Machine-Assisted Script Curation

Manuel R. Ciosici Joseph Cummings, Mitchell DeHaven, Alex Hedges, Yash Kankanampati, Dong-Ho Lee, Ralph Weischedel, Marjorie Freedman
manuelc@isi.edu, weisched@isi.edu, mrf@isi.edu
Information Sciences Institute, University of Southern California

Abstract

We describe Machine-Aided Script Curator (MASC), a system for human-machine collaborative script authoring. Scripts produced with MASC include (1) English descriptions of sub-events that comprise a larger, complex event; (2) event types for each of those events; (3) a record of entities expected to participate in multiple sub-events; and (4) temporal sequencing between the sub-events. MASC automates portions of the script creation process with suggestions for event types, links to Wikidata, and sub-events that may have been forgotten. We illustrate how these automations are useful to the script writer with a few case-study scripts.

1 Introduction

Scripts have been of interest for encoding procedural knowledge and understanding stories for over 40 years (Schank and Abelson, 1977). In the form of checklists, recording procedural knowledge has revolutionized fields like medicine and aviation by encoding expert knowledge and best practices (Degani and Wiener, 1993; Gawande, 2010). In the last few years, researchers have turned their attention to automatic script discovery from text (Chambers, 2013; Weber et al., 2020, 2018). However, exclusively data-driven sub-event discovery methods face the challenge that narrative descriptions often omit common knowledge.[1]

We aim for a process for building a library of scripts through human-machine collaboration leveraging NLP techniques to augment human background knowledge. The resulting demonstration system serves two related purposes. First, it is a knowledge acquisition tool that supports the development of a repository of scripts for use by downstream applications. Second, it is an annotation tool that supports the creation of a library to aid our understanding of how people create scripts. Such a library can inform and/or benchmark future script discovery approaches. Each script includes a natural language description of the steps in the complex event with links to an ontology. Events within a script are connected by (a) temporal order (e.g., negotiating the price of a car happens *before* buying the car) and (b) by shared argument (e.g., the person buying a car is also the person who negotiated its price). We designed *Machine-Aided Script Curator (MASC)*, our script-creation tool, to be used by non-NLP experts.

While approaches to script discovery suffer from the incompleteness of text, human attempts to write machine-interpretable scripts suffer from the writer's own tendency to omit steps and, where required, the challenge of mapping to a formal ontology. To assist the script creators, MASC makes three types of suggestions: (1) the ontological type for each event; (2) a fine-grained ontological type for suggested arguments; and (3) steps that the curator might have forgotten.

In the following sections, we describe the process of creating a script in MASC and the NLP components that support suggestions.[2] While a large-scale script repository is beyond this paper's scope, we have created five sample scripts, which we use as case studies for understanding the script creation process and the suggestion capabilities. In Section 4, we use these scripts to measure the utility of MASC's suggestion capabilities. In Section 5, we describe the scripts' characteristics.

2 Related Work

Schank and Abelson (1977) proposed organizing knowledge about human behavior using scripts. Recent approaches attempt to "induce" scripts from

[1] Common knowledge might be missing from narrative descriptions due to the *quantity* and *relevance* maxims (Grice, 1975).

[2] A video of MASC is available at https://youtu.be/slvZWAYkRmA, and the source code and the sample scripts are at https://github.com/isi-vista/MASC.

Proceedings of NAACL-HLT 2021: Demonstrations, pages 8–17

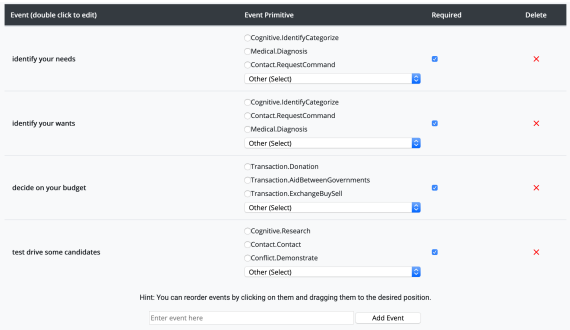

Figure 1: Adding events to the *buying a car* script.

large amounts of data rather than write scripts manually (Rudinger et al., 2015; Weber et al., 2018). Although improving year over year, these models still perform poorly (Recall@100 of ~7%, Weber et al., 2020) at predicting next events, given a set of preceding events - a necessary building block of scripts. These models' training data was obtained by asking human annotators to decide if event B happened because of event A. In contrast, the scripts produced by our curation tool incorporate the complexities of many different events in various causal orderings.

Both symbolic and neural approaches suffer from the lack of generic knowledge to "fill-in-the-blanks" or reject impossible events. Training systems to incorporate common-sense knowledge (Lin et al., 2019; Shwartz et al., 2020) has not yet addressed script creation. Another source of information for script discovery could be extraction from multiple languages and modalities. While some extraction systems have incorporated these other sources (Li et al., 2020), such extractions have not yet fed into script discovery. Resolving the co-occurrence of events or entities between languages and modalities often relies on a common mapping, e.g., a structured ontology, such as ACE (Walker et al., 2006) or ERE (Song et al., 2015). While our Machine-Aided Script Curator (MASC) does employ a structured ontology, it does not currently incorporate multi-modal or non-English sources. However, the limited ontology allows the event-sequencing background knowledge we encode to be used as a supplement to state-of-the-art information extraction systems, like OneIE (Lin et al., 2020) and DYGIE++ (Wadden et al., 2019), providing connections between otherwise disconnected extractions.

3 Overview of Script Creation

The curator initiates script creation by providing a name and description for the script and then enters, as text, the events in the script (Figure 1). Step entry is free-form, but we have noticed a tendency for curators to enter short, imperative sentences around a central agent's actions (e.g., *go to a car dealership*, *take a test drive*). Currently, script creation, unlike traditional annotation, is decoupled from any particular document. In cases where the curator is not familiar with a topic, we have used external resources to provide context (e.g., a Wikihow page open in a different window). In this setting, curation is akin to annotation that encourages the annotator to use both the material they read and prior knowledge.

The curators assign an ontology type to the main event in each step (e.g., *Movement* for both *go to a car dealership* and *take a test drive*). The ontology is configurable and can be replaced. We include a project-specific ontology with MASC's source code. When saved, scripts include both the curators' description and the selected ontology type (described in Section 4.1). This choice allows type decisions to be revisited if the ontology changes and limits the degree to which the small number of event types constrains the script's expressiveness. Downstream applications can choose whether to use the linguistic representation of the events or the normalized ontology types.

After the curators finish entering events, they encode connections between the events (Figure 2). There are two ways to connect events: the first, traditionally the focus of scripts, is temporal order; and the second is shared arguments (e.g., the same person is the agent of both *Movement* events *go to a car dealership* and *take a test drive*). To add sequential order, the curators enter pairwise *before* relations. Alternatively, they select multiple events and anchor them as coming before or after a single event. The latter method is convenient when the complete order is under-defined.[3] The curators add shared arguments to the script by selecting multiple events with the same argument, naming the argument (e.g., *buyer, seller* in Figure 2), and assigning an entity type (e.g., *PER* in Figure 2) and ontological role to each argument

[3]For example, after arriving at the car dealership, the potential buyer is likely to both walk around looking at cars and talk to a salesperson, but there is no defined order between the walking and talking.

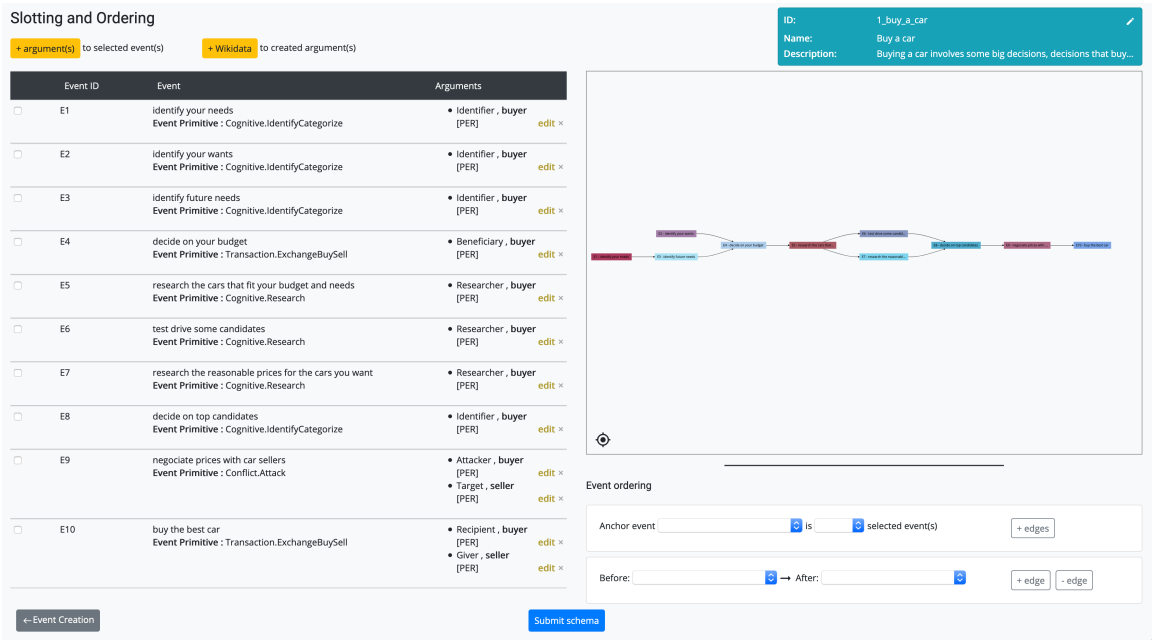

Figure 2: Adding details to events. For each event on the left, curators can add arguments. On the right side, curators can establish temporal order and visualize the script as an interactive graph.

(e.g., *Identifier, Researcher* in Figure 2). While this process is mostly manual, MASC uses the ontology's constraints to limit the available label options. In addition to project-specific entity types, MASC suggests links to the much larger set of types available using Wikidata entities (e.g., suggesting Q786803 for *car dealership*). These links provide a connection to an extensive knowledge graph and can provide additional information when the scripts are applied.

Finally, the curators review events that are automatically generated based on the manually entered description and initial script (described in Section 4.3). The suggestions can add intermediate steps that the curators may have missed, complete a script that was intentionally unfinished by the curator, or suggest alternative related paths (e.g., leasing instead of purchasing a car).

4 Suggestion Capabilities

To aid script creation, MASC incorporates three suggestion capabilities: suggestions for the ontological event type, suggestions for links to Wikidata, and suggestions for additional events to incorporate in the script. Below, we describe the models behind these capabilities and, for each model, report the accuracy using the five sample scripts created for this paper. Given the small sample size, the five sample scripts are best thought of as case studies, not a benchmark. Table 1 provides per-script analysis.

4.1 Event Type Classification

Each sub-event is ontologized with one of 41 event types through a semi-automated process. The ontology labels support connecting information to extraction engines and thus allow a script to provide potential event-event relations given information extraction output. Furthermore, the ontology labels provide language- and media-independent knowledge for identifying potential instances of the scripts.

There has been much work on automatic detection of event types (and triggers) in text (e.g., Bronstein et al. (2015); Lin et al. (2020); Peng et al. (2016)). Here, our input data (and goals) are slightly different. The ontology we use, while overlapping with ACE (Walker et al., 2006), introduces several new event types for which we do not have annotated training examples. Instead, the ontology provides a short definition and template for each event type. The curator's input events tend to be short imperative sentences with different linguistic characteristics than the text annotated in, e.g., ACE. Unlike standard information extraction, we need not identify a specific trigger phase.[4] Thus, we use a different approach to event labeling.

[4]Triggers are often used as a means to identify arguments of interest. But here, partly because of the telegraphic nature of the text entries, the arguments are often missing and, therefore, explicitly added.

To map from the curators' description of an event to the ontology, we use a version of Sentence-RoBERTa (Reimers and Gurevych, 2019)[5] to estimate the similarity of the curators' text input to the prose description of each action in the ontology. For example, for the user input *go to a car dealership*, the action description *Explicit mention of granting or allowing entry or exit from a location* receives the highest similarity score, and the corresponding action type *Movement.Transportation* becomes one of the recommendations. MASC suggests the three ontology actions most similar to the user's description. The user can accept one of the suggestions or pick a different type from the ontology (Figure 1, second column).

As mentioned earlier, the event type similarity depends on the ontology event type definitions and the event type templates. In preliminary experiments, we found using both together outperformed using either only the definitions or only the templates. While MASC's event type classification does not require training data, it depends on both the presence of templates and definitions in the ontology and their quality.

Performance on Case-Study Scripts The five scripts contain 58 events. We measure how often the model correctly predicts the event type that the curator selects. Accuracy of the top-1, -3, and -5 are 24, 48, and 55, respectively.[6] MASC presents the top-three suggestions to the curator; thus, accuracy at top-3 most closely relates to the curator's experience.

4.2 KGTK

In Section 2, we describe identifying the key repeating arguments of script events and labeling those arguments with their entity type and their role in each event using an ontology. That ontology provides only coarse distinctions between entities (e.g., a single category for facilities that does not distinguish *a car dealership* from *a school* or *a bank*). To support finer-grained distinctions and, in the future, leverage external knowledge sources, we incorporate connections to Wikidata[7] using KGTK (Ilievski et al., 2020). MASC's links aim to ground descriptive noun phrases (e.g., *car dealership*) in the large Wikidata ontology and do not require grounding specific, named entities (e.g., *Toyota*).

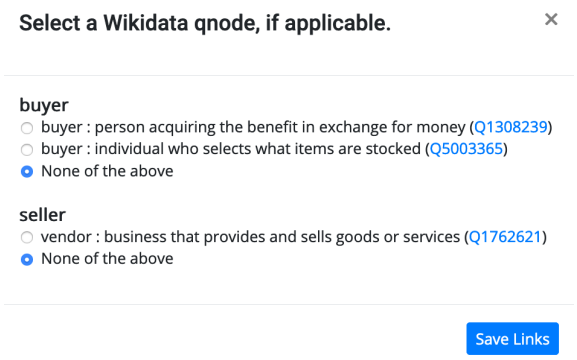

Figure 3: Reviewing the Wikidata link suggestions.

KGTK is an open-source toolkit that simplifies searching and interacting with various knowledge graphs, including Wikidata. KGTK provides a simple API for searching Wikidata entries, via Elasticsearch,[8] based on their titles and aliases (e.g., the Wikidata entry *motor car* also has the aliases *auto*, *automobile*, and *car*). KGTK also provides filtering functionality for candidate Wikidata entries. Since we are not interested in grounding specific named entities, we only return Wikidata entries representing Wikidata classes. Within MASC, KGTK allows users to link terms used in events to Wikidata. During argument creation, the curator provides a text label for each key argument. A background process then queries KGTK using the text label assigned to each argument. Candidates from KGTK are reranked using the Sentence-RoBERTa model to generate similarity scores between the label strings and the candidate Wikidata text descriptions. Before finishing a script, for each term in the script, the curator can select one of the candidates from KGTK or *None of the above* (Figure 3).

Performance on Case-Study Scripts. To evaluate entity linking, we treat the scripts created by the curators (and the mapping from the reference variables to Wikidata) as the labels. This is necessary since we do not have a ground-truth mapping from strings to Wikidata entities, and curators can use the same string to reference different entities. For example, *car* can refer to an automobile, a railway carriage, or a streetcar. The metric we use measures the ratio of reference variables linked to a specific Wikidata entity to the total number of

[5] Our Sentence-RoBERTa model is trained on more data. We use the two data sets in the original paper, SNLI (Bowman et al., 2015) and MNLI (Williams et al., 2018), and add the newer ANLI (Nie et al., 2020).

[6] The mean reciprocal rank (MRR, Radev et al., 2002) was 0.35 on the top three model predictions.

[7] https://www.wikidata.org

[8] https://www.elastic.co/elasticsearch/

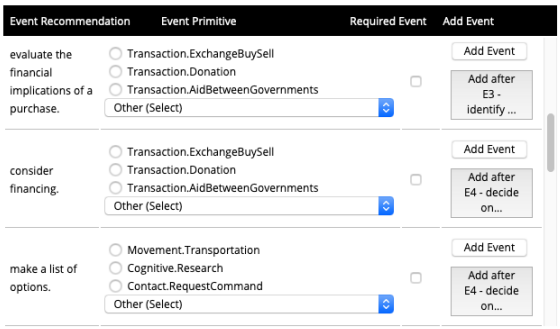

Figure 4: GPT-2 recommendations for *buying a car*.

reference variables used. We find that curators link 67% of the unique reference variables to Wikidata (e.g., *buyer* in Figure 3). We have not measured the ceiling on using Wikidata as an argument ontology. However, we suspect that refining the linking approach could yield more connections to Wikidata. Even at this low level of recall, at least a few concept-specific elements match for most scripts. In the future, these connection points could support script augmentation using common-sense and domain knowledge from Wikidata.

4.3 Event Recommendations

Since even the most experienced curators may overlook an action in an event script, we explored hypothesizing omitted events using GPT-2 (Radford et al., 2019) *without any fine-tuning*.

The first challenge is formulating input to GPT-2. We provide the title/name of the schema (e.g., *buying a car*), a description of the complex event (e.g., *Purchasing a car is a large investment that requires careful documentation and consideration of transportation requirements.*), and a request (e.g., *Describe steps of buying a car.*), followed by the first few events of the script. In the initial version, we used a form of the events as *First, Identify your needs. Then, Decide on your budget. Next, Identify car models you can afford.* However, a numerical formulation proved much more effective (e.g., *1. Identify your needs 2. Decide on your budget 3. Identify car models you can afford 4.*) and resulted in more coherent events.

To filter undesirable or redundant output, we pass GPT-2 outputs through a sequence of filters. We remove undesired strings characteristic of neural text generation, like empty strings (Stahlberg and Byrne, 2019), and outputs that are invalid in the context of schema creation: strings of less than two words and those with sequences of non-alphabetic characters. We address duplicated output, a considerable concern for GPT-2, especially given the short and similar inputs.[9] The filters eliminate strings with duplicates in the alternatives or the human-curated schema. To account for semantic duplicates, such as *go to dealership* and *go to the car dealership*, we use a variant of Gestalt Pattern Matching (Ratcliff and Metzener, 1988) through Python's *difflib*. For usability, we suggest at most 12 sub-events per script. Figure 4 shows the interface for reviewing event recommendations.

Performance on Case Studies. We measure the performance of GPT-2 recommendations in two ways. First, we generate recommendations for five scripts created by curators and ask the curators to accept relevant GPT-2 recommendations. We instruct curators to accept recommendations even if the recommended events represent alternative paths (or are semantically redundant). With these instructions, the curators accept 98% of GPT-2's recommendations. The high acceptance rate indicates that even with our simple setup for event recommendation using a language model, the system suggests domain-relevant events.

For the second evaluation, we instruct the curators to accept only those GPT-2 recommendations that add to their existing script. In other words, they only accept events that add details to the scripts or supply some missing information. We instruct curators to reject recommendations for alternative script scenarios. With these instructions, curators accept 23% of GPT-2's recommendations. This result illustrates the feasibility of supplementing human knowledge with generations from language models. Since MASC uses GPT-2 *after* the human felt the script was complete, the machine identifies events previously overlooked by the human.

Mixed-Initiative script curation. Given the success of GPT-2 recommendations after script curation, a natural next step is for curators to work with GPT-2 interactively. In the *mixed-initiative mode*, a curator specifies a script's name, definition, and first step. GPT-2 then suggests multiple options for the next step. The curator can use one of the suggestions, edit it, or ignore all the suggestions and manually input the next step. Every time the curator adds a step to the script, GPT-2 follows with suggestions for the next step. We found that

[9] GPT-2 often generates strings with a similar meaning, but lexically different, e.g., for a script on buying a car, it might generate *buy*, *buy the car*, and *purchase the car*. It is superfluous to show users all three suggestions.

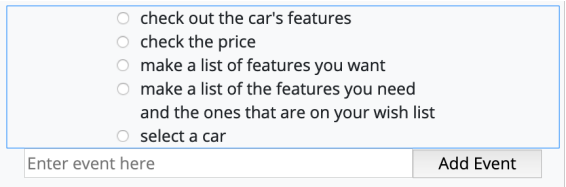

Figure 5: Mixed-Initiative: GPT-2's suggestions for the script *buying a car*, given the first step *Identify your needs*.

automated step generation took less than 3 seconds in the slowest case on modern hardware (NVIDIA GeForce RTX2080Ti).

To evaluate the effectiveness of mixed-initiative mode, we asked four curators to create a total of twelve scripts using the mode. We instructed the curators to accept event suggestions only when they are a natural continuation of the script. Out of GPT-2's 105 suggestion sets, the curators accepted an event from 50 sets (48% acceptance rate). In six more cases, the curators used a GPT-2 suggestion as a starting point and edited the suggestions to suit the script better. We found the mixed-initiative scripts to be just as comprehensive as the scripts detailed in Table 1, where GPT-2 suggested missing events only after the curators created an initial script.

5 Discussion and Future Work

With this demonstration system, we provide an approach to human-machine collaboration for building a repository of scripts. Having such a repository, for a diverse set of events, will allow us to investigate how procedural knowledge introduced to the AI community 40 years ago (Schank and Abelson, 1977) can be broadly applied. By facilitating the human creation of scripts, we can better understand what is required to develop automatic script discovery approaches.

While we have not yet created a large repository of scripts, we have created five scripts with which we start this analysis. The scripts cover topics with varying degree of "common knowledge": *Planning and Managing an Evacuation* (EVAC), *Ordering Food at a Restaurant* (FOOD), *Finding and Starting a New Job* (JOB), *Obtaining Medical Treatment* (MED), and *Corporate Merger or Acquisition* (MERGER). A single curator created these scripts, which we use to illustrate future directions for MASC and interesting properties of the scripts themselves. Having multiple curators for even a small number of scripts would provide insights into the diversity, prior knowledge, and level of detail a script author uses. In our analysis, we have seen that the scripts created with MASC encode knowledge that is uncommon in news-like data sets. For example, our curator included *sign confidentiality agreement* as an event in the script for a MERGER. While news frequently reports the final step of a merger, the full process is rarely described.

Table 1 summarizes the key characteristics of each of these scripts. They vary in (a) the number of steps initially created (row 1), with only 5 steps for MED and 16 for both EVAC and JOB; and (b) the time required for initial script creation (row 6). The script that took the longest was not the one with the most steps (or the most arguments). Instead, it was the domain that the curator knew the least about (and thus chose to research). For all five scripts, there were cases where the event type suggestions were correct, but for three of the five, MASC suggested the correct type less than half the time, suggesting that better automatic event typing could increase the curators' speed.

All scripts contain entities that play a role in multiple events (row 3, first and second numbers). For example, in EVAC, the evacuation manager plays some role in all events, while the evacuee plays a role in most but not all. While some arguments cannot be linked to Wikidata, all five scripts contain at least one argument that can be linked (row 3, last number). Future work could both improve linking accuracy and use Wikidata as a source of knowledge to provide additional context (and suggestions) to the curator.

While the prototypical script is a timeline with complete order between all pairs of events, we see sub-graphs with unordered steps in our data. Three of the five sample scripts display this behavior; for example, in JOB, *searching for open positions* and *notifying network that they are looking for a job* are unordered. The visualization of the schema in Figure 2 illustrates this pattern with no order between *E2* and *E3*.

MASC incorporates machine suggestions of unrecorded events. In four of the five scripts, the curator accepted at least one suggestion. Interestingly, the curator incorporated more suggestions for two events that one thinks of as everyday experiences (FOOD and MED) than they did for the script they were unfamiliar with (MERGER). This suggests that the recommendation functionality can be useful even in a familiar domain; by capturing

		EVAC	FOOD	JOB	MED	MERGER
1	# Events in initial script	16	9	16	5	12
2	Accuracy at top-1, 3, 5 for event types	25/44/50	11/33/67	13/44/44	20/60/60	50/67/67
3	# Entity instances, occurrences of those entities, and unique links to Wikidata	2/26/1	5/18/3	2/24/2	3/11/3	3/24/2
4	# Event suggestions selected for single script and all relevant (max. 12 per script)	4/8	3/12	0/11	5/12	2/12
5	Non-linear path	Y	Y	Y	N	N
6	Self-reported time	1.5 hrs	0.5 hr	1 hr	0.5 hr	2.5 hrs

Table 1: Characteristics of five sample scripts.

what the curator omits through forgetfulness or because they assume common knowledge. Further exploration of how a machine can aid a person whose knowledge may be incomplete or may forget to be explicit seems promising. Examples of possible research directions include incorporating suggestions from approaches that discover scripts (e.g., Rudinger et al. (2015); Weber et al. (2018, 2020)) and leveraging background knowledge (e.g., Wikidata).

6 Ethical Considerations

Many technological innovations require ethical considerations, even more so for those involving machine learning while also being a demonstration paper that provides working technology. Below we address the review questions raised in the NAACL Ethics Review Questions.[10]

Bias. The bias in generative language models has been well documented. In general, using a human-in-the-loop process means that rather than treating an automatically generated label or event as correct, we treat it as a suggestion that the curator can ignore. Still, the suggestions can influence the curator. Thus it is vital that the metrics reported in this paper be interpreted with an understanding of the potential for bias and any use of MASC account for bias.

MASC incorporates both a predefined ontology and the ability to link to an extensive external resource (Wikidata). Given the size of the predefined ontology is small, to apply MASC to a new domain, users would likely need to update the ontology. MASC's approach to aligning English descriptions to the ontology makes adding new event classes easy. Wikidata, while much larger and growing, is also subject to the bias of Wikidata's editors, their knowledge, and their choices about what to include. Wikidata over-represents some issues, while some socially important ones are under-represented or missing. Wikidata linking is optional; thus in a domain that is not well covered, a curator can skip the linking step or replace Wikidata with a domain-relevant resource.

The suggestion capabilities described in Section 4 use pretrained language models (GPT-2 and RoBERTa). The bias of these algorithms, measuring that bias, and mitigating it is an active area of work. Recent work has provided data sets for measuring bias (Nadeem et al., 2020) and meta-studies of the approaches taken to study and address bias (Blodgett et al., 2020). Much work has focused on bias as it impacts demographic groups. MASC focuses on events, not individuals. The publicly available GPT-2 models have learned from data that might not cover current events (e.g., GPT-2 was trained before the COVID-19 epidemic), represents only English dialects from the inner-circle (Dunn and Adams, 2020), and contains toxic language (Gehman et al., 2020). In our immediate context, we mitigate against the challenge presented by language model bias by requiring manual review of all automatically suggested output. If the ideas in this paper were extended to a fully automatic approach, language model and domain-specific studies of the impact of bias on LM-based suggestions would be necessary.

Data Set. To understand how the tool is used and future research directions, we created five sample scripts which we included in the supplementary material. These scripts provide interesting examples of what we could learn from a larger scale data set; however, they are not large enough themselves to serve as a new benchmark. The five scripts were created by full-time research staff compensated following US state and federal law. The scripts were created by a single individual and represent that individual's pre-existing knowledge (and their im-

[10] https://2021.naacl.org/ethics/review-questions/

plicit biases). To counter bias in a large-scale script repository, we recommend that the curator workforce is diverse and that any given activity is represented in scripts written by multiple people. Any released repository should have sufficient reporting about the data set creators to provide users with an understanding of data bias. The paper reports empirical results based on this five script sample. However, the paper acknowledges that the sample is small and treats these results as case studies for MASC, not a new benchmark.

Intended Use. The most immediate use of MASC is to create a repository of script information – either broadly available to researchers or within a specific research community. In some cases, e.g., the steps to plan a rescue operation, both the generation of the script and its application are generally understood as positive. In other cases, e.g., the steps in grooming an individual for human trafficking, the script's conclusion is negative, but understanding the process is necessary to prevent the activity. As AI's ability to discover and apply such knowledge increases, it will be necessary to regularly audit the use cases to ensure the focus remains a benefit to society. If the human-in-the-loop approaches used here were integrated into a fully automated system, further auditing of bias (and accuracy) would be necessary.

Compute Time and Power. Most of the models used for this demonstration are pretrained and publicly available. The pretraining and fine tuning described in Section 4.1 took less than 20 hours using a single GPU.

Acknowledgment

This material is based on research supported by DARPA under agreement number FA8750-19-2-0500. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government.

References

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5476, Online. Association for Computational Linguistics.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Ofer Bronstein, Ido Dagan, Qi Li, Heng Ji, and Anette Frank. 2015. Seed-based event trigger labeling: How far can event descriptions get us? In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 372–376, Beijing, China. Association for Computational Linguistics.

Nathanael Chambers. 2013. Event schema induction with a probabilistic entity-driven model. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Seattle, Washington, USA. Association for Computational Linguistics.

Asaf Degani and Earl L. Wiener. 1993. Cockpit checklists: Concepts, design, and use. *Human Factors*, 35(2):345–359.

Jonathan Dunn and Ben Adams. 2020. Geographically-Balanced Gigaword Corpora for 50 Language Varieties. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 2521–2529, Marseille, France. European Language Resources Association.

Atul Gawande. 2010. *The checklist manifesto: how to get things right*. Metropolitan Books, New York.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369, Online. Association for Computational Linguistics.

H Paul Grice. 1975. Logic and conversation, syntax and semantics. *Speech Acts*, 3:41–58.

Filip Ilievski, Daniel Garijo, Hans Chalupsky, Naren Teja Divvala, Yixiang Yao, Craig Rogers, Ronpeng Li, Jun Liu, Amandeep Singh, Daniel Schwabe, and Pedro Szekely. 2020. KGTK: A toolkit for large knowledge graph manipulation and analysis. In *The Semantic Web – ISWC 2020*, pages 278–293, Cham. Springer International Publishing.

Manling Li, Alireza Zareian, Ying Lin, Xiaoman Pan, Spencer Whitehead, Brian Chen, Bo Wu, Heng Ji, Shih-Fu Chang, Clare Voss, Daniel Napierski, and Marjorie Freedman. 2020. GAIA: A fine-grained

multimedia knowledge extraction system. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 77–86, Online. Association for Computational Linguistics.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. KagNet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2829–2839, Hong Kong, China. Association for Computational Linguistics.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009, Online. Association for Computational Linguistics.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2020. StereoSet: Measuring stereotypical bias in pretrained language models.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Haoruo Peng, Yangqiu Song, and Dan Roth. 2016. Event detection and co-reference with minimal supervision. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 392–402, Austin, Texas. Association for Computational Linguistics.

Dragomir R. Radev, Hong Qi, Harris Wu, and Weiguo Fan. 2002. Evaluating web-based question answering systems. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC'02)*, Las Palmas, Canary Islands - Spain. European Language Resources Association (ELRA).

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

John W Ratcliff and David E Metzener. 1988. Pattern-matching - the gestalt approach. *Dr Dobbs Journal*, 13(7):46.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Rachel Rudinger, Pushpendre Rastogi, Francis Ferraro, and Benjamin Van Durme. 2015. Script induction as language modeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1681–1686, Lisbon, Portugal. Association for Computational Linguistics.

Roger C Schank and Robert P Abelson. 1977. *Scripts, plans, goals, and understanding: An inquiry into human knowledge structures*. Lawrence Erlbaum Associates.

Vered Shwartz, Peter West, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Unsupervised commonsense question answering with self-talk. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4615–4629, Online. Association for Computational Linguistics.

Zhiyi Song, Ann Bies, Stephanie Strassel, Tom Riese, Justin Mott, Joe Ellis, Jonathan Wright, Seth Kulick, Neville Ryant, and Xiaoyi Ma. 2015. From light to rich ERE: Annotation of entities, relations, and events. In *Proceedings of the The 3rd Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, pages 89–98, Denver, Colorado. Association for Computational Linguistics.

Felix Stahlberg and Bill Byrne. 2019. On NMT search errors and model errors: Cat got your tongue? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3356–3362, Hong Kong, China. Association for Computational Linguistics.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. ACE 2005 Multilingual Training Corpus LDC2006T06. Web Download. Philadelphia: Linguistic Data Consortium.

Noah Weber, Rachel Rudinger, and Benjamin Van Durme. 2020. Causal inference of script knowledge. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7583–7596, Online. Association for Computational Linguistics.

Noah Weber, Leena Shekhar, Niranjan Balasubramanian, and Nathanael Chambers. 2018. Hierarchical quantized representations for script generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages

3783–3792, Brussels, Belgium. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. 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 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.