Text Generation and Evaluation for Human-Machine Collaborative Writing

Elizabeth Clark

A dissertation

submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

University of Washington
2021

Reading Committee:
Noah A. Smith, Chair
Yejin Choi
Katharina Reinecke

Program Authorized to Offer Degree:
Computer Science and Engineering

University of Washington

Abstract

Text Generation and Evaluation for Human-Machine Collaborative Writing

Elizabeth Clark

Chair of the Supervisory Committee:

Professor Noah A. Smith
Computer Science and Engineering

Natural language generation (NLG) models' ability to generate long, fluent texts has enabled progress across many NLG subfields and increased the types of contributions models can make to human-machine collaborative writing tasks. However, the improved quality of generated text also poses challenges for NLG model evaluation. In this dissertation, we develop and evaluate methods for using NLG models in a collaborative setting to offer suggestions to people as they write. We identify new modeling directions for this setting and build one such model and demonstrate its effectiveness. Finally, we improve automatic and human evaluations for long, fluent generated text, both by developing and testing new automatic metrics and by evaluating the effectiveness of human evaluations for state-of-the-art language generation models.

First, we explore the possibility of machine-in-the-loop creative writing. We performed two case studies using two system prototypes, one for short story writing and one for slogan writing. Participants found the process fun and helpful and could envision use cases for future systems. At the same time, machine suggestions do not necessarily lead to better written artifacts, and we suggest modeling and design choices that may better support collaborative writing. We explore one such direction (adding character representations as additional context for the model) and find it achieves improved generation results according to human and automatic metrics. We then consider the challenge of evaluating NLG models for collaborative writing and demonstrate how a collaborative writing platform can be used to collect pairwise, utterance-level human evaluations.

For evaluating long machine-generated texts, automatic methods avoid the collection of human judgments, which can be expensive and time-consuming. We introduce methods based on *sentence mover's similarity*; our automatic metrics evaluate text in a continuous space using word and sentence embeddings. We find that sentence-based metrics correlate with human judgments significantly better than ROUGE and can be used as a reward when learning a generation model via reinforcement learning.

Finally, we examine human evaluations of text generated by state-of-the-art models and find non-expert evaluators are unable to distinguish between human- and machine-generated text from three text domains. We explore various evaluator training methods, but find none is able to significantly improve evaluators' performance. We also find that evaluators focus on the form of the text more often than the text's content and often underestimate the capabilities of current NLG models. Based on these findings, we discuss future directions for collecting human evaluations for NLG models.

Acknowledgments

It is because of great mentors, colleagues, and friends that I have completed the work in this dissertation. First and foremost among these is my advisor, Noah Smith. I thank him for his guidance, support, and encouragement over the last six years. His advice, whether about research, professional development, or grammatical pet peeves, has been invaluable, and I will depend on it for years to come. Many thanks to my committee, Yejin Choi, Katharina Reinecke, and Gina-Anne Levow. Their perspectives and feedback strengthened this work, not only in the final dissertation but also over the entire course of my studies. I also thank Asli Çelikyilmaz for her mentorship that extended beyond my internship and continues to this day.

This dissertation is based on work that was written with the help of many others. I'd like to thank my coauthors: Noah Smith, Yangfeng Ji, Annie Ross, Chenhao Tan, Hao Fang, Hao Cheng, Maarten Sap, Ari Holtzman, Mari Ostendorf, Yejin Choi, Asli Çelikyilmaz, Karen Qin, Antoine Bosselut, Chandra Bhagavatula, Rowan Zellers, Ali Farhadi, Jianfeng Gao, Tal August, Katharina Reinecke, Sofia Serrano, Nikita Haduong, and Suchin Gururangan.

This work has also been shaped and supported by the broader UW NLP and CSE communities. Many thanks to the professors and TAs of my classes, to the students I've TAed and mentored, to the CSE staff who has made my life easier in countless ways, and to my friends and classmates. In particular, I have continually been impressed by how smart and how kind the UW NLP community is, and their academic and personal support has made me feel at home at UW and in the broader NLP community. I also owe many thanks to Noah's ARK for all their revisions of paper drafts, talk feedback, and pilot study participation. Finally, I need to thank my fellow 6th-years in my office and in ARK: Mandar Joshi, Julian Michael, Kelvin Luu, Maarten Sap, Lucy Lin, Phoebe Mulcaire, and Rahul Nadkarni. Our many conversations have been a highlight of my time at UW, and I look forward to continuing them over the rest of our careers.

DEDICATION

To my family

with all my love and gratitude.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Automatic tools have changed the way we write and collaborate, supporting writers with contributions like spelling and grammar error detection (e.g., spellcheckers like Grammarly) and enabling real-time collaboration and version control (e.g., Google Docs, Overleaf). Most writing tools, though, are focused on the writing process or a text's style or grammar; they do not involve the machine contributing directly to a text's content.

As natural language generation (NLG) models have rapidly improved, researchers are increasingly questioning the role that automatic tools can play in the writing process [Yang et al., 2019; Swanson and Gordon, 2012; Ghazvininejad et al., 2017]. Large neural models are more flexible and can handle more context than their rule-based or retrieval-based counterparts. Pretrained large models provide high levels of fluency, and finetuning and few-shot learning methods allow them to adapt to specific writing tasks and styles. Given these improvements, NLG models show potential for contributing more than surface-level suggestions to a writing task. Can they play the role of a collaborator instead, contributing content and ideas to writers?

Although the rapid development of NLG models is promising for human-machine collaboration, it has also outpaced our ability to develop effective evaluation methods for open-ended text generation. Traditional word-overlap metrics do not work well for many NLG tasks [Novikova et al., 2017] and are especially poorly suited to creative and collaborative text generation tasks. Human evaluations are typically used to evaluate generated text in these domains, but human evaluations face difficulties of their own. Current NLG models can produce long, fluent text passages, for which human evaluations are expensive and difficult to collect.

Human evaluations are also often disconnected from the end tasks and users who would be interacting with the NLG models [van der Lee et al., 2021].

In this dissertation, we explore the potential of neural models to collaborate with people as they write creative text. We examine different types of human-machine interactions and discuss directions for improving NLG models for human-machine collaboration. We also demonstrate how human-machine collaborative systems can be effectively used as evaluation platforms for NLG researchers to compare and analyze models.

We then discuss the challenge of evaluating state-of-the-art NLG models, looking at both automatic and human evaluations in NLG. We propose a new automatic evaluation metric for multi-sentence text and demonstrate how it can be used both as a text quality metric and as a reward when generating long texts. We finally turn to human evaluations in NLG and investigate how well non-expert evaluators can detect machine-generated text. Given current models' ability to generate fluent and stylistically-consistent text, we discuss the role and the future directions of human evaluations in NLG.

1.1 Background

Human-Machine Collaborative Writing

Creative applications of computing have been proposed for decades [Meehan, 1977; Ryan, 2017; Riedl et al., 2021], and past human-machine collaborative tasks include improvisational music [Hoffman and Weinberg, 2011; Quick and Thomas, 2019] and dance [Jacob and Magerko, 2015]. In natural language generation, past work has proposed collaborative writing tasks ranging from collaborative poetry writing [Ghazvininejad et al., 2017] to headline writing [Gatti et al., 2016] to story writing [Swanson and Gordon, 2012]. Because previous collaborative writing systems have suffered due to limited abilities to handle long contexts and generate fluent text, as NLG models improve, human-machine collaborative writing becomes increasingly viable. Collaborative writing systems for creative writing tasks, particularly for story writing, have risen in popularity and are currently an active area of research.

While early story generation models were based on rules and templates [Ryan, 2017; Meehan, 1977], neural models' ability to generate open-domain text and to adapt to new text styles through finetuning or few-shot training has resulted in their prevalence in recent work in both standalone and collaborative story

generation. Large neural language models can be used directly to generate stories [See et al., 2019], but past work has also incorporated story elements into the generation models, such as the story's structure [Fan et al., 2018], characters [Clark et al., 2018a], and events or plot points [Rashkin et al., 2020; Martin et al., 2018]. Collaborative story writing systems have also leveraged the flexibility of neural models, allowing writers to take turns with a neural model to write a story [Clark et al., 2018b] or request on-demand story suggestions [Roemmele and Gordon, 2018; Akoury et al., 2020].

As researchers develop new models for human-machine collaborative writing, evaluation remains a challenge. As in other areas of NLG, human evaluation is considered the gold standard evaluation in story generation and other creative generation domains, though the use of automatic measures to evaluate story quality has also been explored [Roemmele et al., 2017; Purdy et al., 2018; See et al., 2019; Guan and Huang, 2020]. Most evaluations of collaborative writing systems depend on writers' individual or system-level ratings using Likert scales, though some also include additional analyses, e.g., the writers' edits to the generated text [Roemmele and Gordon, 2015; Akoury et al., 2020].

Automatic Evaluation of Generated Text

Although n-gram overlap metrics like BLEU [Papineni et al., 2002] and ROUGE [Lin, 2004] have become commonplace in NLG evaluation beyond machine translation and summarization tasks, they only capture similarity when information is repeated verbatim between a source and reference text. One way to capture a more nuanced notion of similarity is to represent each text as a collection of word embeddings and consider the collective distance between the word embeddings in a reference text versus a generated text, as in word mover's distance [Kusner et al., 2015]. Due to the improvements in language representation enabled by large pretrained models like BERT [Devlin et al., 2019], this evaluation approach has become increasingly popular, resulting in evaluation metrics like the BERT-based BLEURT [Sellam et al., 2020], BERTScore [Zhang et al., 2020b], and MoverScore [Zhao et al., 2019].

In cases where there is no reference text to compare the generated text against (e.g., a story generated in a collaborative setting) or where a reference text may be just one of many suitable responses (e.g., a dialogue agent's answer to a chitchat question), automatic analysis is still possible, often focusing on different dimensions of the text's quality rather than its similarity to a gold reference text. For example, metrics like

self-BLEU [Zhu et al., 2018] and distinct-n [Li et al., 2016] measure the diversity of generated text, and sentence length and the distribution of the generated words' parts-of-speech have been used as proxies for the complexity and style of a text [Roemmele et al., 2017; See et al., 2019].

Human Evaluation of Generated Text

Human evaluations are considered the gold-standard evaluation method for many NLG tasks [van der Lee et al., 2021]. Though extrinsic human evaluations (i.e., downstream evaluations performed by end-users [Belz and Reiter, 2006]) are encouraged, they are rare in practice, with most evaluations consisting of ratings or rankings of a text's intrinsic qualities [van der Lee et al., 2021]. In most cases, evaluators assign generated text a rating along a Likert scale; common evaluation dimensions include "overall quality," "fluency," and "grammaticality" [van der Lee et al., 2021].

Despite the importance of human evaluations in NLG, there is little consensus in how human evaluations are run in NLG, even within a single task or domain. There is little consistency in the evaluation format, the types of evaluators, and the text quality dimensions that are evaluated (and even when evaluating for the same text dimensions, they can be defining these terms differently) [van der Lee et al., 2021; Howcroft et al., 2020]. These problems are exacerbated by a lack of reporting; descriptions of human evaluation procedures are often underspecified or missing altogther. For example, van der Lee et al. [2019] found only 55% of papers in their survey included the number of evaluators. The inconsistencies in human evaluations, obstacles to transparent and replicable research, have led to recent efforts to better document and clarify human evaluation procedures [Howcroft et al., 2020] and to build human evaluation platforms to standardize human evaluation results [Khashabi et al., 2021; Gehrmann et al., 2021].

1.2 Challenges

The high-level goal of many NLG tasks is to interact with and support people as they complete tasks, but in practice NLG problems are often simplified to static input-output tasks. While this allows rapid model development and evaluation, models trained to complete simplified tasks and evaluated with automatic metrics approximating desired behavior may be mismatched with the goals and expectations of an end user. One challenge is understanding what these user goals are and how current NLG models are or are not aligned

with them.

When modifying NLG models, we want to make sure the changes are improving our models in meaningful ways and in directions that benefit end users. To do this, we need good evaluation methods. Generated text can be evaluated by both automatic metrics and human evaluations, but human evaluations can be expensive and time-consuming, particularly when dealing with long texts.

Most popular automatic metrics require reference texts, "correct" answers against which they compare the generated text. However, in open-ended and creative tasks, there are rarely reference texts, and even when there are, they do not cover the space of all possible "correct" answers. In collaborative generation settings, this is especially true as the models are generating text for a brand-new and dynamic context; any reference texts would need to be collected post-hoc. For these reasons, human evaluations are most often used to evaluate NLG models for open-ended text generation tasks.

Human evaluations are typically treated as the "gold-standard" for evaluating generated text, but there is relatively little discussion of these methods within NLG and how to improve them. In fact, papers frequently omit details of how human evaluations altogether [van der Lee et al., 2019]. While our NLG models have greatly improved, our methods for evaluating them via crowdsourcing have not. The increasing fluency and length of text that current models are able to generate pose challenges to traditional approaches to collecting human evaluations of generated text.

1.3 Approach

To address these challenges, we first consider NLG models and evaluation for human-machine collaboration (Chapters 2-4), before discussing evaluation in NLG more generally (Chapters 5-6).

To see how NLG models can contribute to creative writing, we discuss a framework for human-machine collaboration and use it to design two human-machine collaborative writing systems. We collect participants' feedback about their experience writing with the help of generated text to better understand the strengths and weaknesses of current models in collaborative settings. Based on this feedback, we identify several directions for improving NLG models for collaborative writing, one of which we implement and find improves the generated text.

To address the challenge of NLG model evaluation, particularly in collaborative settings, we demonstrate

how the human-machine collaborative writing platform can also be used for pairwise model evaluation, providing utterance-level paired human evaluations of model quality. We then consider the broader challenges in evaluating state-of-the-art generation models and their ability to generate long, fluent texts. We propose an automatic evaluation metric designed for multi-sentence text passages by measuring similarity at the sentence level. To address the challenge that high-quality generated text poses for human evaluations, we analyze the effectiveness and the focus of non-expert human evaluators when identifying text generated by state-of-the-art NLG models. We explore different evaluator training methods to overcome this challenge, but find them unsuccessful, leading us to recommend future directions for human evaluations in NLG.

1.4 Outline

In Chapter 2, we first consider the role NLG models can play in the creative writing process. We present a "machine-in-the-loop" framework for human-machine collaboration on a creative writing project and run user studies with two different creative writing systems. Based on participants' feedback, we discuss the challenges that are specific to collaborative generation models that are not currently addressed by general-purpose language models and recommendations for improving these models. We explore one such improvement in Chapter 3, using character information to improve generated text. We show that incorporating representations of a story's entities as additional context can improve performance on several generation-related tasks. In Chapter 4 we see how human-machine collaborative writing systems can also be used as an evaluation platform for generation models.

We then discuss current evaluation methods in NLG and how they perform on state-of-the-art NLG models and tasks. In Chapter 5, we introduce an automatic evaluation method for handling long, generated texts. "Sentence Mover's Similarity" is a metric based on word- and sentence-embeddings, rather than word overlap metrics like BLEU and ROUGE, allowing it to capture a more nuanced sense of similarity. We find that the sentence-based metrics perform well at automatically evaluating both machine- and human-authored text, correlating with human judgments better than ROUGE. Furthermore, we show how the metrics can also be used directly in the generation process by incorporating them into a summarization model's loss function.

Finally, in Chapter 6, we discuss human evaluation practices in NLG. We find that non-expert human evaluators struggle to distinguish between human-authored text and text generated by state-of-the-art models

and that they often focus on the form of the text rather than its content. We explore three methods for training evaluators at this task, but their limited success points to the need for new human evaluation methods in NLG.

Chapter 2

Human-Machine Collaborative Writing

This chapter explores the possibility of incorporating a machine in the loop of creative writing. The motivation is twofold. First, writers often experience "cognitive inertia," a phenomenon known in the writing domain as "writer's block" [Garfield, 2008]. A collaborator who provides suggestions and points out new directions might help alleviate writer's block. The new combination of a writer's own ideas with suggested ideas is a form of psychological creativity [Boden, 2004]. Second, recent studies show that machines outperform humans in some tasks [He et al., 2015; Ott et al., 2011; Tan et al., 2014], including identifying which message will be retweeted more on Twitter. Perhaps a machine-learned algorithm can provide valuable suggestions to writers. We explore the space of designing machine-in-the-loop systems for creative writing and learn insights from user studies that can inform future interface design and research on natural language processing models.

The work in this chapter is published in Clark et al. [2018b].

2.1 Introduction

We propose a *machine-in-the-loop* framework, where the goal is to improve the ability of humans, with the machine playing a supporting role.[1] Machines can support writers as they edit, structure, and refine their work, as demonstrated by word processors, grammar and spell checkers, version control, or even language

[1] In contrast, human-in-the-loop machine learning actively includes humans in the process of training machine learning models by asking humans to provide feedback such as labeling difficult examples or suggesting new features [Branson et al., 2010; Cheng and Bernstein, 2015; Fails and Olsen Jr., 2003; Joachims et al., 2017].

or style analysis tools (such as the Hemingway Editor[2]). In this chapter, we focus on systems that assist people by suggesting content as they write and that are designed to inspire creativity throughout the writing process while still leaving writers in control of the final written artifact. In particular, we investigate the following questions:

- How can we design machine-in-the-loop systems to support diverse writing tasks and processes?
- What effect do these systems have on people's writing, both as perceived by the writer and by other people?
- What do people want to see in machine-generated suggestions and creative writing support systems?

To answer these questions, we developed two prototype systems to help writers enhance their creativity in two tasks: story writing and slogan writing. These two creative tasks have different goals and require different writing styles. Thus, the systems that assist in these tasks should provide different types of help. We built two system prototypes that use two different models to generate suggestions. We had study participants write with these prototypes and compared their experiences and the quality of their writing to that of participants who did not receive suggestions. This paper discusses the current capabilities of machine-in-the-loop writing systems and suggests improvements both for system interfaces and models.

Although providing helpful suggestions is important in a machine-in-the-loop writing system, we leave writers with complete editorial control and the freedom to disregard any unwanted suggestions. Our goal is *not* to replace human creativity or automate creative writing; rather, we seek to amplify people's creativity by providing suggestions that are most useful to them. By offering suggestions as a person writes, the writing process has elements of both collaborative writing and constrained writing tasks. It also provides a versatile setup; a machine-in-the-loop writing system could be used as an educational tool, a writer's tool, or for entertainment.

2.2 Machine-in-the-Loop System Characteristics

This chapter considers machine-in-the-loop (MIL) systems that are composed of a person and a machine working together to create output (Figure 2.1). The person and machine are in a loop in which the person

[2] `http://www.hemingwayapp.com`

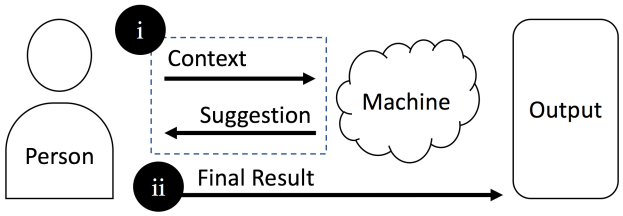


Figure 2.1: Machine-in-the-loop system structure: the loop (i) is initiated with the person providing context and the machine responding with a suggestion. The person always has control over the final result (ii).

	Story writing	**Slogan writing**
structure	additive	iterative
initiation	push	pull
intrusiveness	high	low

Table 2.1: Characteristics of MIL systems for our story and slogan writing tasks.

provides context and the machine responds with suggestions (Figure 2.1 i). The person controls the final output (Figure 2.1 ii).[3] Different creative writing tasks have different demands for MIL systems. We consider the following three characteristics for designing MIL systems: interaction structure, interaction initiation, and interaction intrusiveness. We use our story and slogan writing tasks as examples to explain each aspect, as summarized in Table 2.1.

2.2.1 Interaction Structure

Interaction structure can be *iterative*, where a writer works with the help of a machine to refine a single idea, or *additive*, where the writer and machine work to add multiple ideas together. This can be represented as how many times the loop of the person and machine exchanging context and suggestions (see section "i" in Figure 2.1) is repeated before the person commits to a final result. For example, story writing is additive because as a story unfolds, writers (and machines) need to introduce new ideas, and these ideas are combined into a final story. Slogan writing, by contrast, is a highly iterative process: the loop is repeated for a single phrase or sentence until a final slogan is decided upon.

[3]The setup this paper explores does not represent all possible configurations for writing with a machine. More complex setups might involve the person and the machine working in parallel or the machine directly altering the output in a mixed-initiative fashion.

2.2.2 Interaction Initiation

Interaction initiation refers to how the context-suggestion loop (see section "i" in Figure 2.1) is triggered. It can follow a *push* (automatically initiated) or *pull* (person-initiated) method of initiation, or a combination of the two. To have breadth of exploration, we implemented the story writing system as a push-style system. At every other sentence, the machine presents a suggestion to the writer. The slogan system uses the pull method for retrieving suggestions. Writers provide a slogan-in-progress and keywords and prompt from the system whenever they want new suggestions.

2.2.3 Interaction Intrusiveness

Interaction intrusiveness describes the extent to which computer-generated suggestions are ignorable. Although writers can always edit or reject suggestions, some require more attention than others. We designed the story writing system's suggestions to be highly intrusive. They appear directly in the text box where the person is writing, and the writer must interact with the suggestion (even if only to delete it entirely) before moving on in the writing process (see Figure 2.2). In the slogan writing system, suggestions have low intrusiveness. Suggestions appear in a separate column from the writing space and require no interaction once they are retrieved (see Figure 2.3).

2.3 Related work

We propose "machine-in-the-loop" in contrast with "human-in-the-loop" machine learning. This concept resonates with "mixed-initiative user interfaces" [Horvitz, 1999]. Although Horvitz emphasizes the development of user interfaces in settings where both the human and the computer can drive towards a shared goal (as opposed to our human-driven setup), many of the principles he considers are relevant to this work, including the timing of machine contributions, providing editing capabilities, and understanding the social expectations of collaborators [Horvitz, 1999]. As in mixed-initiative interface work, the goal of our work is to explore interaction paradigms and to combine human and computational strengths to enhance human ability [Allen et al., 1999]. The mixed-initiative setup has been used for creative tasks such as game design [Yannakakis et al., 2014], and adapting our systems to a more complex mixed-initiative setup (e.g.,

dynamically deciding when to offer suggestions and what format of suggestion would be most helpful) is a promising future direction.

Several tools have been developed to provide suggestions to assist people in writing, both within the research community [Swanson and Gordon, 2008; Roemmele and Gordon, 2015] and as personal projects [Sloan, 2016]. Swanson and Gordon's "Say Anything" [Swanson and Gordon, 2008] provides suggestions for writing short stories by prompting writers with full sentences retrieved from a database of stories scraped from the web after every turn of writing. In "Creative Help" [Roemmele and Gordon, 2015], writers are offered suggestions as they write stories, but only when they explicitly request them (i.e., a pull method of interaction). While these systems retrieve their sentences from existing stories, we use natural language generation to provide suggestions.

Author Robin Sloan created a sentence completion story-writing assistant tool that suggests the end to a partially-written sentence when prompted by the writer [Sloan, 2016]. The focus of Sloan's project is on how to provide suggestions to the writer. Our work focuses on the role these suggestions play in the writing process, the interface, and people's interaction with the system. Past collaborative creativity research has also looked at other related writing tasks include headlines for newspaper articles [Gatti et al., 2016] and lines for poetry [Ghazvininejad et al., 2017], and other artistic domains like music [Hoffman and Weinberg, 2011] and dance [Jacob and Magerko, 2015].

Finally, there is work on collaborative writing systems that bring together a group of people to write collaboratively. For example, "Ensemble" was a system that had multiple participants work together to write a single story [Kim et al., 2014]. Each story had a lead author and contributors who submitted alternate versions of a scene and voted on alternatives they liked. The lead author ultimately had the authority to choose the winning scene. The person in our work plays a similar role to the lead author in Ensemble; they control the direction of the story and decide how to incorporate the external (system) input. Similarly, Soylent [Bernstein et al., 2010] uses crowdsourcing to provide assistance to writers. However, Soylent assists in shortening text and editing grammar rather than providing content and new ideas.

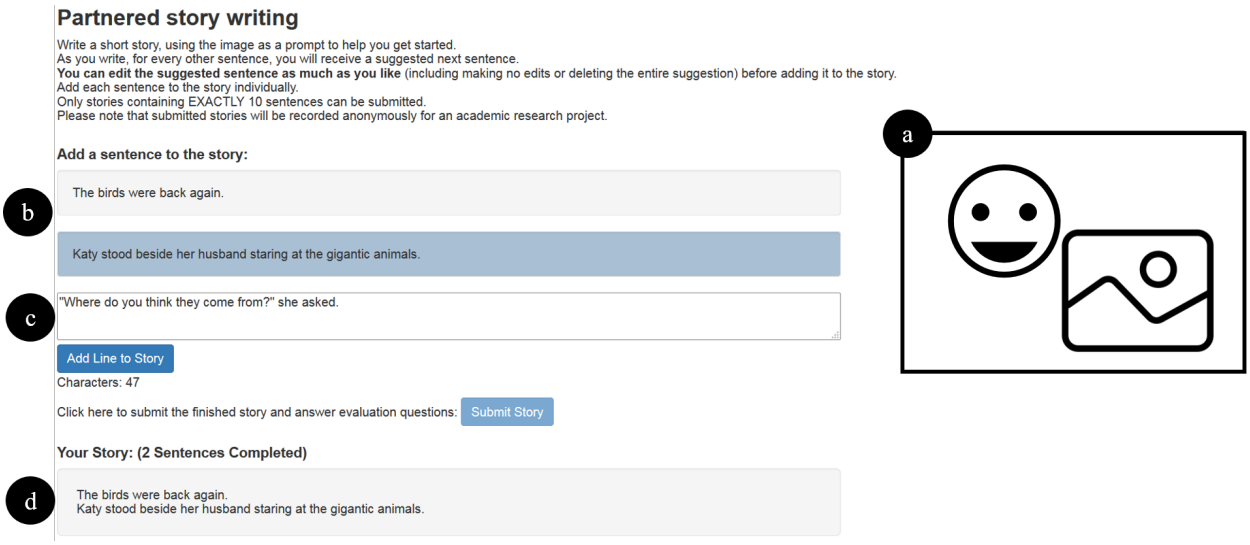


Figure 2.2: The MIL story writing interface: (a) an image prompt from the New Yorker (actual image: Donnelly [2014]) (b) the story so far, dark colored sentence boxes were turns written with machine suggestions, (c) entry box for next sentence, the machine suggestions appeared here, (d) the full story so far. The solo condition interface was the same except no machine suggestions appeared.

2.4 Story Writing System

Our first system explores writing to expand a visual prompt into a story. The setup is inspired by *Exquisite Corpse*, a game played by Surrealist artists in which people take turns contributing to a drawing [Brotchie and Gooding, 1995]. The portions of the drawing that were completed in previous turns are partially or completely hidden from the current artist, resulting in silly and bizarre drawings. A parallel version of the game exists in literature, where each player writes a sentence of a story, folds the paper over to hide all but the most recent round of writing, and then passes it to the next player. With only two players (as in our setting), hiding earlier rounds has no effect (every sentence was written or seen by the person).

We use the turn-taking aspect of the Exquisite Corpse game to help foster creativity while writing. We provide people with machine suggestions to encourage stories that are unexpected, unusual, and novel, all of which are characteristics of creativity [Sternberg, 2005]. These machine suggestions may surprise writers, sway them to change their own ideas about the direction of the story, and include ideas they may not have thought of.

Details on the slogan writing system can be found in 2.5.

2.4.1 User Study Task

For our story system user study, the participant is prompted to write a ten sentence story based on an unlabeled, single-panel cartoon from *The New Yorker* caption contest[4] (in the space indicated in Figure 2.2, section a). The task was either done alone (solo condition) or partnered with machine suggestions (MIL condition). Both versions were done through a web interface, presented in Figure 2.2. Participants entered the story sentence by sentence (Figure 2.2, section c). Once submitted, a sentence could not be edited. The complete story, along with the number of sentences completed, appeared at the bottom of interface (Figure 2.2, section d).

In the solo case, no additional prompting or interactions were provided beyond the image. In the MIL condition, the participant began by writing the first sentence. Once the sentence was submitted, the next sentence would be generated by the machine and "pushed" to the participant, appearing directly in the submission box (Figure 2.2, section c). Full sentences were used based on a preliminary study showing that people liked full-sentence suggestions as much or more than partial sentences or keywords when writing stories. The person was free to edit the machine-suggested sentence to any extent, including deleting it entirely, before submitting it. The third sentence was again written by the participant alone. This turn-taking continued until the story was 10 sentences long. Our demo system is available at `http://bit.ly/iui-story-demo`.

2.4.2 Computational Model for Suggestions

Our computational model for suggesting a sentence given preceding text is built on a neural language model. Neural language models have been used for various natural language generation tasks, including image captioning [Vinyals et al., 2015], conversational modeling [Sordoni et al., 2015], and poetry generation [Ghazvininejad et al., 2016]. To make the generated sentences fluent and coherent in context, our language model uses contextual information both within and across sentence boundaries.

Neural language models are effective at predicting words that fit well in a context locally, thereby generating coherent sentences. A sentence-level language model [Mikolov et al., 2010] defines the probability distribution over the next word, from within a predefined vocabulary, conditioned on the left context (i.e., the

[4] `https://contest.newyorker.com`

context of the generated sentence so far). The language model for our story writing system was trained on 390 adventure novels (about 400 million words) from the Toronto Book Corpus [Zhu et al., 2015]. Although the model generates one word at each step, people writing with the system only see complete generated sentences.

Standard language models only use the left context within a sentence to compute the probability distribution for the next word; they ignore earlier sentences. To overcome this limitation, we adapted a neural language model that takes account of left context from both the current sentence and the previous sentence [Ji et al., 2016].

2.5 Slogan Writing System

Slogans present a challenge to writers distinct from that of writing a story: to generate a concise, memorable, and powerful statement that is representative of the object, organization, or idea it promotes and matches the intention of the authors. Slogans are used in a variety of settings, ranging from organizing a social movement to promoting a product. The process of condensing information into a memorable and informative phrase used to create slogans is paralleled in other tasks, such as writing headlines, titling papers, and naming products. Therefore, a system that supports slogan writing could likely be extended to related tasks that prioritize catchy and succinct language.

2.5.1 User Study Task

For the slogan writing task, participants were asked to write a slogan for three distinct scenarios: a food packaging tool, the movie *Her*[5], and a social cause that protests animal testing for cosmetic products. The prompts included descriptions and images, from which the participant had to invent an original slogan. Like in the story writing case, the task was either done alone or partnered with a machine in the loop. In the MIL condition, participants used a web interface (see Figure 2.3). Participants in the solo case worked in a blank Google Doc.

When writing with the web interface, the writer must provide a few keywords and write an initial version of the slogan (Figure 2.3, section a). The writer can then "pull" machine suggestions at will (Figure 2.3,

[5] `http://www.imdb.com/title/tt1798709/`

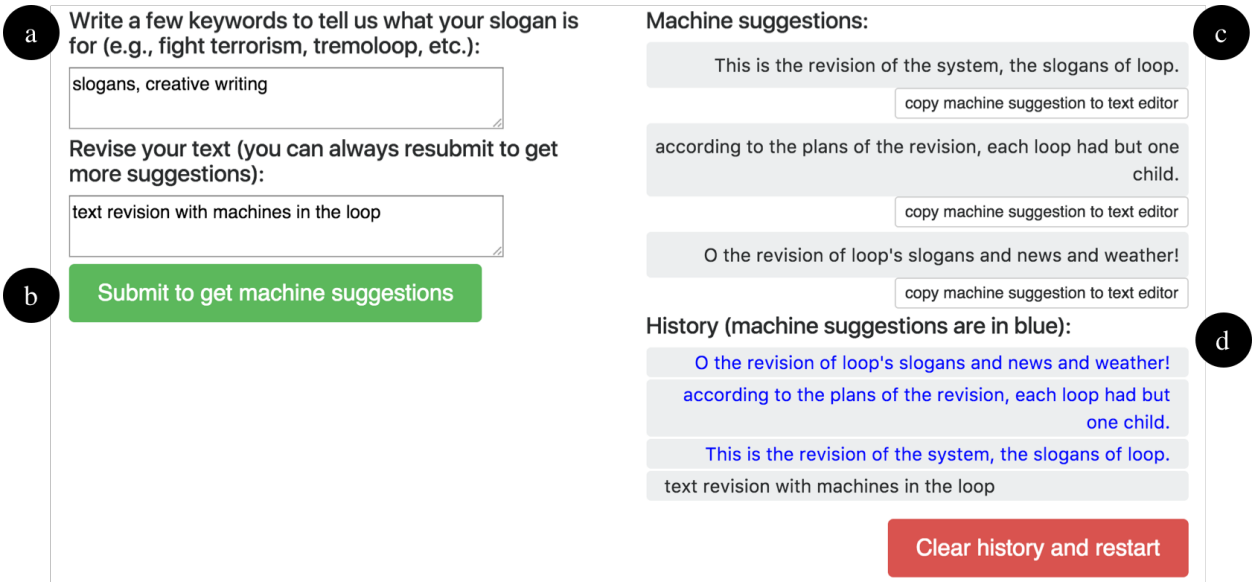


Figure 2.3: The MIL slogan writing interface: (a) the person can provide keywords and the slogan, (b) a button to "pull" suggestions, (c) the current round of suggestions, (d) the history of all slogan submissions and corresponding suggestions.

DT	JJ	NNS	VBP	RBR	JJ	IN	DT	NN	.
Some	Human-AI	interactions	are	more	profound	than	a	movie	.
Some	*	*	are	more	*	than	a	*	.

Figure 2.4: A slogan and its resulting skeleton. The top row shows the corresponding part-of-speech tags. This slogan was written about the movie *Her* with a machine in the loop.

section b). Based on the writer's input, the system suggests alternative slogans (Figure 2.3, section c), and the history of the retrieved suggestions is tracked for future reference (Figure 2.3, section d). The system provides at most three suggestions in response to each request. The writer's input is on the left, and machine suggestions are on the right, reducing the intrusiveness of the suggestions. A demo system is available at `http://tremoloop.com`.

2.5.2 Computational Model for Suggestions

We developed a constrained language model that was inspired by the BRAINSUP system of Ozbal et al. [Özbal et al., 2013]. First, we extract existing syntactic patterns to improve the grammaticality of the generated suggestions. Specifically, we start from quotations from Wikiquotes[6] and replace all content

[6] `https://en.wikiquote.org/wiki/Main_Page`. Ideally, we would use a slogan dataset. Given that there is no public slogan database, we believe that quotations and slogans share some similar characteristics, such as memorability and pithiness.

words with a wildcard symbol. These patterns become *skeletons* for our generation system. Figure 2.4 shows an example; the skeleton is shown at the bottom. Because the skeletons all come from real quotations, their structures are grammatically plausible, as long as the slots are filled with appropriate words. Which words are appropriate (individually or together) is a matter of linguistic judgment.

Information from the writer's input (see Figure 2.3, section a) and the extracted skeletons are combined to generate slogan suggestions. To make sure that generated suggestions contain keywords from the writer's input, we randomly sample content words from the input and treat them as target words. We identify skeletons that have empty slots that match the part-of-speech tags of the target words and choose three candidate skeletons. Given a skeleton, we follow Ozbal et al.'s approach [Özbal et al., 2013] and use beam search to fill in slots with words that approximately maximize a scoring function. The scoring function gives high scores if the target words are used to fill in slots. In addition, the scoring function factors in language model probability scores to encourage grammaticality and a word diversity score to avoid repetition.

2.6 User Study Setup

To explore how people will interact with machine-in-the-loop writing, we had people write with or without a machine in the loop. We obtained third-party reviews of the final written pieces. This gave us insight into what types of interactions and suggestions people are interested in and find most useful when writing with a machine in the loop.

2.6.1 Task Setup

For both the story writing task and slogan writing task, we assigned half of the participants to write with the machine-in-the-loop system prototype (the MIL condition) and half without it (the solo condition). Half of the participants in each condition (solo and MIL) were asked to write three stories and the other half wrote three slogans, based on three different prompts. The order of the prompts was balanced across participants. In both the solo and MIL tasks, after each story or slogan was complete, the participant completed a survey consisting of seven-point Likert scale questions about the final piece of writing. Table 2.2 lists the exact questions for each task under "Writing Survey." Surveys for both tasks additionally asked how satisfied participants were with the final piece of writing.

	Story Task	Slogan Task
Writing Survey *"Is the final product:"*	Creative? Coherent? Entertaining? Grammatical?	Creative? Catchy? Relevant?
Final Survey *"Were the suggested sentences:"*	Surprising? Creative? Grammatical?	Surprising? Creative? Catchy? Relevant?

Table 2.2: Survey questions

After three rounds of writing and evaluation, participants completed a final survey. All participants were asked a seven-point Likert scale question on how enjoyable they found the writing process. They were then asked open-response questions on the interface design, the difficulty of the task, and what improvements could be made to the tool. The solo case participants were also asked to describe any tools that could have helped their creative process. The MIL participants were additionally asked a four-choice question on how likely they were to use the system again and Likert scale questions about the quality of the suggestions the prototype system provided (see the exact questions under "Final Survey" in Table 2.2), whether they liked the suggestions they received, and whether they appreciated the suggestions.

After participants completed the three writing tasks and four surveys (one after each writing task and one final survey), we conducted an open-ended interview about their experience with the task, their creative writing background, and their thoughts about future improvements and uses for the tool.

Although participant enjoyment and personal perception of their own success are important measures of the prototypes, we also wanted to know how third-party evaluators perceived the writing done alone versus with a machine in the loop. Amazon Mechanical Turkers evaluated the writing that participants produced along the same dimensions as the participants who wrote them ("Writing Survey" in Table 2.2).

2.6.2 Analysis Methods

The first two authors created an interview coding scheme [Lazar, 2017] based on the Likert scale prompts and other areas of interest. The first two authors independently coded two interviews, compared coding, resolved conflicts, and revised the coding scheme. The final codes covered insights on: interface, machine suggestions, writing process, writing background, collaboration, use cases, and writing prompt. They then

Condition	None	Some	A Lot
Solo Story	4	2	3
MIL Story	5	4	0
Solo Slogan	3	1	5
MIL Slogan	4	4	1

Table 2.3: Number of participants in each condition with a given level of writing experience.

independently coded four more interviews, one from each of the study conditions. These codings were compared, and disparities were discussed. The first two authors then each independently coded separate halves of the remaining interviews, evenly distributed between conditions. The coders then came together to compile results.

2.6.3 Participant Demographics

We had 36 participants complete the writing tasks, 9 in each of the 4 task categories: solo story writing, MIL story writing, solo slogan writing, and MIL slogan writing. Participants were compensated with a $20 Amazon gift card. We categorized participant writing experience into none, some, or a lot. Participants with a lot of experience included professional creative writers and passionate hobbyists who wrote frequently. Participants with some experience included those that wrote occasionally or had formal creative writing education in their past. Participants with no experience included those who had not done creative writing since primary school. Table 2.3 shows the breakdown of experience by task condition. Participants were randomly assigned to conditions regardless of experience; there were more expert writers in the solo conditions than the MIL conditions, which should be kept in mind when comparing solo and MIL results.

Amazon Mechanical Turkers evaluated the final writing samples, with 9 evaluations collected for each of the 108 writing samples (3 per participant). Turkers were paid $0.15 for each evaluation they completed. To qualify, Turkers had to have completed over 1,000 tasks, have over a 95% task acceptance rate, and be from the United States.

2.6.4 Story Writing Results

Due to differences in the system designs and goals, as described in the above sections, we describe results for each system separately. The participant IDs indicate task and condition: MST (MIL story writing), SST

Jim slouched in the corner, feeling sorry for the patient in front of him.
["This is ridiculous," said Duke.]
"Yesterday I felt fine, and now you're telling me I'm at death's door?!"
["We'll take care of Furble tomorrow,"] the doctor said.
"You've named my tumor?!" Duke shrieked.
["Yeah,]" replied the doctor coolly, "we've found that anthropomorphizing tumors helps people in your position come to terms with their condition more easily."
Jim watched as Duke's eyes got even wider, and he wondered if it was because of the doctor's casual tone or the fact that the tumor had such a ridiculous name as "Furble".
~~**[Lance yells over the speakers "no sudden hammering"]**~~
"Anyway, we feel that Furble will most likely be gone within a month," the doctor said.
Jim grew more concerned about Duke's eyes, they seemed impossibly large now and if he wasn't careful Furble might not be the man's only medical concern.
["You're joking right], **["]** Duke said.

Figure 2.5: Sample story written with the story writing system by participant MST65. The computer suggestions are in color and brackets; struck out text indicates rejected submissions. (Image prompt: Cullum [2005]).

(solo story writing), MSL (MIL slogan writing), and SSL (solo slogan writing).

In this section, we discuss the results from the story writing system. Results from the slogan writing system can be found in 2.6.5.

We present insights from the story writing system from participants and third-party evaluators. Some participant insights did not depend on condition, such as enjoyment, interface suggestions, and opinions about story writing. Other comments from the MIL condition were more focused on the strengths and weaknesses of the machine suggestions.

Enjoyment

Overall, people found the story writing task fun. Two participants liked that it was a low-stakes, non-judgmental experience. Participants in both the MIL and the solo condition rated the task as highly enjoyable (both averaged 6.0).[7]

Participants who wrote in the solo condition had higher average satisfaction with their final stories than

[7]Due to the subjectiveness of evaluating writing and the varied participant background, both the self-evaluations and the Turk scores highly varied, often covering the full Likert scale (1-7).

those in the MIL condition (average rating of 5.03 versus 4.59, p=0.27)[8], and solo condition participants thought their stories were more creative (5.30 versus 4.30, p=0.01). However, as seen in Figure 2.6, the MIL condition did have one more positive vote ($\geq$5 on the Likert scale) for satisfaction than the solo condition.

When Help is Useful

Of the participants who found the suggestions helpful in directing their stories, three said the suggestions were most useful early in their story; they were more able to incorporate new elements from the suggestions before they had an established vision of the story. MST79 said, "By the time I'm hitting sentence 5, 6, 7, 8, I've developed so much of the story in my head already, most of the time suggestions are so far and away from anything I want to consider." This is seen in Figure 2.5; earlier suggestions were incorporated, while a later suggestion was deleted entirely. Five participants liked having the suggestions throughout the writing process to help if they got writer's block. One participant thought it would have been helpful toward the end of his writing to be prompted to start wrapping up the story.

Suggestion Usefulness

When asked what they considered when evaluating the creativity of their final stories, most participants emphasized unexpectedness or deviation from the obvious as key aspects. In the MIL condition, there were mixed reactions to the usefulness of the suggestions in enhancing creativity. All participants said that the suggestions were very random. For eight participants, this meant they disregarded most suggestions, but two participants said that the randomness of the suggestions inspired them to write sillier stories or that incorporating those suggestions lead to more odd and creative writing; "when I took the AI into account and tried to incorporate that, it got a lot more creative because, again, it was just so spontaneous and much more random than I normally write" (MST65). Eight participants said the suggestions did or could have influenced the direction of the story. However, six participants said they did not find the suggestions helpful once they had a clear direction for the story. This is reflected in the ratings participants gave the suggestions; the mean score of how much participants appreciated the suggestions is 3.78, but there was high variation

[8] p-values are calculated using independent two-sample t-tests. Although we report p-values, we encourage caution in drawing firm conclusions from these calculations because of the small sample size (27 for each condition in both self-evaluations and third-party evaluations) and the imbalance across conditions in author expertise.

between participants, with answers ranging between 2 and 6.

Participants found some elements of the suggestions more helpful than others. One participant liked using snippets of suggested dialogue, while a different participant found dialogue-heavy suggestions unhelpful. One participant mentioned taking the tone of the suggestions, and another participant appreciated a plot suggestion, although they didn't incorporate it as it would have required more context. Characters were a divisive element; some appreciated getting suggestions with new character names, while six others found suggestions that introduced new character names hurt the relevance of the suggestions. Consider the example in Figure 2.5. Although the character names in early suggestions are all used, a later suggestion that references "Lance" is deleted as all of the characters in the story have already been introduced. Timing may also affect the usefulness of new characters because characters are often introduced in the beginning of a story, as can be seen in the example stories in Table 2.5.

When asked what type of help they would like to receive as they wrote, eight MIL participants mentioned the machine could contribute by suggesting plot points, tone, or keywords. One participant from the solo condition envisioned a system where the computer took the role of a character in the story and provided dialogue. Other MIL participants appreciated the idea of receiving full sentences, especially if the suggestions had been more relevant. Four participants felt more back-and-forth iterations were needed to treat the machine as a viable collaborator and would appreciate feedback on their writing, such as encouragement, agreement, or advice.

Interaction

Three MIL condition participants found the every-other-sentence injection into their workspace disruptive. Participant MST65 wrote, "The 'partner' [MIL system] often made no sense, so it was difficult to incorporate their responses and often I just deleted the entire suggestion but it was a disruption." Four participants would have liked the ability to edit already submitted sentences, especially professional writers who were not able to follow their normal writing process. However, six participants enjoyed the constrained, non-editable, sentence-by-sentence structure as it kept them moving forward. Participant SST17 wrote, "Even when I got stuck, I eventually could tell myself, 'Just write one sentence!' and then move on. ... it forced me to keep moving forward instead of getting bogged down in getting all the details just right and trying to overhaul the

Use Condition	Story	Slogan
I'd use it, exactly as it is now.	2	0
I'd use it, but only if the suggestions were better.	4	5
I'd use it, but only if the writing set-up changed.	0	1
I wouldn't use it.	3	3

Table 2.4: Responses to "Would you use this system again?"

whole thing when I didn't like one little thing."

Use Cases

Most people who wrote with the story writing system would use it again in some capacity, with the most frequent response being they would use it again if suggestions were better, as seen in Table 2.4. Of the people who said they wouldn't use the system again, one wrote that they might actually use it for fun or ungraded work, one said they didn't need help creating story ideas, and the third expressed skepticism at the general idea of writing technologies.

Of the participants who would use the story writing tool again, two of the participants envisioned "just for fun" applications, such as a game to play with children or a short, fun activity that pops up on Facebook. Four participants saw it as a way to practice writing or a low-stake activity to become motivated to write. Three participants envisioned using the system for outlining, story boarding, or other early idea generation, while two people indicated interest in using the tool to directly write a final product.

As for types of suggestions that may be useful, three participants wanted editorial feedback on grammar, syntax, and sentence structure. One participant envisioned feedback from the machine that more directly influenced the content of the story. When describing experiences with human collaborators, participants said that back-and-forth iteration was a key component that they wanted machines to mimic. Other characteristics of good past human collaborators included trust and like-mindedness.

Third-party Evaluations

There was no statistically significant difference in the Amazon Mechanical Turk third-party evaluation ratings between the solo and MIL conditions. For creativity, the average score for the solo condition was 4.87 and was 4.84 for the MIL condition (p=0.88). Story coherence scores had an average of 5.05 for solo and

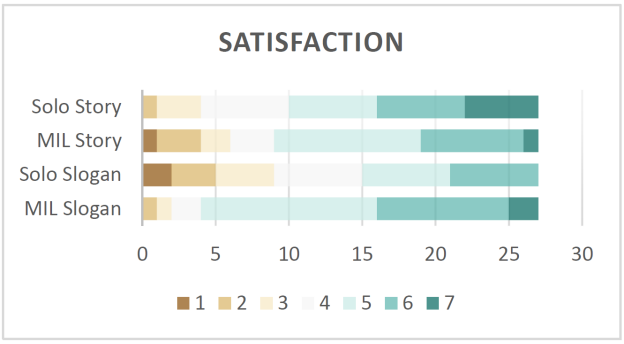


Figure 2.6: Slogan writers in the MIL condition were generally more satisfied with their final product than the solo slogan writers. More MIL story writers were positively satisfied with their final story, but only by one story.

4.77 for MIL (p=0.21).

It is important to note that most of the third-party scores did not correlate with the scores writers assigned their own work. The average Turker score and the self-evaluation score had Pearson correlation coefficients of 0.08, 0.07, and 0.04 for how creative the story was, how entertaining it was, and how much they liked it, respectively. The correlations for coherence and grammaticality were slightly better (0.31 and 0.41, respectively) but still weak.

2.6.5 Slogan Writing Results

As with story writing, some topics from the slogan tasks were mentioned by participants regardless of condition. However, because the solo case participants did not have suggestions nor the interface, MIL participants provided some unique insights.

Enjoyment

Five participants found writing slogans to be very difficult. One participant said they found it demoralizing because they felt so bad at it. Two participants expressed that writing something both succinct and informative was challenging; the top-rated slogans tended to be shorter than the lowest-rated slogans, as seen in the examples in Table 2.5. Four participants enjoyed the task and found it fun.

Participant self-evaluation of the final slogans was generally higher in the MIL condition than the solo condition, even though the scores for the slogan suggestions were low. For example, the mean score for final

slogans from the MIL condition was higher than in the solo case for both satisfaction (5.22 vs. 4.07, p<0.01) and creativity (4.48 vs. 3.93, p=0.15).

Suggestion Usefulness

Slogan participants in the MIL condition felt the suggestions did not introduce enough novelty. Three participants expressed frustration with suggestions that had just reorganized their input slogan words. These suggestions were too close to the original slogan and therefore not creative enough to be helpful. This frustration is reflected in low average scores for how much participants liked the suggestions (2.00) and appreciated the suggestions (2.78). Similar to the story writing task, appreciation varied greatly between participants, with scores ranging from 1 to 6. When participants used the provided suggestions, it was mainly to incorporate novel, relevant words or structural elements.

Seven MIL condition participants noted they would have preferred words instead of full sentences as suggestions; the system could act as a thesaurus to bring relevant but creative words and ideas. Participants also envisioned machine contributions such as searching the web for more information on a product, or finding similar, popular slogans as seed ideas. One participant found the machine had a different impact, "it impacted my overall thought process and creativity rather than actual words" (MSL87). Participants mentioned word play, use of literary devices (like alliteration or puns), cleverness, and catchiness as elements of creative slogans that they were looking for in suggestions.

Four participants described productive collaborations as bouncing ideas off someone else, building and iterating on good ideas, and coming to a feeling of "we found it!" Five participants thought the machine was a good collaborator, "it seemed like kind of having another pair of eyes in the room to give me some feedback and the fact that it was in real time was great and wouldn't argue with you over coffee was great" (MSL87). One participant felt the interaction with the system was not as conducive to enhancing creativity. One challenge five participants described was that the suggestions were not aware enough of context and therefore were not working on the same idea, just with the same words. MSL33C said, "they weren't understanding my keywords correctly, I think, so I could say something like 'end animal harm' and it would suggest harming animals ... so I don't think it was quite interpreting my intentions very well." Three participants also were interested in non-content-based interactions such as feedback on whether an idea was

good, reminders if a slogan was too close to an older discarded slogan, or an expression of closure upon deciding on a final slogan.

Interaction

Two participants liked having the history of suggestions in order to refer back to previous slogans. Another found the history distracting and wanted to curate the list to only keep good recommendations. For the physical placement of suggestions, two participants liked that the suggestions were out of the way and not intrusive. For four others, the suggestions were too far out of the way and required effort to check. These participants would have preferred a more condensed interface. One participant would have liked the ability to have more brainstorming space, either as a free form writing space or to iterate on multiple slogans at the same time.

Use Cases

Like the story writing case, most people who wrote with the slogan writing system would use it again in some capacity, especially if the suggestions were better (see Table 2.4). The people who said they would not use the system again wrote that they received suggestions that were too far from what they wanted to motivate them to use the system again.

The participants who said they would use the system again drew parallels between writing slogans and tasks such as naming courses or products, writing headlines, and writing titles. One participant said he might use a system like this to write emails in order to be reminded to be pithy and catchy. One participant did not think they would use the tool for any regular activities and would just use it for slogans.

Third-party Evaluations

The Amazon Mechanical Turk third-party evaluations rated the slogans written in the MIL condition as slightly less creative than the slogans from the solo condition, giving an average score of 3.84 and 4.37 (p=0.03), respectively. There was no statistically significant difference in any of the other scores, with relevance scoring the closest between the solo (average 5.98) and MIL (average 5.93) conditions (p=0.76).

Like in the story case, most of the third-party scores did not correlate with the scores writers assigned

their own work. The average Turker score and the self-evaluation score had a Pearson correlation coefficient of 0.13 and 0.10, for how creative the slogan was and how much they liked it, respectively. The correlation for catchiness and relevance were slightly better (0.39 and 0.22, respectively) but still weak.

	Highest-rated	**Lowest-rated**
Solo Slogan	Compassion is Always in Style	"Be Beautiful: End Animal Testing! Sign our petition today to show Neutrogena animal testing is unnecessary, unkind, and needs to stop now."
MIL Slogan	The real animals are the ones who test chemicals on living things.	"Stop animal research testing now! Studies confirm ther [sic] is no need for such testing, and msot [sic] adults are opposed to this practice."
Solo Story	Norman walked into the doctor's office just before his last appointment; Norman flicked his cigarette into the cold, winter night behind him as he walked into the building. The pretty blonde receptionist greeted Norman but Norman had no time for flirtation or romance; he was about to find the murderer of Trystan Lee and the doctor was key to this plan.	A middle aged male is meeting with a dentist. This gentleman is so stressed that he looks like his eyes are popping out of his face. This meeting is part of an investigation about a crime so that's why this meeting includes an investigator who looks creepy.
MIL Story	The nervous doctor cleared his throat, "Thank you...eh-hem...Mr. Collin, for coming into the office on such short notice." Craig slicked back his hair, listening to his wife's voice echoing in his head, reminding him that all of those late night trips to McDonalds would catch up with his heart eventually. "You see, we've found some...unusual...results from your recent stress test, and I thought it prudent to bring you in as soon as possible."	"Hey Docta Don, dis is da kat we wuz talkin' about last night, whachu wan' me to do wit 'im?" Fabin said through his cold eyes shaded by his pitch black sunglasses. He tapped his finger on the trigger and shook his head, Docta Don was never happy with how excited Fabin was to get into trouble; he was a good man, but followed all orders without ever thinking things through for himself.

Table 2.5: Highest and lowest rated slogans and stories for a given prompt. Slogans are for an animal rights cause. Only the first few lines of each story are shown.

2.7 Discussion

We found that people generally enjoyed writing with the help of suggestions and were enthusiastic about the concept of writing with a "collaborator," especially once natural language generation capabilities improve. Though some professional authors hesitated at the idea of using computer-generated suggestions when writing a final product, participants envisioned the usefulness of this system as a writing warm-up or game and

for difficult processes that are often collaborative (naming products or papers, writing headlines, etc.).

Another advantage of writing with a machine in the loop that participants observed was that writing with these systems allowed them to write in a judgment-free setting. Although collaborative writing is useful, it can be intimidating for less experienced writers to brainstorm or write with the pressure of a human collaborator. Writing with a machine in the loop can be a low-cost, easy way to provide new ideas and support to writers, particularly in the early stages of writing.

For machine-in-the-loop writing systems, we recommend a high level of writer control over the interaction. This will allow systems to cater to a wider range of writers and to adapt to changing writer needs at different points of the writing process. We also recommend carefully considering the interaction design choices (especially along the characteristics we describe) and how they may affect both the enjoyment of the task and the quality of the final product. Systems with low intrusiveness and a pull method of interaction initiation allow people to write more closely to their normal writing process. However, these characteristics also mean that suggestions are more easily ignored and may never be requested. If the goal is to encourage interaction with the machine or a more structured interaction, a higher intrusiveness and push method system may be better. A careful introduction and framing of the system is also necessary to encourage the desired level of interaction with the machine in the loop.

For story writing and other tasks that expand on a prompt and have an additive interaction structure, systems may benefit from an interface that supports outlining or non-linear writing. For example, Flower and Hayes [Flower and Hayes, 1981] describe the hierarchical nature of the creative writing process; future system designs could reflect knowledge about the cognitive processes of writing to better support the writing process. The slogan writing task, along with other condensing writing tasks that have an iterative structure, may benefit from an interface that provides more space for brainstorming and drafting slogans.

We recommend using models that strike a balance between generating coherent suggestions and surprising suggestions. An element of randomness provides new ideas and directions for a writer, but suggestions too far away from the writer's ideas may be unhelpful and ignored. We recommend pushing towards surprising suggestions for tasks that use a pull method of initiation and have a low level of intrusiveness because when writers decide to initiate a suggestion loop, it generally means they are stuck or at least open to new ideas. Although surprising suggestions run the risk of being irrelevant, a less intrusive system means unhelp-

ful suggestions can be easily ignored and minimally interrupt the writing process. Coherence should be a bigger priority for push method, high-intrusiveness interactions, as high levels of randomness in suggestions may distract writers.

For the story writing task, we found that participants wanted more coherent suggestions from the model. For similar tasks, we recommend working towards incorporating more context into the suggestion-generating models. Characters may be an important aspect of the context to consider, as suggestions with incorrect pronouns or that lack references to existing characters are difficult to work into a story, and we recommend more research into character or entity-focused modeling, like Ji et al. [2017]. Models would also benefit from the ability to play with the content type of its suggestions, such as choosing to offer lines of dialogue, action-driven sentences, or descriptive lines.

Models for writing slogans should provide more variety in their suggestions, particularly on a lexical level, as diverse language is important for an iterative task. Models that can generate related keywords, synonyms, and alliterative words when given a person's ideas would be useful for this type of task.

We noted that there is little to no correlation between the ratings that writers gave themselves and the ratings that Amazon Mechanical Turker workers gave them. This observation echoed the finding in Tan et al. [2014] that it is hard for humans to evaluate the quality of writing. Therefore, machine-in-the-loop writing systems that aim to improve a writer's work should measure the system's success not only as perceived by the writer but also by third-party evaluators.

Chapter 3

Story Generation with Entity Representations as Context

In Chapter 2, we discussed some promising directions for improving language generation models for collaborative writing. In this chapter, we propose a model to address one of the weaknesses our user studies in Chapter 2 exposed: models' difficulty to refer back to existing characters and existing entities in the text in a natural way (as illustrated in Figure 2.5).

The work in this chapter is published in Clark et al. [2018a].

3.1 Introduction

In this chapter, 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].

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,

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 3.1: 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 gold sentence reads, "*Seth yelled at her to get back but she ignored him.*"

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.

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 chapter, 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 3.1. 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 due to 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.

3.2 Model Description

We propose an entity-based generation model (ENGEN) 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 §3.2.1 and §3.2.2, respectively. The combination of all three context representations is described in §3.2.3.

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

$$\boldsymbol{p}_{t-1,i} = \sum_j \alpha_{i,j} \boldsymbol{h}_{t-1,j}, \text{where} \tag{3.1}$$

$$\alpha_{i,j} = \frac{\exp(\boldsymbol{h}_{t-1,j} \mathbf{W}_a \boldsymbol{h}_{t,i})}{\sum_{j'} \exp(\boldsymbol{h}_{t-1,j'} \mathbf{W}_a \boldsymbol{h}_{t,i})} \tag{3.2}$$

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 3.2 to encourage correlation between $\boldsymbol{h}_{t,i}$ and $\boldsymbol{h}_{t-1,j}$ for coherence in text generation. In §3.2.3, 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.

3.2.2 Context from Entities

In S2SA, the context of a sentence is (at best) represented by compressing information about the words that have appeared in 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.[1] 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.[2] 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 (e.g., EMILY in Figure 3.1) in a document 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 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

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

[2] The entity prediction task used in their work is relevant to our mention generation task, which will be discussed in §3.5.

referring to a given entity $i \in \{1, \ldots, m+1\}$ is proportional to:

$$\exp(\boldsymbol{h}_{t-1}^{\top} \mathbf{W}_{entity} \boldsymbol{e}_{i,t-1} + \boldsymbol{w}_{dist}^{\top} \boldsymbol{f}(i)), \tag{3.3}$$

where $\mathbf{W}_{entity}$ is a weight matrix for predicting the entities and $\boldsymbol{w}_{dist}^{\top} \boldsymbol{f}(i)$ 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 $\boldsymbol{e}_{current}$, which is used to generate the word w_t. If the model decided the current word should not refer to an entity, then $\boldsymbol{e}_{current}$ is still used and will be the representation of the most recently mentioned entity. If the choice is a new, previously unmentioned entity, then $\boldsymbol{e}_{current}$ is initialized with a new embedding randomly generated from a normal distribution:

$$\boldsymbol{u} \sim \mathcal{N}(\boldsymbol{r}, \sigma^2 \mathbf{I}), \tag{3.4}$$

where $\sigma = 0.01$ and $\boldsymbol{r}$ is a parameterized embedding 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 $\boldsymbol{h}_t$.

3.2.3 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 ($\boldsymbol{e}_{current}$). 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 $\boldsymbol{p}_t$, $\boldsymbol{e}_{current}$). Specifically, at time step t, each element of the combined context vector $\boldsymbol{c}_t$ is calculated as follows. For $k \in \{1, ..., |\boldsymbol{c}_t|\}$,

$$\boldsymbol{c}_t[k] = \max(\boldsymbol{h}_{t-1}[k], \boldsymbol{p}_t[k], \boldsymbol{e}_{current}[k]). \tag{3.5}$$

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.

3.2.4 Learning

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

$$\ell(\boldsymbol{\theta}) = \log P(\boldsymbol{X}; \boldsymbol{\theta}) = \sum_t \log P(X_t; \boldsymbol{\theta}) \tag{3.6}$$

$\boldsymbol{\theta}$ 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).

3.2.5 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.

3.3 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.

3.4 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

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 3.1: 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.

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 3.6).

3.5 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 3.1, *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 3.1, it would select between all the previous entity mentions (*Emily*, *the dragon*, *Seth*, and *her*) and the correct mention (*she*).

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

Figure 3.2: Candidate lists for each of the mention generation tasks. The asterisk (*) indicates the correct choice.

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 most precision, as it is 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 3.1 are shown in Figure 3.2.

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.[3] 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).

[3]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.

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

Results The ranking results of ENGEN and other systems are reported in Table 3.1. 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 it is not an effective heuristic for mention generation.

3.6 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 (§3.7). 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 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 5.3. 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

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 3.2: 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.

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 3.6. 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 candidate sentences and averaged their performance across all 5 rounds.

Results The accuracy of each of the models is reported in Table 3.2. 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 random sampling to choose a sentence from the following segment as the distractor sentence, the standard deviations in Table 3.2 show the results are generally consistent across rounds, regardless of distractor's distance from the gold sentence.

3.7 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[4] 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.

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 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 §6.2.2. However, for the human evaluation, we only used the final 60 words[5] 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 qualitative portion of this task would be easy to scale, the qualitative portion is not; we kept the

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

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

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 3.3: 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.

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 3.3.

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 they 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 3.3, 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.

3.8 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, that 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 exam-

ple, 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 (§3.6). 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]. Henaff et al. [2017] incorporate entities in their EntNet model, but they do not use coreference information to update specific entity representations as they are mentioned in the text. Neural Process Networks [Bosselut et al., 2018] track and update entity representations, but with the goal of modeling actions and their causal effects on entities.

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.

Story generation Work in story generation has incorporated structure and context through event representations [Martin et al., 2018] or semantic representations, like story graphs [Elson and McKeown, 2009; Rishes et al., 2017]. 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].

Chapter 4

Paired Suggestions in Collaborative Writing for Evaluating Generation Models

In this chapter we turn to the challenge of evaluation in natural language generation. In particular, text generated in an open-ended and collaborative setting can be difficult to evaluate, given its subjective and unrestricted nature. While the previous chapters explored the benefit of NLG in a collaborative setting for writers, this chapter looks at how deploying NLG models in collaborative settings can serve as an evaluation tool for researchers as they develop new generation models.

The work in this chapter is published in Clark and Smith [2021].

4.1 Introduction

Systems that automatically generate text suggestions to human authors have emerged as a new application of natural language generation models. Evaluating such models, however, is challenging. Typically, writers rate a single system's quality after some period of use, for example while authoring an entire story or poem [e.g., Clark et al., 2018b; Ghazvininejad et al., 2017]. A model's quality is measured using Likert scale scores, sometimes combined with additional analysis, like the type or quantity of writer edits [e.g., Roemmele and Gordon, 2015; Akoury et al., 2020].

In contrast, a *pairwise* system evaluation—where evaluators are given two suggestions at the same time

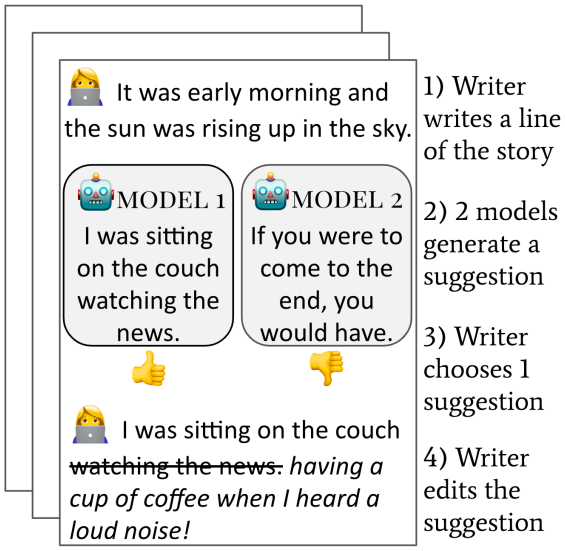


Figure 4.1: CYOA has a writer write a line of the story alone, and then two models generate suggestions for the next line. The writer chooses one (in this case, MODEL1), edits it, and then adds it to the story. They repeat this process 5 times. CYOA collects writers' preferences between the two models, along with the human-authored, machine-generated, and human-edited text, to evaluate the models.

and asked to choose between them—would allow researchers to compare generation models directly. Comparative evaluations have been shown to produce more reliable and consistent results than Likert-scale ratings [Callison-Burch et al., 2007; Kiritchenko and Mohammad, 2017], and they have been used to evaluate natural language generation systems for translation and dialogue [Otani et al., 2016; Sedoc et al., 2019].

We propose CHOOSE YOUR OWN ADVENTURE (CYOA), a protocol for pairwise evaluations of collaborative writing models, focusing on story generation. Instead of scoring a single model, we compare two models. At fixed points during the writing process, each generates a suggestion, and writers choose one to continue their story (see Fig. 6.1). The result is utterance-level feedback on which model's generated text writers prefer at that point in the story. Along with the writer's revisions to the generated suggestions and comparisons between the generated and human-authored portions of the story, this evidence can help a researcher answer the following questions about their model:

1. Is my model better at generating story suggestions than a baseline model?
2. How useful are my model's suggestions?
3. How does my model's generated text compare to human-authored text?

In this chapter, we show how CYOA can answer these questions and provide insights into story model behavior, both in cases when the expected differences in text quality are large (e.g., the text is generated

with two different models; §4.3) and when they are small (e.g., the text is generated with the same model but using two different sampling methods; §4.4).

CYOA allows human and automatic evaluations to be collected simultaneously; we run standard automatic evaluations of text quality on the collaboratively-generated text and get results consistent with previous analyses of "statically"-generated text. CYOA is useful to both NLG researchers and story writers; writers report being happy with the stories they write with the system and that the paired suggestions help them come up with new ideas. We release a template website for CYOA and the evaluation script[1] to support future story and collaborative writing evaluation work.

4.2 CHOOSE YOUR OWN ADVENTURE

CYOA evaluates a pair of story generation models by having people select and interact with text generated by each of the models as they write a story. Both models generate suggestions for the writer at the same point in the story, and the writer must choose between the two suggestions, forcing a pairwise comparison of the two models. By having multiple people write stories with the two models, we can aggregate their preferences and interactions with the suggestions and analyze them to provide feedback on the two models.

4.2.1 Writing Setup

To allow the writers control over the story while still encouraging them to use the suggestions, CYOA uses a turn-taking writing process, with writers alternating between writing by themselves and then receiving suggestions to continue the story [Swanson and Gordon, 2012; Clark et al., 2018b].

The writer begins the story by writing the first sentence alone; an image (Fig. A.1 in App. A.1) is provided as an optional prompt to help them get started. Once the writer submits the writing from their turn, two models each generate a suggestion to continue the story, which are presented to the writer in random order. As shown in Fig. 6.1, the writer then chooses which of the suggestions they prefer and edits it as they wish before adding it to the story. It is then the writer's turn to write alone again. This process repeats 5 times, at which point the story is finished and submitted.

[1] github.com/eaclark07/cyoa

Each "turn" in the story has to be between 20 and 260 characters for it to be submitted to the story. Other than length, there is no restriction on how writers can edit the suggestions; they can delete the suggestion entirely or submit it as-is. When editing a computer-generated suggestion, the writer can change their mind and select the other model's suggestion instead, but once a writer submits a turn, they cannot go back to edit it later.

After the finished story is submitted, participants are asked Likert-scale and open-ended questions about the system and the suggestions they received. We asked participants to indicate on a 5-point Likert scale (ranging from "Strongly Disagree" to "Strongly Agree") how much they agreed with the following statements:

- I'm happy with my final story.
- I felt the system and I were working collaboratively to write the story.
- I thought having the suggestions was useful while writing the story.
- The suggestions connected to what had happened in the story so far.
- The suggestions helped me come up with new ideas.

We then provided textboxes for them to write their responses to the following questions:

- What made you choose one suggestion over another?
- What were you looking for in the suggestions?

We chose these questions for this project to capture people's reactions to the overall writing setup and a general sense of areas for improving story generation models. However, these questions could be eliminated or adjusted to fit the evaluation goals of the researcher.

A demo of CYOA is at `homes.cs.washington.edu/~eaclark7/multi-model-demo`.

4.2.2 Evaluation Setup

From the writing setup, we collect the generated suggestions from each model, the writers' preferences between the two models, and the revisions they make to the generated text. We analyze these sources of information to answer three questions NLP practitioners have when evaluating their models. There are many analyses researchers could run with the data gathered from CYOA beyond those listed here; we include some examples.

(Q1) Is my model better at generating story suggestions than a baseline model? CYOA reports how many of the model's suggestions people chose to work with vs. the baseline's suggestions. We further break this down by the suggestion round (1–5) to see if the writers' preferences change over the course of the story.

Another option would be to break down the writers' preferences by writer attributes, e.g., to analyze the effect of the author on the stories or desired suggestions [August et al., 2020].

(Q2) How useful are the models' suggestions? We analyze the revisions writers make to the suggestions to see how much of the generated text they find useful for continuing their story. We use three metrics to see how much of the original text is preserved after a writer's revisions. Levenshtein edit distance measures the number of character insertions, deletions, and substitutions the writers made, and Jaccard similarity measures the proportion of tokens that are shared between the original and the edited text. User Story Edit Ratings (USER; Akoury et al., 2020)[2] measures similarity by recursively counting the longest contiguous substrings between the edited and the original text.

These edit-based metrics capture exact matches between the texts, measuring how much of the generated content makes it to the final story in the strictest sense. However, other metrics could be used if the researcher is interested in capturing broader notions of similarity, e.g., embedding-based measures like cosine similarity or BERTScore [Zhang et al., 2020b].

(Q3) How do the models' generated texts compare to human-authored text? Pairwise comparison gives us the models' relative quality; comparing them to human-authored text gives an idea of their absolute quality. To do this, we take the parts of the story the writer wrote alone (i.e., the turns without generated suggestions) and compare it to the generated text. We look at average sentence length (a common proxy for text complexity in stories; See et al., 2019; Roemmele et al., 2017) and distinct-n, a measure of repetition [Li et al., 2016]. As in See et al. [2019], we also look the concreteness of the text's nouns and verbs, using the concreteness ratings from Brysbaert et al. [2014].[3]

If the system is being used to evaluate a model that focuses on a specific aspect of stories, e.g., events or characters, this analysis could be extended to compare how these specific elements are introduced and referenced in the machine-generated vs. human-authored text.

[2] `github.com/dojoteef/storium-frontend`

[3] Sentence length, concreteness, and distinct-n: `github.com/abisee/story-generation-eval`

	Total	#1	#2	#3	#4	#5
% GPT2	66	76	70	63	63	57

Table 4.1: % of chosen suggestions that are from GPT2.

4.3 Experiment #1: FUSION vs. GPT2

We first test CYOA with two popular story generation models: (1) FUSION, the fusion model from Fan et al. [2018], which uses a fusion mechanism to combine two convolutional sequence-to-sequence models; and (2) GPT2, the small GPT2 model [Radford et al., 2019] finetuned on story data and using top-k sampling [Fan et al., 2018].

We compare FUSION and GPT2 to see how CYOA can evaluate two models with different underlying architectures; they are also both common story generation baselines [See et al., 2019; Xu et al., 2020; Rashkin et al., 2020].

To train the models, we use the WritingPrompts dataset Fan et al. [2018], a collection of writing prompts from Reddit paired with stories. During the CYOA evaluation, both models generate their suggestions conditioned on the whole story written so far. (Data and model details in App. A.2 and A.3.)

We run CYOA on Amazon Mechanical Turk with 105 Turkers to compare the two models. Each Turker can only complete the task once. Turkers are required to have over 1,000 tasks approved, have an 95% approval rate, and be from the United States, and they are paid $2.50 for participating in the study. The study was approved by our instiution's Institutional Review Board.

We break down our results and discussion by the research questions listed in §4.2.2.

(Q1) Table 4.1 shows that, of the 525 suggestion pairs, Turkers significantly[4] preferred the GPT2 suggestions over FUSION, choosing them 65.7% of the time. Breaking it down by suggestion round 1–5, the writers' preference for the GPT2 was largest at the beginning of the story and decreased over the course of the story. To understand why, we look at how writers edited the suggestions and how the generated text compared to human-authored text.

[4]Binomial test: $p < 0.01$.

	ED (↓)	JS (↑)	USER (↑)
FUSION	37.61	51.13	60.69
GPT2	**29.49**	**61.35**	**71.77**

Table 4.2: Edit distance, Jaccard similarity, and USER scores between the edited and the original suggestions.

	FUSION	GPT2	HUMAN
avg. sent. len.	13.70	10.31	18.86
concrete N	4.04	4.35	4.17
concrete V	2.90	3.10	3.12
distinct-1	0.75	0.53	0.72
distinct-2	0.97	0.70	0.95
distinct-3	1.00	0.76	0.99

Table 4.3: Generated text results for the FUSION and GPT2-generated text, compared to the HUMAN-written portions of the story.

(Q2) In Table 4.2, all three edit metrics show that writers used significantly[5] more of the accepted GPT2 suggestion text in their story than the accepted FUSION suggestion text. When we break down the scores by round, we see that this is true regardless of where in writer is in the story (see Table A.1 in App. A.4.1). Taken with the pairwise results, this points to GPT2 as the better collaborative story generation model. FUSION, perhaps due to its hierarchical structure, did not generate as many useful suggestions as GPT2 in the interactive setting.

(Q3) Finally, we look at how the generated text compares to the story text the writers wrote alone. From Table 4.3, we see that GPT2 generates shorter, more concrete, and more repetitive suggestions than FUSION.

Both models generate shorter sentences than people, and GPT2 generates more concrete nouns and verbs than FUSION, corroborating the analysis of See et al. [2019]. GPT2 generated the most repetitive text, which may explain why it is chosen less frequently as the story goes on. FUSION's sub-human level of repetition indicates it often fails to refer back to the story context, as illustrated by the low Likert-scale scores for *The suggestions connected to what had happened in the story so far.* (Fig. A.2 in App. A.4.2).

[5]Mann-Whitney U test: $p < 0.01$.

	Total	#1	#2	#3	#4	#5
% TOP-K	53	58	53	53	53	49

Table 4.4: % of chosen suggestions that are from TOP-K.

	ED (↓)	JS (↑)	USER (↑)
NUCLEUS	**34.65**	**53.64**	**63.64**
TOP-K	36.69	50.96	62.18

Table 4.5: Edit distance, Jaccard similarity, and USER scores between the edited and the original suggestions.

4.4 Experiment #2: NUCLEUS vs. TOP-K

Our second experiment compares text generated from GPT2 but now using different sampling strategies: TOP-K (as in §4.3) and NUCLEUS sampling [Holtzman et al., 2020]. (Model details in App. A.3.) Here we expect to see narrower differences in the generated text than we did in §4.3. Comparing TOP-K vs. NUCLEUS focuses on CYOA's ability to compare models with fine-grained differences. 103 Turkers[6] write a story with the help of suggestions from this pair of models.

(Q1) Table 4.4 shows Turkers preferred the TOP-K suggestions over the NUCLEUS suggestions for 53.4% of the 515 suggestion pairs writers received; as expected, a smaller difference than in §4.3 and not significant.[7] Again, the writers' preference for TOP-K decreased over the course of the story, with NUCLEUS slightly more popular by the end.

(Q2) In Table 4.5, all three metrics show that writers used more of the NUCLEUS-sampled text than the TOP-K-sampled text, though the difference is not significant.[8] Despite writers' slight preference for TOP-K-sampled suggestions, when they choose NUCLEUS-sampled suggestions, they preserve more of the generated text. Table A.2 (App. A.4.1) shows that difference is largest at the beginning and end of the story. This suggests TOP-K's safer suggestions may be less useful, especially when starting or finishing the task.

[6]These are a separate set of Turkers from §4.3, but subject to the same requirements.
[7]Binomial test: $p = 0.07$.
[8]Mann-Whitney U test: $p = 0.19$ (ED), $p = 0.23$ (JS), and $p = 0.27$ (USER).

	NUCLEUS	TOP-K	HUMAN
avg. sent. len.	12.76	10.53	19.28
concrete N	4.15	4.34	4.23
concrete V	3.08	3.08	3.11
distinct-1	0.77	0.60	0.72
distinct-2	0.96	0.78	0.96
distinct-3	0.99	0.84	0.99

Table 4.6: Generated text results for the TOP-K and NUCLEUS-generated text, compared to the HUMAN-written portions of the story.

(Q3) Table 4.6 shows that TOP-K-generated text is shorter, more concrete, and more repetitive than NUCLEUS-generated text. NUCLEUS's text comes closer to human-levels of repetition, consistent with the findings of Holtzman et al. [2020] and Akoury et al. [2020].

4.5 Writer Feedback

CYOA benefits writers as well as researchers. The results of the writer feedback across both experiments indicate that writers enjoy the paired-suggestion writing experience, regardless of which models they wrote with. The Likert-scale responses were particularly positive for *I'm happy with my final story.* (FUSION vs. GPT2: mean $= 3.83$, NUCLEUS vs. TOP-K: mean $= 3.84$) and *The suggestions helped me come up with new ideas.* (FUSION vs. GPT2: mean $= 3.80$, NUCLEUS vs. TOP-K: mean $= 3.79$). This compares favorably to single-suggestion collaborative story writing systems that use a similar writing process; Clark et al. [2018b] report writers gave a mean score of 3.28[9] for happiness with the story they wrote with their collaborative writing system. Full Likert-scale results are in App. A.4.2.

The positive reactions from participants indicate this format could work well on alternative crowdsourcing platforms, like LabintheWild,[10] or launched as an independent writing game, similar to Akoury et al. [2020].

[9] Scoring adjusted to a 5-point scale.

[10] `www.labinthewild.org`

4.6 Related Work

Collaborative writing systems have been developed in domains like poetry [Ghazvininejad et al., 2017], slogans [Clark et al., 2018b], and stories [Roemmele and Gordon, 2015; Goldfarb-Tarrant et al., 2019; Akoury et al., 2020]. Like Storium [Akoury et al., 2020], we focus on the potential to use these systems as evaluation platforms. However, we suggest using paired suggestions in collaborative writing systems to directly compare generation models.

ChatEval [Sedoc et al., 2019] collects human evaluations for paired chatbot utterances and Otani et al. [2016] for paired translations, but the generated text is static. By having writers interact with dynamically generated suggestions, collaborative writing systems reward *helpful* and *robust* generation models, under-emphasized attributes in current evaluations [Zellers et al., 2021; Ethayarajh and Jurafsky, 2020].

4.7 Conclusion

CYOA allows researchers to collect human and automatic evaluations for story generation models in a single collaborative writing task. The paired suggestions allow direct comparisons between two models, and automatic-metric comparisons among generated text, its revisions, and the human-authored portions provide additional insight. We expect CYOA evaluations to accelerate progress on applications for collaborative writing between humans and machines.

Chapter 5

Automatic Evaluation for Multi-Sentence Text

Human evaluation methods like CYOA (Chapter 4) are the most valuable form of evaluation for generation models, as they point to the usefulness of NLG models in downstream tasks like collaborative story writing. However, automatic metrics are essential to NLG, particularly in the model development and analysis stages. In this chapter, we propose and evaluate an automatic metric for long generated texts.

The work in this chapter is published in Clark et al. [2019].

5.1 Introduction

While automatic metrics allow faster quality feedback and model development, existing automatic metrics for evaluating text are problematic. Due to their computational efficiency, metrics based on word-matching are common, such as ROUGE [Lin, 2004] for summarization, BLEU [Papineni et al., 2002] for machine translation, and METEOR [Banerjee and Lavie, 2005] or CIDER [Vedantam et al., 2015] for image captioning. Nevertheless, these metrics often fail to capture information that has been reworded or reordered from the reference text, as shown in Kilickaya et al. [2017] and Table 5.1.[1] They have also been found to correlate weakly with human judgments [Liu et al., 2016; Novikova et al., 2017].

To avoid these shortcomings, word mover's distance (WMD; Kusner et al., 2015) can be used to evaluate

[1] For readability, we scale ROUGE scores by a factor of 100 and sentence mover's metrics by a factor of 1000.

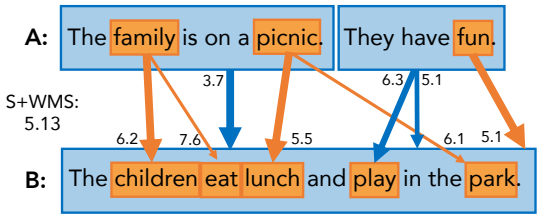


Figure 5.1: An illustration of S+WMS (a sentence mover similarity metric that uses both word and sentence embeddings) between two documents. This metric finds the minimal cost of "moving" both the word embeddings (orange) and the sentence embeddings (blue) in Document A to those in Document B. An arrow's width is the proportion of the embedding's weight being moved, and its label is the Euclidean distance. Here we show only the highest weighted connections.

text in a continuous space using pretrained word embeddings instead of relying on exact word matching. WMD has been used successfully for tasks including image caption evaluation [Kilickaya et al., 2017], automatic essay evaluation [Tashu and Horváth, 2018], and affect detection [Alshahrani et al., 2017]. This bag-of-embeddings approach is flexible but fails to reflect the grouping of words and ideas, a shortcoming that becomes more problematic as the length of the document grows.

Reference passage. the only thing crazier than a guy in snowbound massachusetts boxing up the powdery white stuff and offering it for sale online ? people are actually buying it . for $ 89 , self-styled entrepreneur kyle waring will ship you 6 pounds of boston-area snow in an insulated styrofoam box – enough for 10 to 15 snowballs , he says .				
Summary	**ROUGE-L**	**WMS**	**SMS**	**S+WMS**
Human summary. a man in suburban boston is selling snow online to customers in warmer states . for $ 89 , he will ship 6 pounds of snow in an insulated styrofoam box .	39.30	57.85	99.98	24.06
Word order. in suburban boston , a man is selling snow online to customers in warmer states . he will ship 6 pounds of snow in an insulated styrofoam box for $ 89 .	31.44 (↓ 20%)	57.85 (=)	99.98 (=)	24.06 (=)
Repetition. a man in suburban boston is selling snow is selling snow online to customers in warmer states in warmer states . for $ 89 , he will ship he will ship 6 pounds 6 pounds of snow in an insulated styrofoam box in a styrofoam box .	35.07 (↓ 11%)	57.31 (↓ 1%)	89.40 (↓ 11%)	22.81 (↓ 5%)

Table 5.1: A comparison of scores for three different summaries for a reference passage (the first lines of a news article). The human summary has been permuted with its clauses rearranged (Word order) and repeated (Repetition). Word order changes negatively affect ROUGE-L more than repetition; the other metrics are unaffected by word order choices but, to varying degrees, penalize repetition.

In this chapter, we modify WMD for evaluating multi-sentence texts by basing the score on sentence embeddings (§5.3), giving it access to higher-level representations of the text. We introduce two new metrics: **sentence mover's similarity** (SMS), which relies only on sentence embeddings, and **sentence and word**

mover's similarity (S+WMS), which uses word and sentence embeddings, as in Figure 5.1.

In §5.4, we find that sentence mover's similarity metrics significantly improve correlation with human evaluations over ROUGE-L (the longest common subsequence variant of ROUGE) and WMD when scoring automatically generated summaries (averaging 3.4 sentences). We also automatically evaluate human-authored essays (averaging 7.5 sentences) and find smaller but significant gains. We compute sentence mover's similarity metrics with type-based embeddings and contextual embeddings and find these results hold regardless of embedding type, with no significant difference caused by the choice of embedding.

Finally, we show in §5.6 that sentence mover's similarity metrics can also be used when learning to generate text. Generating summaries using reinforcement learning with sentence mover's similarity as the reward results in higher quality summaries than those generated using a ROUGE-L or WMD reward, according to both automatic metrics and human evaluations.

5.2 Background: Word Mover's Distance

Earth mover's distance (EMD, also known as the Wasserstein metric; Rubner et al., 1998) is a measure of the distance between two probability distributions. Word mover's distance (WMD; Kusner et al., 2015) is a discrete version of EMD that evaluates the distance between two sequences (e.g., sentences, paragraphs, etc.), each represented with relative word frequencies. It combines (1) item similarity[2] on bag-of-word (BOW) histogram representations of text [Goldberg et al., 2018] with (2) word embedding similarity.

For any two documents A and B, WMD is defined as the minimum cost of transforming one document into the other. Each document is represented by the relative frequencies of words it contains, i.e., for the ith word type,

$$d_{A,i} = \text{count}(i)/|A| \tag{5.1}$$

where $|A|$ is the total word count of document A, and $d_{B,i}$ is defined similarly.

Now let the ith word be represented by $\mathbf{v}_i \in \mathbb{R}^m$, i.e., an m-length embedding,[3] allowing us to define distances between the ith and jth words, denoted $\Delta(i, j)$. V is the vocabulary size. We follow Kusner et al.

[2]The similarity can be defined as cosine, Jaccard, Euclidean, etc.

[3]Our evaluation scores depend on pretrained word embeddings, which can be type-based or contextual. Our experiments consider both; see §5.4 and §5.6. When using contextual embeddings, we treat each token as its own type, as each word will have a different embedding depending on its context.

[2015] and use the Euclidean distance $\Delta(i,j) = \|\mathbf{v}_i - \mathbf{v}_j\|_2$. The WMD is then the solution to the linear program:

$$\text{WMD}(A,B) = \min_{\mathbf{T} \geq \mathbf{0}} \sum_{i=1}^{V} \sum_{j=1}^{V} \mathbf{T}_{i,j} \Delta(i,j) \tag{5.2a}$$

s.t.

$$\forall i, \sum_{j=1}^{V} \mathbf{T}_{i,j} = d_{A,i}, \tag{5.2b}$$

$$\forall j, \sum_{i=1}^{V} \mathbf{T}_{i,j} = d_{B,j} \tag{5.2c}$$

$\mathbf{T} \in \mathbb{R}^{V \times V}$ is a nonnegative matrix, where each $\boldsymbol{T}_{i,j}$ denotes how much of word i (across all its tokens) in A is assigned to tokens of word j in B, and the constraints ensure the flow of a given word cannot exceed its weight. Specifically, WMD ensures that the entire outgoing flow from word i equals $d_{A,i}$, i.e., $\sum_j \boldsymbol{T}_{i,j} = d_{A,i}$. Additionally, the amount of incoming flow to word j must match $d_{B,j}$, i.e., $\sum_i \boldsymbol{T}_{i,j} = d_{B,j}$. Following the example of Kilickaya et al. [2017], we transform WMD into a similarity (WMS):

$$\text{WMS}(A,B) = \exp(-\text{WMD}(A,B)) \tag{5.3}$$

WMS measures two documents' similarity by minimizing the total distance to move words between two documents, combining the strengths of BOW and word embedding-based similarity metrics. In Figure 5.1, WMS would calculate the cost of moving from Document A to Document B using only the word embeddings, denoted in orange. WMS is symmetric, and $\text{WMS}(A,A) = 1$ when word embeddings are deterministic.

Empirically, WMD has improved the performance of NLP tasks (see §5.7), specifically sentence-level tasks, such as image caption generation Kilickaya et al. [2017] and natural language inference Sulea [2017]. However, its cost grows prohibitively as the length of the documents increases, and the BOW approach can be problematic when documents become large as the relation between sentences is lost. By only measuring word distances, the metric cannot capture information conveyed by the grouping of words, for which we need higher-level document representations [Dai et al., 2015; Wu et al., 2018].

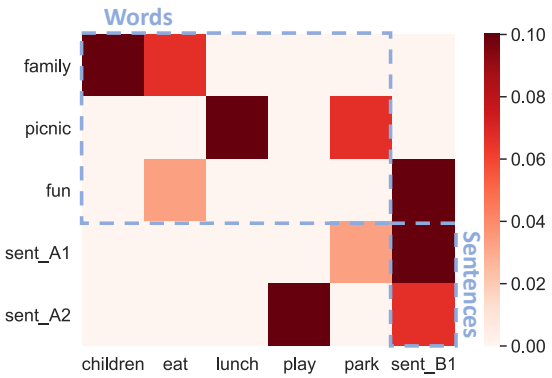


Figure 5.2: The S+WMS $\boldsymbol{T}$ matrix for documents A and B from Figure 5.1 (with empty rows/columns removed). Contrarily, WMS's $\boldsymbol{T}$ matrix only maps between words and has the dimensions of the dashed region labeled "Words," and SMS's maps between sentences in the shape of the dashed region "Sentences." Best viewed in color.

5.3 Sentence Mover's Similarity Metrics

We modify WMS to measure the similarity between two documents using sentence embeddings, which we call a sentence mover's similarity approach. We introduce two new metrics: Sentence Mover's Similarity (SMS) and Sentence and Word Mover's Similarity (S+WMS). SMS replaces the word embeddings in WMS with sentence embeddings (§5.3.1), while S+WMS combines the two metrics and uses both word and sentence embeddings (§5.3.2). Our code (an extension of an existing WMD implementation[4]) and datasets are publicly available.[5]

5.3.1 Sentence Mover's Similarity

Sentence Mover's Similarity (SMS) performs the same linear optimization problem in Eq. 5.2a as WMS, except now each document is represented as a bag of sentence embeddings rather than a bag of word embeddings. In Figure 5.1, SMS considers only the sentence embeddings, denoted in blue.

To get the representation of a sentence in a document, we combine the sentence's word embeddings. Sentence representations based on averaging or pooling word embeddings perform competitively on tasks including sentence classification, recognizing textual entailment, and paraphrase detection [Conneau and Kiela, 2018]. We use sentence representations that are the average of their word embeddings, as this approach outperformed pooling methods in preliminary results.

[4] https://github.com/src-d/wmd-relax
[5] https://github.com/eaclark07/sms

While in WMS word embeddings are weighted according to their frequency in the document (see Eq. 5.1), SMS weights each sentence embedding by the number of words ($|A|$) it contains.[6] So a sentence i in document A will receive a weight of:

$$d_{A,i} = |i|/|A| \tag{5.4}$$

We solve the same linear program, Eq. 5.1, by calculating the cumulative distance of moving a document's sentences to match another document. Now the vocabulary is the set of sentences in the documents instead of the words, as in Figure 5.2.

5.3.2 Sentence and Word Mover's Similarity

Sentence and Word Mover's Similarity (S+WMS) combines WMS and SMS and represents each document as a collection of both words and sentences. Each document is now a bag of both word and sentence embeddings (as seen in Figure 5.1), where each word embedding is weighted according to its frequency and each sentence embedding is weighted according to its length. Now the bag of words and sentences representing document A is normalized by $2|A|$, so that:

$$d_{A,i} = \begin{cases} count(i)/2|A|, & \text{if } i \text{ is a word} \\ |i|/2|A|, & \text{if } i \text{ is a sentence} \end{cases} \tag{5.5}$$

As in WMS and SMS, the same linear program in Eq. 5.1 is solved, this time calculating the cumulative distance of moving both a document's words and sentences to match another document. The vocabulary is the set of sentences and words in the documents (see Figure 5.2). The sentence embeddings are treated the same as word embeddings in the optimization; the only difference is their length-based weights.

This means a sentence embedding can be mapped to a word embedding (e.g., "They have fun." maps to "play" in Figure 5.1) or vice versa. It also means that a sentence's words do not have to move to the same word or sentence embedding(s) that their sentence moves to (as seen in Figure 5.1); a sentence in document A could be transported to an embedding in document B and have none of its words moved to the same embedding. More constraints could be introduced to further control the flow between documents, which we leave to future work.

[6] Preliminary results showed count-based sentence weightings performed better than uniform weightings. Other weighting options, such as frequency-based weighting as done in BERTScore [Zhang et al., 2020b], are a direction for extending this work.

5.4 Intrinsic Evaluation

To test the performance of the SMS and S+WMS metrics, we first examine their usefulness as evaluation metrics. (In §5.6, we evaluate their performance as cost functions for an extrinsic task, abstractive summarization.)

We measure the correlations between the scores assigned to texts by various automatic metrics (ROUGE-L, WMS, SMS, S+WMS) and the scores assigned by human judges. We are interested in *multi-sentence* texts, both machine- and human- generated. Therefore, we consider subsets of two corpora that have been judged by humans: a collection of automatically generated summaries of articles in the CNN/Daily Mail news dataset (alongside reference summaries; see Section 5.5; Chaganty et al., 2018; Hermann et al., 2015; Nallapati et al., 2016) and student essays from the Hewlett Foundation's Automated Student Assessment Prize (Appendix B.2).[7] Statistics describing the datasets are in B.1.

Because the word and sentence mover's similarity metrics are based on pretrained representations, we explore the effect of varying the word embedding method. We present results for two different types of word embeddings: GloVe embeddings [Pennington et al., 2014] and ELMo embeddings[8] [Peters et al., 2018; Gardner et al., 2018]. We obtain GloVe embeddings, which are type-based, 300-dimensional embeddings trained on Common Crawl,[9] using spaCy,[10] while the ELMo embeddings are character-based, 1,024-dimensional, contextual embeddings trained on the 1B Word Benchmark [Chelba et al., 2013]. We use ELMo to embed each sentence, which produces three vectors for each word, one from each layer of the model. We average the vectors to get a single embedding for each word in the sentence.

All correlations are Spearman correlations [Elliott and Keller, 2014; Kilickaya et al., 2017], and significance in the improvement between two metrics' correlations with human judgment is calculated using the Williams [1959] significance test.[11]

[7] https://www.kaggle.com/c/asap-eas
[8] https://allennlp.org/elmo
[9] http://commoncrawl.org/the-data/
[10] https://spacy.io/models/en#en_core_web_md
[11] https://github.com/ygraham/nlp-williams

Samples	Summaries	Metric	Score
Sample #1	**Reference.** Freddie Gray, who is black, asked for medical help but was denied during 00-minute police car ride, eventually paramedics were called. Deputy police commissioner Kevin Davis conceded their failure. But chief commissioner refuses to resign over the death. Six officers are suspended without pay during an investigation.	Human	0.00
		ROUGE-L	*12.44*
	Hypothesis. Baltimore Police Commissioner Anthony Batts ruled out his resignation despite that fact that his deputy admitted they should have sought medical attention for Freddie Gray. Six officers have been suspended with pay as local police and federal authorities investigate. Commissioner Anthony Batts has ruled out the possibility of his resignation.	WMS	**21.41**
		SMS	**128.91**
		S+WMS	**47.89**
Sample #2	**Reference.** Choc on Choc's chocolates come in three different flavours. The face of each politician is emblazoned on milk Belgium chocolate bars. Cameron's has blueberries, Clegg is honeycomb and Miliband is raspberry.	Human	-0.5
		ROUGE-L	**34.57**
	Hypothesis. UNK lollies on 273 invalid chocolates come in three different flavours. Contains three different flavours - the colours associated with each leader. David Cameron, Nick Clegg, Nick Clegg, Nick Clegg and David Cameron.	WMS	5.08
		SMS	*51.39*
		S+WMS	*12.25*

Table 5.2: Two examples from the Summaries dataset along with the scores they received (using GloVe) comparing reference (human summary) to hypothesis (model generated summary). Scores that are in the top quartile for a given metric are in green and **bold**. Scores in the bottom quartile are in red and *italics*. Human scores range from –1 to 1. Please see B.3 for details.

5.5 Summaries Dataset Evaluation

To understand how the sentence mover's similarity metrics evaluate automatically generated text, we use the subset of the CNN/Daily Mail dataset for which Chaganty et al. [2018] collected human annotations. Annotators evaluated summaries (generated with four different neural models) on a scale from –1 to 1. We consider the subset of summaries scored by two or more judges, taking the average to be the summary's score. The automatic evaluation metrics score each generated summary's similarity to the human-authored reference summary from the CNN/Daily Mail dataset.

Table 5.3 shows each metric's correlation with the human judgments. SMS correlates best with human judgments, and both sentence-based metrics outperform ROUGE-L and WMS. We find that the difference between GloVe and ELMo's scores is not significant.[12] Figure 5.3 shows correlation across the metrics.

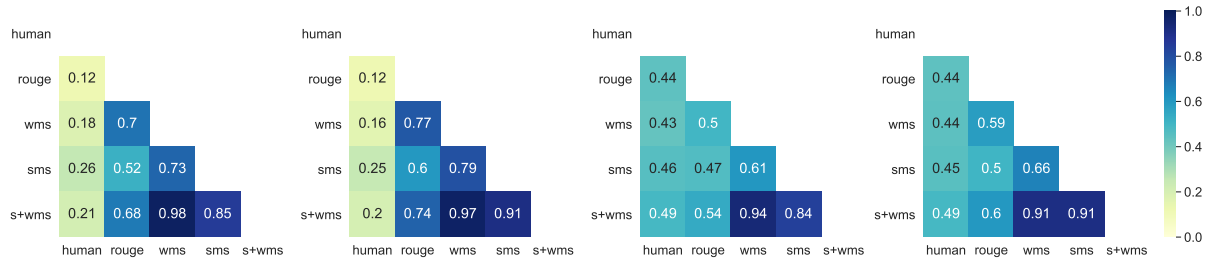
Discussion Two examples of generated summaries and their scores are shown in Table 5.2. Because the scores cannot be directly compared between metrics, we distinguish scores that are in the top quartile for their metric (i.e., the highest rated) and in the bottom quartile (i.e., the lowest rated).

The first example in Table 5.2 is highly rated by metrics using word and sentence embeddings, but judged to be a poor summary by ROUGE-L because information is reworded and reordered from the reference. For example, the phrase "*asked for medical help*" is worded as "*sought medical attention*" in the hypothesis

[12]Williams test: $p = 0.35$ (SMS) and $p = 0.16$ (S+WMS)

	Summaries		Essays	
ROUGE-L	0.117		0.441	
	GloVe	ELMo	GloVe	ELMo
WMS	**0.180	**0.160	0.429	0.443
SMS	**0.258	**0.253	0.457	0.451
S+WMS	**0.214	**0.204	*0.488	*0.490

Table 5.3: Spearman correlation of metrics with human evaluations. Asterisks indicate significant improvement over ROUGE-L, with (*) for $p < 0.05$ and (**) for $p < 0.01$.



(a) Summaries with GloVe embeddings **(b)** Summaries with ELMo embeddings **(c)** Essays with GloVe embeddings **(d)** Essays with ELMo embeddings

Figure 5.3: Spearman correlation with each metric and human evaluations using GloVe and ELMo embeddings on the Summaries and Essays datasets. (Best viewed in color.)

summary. Nevertheless, exact word matching can be important for ensuring factual correctness. While the generated hypothesis summary states "*six officers have been suspended with pay*", the reference states they were actually "*suspended without pay.*"

The second example, which was generated with a seq2seq model, was one of the best summaries according to ROUGE-L but one of the worst according to SMS and S+WMS. It also received low human judgments, most likely due to its nonsensical repetitions. While the short, repeated phrases like "*three different flavours*" match the reference summary well enough to score well with ROUGE-L, the overall sentence representations are distant from those in the reference summary, resulting in low SMS and S+WMS scores.

5.6 Extrinsic Evaluation

In addition to automatically evaluating text, we can also use sentence mover's metrics as rewards while learning text generation models. To demonstrate this, we train an encoder-decoder model on the CNN/Daily Mail dataset to generate summaries using reinforcement learning (RL). Instead of maximizing likelihood, policy gradient RL methods can directly optimize discrete target evaluation metrics that are non-differentiable, such

as ROUGE [Paulus et al., 2018; Jaques et al., 2017; Pasunuru and Bansal, 2017; Wu et al., 2016; Celikyilmaz et al., 2018; Edunov et al., 2018]. Here, we learn policies to maximize WMS/SMS/S+WMS metrics, guiding the model to learn semantic similarities, while policies trained using ROUGE rely only on word n-gram matches between generated and ground-truth text.

Model We encode the input document using 2-layered bidirectional LSTM networks and a 2-layered LSTM network for the decoder. We use the attention mechanism [Bahdanau et al., 2015; See et al., 2017] to force the decoder model to learn to focus (i.e., attend) on specific parts of the input sequence when decoding, instead of relying only on the hidden vector of the decoder's LSTM. We also include pointer networks [See et al., 2017; Cheng and Lapata, 2016], which point to elements of the input sequence at each decoding step.

To train our policy-based generator, we use a mixed training objective that jointly optimizes multiple losses, which we describe below.

MLE Our baseline model uses maximum likelihood training for sequence generation. Given $y^* = \{y_1^*, y_2^*, ..., y_T^*\}$ as the ground-truth summary for a given input document d, we compute the loss as:

$$L_{\text{MLE}} = -\sum_{T=1}^{N} \log p(y_t^* \mid y_1^* \dots y_{t-1}^*, d) \tag{5.6}$$

by taking the negative log-likelihood of the target word sequence.

Model Loss w/ Reward Metric	ROUGE-1	ROUGE-2	ROUGE-L	WMS	SMS	S+WMS
MLE+Pgen [1] (no reward)	36.44	15.66	33.42	-	-	-
MLE+Pgen+RL Mixed w/ ROUGE-L [2]	38.01	16.43	35.49	-	-	-
MLE+Pgen+RL+Intra-Attn Mixed w/ ROUGE-L [3]	39.87	15.82	36.90	-	-	-
MLE+Pgen (no reward) (re-trained baseline)	36.95	15.56	34.00	13.02	90.05	32.15
MLE+Pgen+RL Mixed w/ ROUGE-L	37.46	16.10	34.39	13.07	86.48	31.87
MLE+Pgen+RL Mixed w/ WMS	38.17	**16.52**	34.97	14.52	95.68	34.77
MLE+Pgen+RL Mixed w/ SMS	**38.52**	**16.52**	**35.33**	**15.15**	**96.65**	**35.50**
MLE+Pgen+RL Mixed w/ S+WMS	37.20	15.67	34.15	13.32	91.09	32.64

Table 5.4: Evaluation on summarization task when various metrics are used as rewards during learning. Columns show average score of each model's generated summaries according to various metrics. Previously reported results (upper block): [1] MLE training with pointer networks (Pgen) [See et al., 2017] ; [2] Mixed MLE and RL training with Pgen [Celikyilmaz et al., 2018], [3] Mixed MLE and RL training with Pgen and intra-decoder attention Paulus et al. [2018]. The lower block reports re-trained baselines and our models with new metrics. **Bold** indicates best among the lower block.

Reinforcement Learning (RL) Loss The decoder generates the summary sequence $\hat{y}$, which is then compared against the ground truth sequence y^* to compute the reward $r(\hat{y})$. Our model learns using a *self-critical*

training approach Rennie et al. [2016], by exploring new sequences and comparing them against the best greedily decoded sequence. For each training example d, we generate two output sequences: $\hat{y}$, which is sampled from the probability distribution at each time step, $p(\hat{y}_t \mid \hat{y}_1 \ldots \hat{y}_{t-1}, d)$, and $\tilde{y}$, the baseline output, which is greedily generated by argmax decoding from $p(\tilde{y}_t \mid \tilde{y}_1 \ldots \tilde{y}_{t-1}, d)$. Our mixed training objective is then to minimize:

$$L_{\text{RL}} = (r(\tilde{y}) - r(\hat{y})) \sum_{t=1}^{T} \log p(\hat{y}_t \mid \hat{y}_1 \ldots \hat{y}_{t-1}, d) \tag{5.7}$$

It ensures that, with better exploration, the model learns to generate sequences $\hat{y}$ that receive higher rewards than the baseline $\tilde{y}$, increasing the overall reward expectation of the model.

Mixed Loss While training with only MLE loss will learn a better language model, it may not guarantee better results on discrete performance measures such as WMS and SMS. Similarly, optimizing with only RL loss using SMS as a reward may increase the reward gathered at the expense of diminished readability and fluency of the generated summary. A combination of the two objectives can yield improved task specific scores while maintaining a good language model:

$$L_{\text{MIXED}} = \gamma L_{\text{RL}} + (1 - \gamma) L_{\text{MLE}} \tag{5.8}$$

where γ is a hyperparameter balancing the two objective functions. We pre-train models with MLE loss, and then continue with the mixed loss.

We train four different models on the CNN/Daily Mail dataset using mixed loss (MLE+RL) with ROUGE-L, WMS, SMS, and S+WMS as the reward functions. Training details are in B.4 and B.5.

5.6.1 Generated Summary Evaluation

We evaluate the generated summaries from each model with ROUGE-L, WMS, SMS, and S+WMS in Table 5.4. While we include previously reported numbers, we re-trained the mixed loss models using ROUGE-L and use those as our baseline, as previously trained models should be heavily optimized and use more complex networks than ours. For fair comparison, we kept the encoder-decoder network type, structure, hyperparameters, and initialization the same for each model, changing only the reward. We pre-trained an MLE model ("MLE+Pgen (no reward) (re-trained baseline)" in Table 5.4) and used it to initialize the mixed

loss models with different reward functions.

Across all metrics, the models trained using WMS and SMS metrics as the reward outperform models trained with ROUGE-L as the reward function. S+WMS models lag behind ROUGE-L. The SMS model outperforms all other models across all metrics on the abstractive summarization task, consistent with SMS's performance at evaluating summaries in §5.5.

Table B.4 shows summaries generated from each of the mixed loss models.

5.6.2 Human Evaluation

We also run a human evaluation on the generated summaries, which show people prefer the SMS and S+WMS models' generated summaries over the ROUGE model's. Details are in B.7.

5.7 Related Work

Evaluation has been among the most discussed topics of the natural language generation (NLG) research area Lapata and Barzilay [2005]; Belz and Reiter [2006]; Reiter and Belz [2006]; Barzilay and Lapata [2008]; Reiter and Belz [2009]; Reiter [2011]; Novikova et al. [2017]. There are three main ways to evaluate NLG methods: (1) automatic metrics to compare NLG texts against reference texts, (2) task-based (extrinsic) evaluation to measure the impact of a NLG system on a downstream task, and (3) human evaluations, which ask people to rate generated texts. In this work we introduce new automatic evaluation metrics for long text generation and evaluation.

Automatic evaluation metrics compare generated text against reference texts using word overlap metrics such as: BLEU Papineni et al. [2002]; ROUGE Lin [2004]; NIST Doddington [2002], a version of BLEU; METEOR Lavie and Agarwal [2007], unigram precision and recall; CIDER Vedantam et al. [2015], the average n-gram cosine similarity; *cosine similarity* between the average word embedding; and WMD, which calculates the word embedding-based "travel cost". Though all have strengths and weaknesses, ROUGE metrics (particularly ROUGE-L) are common for multi-sentence text evaluations. Textual metrics that consider specific qualities in the system outputs, like complexity and diversity, are also used to evaluate NLG systems [Dusek et al., 2019; Hashimoto et al., 2019; Sagarkar et al., 2018; Purdy et al., 2018].

Word mover's distance has recently been used for NLP tasks like learning word embeddings [Zhang et al.,

2017; Wu et al., 2018], textual entailment [Sulea, 2017], document similarity and classification [Kusner et al., 2015; Huang et al., 2016; Atasu et al., 2017], image captioning [Kilickaya et al., 2017], document retrieval Balikas et al. [2018], clustering for semantic word-rank Zhang and Wang [2018], and as additional loss for text generation that measures the optimal transport between the generated hypothesis and reference text Chen et al. [2019]. We investigate WMD for multi-sentence text evaluation and generation and introduce sentence embedding-based metrics.

Chapter 6

Human Evaluation of Machine-Generated Text

Though new metrics like SMS and S+WMS (Chapter 5) continue to be proposed to match generation models' improving quality, there is comparatively little research and discussion of human evaluations for NLG, despite its role as the gold-standard evaluation for most NLG tasks. In this chapter, we turn to human evaluations of NLG and look at how crowdsourced evaluators read and evaluate text generated from a state-of-the-art text generation model.

The work in this chapter is published in Clark et al. [2021].

6.1 Introduction

Human-quality text has long been a holy grail for the output of natural language generation (NLG) systems, serving as an upper bound on their performance. Since we lack a good way of encoding many aspects of what constitutes human-quality output in an automated method, we often must rely on human evaluation for our models. Though evaluations with end-users in an applied setting are encouraged Belz and Reiter [2006], in practice, most human evaluations instead ask people to rate generated text's *intrinsic* quality [van der Lee et al., 2019; Howcroft et al., 2020]. Sometimes the generated text is explicitly compared to human-authored text [e.g., Liu et al., 2016; Zellers et al., 2021; Zhang et al., 2020a], but even when no human-authored text

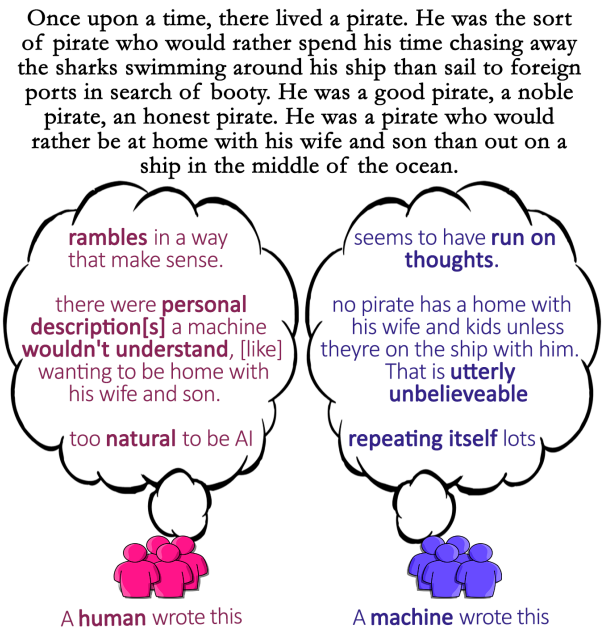


Figure 6.1: Excerpts from human evaluators' explanations for why they believe a GPT3-generated story (also excerpted) was written by a human (left) or a machine (right). The evaluators point to a wide range of text attributes to make their decisions, sometimes using the same aspect of the text to come to opposite conclusions.

is evaluated, evaluators implicitly compare the generated text to their knowledge of language and norms within specific domains.

Evaluators are often asked to assess a text holistically, e.g., based on its overall quality, naturalness, or humanlikeness [van der Lee et al., 2021; Howcroft et al., 2020], where the exact evaluation criteria is left to the discretion of the evaluator. Though other evaluations are broken down along specific dimensions of text quality (e.g., grammaticality, coherence, etc.), Novikova et al. [2017, 2018] and Callison-Burch et al. [2007] found that these dimensions are often correlated and may be conflated in some evaluation settings. This is concerning because, as NLG models improve, evaluators are asked to read longer passages of text conditioned on large amounts of context. In these cases, fluency-related aspects of quality (i.e., the ones that don't require careful reading of the context and meaning of the passage) are the easiest to assess, particularly in small-batch evaluations with non-expert evaluators where speed is incentivized. This poses a challenge when collecting human evaluations for state-of-the-art language models, as errors are often content-based (e.g., factual inaccuracies or inconsistencies with the context) rather than fluency-based [Brown et al., 2020], so a superficial read may not be sufficient to catch model errors. For accurate assessments of generated

text, we need human evaluations that are designed to encourage a sufficiently careful reading of the text to examine these subtler aspects of text quality.

We asked non-expert evaluators to assess the humanlikeness (operationalized as how believably human an evaluator finds a text) of text generated by current NLG models (GPT2 and GPT3) to test what current human evaluation practices can reveal about the models' quality (§6.2). We found that evaluators were unable to distinguish between GPT3- and human-authored text across story, news, and recipe domains. However, when we categorized the aspects of text the evaluators used to make their judgments, we found they primarily focused on the grammar, spelling, and style of the text. The evaluators' responses also indicated that they underestimated the quality of text current models are capable of generating (as seen in Figure 6.1). To our knowledge, this work is the first to evaluate human evaluations of GPT3-generated text across multiple domains.

We then looked at three different evaluator training methods—providing detailed instructions, annotated examples, and human-machine paired examples—to test whether we could improve evaluators' accuracy (§6.3). While we found including examples in the task increased the set of texts evaluators thought could be machine-generated and increased their focus on textual content, no training method significantly increased evaluators' performance consistently across domains.

Based on our results (discussed in §6.4), we recommend moving away from small-batch evaluations with little training when collecting human evaluations of NLG models (§6.5). We also encourage practitioners to consider alternative evaluation frameworks that capture the usefulness of generated text in downstream settings rather than its humanlikeness.

6.2 How well can untrained evaluators identify machine-generated text?

In our first study, we ask how well untrained evaluators can distinguish between human- and machine-generated text. This task format, inspired by the Turing [1950] Test, is used to compare the quality of machine-generated text to human-authored text and, as models' fluency improves, to analyze NLG models' ability to "fool" readers [Ippolito et al., 2020; Brown et al., 2020].

By asking evaluators to assess the humanlikeness of the text with only minimal instructions (see Figure 6.2), we observe how well untrained evaluators can detect state-of-the-art machine-generated text and which

attributes evaluators focus on and think are important for detecting machine-generated text.

6.2.1 The Task

We gave evaluators 5 text passages, some of which were written by people and some generated by a model. We asked them to rate the text on a 4-point scale [Ippolito et al., 2020]:

1. Definitely human-written
2. Possibly human-written
3. Possibly machine-generated
4. Definitely machine-generated

If they selected option 1, we asked them: "Why did you select this rating?" Otherwise, they were asked, "What would you change to make it seem more human-like?" The interface is shown in Figure 6.2.

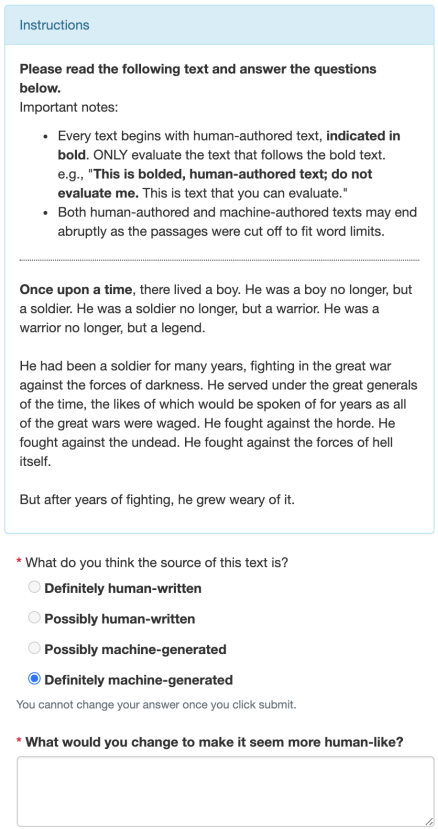


Figure 6.2: The task interface (story domain)

6.2.2 Data

We considered human- and machine-generated text in three different domains: stories, news articles, and recipes. In all three cases, we collected 50 human-authored texts in English and generated 50 texts from both the 175B parameter GPT3 model (also known as Davinci; Brown et al., 2020)[1] and GPT2-XL [Radford et al., 2019].[2] Evaluators were assigned to one domain and one model; the texts read by any given evaluator included some human-authored texts and some texts generated by their assigned model. We only considered texts 100 words or longer, and after reaching 100 words, all texts were truncated at the end of the next sentence.[3]

To generate text, we used the "three-shot" setting described in Brown et al. [2020], conditioning the text on three additional samples of in-domain, human-authored text, which we refer to as the *priming texts* (all priming texts are in the supplementary materials and at `ark.cs.washington.edu/human_evals_ACL21`). While this setting is not typically how GPT2 is used in practice, we held this approach constant to directly compare how model quality changes evaluators' ability to distinguish between texts. For each domain, each generated text was conditioned on the same set of priming texts. The texts were delimited with an $\langle$EOS$\rangle$ token and generated using the default GPT3 generation settings (i.e., sampling with temperature $= 0.7$).

Stories

The human-authored texts came from the Reddit WritingPrompts dataset [Fan et al., 2018].[4] We collected all the stories that began with *Once upon a time* (255 stories total) and randomly chose 50 human-authored stories from this set. For the machine-generated text, we conditioned the models on the three priming texts and on the phrase *Once upon a time*. We removed generated stories that directly copied a priming text (with $> 80\%$ overlap) and regenerated those texts (9 instances with GPT2, 2 with GPT3).

This is the most open-ended of the three domains, as the story's content is virtually unrestricted, and the only creative domain. It is also the noisiest of the human-authored datasets, as the stories were originally collected from social media comments with no quality-based filtering.

[1] `beta.openai.com/`
[2] `huggingface.co/gpt2-xl`
[3] Using NLTK; `www.nltk.org/`
[4] `github.com/pytorch/fairseq/tree/master/examples/stories`

News Articles

We collected 2,111 recent local news articles from 15 different newspapers using Newspaper3k[5] (details in Appendix C.1). After filtering out articles under 100 words, we manually filtered out articles that weren't local news or that referenced the coronavirus pandemic. We randomly chose 50 articles to use as our human-authored news articles and another 50 to use as prompts for our generation models. We conditioned each generated text on the headline and first sentence from the prompt articles, along with the three priming texts.

Because the title and the first sentence of a news article often summarize its contents, the generated content must adhere to the topics they introduce. By using local, recent news, we also limit the models' ability to copy from their training data. The models seemed to have the most trouble with this dataset structurally, e.g., generating new headlines without ending the current article or outputting invalid end-of-file tags.

Recipes

We collected 50 human-authored recipes from the RecipeNLG dataset [Bień et al., 2020], which contains 2,231,142 recipes scraped from the web. We randomly chose an additional 50 recipes and used their titles and ingredient lists as prompts, appending them to the end of the priming texts.

This is the most closed of the three domains, as the recipe must incorporate the listed ingredients and result in the dish described by the title. Recipes are typically written in clear commands, leaving little room for surprising or unexpected text.

6.2.3 Participants

We used Amazon Mechanical Turk (AMT) to collect the text evaluations with non-expert evaluators, commonly used in NLG evaluations [van der Lee et al., 2019]. To have adequate power in our analyses (based on a power analysis with $\beta = 0.8$; Card et al., 2020), we had 130 different evaluators for each of the 6 task settings (3 domains $\times$ 2 models). Each participant evaluated 5 texts each, giving us a total of 780 participants and 3,900 text evaluations.

We paid evaluators US$1.25 for completing the task. Following common best practice on AMT Berinsky

[5] `github.com/codelucas/newspaper`

Model	Overall Acc.	Domain	Acc.	F_1	Prec.	Recall	Kripp. α	% human	% confident
GPT2	*0.58	Stories	*0.62	0.60	0.64	0.56	0.10	55.23	52.00
		News	*0.57	0.52	0.60	0.47	0.09	60.46	51.38
		Recipes	0.55	0.48	0.59	0.40	0.03	65.08	50.31
GPT3	0.50	Stories	0.48	0.40	0.47	0.36	0.03	62.15	47.69
		News	0.51	0.44	0.54	0.37	0.05	65.54	52.46
		Recipes	0.50	0.41	0.50	0.34	0.00	66.15	50.62

Table 6.1: §6.2 results, broken down by domain and model, along with the F_1, precision, and recall at identifying machine-generated text, Krippendorff's α, % human-written guesses, and % confident guesses (i.e., *Definitely* machine- or human-authored). * indicates the accuracies significantly better than random (two-sided t-test, for Bonferroni-corrected $p < 0.00333$).

et al. [2012], evaluators had to have over a 95% acceptance rate, be in the United States, and have completed over 1,000 HITs (AMT tasks). We excluded evaluators' work if their explanations were directly copied text from the task, did not match their responses, did not follow the instructions, or were short, vague, or otherwise uninterpretable. Across experiments, 445 participants (18.6%) were rejected and not included in the §6.2 results (780 approved participants) and §6.3 results (1,170 approved participants).

6.2.4 Results

Overall, evaluators choosing between human and GPT2-generated text correctly identified the author of the text 57.9% of the time,[6] but the evaluators choosing between human- and GPT3-generated text only guessed correctly 49.9% of the time (Table 6.1), compared to 50% random chance. While the accuracy of classifying GPT2- vs. human-authored text is significantly[7] different from chance, evaluators' accuracy distinguishing GPT3- and human-authored text is not.[8] This remains the case regardless of text domain; we failed to find any evidence that evaluators' accuracy on any one domain for GPT3 differs from the overall GPT3 accuracy of $\approx 50\%$.[9] The story texts saw the biggest drop in evaluator accuracy from GPT2 to GPT3 (62% to 48%, Cohen's $d = 0.57$). The distribution of evaluators' scores are shown in Appendix C.2.

In Table 6.1, we see other statistics worsen as well between GPT2 and GPT3: how well evaluators identified the machine-generated text (F_1, precision, and recall), evaluators' agreement (Krippendorff's α,

[6] Unless otherwise noted, all analyses binned the responses into 2 categories (*human* and *machine*).
[7] $t_{388} = 6.58$, $p < 0.0001$
[8] $t_{388} = -0.09$, $p = 0.93$
[9] ANOVA with $F_{2,390} = 0.78$, $p = 0.46$

a measure of annotator agreement that corrects for the probability of random agreement), and the percent of guesses that the text was human-written (% human). Given that the texts are equally likely to be human- and machine-written, there are disproportionately many *human* guesses, making up two thirds of the responses in the GPT3 experiments. Despite the significantly lower scores, evaluators' confidence (the percent of *Definitely* responses) remains fairly constant across conditions.

6.2.5 Analysis

Taken on its own, the evaluators' difficulty identifying GPT3-generated text compared to GPT2 points to the improvement of new NLG models. However, it also points to concerns about extending current human evaluation methodologies to state-of-the-art text generation. In particular, the evaluators' explanations reveal underlying confusion and misconceptions about state-of-the-art NLG.

To better understand what untrained evaluators focused on in the text to make their decisions, the authors annotated 150 random responses from the evaluators who distinguished between human- and GPT3-generated text (see Appendix C.3 for annotation details). We divided the text annotation labels into three categories: *form*, *content*, and *machine capabilities*. *Form* qualities focus on the format, style, and tone of the text, while *content* focuses on the text's meaning. We also coded for comments that explicitly referenced people's perceptions of what types of language machines are capable (or incapable) of generating (*machine capabilities*).

We found nearly twice as many comments about the form of the text than the content (*form*: 47% of labels, *content*: 25%). Evaluators in our sample focused most on the spelling, grammar, or punctuation of the texts (45 out of 150 comments) and the style or tone of the text (24 out of 150 comments). However, these dimensions of text are unlikely to be helpful in identifying text generated by current models, considering that GPT3 has already been shown to generate fluent text and to adapt easily to new generation domains [Brown et al., 2020].

We also found that the reasons evaluators gave for their answers often contradicted each other. The formality of the text, spelling and grammar errors, and clarity were all cited to justify both *human* and *machine* judgments. This was also reflected in the low agreement scores between evaluators, with Krippendorff's $\alpha \approx 0$ across domains.

Evaluators' expectations about what NLG models are capable of ranged from thinking their text is already indistinguishable from human-authored text ("I have no idea if a human wrote anything these days. No idea at all.") to doubting machines' ability to use basic language ("Usually AI has terrible grammer [sic] and messes up."). But overall we found most evaluators' beliefs about generated language underestimated or misunderstood current NLG models, as seen in Appendix C.4.

6.3 Can we train evaluators to better identify machine-generated text?

Given evaluators' inability to distinguish GPT3- and human-authored text and their inconsistent reasoning for their decisions, we investigated whether there were simple ways of improving evaluators' ability to spot attributes of GPT3-generated text. Inspired by crowdsourcing research on guiding workers on writing or other subjective tasks Kim et al. [2017]; Mitra et al. [2015], we tested three *lightweight* evaluator-training methods to see if we could improve people's ability to identify machine-generated text while maintaining the short, low-cost nature of the evaluations.

6.3.1 Evaluator Training Methods

We considered 3 evaluator trainings that can be added to the beginning of a human evaluation task, at most requiring only 3 extra samples of human- and machine-generated text. To test the effectiveness of each type of training, we re-ran the experiments from §6.2, but this time, we prepended one of three evaluator-training methods to the evaluation task: an *instruction-based* training, an *example-based* training, and a *comparison-based* training. Screenshots of the training interfaces are in Appendix C.6; the full set of training materials are in the supplementary materials and at `ark.cs.washington.edu/human_evals_ACL21`.

Other than the training, the task setup was identical to the GPT3-based tasks in §6.2. We again ran the task on Amazon Mechanical Turk across three domains (stories, news, and recipes), using the same texts. As each individual participant was only permitted to complete one set of evaluations, the set of evaluators who received these trainings was completely disjoint from the set of evaluators from our first study. The participants were subject to the same restrictions described in §6.2.3 and excluded according the same criteria; we did not use the trainings to filter out evaluators. For each domain and training method pair, we had 130 unique evaluators complete the task, giving us 5,850 text annotations from 1,170 evaluators.

Training with Instructions

To give evaluators a better sense of which parts of the text to pay attention to, we extended the original task instructions to include dimensions of the text that could be helpful for identifying machine-generated text (repetition and factuality) and ones that could be misleading (grammar, spelling, and style). We chose these dimensions based on previous work [Ippolito et al., 2020] and evaluators' comments in a pilot study (see Appendix C.5).

The Instructions training was the simplest of our 3 evaluator training methods. It was general enough to be applied across the 3 domains but provided little information about the quality and domain of text the evaluator would be rating. It did not increase the cost of collecting evaluations (US$1.25 per HIT) because it does not require any extra work on the part of the evaluator, though this also made it the easiest training to ignore. The instruction-based training is the most prescriptive of the training methods, as the researcher has to choose the dimensions they want the evaluators to focus on.

Training with Examples

Our Examples training consisted of 3 practice rounds of the actual task: given a text, guess if it is machine- or human-authored. We collected 3 additional texts in the same manner described in §6.2.2 and wrote a short explanation of which aspects of the text hinted at its source. After an evaluator makes their guess, the correct answer and explanation are shown. Each domain had its own set of examples and explanations.

By showing examples, this training helps set the evaluators' expectations about the quality of the human- and machine-generated text. We paid evaluators more for completing this task (US$1.75 per HIT) to compensate for the extra texts they needed to read. As with the instruction-based training, while pointing out specific text dimensions can help evaluators focus on important features, it may also restrict their search space.

Training with Comparison

In the Comparison training, we took the example passages from the Examples training and paired them with a text from the opposite source (machine or human) that began with the same prompt. We asked evaluators to guess which of the two texts was the machine-generated one. We then provided the correct answer to the

evaluator, along with the same explanations used in the Examples training.

This training allows evaluators to directly compare human and machine texts written from the same prompt. It is also the most expensive training, as it required evaluators to read three more passages than the Examples training; we paid evaluators US$2.25 per HIT.

Training	Overall Acc.	Domain	Acc.	F_1	Prec.	Recall	Kripp. α	% human	% confident
None	0.50	Stories	0.48	0.40	0.47	0.36	0.03	62.15	47.69
		News	0.51	0.44	0.54	0.37	0.05	65.54	52.46
		Recipes	0.50	0.41	0.50	0.34	0.00	66.15	50.62
Instructions	0.52	Stories	0.50	0.45	0.49	0.42	0.11	57.69	45.54
		News	0.56	0.48	0.55	0.43	0.05	62.77	52.15
		Recipes	0.50	0.41	0.52	0.33	0.07	67.69	49.85
Examples	*0.55	Stories	0.57	0.55	0.58	0.53	0.06	53.69	64.31
		News	0.53	0.48	0.52	0.45	0.05	58.00	65.69
		Recipes	0.56	0.56	0.61	0.51	0.06	55.23	64.00
Comparison	0.53	Stories	0.56	0.56	0.55	0.57	0.07	48.46	56.62
		News	0.52	0.51	0.53	0.48	0.08	53.85	50.31
		Recipes	0.51	0.49	0.52	0.46	0.06	54.31	53.54

Table 6.2: §6.3 results, broken down by domain and training method, along with the F_1, precision, and recall at identifying machine-generated text, Krippendorff's α, % human-written guesses, and % confident guesses (i.e., *Definitely* machine- or human-authored). "None" training refers to the GPT3 results from §6.2. * indicates accuracies significantly better than None (no training; two-sided t-test, for Bonferroni-corrected $p < 0.00333$).

6.3.2 Results

We found that while all 3 training methods improved evaluators' accuracy at identifying machine- vs. human-authored text over the no-training accuracy, the Examples training was the only one that showed significant improvement (see Table 6.2).[10]

Breaking down the results by domain, however, we find the Examples accuracy did not significantly increase over the no-training accuracy when considering any of the three domains individually. Even so, the significant difference in overall performance is mainly contributed by the story domain; when comparing evaluators' performance with no training to its Examples training counterpart, we see a change of 0.019 and

[10]Tukey's HSD adjusted $p < 0.003$ for distinguishing between the Examples training and no training, $d = 0.25$

Training	Form	Content	Machine capabilities
None	47.1	24.6	28.3
Examples	32.5	50.0	17.5

Table 6.3: % of annotation labels that reference the text's form and content and the evaluators' perception of machines' capabilities

0.062 mean accuracy in the news and recipe domains, respectively, versus 0.086 on the story domain. This is perhaps due to the examples helping override the preconception that machines cannot generate "creative" text.

Across all 3 domains, the Examples and Comparison trainings produced the highest recall and F_1 scores for evaluators' judgments and decreased the percentage of texts they guessed were human-written, which indicate that evaluators were willing to consider a broader set of texts to be machine-generated than the evaluators in §6.2. However, despite the trainings and the increased proportion of confident responses, evaluator agreement remained low across domain and training settings ($\alpha \leq 0.11$), and higher agreement did not correspond to higher accuracy.

6.3.3 Analysis

We again annotated 150 comments along the dimensions listed in Appendix C.3, divided into *form*, *content*, and *machine capabilities* categories, this time from evaluators who received the best-performing Examples training. As shown in Table 6.3, we found that the proportion of *form* comments dropped in the sample of evaluators who went through the Examples training, while the proportion of *content* comments doubled. We also saw a drop in the number of comments mentioning evaluators' expectations of machine-generated text. While this change in focus doesn't necessarily correspond to correct judgments, *content* reasons are more in-line with current NLG model capabilities Brown et al. [2020].

6.4 Discussion

Overall, none of our three training methods significantly improved evaluators' ability to detect machine-generated text reliably across text domains while still maintaining the small-batch nature of Amazon Mechanical Turk. This speaks to the improving quality of NLG models, but we also found that untrained

evaluators mainly focused on the format of the text, deciding if it was human or machine-generated based on whether the text was grammatically or stylistically correct. This, combined with the high percentage of *human* guesses, the low recall scores for the *machine* guesses, and the evaluators' comments on their expectations of NLG models, indicates a systematic underestimation by the evaluators of the quality of machine-generated text. Evaluators who were trained with examples had higher expectations of machine-generated text and focused more on the text's content; however, the training was not sufficient to significantly raise evaluators' scores across all three domains.

Many of the explanations given by evaluators included references to the text that reflected human attributes or intent that they suspected machines could not generate (e.g., "personal description a machine wouldn't understand, [like a pirate] wanting to be home with his wife and son" from Figure 6.1 and the examples in Appendix C.4). However, current NLG models are capable of generating text with at least superficial reference to human attributes or intent, as seen in the generated story in Figure 6.1. This assumption that machines can't generate text with these aspects of humanlikeness led many evaluators astray, and we suspect it is one cause of the low accuracy we found.

Crowdsourcing studies dealing only with human-authored texts often include extensive training, quality checks, or coordination Kittur and Kraut [2008]; Kim et al. [2017]; Bernstein et al. [2010]. NLG evaluations usually forego such structures, based, we suspect, on the assumption that evaluating machine-generated text requires only fluency in the language the text is generated in. Our results suggest otherwise. Evaluators often mistook machine-generated text as human, citing superficial textual features that machine generation has surpassed Brown et al. [2020]. One potential remedy for this is to focus evaluator training on debunking this misconception. We did see evidence that the increase in accuracy we saw with our Examples training was associated with fewer explanations mistakenly referencing machine capabilities, even though the training did not specifically focus on this.

6.5 Recommendations

Based on our findings, if NLG researchers must run human evaluations as small-batch evaluations on Amazon Mechanical Turk or similar platforms, we recommend they train evaluators with examples. This will help calibrate the evaluators' expectations of generated text and indicate the careful reading they may need

to do to properly assess the text's quality. Our experiments also indicate the importance of confirming with evaluators why they have made the decisions they have, as the criteria they might implicitly be evaluating may be mismatched with researchers' intended criteria. However, other evaluation setups may be more successful on Amazon Mechanical Turk, such as long-term evaluations with qualified evaluators who have gone through an extended training (like those in Kittur and Kraut, 2008; Zellers et al., 2019a) or third-party evaluator quality tools (e.g., Positly, used by Brown et al., 2020).

However, given the increasing length of text NLG models can handle and the careful reading needed to detect many errors in generated text, we encourage NLG researchers to move away from standalone, intrinsic human evaluation tasks. We found that, by default, our evaluators in this evaluation setting were most likely to focus on surface-level, fluency-related aspects of quality. We join past work [Belz and Reiter, 2006; van der Lee et al., 2021] in recommending a move towards evaluation settings where evaluators are better motivated to carefully consider the content and usefulness of generated text. For example, TuringAdvice [Zellers et al., 2021] asks evaluators to rate NLG models by their ability to generate helpful advice, and RoFT [Dugan et al., 2020] engages evaluators through a guessing game to determine the boundary between human- and machine-generated text. Other evaluation methods ask the evaluators to directly interact with the generated text; for example, Choose Your Own Adventure [Clark and Smith, 2021] and Storium [Akoury et al., 2020] evaluate story generation models by having people write stories with the help of generated text.[11] We see that GPT3 can successfully mimic human-authored text across several domains, renewing the importance of evaluations that push beyond surface-level notions of quality and consider whether a text is helpful in a downstream setting or has attributes that people would want from machine-generated text.

Finally, given the mixed effect we found different trainings can have on evaluators' performance and the lack of human evaluation details typically presented in NLG papers [van der Lee et al., 2019; Howcroft et al., 2020], we encourage NLG researchers to include details of any instructions and training they gave evaluators in their publications. This, along with efforts to standardize human evaluation design [Belz et al., 2020; Howcroft et al., 2020] and deployment [Khashabi et al., 2021; Gehrmann et al., 2021], will support future development of evaluator training procedures and the comparison of human evaluation results in

[11]Note that we initially tried a fourth training condition along these lines, where we asked evaluators to directly interact with the generated text by rewriting it be more humanlike. We found we were unable to successfully recruit evaluators to complete this task. The rate of retention was less than 30%, and the rejection rate was over 50%. We found AMT was not a good platform for this type of task, at least not for the format and the price point we explored in this work.

future NLG evaluation work.

6.6 Related Work

A subfield of NLG analyzes the role of human evaluations, including discussions of the tradeoffs of human and automatic evaluations [Belz and Reiter, 2006; Hashimoto et al., 2019]. There are critiques and recommendations for different aspects of human evaluations, like the evaluation design [Novikova et al., 2018; Santhanam and Shaikh, 2019], question framing [Schoch et al., 2020], and evaluation measures like agreement [Amidei et al., 2018], as well as analyses of past NLG papers' human evaluations [van der Lee et al., 2021; Howcroft et al., 2020]. Additionally, crowdsourcing literature has work on effectively using platforms like Amazon Mechanical Turk [e.g., Daniel et al., 2018; Oppenheimer et al., 2009; Weld et al., 2014; Mitra et al., 2015]. In this work, we focus on the role evaluator training can play for producing better accuracy at distinguishing human- and machine-generated text, though other quality control methods are worth exploring.

Previous work has asked evaluators to distinguish between human- and machine-authored text; for example, Ippolito et al. [2020] found that trained evaluators were able to detect open-ended GPT2-L-generated text 71.4% of the time and Brown et al. [2020] found evaluators could guess GPT3-davinci-generated news articles' source with 52% accuracy, though these results are not directly comparable to ours due to differences in the evaluation setup, data, and participants.

Finally, our findings that untrained evaluators are not well equipped to detect machine-generated text point to the importance of researching the safe deployment of NLG systems. Gehrmann et al. [2019] proposed visualization techniques to help readers detect generated text, and work like Zellers et al. [2019b], Ippolito et al. [2020], and Uchendu et al. [2020] investigated large language models' ability to detect generated text.

6.7 Conclusion

We found that untrained evaluators were unable to distinguish between human- and GPT3-generated text from three domains. However, we also found that the evaluators focused on surface-level text qualities to

make these decisions and underestimated current NLG models' capabilities. We experimented with three methods for training evaluators, and while example-based trainings led to increases in recall and the amount of content-based evaluations, they did not lead to significant improvements in accuracy across all domains. Given that evaluators struggled to distinguish between human- and machine-generated text in this setting, we should shift how we think about collecting human evaluations for current NLG models.

Chapter 7

Conclusion

In this dissertation, we've discussed the role that NLG models can play in a human-machine writing collaboration. In Chapter 2, we saw two "machine-in-the-loop" collaborative writing systems (one for stories and one for slogans) and discussed directions for improving NLG models for human-machine collaboration. Chapter 3 explored one such direction: incorporating entity information in the generation model as an additional form of context. Finally, Chapter 4 discussed the evaluation of NLG models for collaborative writing systems and proposed a method for collecting pairwise judgments of models' generated text.

We then discussed the use of automatic metrics in NLG evaluations and challenges to evaluation posed by long generated documents. We introduced "Sentence Mover's Similarity" metrics to evaluate multi-sentence text by measuring documents' similarity based on the distance between their sentence embeddings. These metrics can also be used as a reward when generating text, and we found this approach outperformed baseline generation methods according to both human and machine evaluations.

We concluded with an examination of human evaluations in NLG. We found that non-expert judges were unable to accurately detect GPT3-generated text across three domains (stories, news articles, and recipes). Moreover, the reasons they gave for their judgments were more focused on the form of the text, rather than its content. None of the three training methods we experimented with were able to significantly improve the evaluator accuracy, leading us to recommend directions for collecting human evaluations for state-of-the-art NLG models.

Future work

This work begins to answer the challenges described at the beginning of this dissertation, but there are many interesting research directions remaining, particularly in the face of the rapid improvements currently being made in NLG. Some examples of promising research areas addressing the challenges described in this dissertation follow.

Flexible writing support

While in Chapter 2 we saw that some writers appreciated the turn-taking approach to collaborative writing, others prefer a more flexible writing process. One way to increase writers' control over the written piece is to let them request suggestions when they need them. This approach has been used in systems like Storium [Akoury et al., 2020] and Creative Help [Roemmele and Gordon, 2015], but added flexibility for the writer often presents challenges for system-level model evaluation, as two writers may have completely different types of interactions. Paired model suggestions (as proposed in Chapter 4) would alleviate some of the evaluation challenges of on-demand writing systems because the models are generating text in the same context and are evaluated by the same writer. Removing the turn-taking structure would also allow for nonlinear writing and the use of both backward- and forward-context, as studied in text infilling work like Donahue et al. [2020]; Ippolito et al. [2019].

Automatic evaluation of long generated text

In Chapter 5, we showed how texts can be evaluated at the sentence level rather than the word level. As the length of text that models can generate continues to grow [Dai et al., 2019; Zaheer et al., 2020; Beltagy et al., 2020], higher level representations or hierarchical evaluation methods may be better suited to capture the similarity of long generated texts. Discourse structure and the ordering and location of information may also be useful for deciding when (and when not) to reward similarity between two long passages. Long generated texts will also benefit from multi-dimensional analysis and evaluation.

Interactive text evaluation

As we saw in Chapter 6, quick evaluator trainings are not sufficient for improving human evaluators' ability to detect machine-generated text. Given the evaluators' preference for judging text based on its form rather than its content and the increasing length and fluency of generated passages, there is a need for human evaluations that motivate evaluators to carefully read and consider generated text. Interactive evaluation methods (like Choose Your Own Adventure from Chapter 4) encourage evaluators to directly interact with the text, e.g., via editing. In Choose Your Own Adventure (Chapter 4), this is combined with asking evaluators to use the generated text for an end task (i.e., story writing) to additionally motivate them to consider the text's meaning. Another direction for motivating evaluators to carefully read text is through text evaluation games, like RoFT [Dugan et al., 2020] and Spot the Bot [Deriu et al., 2020].

Bibliography

Nader Akoury, Shufan Wang, Josh Whiting, Stephen Hood, Nanyun Peng, and Mohit Iyyer. 2020. STORIUM: A Dataset and Evaluation Platform for Machine-in-the-Loop Story Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6470–6484, Online. Association for Computational Linguistics.

James F. Allen, Curry I. Guinn, and Eric Horvitz. 1999. Mixed-initiative interaction. *IEEE Intelligent Systems*, 14(5):14–23.

Mohammed Alshahrani, Spyridon Samothrakis, and Maria Fasli. 2017. Word mover's distance for affect detection. *2017 International Conference on the Frontiers and Advances in Data Science*.

Jacopo Amidei, Paul Piwek, and Alistair Willis. 2018. Rethinking the agreement in human evaluation tasks. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3318–3329, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Kubilay Atasu, Thomas P. Parnell, Celestine Dünner, Manolis Sifalakis, Haralampos Pozidis, Vasileios Vasileiadis, Michail Vlachos, Cesar Berrospi, and Abdel Labbi. 2017. Linear-complexity relaxed word mover's distance with GPU acceleration. In *IEEE International Conference on Big Data*.

Tal August, Maarten Sap, Elizabeth Clark, Katharina Reinecke, and Noah A. Smith. 2020. Exploring the effect of author and reader identity in online story writing: the STORIESINTHEWILD corpus. In *Proceedings of the First Joint Workshop on Narrative Understanding, Storylines, and Events*, pages 46–54, Online. Association for Computational Linguistics.

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

Georgios Balikas, Charlotte Laclau, Ievgen Redko, and Massih-Reza Amini. 2018. Cross-lingual document retrieval using regularized Wasserstein distance. *CoRR*, abs/1805.04437.

Paul Baltescu and Phil Blunsom. 2015. Pragmatic neural language modelling in machine translation. In *NAACL*.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *IEEvaluation@ACL*.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *ArXiv*, abs/2004.05150.

Anja Belz and Ehud Reiter. 2006. Comparing automatic and human evaluation of NLG systems. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, Trento, Italy. Association for Computational Linguistics.

Anya Belz, Simon Mille, and David M. Howcroft. 2020. Disentangling the properties of human evaluation methods: A classification system to support comparability, meta-evaluation and reproducibility testing. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 183–194, Dublin, Ireland. Association for Computational Linguistics.

Adam J. Berinsky, Gregory A. Huber, and Gabriel S. Lenz. 2012. Evaluating online labor markets for experimental research: Amazon.com's Mechanical Turk. In *Political Analysis*, volume 20, pages 351–368. Cambridge University Press.

Michael Bernstein, Greg Little, Robert Miller, Björn Hartmann, Mark Ackerman, David Karger, David Crowell, and Katrina Panovich. 2010. Soylent: A word processor with a crowd inside. In *UIST 2010 - 23rd ACM Symposium on User Interface Software and Technology*, volume 58, pages 313–322.

Michał Bień, Michał Gilski, Martyna Maciejewska, Wojciech Taisner, Dawid Wisniewski, and Agnieszka Lawrynowicz. 2020. RecipeNLG: A cooking recipes dataset for semi-structured text generation. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 22–28, Dublin, Ireland. Association for Computational Linguistics.

Margaret A. Boden. 2004. *The Creative Mind: Myths and Mechanisms*. Routledge, London; New York.

Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi. 2018. Simulating Action Dynamics with Neural Process Networks. In *ICLR*.

Steve Branson, Catherine Wah, Florian Schroff, Boris Babenko, Peter Welinder, Pietro Perona, and Serge Belongie. 2010. Visual recognition with humans in the loop. In *Proceedings of ECCV*.

Alastair Brotchie and Mel Gooding, editors. 1995. *A Book of Surrealist Games: Including the Little Surrealist Dictionary*. Shambhala Redstone Editions, distributed in the United States by Random House, Boston.

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.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Marc Brysbaert, Amy Beth Warriner, and Victor Kuperman. 2014. Concreteness ratings for 40 thousand generally known English word lemmas. *Behavior Research Methods*, 46:904–911.

Chris Callison-Burch, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. 2007. (meta-) evaluation of machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 136–158, Prague, Czech Republic. Association for Computational Linguistics.

Dallas Card, Peter Henderson, Urvashi Khandelwal, Robin Jia, Kyle Mahowald, and Dan Jurafsky. 2020. With little power comes great responsibility. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9263–9274, Online. Association for Computational Linguistics.

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.

Asli Celikyilmaz, Antoine Bosselut, Xiadong He, and Yejin Choi. 2018. Deep communicating agents for abstractive summarization. In *NAACL*.

Arun Tejasvi Chaganty, Stephen Mussmann, and Percy Liang. 2018. The price of debiasing automatic metrics in natural language evaluation. In *ACL*.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, and Phillipp Koehn. 2013. One billion word benchmark for measuring progress in statistical language modeling. In *INTERSPEECH*.

Liqun Chen, Yizhe Zhang, Ruiyi Zhang, Chenyang Tao, Zhe Gan, Haichao Zhang, Bai Li, Dinghan Shen, Changyou Chen, and Lawrence Carin. 2019. Improving sequence-to-sequence learning via optimal transport. In *ICLR*.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *ACL*.

Justin Cheng and Michael S. Bernstein. 2015. Flock: hybrid crowd-machine learning classifiers. In *Proceedings of CSCW*.

Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A. Smith. 2021. All that's 'human' is not gold: Evaluating human evaluation of generated text. In *ACL-IJCNLP*.

Elizabeth Clark, Asli Celikyilmaz, and Noah A. Smith. 2019. Sentence mover's similarity: Automatic

evaluation for multi-sentence texts. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2748–2760, Florence, Italy. Association for Computational Linguistics.

Elizabeth Clark, Yangfeng Ji, and Noah A. Smith. 2018a. Neural text generation in stories using entity representations as context. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2250–2260, New Orleans, Louisiana. Association for Computational Linguistics.

Elizabeth Clark, Anne Spencer Ross, Chenhao Tan, Yangfeng Ji, and Noah A. Smith. 2018b. Creative writing with a machine in the loop: Case studies on slogans and stories. In *Proceedings of the 23rd International Conference on Intelligent User Interfaces*, pages 329–340, Tokyo, Japan. Association for Computing Machinery.

Elizabeth Clark and Noah A. Smith. 2021. Choose your own adventure: Paired suggestions in collaborative writing for evaluating story generation models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3566–3575, Online. Association for Computational Linguistics.

Kevin Clark and Christopher D. Manning. 2016a. Deep reinforcement learning for mention-ranking coreference models. In *EMNLP*.

Kevin Clark and Christopher D. Manning. 2016b. Improving coreference resolution by learning entity-level distributed representations. In *ACL*.

Alexis Conneau and Douwe Kiela. 2018. SentEval: An evaluation toolkit for universal sentence representations. In *LREC*.

Leo Cullum. 2005. Doctor talking to patient in his office as a man. *The New Yorker*, 81(39).

Andrew M. Dai, Christopher Olah, and Quoc V. Le. 2015. Document embedding with paragraph vectors. In *NeurIPS Deep Learning Workshop*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th*

Annual Meeting of the Association for Computational Linguistics, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Robert Dale and Ehud Reiter. 1995. Computational interpretations of the gricean maxims in the generation of referring expressions. *Cognitive science*, 19(2):233–263.

Florian Daniel, Pavel Kucherbaev, Cinzia Cappiello, Boualem Benatallah, and Mohammad Allahbakhsh. 2018. Quality control in crowdsourcing: A survey of quality attributes, assessment techniques, and assurance actions. In *ACM Computing Surveys*, volume 51. Association for Computing Machinery.

Jan Deriu, Don Tuggener, Pius von Däniken, Jon Ander Campos, Alvaro Rodrigo, Thiziri Belkacem, Aitor Soroa, Eneko Agirre, and Mark Cieliebak. 2020. Spot the bot: A robust and efficient framework for the evaluation of conversational dialogue systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3971–3984, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Second International Conference on Human Language Technology Research*.

Chris Donahue, Mina Lee, and Percy Liang. 2020. Enabling language models to fill in the blanks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2492–2501, Online. Association for Computational Linguistics.

Liza Donnelly. 2014. A man and woman look at three very large birds. *The New Yorker*, 90(22).

Liam Dugan, Daphne Ippolito, Arun Kirubarajan, and Chris Callison-Burch. 2020. RoFT: A tool for evaluating human detection of machine-generated text. In *Proceedings of the 2020 Conference on Empirical*

Methods in Natural Language Processing: System Demonstrations, pages 189–196, Online. Association for Computational Linguistics.

Ondrej Dusek, Jekaterina Novikova, and Verena Rieser. 2019. Evaluating the state-of-the-art of end-to-end natural language generation: The E2E NLG challenge. In *Computational Linguistics*.

Sergey Edunov, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Classical structured prediction losses for sequence to sequence learning. In *NAACL-HLT*.

Desmond Elliott and Frank Keller. 2014. Comparing automatic evaluation measures for image description. In *ACL*.

David K. Elson and Kathleen McKeown. 2009. A tool for deep semantic encoding of narrative texts. In *ACL-IJCNLP*.

Kawin Ethayarajh and Dan Jurafsky. 2020. Utility is in the eye of the user: A critique of NLP leaderboards. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4846–4853, Online. Association for Computational Linguistics.

Jerry Alan Fails and Dan R. Olsen Jr. 2003. Interactive machine learning. In *Proceedings of IUI*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Linda Flower and John R. Hayes. 1981. A cognitive process theory of writing. *College Composition and Communication*, 32(4):365–387.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *ACL workshop for NLP Open Source Software*.

Monica J. Garfield. 2008. Creativity support systems. In *Handbook on Decision Support Systems 2*, pages 745–758. Springer, Berlin, Heidelberg.

Lorenzo Gatti, Gözde Özbal, Marco Guerini, Oliviero Stock, and Carlo Strapparava. 2016. Heady-Lines: a creative generator of newspaper headlines. In *Companion Publication of the 21st International Conference on Intelligent User Interfaces*, pages 79–83.

Sebastian Gehrmann, Tosin Adewumi, Karmanya Aggarwal, Pawan Sasanka Ammanamanchi, Aremu Anuoluwapo, Antoine Bosselut, Khyathi Raghavi Chandu, Miruna Clinciu, Dipanjan Das, Kaustubh D. Dhole, Wanyu Du, Esin Durmus, Ondřej Dušek, Chris Emezue, Varun Gangal, Cristina Garbacea, Tatsunori Hashimoto, Yufang Hou, Yacine Jernite, Harsh Jhamtani, Yangfeng Ji, Shailza Jolly, Mihir Kale, Dhruv Kumar, Faisal Ladhak, Aman Madaan, Mounica Maddela, Khyati Mahajan, Saad Mahamood, Bodhisattwa Prasad Majumder, Pedro Henrique Martins, Angelina McMillan-Major, Simon Mille, Emiel van Miltenburg, Moin Nadeem, Shashi Narayan, Vitaly Nikolaev, Rubungo Andre Niyongabo, Salomey Osei, Ankur Parikh, Laura Perez-Beltrachini, Niranjan Ramesh Rao, Vikas Raunak, Juan Diego Rodriguez, Sashank Santhanam, João Sedoc, Thibault Sellam, Samira Shaikh, Anastasia Shimorina, Marco Antonio Sobrevilla Cabezudo, Hendrik Strobelt, Nishant Subramani, Wei Xu, Diyi Yang, Akhila Yerukola, and Jiawei Zhou. 2021. The GEM benchmark: Natural language generation, its evaluation and metrics. *ArXiv*, abs/2102.01672.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander Rush. 2019. GLTR: Statistical detection and visualization of generated text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 111–116, Florence, Italy. Association for Computational Linguistics.

Pablo Gervás. 2009. Computational approaches to storytelling and creativity. *AI Magazine*, 30:49–62.

Marjan Ghazvininejad, Xing Shi, Yejin Choi, and Kevin Knight. 2016. Generating topical poetry. In *Proceedings of ACL*, pages 1183–1191.

Marjan Ghazvininejad, Xing Shi, Jay Priyadarshi, and Kevin Knight. 2017. Hafez: an interactive poetry generation system. In *Proceedings of ACL 2017, System Demonstrations*, pages 43–48, Vancouver, Canada. Association for Computational Linguistics.

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

Yoav Goldberg, Graeme Hirst, Yang Liu, and Meng Zhang. 2018. Neural network methods for natural language processing. *Computational Linguistics*, 44(1).

Seraphina Goldfarb-Tarrant, Haining Feng, and Nanyun Peng. 2019. Plan, write, and revise: an interactive system for open-domain story generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 89–97, Minneapolis, Minnesota. Association for Computational Linguistics.

Joshua Goodman. 2001. Classes for fast maximum entropy training. In *ICASSP*.

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.

Jian Guan and Minlie Huang. 2020. UNION: An Unreferenced Metric for Evaluating Open-ended Story Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9157–9166, Online. Association for Computational Linguistics.

Tatsunori Hashimoto, Hugh Zhang, and Percy Liang. 2019. Unifying human and statistical evaluation for natural language generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1689–1701, Minneapolis, Minnesota. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: surpassing human-level performance on ImageNet classification. In *Proceedings of ICCV*.

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

Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. 2017. Tracking the world state with recurrent entity networks. *ICLR*.

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

Guy Hoffman and Gil Weinberg. 2011. Interactive improvisation with a robotic marimba player. *Autonomous Robouts*, 31(2–3):133–153.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *Proceedings of the 2020 International Conference on Learning Representations*.

Eric Horvitz. 1999. Principles of mixed-initiative user interfaces. In *Proceedings of CHI*.

David M. Howcroft, Anya Belz, Miruna-Adriana Clinciu, Dimitra Gkatzia, Sadid A. Hasan, Saad Mahamood, Simon Mille, Emiel van Miltenburg, Sashank Santhanam, and Verena Rieser. 2020. Twenty years of confusion in human evaluation: NLG needs evaluation sheets and standardised definitions. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 169–182, Dublin, Ireland. Association for Computational Linguistics.

Gao Huang, Chuan Guo, Matt J. Kusner, Yu Sun, Fei Sha, and Kilian Q. Weinberger. 2016. Supervised word mover's distance. In *NeurIPS*.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2020. Automatic detection of generated text is easiest when humans are fooled. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1808–1822, Online. Association for Computational Linguistics.

Daphne Ippolito, David Grangier, Chris Callison-Burch, and Douglas Eck. 2019. Unsupervised hierarchical story infilling. In *Proceedings of the First Workshop on Narrative Understanding*, pages 37–43, Minneapolis, Minnesota. Association for Computational Linguistics.

Mikhail Jacob and Brian Magerko. 2015. Viewpoints AI. In *Proceedings of the 2015 ACM SIGCHI Conference on Creativity and Cognition*, pages 361–362, New York, NY. ACM.

Natasha Jaques, Shixiang Gu, Dxmitry Bahdanau, Jose Miguel Hernandez-Lobato, Richard E. Turner, and Douglas Eck. 2017. Counterfactual multi-agent policy gradients. In *ICML*.

Yangfeng Ji, Trevor Cohn, Lingpeng Kong, Chris Dyer, and Jacob Eisenstein. 2016. Document context language models. In *Proceedings of the 4th International Conference on Learning Representations (Workshop Track)*.

Yangfeng Ji, Chenhao Tan, Sebastian Martschat, Yejin Choi, and Noah A. Smith. 2017. Dynamic entity representations in neural language models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1830–1839, Copenhagen, Denmark. Association for Computational Linguistics.

Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased learning-to-rank with biased feedback. In *Proceedings of WSDM*.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *ACL*.

Daniel Khashabi, Gabriel Stanovsky, Jonathan Bragg, Nicholas Lourie, Jungo Kasai, Yejin Choi, Noah A. Smith, and Daniel S. Weld. 2021. GENIE: A leaderboard for human-in-the-loop evaluation of text generation. *ArXiv*, abs/2101.06561.

Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. 2016. Globally coherent text generation with neural checklist models. In *EMNLP*.

Mert Kilickaya, Aykut Erdem, Nazli Ikizler-Cinbis, and Erkut Erdem. 2017. Re-evaluating automatic metrics for image captioning. In *EACL*.

Joy Kim, Justin Cheng, and Michael S. Bernstein. 2014. Ensemble: exploring complementary strengths of leaders and crowds in creative collaboration. In *Proceedings of CSCW*.

Joy Kim, Sarah Sterman, Allegra Argent Beal Cohen, and Michael S Bernstein. 2017. Mechanical novel: Crowdsourcing complex work through reflection and revision. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*, pages 233–245. Association for Computing Machinery.

Svetlana Kiritchenko and Saif Mohammad. 2017. Best-worst scaling more reliable than rating scales: A case study on sentiment intensity annotation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 465–470, Vancouver, Canada. Association for Computational Linguistics.

Aniket Kittur and Robert E. Kraut. 2008. Harnessing the wisdom of crowds in wikipedia: Quality through coordination. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work*, CSCW '08, page 37–46, New York, NY, USA. Association for Computing Machinery.

Emiel Krahmer and Kees Van Deemter. 2012. Computational generation of referring expressions: A survey. *Computational Linguistics*, 38(1):173–218.

Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. 2015. From word embeddings to document distances. *ICLR*, 37.

Mirella Lapata and Regina Barzilay. 2005. Automatic evaluation of text coherence: Models and representations. In *IJCAI*.

Jey Han Lau, Timothy Baldwin, and Trevor Cohn. 2017. Topically driven neural language model. In *ACL*.

Alon Lavie and Abhaya Agarwal. 2007. METEOR: An automatic metric for MT evaluation with high levels of with human judgments. In *Second Workshop on Statistical Machine Translation*.

Jonathan Lazar. 2017. In *Research Methods in Human Computer Interaction*, 2nd edition edition, page Chapter 11. Elsevier, Cambridge, MA.

Chris van der Lee, Albert Gatt, Emiel van Miltenburg, and Emiel Krahmer. 2021. Human evaluation of automatically generated text: Current trends and best practice guidelines. *Computer Speech & Language*, 67:101151.

Chris van der Lee, Albert Gatt, Emiel van Miltenburg, Sander Wubben, and Emiel Krahmer. 2019. Best practices for the human evaluation of automatically generated text. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 355–368, Tokyo, Japan. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

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

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

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132, Austin, Texas. Association for Computational Linguistics.

Lara J. Martin, Prithviraj Ammanabrolu, William Hancock, Shruti Singh, Brent Harrison, and Mark O. Riedl. 2018. Event representations for automated story generation with deep neural nets. In *AAAI*.

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

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *LREC*.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proceedings of Interspeech*.

Tanushree Mitra, C.J. Hutto, and Eric Gilbert. 2015. Comparing person- and process-centric strategies for obtaining quality data on Amazon Mechanical Turk. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 1345–1354. Association for Computing Machinery.

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

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. SummaRuNNer: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*.

Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Caglar Gülçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *CoNLL*.

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 preprint arXiv:1701.03980*.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark. Association for Computational Linguistics.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2018. RankME: Reliable human ratings for natural language generation. 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 72–78, New Orleans, Louisiana. Association for Computational Linguistics.

Daniel M. Oppenheimer, Tom Meyvis, and Nicolas Davidenko. 2009. Instructional manipulation checks: Detecting satisficing to increase statistical power. In *Journal of Experimental Social Psychology*, volume 45, pages 867–872. Elsevier.

Naoki Otani, Toshiaki Nakazawa, Daisuke Kawahara, and Sadao Kurohashi. 2016. IRT-based aggregation model of crowdsourced pairwise comparison for evaluating machine translations. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 511–520, Austin, Texas. Association for Computational Linguistics.

Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T. Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of ACL*.

Gözde Özbal, Daniele Pighin, and Carlo Strapparava. 2013. BRAINSUP: brainstorming support for creative sentence generation. In *Proceedings of ACL*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Ramakanth Pasunuru and Mohit Bansal. 2017. Reinforced video captioning with entailment rewards. In *EMNLP*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *ICLR*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL*.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL-Shared Task*.

Christopher Purdy, Xinyu Wang, Larry He, and Mark O. Riedl. 2018. Predicting generated story quality with quantitative measures. In *AIIDE*.

D. Quick and Kelland Thomas. 2019. A functional model of jazz improvisation. *Proceedings of the 7th ACM SIGPLAN International Workshop on Functional Art, Music, Modeling, and Design*.

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

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. In *ICLR*.

Hannah Rashkin, Asli Celikyilmaz, Yejin Choi, and Jianfeng Gao. 2020. PlotMachines: Outline-conditioned generation with dynamic plot state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4274–4295, Online. Association for Computational Linguistics.

Ehud Reiter. 2011. Task-based evaluation of NLG systems: Control vs real-world context. In *UCNLG+Eval*.

Ehud Reiter and Anja Belz. 2006. GENEVAL: A proposal for shared-task evaluation in NLG. In *INLG*.

Ehud Reiter and Anja Belz. 2009. An investigation into the validity of some metrics for automatically evaluating natural language generation systems. In *CVPR*.

Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. 2016. Self-critical sequence training for image captioning. In *CVPR*.

Mark Riedl, Amal Alabdulkarim, Louis Castricato, Siyan Li, and Xiangyu Peng. 2021. An introduction to AI story generation. *Blogpost*.

Elena Rishes, Stephanie M. Lukin, David K. Elson, and Marilyn A. Walker. 2017. Generating different story tellings from semantic representations of narrative. *arXiv preprint arXiv:1708.08573*.

Melissa Roemmele and Andrew S. Gordon. 2015. Creative Help: A story writing assistant. In *Proceedings of the International Conference on Interactive Digital Storytelling*.

Melissa Roemmele and Andrew S. Gordon. 2018. Automated assistance for creative writing with an rnn language model. In *Proceedings of the 23rd International Conference on Intelligent User Interfaces Companion*, IUI '18 Companion, New York, NY, USA. Association for Computing Machinery.

Melissa Roemmele, Andrew S. Gordon, and Reid Swanson. 2017. Evaluating story generation systems using automated linguistic analyses. In *Proceedings of the Workshop on Machine Learning for Creativity*. ACM SIGKDD Conference on Knowledge Discovery and Data Mining.

Y. Rubner, C. Tomasi, and L. J. Guibas. 1998. A metric for distributions with applications to image databases. In *IEEE*.

James Ryan. 2017. Grimes' fairy tales: A 1960s story generator.

Manasvi Sagarkar, John Wieting, Lifu Tu, and Kevin Gimpel. 2018. Quality signals in generated stories. In **SEM 2018*.

Sashank Santhanam and Samira Shaikh. 2019. Towards best experiment design for evaluating dialogue system output. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 88–94, Tokyo, Japan. Association for Computational Linguistics.

Stephanie Schoch, Diyi Yang, and Yangfeng Ji. 2020. "This is a problem, don't you agree?" Framing and bias in human evaluation for natural language generation. In *Proceedings of the 1st Workshop on Evaluating NLG Evaluation*, pages 10–16, Online (Dublin, Ireland). Association for Computational Linguistics.

João Sedoc, Daphne Ippolito, Arun Kirubarajan, Jai Thirani, Lyle Ungar, and Chris Callison-Burch. 2019. ChatEval: A tool for chatbot evaluation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 60–65, Minneapolis, Minnesota. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *ACL*.

Abigail See, Aneesh Pappu, Rohun Saxena, Akhila Yerukola, and Christopher D. Manning. 2019. Do massively pretrained language models make better storytellers? In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 843–861, Hong Kong, China. Association for Computational Linguistics.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Robin Sloan. 2016. Writing with the machine.

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015. A neural network approach to context-sensitive generation of conversational responses. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 196–205, Denver, Colorado. Association for Computational Linguistics.

Robert J. Sternberg. 2005. Creativity or creativities? *International Journal of Human-Computer Studies*, 63(4-5):370–382.

Octavia-Maria Sulea. 2017. Recognizing textual entailment in Twitter using word embeddings. In *2nd Workshop on Evaluating Vector-Space Representations for NLP*.

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 *Advances in neural information processing systems*, pages 3104–3112.

Reid Swanson and Andrew S. Gordon. 2008. Say Anything: a massively collaborative open domain story writing companion. In *Proceedings of ICIDS*.

Reid Swanson and Andrew S. Gordon. 2012. Say anything: Using textual case-based reasoning to enable open-domain interactive storytelling. *ACM Transactions on Interactive Intelligent Systems*, 2:16:1–16:35.

Chenhao Tan, Lillian Lee, and Bo Pang. 2014. The effect of wording on message propagation: Topic- and author-controlled natural experiments on Twitter. In *Proceedings of ACL*.

Tsegaye Misikir Tashu and Tomás Horváth. 2018. Pair-Wise: Automatic essay evaluation using word mover's distance. In *CSEDU*.

Alan Turing. 1950. Computing Machinery and Intelligence. In *Mind*, volume LIX, pages 433–460. Oxford University Press (OUP).

Adaku Uchendu, Thai Le, Kai Shu, and Dongwon Lee. 2020. Authorship attribution for neural text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8384–8395, Online. Association for Computational Linguistics.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. CIDEr: Consensus-based image description evaluation. In *CVPR*.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: a neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3156–3164.

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*.

Daniel S. Weld, Mausam, Christopher H. Lin, and Jonathan Bragg. 2014. Artificial intelligence and collective intelligence. In *Handbook of Collective Intelligence*, chapter 3. MIT Press.

Evan J. Williams. 1959. *Regression Analysis*, volume 14. Wiley.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Machine Learning*.

Lingfei Wu, Ian En-Hsu Yen, Kun Xu, Fangli Xu, Avinash Balakrishnan, Pin-Yu Chen, Pradeep Ravikumar, and Michael J. Witbrock. 2018. Word mover's embedding: From word2vec to document embedding. In *EMNLP*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. ArXiv:1609.08144.

Peng Xu, Mostofa Patwary, Mohammad Shoeybi, Raul Puri, Pascale Fung, Anima Anandkumar, and Bryan Catanzaro. 2020. MEGATRON-CNTRL: Controllable story generation with external knowledge using large-scale language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2831–2845, Online. Association for Computational Linguistics.

Qian Yang, Justin Cranshaw, Saleema Amershi, Shamsi T. Iqbal, and J. Teevan. 2019. Sketching NLP: A case study of exploring the right things to design with language intelligence. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*.

Georgios N. Yannakakis, Antonios Liapis, and Constantine Alexopoulos. 2014. Mixed-initiative co-creativity. In *Proceedings of FDG*.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big Bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019a. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Elizabeth Clark, Lianhui Qin, Ali Farhadi, and Yejin Choi. 2021. TuringAdvice: A generative and dynamic evaluation of language use. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4856–4880, Online. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019b. Defending against neural fake news. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Hao Zhang and Jie Wang. 2018. Semantic WordRank: Generating finer single-document summarizations. In *IDEAL*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020a. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 11328–11339. PMLR.

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Earth mover's distance minimization for unsupervised bilingual lexicon induction. In *EMNLP*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020b. BERTScore: Evaluating text generation with BERT. In *International Conference on Learning Representations*.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceed-*

ings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 563–578, Hong Kong, China. Association for Computational Linguistics.

Yaoming Zhu, S. Lu, L. Zheng, Jiaxian Guo, W. Zhang, J. Wang, and Y. Yu. 2018. Texygen: A benchmarking platform for text generation models. *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*.

Yukun Zhu, Ryan Kiros, Rich 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 *Proceedings of ICCV*, pages 19–27.

Chapter A

Appendix One

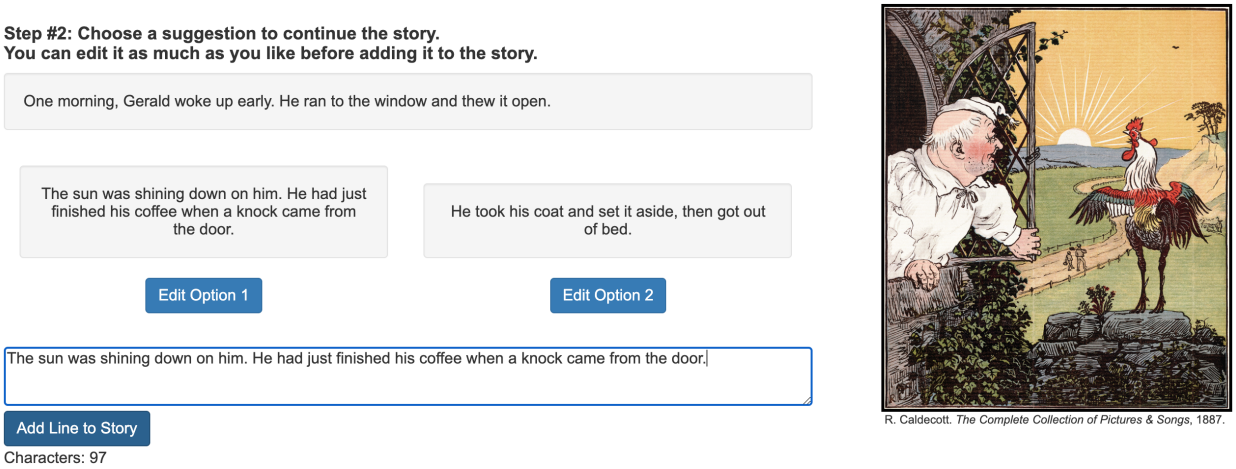


Figure A.1: The story writing interface. The first box was the first turn of writing (author writing alone). In this case, Option 1 was generated with NUCLEUS sampling and Option 2 with TOP-K sampling. The writer has chosen Option 1, which shows up in the text box below and can now be edited before adding it to the story.

A.1 Writing Interface

A screenshot of the interface is shown in Fig. A.1.

A.2 Data Details

We filter the WritingPrompts dataset to contain the first 500 words of all the stories; we do not use the prompts. After filtering, the dataset has 56,582 types and 55,785,118 tokens. 1.3% of the data is replaced with UNK. The original dataset can be found at `https://github.com/pytorch/fairseq/tree/master/examples/stories`.

Because the fusion model was originally trained to map from "prompt" to "story," we reconfigure the data and retrain the model to map from "story beginning" to "story end." To do this, we randomly split the stories at a newline and make the first portion of the story the "source" and the second portion the "target." In cases where there are no newlines within the text, we instead split on a space.

	OVERALL			FUSION			GPT2		
	ED (↓)	JS (↑)	USER (↑)	ED (↓)	JS (↑)	USER (↑)	ED (↓)	JS (↑)	USER (↑)
Total	32.27	57.85	67.97	37.61	51.13	60.69	29.49	61.35	71.77
Sugg. #1	24.10	65.11	73.77	25.76	62.09	70.17	23.57	66.05	74.90
Sugg. #2	27.26	62.15	71.71	34.19	48.09	57.79	24.22	68.31	77.81
Sugg. #3	34.40	55.96	67.09	36.51	52.20	63.74	33.15	58.17	69.06
Sugg. #4	31.52	59.74	70.89	36.95	56.40	65.55	28.32	61.71	74.05
Sugg. #5	44.08	46.29	56.39	48.13	41.71	50.61	41.03	49.73	60.72

Table A.1: Edit distance (ED), Jaccard similarity (JS), and USER scores between the edited and the original generated suggestions overall, from FUSION, and from GPT2.

A.3 Model Details

A.3.1 Fusion model

We train the fusion model with the data split in "source" and "target" as described in App. A.2, using the settings described at `https://github.com/pytorch/fairseq/tree/master/examples/stories`. We pretrain the model for 9 epochs before adding the fusion model and training for 14 epochs.

To generate, we assign an UNK penalty $= 10$ to suppress UNKs and use top-k sampling with $k = 40$.

A.3.2 GPT2 model

We finetune small GPT2 model on the WritingPrompts data using the code and settings at `https://github.com/huggingface/transformers`. We finetune the model for 3 epochs.

To generate, we use either top-k sampling with $k = 40$ (for GPT2 and TOP-K) or nucleus sampling with $p = 0.9$ (for NUCLEUS).

A.4 Results

A.4.1 Edit Results by Suggestion

The full results, broken down by suggestion #, for edit distance, Jaccard similarity, and USER are in Table A.1 (for FUSION vs. GPT2) and Table A.2 (for NUCLEUS vs. TOP-K).

	OVERALL			NUCLEUS			TOP-K		
	ED (↓)	JS (↑)	USER (↑)	ED (↓)	JS (↑)	USER (↑)	ED (↓)	JS (↑)	USER (↑)
Total	35.74	52.21	62.86	34.65	53.64	63.64	36.69	50.96	62.18
Sugg. #1	32.40	54.12	65.56	26.98	61.37	71.31	36.28	48.93	61.44
Sugg. #2	37.13	49.24	60.22	37.71	47.86	58.25	36.62	50.44	61.95
Sugg. #3	32.93	55.56	65.04	34.79	51.43	60.58	31.31	59.16	68.92
Sugg. #4	33.35	55.37	66.60	34.38	57.31	66.77	32.45	53.68	66.45
Sugg. #5	42.89	46.75	56.90	38.23	51.28	62.24	47.84	41.94	51.24

Table A.2: Edit distance (ED), Jaccard similarity (JS), and USER scores between the edited and the original generated suggestions overall, from NUCLEUS, and from TOP-K.

A.4.2 Likert-Scale Results

As shown in Fig. A.2, the majority of participant responses are positive about their experience of writing with the CYOA, regardless of which model pair they were working with. The median score for almost all questions is 4 ("Agree").

The one exception is the median response for *The suggestions connected to what had happened in the story so far.* for FUSION vs. GPT2 is slightly lower–3 ("Neutral"). As hypothesized in §4.3, this is likely due to the higher degree of randomness in the FUSION-generated text.

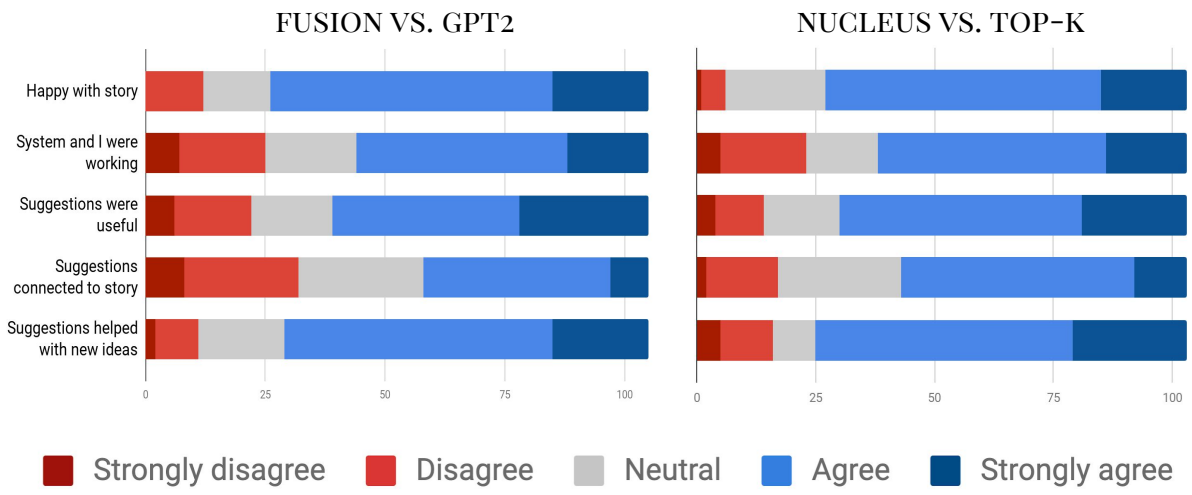


Figure A.2: The Likert-scale results for FUSION vs. GPT2 and NUCLEUS vs. TOP-K.

Chapter B

Appendix Two

B.1 Datasets

Summaries and Essays: For the intrinsic tasks in §5.4, we use two types of human-evaluated texts: machine-generated summaries and human-authored essays. We follow Kusner et al. [2015] and remove punctuation and stopwords. (For contextual embeddings, these are removed after the embeddings are obtained.) The details of the subsets we used are in Table B.1.

	Summaries	Essays
# documents	2,085	1,088
# tokens	255,609	164,776
# types	12,882	6,381
average length (tokens)	65	151
average length (sent.)	3.4	7.5

Table B.1: Corpora statistics.

CNN/Daily Mail: CNN/Daily Mail dataset Nallapati et al. [2017]; Hermann et al. [2015] is a collection of online news articles along with multi-sentence summaries. We use the same data splits as in Nallapati et al. [2017]. Earlier work anonymized entities by replacing each named entity with a unique identifier (e.g., *Dominican Republic*→`entity15`). In this work we used the non-anonymized version.

Stats	**CNN/DM**
Avg. # tokens document	781
Avg. # tokens summary	56
Total # train doc-summ. pair	287,229
Total # validation doc-summ. pair	13,368
Total # test doc-summ. pair	11,490
Input token length	400/800
Output token length	100

Table B.2: Summary statistics of CNN/Daily Mail (CNN/DM) Datasets.

B.2 Essays Dataset Evaluation

To test the metrics on human-authored text, we use a dataset of graded student essays that consists of responses to standardized test questions for tenth graders. We use a subset of Question #3 from the exam, which asks the test-taker to synthesize information from a reading passage, where student responses contain 5–15 sentences. Graders assigned the student-authored responses with scores ranging from 0 to 3. For the

reference essay, we use a top-scoring sample essay, which the graders had access to as a reference while assigning scores. The full reference essay is in B.3.

Table 5.3 shows the correlation of each metric with the evaluators' scores. As in the summarization task, SMS outperforms both ROUGE-L and WMS. However, in this case, having the sentence representations in the metric gives the best result, with S+WMS correlating best with human scores, significantly better than ROUGE-L. This is consistent across embedding type; once again, the choice of embedding does not create a significant difference between the sentence mover's metrics.[1]

Discussion Aside from the length of the text, the Essays dataset presents the metrics with several challenges not found in the Summaries dataset. For example, the dataset contains a large number of spelling mistakes, due to both author misspellings and errors in the transcription process. One essay begins, "The setting of the story had *effected* the *cycle's becuse* if it was *sub earbs* he could have *stoped any where* and got water ..."

The tone and style of the essay can also vary from the reference essay. (For example, the author of Sample #3 in B.3 ends their essay by reflecting on how they would respond in the protagonist's place.) Embedding-based metrics may be more forgiving to deviations in writing style from the reference essay, such as the use of first person.

While Table 5.3 indicates sentence mover's similarity metrics significantly improve correlation with human judgments over standard methods, there is still enough disagreement that we believe automatic metrics should not replace human evaluations. Rather, they should complement human evaluations as an automatic proxy that can be used for intermediate evaluation and as a reward signal when learning, as we show in §5.6.

B.3 More Examples

In Table B.3, we show samples of the summaries that we used to perform intrinsic evaluations in the main text.

[1] Williams test: $p = 0.33$ (SMS) and $p = 0.46$ (S+WMS)

B.4 Extrinsic Model Training Details

We use 128 dimensional bidirectional 2-layered LSTMs for the encoder and 128 unidirectional LSTMs for the decoder. For both datasets, we limit the input and output vocabulary size to the 30,000 most frequent tokens in the training set. We initialize word embeddings with FastText[2] [Mikolov et al., 2018] 300-dimensional vectors and finetune them during training. For WMS, SMS and S+WMS embeddings, we use the GloVe word embeddings described in §5.4. We train using Adam with a learning rate of 0.001 for the MLE models and 10^{-5} for the MLE+RL models. We select the MLE models with the lowest cross-entropy loss and the MLE+RL models with the highest reward on a sample of validation data to evaluate on the test set. At test time, we use beam search of width 5 on all our models to generate final predictions. For the Mixed RL trained models, we initialize the weights with pre-trained MLE model, and we start with $\gamma = 0.97$ and gradually increase its value. We train our models for $\sim$25 epochs which took 1–2 days on an NVIDIA V100 GPU machine.

B.5 Policy Gradient Reinforce Training

Maximum likelihood-based training of sequence generation models poses exposure bias issues since the model is evaluated by comparing the model to empirical distribution, whereas at test time we use automatic metrics to evaluate the model generated text Ranzato et al. [2015]. Reinforced based policy gradient approach is used to address this issue by learning to optimize discrete target evaluation metrics that are non-differentiable. We use REINFORCE Williams [1992] to learn a policy p_θ defined by the model parameters θ to predict the next action (word). The RL loss function is defined as:

$$L_{RL} = \mathbb{E}_{\hat{y} \sim p_\theta}[r(\hat{y})] \tag{B.1}$$

where $\hat{y}$ is the sequence of sampled words. The derivative of the the objective function based on Monte Carlo sampling yields:

$$\bigtriangledown_\theta \, L_{RL} = -(r(\hat{y}) - b) \bigtriangledown_\theta \log p_\theta(\hat{y}) \tag{B.2}$$

[2] https://fasttext.cc/docs/en/english-vectors.html

The baseline b is a bias estimator and is used for variance reduction in RL training. In this work we use *self-critical training* and use the reward obtained from a sequence that is generated by greedily decoding, $\tilde{y}$, as a baseline:

$$\bigtriangledown_\theta L_{RL} = -(r(\hat{y}) - r(\tilde{y})) \bigtriangledown_\theta \log p_\theta(\hat{y}) \tag{B.3}$$

B.6 Sample Generated Summaries

Examples of the generated summaries are in Table B.4.

B.7 Human Evaluations

We collected human evaluations for 100 summaries generated by the mixed loss models to compare ROUGE-L as a reward to WMS, SMS, and S+WMS. Amazon Mechanical Turkers chose between two generated summaries, one from the ROUGE-L model and one from WMS, SMS, or S+WMS. They selected one of the two summaries based on: (1) *non-redundancy*, fewer repeated ideas, (2) *coherence*, clearly expressed ideas, (3) *focus*, ideas free of superfluous details, and (4) *overall*, the summary effectively communicates the article's content. These criteria help evaluate the impact of the metrics used as reward.

We randomly selected 100 samples from the CNN/Daily Mail test set and use workers from Amazon Mechanical Turk as judges to evaluate them on the four criteria (redundancy, focus, coherence, and overall). Following DUC (Document Understanding Conferences) style evaluations (`https://duc.nist.gov/`), we performed a head-to-head evaluation and randomly showed Turkers two model-generated summaries. We asked the human annotators to rate each summary on the same metrics as before without seeing the source document or ground truth summaries.

Results We asked human judges to evaluate the output of the mixed loss model trained with a ROUGE-L reward versus models trained with WMS, SMS, and S+WMS the reward. The results are shown in Table B.5.

Human judges significantly prefer summaries produced by models optimized with WMS, SMS, and S+WMS over ROUGE-L. SMS and S+WMS were preferred over ROUGE-L more often than WMS was. There is no significant difference between the evaluations of SMS and S+WMS. Among all other metrics, SMS was rated the highest on the non-redundancy question (69% improvement over the ROUGE-L score), indicating

that the model learns to generate summaries that contain less repetition between sentences.

While the SMS model's output was highly-scored by both the automatic and human evaluations, removing word-level scoring does come with a cost, as seen in the example in Table B.4. The SMS summary contains a mistake, stating that "*priscilla will tie the knot*" instead of "*serve as a witness*". This issue may be mitigated by a better encoder for the summarization task and better sentence and word representations. As future work, we will investigate summarization models with more complex sentence embeddings and encoder structures (e.g., self-attention models).

Samples	Summaries
Sample #1	**Reference.** Freddie Gray, who is black, asked for medical help but was denied during 00-minute police car ride, eventually paramedics were called. Deputy police commissioner Kevin Davis conceded their failure. But chief commissioner refuses to resign over the death. Six officers are suspended without pay during an investigation. **Hypothesis.** Baltimore Police Commissioner Anthony Batts ruled out his resignation despite that fact that his deputy admitted they should have sought medical attention for Freddie Gray. Six officers have been suspended with pay as local police and federal authorities investigate. Commissioner Anthony Batts has ruled out the possibility of his resignation.
Sample #2	**Reference.** Choc on Choc's chocolates come in three different flavours. The face of each politician is emblazoned on milk Belgium chocolate bars. Cameron's has blueberries, Clegg is honeycomb and Miliband is raspberry. **Hypothesis.** UNK lollies on 273 invalid chocolates come in three different flavours. Contains three different flavours - the colours associated with each leader. David Cameron, Nick Clegg, Nick Clegg, Nick Clegg and David Cameron.
Sample #3	**Reference Essay.** The setting seems to be as formidable an opponent as the actual workout. It seems as if everything is against the cyclist, including nature. As the day progresses, and the cyclist's journey continues, the setting becomes harsher and harsher. After passing the first "town", the "sun was beginning to beat down." In need of water, all a cruel pump gives him is "a tarlike substance." His sufferings continue, increasingly pummeled by his surroundings and his thirst for water. If dehydration was not enough, the flat terrain gave way to "rolling hills", which would only punish his legs more. Reaching possible salvation, his hopes are crushed when the "Welch's Grape Juice Factory" turns out to be abandoned. All these events are enough to destroy anyone's spirit. The cyclist almost gives up hope to accept certain death. He has become ferociously beaten by his very surroundings. It appears as if he is fated to die alone in the blistering heat. Although he hangs his head in despair, he still continues on the path of disappointment. In a twist of fate, he encounters a thriving store where he halts and drinks. Finally encountering his salvation, this particular setting brings new hope and relief to the cyclist who has finally survives his trek through nature. **Hypothesis.** The features of the setting affect the cyclist alot. The hot sun beating down on him makes him sweat and makes him thirsty. The bumpy roods and hills make him work harder. The abandoned places make him lose hope. If faced with these obstacles I would have been affected in the same way. As I believe any human would be.

Table B.3: Examples of human generated and model generated summaries from Summaries and Essays datasets

Human Summary	the 69 - year - old collaborated with nbc 's today show to launch a contest for an elvis - obsessed couple to win the ' ultimate wedding ' . the winning duo will get married in the brand new elvis presley 's graceland wedding chapel at the westgate hotel on thursday , april 23 . while she agreed to make an appearance , the woman who wed elvis in 1967 made one thing clear before unveiling the latest wedding chapel to bear his name : no impersonators .
Model	**Generated Summary**
ROUGE-L	priscilla presley will serve as a witness at the first wedding to be held at an all - new chapel of love in las vegas . the 69 - year - old collaborated with nbc 's today show to launch a contest for one elvis - obsessed couple to win the ' ultimate wedding ' . elvis performed more than 830 sold - out shows .
WMS	the 69 - year - old collaborated with nbc 's today show to launch a contest for one elvis - obsessed couple to win the ' ultimate wedding ' . the winning duo – announced next monday – will tie the knot at elvis presley 's graceland wedding chapel inside the westgate hotel on thursday , april 23 .
SMS	priscilla presley will tie the knot at elvis presley 's graceland wedding chapel inside the westgate hotel on thursday , april 23 . the 69 - year - old collaborated with nbc 's today show to launch a contest for one elvis - obsessed couple to win the ' ultimate wedding ' .
S+WMS	priscilla presley will serve as a witness at the first wedding to be held at an all - new chapel of love in las vegas . the 69 - year - old collaborated with nbc 's today show to launch a contest for one elvis - obsessed couple to win the ' ultimate wedding ' .

Table B.4: Summaries generated from the mixed MLE+RL loss models with ROUGE-L, WMS, S+WMS, and SMS metrics as rewards, along with the corresponding human-authored reference summary.

	ROUGE-L vs. WMS			ROUGE-L vs. SMS			ROUGE-L vs. S+WMS		
Criteria	ROUGE-L	WMS	=	ROUGE-L	SMS	=	ROUGE-L	S+WMS	=
non-redundancy	76	**122**	102	64	**144**	92	66	**132**	102
coherence	102	**158**	40	83	**170**	47	83	**166**	51
focus	99	**161**	40	79	**174**	47	84	**166**	50
overall	108	**160**	32	85	**179**	36	84	**179**	37

Table B.5: Human evaluations on a random subset of 100 summaries. The frequencies from the head-to-head comparison of models trained with ROUGE-L against WMS/SMS/S+WMS are shown. Each summary is evaluated by 3 judges (300 summaries per criteria). '=' indicates no difference. All improvements are statistically significance at $p < 0.001$.

Chapter C

Appendix Three

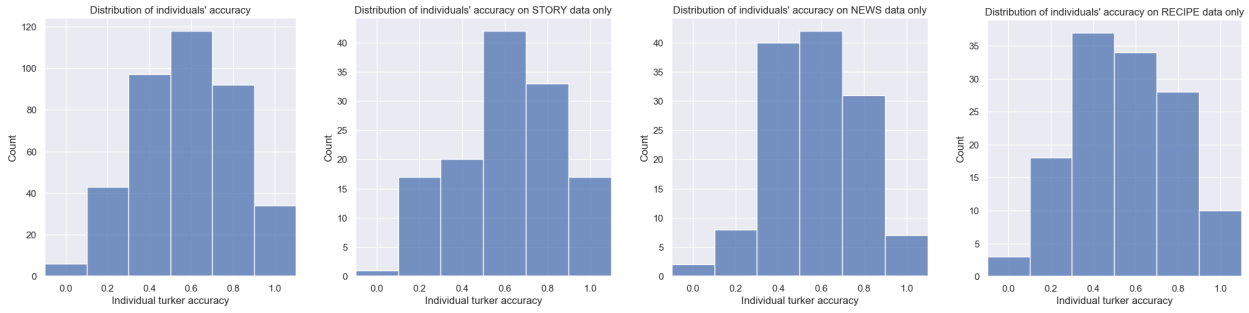
C.1 Newspapers

Each newspaper came from a randomly chosen U.S. state and was selected from Wikipedia's lists of newspapers by state (en.wikipedia.org/wiki/List_of_newspapers_in_the_United_States#By_state_and_territory). The human-authored news articles and prompts came from the following states and websites:

- HI: www.westhawaiitoday.com
- CT: www.greenwichtime.com/
- WA: www.vashonbeachcomber.com/
- SD: www.argusleader.com/
- CA: www.redding.com/
- MA: www.lowellsun.com/
- NE: starherald.com/
- VA: dailyprogress.com/
- WV: www.theintermountain.com/
- NM: www.lcsun-news.com/
- LA: www.nola.com/
- IA: qctimes.com/
- NY: www.pressconnects.com/
- IN: www.pal-item.com/
- NJ: www.northjersey.com/

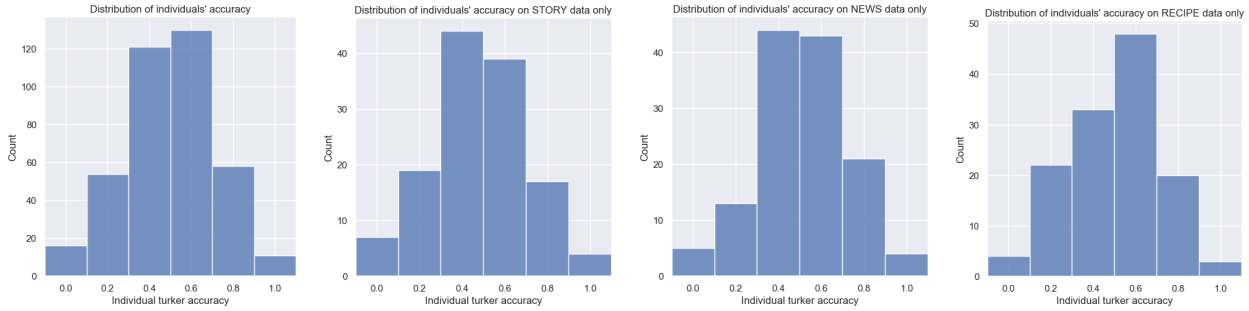
C.2 Score Frequencies

The frequency of the scores (out of 5) received by evaluators is shown in Figures C.1 (for GPT2 experiments) and C.2 (for GPT3 experiments).



(a) GPT2 overall **(b)** GPT2 story **(c)** GPT2 news **(d)** GPT2 recipe

Figure C.1: Histogram of scores classifying human and GPT2 texts.



(a) GPT3 overall **(b)** GPT3 story **(c)** GPT3 news **(d)** GPT3 recipe

Figure C.2: Histogram of scores classifying human and GPT3 texts.

C.3 Annotation Details

The authors annotated 300 comments (150 from the No Training experiment and 150 from the Examples experiment). For each experiment, we randomly chose 50 authors from each setting and randomly added 1 of their responses to the annotation set. Each comment was annotated by 2 of the authors. The annotation labels are shown in Table C.1. To create the set of annotation labels, the authors created a candidate list of labels, annotated a subset of the data collected in the pilot study (Appendix C.5) together, then another subset separately, and finally refined the labels based on feedback from that process. Because evaluators' responses often contained more than one reason for their choice, comments could receive more than one label.

Category	Label	Description	Example
Form	Grammar	The spelling and grammar of the text, punctuation/formatting issues	I would make the text more grammatical by adding more punctuation where necassary.
	Level of detail	Is the text simple or does it go more in-depth?	i would include more examples and explanations of the statements. The author need to elaborate more on the topic.
	Genre	If the text is the genre/domain/style/formality that the reader expects, adheres to style norms	written exactly the way a human will tell a story
Content	Repetition	Words/phrases/content repeated itself	Repeating "or some would say" seemed very unnatural.
	Factuality	The accuracy of the text, whether it describes things that are "true."	The article lists many facts that make the information seem like it was machine-generated.
	Consistency	How the text relates to the context and other pieces of the text	The subject of the article follows the headline well without repeating it exactly
	Common sense	Whether the text "makes sense" within the world that it is written	Change the "bake in the preheated oven for 20 minutes on top of the stove." You can't bake on top of the stove but to bake in the oven.
	Coherence	The structure and coherence of the text. Order issues go here.	More cohesion between sentences. Feel loosely related, but wording is strange.
Machine capabilities	Writer intent and expression	Speculating about writer's intent or capabilities (e.g., ability to express emotions)	The text is thorough and tries to cover all basis of the situation. It is very inclusive and humans worry about being inclusive not machines.
Null	Miscellaneous	Everything else	too many dialogue-like things, and make it less gender-dicey.
	Null/Vague	No reasons given, or too vague to be considered a real reason	i selected this rating because it is definitely written by human

Table C.1: The annotation labels, along with an example of each label. Note that some example sentences would also be labeled with additional labels. We did not use the Null category in the paper's analyses.

C.4 Evaluators' Expectations of Generated Text

Because we asked evaluators whether they thought the text was human- or machine-authored, they often justified their choices by explaining what types of human language they believed machines could (or could not) generate. We took note of these comments and annotated for them in our data annotation process (Appendix C.3) because they demonstrate the expectations evaluators have for the quality of machine-generated text. Some example comments shown in Table C.2.

Punctuation is perfect as well as the flow of the text. There is also more complex punctuation, such as quotes, that I think a computer would get wrong.
"fried anyone to a crisp." That is a human if I've ever seen one. a bot or AI is more proper, they wouldn't write so casual.
Because it talked about love which robots know nothing about.
Lack of oxford comma. A computer would know better.
The article flows properly, has appropriate English and multiple quotes. This would seem to be more than a bot could create. How would a bot create random quotes?
This was more of a ramble which humans do, not computers.
There are details and key phrases used in this article that computer generated text would not have in it, such as "came up short", "put together a solid drive", "put up any points". These are human specific terms and are not generally able to be programmed into a text program.
This piece quotes the host and I don't believe AI can interview people yet so this has to be human written.
It has a lot of detail in an emotional description that a machine isn't capable of giving to its readers.
The way some words are phrased here again shows the human uncertainty, "let the apples marinate for about 30 minutes". If this was machine-generated, it would most likely just say marinate for 30 minutes.
It seems to know when to use semicolns very well. This could be a human or a really smart computer.
I don't think AIs are capable of writing recipes on their own just yet.
I don't believe a machine could come up with this level of whimsy or creativity and have it make sense.
I don't think AI would use the term 'literally'.
There is a lot of every day language written in this recipe that I couldn't see a machine possibly replicating.
It adds that she is both nervous and excited whereas a machine wouldn't care what emotions are involved.
The writer used proper grammar and punctuation. No bot could write this,
I'm not sure if a computer would get the concept or use the word "your" where the recipe begins with "Start by doing your prep."

Table C.2: Example reasons evaluators gave for their decisions that spoke to their beliefs about current NLG capabilities.

C.5 Pilot Study

Before running the experiments described in the paper, we ran a smaller-scale version with both Amazon Mechanical Turk ($n = 22$) and "expert" evaluators (NLP graduate students; $n = 11$). We asked the evaluators to distinguish between stories authored by humans, GPT2, and GPT3 and to explain their reasoning. When we coded and analyzed their responses, we found that the most accurate evaluators focused on textual aspects like repetition and were less likely to mention aspects like style. The AMT evaluators mentioned grammar and spelling far more frequently than the expert evaluators, who were more likely to mention the

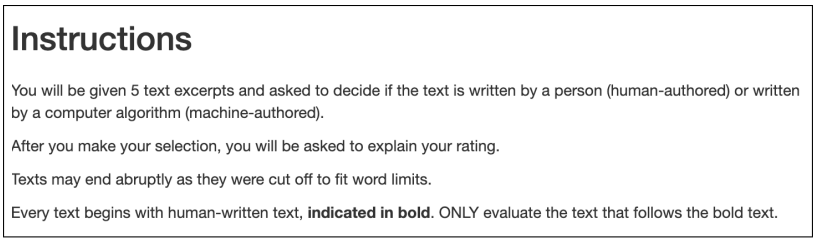


Figure C.3: Basic instructions shown to all evaluators.

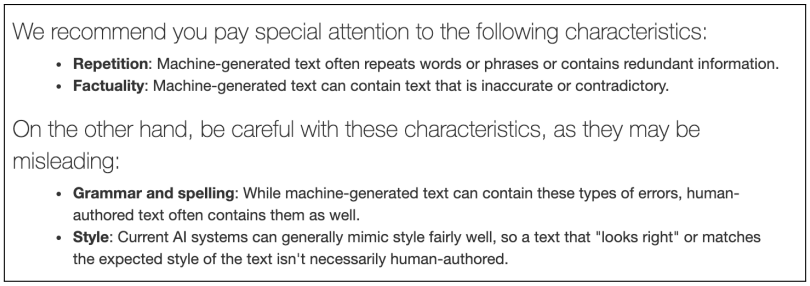


Figure C.4: The Instruction training.

repetition, factuality, and commonsense of the passage.

C.6 Training and Instructions

Figure C.3 shows the basic instructions that were shown to all evaluators, in both §6.2 and §6.3, regardless of training or domain. All training information occurred after receiving the basic instructions.

C.6.1 Instruction Training

The training shown to evaluators in the Instruction training condition is shown in Figure C.4.

C.6.2 Example Training

A screenshot of the Examples and Comparison training is in Figure C.5. The full set of examples and annotations used in the Examples and Comparison trainings can be found in the supplementary materials and at `ark.cs.washington.edu/human_evals_ACL21`.

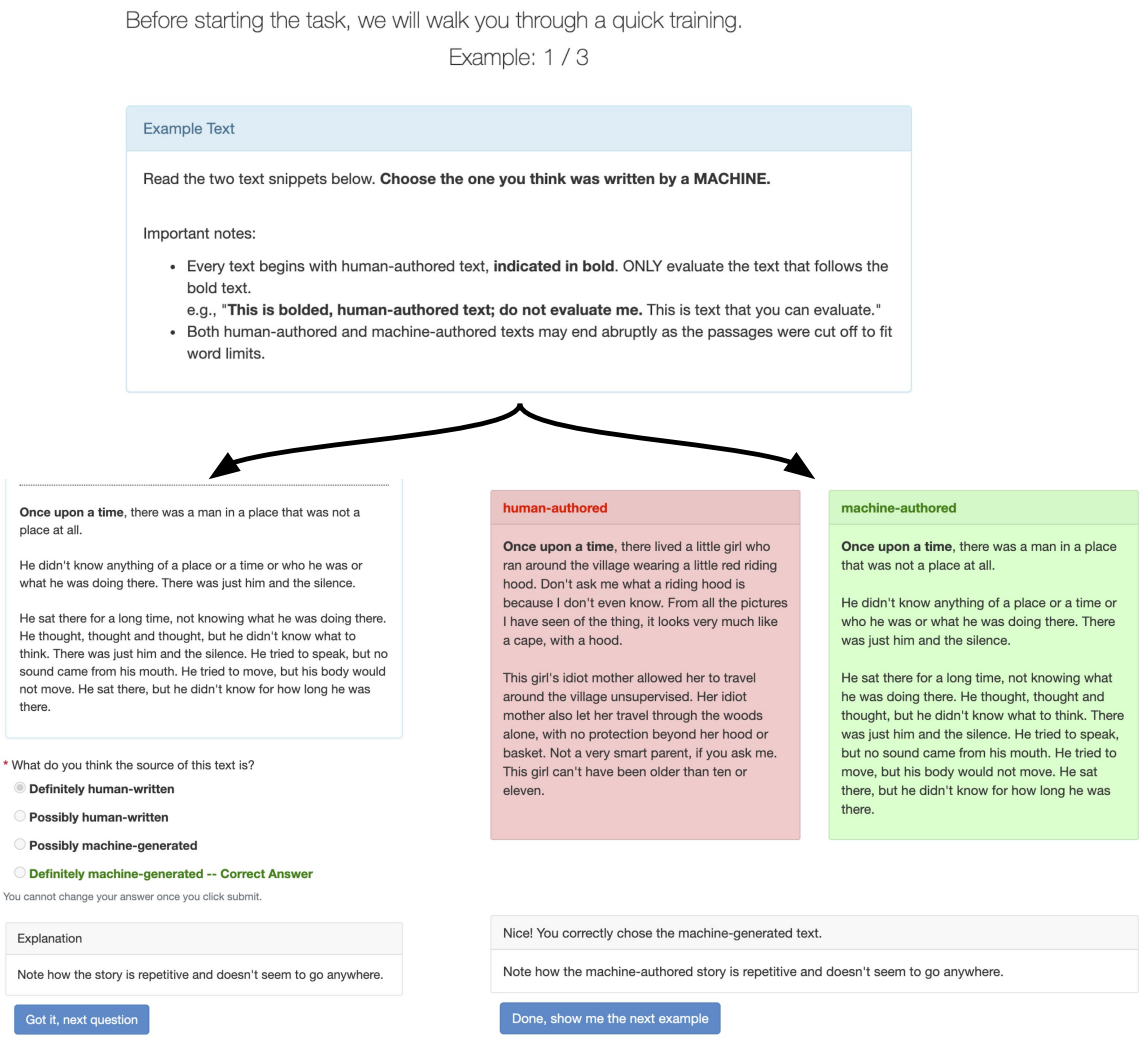


Figure C.5: The Example training (left) and Comparison training (right) in the story domain. The instructions are the same for both, except "Choose the one you think was written by a machine." was in Comparison only.