What do Large Language Models Learn about Scripts?

Abhilasha Sancheti
University of Maryland, College Park
sancheti@umd.edu

Rachel Rudinger
University of Maryland, College Park
rudinger@umd.edu

Abstract

Script Knowledge (Schank and Abelson, 1975) has long been recognized as crucial for language understanding as it can help in filling in unstated information in a narrative. However, such knowledge is expensive to produce manually and difficult to induce from text due to reporting bias (Gordon and Van Durme, 2013). In this work, we are interested in the scientific question of whether explicit script knowledge is present and accessible through pre-trained generative language models (LMs). To this end, we introduce the task of generating full event sequence descriptions (ESDs) given a scenario in the form of natural language prompts. In zero-shot probing experiments, we find that generative LMs produce poor ESDs with mostly omitted, irrelevant, repeated or misordered events. To address this, we propose a pipeline-based script induction framework (SIF) which can generate good quality ESDs for unseen scenarios (e.g., bake a cake). SIF is a two-staged framework that fine-tunes LM on a small set of ESD examples in the first stage. In the second stage, ESD generated for an unseen scenario is post-processed using RoBERTa-based models to filter irrelevant events, remove repetitions, and reorder the temporally misordered events. Through automatic and manual evaluations, we demonstrate that SIF yields substantial improvements (1-3 BLUE points) over a fine-tuned L.M. However, manual analysis shows that there is great room for improvement, offering a new research direction for inducing script knowledge.

1 Introduction

Scripts are structured commonsense knowledge in the form of event sequences that characterize commonplace scenarios, such as, eating at a restaurant (Schank and Abelson, 1975). Scripts are fundamental pieces of commonsense knowledge that humans share and assume to be tacitly understood by each other. When someone says “I went to a

restaurant for lunch”, our script knowledge allows us to infer that a waiter would have taken the order, the speaker would have eaten the lunch, payed for it, and tipped the waiter, even if these events are not explicitly mentioned. Knowledge of scripts, whether implicit or explicit, has been recognized as important for language understanding tasks (Miikkulainen, 1995; Mueller, 2004).

Earlier efforts to automatically induce scripts from text on a large scale include Chambers and Jurafsky (2008) who treat the problem of script induction as one of learning narrative chains using textual co-occurrence statistics. However, reporting bias (Gordon and Van Durme, 2013) remains an obstacle for script induction as many events are not explicitly mentioned. Knowledge of scripts, whether implicit or explicit, has been recognized as important for language understanding tasks (Miikkulainen, 1995; Mueller, 2004).

With the success of pre-trained LMs (Devlin et al., 2018), these methods have been applied to a variety of domains and tasks, including language understanding and generation. However, the effectiveness of these methods in capturing script knowledge has not been thoroughly evaluated.

Figure 1: Sample event sequence description (ESD) from Wanzare et al. (2016) for BAKING A CAKE scenario. We use natural language prompts (Table 2) to generate completely ordered ESDs for evaluating extent of script knowledge accessible through LMs.
et al., 2018; Liu et al., 2019; Radford et al., 2019) in various natural language understanding tasks, we are interested in investigating the extent and accessibility of explicit script knowledge present in pre-trained LMs. In this work, unlike cloze-based script evaluations (Chambers and Jurafsky, 2008; Mostafazadeh et al., 2016) which LMs are uniquely optimized for (Rudinger et al., 2015), we evaluate pre-trained LMs on the ability to fully generate event sequence descriptions (ESDs) (Regneri et al., 2010) in free-form natural language (Figure 1). This is a challenging task as scripts are complex structures with varied granularity of describing a scenario (e.g. starting from going to grocery store to buy ingredients or starting with finding a recipe for BAKING A CAKE scenario), and the requirement to produce all the scenario-relevant events in the correct temporal order.

To this end, we first probe LMs via carefully crafted prompts to analyze the quality of ESDs generated in a zero-shot setting (§3) and find that the generated ESDs are of poor quality with many scenario-irrelevant, repeated, temporally misordered, and missing events. To address this we propose a, LM-agnostic, pipeline-based script induction framework (§4), SIF, which can generate good quality ESDs for novel scenarios that LM has not seen during the training phase of the framework. SIF is a two-staged framework with fine-tuning LM on a small set of ESDs as the first stage followed by a three-stepped post-processing stage which corrects the ESDs generated from a fine-tuned LM for irrelevant, repeated, and temporally misordered events. This work makes the following contributions: (1) present an analysis of the extent of script knowledge accessible through LMs using probing techniques, in a zero-shot setting (§3) and find that the generated ESDs are of poor quality with many scenario-irrelevant, repeated, temporally misordered, and missing events. To address this we propose a, LM-agnostic, pipeline-based script induction framework (§4), SIF, which can generate good quality ESDs for novel scenarios that LM has not seen during the training phase of the framework. SIF is a two-staged framework with fine-tuning LM on a small set of ESDs as the first stage followed by a three-stepped post-processing stage which corrects the ESDs generated from a fine-tuned LM for irrelevant, repeated, and temporally misordered events. This work makes the following contributions: (1) present an analysis of the extent of script knowledge accessible through LMs using probing techniques, in a zero-shot setting, by introducing a new task of generating full ESDs from natural language prompts; (2) propose script induction framework that can generate ESDs for held-out as well as novel scenarios drawn from a different distribution; and (3) present automatic and manual evaluation of the generated ESDs, establishing the viability of our framework and paving way for future research in this direction.

2 Related Work

Narrative Chain Induction There has been a growing body of research into statistical script learning systems which can automatically infer implicit events from text. Seminal work by (Chambers and Jurafsky, 2008, 2009) describe a number of simple event co-occurrence based systems which infer (verb, dependency) pairs (known as narrative events) with partial-ordering related to one or multiple participants in discourse (known as narrative chains). They also introduce narrative cloze task, where a model is expected to predict one removed narrative event, given all the other narrative events. However, much of the information about events is usually left implicit in texts. Moreover, narrative events are highly abstracted (Ostermann, 2020) and cloze task is insufficient to evaluate script knowledge (Chambers, 2017). Therefore efforts have been made to acquire crowdsourced ESDs (Singh et al., 2002; Regneri et al., 2010; Modi et al., 2017; Wanzare et al., 2016; Ostermann et al., 2018, 2019) and to learn similar events in a scenario using unsupervised (Regneri et al., 2010) and semi-supervised (Wanzare et al., 2017a) approaches.

Temporal Ordering and Relevance Previous works (Modi and Titov, 2014; Wanzare et al., 2017b; Lyu et al., 2020) have investigated induction or prediction of temporal ordering of prototypical events. Others have predicted next (Pichotta and Mooney, 2016) or related (Lyu et al., 2020) events in natural language form. Zhou et al. (2019) acquire commonsense procedural knowledge directly from natural language source, like wikiHow. by learning representations for scenarios and events which are predictive of both relevance of event to the scenario and temporal ordering. Zhang et al. (2020) propose a non-learning based approach to predict fixed-length events given an unseen scenario and related scenarios with their events. A recent contemporaneous work (Sakaguchi et al., 2021) generate partially-ordered scripts using pre-trained LMs by predicting events and edges for partial-order.

Knowledge-acquisition from Pre-trained LMs With the success of pre-trained LMs (Devlin et al., 2018; Liu et al., 2019; Radford et al., 2019) in various natural language understanding tasks, a number of works investigate how commonsense knowledge is captured in these models (Feldman et al., 2019; Petroni et al., 2020; Weir et al., 2020; Shwartz et al., 2020). Successful efforts have been made to induce relational (Bouraoui et al., 2020), numerical (Lin et al., 2020), temporal (Zhou et al.,
that occur when you bake a cake:

here is an ordered sequence of events that occur when you bake a cake:

when you bake a cake:

happen while baking a cake:

these are the things that happen when you bake a cake:

describe baking a cake in small sequences of short sentences:

Figure 2: Different prompt formulations for BAKING A CAKE scenario for probing. 16 prompts are created by combining a prompt beginning with a continuation.

2020) and commonsense knowledge in pre-trained LMs using fine-tuning.

Unlike previous works, we focus on investigating the extent and accessibility of script knowledge through pre-trained LMs via probing techniques and inducing such knowledge in them using a pipeline-based framework to generate full event sequence descriptions for novel scenarios in free-form natural language.

3 Probing for Script Knowledge

We first design a zero-shot probing experiment to evaluate pre-trained LMs’ ability to generate ESDs by carefully selecting natural language prompts, which LMs are known to be sensitive to (Bouraoui et al., 2020). We experiment with 16 manually crafted prompts1 (Table 2) with different phrasing and levels of conditioning to enquire large versions of GPT2, BART, and T5 for script knowledge. The intuition behind these prompts is similar to asking questions (prompts) to a knowledge source in various ways to get the required answer (ESD for a scenario).

BART and T5 were not able to output anything except the input prompt or start, end, and pad tokens hence we only present qualitative outputs from GPT2, when probed with various prompts for BAKING A CAKE scenario, in Table 2. We observe that the quality of generated ESDs vary for different prompts. Although GPT2 is able to generate some scenario-relevant events with just the prompt beginnings and no continuations (e.g. 1 and 2 in Table 2), the ESDs are incomplete with many auxiliary details, and incorrect event ordering (e.g. ‘3. The cake is served at the table’ before ‘6. The cake is transferred to the oven.’ in 2). It sometimes outputs (e.g. 4) narrations rather than procedural descriptions. As generation from scratch is an open-ended task, we use prompt with a number to guide GPT2 to generate a procedural script. Although 4 and 5 are more procedural, the events are still at a coarse-grained level with most of the intermediate events missing. To further guide the generation towards a fine-grained level, we condition the generation on a few events along with the prompt beginning. This helps us in examining whether GPT2 has temporal knowledge about the events related to a scenario. Conditioning on the events results in a better quality ESD (e.g. 6, 7, 8). However, there is a repetition of events (‘let it cool for another 10 minutes’ in 6, ‘add in your flour and mix by hand’ in 7) in addition to wrong event ordering, irrelevant (e.g. ‘is it hot?’ in 8) and missing events. As GPT2 produces poor quality ESDs in this zero-shot setting with BART and T5 not even been able to output any events, we propose a script induction framework detailed in the following section.

4 Script Induction Framework (SIF)

In this section, we provide details on our pipeline-based script induction framework, SIF (Figure 3), which addresses the limitations of zero-shot ESD generation. SIF is a two-staged framework which fine-tunes LM on a small set of ESDs in the first stage. In the second stage, ESDs generated using the fine-tuned LM are passed through a sequence of RoBERTa-based classifiers (Liu et al., 2019) to identify relevant events, remove redundant events, and predict pair-wise temporal ordering between the events. These pair-wise orderings are then used to create a full event ordering using topological sorting on a directed graph created from the predicted orderings.

---

1We also experiment with capitalized prompts but they had no impact on the quality of generations.
We sample ESDs for an unseen scenario using a negative log-likelihood objective. For encoding ESDs (Table 2) to study the effect of edit distance between the events and an event pre-token. We use a mix. Errors are really really good. A great idea is to invest in one of these mixers, little portable ones.

Table 1: Scripts generated from GPT2-L for BAKING A CAKE scenario with bold-faced prompts.

Table 2: Different prompt formulations for BAKING A CAKE scenario with two events (e1 and e2).

4.1 Stage I: Fine-tuning Pre-trained LMs

Pre-trained LMs fine-tuned on commonsense datasets like ATOMIC (Sap et al., 2019) can generalize beyond the scenarios observed during fine-tuning (Bosselut et al., 2019). Hence, we investigate the learning capability of LMs when a small number of script examples are available. We fine-tune LMs on ESDs using different natural language and pseudo-natural language prompt formulations for encoding ESDs (Table 2) to study the effect of prompt formulations on this task as observed during the probing experiments. We fine-tune LMs using negative log-likelihood objective.

4.2 Stage II: Post-processing Generated Scripts

We sample ESDs for an unseen scenario using the fine-tuned LMs and employ a 3-step post-processing method to correct them for relevance, repetitions, and ordering.
4.2.3 Step 3: Temporal Order Correction

The final step is to correct the order of events in an ESD. We correct the ESDs for ordering by first obtaining pair-wise event orderings and then using a graph-based approach to get the final ordering. We pose the problem of pair-wise event ordering as a binary classification task to predict if the order of a given pair of events is correct with respect to the given scenario. We sample event pairs from gold ESDs to construct positive (sequence order) and negative (reverse order) examples to train a RoBERTa-L-based classifier. Topological sort is then used to get the final ESD for a scenario from the ordering predictions for all the \( \binom{n}{2} \) pairs of events in an ESD. We construct a directed graph \( G = (V, E) \) of events in a scenario with events as nodes \( V \) of the graph and a directed edge from node \( v_1 \in V \) to \( v_2 \in V \) if event represented by \( v_2 \) is predicted to occur after the event represented by \( v_1 \). We keep the original ordering of events in case the constructed graph is cyclic\(^2\) due to incorrect predictions from the classifiers.

4.3 Implementation Details

4.3.1 Dataset pre-processing

We fine-tune LMs on ESDs from DeScript (Wan-zare et al., 2016) dataset which consists of 100 ESDs each for 40 scenarios, collected via crowdsourcing. The scenarios are randomly partitioned into 8 folds with each fold consisting of ESDs from 5 scenarios to perform 8-fold cross-validation of SIF for each of the prompt formulation. We lowercase and enclose each ESD within a begin of scenario \(<BOS>\) and an end of scenario \(<EOS>\) token for fine-tuning. The input to the relevance classifier is: scenario \(<s>\) e and to the temporal classifier is scenario name \( <s> e_1 <s> e_2 \), where \(<s>\) is a separator token and \( e, e_1, e_2 \) are events.

4.3.2 Training details

We use huggingface’s transformers library (Wolf et al., 2020) to fine-tune LMs on each of the 7 prompt formulations, leading to 7 variations for each LM, for 1 epoch with a batch size of 1, gradient accumulation per 16 steps, and block size of 150. At inference time, 5 ESDs are sampled for each of the given scenarios with top 50 probable tokens, nucleus sampling (Holtzman et al., 2019) probability of 0.9, and maximum length set at 150. We use RoBERTa-L architecture from the transformers library for relevance and temporal order classifiers. Relevance (Temporal) classifier is trained for 10 (5) epochs with average validation accuracy of 84.50% (83.87%) across the folds. The model with the best accuracy on the valid split is used in the post-processing stage. We use python’s editdistance library to compute edit distance for the de-duplication step. We use Adam optimizer with an initial learning rate of \( 2e^{-5} \), warm-up steps set at 0.06 of total steps, batch size of 16, and maximum input length 150 for both the classifiers. All the models are trained and tested on NVIDIA Tesla V100 SXM2 16GB GPU machine.

5 Evaluation

We use SIF to induce script knowledge in GPT2, BART, and T5, and evaluate full ESDs generated for a given unseen scenario using BLEU metric (Papineni et al., 2002), following Pichotta and Mooney (2016) who use BLEU to score individual LM-generated events. As BLEU is a precision-based metric, we measure n-gram overlap of the sampled ESDs against multiple gold-reference ESDs\(^3\) for each scenario in the test fold.

Additionally, for deeper analysis of the generated ESDs, two of the authors evaluate a subset of the generated ESDs (blinded to the identity of the models and prompt variants) on three levels – individual events (Relevance \( R \)), pairwise events (Order \( O \)), and the overall sequence (Missing \( M \)). \( R \) measures the % of generated events relevant to a scenario; \( O \) measures the % of consecutive event pairs correctly ordered given a scenario; and \( M \) measures the degree to which important events are missing on a 4-point Likert scale defined as (1) no or almost no missing events, (2) some insignificant missing events, (3) notable missing events, and (4) severe missing events. As scripts are complex structures and require an understanding of scenarios, we chose not to resort to a crowdsourcing platform for manual analysis. We manually analyze the outputs to evaluate SIF as well as perform an error analysis to identify opportunities for future research directions.

We evaluate our framework on scenarios in each of the eight folds as well as novel scenarios from\(^3\) We use NLTK python library to calculate BLEU score with add-1 smoothing function and n-grams up to \( n = 4 \). We convert the outputs of different variants and gold references into numbered form, 1. \( e_1 \) 2. \( e_2 \) . . . \( e_n \) for a fair comparison.

\(^2\)66±15% (averaged across all the input variants and folds) of the complete graphs are acyclic for GPT2
Table 3: Automatic evaluation results: Mean BLEU scores (and std. dev.) over 8 folds of held-out scenarios are reported. (1) is pre-trained GPT2 (no fine-tuning or post-processing); (2) is randomly initialized GPT2 with fine-tuning; (3-4) are fine-tuned BART and GPT2; (5-6) are SIF applied to BART and GPT2.

| Models            | TOKENS | EXPECT | SEQUENCE | ALLTOKENS | DESCRIBE | DIRECT | ORDERED |
|-------------------|--------|--------|----------|-----------|----------|--------|---------|
| (1) Zero-shot     | 03.1   | 03.6   | 05.4     | 03.1      | 03.2     | 03.9   | 06.2    |
| (2) GPT2-SCRATCH  | 17.2   | 19.3   | 16.8     | 16.8      | 16.8     | 16.8   | 16.8    |
| (3) BART-FT       | 15.5   | 20.8   | 19.6     | 19.7      | 19.2     | 18.9   | 18.0    |
| (4) GPT2-FT       | 30.7   | 31.3   | 32.4     | 30.7      | 32.3     | 31.4   | 31.0    |
| (5) BART-SIF      | 30.8   | 21.1   | 19.9     | 20.3      | 20.0     | 19.6   | 18.7    |
| (6) GPT2-SIF      | 33.6   | 33.9   | 35.2     | 32.5      | 34.3     | 33.0   | 33.2    |

Table 4: Ablation analysis of each step in the proposed pipeline for GPT2. Mean BLEU scores (and std. dev.) over 8 folds of held-out scenarios are reported. (1) fine-tuned GPT2; (2-4) are fine-tuned GPT2 with successive post-processing steps.

| Models            | TOKENS | EXPECT | SEQUENCE | ALLTOKENS | DESCRIBE | DIRECT | ORDERED |
|-------------------|--------|--------|----------|-----------|----------|--------|---------|
| (1) GPT2-FT       | 30.7   | 31.3   | 32.4     | 30.7      | 32.3     | 31.4   | 31.0    |
| (2) GPT2-FT+Relevance (R) | 33.1 | 33.1   | 34.7     | 31.9      | 33.7     | 32.6   | 33.2    |
| (3) GPT2-FT+De-duplicate (D) | 33.5 | 33.6   | 35.1     | 32.1      | 34.3     | 32.9   | 33.6    |
| (4) GPT2-FT+R+De-duplicate (D) | 33.6 | 33.9   | 35.2     | 32.5      | 34.2     | 33.0   | 33.2    |

6 Results and Analysis

6.1 Automatic Evaluation

We present the automatic evaluation results on held-out scenarios in Table 3. As baselines, we report scores from non-fine-tuned GPT2-L (Zero-shot), a randomly-initialized GPT2-L-SCRATCH model fine-tuned on DeScript ESDs, and BART-FT and GPT2-FT models which are fine-tuned in the first stage of SIF. We do not report any results for T5 as it was even struggling to learn the input ESD formulations during fine-tuning. We explain the findings from automatic evaluation below.

**SIF significantly outperforms fine-tuning baselines** Both GPT2-SIF and BART-SIF have higher BLEU scores as compared to their corresponding fine-tuned (GPT2-FT and BART-FT) models across all the prompt variants. This clearly reflects the advantage of the post-processing stage in SIF framework. Improvement across different LMs reinforces the LM-agnostic nature of our framework. Variation in the extent of induction across prompt variants indicates the sensitivity of LMs to prompt formulations.

**Script knowledge is best accessible through GPT2 than other LMs** As previously mentioned in probing experiments, BART and T5 were not able to output anything useful in the zero-shot setting while GPT2 could produce ESDs, although erroneous and of poor quality. We observe same trends even after fine-tuning these LMs or using SIF to induce script knowledge in these LMs. Interestingly, a randomly initialized and fine-tuned GPT2 (GPT2-L-SCRATCH) is able to perform comparable to a pre-trained BART fine-tuned using DeScript (BART-FT), and even better for TOKENS and ORDERED variants. Overall, GPT2 is found to be better than BART in terms of the presence and accessibility of script knowledge through them. One possible explanation for this is that GPT2 is a generative language model while BART and T5 are encoder-decoder-based language models making it challenging to encode complete script knowledge within a scenario name.

**Performance across LMs is sensitive to prompt formulation and scenario** We consistently observe variation in performance across prompt variants. Moreover, this variation is also observed across LMs. For BART, EXPECT outperforms other prompt variants while SEQUENCE performs the best for GPT2. High variance across folds also shows that different prompts perform differently depending upon a scenario. This indicates the sensitivity of LMs to prompt formulations and thus justifies our experiments with different prompt formulations to study the extent of script knowledge that can be accessed through pre-trained LMs.

6.2 Ablation Analysis of SIF

We next analyze the contribution of each the stage of SIF and each step of stage II leading to improvement in the performance via an ablation study, on GPT2, in Table 4. As expected stage I contributes...
We manually evaluate a total of 652 variants (Table 5). BLEU scores are also reported for a stratified sample of outputs (one ESD per scenario across two folds). Mean scores across two annotators are reported. Annotator agreement is measured with Cohen’s Kappa (Cohen, 1960) ($\kappa=0.61$ for O, $\kappa=0.56$ for R) and Spearman’s correlation ($\rho=0.64$ for M). Underline and bold denotes the best across variants, and between FT and Ours, respectively. O scores are calculated only when both the events are marked as relevant by the two annotators.

### 6.3 Manual Evaluation and Error Analysis

We manually evaluate a total of 140 ESDs (for M) comprising 652 individual events (for R) and 582 consecutive pair of events (for O) generated from GPT2-FT and GPT2 SIF across all the prompt variants (Table 5). BLEU scores are also reported for the same set of ESDs to study the correlation between manual and automatic metrics. We find that outputs from SIF have higher BLEU, R, and O scores than FT across all prompt variants (except O for DIRECT and BLEU for ORDERED). M scores do not change, which shows that significantly important events are not dropped during the irrelevant events removal step. Different prompts perform well in different aspects. DESCRIBE generates most relevant events, ALLTOKENS has the best temporal ordering knowledge, and SEQUENCE leads to least severe missing events after Stage II of SIF. To our surprise, we find no statistically significant correlation between BLEU and any of the manual evaluation metrics (pearson correlation between BLEU and R, O and M was $r = 0.23, -0.06, -0.49$ with $p>0.1$, respectively), emphasizing a need for more sophisticated automatic metrics than BLEU for evaluating full ESDs, having a complex structure.

The best performing variant as per BLEU score differs from the best one in Table 3 due to variance in performance across scenarios as well as different sampled ESDs of the same scenario in Table 5.

During the manual evaluation, we observed that a model can miss significant events, even while generating many relevant ones. As we only de-duplicate multiple occurrences of exactly the same events in a scenario, we observe repeated paraphrases (4.6% across all prompt variants) of the same event, such as ‘pour some milk in the pot’ and ‘pour the milk into the coffee pot’ (MAKING COFFEE scenario). 23.9% of the irrelevant events (13.5% across all prompt variants) are incoherent (‘take the flat to the bathroom’ for CLEANING A FLAT), 11.4% mixed (‘sit in front of coffee shop’ for MAKING COFFEE), 61.4% unrelated (‘add shampoo’ for WASHING DISHES), and rest ungrammatical.

We present a manual evaluation of novel scenarios to gauge the generalizability of our framework in Table 6. The framework generalizes to most of the novel scenarios except for those which involve very granular events like MAKING GINGER PASTE or TYING SHOE LACES. Although GPT2 is a contextualized model, it confuses BUYING FROM VENDING MACHINE with buying from a store, SURFING THE INTERNET with the ‘surfing’ activity, or ATTENDING A WEDDING with ‘getting married’. Additionally, we provide a few good and bad quality outputs from GPT2 models for held-out (Table 7) and novel (Table 8) scenarios to identify the avenues for improving script induction in LMs. Qualitative outputs from BART are presented in Table 10 in the appendix.

### Table 5: Manual and BLEU scores on fine-tuned GPT2 (GPT2-FT) SIF applied to GPT2 (FT/SIF), computed for a stratified sample of outputs (one ESD per scenario across two folds). Mean scores across two annotators are reported. Annotator agreement is measured with Cohen’s Kappa ($\kappa=0.61$ for O, $\kappa=0.56$ for R) and Spearman’s correlation ($\rho=0.64$ for M). Underline and bold denotes the best across variants, and between FT and Ours, respectively.

| Variants | BLEU | Manual Evaluation |
|----------|------|--------------------|
|          | R1  | O1    | M1   |
| TOKENS   | 19.2/22.8 | 77.3/84.3 | 72.3/89.3 | 2.6/2.6 |
| EXPECT   | 22.8/26.0 | 81.9/82.7 | 74.5/86.5 | 3.0/3.0 |
| SEQUENCE | 27.3/33.4 | 73.3/83.2 | 74.0/87.5 | 2.5/2.5 |
| ALLTOKENS| 31.5/35.0 | 83.5/85.7 | 82.7/89.5 | 2.6/2.6 |
| DESCRIBE | 27.1/28.6 | 80.7/86.3 | 83.9/85.9 | 2.8/2.8 |
| DIRECT   | 30.9/34.1 | 81.2/84.2 | 88.5/86.1 | 2.6/2.6 |
| ORDERED  | 31.9/31.5 | 82.9/86.2 | 78.6/86.8 | 2.6/2.6 |

### Table 6: Manual evaluation of ESDs for novel scenarios. Averaged across 5 sampled ESDs per scenario generated using the best performing SEQUENCE variant of GPT2-SIF as per automatic measure.

| Scenario | R1  | O1    | M1   |
|----------|------|--------|------|
| Order fastfood online | 81.5 | 84.6 | 2.6 |
| Cook in a microwave | 89.5 | 92.0 | 2.4 |
| Answer telephone | 65.5 | 91.7 | 2.0 |
| Buy from vending machine | 77.1 | 81.3 | 3.4 |
| Tie shoe laces | 65.8 | 66.7 | 3.6 |
| Brush teeth | 75.9 | 71.4 | 2.6 |
| Make ginger paste | 41.5 | 85.7 | 3.4 |
| Attend a wedding | 71.9 | 100.0 | 2.4 |
| Wash a car | 85.7 | 90.0 | 3.0 |
| Take out trash | 88.5 | 92.3 | 2.2 |
| Take a taxi | 85.7 | 76.2 | 2.0 |
| Surf the internet | 73.3 | 62.5 | 2.8 |
| Watch television | 77.4 | 73.7 | 3.0 |
| Go to a club to dance | 100.0 | 93.5 | 1.4 |

Average Score | 77.1 | 85.0 | 2.6 |
## Table 7: Scripts generated using \textsc{sequence} variant of GPT2 for held-out scenarios. FT denotes output from the fine-tuned model and SIF refers to outputs from our framework applied to GPT2.

| Scenario | Good quality generation | Poor quality generation |
|----------|-------------------------|-------------------------|
| **GOING ON A TRAIN** | PT 1. get dressed 2. go to station 3. buy ticket 4. go on train 5. wait for train 6. get on train 7. sit in seat 8. read newspaper 9. wait for train 10. get off train 11. get dressed 12. leave station | SIF 1. go to station 2. buy ticket 3. wait for train 4. get on train 5. sit in seat 6. get off train 7. leave station |
| **FLYING IN AN AIRPLANE** | PT 1. go to airport 2. get boarding pass 3. go to the airline counter 4. present boarding pass and boarding pass id to attendant 5. present ticket and id to attendant 6. sit down in seat 7. make sure you are in seat 8. board airplane | SIF 1. go to airport 2. get the airline counter 3. present ticket and id to attendant 4. present boarding pass and boarding pass id to attendant 5. get boarding pass 6. board airplane 7. make sure you are in seat 8. sit down in seat |
| **CLEANING UP** | PT 1. get the flat 2. take the flat to the bathroom 3. take some water 4. take the flat to the sink 5. turn on the water 6. let it wash the flat 7. then put in some dryer sheet 8. get a sponge 9. scrub the flat 10. put the flat in the dryer 11. when done dry the flat with a towel 12. put away the flat | SIF 1. get the flat 2. take the flat to the sink 3. turn on the water 4. let it wash the flat 5. when done dry the flat with a towel 6. put away the flat |
| **TAKE THE UNDERGROUND** | PT 1. drive at the designated location 2. get in car 3. park in car 4. walk into the designated area 5. enter the room 6. find a chair 7. sit on a chair 8. watch movie | SIF 1. walk into the designated area |

| Scenario | Good quality generation | Poor quality generation |
|----------|-------------------------|-------------------------|
| **BUYING FROM A VENDING MACHINE** | PT 1. schedule an appointment for the reception 2. drive to the venue 3. show your guest information 4. present your wedding ceremony/reception to the guests 5. invite the family/friends/etc. to the wedding 6. present your wedding present to the guests 7. give the invitation 8. take the invitation 9. go to the reception 10. present the invitation 11. give the ticket 12. pay the fee/ gratuity 13. give the guest his/her ticket 14. leave the venue | SIF 1. schedule an appointment for the reception 2. drive to the venue 3. show your guest information 4. present your wedding ceremony/reception to the guests 5. invite the family/friends/etc. to the wedding 6. present your wedding present to the guests 7. give the invitation 8. take the invitation 9. go to the reception 10. present the invitation 11. give the ticket 12. pay the fee/ gratuity 13. give the guest his/her ticket 14. leave the venue |
| **FLYING IN AN AIRPLANE** | PT 1. get your hot water 2. get your bowl 3. turn on the hot water 4. whisk a bowl of sugar into a paste 5. put the paste on the stove 6. turn on the hot water 7. boil the paste 8. add salt to the paste 9. turn off the water 10. put the bowl on a rack 11. pour the hot water into a saucepan 12. put some oil and sugar in the saucepan 13. turn the heat on 14. pour the sauce on to the bowl 15. put the paste | SIF 1. get your hot water 2. get your bowl 3. turn on the hot water 4. whisk a bowl of sugar into a paste 5. put the paste on the stove 6. turn on the hot water 7. boil the paste 8. add salt to the paste 9. turn off the water 10. put the paste on a rack 11. pour the hot water into a saucepan 12. put some oil and sugar in the saucepan 13. turn the heat on 14. pour the sauce on to the bowl 15. put the paste |

## Table 8: Scripts generated using \textsc{sequence} variant of GPT2 for novel scenarios. FT denotes output from the fine-tuned model and SIF refers to outputs from our framework applied to GPT2.

| Scenario | Good quality generation | Poor quality generation |
|----------|-------------------------|-------------------------|
| **FLYING IN AN AIRPLANE** | PT 1. go to airport 2. get boarding pass 3. go to the airline counter 4. present boarding pass and boarding pass id to attendant 5. present ticket and id to attendant 6. sit down in seat 7. make sure you are in seat 8. board airplane | SIF 1. go to airport 2. get the airline counter 3. present ticket and id to attendant 4. present boarding pass and boarding pass id to attendant 5. get boarding pass 6. board airplane 7. make sure you are in seat 8. sit down in seat |

## 7 Conclusion

To the best of our knowledge, this is the first work that investigates whether pre-trained GPT2 has an incomplete understanding of scripts, while BART and T5 did not even produce anything useful through zero-shot probing experiments. We propose SIF, an LM-agnostic script induction framework, which is shown to produce meaningful ESDs for unseen scenarios and mitigate errors (scenario-irrelevant, repeated, and misordered events) observed during probing experiments, as measured by automatic and manual evaluation. We also provide evidence for the generalization capability of our framework to novel scenarios. However, there is great room for improvement which is evident from manual error analysis and qualitative outputs. Future work may focus on developing more sophisticated automatic metrics as well as an end-to-end system for script induction which might help in mitigating cascading of errors, due to each component, common to any pipeline-based approaches.

## References

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. \textsc{COMET}: Commonsense transformers for automatic knowledge graph construction. In \textit{Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics}, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Zied Bouraoui, Jose Camacho-Collados, and Steven Schockaert. 2020. Inducing relational knowledge from BERT. In \textit{Proceedings of the AAAI Conference on Artificial Intelligence}, volume 34, pages 7456–7463.

Nathanael Chambers. 2017. Behind the scenes of an evolving event cloze test. In \textit{Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics}, pages 41–45.
Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In Proceedings of ACL-08: HLT, pages 789–797.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, pages 602–610.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. Educational and Psychological Measurement, 20(1):37–46.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In Proceedings of the Third International Workshop on Paraphrasing (IWP2005).

Joshua Feldman, Joe Davison, and Alexander M Rush. 2019. Commonsense knowledge mining from pre-trained models. arXiv preprint arXiv:1909.00505.

Jonathan Gordon and Benjamin Van Durme. 2013. Reporting bias and knowledge acquisition. In Proceedings of the 2013 workshop on Automated knowledge base construction, pages 25–30.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. arXiv preprint arXiv:1904.09751.

Bill Yuchen Lin, Seyeon Lee, Rahul Khanna, and Xiang Ren. 2020. Birds have four legs?? Numbersense: Probing numerical commonsense knowledge of pre-trained language models. arXiv preprint arXiv:2005.00683.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv preprint arXiv:1907.11692.

Qing Lyu, Li Zhang, and Chris Callison-Burch. 2020. Reasoning about goals, steps, and temporal ordering with wikihow. In Proceedings of The 2020 Conference on Empirical Methods In Natural Language Proceedings (EMNLP).

Risto Miikkulainen. 1995. Script-based inference and memory retrieval in subsymbolic story processing. Applied Intelligence, 5(2):137–163.

Ashutosh Modi, Tatjana Anikina, Simon Ostermann, and Manfred Pinkal. 2017. Inscript: Narrative texts annotated with script information. arXiv preprint arXiv:1703.05260.

Ashutosh Modi and Ivan Titov. 2014. Inducing neural models of script knowledge. In Proceedings of the Eighteenth Conference on Computational Natural Language Learning, pages 49–57.

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

Erik T Mueller. 2004. Understanding script-based stories using commonsense reasoning. Cognitive Systems Research, 5(4):307–340.

Simon Ostermann. 2020. Script knowledge for natural language understanding.

Simon Ostermann, Ashutosh Modi, Michael Roth, Stefan Thater, and Manfred Pinkal. 2018. Mcscript: A novel dataset for assessing machine comprehension using script knowledge. arXiv preprint arXiv:1803.05223.

Simon Ostermann, Michael Roth, and Manfred Pinkal. 2019. Mcscript2.0: A machine comprehension corpus focused on script events and participants. arXiv preprint arXiv:1905.09531.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2020. How context affects language models’ factual predictions. arXiv preprint arXiv:2005.04611.

Karl Pichotta and Raymond J Mooney. 2016. Using sentence-level lstm language models for script inference. arXiv preprint arXiv:1604.02993.

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.

Michaela Regneri, Alexander Koller, and Manfred Pinkal. 2010. Learning script knowledge with web experiments. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 979–988, Uppsala, Sweden. 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
A Appendix

A.1 Experimental Details

Train time for fine-tuning GPT2-L (774M parameters) on each of the folds for each of the variant was 13(±1) minutes on an average. RoBERTa-L architecture has 355M parameters and it took 282 minutes on an average to train relevance classifier for each of the folds while 294 minutes for temporal ordering classifier. Relevance and temporal ordering classifiers were trained on each of the training fold by keeping aside all the ESDs from a randomly chosen scenario for validation purposes as shown in Table 9.

A.2 Qualitative Outputs

We present a few outputs from BART-FT and BART-SIF in Table 10. We observe that the outputs are of poor quality. Although SIF is able to filter out a few irrelevant events in all the examples and correct the ordering (e.g. GOING ON A TRAIN), there are many mistakes like the events are not coherent ‘board the train Borders Boarders’, or repetitions such as ‘enjoy the train’, ‘enjoy train’. One reason for poor performance is that BART is an encode-decoder-based language model and thus encoding script knowledge in a scenario is a challenging task. On the other hand, GPT2 is a decoder-based language model which is trained with a language modeling objective and is able to generate the following events given a scenario in the form of a natural language prompt.
Table 9: Dataset partitioning details. Held-out refers to the scenarios kept aside from training-folds for validation purposes of relevance and temporal ordering classifiers.

| Fold | Scenarios | Held-out         |
|------|-----------|------------------|
| 1    | baking a cake, borrowing a book from the library, flying in an airplane, taking a train, riding on a bus | cooking pasta, going bowling, planting a tree, going grocery shopping, taking the underground |
| 2    | getting a haircut, going grocery shopping, planting a tree, repairing a flat bicycle tire, taking a bath | going bowling, planting a tree, going grocery shopping, paying with a credit card |
| 3    | eating in a fast food restaurant, paying with a credit card, playing tennis, going to the theater, taking a child to bed | going bowling, planting a tree, going grocery shopping, paying with a credit card |
| 4    | washing dishes, making a bonfire, going to the sauna, making coffee, going to the swimming pool | going bowling, planting a tree, going grocery shopping, paying with a credit card |
| 5    | getting a haircut, feeling a cat, washing food back in a restaurant, changing batteries in an alarm clock, checking in at an airport | going bowling, planting a tree, going grocery shopping, paying with a credit card |
| 6    | having a barbecue, ordering a pizza, cleaning up a flat, making scrambled eggs, taking the underground | going bowling, planting a tree, going grocery shopping, paying with a credit card |
| 7    | renovating a room, cooking pasta, setting a dinner, doing laundry, going bowling | going bowling, planting a tree, going grocery shopping, paying with a credit card |

Table 10: Scripts generated using EXPECT variant of BART for held-out scenarios. FT denotes output from the fine-tuned model and SIF refers to outputs from our framework when applied to BART. We filter extra tokens (⟨s⟩, ⟨/s⟩, ⟨SEP⟩) generated in between the events to present a clean output. These extra tokens were not generated in the case of GPT2.