Neural Text Generation in Stories Using Entity Representations as Context

Elizabeth Clark Yangfeng Ji Noah A. Smith
Paul G. Allen School of Computer Science & Engineering
University of Washington
Seattle, WA, USA
{eaclark7,yangfeng,nasmith}@cs.washington.edu

Abstract

We introduce an approach to neural text generation that explicitly represents entities mentioned in the text. Entity representations are vectors that are updated as the text proceeds; they are designed specifically for narrative text like fiction or news stories. Our experiments demonstrate that modeling entities offers a benefit in two automatic evaluations: mention generation (in which a model chooses which entity to mention next and which words to use in the mention) and selection between a correct next sentence and a distractor from later in the same story. We also conduct a human evaluation on automatically generated text in story contexts; this study supports our emphasis on entities and suggests directions for further research.

1 Introduction

We consider the problem of automatically generating narrative text, a challenging problem at the junction of computational creativity and language technologies (Gervás, 2009). We are motivated in particular by potential applications in personalized education and assistive tools for human authors, though we believe narrative might also play a role in social conversational agents (Sordoni et al., 2015). In this work, the term "narrative text" refers primarily to fiction but might also include news and other kinds of stories.

A notable difference between longstanding work in natural language generation and recent "neural" models is in the treatment of *entities* and the words used to refer to them. Particularly in the generation of narrative text, character-centered generation has been shown important in character dialogue generation (Walker et al., 2011; Cavazza and Charles, 2005) and story planning (Cavazza et al., 2002). Neural models, on the other hand, treat mentions as just more words, relying on representation learning to relate the people in a story through the words alone.

Context	All of a sudden, [*Emily*]$_1$ walked towards [*the dragon*]$_2$.
Current Sentence	[*Seth*]$_3$ yelled at [*her*]$_1$ to get back but ________________

Figure 1: An example of entity-labeled story data. The brackets indicate which words are part of entity mentions. Mentions marked with the same number refer to the same entity. The goal is to continue the story in a coherent way. The actual story reads, "*Seth yelled at her to get back but she ignored him.*"

Entities are an important element of narrative text. Centering Theory places entities at the center of explaining what makes text coherent (Grosz et al., 1995). In this work, we incorporate entities into neural text generation models; each entity in a story is given its own vector representation, which is updated as the story unfolds. These representations are learned specifically to predict words—both mentions of the entity itself and also the following context. At a given moment in the story, the current representations of the entities help to predict what happens next.

Consider the example in Figure 1. Given the context, the reader expects the subsequent words and sentences of the passage to track the results of Emily approaching the dragon. Future text should include references to Emily's character and the dragon and the result of their interaction. The choice of entity generated next in the sentence will change what language should follow that mention and will shape and drive the direction of the story. For this reason, we propose using entity representations as context for generation.

Of course, entities are not the only context needed for coherent language generation; previously generated content remains an important source of information. We use a simple, parameter-free method for combining preceding context with entity context within an end-to-end–

trainable neural language generator.

We evaluate our model's performance through two automatic evaluation tasks. The first is a new mention generation task inspired by earlier work in referring expression generation (Dale and Reiter, 1995). The second is a sentence selection task inspired by coherence tests from Barzilay and Lapata (2008). Our model outperforms strong baselines on both tasks.

We further conduct a human evaluation in which our model's generated sentences are compared to a strong baseline model. This evaluation elucidates strengths and weaknesses of our model and offers guidance for future work on narrative text generation.

2 Model Description

We propose an entity-based generation model (ENGEN)[1] that combines three different sources of contextual information for text generation:

1. The content that has already been generated within the current sentence
2. The content that was generated in the previous sentence
3. The current state of the entities mentioned in the document so far

Each of these types of information is encoded in vector form, following extensive past work on recurrent neural network (RNN) language models. The first source of context is the familiar hidden state vector of the RNN; more precisely, our starting point is a sequence-to-sequence model (Sutskever et al., 2014). Representations of the second and third forms of context are discussed in §2.1 and §2.2, respectively. The combination of all three context representations is described in §2.3.

2.1 Context from Previous Sentence

As noted, our starting point is a sequence-to-sequence model (Sutskever et al., 2014); the last hidden state from the previous sentence offers a representation of the preceding context. We add an attention mechanism (Bahdanau et al., 2015). Let $\boldsymbol{h}_{t,i}$ and $\boldsymbol{h}_{t-1,j}$ be the LSTM hidden states of sentence t at timestep i and the previous sentence $t-1$ at timestep j, where j ranges over the number of words in the previous sentence. To summarize the contextual information from the previous sentence for predicting the next word at timestep $i+1$ in sentence t, we have

$$\boldsymbol{p}_{t-1,i} = \sum_j \alpha_{i,j} \boldsymbol{h}_{t-1,j}, \text{ where} \tag{1}$$

$$\alpha_{i,j} = \frac{\exp(\boldsymbol{h}_{t-1,j} \mathbf{W}_a \boldsymbol{h}_{t,i})}{\sum_{j'} \exp(\boldsymbol{h}_{t-1,j'} \mathbf{W}_a \boldsymbol{h}_{t,i})} \tag{2}$$

is the attention weight for $\boldsymbol{h}_{t-1,j}$. Unlike the definition of attention in Bahdanau et al. (2015), here we use the bilinear product in Equation 2 to encourage correlation between $\boldsymbol{h}_{t,i}$ and $\boldsymbol{h}_{t-1,j}$ for coherence in text generation. In §2.3, we will combine this with $\boldsymbol{h}_{t,i}$ for predicting the next word; we refer to that model as S2SA, and it serves as an entity-unaware baseline in our experiments.

2.2 Context from Entities

In S2SA, the context of a sentence is (at best) represented by compressing information about the words that have appeared in the previous sentence. Past research has suggested several approaches to capturing other contextual information. For example, Lau et al. (2017) and Ghosh et al. (2016) have sought to capture longer contexts by modeling topics. Recently, Ji et al. (2017) introduced a language model, ENTITYNLM, that adds explicit tracking of entities, which have their own representations that are updated as the document progresses.[2] That model was introduced for analysis tasks, such as language modeling and coreference resolution, where the texts (and their coreference information) are given, and the model is used to score the texts to help resolve coreference relationships.[3] ENTITYNLM's strong performance on language modeling suggests the potential of distributed entity representations as another source of contextual information for text generation. Inspired by that work, we maintain the dynamic representation of entities and use them as contextual information when generating text.

In general, every entity in a document (e.g., EMILY in Figure 1) is assigned a vector representation; this vector is updated every time the entity is mentioned. This is entirely appropriate for generating narrative stories in which characters develop and change over long contexts. When we

[1]Code available at `github.com/eaclark07/engen`.

[2]Because space does not permit a full exposition of all the details of ENTITYNLM, we refer the interested reader to Ji et al. (2017).

[3]The entity prediction task used in their work is relevant to our mention generation task, which will be discussed in §5.

generate text, the model will have access to the current representation of every participant (i.e., every entity) in the story at that time (denoted by $\boldsymbol{e}_{i,t}$ for entity i at timestep t).

When choosing which entity is referred to at timestep t, there are $m + 1$ options, where m is the number of entities tracked in the document so far—the $(m+1)$th is for a new, previously unmentioned entity. Given that a word is part of an entity mention and given the previous hidden state, the probability that the word is referring to a given entity $i \in \{1, \ldots, m + 1\}$ is proportional to:

$$\exp(\boldsymbol{h}_{t-1}^{\top} \mathbf{W}_{\mathit{entity}} \boldsymbol{e}_{i,t-1} + \boldsymbol{w}_{\mathit{dist}}^{\top} \boldsymbol{f}(i)), \tag{3}$$

where $\mathbf{W}_{\mathit{entity}}$ is a weight matrix for predicting the entities and $\boldsymbol{w}_{\mathit{dist}}^{\top} \boldsymbol{f}(i)$ is a term that takes into account distance features between the current and past entity mentions.

Once an entity is selected, its vector is assigned to $\boldsymbol{e}_{\mathit{current}}$, which is used to generate the word w_t. If the model decided the current word should not refer to an entity, then $\boldsymbol{e}_{\mathit{current}}$ is still used and will be the representation of the most recently mentioned entity. If the choice is a new, previously unmentioned entity, then $\boldsymbol{e}_{\mathit{current}}$ is initialized with a new embedding randomly generated from a normal distribution:

$$\boldsymbol{u} \sim \mathcal{N}(\boldsymbol{r}, \sigma^2 \mathbf{I}), \tag{4}$$

where $\sigma = 0.01$ and $\boldsymbol{r}$ is a parameter vector that is used to determine whether the next word should refer to an entity.

Once the word w_t has been generated, the entity representation is updated based on the new hidden state information, $\boldsymbol{h}_t$.

2.3 Combining Contexts

Our new model merges S2SA and ENTITYNLM. Both provide a representation of context: respectively, the previous sentence's representation ($\boldsymbol{p}_t$) and the most salient entity's representation ($\boldsymbol{e}_{\mathit{current}}$). The hidden state $\boldsymbol{h}_{t-1}$ is, of course, also available, and is intended to capture local contextual effects. The challenge is how to combine these representations effectively for text generation.

In this work, for simplicity, we choose a combination function without any extra parameters, and leave the detailed investigation of paramaterized composition functions as future work. We use a max-pooling function to form a context vector $\boldsymbol{c}_t$ with the same dimensionality as $\boldsymbol{h}_{t-1}$ (and, of course, $\boldsymbol{p}_t$ and $\boldsymbol{e}_{\mathit{current}}$). Specifically, at time step t, each element of the combined context vector $\boldsymbol{c}_t$ is calculated as follows. For $k \in \{1, ..., |\boldsymbol{c}_t|\}$,

$$\boldsymbol{c}_t[k] = \max(\boldsymbol{h}_{t-1}[k], \boldsymbol{p}_t[k], \boldsymbol{e}_{\mathit{current}}[k]). \tag{5}$$

The max pooling technique originates from the design of convolutional neural networks and has been found useful elsewhere in NLP (Kalchbrenner et al., 2014). Other alternatives, including average pooling, min pooling, and element-wise multiplication on all three vectors, were considered in informal preliminary experiments on development data and found less effective than max pooling.

This combined context vector $\boldsymbol{c}_t$ is used to generate word w_t by calculating the probability of each word type in the vocabulary. We use a class-factored softmax function (Goodman, 2001; Baltescu and Blunsom, 2015). This choice greatly reduces the runtime of word prediction. In practice, we often find it gives better performance than standard softmax.

2.4 Learning

The training objective is to maximize the log-probability of $\boldsymbol{X}$:

$$\ell(\boldsymbol{\theta}) = \log P(\boldsymbol{X}; \boldsymbol{\theta}) = \sum_t \log P(X_t; \boldsymbol{\theta}) \tag{6}$$

$\boldsymbol{\theta}$ denotes all of the model's parameters. X_t represents all decisions at timestep t about the word (whether it is part of a entity mention, and if so, the entity the mention refers to, the length of the mention, and the word itself).

These decisions are made by calculating probabilities for each available option using the current state of the neural network (a vector) and the current vector representations of the entities. Given the probabilities, the next word is assumed to have been randomly generated by sampling.

While we might consider training the model to maximize the probability of the generated words directly, treating the entity-related variables as *latent*, this would create a mismatch between how we train and use the model. For generation, the model explicitly predicts not just the word, but also the entity information associated with that word. Training with latent variables is also expensive. For these reasons, we use the same training method used for ENTITYNLM, which requires training data annotated with mention and coreference information (entity clusters).

2.5 Variants

In our experiments, we consider the combined model (ENGEN) and two ablations: S2SA and a model similar to ENTITYNLM. Note that, unlike past work with previous-sentence context, S2SA uses max pooling for $\boldsymbol{h}_{t-1}$ and $\boldsymbol{p}_t$ and class-factored softmax; our version of ENTITYNLM also uses max pooling and class-factored softmax. All of these models are trained in a similar way.

3 Implementation Details

The models are implemented using DyNet (Neubig et al., 2017) with GPU support. We optimize with SGD, with a learning rate of $\lambda = 0.1$. The dimensions of input layer, hidden layer, and entity representation are fixed at 512 (hyperparameter optimization might lead to better solutions). The input word embeddings are randomly initialized with the default method in DyNet and updated during training jointly with other parameters. For class-factored softmax, we use 160 Brown clusters (Brown et al., 1992; Liang, 2005) estimated from the training data.

4 Data

We trained all models on 312 adventure books from the Toronto Book Corpus (Zhu et al., 2015), with development and test sets of an additional 39 books each. We divided the books into smaller segments, where each segment includes up to 50 sentences. There are 33,279 segments in the training set, 4,577 in the dev. set, and 4,037 in the test set. This helps with memory efficiency, allowing us to train the model without building a recurrent neural network on the entire book.

All the tokens in the data were downcased, and numbers were replaced with a special NUM token. The vocabulary was selected by replacing the lowest frequency (less than 10) word types with a special UNK token. There are 43 million tokens, and the vocabulary size is 35,443.

To obtain entity annotations, we used the Stanford CoreNLP system (Clark and Manning, 2016a,b), version 3.8.0. From the coreference resolution results, we noticed that some entity mentions include more than 70 tokens, which is likely in error. To simplify the problem, we only kept the mentions consisting of three words or fewer, which covers more than 95% of the mentions in the training data. For mentions of more than three words, we replaced them with their head word, as determined by the Stanford CoreNLP system. While truncating these mentions sacrifices some information, we believe this preprocessing step is justifed as it retains most character names and pronouns, an especially important entity type for stories.

Of course, the use of automatic annotations from a coreference system will introduce noise and risks "confusing" the entity-aware models. The benefit is that we were able to train on a much larger corpus than any existing coreference dataset (e.g., the CoNLL 2012 English shared task training set has only 1.3 million tokens; Pradhan et al., 2012). Further, a corpus of books offers language that is much closer to our intended narrative text generation applications. Our experiments aim to measure some aspects of our models' intrinsic correctness, though we emphasize that even if entity information is incorrect at training time, it may still be helpful.

For all experiments, the same preprocessed dataset and trained models were used. The best models were selected based on development set log likelihood (Equation 6).

5 Experiment: Mention Generation

The goal of our first experiment is to investigate each model's capacity to mention an entity in context. For example, in Figure 1, *Emily* and *her* are both possible mentions of EMILY's character, but the two cannot be used interchangeably. Inspired by early work on referring expression generation (Dale and Reiter, 1995) and recent work on entity prediction (Modi et al., 2017), we propose a new task we call *mention generation*. Given a text and a slot to be filled with an entity mention, a model must choose among all preceding entity mentions and the correct mention. So if the model was choosing the next entity mention to be generated in Figure 1, it would select between all the previous entity mentions (*Emily*, *the dragon*, *Seth*, and *her*) and the correct mention (*she*).

In our model, each candidate mention is augmented with the index of its entity. Therefore, performing well on this task requires choosing both the entity and the words used to refer to it; this notion of quality is our most stringent evaluation measure. It requires the greatest precision, as it is

model	cluster and mention	cluster only	mention only
1. Reverse order	0.12	0.38	0.15
2. S2SA	—	—	0.44
3. ENTITYNLM	0.52	0.46	0.54
4. ENGEN	0.53	0.46	0.55

Table 1: MAP on the mention generation task. Note that these results can only be compared between models, not between tasks, as there are a different number of candidates for each of the tasks.

cluster and mention	cluster only	mention only
[*Emily*]$_1$	*EMILY	Emily
[*the dragon*]$_2$	THE DRAGON	the dragon
[*Seth*]$_3$	SETH	Seth
[*her*]$_1$		her
*[*she*]$_1$		*she

Figure 2: Candidate lists for each of the mention generation tasks for completing the blank in Figure 1. The asterisk (*) indicates the correct choice.

possible to select the correct mention but not the correct cluster and vice versa. Since S2SA does not model entities, we also compare systems on quality of mentions alone (without entity clusters). For completeness, we include cluster quality for the entity-aware models. Candidate lists for each task to generate the next mention in the example in Figure 1 are shown in Figure 2.

The experiment setup does not require manual creation of candidate lists. However, it makes the mention generation task even more challenging, because the size of a candidate list can exceed 100 mention candidates.

We note that the difficulty of this task increases as we consider mention slots later and later in the document. The first mention generation choice is a trivial one, with a single candidate that is by definition correct. As more entity mentions are observed, the number of options will increase.[4] To enable aggregation across contexts of all lengths, we report the mean average precision (MAP) of the correct candidates, where the language model scores are used to rank candidates.

Baselines Along with the two ablated models (S2SA and ENTITYNLM), we include a "reverse order" baseline, which ranks mentions by recency (the first element in the ranking is the most recent mention, then the second-most-recent, and so on).

[4]Note that the list of candidates may include duplicate entries with the same mention words and cluster. These are collapsed since they will have the same score under a language model.

Results The ranking results of ENGEN and other systems are reported in Table 1. A higher MAP score implies a better system. We measure the overall performance of all the systems, along with their performance on selecting the *mention only* and *entity cluster only*. Across all the evaluation measures, ENGEN gives the highest MAP numbers. Recall that S2SA does not have a component for entity prediction, therefore we only compare it with ENGEN in the *mention only* case. The difference between line 4 and line 2 on the *mention only* column shows the benefit of adding entity representations for text generation. The difference between lines 3 and 4 shows that local context also gives a small boost. Although the distance between the current slot and previous entity mention has been shown as a useful feature in coreference resolution (Clark and Manning, 2016b), line 1 shows distance alone is not an effective heuristic for mention generation.

6 Experiment: Pairwise Sentence Selection

The sentence selection task is inspired by tests of coherence used to assess text generation components automatically, without human evaluation (Barzilay and Lapata, 2008). It serves as a sanity check, as it was conducted prior to full generation and human evaluations (§7). Since the models under consideration are generative, they can be used to assign scores to candidate sentences, given a context.

In our version of this task, we provide a model with $n - 1 = 49$ sentences of preceding context, and offer two choices for the nth (50th) sentence: the actual 50th sentence or a distractor sentence randomly chosen from the next 50 sentences. A random baseline would achieve 50% accuracy.

Because the distractor comes from the same

Context	All of a sudden, [*Emily*]$_1$ walked towards [*the dragon*]$_2$.
1.	[*Seth*]$_3$ yelled at [*her*]$_1$ to get back but [*she*]$_1$ ignored [*him*]$_3$.
2.	[*She*]$_1$ patted [*its head*]$_4$ and [*it*]$_2$ curled up outside [*the cave*]$_5$.
3.	"[*Emily*]$_1$, how did [*you*]$_1$ keep [*that dragon*]$_2$ from attacking [*us*]$_6$?"

Figure 3: A passage's last sentence of context, and 3 sentences from various points in the next passage.

story (with similar language, characters, and topics) and relatively nearby (in 2% cases, the very next sentence), this is not a trivial task. Consider the example in Figure 3. All of the sentences share lexical and entity information with the last line of the context. However, the first sentence immediately follows the context, while the second and third sentences are 10 lines and 48 lines away from the context, respectively. These entity and lexical similarities make distinguishing the actual sentence from the random sentence a challenging problem for the model.

To select the sentence, the model scores each of the two candidate sentences based on its probability on words and all entity-related information as defined in Equation 6. (Both candidate sentences come from the preprocessed data and have the entity annotations described in §4.)

The sentence that receives the higher probability is chosen. For each of the 4,037 segments of context in the test set, we calculated the accuracy of each model at distinguishing the gold sentence from a distractor sentence. We ran this pairwise decision 5 times, each time with a different set of randomly selected distractor sentences and averaged their performance across all 5 rounds.

Results The accuracy of each of the models is reported in Table 2. The best performance is obtained by ENGEN, which is significantly better than the other two models ($p < 0.05$, binomial test). Unlike the mention generation task, S2SA beats ENTITYNLM at this task; this difference in performance shows the importance of local context. Although we performed five different rounds of random sampling to choose a sentence from the following segment as the distractor sentence, the standard deviations in Table 2 show the results are generally consistent across rounds, regardless of the distractor's distance from the gold sentence.

model	mean accuracy	s.d.
1. S2SA	0.546	0.01
2. ENTITYNLM	0.534	0.006
3. ENGEN	*0.566	0.008

* signficantly better than lines 1 and 2 with $p < 0.05$.

Table 2: Accuracy in choosing the actual next sentence, given 49 sentences of context, with a distractor from slightly later in the story. The mean accuracies and standard deviation are calculated across the five rounds of pairwise sentence selection.

7 Human Evaluation: Sentence Generation

The task motivating the work in this paper is narrative text generation. As such, evaluation by human judges of the quality of generated text is the best measure of our methods' quality. This study simplifies that evaluation by distilling the judgment down to a forced choice between contextually generated sentences generated by two different models. We use this task to investigate the strengths and weaknesses of our model in a downstream application. By asking humans to decide which sentences they prefer (in a given context) and to explain why, we can analyze where our model is helping and where text generation for stories still needs to improve, both with respect to entities and to other aspects of language. Here we control for training data and assess the benefit of including entity information for generating sentences to continue a story.

We presented Amazon Mechanical Turkers[5] with a short excerpt from a story and two generated sentences, one generated by ENGEN and one generated by the entity-unaware S2SA. We asked them to *"choose a sentence to continue the story"* and to briefly explain why they made the choice they did, an approach similar to that in other story-based work such as Lukin et al. (2015).

Note that we did not prime Turkers to focus on entities. Rather, the purpose of this experiment was to examine the performance of the model in a story generation setting and to get feedback on what people generally notice in generated text, not only with regard to entities. By keeping the task

[5]We selected workers who had completed over 1,000 tasks, had over a 95% task acceptance rate, and were from the United States.

open-ended, we can better analyze what people value in generated text for stories, and where our model supports that and where it doesn't.

We used a subset of 50 randomly selected text segments from the test set described in §4. However, for the human evaluation, we only used the final 60 words[6] of the story segments to keep the amount of reading and context manageable for Turkers. The models had access to the same subset of the context that the evaluator saw, not all 50 sentences from the original segment as in earlier experiments. For each context, we randomly sampled a sentence to continue the document, using each of two models: ENGEN and S2SA. These two models allowed us to see if adding the entity information noticeably improved the quality of the generation to evaluators.

Initial experiments showed that fluency remains a problem for neural text generation. To reduce the effect of fluency on Turkers' judgments, we generated 100 samples for each context/model pair and then reranked them with a 5-gram language model (Heafield, 2011) that was trained on the same training data. The two top ranked sentences (one for ENGEN and one for S2SA) were presented in random order and without reference to the models that generated them.

For each of the 50 contexts, we had 11 Turkers pick a candidate sentence to continue the story passage. Turkers were paid $0.10 for each evaluation they completed. In total, 93 Turkers completed the task. The number of passages Turkers completed ranged from 1 to all 50 story segments (with an average of 6.1). While the quantitative portion of this task would be easy to scale, the qualitative portion is not; we kept the human evaluation small, running it until reaching saturation.

Results Each pair of sentences was evaluated by 11 Turkers, so each of the passages could receive up to 11 votes for ENGEN. For 27 of the passages, the majority of Turkers (6 or more) chose the sentence from ENGEN, versus 23 passages that went to the baseline model, S2SA. The scores were close in many cases, and for several passages, Turkers noted in their explanations that while they were required to choose one sentence, both would have worked. Examples of the context and sentence pairs that were strongly in favor of ENGEN, strongly in favor of S2SA, and that received mixed reviews are shown in Table 3.

[6]We included the whole sentence that contained the 60th word, so most documents were slightly over 60 words.

When asked to explain why they selected the sentence they did, a few Turkers attributed their choices to connections between pronouns in ENGEN's suggestions to characters mentioned in the story excerpt. However, a more frequent occurrence was Turkers citing a mismatch in entities as their reason for rejecting an option. For example, one Turker said they chose ENGEN's sentence because the S2SA sentence began with "she," and there were no female characters in the context.

Interestingly, while pronouns not mentioned in the context were cited as a reason for rejecting candidate sentences, new proper noun entity mentions were seen as an asset by some. One Turker chose a S2SA sentence that referenced "Richard," a character not present in the context, saying, *"I believe including Richard as a name gives some context of the characters of the story."* This demonstrates the importance of the ability to generate new entities, in addition to referring back to exisiting entities.

However, due to the open-ended nature of the task, the reasons Turkers cited for selecting sentences extended far beyond characters and entity mentions. In fact, most of the responses credited other aspects of stories and language for their choice. Some chose sentences based on their potential to move the plot forward or because they fit better with *"the theme"* or *"the tone"* of the context. Others made decisions based on whether they thought a sentence of dialogue or a descriptive sentence was more appropriate, or a statement versus a question. Many made their decisions using deeper knowledge about the story's context. For example, in the second story listed in Table 3, one Turker used social knowledge to choose the S2SA sentence because *"the introduction makes the man sound like he is a stranger, so 'I'm proud of you' seems out of place."* In this case, even though the sentence from ENGEN correctly generated pronouns that refer to entities in the context, the mismatch in the social aspects of the context and ENGEN's sentence contributed to 7 out of 11 Turkers choosing the vaguer S2SA sentence.

While neither S2SA nor ENGEN explicitly encodes these types of information, these qualities are important to human evaluators of generated text and should influence future work on narrative text generation.

Context	EnGen	S2SA	#
he says that it was supposed to look random , but he feels it was planned . i was the target . he 's not sure , but he feels that you might have something to do with this , " cassey said sadly . " he ca n't do that ! " manny yelled . " he ca n't accuse me with no justification .	it 's not me . "	he has nothing to do with my life	10
he was wearing brown slacks and a tan button-down shirt , with wool slippers . he looked about sixty , a little paunchy , with balding brown hair and a bushy mustache . ice blue eyes observed alejo keenly , then drifted over to wara ." welcome to my home . " the man 's voice was deep and calm .	" i 'm proud of you , " he said .	" what 's going on ? '	4
bearl looked on the scene , and gasped . this was the white rock of legend , the rock that had lured him to this land . then he stopped . " look , geron . the white rock we saw from the sea . " the struggle was taking place on the white rock . the monster had his back to bearl .	" oh my god ! "	he could not believe his eyes	1

Table 3: Example generated sentences, for three different contexts. The last column indicates the number of Turkers who voted for EnGen's sentence (out of 11). While entity mentions appear in most of the generated texts, correct entity mentions are not sufficient to guarantee a win, as seen in the second example.

8 Related Work

Beyond past work already discussed, we note a few additional important areas of research relevant to our work.

Neural models for text generation Natural language generation is a classic problem in artificial intelligence. Recent use of RNNs (Sutskever et al., 2011) has reignited interest in this area. Our work provides an additional way to address the well-known drawback of RNNs: they use only limited context. This has been noted as a serious problem in conversational modeling (Sordoni et al., 2015) and text generation with multiple sentences (Lau et al., 2017). Recent work on context-aware text generation (or the related task, language modeling) has studied the possibilities of using different granularity of context. For example, in the scenario of response generation, Sordoni et al. (2015) showed a consistent gain by including one more utterance from context. Similar effects are also observed by adding topical information for language modeling and generation (Lau et al., 2017).

Entity-related generation Choosing an appropriate entity and its mention has a big influence on the coherence of a text, as studied in Centering Theory (Grosz et al., 1995). Recently, the EntityNLM proposed by Ji et al. (2017) shows that adding entity related information can improve the performance of language modeling, which potentially provides a method for entity related text generation. We build on EntityNLM, combining entity context with previous-sentence context, and demonstrate the importance of the latter in a coherence test (§6). The max pooling combination we propose is simple but effective.

Another line of related work on recipe generation included special treatment of entities as candidates in generating sentences, but not as context (Kiddon et al., 2016). Bosselut et al. (2018) also generated recipes, using neural process networks to track and update entity representations with the goal of modeling actions and their causal effects on entities. However, the entity representations are frozen during generation, rather than dynamically updated.

Mention generation Our novel mention generation task is inspired by both referring expression generation (Dale and Reiter, 1995) and entity prediction (Modi et al., 2017). The major difference is that, unlike referring expression generation, our task includes all the mentions used for entities, including pronouns; we believe it is a more realistic test of a model's handling of entities. Krahmer and Van Deemter (2012) give a comprehensive survey on early work of referring expression generation.

The mention only version of the mention generation task is related to cloze tests like the Children's Book Test (Hill et al., 2016), the "Who-did-What" Test (Onishi et al., 2016), and the CNN and Daily Mail test described by Hermann et al. (2015). However, unlike these tests, we predict all entity mentions in the text and from a dynamically expanding candidate list, typically much larger than those in other cloze tests.

Story generation Work in story generation has incorporated structure and context through event representations (Martin et al., 2017) or semantic representations, like story graphs (Rishes et al., 2013; Elson and McKeown, 2009). In this work, we provide evidence for the value of entity representations as an additional form of structure, following work by Walker et al. (2011), Cavazza and Charles (2005), and Cavazza et al. (2002).

9 Conclusion

Inspired by Centering Theory and the importance of characters in stories, we propose a neural model for text generation that incorporates context via entities. We found that combining entity representations with representations of the previous sentence and the hidden state (from a neural language model) improves performance on three tasks: mention generation, sentence selection, and sentence generation. By collecting human evaluations of sentences generated with entity information, we find that while coherently referring back to entities in the context was cited by several Turkers as a factor in their decision, the introduction of new entities and moving the narrative forward were also valued.

Therefore, while entities are a useful structure to incorporate in story generation, other structures may also prove useful, including other aspects of discourse (e.g., discourse relations or planning) or story-related structures (e.g., narrative structure).

Acknowledgments

This research was supported in part by a NSF graduate research fellowship, the DARPA CwC program through ARO (W911NF-15-1-0543), and the NVIDIA Corporation with the donation of the Tesla GPU used for this research. The authors also thank Maarten Sap for his feedback and helpful suggestions, the anonymous reviewers for their useful comments, and the participants who took part in our study.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Paul Baltescu and Phil Blunsom. 2015. Pragmatic neural language modelling in machine translation. In *NAACL*. `https://doi.org/10.3115/v1/n15-1083`.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics* 34(1):1–34. `https://doi.org/10.1162/coli.2008.34.1.1`.

Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi. 2018. Simulating action dynamics with neural process networks. In *ICLR*.

Peter F. Brown, Vincent J. Della Pietra, Peter V. de Souza, Jennifer C. Lai, and Robert L. Mercer. 1992. Class-based n-gram models of natural language. *Computational Linguistics* 18:467–479.

Marc Cavazza and Fred Charles. 2005. Dialogue generation in character-based interactive storytelling. In *AIIDE*.

Marc Cavazza, Fred Charles, and Steven J. Mead. 2002. Character-based interactive storytelling. *IEEE Intelligent Systems* 17:17–24. `https://doi.org/10.1109/mis.2002.1024747`.

Kevin Clark and Christopher D. Manning. 2016a. Deep reinforcement learning for mention-ranking coreference models. In *EMNLP*. `https://doi.org/10.18653/v1/d16-1245`.

Kevin Clark and Christopher D. Manning. 2016b. Improving coreference resolution by learning entity-level distributed representations. In *ACL*. `https://doi.org/10.18653/v1/p16-1061`.

Robert Dale and Ehud Reiter. 1995. Computational interpretations of the Gricean maxims in the generation of referring expressions. *Cognitive Science* 19(2):233–263. `https://doi.org/10.1207/s15516709cog1902_3`.

David K. Elson and Kathleen McKeown. 2009. A tool for deep semantic encoding of narrative texts. In *ACL-IJCNLP*. `https://doi.org/10.3115/1667872.1667875`.

Pablo Gervás. 2009. Computational approaches to storytelling and creativity. *AI Magazine* 30:49–62. https://doi.org/10.1609/aimag.v30i3.2250.

Shalini Ghosh, Oriol Vinyals, Brian Strope, Scott Roy, Tom Dean, and Larry Heck. 2016. Contextual LSTM (CLSTM) models for large scale NLP tasks. arXiv:1602.06291.

Joshua Goodman. 2001. Classes for fast maximum entropy training. In *ICASSP*. https://doi.org/10.1109/icassp.2001.940893.

Barbara J. Grosz, Scott Weinstein, and Aravind K. Joshi. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics* 21(2):203–225. https://doi.org/10.21236/ada324949.

Kenneth Heafield. 2011. KenLM: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NIPS*.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2016. The Goldilocks principle: Reading children's books with explicit memory representations. In *ICLR*.

Yangfeng Ji, Chenhao Tan, Sebastian Martschat, Yejin Choi, and Noah A. Smith. 2017. Dynamic entity representations in neural language models. In *EMNLP*. https://doi.org/10.18653/v1/d17-1195.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *ACL*. https://doi.org/10.3115/v1/p14-1062.

Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. 2016. Globally coherent text generation with neural checklist models. In *EMNLP*. https://doi.org/10.18653/v1/d16-1032.

Emiel Krahmer and Kees Van Deemter. 2012. Computational generation of referring expressions: A survey. *Computational Linguistics* 38(1):173–218. https://doi.org/10.1162/coli_a_00088.

Jey Han Lau, Timothy Baldwin, and Trevor Cohn. 2017. Topically driven neural language model. In *ACL*. https://doi.org/10.18653/v1/p17-1033.

Percy Liang. 2005. *Semi-supervised learning for natural language*. Ph.D. thesis, Massachusetts Institute of Technology.

Stephanie M. Lukin, Lena Reed, and Marilyn A. Walker. 2015. Generating sentence planning variations for story telling. In *SIGDIAL*. https://doi.org/10.18653/v1/w15-4627.

Lara J. Martin, Prithviraj Ammanabrolu, William Hancock, Shruti Singh, Brent Harrison, and Mark O. Riedl. 2017. Event representations for automated story generation with deep neural nets. arXiv:1706.01331.

Ashutosh Modi, Ivan Titov, Vera Demberg, Asad B. Sayeed, and Manfred Pinkal. 2017. Modeling semantic expectation: Using script knowledge for referent prediction. *TACL* 5:31–44.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, Kevin Duh, Manaal Faruqui, Cynthia Gan, Dan Garrette, Yangfeng Ji, Lingpeng Kong, Adhiguna Kuncoro, Gaurav Kumar, Chaitanya Malaviya, Paul Michel, Yusuke Oda, Matthew Richardson, Naomi Saphra, Swabha Swayamdipta, and Pengcheng Yin. 2017. DyNet: The dynamic neural network toolkit. arXiv:1701.03980.

Takeshi Onishi, Hai Wang, Mohit Bansal, Kevin Gimpel, and David A. McAllester. 2016. Who did what: A large-scale person-centered cloze dataset. In *EMNLP*. https://doi.org/10.18653/v1/d16-1241.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *EMNLP-CoNLL*.

Elena Rishes, Stephanie M. Lukin, David K. Elson, and Marilyn A. Walker. 2013. Generating different story tellings from semantic representations of narrative. In *ICIDS*.

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Meg Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015. A neural network approach to context-sensitive generation of conversational responses. In *NAACL*. https://doi.org/10.3115/v1/n15-1020.

Ilya Sutskever, James Martens, and Geoffrey E Hinton. 2011. Generating text with recurrent neural networks. In *ICML*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*.

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 *ICIDS*. https://doi.org/10.1007/978-3-642-25289-1_12.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *ICCV*. https://doi.org/10.1109/iccv.2015.11.