; Gatt and Krahmer, 2018). The input may take on several guises including tables of records, simulations of physical systems, spreadsheets, and so on. As an example, Figure 1 shows (in a table format) the scoring summary of a major league baseball (MLB) game, a play-by-play summary with details of the most important events in the game recorded chronologically (i.e., in which play), and a human-written summary.

Modern approaches to data-to-text generation have shown great promise (Lebret et al., 2016; Mei et al., 2016; Perez-Beltrachini and Lapata, 2018; Puduppully et al., 2019; Wiseman et al., 2017) thanks to the use of large-scale datasets and neural network models which are trained end-to-end based on the very successful encoder-decoder architecture (Bahdanau et al., 2015). In contrast to traditional methods which typically implement pipeline-style architectures (Reiter and Dale, 2000) with modules devoted to individual generation components (e.g., content selection or lexical choice), neural models have no special-purpose mechanisms for ensuring how to best generate a text. They simply rely on representation learning to select content appropriately, structure it coherently, and verbalize it grammatically.

In this paper we are interested in the generation of descriptive texts such as the game summary shown in Figure 1. Descriptive texts are often characterized as “entity coherent” which means that their coherence is based on the way entities (also known as domain objects or concepts) are introduced and discussed in the discourse (Karamanis et al., 2004). Without knowing anything about baseball or how game summaries are typically written, a glance at the text in Figure 1 reveals that it is about a few entities, namely players who had an important part in the game (e.g., Brad Keller, Hunter Dozier) and their respective teams (e.g., Orioles, Royals). The prominent role of entities in achieving discourse coherence has been long recognized within the linguistic and cognitive science literature (Kuno, 1972; Chafe, 1976; Halliday and Hasan, 1976; Karttunen, 1976; Clark and Haviland, 1977; Prince, 1981), with Centering Theory (Grosz et al., 1995) being most prominent at formalizing how entities are linguistically realized and distributed in texts.

In this work we propose an entity-centric neural architecture for data-to-text generation. Instead of treating entities as ordinary tokens, we create entity-specific representations (i.e., for players and teams) which are dynamically updated as text is...
ing content plans into account (Puduppully et al., 2019). Our results show that modeling entities explicitly is beneficial and leads to output which is not only more coherent but also more concise and grammatical across both datasets.

Our contributions in this work are three-fold: a novel entity-aware model for data-to-text generation which is linguistically motivated, yet resource lean (no preprocessing is required, e.g., to extract document plans); a new dataset for data-to-text generation which we hope will encourage further work in this area; a comprehensive evaluation and comparison study which highlights the merits and shortcomings of various recently proposed data-to-text generation models on two datasets.

2 Related Work

The sports domain has attracted considerable attention since the early days of generation systems (Robin, 1994; Tanaka-Ishii et al., 1998). Likewise, a variety of coherence theories have been developed over the years (e.g., Mann and Thomson 1988; Grosz et al. 1995) and their principles have found application in many symbolic text generation systems (e.g., Scott and de Souza 1990; Kibble and Power 2004). Modeling entities and their communicative actions has also been shown to improve system output in interactive storytelling

| BATTER | PITCHER | SCORER | EVENT | TEAM | INN | RUNS |
|--------|---------|--------|-------|------|-----|------|
| C. Mullins | B. Keller | Home run | Orioles | 1 | 1 |
| H. Dozier | A. Cashner | W. Merrifield | Grounded into DP | Royals | 1 | 1 |
| W. Merrifield | A. Cashner | B. Goodwin | Sac fly | Royals | 4 | 2 |
| H. Dozier | A. Cashner | - | Home run | Royals | 4 | 3 |

Figure 1: MLB statistics tables and game summary. The tables summarize the performance of the two teams and of individual team members who played as batters and pitchers as well as the most important events (and their actors) in each play. Recurring entities in the summary are boldfaced and colorcoded, singletons are shown in black.
(Cavazza et al., 2002; Cavazza and Charles, 2005) and dialogue generation (Walker et al., 2011).

More recently, the benefits of modeling entities explicitly have been demonstrated in various tasks and neural network models. Ji et al. (2017) make use of dynamic entity representations for language modeling. And Clark et al. (2018) extend this work by adding entity context as input to the decoder. Both approaches condition on a single entity at a time, while we dynamically represent and condition on multiple entities in parallel. Kiddon et al. (2016) make use of fixed entity representations to improve the coverage and coherence of the output for recipe generation. Bosselut et al. (2018) model actions and their effects on entities in machine reading. Henaff et al. (2017) make use of dynamic entity representations in machine reading. Entity representations are scored against a query vector to directly predict an output class or combined as a weighted sum followed by softmax over the vocabulary. We make use of a similar entity representation model, extend it with hierarchical attention and apply it to data-to text generation. The hierarchical attention mechanism was first introduced in Yang et al. (2016) as a way of learning document-level representations. We apply attention over records and subsequently over entity memories.

Several models have been proposed in the last few years for data-to-text generation (Mei et al. 2016; Lebret et al. 2016; Wiseman et al. 2017, inter alia) based on the very successful encoder-decoder architecture (Bahdanau et al., 2015). Various attempts have also been made to improve these models, e.g., by adding content selection (Perez-Beltrachini and Lapata, 2018) and content planning (Puduppully et al., 2019) mechanisms. However, we are not aware of any prior work in this area which explicitly handles entities and their generation in discourse context.

3 Background: Encoder-Decoder with Conditional Copy

The input to our model is a table of records (see Figure 1). Records in turn have features, represented as \(r_{j,l}^1 \) where \(L\) is the number of features in each record. Examples of features are values (\(r_{j,1};\) e.g., 8.0, Baltimore) or entities (\(r_{j,2};\) e.g., Orioles, C. Mullins). The model output \(y\) is a document containing words \(y = y_1 \cdots y_{|y|}\) where \(|y|\) is the document length. Following previous work (Wiseman et al., 2017; Puduppully et al., 2019), we construct embeddings of features, and then obtain a vector representation \(r_j\) of each record by using a multilayer perceptron:

\[
    r_j = \text{ReLU}(W_r[r_{j,1};r_{j,2};\ldots;r_{j,L}] + b_r) \tag{1}
\]

where \(\text{ReLU}\) is the rectifier activation function, \(W_r \in \mathbb{R}^{n \times nL}, b_r \in \mathbb{R}^n\) are parameters and \([;]\) indicates vector concatenation.

Let \(\{e_j\}_{j=1}^{|r|}\) denote the output of the encoder. We use an LSTM decoder to compute the probability of each target word, conditioned on previously generated words, and on \(e_j\). In the case of ROTOwire, we follow previous work (Wiseman et al., 2017; Puduppully et al., 2019) and consider \(e_j = r_j\). The first hidden state of the decoder is initialized by the average of the record vectors, \(\text{avg}((e_j)_{j=1}^{|r|})\).

In the case of MLB, information encoded in play-by-play is sequential. Recall, that it documents the most important events in a game in chronological order. To account for this, we encode MLB records into \(\{e_j\}_{j=1}^{|r|}\) with a bidirectional LSTM. We impose an ordering on records in the box score (i.e., home team followed by away team) which is in turn followed by play-by-play where records are naturally ordered by time. The decoder is initialized with the concatenation of the hidden states of the final step of the encoder.

At time step \(t\), the input to the decoder LSTM is the embedding of the previously predicted word \(y_{t-1}\). Let \(d_t\) denote the hidden state of the \(t\)-th LSTM unit. We compute attention scores \(\alpha_{t,j}\) over the encoder output \(e_j\) and obtain dynamic context vector \(q_t\) as the weighted sum of the hidden states of the input:

\[
    \alpha_{t,j} \propto \exp(d_t^\top W_a e_j) \quad q_t = \sum_j \alpha_{t,j} e_j \quad d^{\text{att}}_t = \tanh(W_c[d_t; q_t]) \tag{2}
\]

where \(W_a \in \mathbb{R}^{n \times n}, \sum_j \alpha_{t,j} = 1, W_c \in \mathbb{R}^{n \times 2n}\), and \(d^{\text{att}}_t\) is the attention vector.

The probability of output text \(y\) conditioned on the input table \(r\) is modeled as:

\[
    p_{\text{gen}}(y_{t}\mid y_{<t},r) = \text{softmax}(y_t \cdot W_y d^{\text{att}}_t + b_y) \tag{3}
\]
where $W_y \in \mathbb{R}^{|V_y| \times n}$, $b_y \in \mathbb{R}^{|V_y|}$ are parameters and $|V_y|$ is the output vocabulary size.

We further augment the decoder with a copy mechanism i.e., the ability to copy values from the input; copy implies $y_t = r_{j,1}$ for some $t$ and $j$ (e.g., Royals, Orioles, 9, 2 in the summary in Figure 1 are copied from $r$). We use the conditional copy method proposed in Gulcehre et al. (2016) where a binary variable is introduced as a switch gate to indicate whether $y_t$ is copied or not.

4 Entity Memory and Hierarchical Attention

We extend the basic model from Section 3 with entity memory and hierarchical attention. Figure 2 provides a schematic overview of our architecture.

4.1 Entity Memory

In order to render the model entity-aware, we compute $x_k$ as an average of record representation for each unique entity $k$ (i.e., one of $r_{j,2}$ values):

$$x_k = \sum_j (\mathbb{I}[r_{j,2} = k] r_j) / \sum_j \mathbb{I}[r_{j,2} = k]$$

(4)

where $\mathbb{I}[x] = 1$ if $x$ is true, and 0 otherwise.

We initialize $u_{t=-1,k}$, the memory representation of an entity at time $t = -1$, as:

$$u_{t=-1,k} = W_i x_k$$

(5)

where $u_{t=-1,k} \in \mathbb{R}^p$ and $W_i \in \mathbb{R}^{p \times n}$.

To capture the fact that discourse in descriptive texts may shift from one entity to the next, e.g., some entities may be salient in the beginning of the game summary (see Brad Kelly in the text in Figure 1), others only towards the end (see Dozier in Figure 1), and a few throughout (e.g., references to teams), we update entity representations at each time step during decoding. We use gate $\gamma_t$ to indicate whether there should be an update in the entity representation:

$$\gamma_t = \sigma(W_d d_t + b_d)$$

(6)

where $t > 0$, $\sigma$ is the sigmoid function, $W_d \in \mathbb{R}^{p \times p}$, and $b_d \in \mathbb{R}^p$.

We also compute $\delta_{t,k}$, the extent to which the entity representation should change, and $\tilde{u}_{t,k}$, the memory of the candidate entity:

$$\delta_{t,k} = \gamma_t \odot (W_e d_t + b_e + W_f u_{t-1,k} + b_f)$$

(7)

$$\tilde{u}_{t,k} = W_g d_t$$

(8)

where $\odot$ denotes element-wise multiplication, $W_e \in \mathbb{R}^{p \times n}$, $W_f \in \mathbb{R}^{p \times p}$, $b_e, b_f \in \mathbb{R}^p$, and $\gamma_t, \delta_{t,k} \in [0, 1]^p$ (see block A in Figure 2).

An element in gate $\gamma_t$ will have value approaching 1 if an update in any $u_{t-1,k}$ is required. The value of an element in gate $\delta_{t,k}$ will approach 1 if the corresponding value of the element in $u_{t-1,k}$ changes. Equation (9) computes the update in entity memory as an interpolation over the gated representation of the previous value of the entity.
memory and the candidate entity memory:

\[ u_{t,k} = (1 - \delta_{t,k}) \odot u_{t-1,k} + \delta_{t,k} \odot \tilde{u}_{t,k} \] (9)

where \( u_{t,k} \) represents entity \( k \) at time \( t \).

Previous work (Henaff et al., 2017; Ji et al., 2017; Clark et al., 2018) employs a normalization term over \( u_{t,k} \). We empirically found that normalization hurts performance and hence did not include it in our model.

### 4.2 Hierarchical Attention

We hypothesize that our generator should first focus on entities (e.g., the main players and their teams) and then on the records corresponding to these entities (e.g., player performance in the game). Our model implements this view of text generation via a hierarchical attention mechanism which we explain below. We also expect that focusing on entities first should improve the precision of the texts we generate as the entity distribution of the theses entities (e.g., player performance in the game) will constrain the probability distribution of records corresponding to each entity.

To better understand the hierarchical attention mechanism, we can view the encoder output \( e_j \) as a 2-dimensional array \( g_{k,z} \), where \( k \in [1, K] \) represents entities and \( z \in [1, Z] \) represents records of entities and there is a one-to-one correspondence between positions \( j \) and \( k, z \). We compute attention over \( g_{k,z} \), the encoder output, as:

\[ \alpha_{t,k,z} \propto \exp(d_j^T W_a g_{k,z}) \] (10)

where \( W_a \in \mathbb{R}^{n \times n}, \sum_z \alpha_{t,k,z} = 1 \) (see block B in Figure 2). We compute the entity context as:

\[ s_{t,k} = \sum_z \alpha_{t,k,z} g_{k,z} \] (11)

while attention over entity vectors \( u_{t,k} \) is:

\[ \Psi_{t,k} \propto \exp(d_j^T W_b u_{t,k}) \] (12)

with \( W_b \in \mathbb{R}^{n \times P}, \sum_k \Psi_{t,k} = 1 \). And the encoder context \( q_t \) (see block C in Figure 2) is computed as follows:

\[ q_t = \sum_k \Psi_{t,k} s_{t,k} \] (13)

We feed \( q_t \) into Equation (2) and compute \( p_{gen}(y_t|y_{<t}, r) \), the probability of generating output text \( y_t \) conditioned on records \( r \), as shown in Equation (3).

---

### Table 1: Vocabulary size, number of tokens, number of instances (i.e., record-summary pairs), average summary length, number of record types and average number of records in ROTOWire and MLB datasets.

|               | ROTOWire | MLB |
|---------------|----------|-----|
| Vocab Size    | 11.3K    | 38.9K |
| # Tokens      | 1.5M     | 14.3M|
| # Instances   | 4.9K     | 26.3K|
| Avg Length    | 337.1    | 542.05|
| # Record Types| 39       | 53   |
| Avg Records   | 628      | 565  |

We experimented with feeding \( \sum_k \Psi_{t,k} u_{t,k} \) as input context along the lines of Clark et al. (2018); however, results on the development dataset degraded performance, and we did not pursue this approach further.

### 5 Training and Inference

Our training objective maximizes the log likelihood of output text given an input table of records:

\[ \max_{(r,y) \in D} \sum \log p(y|r) \]

where \( D \) is the training set consisting of pairs of record tables and output game summaries. During inference, we make use of beam search to approximately obtain the best output \( \hat{y} \) among candidate outputs \( y' \):

\[ \hat{y} = \arg \max_{y'} p(y'|r) \]

### 6 Experimental Setup

**Data** We performed experiments on two datasets. The first one is ROTOWire (Wiseman et al., 2017) which contains NBA basketball game statistics matched with human-written summaries. In addition, we created MLB, a new dataset which contains baseball statistics and corresponding human-authored summaries obtained from the ESPN website. Basic statistics on the two datasets are given in Table 1. As can be seen, MLB is approximately five times larger than ROTOWire, with richer vocabulary and longer summaries. For ROTOWire, we used the official training, development, and test splits of 3,398/727/728 instances. Analogously, for MLB we created a split of 22,821/1,739/1,744 instances. Game summaries in MLB were tokenized

---

2http://www.espn.com/mlb/recap?gameId={gameid}
using nltk and hyphenated words were separated. Sentences containing quotes were removed as they included opinions and non-factual statements unrelated to the input tables. Sometimes MLB summaries contain a “Game notes” section with incidental information which was also removed.

For MLB, the value of $L$ in Equation (1) is 6, and for ROTOWire it is 4. The first four features are similar in both datasets and include value $(r_{j,1}; \text{e.g., 8.0, Baltimore})$, entity $(r_{j,2}; \text{e.g., Orioles, C. Mullins})$, record type $(r_{j,3}; \text{e.g., RBI, R,H})$ and whether a player is on the home- or away- team $(r_{j,4})$. MLB has two additional features which include the inning of play $(r_{j,5}; \text{e.g., 9, 7, and -1})$ for records in the box score, and play index, a unique play identifier for a set of records in a play $(r_{j,6}; \text{e.g., 0, 10, and -1})$ for records in the box score.

**Information Extraction** For automatic evaluation, we make use of the Information Extraction (IE) approach proposed in Wiseman et al. (2017). The idea is to use a fairly accurate IE tool to extract relations from gold summaries and model summaries and then quantify the extent to which the extracted relations align or diverge (see Section 7 for the specific metrics we use).

The IE system first identifies candidate entities (i.e., players, teams) and values (i.e., numbers), and given an “entity, value” pair it predicts the type of relation. For example, in ROTOWire, the relation for the pair “Kobe Bryant, 40” is PTS. Training data for the IE system is obtained automatically by matching entity-value pairs from summary sentences against record types. The IE system has an ensemble architecture which combines convolutional and bidirectional LSTM models.

We reused the updated IE models from Puduppully et al. (2019) for ROTOWire$^3$ and trained our own IE system for MLB. Box and line scores in MLB are identical in format to ROTOWire and pose no particular problems to the IE system. However, it is difficult to extract information from play-by-play and match it against the input tables. Consider the sentences *Ryan O’Hearn homered* or *Keller gave up a home run* from Figure 1 where we can identify entities (Ryan O’Hearn, Keller) and record types (home-run-batter, home-run-pitcher) but no specific values. We created a dummy value of -1 for such cases and the IE system was trained to predict the record type of entity value pairs such as (Ryan O’Hearn, -1) or (Keller, -1). Moreover, the IE system does not capture attributes such as inning and team scores in play-by-play as it is difficult to deterministically match these against corresponding spans in text. The IE system thus would not be able to identify any records in the snippet *tied 1–1 in the fourth*. On MLB, the system achieved 83.4% precision and 66.7% recall (on held out data). We note that designing a highly accurate IE module for MLB is in itself a research challenge and outside the scope of this paper.

In order to compare our model against Puduppully et al. (2019), we must have access to content plans which we extracted from ROTOWire and MLB by running the IE tool on gold summaries (training set). We expect the relatively low IE recall on MLB to disadvantage their model which relies on accurate content plans.

**Training Configuration** Model hyperparameters were tuned on the development set. We used the Adagrad optimizer (Duchi et al., 2011) with an initial learning rate of 0.15, decayed by 0.97 for every epoch after the 4th epoch. We used truncated BPTT (Williams and Peng, 1990) of length 100 and made use of input feeding (Luong et al., 2015). We summarize the hyperparameters of the ROTOWire and MLB models in the Appendix. All models were implemented on a fork of OpenNMT-py (Klein et al., 2017).

**System Comparison** We compared our entity model against the following systems:

- **TEMPL** is a template-based generator; we reused TEMPL from Wiseman et al. (2017) for ROTOWire and created a new system for MLB. The latter consists of an opening sentence about the two teams playing the game. It then describes statistics of pitchers (innings pitched, runs and hits given etc.) followed by a description of play-by-play (home run, single, double, triple etc.).

- **ED+CC** is the encoder-decoder model with conditional copy from Section 3 and the best performing system in Wiseman et al. (2017).

- **NCP+CC** is the best performing system in Puduppully et al. (2019); it generates content plans by making use of pointer networks (Vinyals et al., 2015) to point to the input $e_j$; the resultant content plans are then encoded using a BiLSTM followed by an LSTM decoder with an attention and copy mechanism.

---

$^3$https://github.com/ratishsp/data2text-1/
Table 2: Evaluation on ROTOWire (RW) and MLB test sets using relation generation (RG) count and precision, content selection (CS) precision and recall, content ordering (CO) in normalized Damerau-Levenshtein distance, and BLEU.

Table 3: Ablation results on ROTOWire (RW) and MLB development set using relation generation (RG) count and precision, content selection (CS) precision and recall, content ordering (CO) in normalized Damerau-Levenshtein distance, and BLEU.

7 Results

Automatic Evaluation We first discuss the results of automatic evaluation using the metrics defined in Wiseman et al. (2017). Let $\hat{y}$ be the gold output and $y$ the model output. Relation Generation measures how factual $y$ is compared to input $r$. Specifically, it measures the precision and number of relations extracted from $y$ which are also found in $r$. Content Selection measures the precision and recall of relations between $\hat{y}$ and $y$. Content Ordering measures the Damerau-Levenshtein distance between relations in $y$ and relations in $\hat{y}$. In addition, we also report BLEU (Papineni et al., 2002) with the gold summaries as reference.

Table 2 (top) summarizes our results on the ROTOWire test set (results on the development set are available in the Appendix). We report results for our dynamic entity memory model (ENT), the best system of Wiseman et al. (2017) (WS-2017) which is an encoder-decoder model with conditional copy, and NCP+CC (Puduppully et al., 2019). We see that ENT achieves scores comparable to NCP+CC, but performs better on the metrics of RG precision, CS precision, and CO. ENT achieves substantially higher scores in CS precision compared to WS-2017 and NCP+CC, without any planning component; CS recall is worse for ENT compared to NCP+CC mainly because the latter model is trained to first create a content plan with good coverage of what to say.

Table 2 (bottom) also presents our results on MLB (test set). Note that ED+CC is a reimplementation of Wiseman et al.’s (2017) encoder-decoder model (with conditional copy) on MLB. We see that ENT achieves highest BLEU amongst all models and highest CS recall and RG count amongst neural models. The RG precision of ENT is lower than ED+CC. Inspection of model output revealed that on MLB, ED+CC tends to focus on one or two players getting most of the facts about them right, whereas ENT sometimes gets the coreference wrong, and thus lower RG precision. The TEMPL system scores highest on RG precision and count, and CS recall on both datasets. This is because TEMPL can make use of domain knowledge which is not available to the neural models. TEMPL performs poorly on MLB in terms of BLEU, in fact it is considerably worse compared to the similar template system on ROTOWire (see Table 2). This suggests that the task of creating MLB game summaries is hard, even for a template system which does not perform any sophisticated generation.

Ablation Experiments We further examined how individual model components contribute to the quality of the generated summaries. To assess the impact of hierarchical attention (Section 4.2) over ED+CC, we report the performance of a stripped-down variant of our model without dynamic entity memory. Specifically, the entity memory was kept static and set to $u_{t=-1,k}$ (see Equation (5)). In this model, attention over entity vectors is:

$$\Psi_{t,k} \propto \exp(d_t^\top W_A u_{t=-1,k})$$ (14) 

We next examined the contribution of dynamic memory, by adding it to this model without the
gate \( \gamma_t \) (i.e., we set \( \gamma_t \) to one) and Equation (7) then becomes:

\[
\delta_{t,k} = \sigma(W_e d_t + b_e + W_f u_{t-1,k} + b_f) \quad (15)
\]

Finally, we obtain our final ENT model, by incorporating the update gate mechanism.

The results of the ablation study are shown in Table 3. We compare ED+CC against variants “+Hier”, “+Dyn” and “+Gate” corresponding to successively adding hierarchical attention, dynamic memory, and the update gate mechanism. On both datasets, hierarchical attention, improves relation generation, content selection, and BLEU. Dynamic memory and the update gate brings further improvements to content selection and BLEU.

Because it conditions on entities, ENT is able to produce text displaying nominal coreference which is absent from the outputs of ED+CC and WS-2017. We present an example in Table 4 (and in the Appendix) where entities Dwight Howard and James Harden are introduced and then later referred to as Howard and Harden. We also see that while generating the last sentence about the next game, ENT is able to switch the focus of attention from one team (Rockets) to the other (Nuggets), while NCP+CC verbalises Nuggets twice.

**Human-Based Evaluation** Following earlier work (Wiseman et al., 2017; Puduppully et al., 2019), we also evaluated our model by asking humans to rate its output in terms of relation generation, coherence, grammaticality, and conciseness. Our studies were conducted on the Amazon Mechanical Turk platform. For ROTOWire, we compared ENT against NCP+CC, Gold, and TEMPL. We did not compare against WS-2017 or ED+CC, since prior work (Puduppully et al., 2019) has shown that NCP+CC is superior to these models in terms of automatic and human-based evaluation. For MLB, we compared ENT against NCP+CC, ED+CC, Gold, and TEMPL.

In the first study, participants were presented with sentences randomly selected from the game summary (test set) together with corresponding box and line score tables and were asked to count supporting and contradicting facts in these sentences. We evaluated 30 summaries and 4 sentences per summary for each of ROTOwire and MLB. We elicited 5 responses per summary.

As shown in Table 5, on ROTOwire ENT yields a comparable number of supporting and contradicting facts to NCP+CC (the difference is not statistically significant). TEMPL has the highest number of supporting facts, even relative to gold summaries, and very few contradicting facts. This is expected as TEMPL output is mostly factual, it essentially parrots statistics from the tables. On MLB, ENT yields a number of supporting facts comparable to Gold and NCP+CC, but significantly lower than ED+CC and TEMPL. Contradicting facts are significantly lower for ENT compared to NCP+CC, but comparable to ED+CC and higher than TEMPL and Gold.

We also evaluated the quality of the generated summaries. Following earlier work (Puduppully et al., 2019), we presented participants with two summaries at a time and asked them to choose which one is better in terms of *Grammaticality* (is the summary written in well-formed English?), *Coherence* (do the sentences in summary follow a coherent discourse?), and *Conciseness* (does the summary tend to repeat the same content?) We divided the four competing systems (Gold, TEMPL, NCP+CC, and ENT) into six pairs of summaries for ROTOwire and the five competing systems (Gold, TEMPL, ED+CC, NCP+CC, and ENT) into ten pairs for MLB. We used Best-Worst scaling (Louviere and Woodworth, 1991; Louviere

---

**Table 4:** Examples of model output for NCP+CC (top) and ENT (bottom) on ROTOwire. Recurring entities in the summaries are boldfaced and color-coded, singletons are shown in black.

---

The Houston Rockets (18–5) defeated the Denver Nuggets (10–13) 108–96 on Tuesday at the Toyota Center in Houston. The Rockets had a strong first half where they out-scored ... The Rockets were led by Donatas Motiejunas, who scored a game-high of 25 points ... James Harden also played a factor in the win, as he went 7–for ... Coming off the bench, Donatas Motiejunas had a big game and finished with 25 points ... The only other player to reach double figures in points was Arron Afflalo, who came off the bench for 12 points ... Coming off the bench, Arron Afflalo chipped in with 12 points ... The Nuggets’ next game will be on the road against the Boston Celtics on Friday, while the Nuggets will travel to Boston to play the Celtics on Wednesday.

The Houston Rockets (18–5) defeated the Denver Nuggets (10–13) 108–96 on Monday at the Toyota Center in Houston. The Rockets were the superior shooters in this game, going ... The Rockets were led by the duo of Dwight Howard and James Harden. Howard shot 9–for–11 from the field and ... Harden on the other hand recorded 24 points (7–20 FG, 2–5 3Pt, 8–9 FT), 10 rebounds and 10 assists. The only other Nugget to reach double figures in points was Arron Afflalo, who finished with 12 points (4–17 FG, ... The Rockets’ next game will be on the road against the New Orleans Pelicans on Wednesday, while the Nuggets will travel to Los Angeles to play the Clippers on Friday.
Table 5: Average number of supporting and contradicting facts in game summaries and best-worst scaling evaluation (higher is better) on ROTOwire and MLB datasets. Systems significantly different from ENT are marked with an asterisk * (using a one-way ANOVA with posthoc Tukey HSD tests; \( p \leq 0.05 \)).

|          | Supp | Contra | Gram  | Coher  | Concis |
|----------|------|--------|-------|--------|--------|
| Gold     | 2.98*| 0.28*  | 4.07* | 3.33   | -10.74*|
| TEMPL    | 6.98*| 0.21*  | -3.70*| -3.33* | 17.78* |
| NCP+CC   | 4.90 | 0.90   | -3.33*| -3.70* | -3.70  |
| ENT      | 4.77 | 0.80   | 2.96  | 3.70   | -3.33  |

|          | Supp | Contra | Gram  | Coher  | Concis |
|----------|------|--------|-------|--------|--------|
| Gold     | 2.81 | 0.15*  | 1.24* | 3.48*  | -9.33* |
| TEMPL    | 3.98*| 0.04*  | -10.67*| -7.30* | 8.43*  |
| ED+CC    | 3.24*| 0.40   | 0.22* | -0.90* | -2.47* |
| NCP+CC   | 2.86 | 0.88*  | 0.90* | -1.35* | -1.80* |
| ENT      | 2.86 | 0.52   | 8.31  | 6.07   | 5.39   |

As shown in Table 5, on ROTOwire Gold receives highest scores in terms of Grammaticality, which is not unexpected. ENT comes close, achieving better scores than NCP+CC and TEMPL, even though our model only enhances the coherence of the output. Participants find ENT on par with Gold on Coherence and better than NCP+CC and TEMPL, whose output is stilted and exhibits no variability. In terms of Conciseness, TEMPL is rated best, which is expected since it does not contain any duplication, the presented facts are mutually exclusive; ENT is comparable to NCP+CC and better than Gold.

As far as MLB is concerned, ENT achieves highest scores on Grammaticality and Coherence. It is rated high on Conciseness also, second only to TEMPL whose scores are lowest on Grammaticality and Coherence. Perhaps surprisingly, Gold is rated lower than ENT on all three metrics; we hypothesize that participants find Gold’s output too verbose compared to the other systems. Recall that MLB gold summaries are relative long, the average length is 542 tokens compared to ROTOwire whose summaries are almost half as long (see Table 1). The average length of output summaries for ENT is 327 tokens.

Taken together, our results show that ENT performs better than comparison systems on both ROTOwire and MLB. Compared to NCP+CC, it is conceptually simpler and more portable, as it does not rely on content plans which have to be extracted via an IE system which must be reconfigured for new datasets and domains.

8 Conclusions

In this work we presented a neural model for data-to-text generation which creates entity-specific representations (that are dynamically updated) and generates text using hierarchical attention over the input table and entity memory. Extensive automatic and human evaluation on two benchmarks, ROTOwire and the newly created MLB, show that our model outperforms competitive baselines and manages to generate plausible output which humans find coherent, concise, and factually correct. However, we have only scratched the surface; future improvements involve integrating content planning with entity modeling, placing more emphasis on play-by-play, and exploiting dependencies across input tables.

Acknowledgments

We would like to thank Adam Lopez for helpful discussions. We acknowledge the financial support of the European Research Council (Lapata; award number 681760).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.

Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi. 2018. Simulating action dynamics with neural process networks. In 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. OpenReview.net.

Marc Cavazza and Fred Charles. 2005. Dialogue generation in character-based interactive storytelling. In AIIDE, pages 21–26.

Marc Cavazza, Fred Charles, and Steven J Mead. 2002. Character-based interactive storytelling. IEEE Intelligent systems, 17(4):17–24.
Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. **Bleu: a method for automatic evaluation of machine translation.** In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics.

Laura Perez-Beltrachini and Mirella Lapata. 2018. **Bootstrapping generators from noisy data.** In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Volume 1 (Long Papers), pages 1516–1527. Association for Computational Linguistics.

Ellen Prince. 1981. Toward a taxonomy of given-new information. In Peter Cole, editor, *Radical pragmatics*, pages 223 – 255. Academic Press, New York/London.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019. **Data-to-text generation with content selection and planning.** In Proceedings of the 33rd AAAI Conference on Artificial Intelligence, Honolulu, Hawaii.

Ehud Reiter and Robert Dale. 1997. *Building applied natural language generation systems*. Nat. Lang. Eng., 3(1):57–87.

Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge University Press, New York, NY.

Jacques Robin. 1994. **Revision-based generation of Natural Language Summaries providing historical Background.** Ph.D. thesis, Ph. D. thesis, Columbia University.

Donia Scott and Clarisse Siekenius de Souza. 1990. Getting the message across in RST-based text generation. In Robert Dale, Chris Mellish, and Michael Zock, editors, *Current Research in Natural Language Generation*, pages 47–73. Academic Press, New York.

Kumiko Tanaka-Ishii, Koiti Hasida, and Itsuki Noda. 1998. **Reactive content selection in the generation of real-time soccer commentary.** In 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 2.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. **Pointer networks.** In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems* 28, pages 2692–2700. Curran Associates, Inc.

Marilyn A Walker, Ricky Grant, Jennifer Sawyer, Grace I Lin, Noah Wardrip-Fruin, and Michael Buell. 2011. Perceived or not perceived: Film character models for expressive nlg. In *International Conference on Interactive Digital Storytelling*, pages 109–121. Springer.

Ronald J. Williams and Jing Peng. 1990. **An efficient gradient-based algorithm for on-line training of recurrent network trajectories.** *Neural Computation*, 2(4):490–501.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. **Challenges in data-to-document generation.** In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 2253–2263. Association for Computational Linguistics.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. **Hierarchical attention networks for document classification.** In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1480–1489. Association for Computational Linguistics.

### A Appendix

#### Hyperparameters
Table 6 contains the hyperparameters used for our ENT model on the ROTOwire and MLB datasets.

#### Results on the Development Set
Table 7 (top) shows results on the ROTOwire development set for our dynamic entity memory model (ENT), the best system of Wiseman et al. (2017) (WS-2017) which is an encoder-decoder model with conditional copy, the template generator (TEMPL), our implementation of encoder-decoder model with conditional copy (ED+CC), and NCP+CC (Puduppully et al., 2019). We see that ENT achieves scores comparable to NCP+CC, but performs better on the metrics of RG precision, CS precision, and CO. Table 7 (bottom) also presents our results on MLB. ENT achieves highest BLEU amongst all models and highest CS recall and RG count amongst neural models.

#### Qualitative Examples
Tables 8 and 9 contain examples of model output for ROTOwire and MLB, respectively. Because it conditions on entities, ENT is able to produce text displaying nominal coreference compared to other models.

---

*Note: This text is a summary of the original content and does not include all details.*
Table 6: Hyperparameters for ROTOWire and MLB.

| System       | Word Embeddings | ROTOWire | MLB |
|--------------|-----------------|----------|-----|
|              |                 | 600      | 300 |
| Hidden state size | 600             | 600      | 300 |
| Entity memory size | 300             | 300      | 300 |
| LSTM Layers  | 2               | 1        | 1   |
| Input Feeding | Yes             | Yes      | Yes |
| Dropout      | 0.3             | 0.3      | 0.3 |
| Optimizer    | Adagrad         | Adagrad  | Adagrad |
| Initial learning rate | 0.15        | 0.15      | 0.15 |
| Learning rate decay | 0.97         | 0.97      | 0.97 |
| Epochs       | 25              | 25       | 25  |
| BPTT size    | 100             | 100      | 100 |
| Batch size   | 5               | 12       | 5   |
| Inference beam size | 5            | 5        | 5   |

Table 7: Results on ROTOWire (RW) and MLB development sets using relation generation (RG) count and precision, content selection (CS) precision and recall, content ordering (CO) in normalized Damerau-Levenshtein distance, and BLEU.

| System       | RW    | RG | CS | CO | DLD% | BLEU |
|--------------|-------|----|----|----|------|------|
| Template     | TEMPL | #  | P% | P% | R%   |      |
|              |       |    |    |    |      |      |
|              | 54.29 | 99.92 | 26.61 | 59.16 | 14.42 | 8.51 |
|              | 23.95 | 75.10 | 28.11 | 35.86 | 15.33 | 14.57 |
|              | 22.68 | 79.40 | 29.96 | 34.11 | 16.00 | 14.00 |
|              | 33.88 | 87.51 | 33.52 | 51.21 | 18.57 | 16.19 |
|              | 31.84 | 91.97 | 36.65 | 48.18 | 19.68 | 15.97 |
| NCP+CC       | ED+CC | 18.69 | 92.65 | 8.51 | 60.00 | 9.43 |
|              | 33.88 | 87.51 | 33.52 | 51.21 | 18.57 | 16.19 |
|              | 25.93 | 75.10 | 29.96 | 34.11 | 16.00 | 14.00 |
|              | 17.70 | 88.01 | 33.52 | 51.21 | 18.57 | 16.19 |
|              | 21.32 | 88.16 | 36.65 | 48.18 | 19.68 | 15.97 |
|              | 16.19 | 92.65 | 8.51 | 60.00 | 9.43 | 11.13 |

Table 8: Example output from the template-based system, NCP+CC (Puduppully et al., 2019) and our ENT model for ROTOWire. Recurring entities in the summaries are boldfaced and colorcoded, singletons are shown in black.
The Tampa Bay Rays defeated the Oakland Athletics 13–4. Jason Isringhausen (0–0) allowed 0 runs, 1 hits and 0 walks in 1 innings. Jeff Niemann (4–3) allowed 4 runs, 8 hits and 0 walks in 8 innings. Sean Gallagher (1–2) allowed 9 runs, 3 hits and 5 walks in 2 1/3 innings. Kevin Cameron (0–0) allowed 0 runs, 0 hits and 1 walks in 2 innings. Gio Gonzalez (0–0) allowed 4 runs, 6 hits and 3 walks in 3 2/3 innings. Jason Bartlett hit 2 RBI single in the first. Orlando Cabrera hit 1 RBI homer in the third Athletics. Akinori Iwamura hit 2 RBI single in the third. Jason Bartlett hit 1 RBI triple in the fifth. Akinori Iwamura hit 1 RBI single in the fifth. Jack Cust hit 2 RBI single in the sixth. Jack Cust hit 1 RBI triple in the first. Jason Bartlett drove in five runs, the Rays beat the Oakland Athletics on Tuesday night. Bartlett hit a two–run homer in the eighth. Jack Cust got a big boost from Jason Bartlett’s grand slam. Bartlett drove in five runs, the Rays beat the Oakland Athletics on Friday night. Jeff Niemann (4–3) allowed four runs, three earned, and eight hits in eight innings. Sean Gallagher (1–2) gave up a career–high nine runs in 2 1/3 innings. Jack Cust hit a two–run homer in the seventh and Jack Cust added a two–run homer for the A’s, who have lost four of five. Tampa Bay has lost four of five. The Rays have lost four of five. Oakland has lost four of five. The Rays have lost four of five. Tampa Bay has lost four of five. The Rays have lost four of five. Tampa Bay has lost five of six. The Rays have lost four of five. Tampa Bay has lost five of six. The Rays have lost four of five. Oakland has lost four of five. The Rays have lost four of five. The Rays have lost four of five. Tampa Bay has lost five of six. The Rays have lost four of five. Tampa Bay has lost five of six. The Rays have lost four of five. Oakland has lost four of five. The Rays have lost four of five. Tampa Bay has lost five of six. The Rays have lost four of five. Oakland has lost four of five. The Rays have lost four of five. Tampa Bay has lost five of six. The Rays have lost four of five. Oakland has lost four of five. The Rays have lost four of five. There is no way to beat the A’s. Jason Isringhausen pitched eight strong innings and the Rays beat the Oakland Athletics 13–4 on Tuesday night. Bartlett had a two–run single in the first and added a two–run single in the third to help the Rays take a 6–1 lead. Bartlett had a two–run single in the third and a two–run single in the sixth. Jack Cust hit a two–run homer in the eighth. The A’s scored six runs off Sean Gallagher (1–2), who gave up seven earned runs. Jack Cust had three hits and drove in five runs, the Rays beat the Oakland Athletics on Tuesday night. Bartlett had a two–run single in the first and added a two–run single in the third to help the Rays take a 6–1 lead. Bartlett had a two–run single in the third. Jack Cust hit a two–run homer in the eighth for the Rays, who have lost four of five. Oakland starter Gio Gonzalez gave up four runs and six hits in 3 2/3 innings. The right–hander struck out three and did not walk a batter for the second time this season. The right–hander is 4–0 in six career starts against the A’s. Orlando Cabrera hit a solo homer in the third for the A’s, who have lost four of five. Oakland starter Gio Gonzalez gave up four runs and six hits in 3 2/3 innings. The right–hander struck out six and walked three. The right–hander was coming off a 1–0 loss to the A’s in his previous start, when he gave up six runs in 4 1/3 innings of a 10–0 loss to the A’s. The A’s took a 1–0 lead in the first when Ben Zobrist drew a bases–loaded walk and Bartlett had a two–run single.

Table 9: Example output from the template-based system, ED+CC, NCP+CC (Puduppully et al., 2019) and our ENT model for MLB. Recurring entities are boldfaced and colorcoded, singletons are shown in black.