STORIUM: A Dataset and Evaluation Platform for Machine-in-the-Loop Story Generation

Nader Akoury[†*] Shufan Wang[†] Josh Whiting[‡] Stephen Hood[‡]
Nanyun Peng[§] Mohit Iyyer[†]
[†]University of Massachusetts Amherst [‡]Storium [§]University of California Los Angeles
{nsa,shufanwang,miyyer}@cs.umass.edu
{josh,stephen}@storium.com
violetpeng@cs.ucla.edu

Abstract

Systems for *story generation* are asked to produce plausible and enjoyable stories given an input context. This task is underspecified, as a vast number of diverse stories can originate from a single input. The large output space makes it difficult to build and evaluate story generation models, as (1) existing datasets lack rich enough contexts to meaningfully guide models, and (2) existing evaluations (both crowdsourced and automatic) are unreliable for assessing long-form creative text. To address these issues, we introduce a dataset and evaluation platform built from STORIUM, an online collaborative storytelling community. Our author-generated dataset contains 6K lengthy stories (125M tokens) with fine-grained natural language annotations (e.g., character goals and attributes) interspersed throughout each narrative, forming a robust source for guiding models. We evaluate language models fine-tuned on our dataset by integrating them onto STORIUM, where *real* authors can query a model for suggested story continuations and then edit them. Automatic metrics computed over these edits correlate well with both user ratings of generated stories and qualitative feedback from semi-structured user interviews. We release both the STORIUM dataset and evaluation platform to spur more principled research into story generation.

1 Introduction

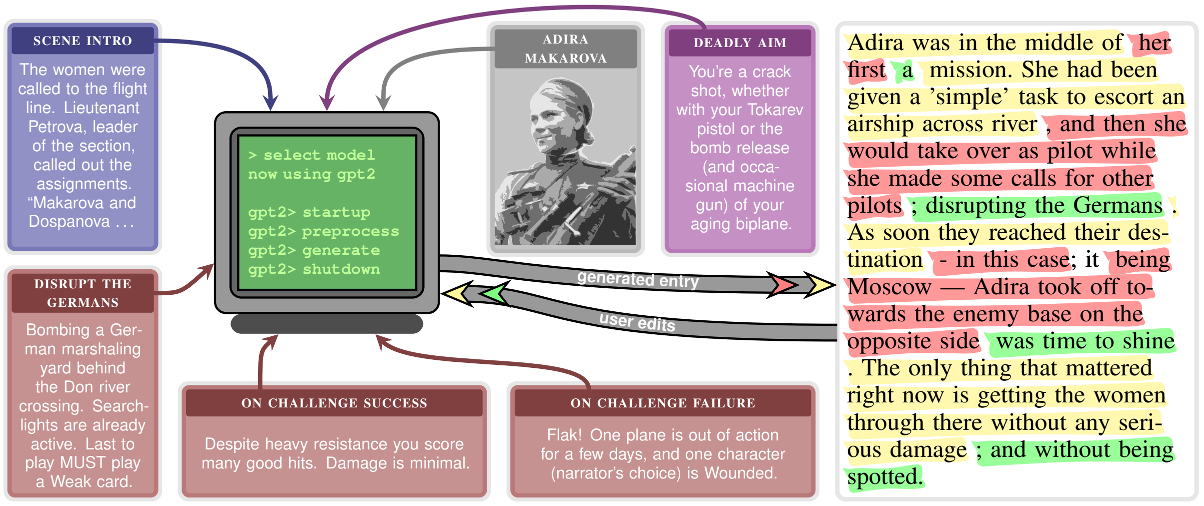
Fiction writers express their creativity through both low-level linguistic choices and discourse-level sequencing of narrative elements (e.g., plot events and character development). Unlike more constrained text generation tasks, such as translation or summarization, fiction writing allows for almost infinite creative freedom, which budding authors often find cognitively overwhelming (Rose, 1980). Machine-in-the-loop storytelling (Clark et al., 2018), in which an author obtains automatically generated sentences or paragraphs when stuck with writer's block, lowers the barrier to entry for creative writing (Roemmele and Gordon, 2015). To spur research in this area, we partner with STORIUM,[1] an online collaborative storytelling platform, to introduce a new dataset and evaluation methodology for story generation.

The open-endedness of story writing does not just pose a barrier to humans—it also presents a challenge for building and evaluating computational models. Prior work relies on datasets that are either too artificial to generalize to long-form stories, such as the crowdsourced ROCStories (Mostafazadeh et al., 2016) corpus, or too unconstrained, as in the `r/writingprompts` dataset (Fan et al., 2018), which pairs medium-length stories with short prompts. Furthermore, lack of standardized evaluation makes measuring progress difficult: most prior work evaluates outputs using a combination of simple automatic metrics not designed for long-form creative text generation (e.g., BLEU and ROUGE against a single reference) and crowdsourced ratings (McIntyre and Lapata, 2009; Yao et al., 2019; Fan et al., 2019) that preclude evaluating long-form narratives.

We address these limitations by (1) collecting a dataset of stories (Section 2) containing fine-grained structural annotations written in natural language, and (2) providing a platform for evaluating models in a machine-in-the-loop setting by allowing real STORIUM authors to interact with the generated stories (Section 4). Our dataset contains nearly 6K longform stories (125M tokens) written by STORIUM authors, each of which is broken into discourse-level scene entries annotated with narrative elements, such as character goals or abilities. Conditioning story generation models on this infor-

[1]`https://storium.com`

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 6470–6484,
November 16–20, 2020. ©2020 Association for Computational Linguistics



The women were called to the flight line. Lieutenant Petrova, leader of the section, called out the assignments. "Makarova and Dospanova . . .

You're a crack shot, whether with your Tokarev pistol or the bomb release (and occasional machine gun) of your aging biplane.

```
> select model
now using gpt2

gpt2> startup
gpt2> preprocess
gpt2> generate
gpt2> shutdown
```

Bombing a German marshaling yard behind the Don river crossing. Searchlights are already active. Last to play MUST play a Weak card.

Despite heavy resistance you score many good hits. Damage is minimal.

Flak! One plane is out of action for a few days, and one character (narrator's choice) is Wounded.

Adira was in the middle of her first a mission. She had been given a 'simple' task to escort an airship across river , and then she would take over as pilot while she made some calls for other pilots ; disrupting the Germans . As soon they reached their destination - in this case; it being Moscow — Adira took off towards the enemy base on the opposite side was time to shine . The only thing that mattered right now is getting the women through there without any serious damage ; and without being spotted.

Figure 1: A high-level outline of our dataset and platform. In this example from a real STORIUM game, the character ADIRA MAKAROVA uses the strength card DEADLY AIM to DISRUPT THE GERMANS, a challenge card. Our model conditions on the natural language annotations in the scene intro, challenge card, strength card, and character, along with the text of the previous scene entry (not shown) to generate a suggested story continuation. Players may then edit the model output, by adding or deleting text, before publishing the entry. We collect these edits, using the matched text as the basis of our USER metric. New models can be added to the platform by simply implementing four methods: `startup`, `shutdown`, `preprocess`, and `generate`.

mation thus imposes loose constraints on what the model should produce, compared to unstructured datasets such as `r/writingprompts`, and also enables modeling of narrative planning processes.

We fine-tune large-scale pretrained language models on our dataset (Section 3) and integrate them with the STORIUM platform, where authors can query a model for the next few sentences in their story and then edit the resulting text to their liking. We devise a metric (inspired by ROUGE) on top of these edits that measures how much of the generated text is preserved in the post-edited version, and discover that this metric correlates with Likert judgments of linguistic properties such as relevance and coherence. Detailed analyses of the edits (Section 5), including semi-structured interviews with STORIUM users, suggests that generating text *relevant* to the current story context is the most important open problem in this area. We publicly release both the STORIUM dataset and user-facing evaluation platform to facilitate future research on story generation.[2]

2 STORIUM Dataset

Our STORIUM dataset derives from an online collaborative storytelling community that provides rich metadata useful for guiding computational storytelling systems. In this section, we describe how the structural elements of STORIUM stories fit together, and verify via an annotation task that this metadata indeed influences the text of the stories. Finally, we use neural topic models to highlight the thematic content and narrative sequencing of STORIUM.

2.1 STORIUM: Gamified Storytelling

The STORIUM platform enables a small group of users to collaboratively write a single story by transforming the writing process into a turn-based game. In each game, one player acts as the *narrator*, while other players take on the role of individual *characters* within the story (e.g., ADIRA MAKAROVA in Figure 1). Stories unfold through a series of high-level *scenes* that consist of multiple short *entries*, each of which is written from the perspective of a character (or the narrator). Scenes commonly revolve around *challenges* (e.g., DISRUPT THE GERMANS), that the characters tackle within the text of their entries; to help address these challenges, each character has access to a set of *cards* (e.g., DEADLY AIM, a strength card) that define various properties such as strengths, weaknesses, items, and goals. The narrator moves the story forward by introducing new challenges, locations, and characters, in the form of cards. These are either created from scratch by the narrator or se-

[2]https://storium.cs.umass.edu

Dataset	# Stories	# Tokens per Story	Prompts	Turns	Annotations
roleplayerguild	1,439	3,079*	✗	✓	✗
PG-19	28,752	68,973	✗	✗	✗
ROCStories	98,156	88	✓	✗	✗
r/writingprompts	303,358	735	✓	✗	✗
STORIUM	5,743	19,278	✓†	✓	✓

Table 1: While STORIUM has fewer stories than other popular story datasets, each story is considerably longer and contains natural language annotations to guide story generation. *We combine character and action sets to determine average story length. †We count narrator actions introducing challenges and locations as prompts.

lected from a predefined *world* that contains a common set of story elements. Collectively, the cards played form a set of structural natural language annotations that guide the story being written.

Dataset details: We collect 5,743 publicly available stories written on STORIUM from January 2015 to August 2019. We reserve 569 stories for validation and 570 stories for test — carefully ensuring an 8:1:1 split with respect to both the number of stories and tokens. Altogether, the stories are broken down into 25,092 scenes with 448,264 individual scene entries (126,041,738 tokens), conditioned on 232,596 cards, 204,698 of which are unique.

Stories	5,743
Authors	30,119
Characters	25,955
Scenes	25,092
Scene Entries	448,264
Cards Played	232,596
Average Tokens* (per Entry)	247
Average Tokens* (per Story)	19,278
Total Tokens* (Entries & Cards)	126,041,738

Table 2: An overview of our dataset, which contains long stories, broken down into scene entries, with structural annotations in the form of cards played to guide the narrative. *We count tokens as contiguous spans of either alphanumeric or non-alphanumeric symbols.

Cards influence entry text: STORIUM does not force players to relate their written entries to selected cards or challenges, instead relying on game conventions to guide user behavior. To validate whether the structural metadata influences story text, we conduct a small-scale annotation of 235 scene entries, where we ask annotators[3] to provide binary judgments for (1) whether the card played influences the scene entry, and (2) if the scene entry addresses the current challenge. We find that 77% of scene entries reference the played cards, and 80% address the current challenge (Table A1).

Related datasets: Prior story generation papers have frequently focused on the ROCStories (Mostafazadeh et al., 2016) and r/writingprompts (Fan et al., 2018) datasets. While STORIUM has comparatively fewer stories than these datasets, our stories are over an order of magnitude longer (Table 1). Rather than containing a single short prompt to start the story, our stories on average contain 14 narrator prompts per story, with 41 natural language annotations which describe character goals, attributes, and key items useful for conditioning story generation models.[4] Like STORIUM, the stories in roleplayerguild (Louis and Sutton, 2018) are also formed from collaborative storytelling turns via a role-playing game, though this dataset lacks any prompts or annotations. Finally, datasets consisting of novels and other fiction, like PG-19 (Rae et al., 2020), provide long-form narratives without explicit structure to constrain generation.

2.2 Common Themes and Story Arcs

To provide insight into common narrative themes and substructures within our dataset, we train a neural topic model on text from entries and challenges and analyze the resulting topics and their transitions.

2.2.1 Topic model specification

Our topic model is a simplified version of the *relationship modeling network* (RMN) proposed

[3]The annotators were NLP graduate students.

[4]While Fan et al. (2019) extract internal structure via SRL, this is not inherent to the dataset, and can be applied to other datasets, including our own.

by Iyyer et al. (2016).[5] As in the RMN, our model relies on dictionary learning to compute topics; however, it models each entry and challenge independently, instead of considering the temporal order of scenes through recurrence. We ignore the temporal component because STORIUM contexts do not neatly fit into a chronologically-ordered timeline (e.g., entries within a single scene may not depend on each other). Building a specialized topic model for this data is beyond the scope of this work.

Concretely, given an input text (either an entry or a challenge), we first encode it by computing an average of pretrained GloVe[6] embeddings $\boldsymbol{x}$. Next, we compute the dot product between $\boldsymbol{x}$ and each row of a global *dictionary matrix* $\mathbf{R}$. Intuitively, each row of $\mathbf{R}$ is a vector representation of an individual topic. These row-wise dot products are converted to a probability distribution via a softmax function and then used to compute a weighted average $\boldsymbol{r}$ of the dictionary rows, which is then trained through a contrastive max-margin loss to reconstruct the input vector $\boldsymbol{z}$. At test time, the dictionary rows are interpreted by their nearest neighbors (using cosine distance) in the GLoVe word embedding space.[7]

Worlds	Topic words
Fantasy Classic	rotunda, courtyard, staircase, foyer
Urban Fantasy	analyze, investigate, analyse, uncover
The Mysterious Island	convoy, hiking, river, reconnaissance
Cyberpunk	synchronization, decryption, device, apparatus
Steampunk	freighter, crewmembers, cockpit, airship
The Heroes Return	thine, fealty, uphold, valor
Medical Drama	tumor, ligament, laceration, mortem
Los Chicos Malos	sublight, biosphere, aetheric, gravitational
The University	explanation, undergrad, spelling, reasoning
The 33	melodramatic, reenactment, film, thriller
Scrapjack	brake, soldering, heater, corrosion

Table 3: Topics with the highest relative importance for a sample of STORIUM worlds, which illustrate the diversity of the dataset.

[5]Preliminary experiments with LDA (Blei et al., 2003) yielded less coherent topics, which is consistent with evaluations in Iyyer et al. (2016).

[6]glove.840B.300d

[7]We encourage interested readers to see Iyyer et al. (2016) for more details. The only difference between our setup and theirs is that we directly use $\boldsymbol{x}$ to compute the row weights without any feed-forward or recurrent layers in between.

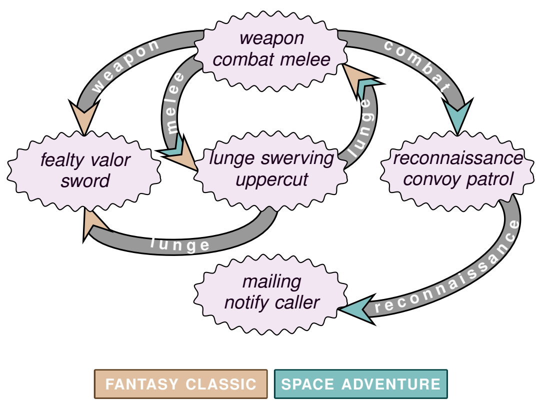


Figure 2: Example story arcs derived from the adjacency matrix of topic transitions over the text of entries (e.g., in FANTASY CLASSIC stories, the *weapon, combat, melee* topic is often followed by a transition, as denoted by weapon, to the *fealty, valor, sword* topic).

2.2.2 Examining topics and their transitions

To explore the content of the STORIUM dataset, we train our model with 50 topics (i.e., $\mathbf{R}$ has 50 rows) on the union of entry and challenge text. Table 3 shows the most distinguishing topic (ranked by relative importance) for a sample of different STORIUM worlds. These topics illustrate the diversity of our dataset: topics range from science fiction (*Cyberpunk*, *Steampunk*) to detective fiction (*Urban Fantasy*) and stories set in hospitals (*Medical Drama*) and schools (*The University*).

Following the methodology of Antoniak et al. (2019), we also examine common local topic transitions between entries written by the *same* character across different scenes in a story. We compute the transition probability from topic A to topic B by counting how many times A and B are the most probable topics for two consecutive entries, respectively, and normalizing by the total number of occurrences of topic A. Figure 2 shows a topic transition diagram originating from a weapons-related topic. In the *Space Adventure* world, stories progress into vehicle and technology-related topics, while in *Fantasy Classic*, they tend to transition to topics about valor instead. That said, both of these worlds are not completely different, as they share a transition topic associated with physical action.

3 Generating Scene Entries

We focus our modeling efforts on generating scene *entries*, which are the smallest units of each story, because we want to evaluate the generated text on

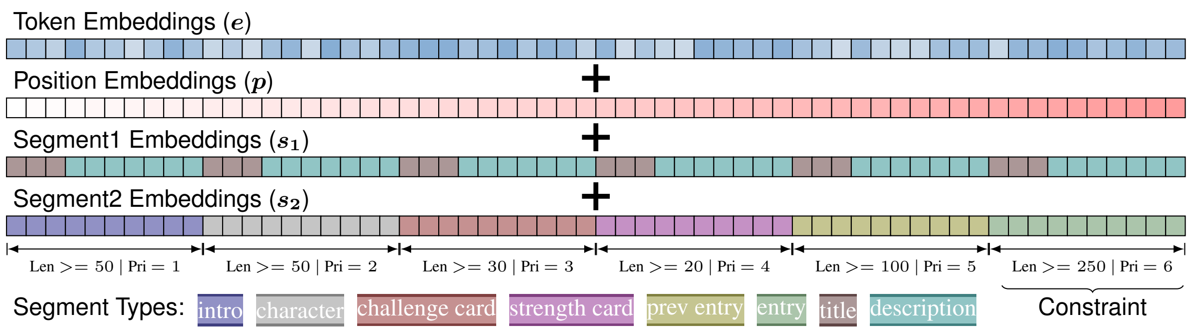


Figure 3: An illustration of our segment embeddings and packing strategy. In addition to token and position embeddings, common to all Transformer models, we employ compositional segment embeddings for conditioning on story metadata (e.g., DEADLY AIM is the title of a strength card). Each metadata segment has linear constraints with associated priorities (e.g., Len >= 30 | Pri = 3) for optimally packing tokens within the available space.

the STORIUM platform within a machine-in-the-loop framework.[8] Our method relies on fine-tuning a pretrained language model (GPT-2) on the STORIUM dataset using segment embeddings to differentiate each type of context. While GPT-2 has successfully been used as a state-of-the-art model for story generation (Mao et al., 2019; Guan et al., 2020), one crucial challenge is the length of the contexts: each entry in a story can condition on any narrative element that comes before it (e.g., previous entries, scenes, challenges). Thus, the number of context tokens quickly grows larger than what is feasible to fit in GPU memory. Another challenge lies in how to properly tune hyperparameters in a machine-in-the-loop setting, as it is infeasible to obtain human judgments for a huge number of configurations. The rest of this section fully specifies our model, a token-packing strategy to optimize use of the input context, and preliminary user-facing experiments that helped us decide on our final model hyperparameters.

3.1 Model Specification

We fine-tune the GPT-2 medium-sized (355M parameters) language model (Radford et al., 2019) for story generation, as it has been shown to generate coherent long-form prose. Before fine-tuning, we need to account for the complexity of STORIUM contexts: each scene consists of multiple entries, each of which may reference a different number of semi-structured cards (e.g., both the DEADLY AIM strength card and the ADIRA MAKAROVA character in Figure 1 contain a title and description). To handle the compositional and semi-structured nature of the scenes and cards, we allow each input token to condition on an arbitrary number of *segment embeddings* (Wolf et al., 2019) (Figure 3). Concretely, we augment the token vocabulary V of GPT-2 with a segment vocabulary S for delineating each segment. The final embedding vector $\boldsymbol{e}_i$ at position i is computed by summing the token embedding $\boldsymbol{v}_i$ with the positional embedding $\boldsymbol{p}_i$ and the corresponding set of n segment embeddings $\{\boldsymbol{s}_{i_1}, \dots, \boldsymbol{s}_{i_n}\}$:

$$e_i = p_i + v_i + \sum_{m=1}^{n} s_{i_m} \qquad (1)$$

During training, a single input instance to our models contains the text of the current entry, its associated challenge, card metadata, as well as the current character's biography and the scene's introductory text (Figure 1). Our final model also includes the text of the immediately preceding story entry,[9] which improves human and automatic evaluation scores (Table 4). At test time, we provide only the story context and autoregressively sample a scene entry.

3.1.1 Context packing

The average story in our dataset has over 19K tokens broken up into 78 scene entries, which is much longer than GPT-2's maximum sequence length of 1024 tokens. We thus face the challenge of how best to optimize our usage of the limited input space, which is made more difficult by the many different types of input context (e.g., entries, characters, challenges) within STORIUM. Naïvely reserving a fixed number of tokens per context type

[8]Our dataset also enables modeling high-level decisions made by the narrator, such as challenge sequencing; we leave this for future work.

[9]If the preceding entry is not written by the current character, we also include the current character's last entry.

wastes significant space, as the number and length of metadata instances varies considerably per entry. For example, some scene entries do not make use of cards (Table 2), while others reference multiple cards.

Our solution applies the Cassowary algorithm (Badros et al., 2001), well-known for arranging UI elements in Apple's iOS, to pack the input tokens more efficiently. Cassowary allows for efficiently solving linear equality and inequality constraints incrementally, using a dual simplex based method. We define a set of linear constraints on the size of each metadata segment (e.g., include at least 250 tokens from an entry when possible), and Cassowary's solver produces an optimal arrangement of context tokens with respect to these constraints (Figure 3). Compared to naïvely packing tokens into fixed length segments, Cassowary allows us to vary the minimum and maximum bounds on segments, as well as collapse missing segments. This flexibility results in increased human and automatic evaluation scores (Table 4).

3.2 Hyperparameter Selection

Before launching our full machine-in-the-loop evaluation, we conduct preliminary experiments on the STORIUM platform to validate our design choices. Since we want *real* users on STORIUM to enjoy interacting with the generated text, we want to avoid alienating them with poorly performing models. We measure the impact of (1) including history information from the immediately preceding entry in the story, and (2) using Cassowary to densely pack the context. In total, we fine-tune four models on the Cartesian product of these complementary modeling ideas, keeping all other hyperparameters constant, and deploy these models to STORIUM.

The results (Table 4) highlight the importance of both modeling choices: after including more story history and applying the Cassowary solver, validation perplexity decreases while STORIUM user ratings of fluency, coherence, relevance, and likability all increase. This motivates us to use only the best-performing model for the full-scale evaluation. Additionally, user feedback from these experiments suggested that we generate *shorter* entries, as longer ones frequently devolved into unrelated and incoherent sentences. Thus, for our final experiments detailed in the next section, we also truncate model outputs to a maximum of four sentences.

Cas	His	F	C	L	R	Ppl	Jdg
		3.4	2.9	3.8	2.3	25.1	90
	✓	3.1	2.7	3.9	2.3	22.4	77
✓		3.6	2.8	3.6	2.4	22.9	62
✓	✓	**3.7**	**3.2**	**4.1**	**2.7**	**21.0**	85

Table 4: Exploratory experiments indicate optimally packing tokens using Cassowary (Cas), and including more history (His) is key to achieving low perplexity (Ppl), along with high fluency (F), coherence (C), likability (L), and relevance (R) based on a number of user judgments (Jdg).

4 A Machine-in-the-Loop Evaluation Platform

The inadequacies of existing human and automatic evaluation methods are a major roadblock for story generation research. Automatic evaluations correlate weakly with human judgments (Sagarkar et al., 2018), and these judgments are obtained from crowd workers who are not invested in the narratives they are assessing. These concerns are magnified with STORIUM, as the story contexts are far too long for crowd workers to reliably evaluate (Section 5). In this section, we propose an improved evaluation methodology by directly integrating our models onto the STORIUM platform. This allows story *authors* to query a machine (Clark et al., 2018) for suggestions during the process of writing their own stories. We develop a new evaluation metric, User Story Edit Ratings (USER), computed on top of the *edits* that STORIUM users make to generated entries. Finally, we provide experimental results that compare two configurations of our best model from Section 3.2.

4.1 Evaluation Lifecycle

To evaluate generated stories, we develop a dedicated web service for serving model outputs to the STORIUM platform. STORIUM users simply press a button on the user interface to obtain a generated scene entry conditioned on the story context. Users can then add new text while deleting any of the generated text that they wish (Figure 1). When users publish their edited entry, they are also asked to evaluate the generated text on a 5-point Likert scale[10] with respect to *relevance* (fit with the current story), *fluency* (judgment of grammaticality), *coherence* (logical ordering of sentences), and *likability* (subjective assessment of enjoyability). This process allows experts (STORIUM authors)

[10]They also provide optional freeform comments on generated text; we leave analysis of the comments to future work.

to evaluate generated stories, which is a substantial improvement over prior evaluation efforts. We make our evaluation platform publicly accessible for researchers to develop and integrate their own models. Our framework makes adding a new model using any Python-based deep learning framework very easy, requiring implementation of only four methods: `startup`, `shutdown`, `preprocess`, and `generate`.

4.2 A Metric Over User Edits

Intuitively, the amount of generated text that a user preserves in their final published entry clearly indicates the usefulness of the generated text. We quantify this by developing User Story Edit Ratings (USER), inspired by the longest common subsequence (LCS) variant of ROUGE (Lin, 2004), applied to user edits. Given a generated entry X and the final published entry Y, we compute $\text{USER}(X, Y) = \frac{|\text{MATCH}(X,Y)|}{|X|}$, where $\text{MATCH}(X, Y)$ considers *contiguous substrings* with at least one non-stopword as matches (see Figure 1 for an example and Appendix C for a more thorough treatment). We do not use ROUGE-L because vanilla LCS typically favors subsequences of unigram matches (often stopwords) over longer contiguous n-gram matches. In our STORIUM setting, users preserving n-grams or full sentences is a clear indication that the generated text was useful.

5 Analysis

Compared to existing work on story generation, the main novelty of our STORIUM evaluation platform is that it enables authors to interact directly with model-generated text through their edits. In this section, we conduct experiments on our platform and analyze the edits by examining the correlation of USER to Likert scores. We explore linguistic properties of text that users preserve and also conduct a crowdsourced evaluation on Amazon Mechanical Turk that demonstrates its unsuitability for this task. Finally, we qualitatively describe feedback obtained from interviews with ten STORIUM users who engaged with our models, which provides a roadmap for future work.

Top-k vs. nucleus sampling: Using our platform (Section 4), we evaluate our best model (Table 4) with two different decoding strategies: (1) top-k sampling (Fan et al., 2018) with $k = 40$, and (2) nucleus sampling (Holtzman et al., 2020) with $p = 0.9$.[11] The sampling parameters, such as the k in top-k sampling, can significantly affect output quality of story generation models (See et al., 2019), so we choose values that worked well in prior work (Qin et al., 2019).[12]

		Lik	Flu	Coh	USER	Rating
Rel	top-k	0.51	0.28	0.55	0.51	**2.55**
	nucleus	0.53	0.40	0.57	0.39	2.47
Lik	top-k	—	0.28	0.35	0.34	**3.32**
	nucleus	—	0.38	0.55	0.35	3.21
Flu	top-k	—	—	0.54	$0.13^\dagger$	**3.96**
	nucleus	—	—	0.61	0.23	3.76
Coh	top-k	—	—	—	0.25	**3.41**
	nucleus	—	—	—	0.36	2.96
USER	top-k	—	—	—	—	**15.63**
	nucleus	—	—	—	—	9.86

Table 5: Despite its low rating, relevance is clearly important as indicated by the moderately strong Pearson's r correlations (first four columns) with USER and the remaining human judgments. All correlations are significant ($p < 0.01$), except those indicated by † ($p > 0.05$).

Interestingly, while Holtzman et al. (2020) show that nucleus sampling improves over top-k sampling on measures like repetition, STORIUM users clearly prefer the top-k variant across all categories (last column of Table 5). We collect roughly 200 feedback ratings and 175 edits for each model over a span of three months beginning in late February 2020. We discover that both configurations score best on *fluency* and worst on *relevance*. This is unsurprising as (1) GPT-2 is known to produce fluent text and (2) the complex and lengthy STORIUM data is a challenge for limited-context models. Finally, USER scores are generally low (15.6 for top-k vs. 9.9 for nucleus sampling), indicating that users delete most of the current model's generated text. This result demonstrates that story generation models still have a long way to go.[13]

USER correlates with human judgments: A natural question is whether our USER metric correlates with judgments of fluency, coherence, relevance, and likability. Table 5 shows that for the top-k configuration, relevance has a significantly higher correlation (Pearson's r) with USER than the other properties. In other words, users are most

[11] We use a temperature of 0.9, a repetition penalty (Keskar et al., 2019) of 1.2, and an analogous length penalty that dynamically penalizes producing the end of sequence token inversely proportionally to a desired length l_d.

[12] It is possible that a better set of sampling hyperparameters exists, which we leave to future work.

[13] See the supplementary HTML for an export of all results (including generated text and edits) used for this paper.

	Top-k		Nucleus	
First Run	Rating	κ	Rating	κ
Fluency	**3.59**	0.17	3.47	0.11
Coherence	**3.50**	0.10	3.44	0.20
Likability	**3.27**	0.07	3.22	0.11
Relevance	**3.32**	0.09	3.27	0.13

	Top-k		Nucleus	
Second Run	Rating	κ	Rating	κ
Fluency	**4.01**	0.46	3.77	0.33
Coherence	**3.63**	0.27	3.38	0.23
Likability	**3.28**	0.12	3.06	0.16

Table 6: Despite our best efforts, our first crowd sourced judgments show low agreement (κ) on open-ended story generation. Our second run, which removes context, thus excluding relevance judgments, greatly increases agreement for fluency and coherence.

likely to preserve generated text when it is relevant to the overall story. Fluency correlates only weakly with USER, which makes sense as most generated entries are fluent due to GPT-2's pretraining. Finally, nucleus sampling exhibits lower correlation for relevance, but higher correlation for the other three properties, possibly due to its lower average scores for these properties (see Appendix C for a comparison of USER to ROUGE-based metrics).[13]

Linguistic properties of preserved text: Knowing that users delete most of the generated text, we instead explore the linguistic commonalities of the preserved text. We run spaCy part-of-speech tagging and named entity recognition (Honnibal and Montani, 2017) over the edited entries. Strikingly, 29.5% of generated proper nouns are preserved in the edited text, compared to only 13.5% for all other POS tags. A major confound is that our model could unfairly receive credit for simply copying character names from the input context, as users are likely to write about these characters anyway. To measure the extent of this effect, we match all generated named entities that users preserve to predefined character lists from each story, and discover that 63% of generated entities already exist within the story context. The remaining 37% of entities are often completely new character names. User interviews also suggest that this ability to generate new names is a useful feature.

Crowdsourced evaluation is unreliable: Thus far, we have argued for our evaluation platform by claiming that crowdsourced methods are unsuitable for evaluating stories with complex and lengthy contexts. Here, we measure fluency, coherence, relevance, and likability of our generated entries with a crowdsourced Amazon Mechanical Turk task, to see if the results correspond to STORIUM user ratings. Designing this crowdsourced task is difficult, as we cannot show crowd workers the entire story context due to its length; we thus decide to show the same inputs that the model receives (Section 3). We collect ratings of 100 examples per model, with three judgments per example.[14]

Table 6 (top) shows that workers have very low agreement (Fleiss' κ) for all properties, including even fluency. An analysis of the median task completion time[15] reveals most workers did not actually read the context. We run a second experiment, showing only the generated text (no context), and remove the relevance rating. Table 6 (bottom) shows this improves agreement (Table 6), and that the average fluency scores align closely with those from STORIUM users. Overall, our struggle to obtain quality judgments from Mechanical Turk further validates our platform: STORIUM provides free expert judgments from people invested in storytelling.

Feedback from user interviews: To better understand the strengths and weaknesses of our current model, we conduct semi-structured interviews with ten STORIUM users. Most were surprised with the overall fluency of our models. This partly explains the low correlation of fluency with USER. Relevance was mentioned by 9 out of 10 users as the number one area of improvement for our model, confirming our experimental results (Table 5). Four users called out the model's tendency to fabricate facts and introduce new characters. Despite these concerns, three users explicitly stated the model inspired them to write or found portions of the generated text useful, though mostly as a source for character and place names (supporting the linguistic analysis in Section 5). Finally, some users considered the system a curiosity and decided to write stories using only generated text (without edits).[16]

[14]We limit annotations to crowd workers living in the US and the UK, with over 1000 completed annotations and a 99% approval. We pay \$0.50 per annotation, by assuming 2 minutes per annotation, for an effective hourly rate of \$15.

[15]Mechanical Turk automatically reports a *WorkTimeInSeconds* field for each annotation, which is ten minutes on average for our task — more than enough time to read and assess the generated entry and associated context. Sadly, this interval is misleading. Analyzing the median time between submits, we see workers accept multiple concurrent tasks, wait a few minutes, then submit each annotation in quick succession, thus inflating the *WorkTimeInSeconds* interval.

[16]These AI-guided narratives are prevalent enough that we manually exclude these games from our experiments as they

6 Related Work

Our work builds on prior research in computational modeling for story generation. Early narrative prose generation systems (Meehan, 1977; Callaway and Lester, 2001; Riedl and Young, 2004) relied on graph-based planning formalisms and custom rules to structure their narratives, while story graphs have been used for interactive storytelling (Riedl and Bulitko, 2013). More recent work uses deep learning to generate stories by training neural models with limited context (Peng et al., 2018; Fan et al., 2018; Goldfarb-Tarrant et al., 2019) and structured knowledge, either external (Mao et al., 2019; Guan et al., 2020; Goldfarb-Tarrant et al., 2020) or derived (Yao et al., 2019; Fan et al., 2019). Compared to the datasets studied in those works, our STORIUM dataset contains much longer stories with built-in structural annotations written in natural language in the form of cards (Table 2).

Our work connects more closely to existing machine-in-the-loop storytelling work (Roemmele and Gordon, 2015; Samuel et al., 2016; Clark et al., 2018), in which systems work in concert with users to collaboratively author a narrative. Much like the Creative Help platform of Roemmele and Gordon (2015), we provide writing assistance by interactively generating continuations of STORIUM stories. We improve over Roemmele and Gordon (2015) by evaluating a trained model (instead of a retrieval-based approach) with a large user population.

Finally, our STORIUM evaluation takes a different approach to prior research that measures the quality of generated stories. Sagarkar et al. (2018) train an automatic scorer on human annotations of overall story quality, relevance, and interestingness based on evaluation criteria from (McIntyre and Lapata, 2009). See et al. (2019) consider a number of *diversity* related measures for automated evaluation of story generation systems by focusing on the GPT-2 small model, noting that *quality* assessments are still best measured through human evaluation.

Limitations

Evaluating on the STORIUM platform enables researchers to receive high-quality judgements on the outputs of their story generation models. These judgements are made possible by the significant time and effort spent by real authors on crafting their narratives, as their incentives are substantially different than those of crowdsourced workers. The amount of author effort involved in evaluation, when combined with the relatively small size of the STORIUM community, can cause evaluation to take a considerable amount of time (i.e., to collect hundreds of judgements) as evidenced in our analysis (Section 5). Thus, our platform is not currently suitable for "instant" evaluation of generated stories. Furthermore, as the evaluation platform is specifically deployed on STORIUM, it cannot be trivially used to evaluate models trained on other story generation datasets, as users of the website are mainly invested in writing narratives that follow the STORIUM format.

artificially increase the automatic metrics.

7 Conclusion

We introduce the STORIUM dataset and evaluation platform for machine-in-the-loop story generation, built from an online collaborative storytelling community. STORIUM contains 6K long stories annotated with structural metadata useful for conditioning language models. Importantly, *real* STORIUM *authors* evaluate model outputs by adding and removing text to create their own stories. We devise a metric on top of their edits that correlates strongly with judgments of the *relevance* of the generated text, which user interviews suggest is the most important area for improvement moving forward. Our dataset and evaluation platform will be made publicly available to spur progress into story generation.

Author Contributions

Dataset Analysis: Akoury, Wang
Generation Model: Akoury, Wang
Evaluation Platform: Akoury, Whiting, Hood
Research Guidance: Iyyer, Peng

Acknowledgements

We thank the wonderful STORIUM users for actively using our story generation models and generously providing their time to be interviewed. We also thank the amazing UMass NLP community for thoughtful insights on our paper and helping to validate whether structural metadata influences story text on STORIUM. Akoury and Iyyer were supported during this project by a research gift from Genpact. Peng was supported in part by the CwC program under Contract W911NF-15-1-0543 with the US Defense Advanced Research Projects Agency (DARPA).

References

Maria Antoniak, David Mimno, and Karen Levy. 2019. Narrative paths and negotiation of power in birth stories. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work and Social Computing*.

Greg J. Badros, Alan Borning, and Peter J. Stuckey. 2001. The cassowary linear arithmetic constraint solving algorithm. *ACM Trans. Comput. Hum. Interact.*, 8:267–306.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*.

Charles B. Callaway and James C. Lester. 2001. Narrative prose generation. *Artif. Intell.*, 139:213–252.

Elizabeth Clark, Anne Spencer Ross, Chenhao Tan, Yangfeng Ji, and Noah A. Smith. 2018. Creative writing with a machine in the loop: Case studies on slogans and stories. *23rd International Conference on Intelligent User Interfaces*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the Association for Computational Linguistics*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2019. Strategies for structuring story generation. In *Proceedings of the Association for Computational Linguistics*.

Markus Freitag, David Grangier, and Isaac Caswell. 2020. Bleu might be guilty but references are not innocent. *ArXiv*, abs/2004.06063.

Seraphina Goldfarb-Tarrant, Tuhin Chakrabarty, Ralph Weischedel, and Nanyun Peng. 2020. Content planning for neural story generation with aristotelian rescoring. In *the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Seraphina Goldfarb-Tarrant, Haining Feng, and Nanyun Peng. 2019. Plan, write, and revise: an interactive system for open-domain story generation. In *NAACL-HLT, system demonstration*.

Jian Guan, Fei Huang, Zhihao Zhao, Xiaoyan Zhu, and Minlie Huang. 2020. A knowledge-enhanced pretraining model for commonsense story generation. *Transactions of the Association for Computational Linguistics*, 8:93–108.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. *iclr*.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Mohit Iyyer, Anupam Guha, Snigdha Chaturvedi, Jordan L. Boyd-Graber, and Hal Daume III. 2016. Feuding families and former friends: Unsupervised learning for dynamic fictional relationships. In *NAACL-HLT*.

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *ArXiv*, abs/1909.05858.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *ACL 2004*.

Annie Louis and Charles Sutton. 2018. Deep Dungeons and Dragons: Learning Character-Action Interactions from Role-Playing Game Transcripts. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 708–713, New Orleans, Louisiana. Association for Computational Linguistics.

Huanru Henry Mao, Bodhisattwa Prasad Majumder, Julian McAuley, and Garrison Cottrell. 2019. Improving neural story generation by targeted common sense grounding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5988–5993, Hong Kong, China. Association for Computational Linguistics.

Neil Duncan McIntyre and Mirella Lapata. 2009. Learning to tell tales: A data-driven approach to story generation. In *ACL/IJCNLP*.

James R. Meehan. 1977. Tale-spin, an interactive program that writes stories. In *IJCAI*.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Nanyun Peng, Marjan Ghazvininejad, Jonathan May, and Kevin Knight. 2018. Towards controllable story generation. In *Proceedings of the First Workshop on Storytelling*, pages 43–49, New Orleans, Louisiana. Association for Computational Linguistics.

Lianhui Qin, Antoine Bosselut, Ari Holtzman, Chandra Bhagavatula, Elizabeth Clark, and Yejin Choi. 2019. Counterfactual story reasoning and generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5043–5053, Hong Kong, China. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. 2020. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*.

Mark O. Riedl and Vadim Bulitko. 2013. Interactive narrative: An intelligent systems approach. *AI Magazine*, 34:67–77.

Mark O. Riedl and Robert Michael Young. 2004. An intent-driven planner for multi-agent story generation. *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, 2004. AAMAS 2004.*, pages 186–193.

Melissa Roemmele and Andrew S Gordon. 2015. Creative help: a story writing assistant. In *International Conference on Interactive Digital Storytelling*.

Mike Rose. 1980. Rigid rules, inflexible plans, and the stifling of language: A cognitivist analysis of writer's block. *College Composition and Communication*, 31(4).

Manasvi Sagarkar, John Wieting, Lifu Tu, and Kevin Gimpel. 2018. Quality signals in generated stories. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 192–202, New Orleans, Louisiana. Association for Computational Linguistics.

Ben Samuel, Michael Mateas, and Noah Wardrip-Fruin. 2016. The design of writing buddy: A mixed-initiative approach towards computational story collaboration. In *ICIDS*.

Abigail See, Aneesh Pappu, Rohun Saxena, Akhila Yerukola, and Christopher D Manning. 2019. Do massively pretrained language models make better storytellers? In *Conference on Computational Natural Language Learning*.

Thibault Sellam, Dipanjan Das, and Ankur P. Parikh. 2020. Bleurt: Learning robust metrics for text generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *ArXiv*, abs/1901.08149.

Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *Association for the Advancement of Artificial Intelligence*.

Appendix

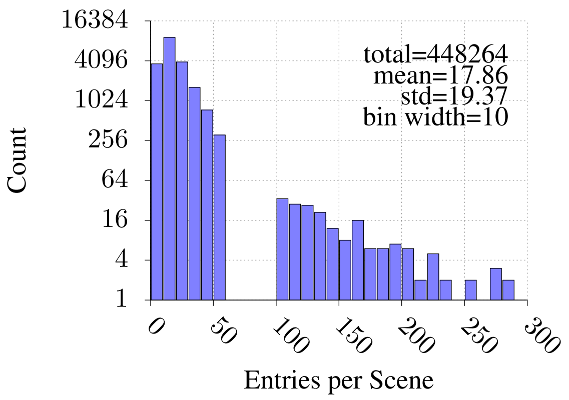
A Additional Dataset Statistics

As our dataset derives from a collaborative storytelling game that is highly compositional by nature, it is difficult to concisely capture the full scope of the data within the main body. Here we highlight the full results of our small scale annotation that indicates cards influence the scene entry text.

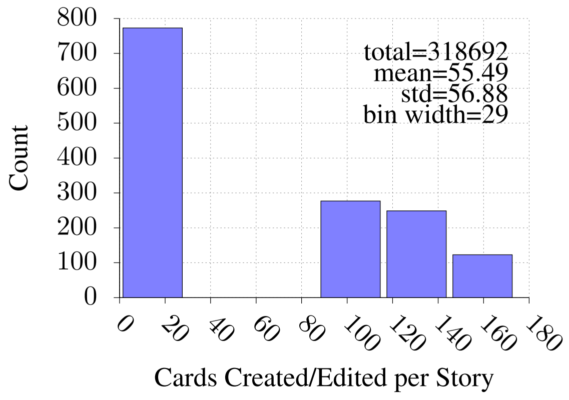
Total Annotations	248
Valid Entries†	235
Card Influences Entry	182
Entry Addresses Challenge	189
Card Influence ∩ Challenge Addressed	151

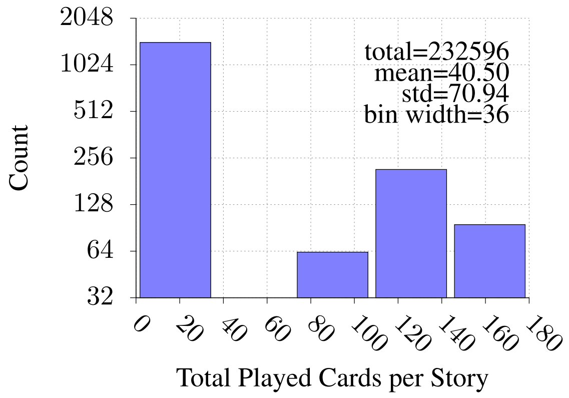
Table A1: We ask annotators to determine how frequently cards influence an entry, and if the entry addresses the challenge. †Annotators were asked to flag stories not written in English or otherwise could not be understood.

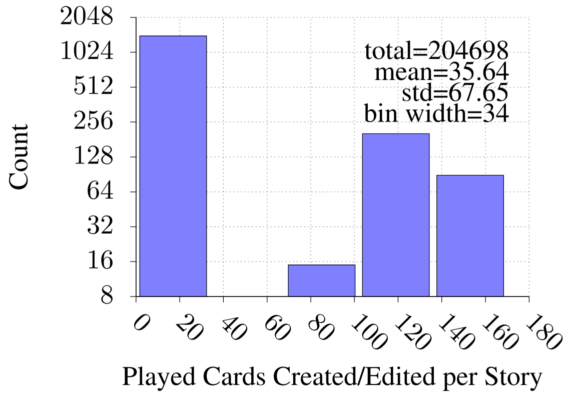
Additionally, there are many small details which are important distinctions in the game, but may not require separate modeling for generating a scene entry. For example, there is a distinction between regular cards, which have a fixed title and description provided by the narrator; versus wild cards, which allow individual characters to write their own title and description. For the sake of completeness, we provide Table A2 to help further explore the depths of this unique dataset. The following histograms[1] further break down the data in Table A2, clearly demonstrating the long tail distributions indicative of user generated stories:

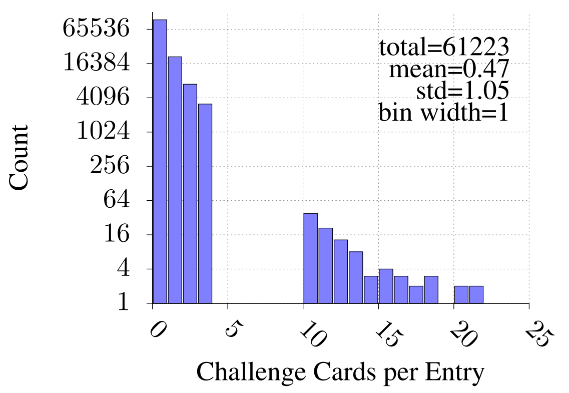


[1] These histograms provide context for the meaning of **Mean** and **Std Dev** for Table A2.



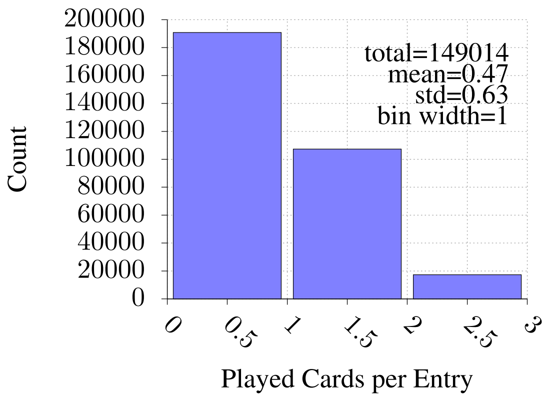


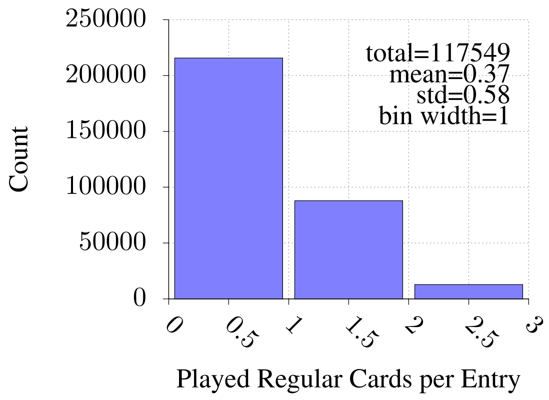


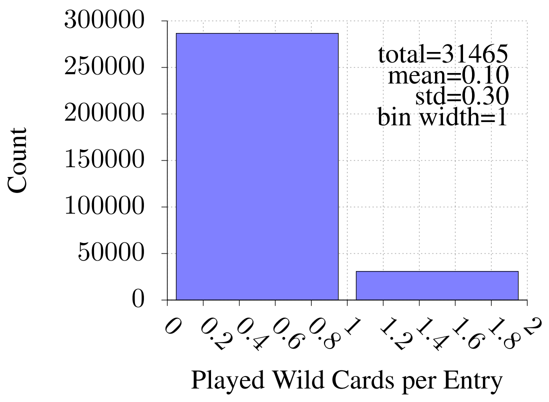


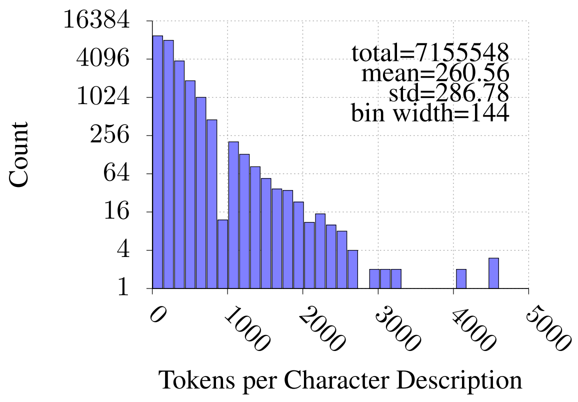
Feature	**Total**	**Mean**[1]	**Std Dev**[1]
Stories	5,743	—	—
Completed Stories	586	—	—
Users	30,119	—	—
Characters Created	27,462	4.78	3.04
Characters Played	25,955	4.52	3.04
Total Played Roles	31,698	—	—
Scenes	25,092	4.37	6.96
Scene Entries	448,264	17.86	19.37
Cards Created/Edited	318,692	55.49	56.88
Total Played Cards by Users	232,596	40.50	70.94
Played Cards Created/Edited by Users	204,698	35.64	67.65
Location Cards Played by Narrators	16,887	0.67	0.47
Challenge Cards Played by Narrators	61,223	0.47	1.05
Cards Played by Characters	149,014	0.47	0.63
Wild Cards Played by Characters	31,465	0.10	0.30
Regular Cards Played by Characters	117,549	0.37	0.58
Stories Played Without Cards	736	—	—
Tokens in Character Descriptions	7,155,548	260.56	286.78
Tokens in Scene Entries	110,772,426	247.11	307.13
Tokens in Played Location Cards	438,044	0.98	6.14
Tokens in Played Challenge Cards	3,837,860	24.84	15.30
Tokens in Played Regular Cards	3,053,152	25.08	15.78
Tokens in Played Wild Cards	784,708	23.96	13.32
Unique Tokens	424,768	—	—

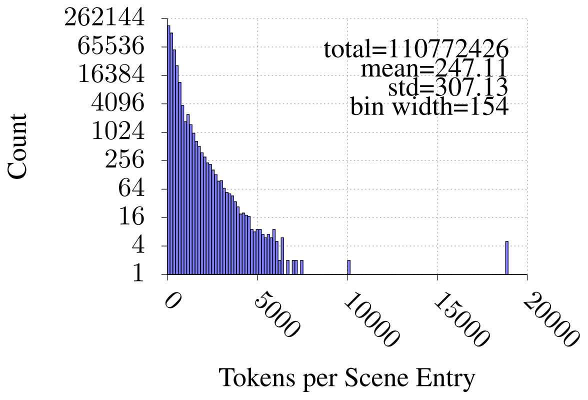
Table A2: A small look at the highly compositional nature of our dataset.

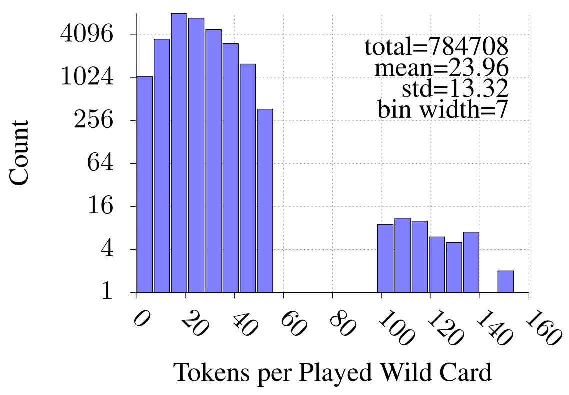


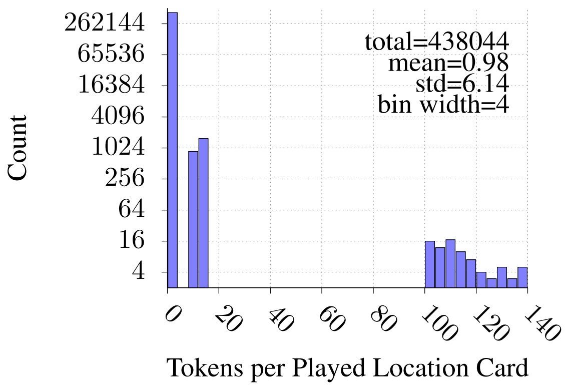


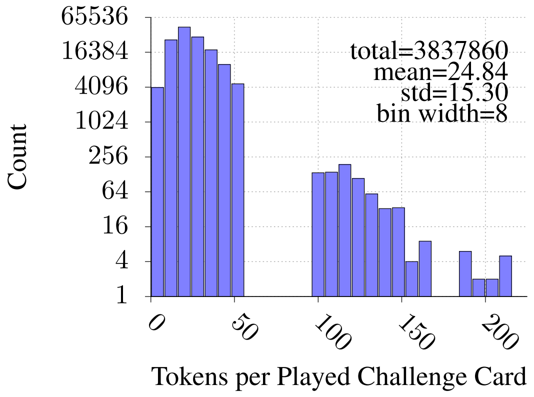


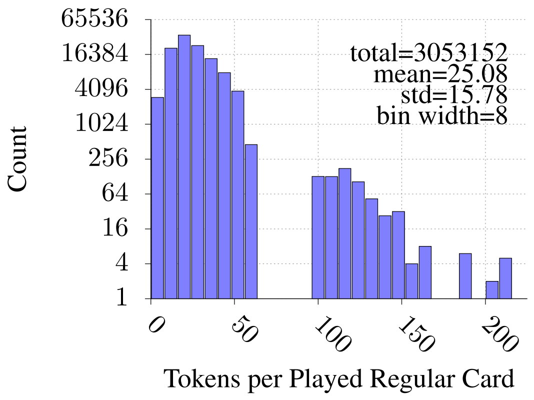












B Web Service

Our web service is modular and allows easily adding new models. It consists of a frontend service, which acts as a mediator between STORIUM and each backend service responsible for serving model outputs. The frontend stores data in a PostgreSQL database and provides a dashboard for viewing realtime ratings and evaluation metrics. It also displays user comments, scene entry diffs based on user edits, and Pearson's r correlations among metrics and user ratings — all sortable per model. A new model can be served by simply implementing four methods (`startup`, `shutdown`, `preprocess`, and `generate`). The backend automatically installs all Python requirements for serving a model and is agnostic to the underlying tensor library used. Additionally, we follow the latest best practices, including the use of Docker containers and the Asynchronous Server Gateway Interface (ASGI)[2],the latest Python web standard, which allows for asynchronous programming using `asyncio`.[3] We host the web service using an on-premise server with four 2080Ti GPUs.

C User Story Edit Ratings

Recently, the discriminative power of BLEU has been called into question when evaluating state-of-the-art machine translation systems, leading researchers to investigate alternative evaluation metrics (Freitag et al., 2020; Sellam et al., 2020). Similarly, we question the use of ROUGE metrics for automatic evaluation of open-ended story generation. Using our evaluation platform, we show that USER improves upon ROUGE in the story generation domain.

[2]FastAPI (https://fastapi.tiangolo.com)

[3]https://docs.python.org/3/library/asyncio.html

	Likability		Fluency		Coherence		ROUGE-L		ROUGE-W		USER	
	top-k	nuc	top-k	nuc	top-k	nuc	top-k	nuc	top-k	nuc	top-k	nuc
Relevance	0.51	0.53	0.28	0.40	0.55	0.57	0.52	0.38	0.50	0.36	0.51	0.39
Likability	—		0.28	0.38	0.35	0.55	0.29	0.34	0.28	0.31	0.34	0.35
Fluency	—		—		0.54	0.61	$0.11^{\dagger}$	0.23	$0.10^{\dagger}$	0.22	$0.13^{\dagger}$	0.23
Coherence	—		—		—		0.27	0.38	0.24	0.34	0.25	0.36
ROUGE-L	—		—		—		—		0.98	0.98	0.95	0.93
ROUGE-W	—		—		—		—		—		0.97	0.94

Table A3: USER correlates well with both ROUGE-L and ROUGE-W when removing stopwords.

When evaluating story continuations, we cannot compare against an *a priori* gold standard. Rather, we consider the final published story a user generates to be the gold standard, and thus evaluate models by how much text the user retains. Using ROUGE-L *precision*, which simply computes the ratio of the longest common subsequence (LCS) with the number of tokens in the generated text, we can measure this quantity.

As highlighted by Lin (2004), ROUGE-L contains a subtle mismatch with expectations, as the LCS does not consider *locality* of matches — assigning equal weight to subsequences of the same length even when the distance between matched words differs. Given a reference sequence X, the following two candidate sequences Y_1 and Y_2 produce the same ROUGE-L score (an underscore indicates a subsequence match):

$$\begin{aligned} X &: [\underline{A}\ \underline{B}\ \underline{C}\ \underline{D}\ E\ F\ G] \\ Y_1 &: [\underline{A}\ \underline{B}\ \underline{C}\ \underline{D}\ H\ I\ K] \\ Y_2 &: [\underline{A}\ H\ \underline{B}\ K\ \underline{C}\ I\ \underline{D}] \end{aligned}$$

ROUGE-W tries to address this shortcoming by introducing a weighting which favors subsequences with less separation. Sadly, for long texts, both ROUGE-L and ROUGE-W often favors long *subsequences* of stopwords over *contiguous substrings*, a sign that a user clearly used part of the output unchanged. While acceptable for short summaries, this is much less appropriate for long-form open-ended text generation. Removing stopwords helps alleviate the mismatch, so we do so in our comparison to ROUGE (Table A4), though the fundamental issue still remains. This mismatch calls into question the ability of ROUGE-L and ROUGE-W to distinguish among models with strong story generation capability.

	Top-k		Nucleus	
	Score	Count	Score	Count
ROUGE-L	**28.61**	174	20.66	178
ROUGE-W	**20.73**	174	13.80	178
USER	**15.63**	174	9.86	178

Table A4: USER produces lower scores on average than ROUGE-L or ROUGE-W.

Our new metric, User Story Edit Ratings (USER), is based on a diff-like approach. We begin by applying the same text preprocessing as ROUGE. Afterwhich, we find the longest contiguous substring, then use it as a pivot to divide the remaining string into two halves (excluding the pivot), and recursively repeat the process in each half.[4] We then only consider substrings with at least one non-stopword as matches (careful scrutiny of Figure 1 reveals an unmatched stopword *it*). Subsequently, we compute precision, recall, and F1 identically to ROUGE.

Table A3 shows USER correlates with user judgments approximately similarly to ROUGE metrics, while correlating strongly with both metrics. Additionally, USER produces lower scores on average compared to ROUGE (Table A4). Taken in combination, these insights indicate USER is better capable of discerning differences among the strong story generation models of the future, as it provides more stark evaluations while still correlating well with human judgments.

[4]We use `SequenceMatcher` from Python's `difflib`: https://docs.python.org/3/library/difflib.html