Causal Inference of Script Knowledge

Noah Weber [1], Rachel Rudinger [2,3], Benjamin Van Durme [1]
[1]Johns Hopkins University
[2]University of Maryland, College Park
[3]Allen Institute for AI

Abstract

When does a sequence of events define an everyday scenario and how can this knowledge be induced from text? Prior works in inducing such *scripts* have relied on, in one form or another, measures of correlation between instances of events in a corpus. We argue from both a conceptual and practical sense that a purely correlation-based approach is insufficient, and instead propose an approach to script induction based on the causal effect between events, formally defined via interventions. Through both human and automatic evaluations, we show that the output of our method based on causal effects better matches the intuition of what a script represents.

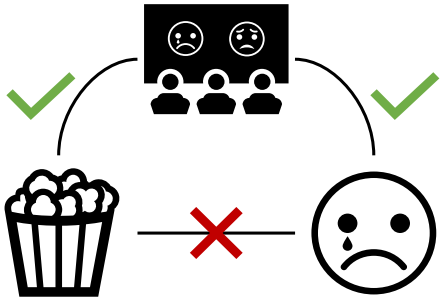

Figure 1: The events of *Watching a sad movie*, *Eating popcorn*, and *Crying*, may highly co-occur in a hypothetical corpus. What distinguishes valid event pair inferences (event pairs linked in a commensense scenario; noted by checkmarks above) versus invalid inferences (noted by a 'X')?

1 Introduction

Commonsense knowledge of everyday situations, as defined in terms of prototypical sequences of events, has long been held to play a major role in text comprehension and understanding (Minsky, 1974; Schank and Abelson, 1975, 1977; Bower et al., 1979; Abbott et al., 1985). Naturally, this has motivated a large body of work looking to learn such knowledge, such *scripts*,[1] from text corpora through data-driven approaches.

A minimal and often implicit requirement for any such approach is to resolve for any pair of events e_1 and e_2 what quantitative measure should be used to determine whether e_2 should "follow" e_1 in script. That is, documents may serve as descriptions of events that occur in the same situation as other events: what function may we compute over the raw presence or absence of events in documents that is most useful for script induction?

Chambers and Jurafsky (2008; 2009) adopted point-wise mutual information (PMI) (Church and Hanks, 1990) between event mentions. Others employed probabilities from a language model over event sequences (Jans et al., 2012; Rudinger et al., 2015; Pichotta and Mooney, 2016; Peng and Roth, 2016; Weber et al., 2018b), or other measures of event co-occurrence (Balasubramanian et al., 2013; Modi and Titov, 2014).

In this work we ask: do measures rooted in co-occurrence best capture the notion of whether one event should follow another in a script? We posit that it does not, that while observed correlations between events indicate relatedness, relatedness is not the only factor in determining whether events form a meaningful script.

Consider the example of Ge et al. (2016): *hurricane* events are prototypically connected with events of *donations* coming in. Likewise, *hurricane* events are connected to *evacuation* events. However, while *donation* and *evacuation* events are not conceptually connected in the same sense, there will exist strong statistical associations between the two. Figure 1 provides a second example: *eating popcorn* is not conceptually associated with *crying*,

[1]For simplicity we will refer to these 'prototypical event sequences' as scripts throughout the paper, though it should be noted scripts as originally proposed contain further structure not captured in this definition.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 7583–7596,
November 16–20, 2020.

but they might co-occur in a hypothetical corpus describing situations of *watching a sad movie*.

What do strict co-occurrence measures miss? In both examples the 'invalid' inferences arise from the same issue: an event such as *eating popcorn* may raise the probability of the event *crying*, but it does so only through a shared association with a *movie watching* context: the increase in probability is **not** due to the *eating popcorn* itself. In other words, what is lacking is a direct *causal effect* between these events, a quantity that can be formally defined using tools from the causal inference literature (Hernan and Robins, 2019).

In this work we demonstrate how a measure based on causal effects can be derived, computed, and employed for the extraction of script knowledge. Using crowdsourced human evaluations and a variant of the automatic cloze evaluation, we show how this definition better captures the notion of scripts as compared to prior standard measures, PMI and event sequence language models. Code and data available at `github.com/weberna/causalchains`.

2 Motivation

Does that fact that event e_2 is often observed after e_1 in the data (i.e. $p(e_2|e_1)$ is "high") mean that e_2 prototypically follows e_1, in the sense of being part of a script? In this section we argue that observed associations are not sufficient for the purpose of extracting script knowledge from text. We argue from a conceptual standpoint that some notion of causal relevance is required. We then give examples showing the practical pitfalls that may arise from ignoring this component. Finally, we propose our intervention based definition for script events, and show how it both explicitly defines a notion of 'causal relevance,' while simultaneously fixing the aforementioned practical pitfalls.

2.1 The Significance of Causal Relevance

The original works defining scripts are unequivocal about the importance of causal linkage between script events,[2] and other components of the original script definition (e.g. what-ifs, preconditions, postconditions, etc.) are arguably causal in nature. Early rule-based works on inducing scripts heavily used causal concepts in their schema representations (DeJong, 1983; Mooney and DeJong, 1985), as do related works in psychology (Black and Bower, 1980; Trabasso and Sperry, 1985).

But any measure based solely on $p(e_2|e_1)$ is agnostic to notions of causal relevance. Does this matter in practice? A high $p(e_2|e_1)$ indicates either: (1) a causal influence of e_1 on e_2, or (2) a common cause e_0 between them, meaning the relation between e_1 and e_2 is spurious. In the latter case, e_0 acts as a *confounder* between e_1 and e_2.

Ge et al. (2016) acknowledges that the associations picked up by correlational measures may often be spurious. Their solution relies on using trends of words in a temporal stream of newswire data, and hence is fairly domain specific.

2.2 Defining Causal Relevance

Early works such as Schank and Abelson (1975) are vague with respect to the meaning of "causally chained." Can one say that *watching a movie* has causal influence on the subsequent event of *eating popcorn* happening? Furthermore, can this definition be operationalized in practice?

We argue that both of these questions may be elucidated by taking a *manipulation*-based view of causation. Roughly speaking, this view holds that a causal relationship is one that is "*potentially exploitable for the purposes of manipulation and control*" – Woodward (2005). In other words, a causal relationship between *A* and *B* means that (in some cases) manipulating the value of *A* should result in a change in the value of *B*. A primary benefit of this view is that the meaning of a causal claim can be clarified by specifying what these 'manipulations' are exactly. We take this approach below to clarify what exactly is meant by 'causal relevance' between script events.

Imagine an agent reading a discourse. After reading a part of the discourse, the agent has some expectations for events that might happen next. Now imagine that, before the agents reads the next passage, we surreptitiously replace it with an alternate passage in which the event e_1 happens. We then allow the agent to continue reading. If e_1 is *causally relevant* to e_2, then this replacement should, in some contexts, raise the agents degree of belief in e_2 happening next (contra the case where we didn't intervene to make e_1 happen).

So, for example, if we replaced a passage such that e_1 = *watching a movie* was true, we could expect on average that the agent's degree of belief that e_2 = *eating popcorn* happens next will be

[2] "*...a script is not a simple list of events but rather a linked causal chain*" (Schank and Abelson, 1975)

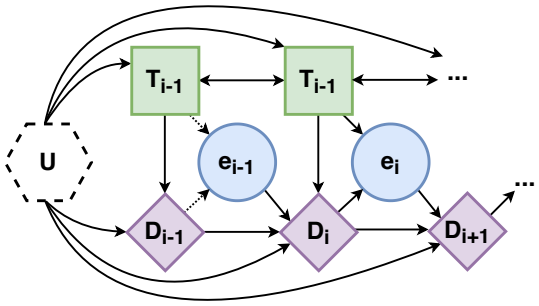

Figure 2: The diagram for our causal model up to time step i. Intervening on e_{i-1} acts to remove the dotted edges. See 3.1 for a description of the variables.

higher. In this way, we say these events are causally relevant, and are for our purposes, script events.

With this little 'story,' we have clarified the conceptual notion of causal relevance in our problem. In the next section, we formalize this story and its notion of intervention into a causal model.

3 Method

We would like to compute the effect of forcing an event of a certain type to occur in the text. The event types that get the largest increase in probability due to this are held to be 'script' events. Computing these quantities falls within the domain of causal inference, and hence will require its tools be used. There are three fundamental steps in causal inference we will need to work through to accomplish this: (1) **Define a Causal Model**: Identify the variables of interest in the problem, and define causal assumptions regarding these variables, (2) **Establish Identifiability**: With the given model, determine whether the causal quantity can be computed as a function of observed data. If it can, derive this function and move to (3) **Estimation**: Estimate this function using observed data. We go through each step in the next three subsections.

To best contrast with prior work, we use the event representation of Chambers and Jurafsky (2008) and others (Jans et al., 2012; Rudinger et al., 2015). A description of this representation is provided in the Supplemental.

3.1 Step 1: Define a Causal Model

A causal model defines a set of causal assumptions on the variables of interest in a problem. While there exists several formalisms that accomplish this, in this paper we make use of causal Bayesian networks (CBN) (Spirtes et al., 2000; Pearl, 2000). CBNs model dependencies between variables graphically in a manner similar to Bayesian networks; the key distinction being that the edges in a CBN posits a direction of *causal* influence between the variables [3].

We will define our causal model from a top down, data generating perspective in a way that aligns with our conceptual story from the previous section. Below we describe the four types of variables in our model, as well as their causal dependencies.

The World, U: The starting point for the generation of our data is the real world. This context is explicitly represented by the unmeasured variable U. This variable is unknowable and in general unmeasurable: we don't know how it is distributed, nor even what 'type' of variable it is. This variable is represented by the hexagonal node in Figure 2.

The Text, T: The next type of variable represents the text of the document. For indexing purposes, we segment the text into chunks T_1,...,T_N, where N is the number of realis events explicitly mentioned in the text. The variable T_i is thus the text chunk corresponding to the i^{th} event mentioned in text. These chunks may be overlapping, and may skip over certain parts of the original text.[4] The causal relationship between various text chunks is thus ambiguous. We denote this by placing bi-directional arrows between the square text nodes in Figure 2. The context of the world also causally influences the content of the text, hence we include an arrow from U to all text variables, T_i.

Event Inferences, e: In our story in Section 2, an agent reads a chunk of text and infers the type of event that was mentioned in the piece of text. This inference is represented (for the i^{th} event in text) in our model by the variable $e_i \in E$ where E is the set of possible atomic event types (described at the end of this section).[5]

[3] See Pearl (2000); Bareinboim et al. (2012) for a comprehensive definition of CBNs and their properties.

[4] Keeping with prior work, we use the textual span of the event predicate syntactic dependents as the textual content of an event. The ordering of variables T_i corresponds to the positions of the event predicates in the text.

[5] For this study we use the output of information extraction tools as a proxy for the variable e_i (see supplemental). As such, it is important to note that there will be bias in computations due to *measurement error*. Fortunately, there do exists methods in the causal inference literature that can adjust for this bias (Kuroki and Pearl, 2014; Miao et al., 2018). Wood-Doughty et al. (2018) derive equations in a case setting related to ours (i.e. with measurement bias on the variable being intervened on). Dealing with this issue will be an important next step for future work.

The textual content of T_i causally influences the inferred type e_i, hence directional connecting arrows in Figure 2.

Discourse Representation, *D* The variable e_i represent a high level abstraction of part of the semantic content found in T_i. Is this information about events used for later event inferences by an agent reading the text? Prior results in causal network/chain theories of discourse processing (Black and Bower, 1980; Trabasso and Sperry, 1985; Van den Broek, 1990) seem to strongly point to the affirmative. In brief, these theories hold that the identities of the events occurring in the text – and the causal relations among them – are a core part of how a discourse is represented in human memory while reading, and more-so, that this information significantly affects a reader's event based inferences(Trabasso and Van Den Broek, 1985; Van den Broek and Lorch Jr, 1993). Thus we introduce a *discourse representation* variable, D_i, itself a combination of two sub-variables, D_i^I and D_i^O.

The variable $D_i^I \in E^*$ is a sequence[6] of events that were explicitly stated ***in the text***, up to step i. After each step, the in-text event inferred at i (the variable e_i) is appended to D_{i+1}^I. The causal parents of D_{i+1}^I are thus e_i and D_i^I (which is simply copied over). We posit that the information in D_i^I provides information in the inference of e_i, and thus draw an arrow from D_i^I to e_i.

Unstated events not found in the text but inferred by the reader also have an effect on event inferences (McKoon and Ratcliff, 1986, 1992; Graesser et al., 1994). We thus additionally take this into consideration in our causal model by including an ***out of text*** discourse representation variable, $D_i^O \in 2^{|E|}$. This variable is a bag of events that a reader may infer implicitly from the text chunk T_i using common sense. Its causal parents are thus both the text chunk T_i, as well as the world context U; its causal children are e_i. Obtaining this information is done via human annotation and discussed later. D_i is thus equal to (D_i^I, D_i^O), and inherits the incoming and outgoing arrows of both in Figure 2.

3.2 Step 2: Establishing Identifiability

Our goal is to compute the effect that intervening and setting the preceding event e_{i-1} to $k \in E$ has on the distribution over the subsequent event e_i. Now that we have a causal model in the form of Fig. 2, we can now define this effect. Using the notation of Pearl (2000), we write this as:

$$p(e_i|do(e_{i-1} = k)) \quad (1)$$

The semantics of $do(e_{i-1} = k)$ are defined as an 'arrow breaking' operation on Figure 2 which deletes the incoming arrows to e_{i-1} (the dotted arrows in Figure 2) and sets the variable to k. Before a causal query such as Eq. 1 can be estimated we must first establish identifiability (Shpitser and Pearl, 2008): can the causal query be written as a function of (only) the observed data?

Eq. 1 is identified by noting that variables T_{i-1} and D_{i-1} meet the 'back-door criterion' of Pearl (1995), allowing us to write Eq. 1 as the following:

$$\mathbb{E}_{T_{i-1},D_{i-1}}\Big[p(e_i|e_{i-1} = k, D_{i-1}, T_{i-1})\Big] \quad (2)$$

Our next step is to estimate the above equation. If one has an estimate for the conditional $p(e_i|e_{i-1}, D_{i-1}, T_{i-1})$, then one may "plug it into" Eq. 2 and use a Monte Carlo approximation of the expectation (using samples of (T, D)). This simple *plug in* estimator is what we use here

It is important to be aware of the fact that This estimator, specifically when plugging in machine learning methods, is quite naive (e.g. Chernozhukov et al. (2018)), and will suffer from an asymptotic (first order) bias. [7] which prevents one from constructing meaningful confidence intervals or performing certain hypothesis tests. That said, in practice these machine learning based plug in estimators can achieve quite reasonable performance (see for example, the results in Shalit et al. (2017)), and since our current use case can be validated empirically, we save the usage of more sophisticated estimators (and proper statistical inference) for future work[8].

3.3 Step 3: Estimation

Eq. 2 depends on the conditional, $p_{e_i} = p(e_i|e_{i-1}, D_{i-1}, T_{i-1})$, which we estimate via stan-

[6] We don't explicitly model the causal structure between events in D_i, the importance of which is a key finding in the above referenced literature. While this wouldn't change the structure of our causal model, it would impact the estimation stage, and would be an interesting line of future work.

[7] See Fisher and Kennedy (2018) for an introduction on how this bias manifests.

[8] Semiparametric estimation of equations such as Eq. 2 involving high dimensional variables (like text) is an open problem that we do not address here. See (D'Amour et al., 2020; Kennedy, 2016; Chernozhukov et al., 2018) for an analysis of some of the problems that arise in both high dimensional causal inference and semiparametric estimation (ie estimation without full parametric assumptions). See Keith et al. (2020) for an overview of problems that arise particularly when dealing with text.

dard ML techniques with a dataset of samples drawn from $p(e_i, e_{i-1}, D_{i-1}, T_{i-1})$. There are two issues: (1) How do we deal with out-of-text events in D_{i-1}?, and (2) What form will p_{e_i} take?

Dealing with Out-of-Text Events Recall that D_i is combination of the variables D_i^I and D_i^O. To learn a model for p_{e_i} we require samples from the full joint. Out of the box however, we only have access to $p(e_i, e_{i-1}, D_{i-1}^I, T_{i-1})$. If, for the samples in our current dataset, we could draw samples from $p_D = p(D_{i-1}^O | e_i, e_{i-1}, D_{i-1}^I, T_{i-1})$, then we would have access to a dataset with samples drawn from the full joint.

In order to 'draw' samples from p_D we employ human annotation. Annotators are presented with a human readable form of $(e_i, e_{i-1}, D_{i-1}^I, T_{i-1})$[9] and are asked to annotate for possible events belonging in D_{i-1}^O. Rather than opt for noisy annotations obtained via freeform elicitation, we instead provide users with a set of 6 candidate choices for members of D_{i-1}^O. The candidates are obtained from various knowledge sources: ConceptNet (Speer and Havasi, 2012), VerbOcean (Chklovski and Pantel, 2004), and high PMI events from the NYT Gigaword corpus (Graff et al., 2003). The top two candidates are selected from each source.

In a scheme similar to Zhang et al. (2017), we ask users to rate candidates on an ordinal scale and consider candidates rated at or above a 3 (out of 4) to be considered within D_{i-1}^O. We found annotator agreement to be quite high, with a Krippendorf's α of 0.79. Under this scheme, we crowdsourced a dataset of 2000 fully annotated examples on the Mechanical Turk platform. An image of our annotation interface is provided in the Appendix.

The Conditional Model We use neural networks to model p_{e_i}. In order to deal with the small amount of fully annotated data available, we employ a finetuning paradigm. We first train a model on a large dataset that does not include annotations for D_{i-1}^O. This model consists of a single layer, 300 dimensional GRU encoder which encodes $[D_{i-1}^I, e_{i-1}]$ into a vector $v_e \in R^d$ and a CNN-based encoder which encodes T_{i-1} into a vector $v_t \in R^d$. The term p_{e_i} is modeled as:

$$p_{e_i} \propto Av_e + Bv_t$$

for matrices A and B of dimension $|E| \times d$. We then finetune this model on the 2000 annotated examples including D_{i-1}^O. We add a new parameter matrix, C, to the previously trained model (allowing it to take D_{i-1}^O as input) and model p_{e_i} as:

$$p_{e_i} \propto Av_e + Bv_t + Cv_o$$

The input v_o is the average of the embeddings for the events found in D_{i-1}^O. The parameter matrix C is thus the only set of parameters trained 'from scratch,' on the 2000 annotated examples. The rest of the parameters are initialized and finetuned from the previously trained model. See Appendix for further training details.

3.4 Extracting Script Knowledge

Provided a model of the conditional p_{e_i} we can approximate Eq. 2 via Monte Carlo by taking our annotated dataset of $N = 2000$ examples and computing the following average:

$$\hat{P}_k = \frac{1}{N} \sum_{j=1}^{N} p(e_i | e_{i-1} = k, D_j, T_j) \quad (3)$$

This gives us a length $|E|$ vector $\hat{P}_k$ whose l^{th} component, $\hat{P}_{kl}$ gives $p(e_i = l | do(e_{i-1} = k))$. We compute this vector for all values of k. Note that this computation only needs to be done once.

There are several ways one could extract script-like knowledge using this information. In this paper, we define a normalized score over intervened-on events such that the script compatibility score between two concurrent events is defined as:

$$S(e_{i-1} = k, e_i = l) = \frac{\hat{P}_{kl}}{\sum_{j=1}^{E} \hat{P}_{jl}} \quad (4)$$

We term this as the 'Causal' score in the eval below.

4 Experiments and Evaluation

Automatic evaluation of methods that extract script-like knowledge is an open problem that we do not attempt to tackle here,[10] relying foremost on crowdsourced human evaluations to validate our method.

However, as we aim to provide a contrast to prior script-induction approaches, we perform an experiment looking at a variant of the popular automatic narrative cloze evaluation.

[9] In the final annotation experiment, we found it easier for annotators to be only provided the text T_{i-1}, given that many events in D_{i-1}^I are irrelevant.

[10] See discussions by Rudinger et al. (2015) and Chambers (2017).

4.1 Dataset

For these experiments, we use the Toronto Books corpus (Zhu et al., 2015; Kiros et al., 2015), a collection of fiction novels spanning multiple genres. The original corpus contains 11,040 books by unpublished authors. We remove duplicate books from the corpus (by exact file match), leaving a total of 7,101 books. The books are assigned randomly to train, development, and test splits in 90%-5%-5% proportions. Each book is then run through a pipeline of tokenization with CoreNLP 3.8 (Manning et al., 2014), parsing with CoreNLP's universal dependency parser (Nivre et al., 2016) and coreference resolution (Clark and Manning, 2016b), before feeding the results into PredPatt (White et al., 2016). We additionally tag the events with factuality predictions from Rudinger et al. (2018b) (we only consider factual events). The end result is a large dataset of event chains centered around a single protagonist entity, similar to (Chambers and Jurafsky, 2008). We make this data public to facilitate further work in this area. See the Appendix for a full detailed overview of our pipeline.

4.2 Baselines

In this paper, we compare against the two dominant approaches for script induction (under a atomic event representation[11]): PMI (similar to Chambers and Jurafsky (2008, 2009)) and LMs over event sequences (Rudinger et al., 2015; Pichotta and Mooney, 2016). We defer definitions for these models to the cited papers, below we provide the relevant details for each baseline, with further training details provided in the Appendix.

For computing PMI we follow many of the details from (Jans et al., 2012). Due to the nature of the evaluations, we utilize their 'ordered' PMI-variant. Also like Jans et al. (2012), we use skip-bigrams with a window of 2 to deal with count sparsity. Consistent with prior work we additionally employ the discount score of Pantel and Ravichandran (2004). For the LM, we use a standard, 2 layer, GRU-based neural network language model, with 512 dimensional hidden states, trained on a log-likelihood objective.

Method	Average Score	Average Rank (1-6)
Causal	49.71	4.10
LM	35.95	3.39
PMI	34.92	3.02

Table 1: Average Annotator Scores in Pairwise annotation experiment

Causal	LM	PMI	Target
X tripped	*X came*	*X featured*	*X fell*
X lit	*X sat*	*X laboured*	*X inhaled*
X aimed	*X came*	*X alarmed*	*X fired*
X poured	*X nodded*	*X credited*	*X refilled*
X radioed	*X made*	*X fostered*	*X ordered*

Table 2: Examples from each system, each of which outputs a previous event that maximizes the score/likelihood that the Targeted event follows in text.

4.3 Eval I: Pairwise Event Associations

Any system aimed at extracting script-like knowledge should be able to answer the following *abductive* question: given an event e_i happened, what previous event e_{i-1} best explains why e_i is true? In other words, what e_{i-1}, if it were true, would maximize my belief that e_i was true. We evaluate each method's ability to do this via a human evaluation.

On each task, annotators are presented with six event pairs (e_{i-1}, e_i), where e_i is the same for all pairs, but e_{i-1} is generated by one of the three systems. Similar to the human evaluation in Pichotta and Mooney (2016), we filter out outputs in the top-20 most frequent events list for all systems. For each system, we pick the top two events that maximize $S(\cdot, e_i)$, $PMI(\cdot, e_i)$, and $p_{lm}(\cdot, e_i)$, for the Causal, PMI, and LM systems respectively, and present them in random order. For each pair, users are asked to provide a scalar annotation (from 0%-100%, via a slider bar) on the chance that e_i is *true afterwards or happened as a result of* e_{i-1}. The annotation scheme is modeled after the one presented in Sakaguchi and Van Durme (2018), and shown to be effective for paraphrase evaluation in Hu et al. (2019). Example outputs for systems are provided for several e_1 choices for this task in Table 2.

The evaluation is done for 150 randomly[12] chosen instances of e_i, each with 6 candidate e_{i-1}. We

[11] There are also a related class of methods based on creating compositional event embeddings (Modi, 2016; Weber et al., 2018a). Since the event representation used here is atomic it makes little sense to use them here.

[12] Note that we do manually filter out of the initial random list events which we judge as difficult to understand

Method	Average Score	Average Rank (1-3)
Causal	60.12	2.19
LM	57.40	2.12
PMI	44.26	1.68

Table 3: Average Annotator Scores in Chain annotation experiment

have two annotators provide annotations for each task, and similar to Hu et al. (2019), average these annotations together for a gold annotation.

In Table 1 we provide the results of the experiment, providing both the average annotation score for the outputs of each system, as well as the average relative ranking (with a rank of 6 indicating the annotators ranked the output as the highest/best in the task, and a rank of 1 indicating the opposite). We find that annotators consistently rated the Causal system higher. The differences (in both Score and Rank) between the Causal system and the next best system are significant under a Wilcoxon signed-rank test ($p < 0.01$).

4.4 Eval II: Event Chain Completion

Of course, while pairwise knowledge between events is a minimum prerequisite, we would also like to generalize to handle chains of events containing multiple events. In this section, we look at each system's ability to provide an intuitive completion to an event chain. More specifically, the model is provided with a chain of three context events, (e_1, e_2, e_3), and is tasked with providing a suitable e_4 that might follow given the first three events. We evaluate each method's ability to do this via a human evaluation.

Since both PMI and the Causal model [13] work only as pairwise models, we adopt the method of Chambers and Jurafsky (2008) for chains. For both the PMI and Causal model, we pick the e_4 that maximizes $\frac{1}{3}\sum_{i=1}^{3} Score(e_i, e_4)$, where $Score$ is either PMI or Eq 4. The LM model chooses an e_4 that maximizes the joint over all events.

Our annotation task is similar to the one in 4.3, except the pairs provided consist of a context (e_1, e_2, e_3) and a system generated e_4. Each system generates its top choice for e_4, giving annotators 3 pairs[14] to annotate for each task (i.e. each context). On each task, human annotators are asked to provide a scalar annotation (from 0%-100%, via a slider) on the chance that e_4 is *true afterwards or happened as a result of* the chain of context events. The evaluation is done for 150 tasks, with two annotators on each task. As before, we average these annotations together for a gold annotation.

In Table 3 we provide results of the experiment. Note the the rankings are now from 1 to 3 (higher is better). We find annotators usually rated the Causal system higher, though the LM model is much closer in this case. The differences (in both Score/Rank) between the Causal and LM system outputs are not significant under a Wilcoxon signed-rank test, though the differences between the Causal and PMI system is $(p < 0.01)$. The fact that the pairwise Causal model is still able to (at minimum) match the full sequential model on a chain-wise evaluation speaks to the robustness of the event associations mined from it, and further motivates work in extending the method to the sequential case.

4.5 Diversity of System Outputs

But what type of event associations are found from the Causal model? As noted both in Rudinger et al. (2015) and in Chambers (2017), PMI based approaches can often extract intuitive event relationships, but may sometimes overweight low frequency events or suffer problems from count sparsity. LM based models, on the other hand, were noted for their preference towards boring, uninformative, high frequency events (like 'sat' or 'came'). So where does the Causal model lay on this scale?

We study this by looking at the percentage of unique words used by each system in the previous evaluations, presented in Table 5. Unsurprisingly, we find that PMI chooses a new word to output often (77%-84% of the time), while the LM model very rarely does (only 7%-13%). The Causal model, while not as adventurous as the PMI system, tends to produce very diverse output, generating a new output 60%-76% of the time. Both the PMI and Causal system produce relatively less diverse output on the chain task, which is expected due to the averaging scheme used by each to select events.

4.6 Infrequent Narrative Cloze

The narrative cloze task, or some variant of it, has remained a popular automatic test for systems aiming to extract 'script' knowledge. The task is usually formulated as follows: given a chain of events $e_1, ...e_{n-1}$ that occurs in the data, predict the held out next event that occurs in the data, e_n. There

[13] Generalizing the Causal model to multiple interventions, though out of scope here, is a clear next step for future work.

[14] We found providing six pairs per task to be overwhelming given the longer context

Method	Pairwise	Chain
Causal	*X awoke* (2%) *X parried* (1%)	*X collided* (4%) *pinched X* (3%)
LM	*X came* (30%) *X sat* (27%)	*X made* (23%) *X came* (15%)
PMI	*X lurched* (1%) *X patroled* (1%)	*bribed X* (3%) *X swarmed* (2%)

Table 4: Two most used output events (and % of times it is used) for each system, for each human evaluation

Method	Pairwise	Chain
Causal	76.0%	60.1%
LM	7.30%	13.3%
PMI	84.0%	77.6%

Table 5: % of times a system outputs a new event it previously had not used before.

exists various measures to calculate a models ability to perform in this task, but arguably the most used one is the Recall@N measure introduced in Jans et al. (2012). Recall@N works as follows: for a cloze instance, a system will return the top N guesses for e_n. Recall@N is the percentage of times e_n is found anywhere in the top N list.

The automatic version of the cloze task has notable limitations. As noted in Rudinger et al. (2015), the cloze task is essentially a language modeling task; it measures how well a model *fits* the data. The question then becomes whether data fit implies valid script knowledge was learned. The work of Chambers (2017) casts serious doubts on this, with various experiments showing automatic cloze evaluations are biased to high frequency, uninformative events, as opposed to informative, *core*, script events. They further posit human annotation as a necessary requirement for evaluation.

In this experiment, we provide another datapoint for the inadequacy of the automatic cloze, while simultaneously showing the relative robustness of the knowledge extracted from our Causal system. For the experiment, we make the following assumptions: (1) Highly frequent events tend to appear in many scenarios, and hence are less likely to be an informative 'core' event for a script, and (2) Less frequent events are more likely to appear only in specific scenarios, and are thus more likely to be informative events. If these are true, then a system that has extracted useful script knowledge should keep (or even improve) cloze performance when the correct answer for e_n is a less frequent event.

We thus propose a Infrequent Cloze task. In this task we create a variety of different cloze datasets (each with 2000 instances) from our test set. Each set is indexed by a value C, such that the indicated dataset does not include instances from the top C most frequent events ($C = 0$ is the normal cloze setting). We compute a Recall@100 cloze task on 7 sets of various C and report results in Table 6.

At $C = 0$, as expected, the LM model is vastly superior. The performance of the LM model drastically drops however, as soon as C increases, indicating an overreliance on prior probability. The LM performance drops below 2% once $C = 200$, indicating almost no ability in predicting informative events such as *drink* or *pay*, both of which occur in this set in our case. The PMI and Causal model's performance on the other hand, steadily improve while C increases, with the Causal model consistently outperforming PMI. This result, *when combined with* the results of the human evaluation, give further evidence towards the relative robustness of the Causal model in extracting informative core events. The precipitous drop in performance of the LM further underscores problems that a naive automatic cloze evaluation may cover up.

5 Related Work

Our work looks at script like associations between events in a manner similar to Chambers and Jurafsky (2008), and works along similar lines (Jans et al., 2012; Pichotta and Mooney, 2016). Related lines of work exist, such as work using generative models to induce probabilistic schemas(Chambers, 2013; Cheung et al., 2013; Ferraro and Van Durme, 2016), work showing how script knowledge may be mined from user elicited event sequences (Regneri et al., 2010; Orr et al., 2014), and approaches take advantage of hand coded schematic knowledge (Mooney and DeJong, 1985; Raskin et al., 2003). The cognitive linguistics literature is rich with work studying the role of causal semantics in linguistic constructions and argument structure (Talmy, 1988; Croft, 1991, 2012), as well as the causal semantics of lexical items themselves (Wolff and Song, 2003). Work in the NLP literature on extracting causal relations has benefited from this line of work, utilizing the systematic way in which causation in expressed in language to mine relations (Girju and Moldovan, 2002; Girju, 2003; Riaz and Girju, 2013; Blanco et al., 2008; Do et al., 2011; Bosselut et al., 2019). This line work aims to extract

Method	Exclusion Threshold						
	< 0	< 50	< 100	< 125	< 150	< 200	< 500
Causal	5.60	7.10	7.00	**7.49**	**7.20**	**8.20**	**9.10**
LM	**65.3**	**28.1**	**9.70**	6.30	3.60	1.70	0.25
PMI	1.80	3.30	3.36	4.10	4.00	4.90	7.00

Table 6: Recall@100 Narrative Cloze Results. $< C$ indicates that instances whose cloze answer is one of the top C most frequent events are not evaluated on

causal relations between events that are in some way explicitly expressed in the text (e.g. through the use of particular constructions).Taking advantage of how causation is expressed in language may benefit our causal model, and is a potential path for future work.

6 Conclusions and Future Work

In this work we argued for a causal basis in script learning. We showed how this causal definition could be formalized and used in practice utilizing the tools of causal inference, and verified our method with human and automatic evaluations. In the current work, we showed a method calculating the 'goodness' of a script in the simplest case: between pairwise events, which we showed still to be quite useful. A causal definition is in no way limited to this pairwise case, and future work may generalize it to the sequential case or to event representations that are compositional. Having a causal model shines a light on the assumptions made here, and indeed, future work may further refine or overhaul them, a process which may further shine a light on the nature of the knowledge we are after.

Acknowledgements

This work was supported by DARPA KAIROS and the Allen Institute for AI. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes.The views and conclusions contained in this pub-lication are those of the authors and should not be interpreted as representing official policies or endorsement.

References

Valerie Abbott, John B Black, and Edward E Smith. 1985. The representation of scripts in memory. *Journal of memory and language*, 24(2):179–199.

Niranjan Balasubramanian, Stephen Soderland, Oren Etzioni Mausam, and Oren Etzioni. 2013. Generating coherent event schemas at scale. In *Proceedings of EMNLP*.

Elias Bareinboim, Carlos Brito, and Judea Pearl. 2012. Local characterizations of causal bayesian networks. In *Graph Structures for Knowledge Representation and Reasoning*, pages 1–17. Springer.

John B Black and Gordon H Bower. 1980. Story understanding as problem-solving. *Poetics*, 9(1-3):223–250.

Eduardo Blanco, Nuria Castell, and Dan I Moldovan. 2008. Causal relation extraction. In *Lrec*.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. *ACL 2019*.

Gordon H Bower, John B Black, and Terrence J Turner. 1979. Scripts in memory for text. *Cognitive psychology*, 11(2):177–220.

Paul Van den Broek. 1990. The causal inference maker: Towards a process model of inference generation in text comprehension. *Comprehension processes in reading*, pages 423–445.

Paul Van den Broek and Robert F Lorch Jr. 1993. Network representations of causal relations in memory for narrative texts: Evidence from primed recognition. *Discourse processes*, 16(1-2):75–98.

Nathanael Chambers. 2013. Event schema induction with a probabilistic entity-driven model. In *Proceedings of EMNLP*, volume 13.

Nathanael Chambers. 2017. Behind the scenes of an evolving event cloze test. In *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 the Association for Computational Linguistics (ACL)*, Hawaii, USA.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Association for Computational Linguistics (ACL)*, Singapore.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar. Association for Computational Linguistics.

Victor Chernozhukov, Denis Chetverikov, Mert Demirer, Esther Duflo, Christian Hansen, Whitney Newey, and James Robins. 2018. Double/debiased machine learning for treatment and structural parameters.

Jackie Chi Kit Cheung, Hoifung Poon, and Lucy Vanderwende. 2013. Probabilistic frame induction. In *Proceedings of NAACL*.

Timothy Chklovski and Patrick Pantel. 2004. Verbocean: Mining the web for fine-grained semantic verb relations. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 33–40.

Kenneth Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1).

Kevin Clark and Christopher D. Manning. 2016a. Deep reinforcement learning for mention-ranking coreference models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2256–2262, Austin, Texas. Association for Computational Linguistics.

Kevin Clark and Christopher D. Manning. 2016b. Improving coreference resolution by learning entity-level distributed representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 643–653, Berlin, Germany. Association for Computational Linguistics.

William Croft. 1991. *Syntactic categories and grammatical relations: The cognitive organization of information*. University of Chicago Press.

William Croft. 2012. *Verbs: Aspect and causal structure*. OUP Oxford.

Gerald DeJong. 1983. Acquiring schemata through understanding and generalizing plans. In *IJCAI*.

Quang Xuan Do, Yee Seng Chan, and Dan Roth. 2011. Minimally supervised event causality identification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 294–303. Association for Computational Linguistics.

Alexander D'Amour, Peng Ding, Avi Feller, Lihua Lei, and Jasjeet Sekhon. 2020. Overlap in observational studies with high-dimensional covariates. *Journal of Econometrics*.

Francis Ferraro and Benjamin Van Durme. 2016. A unified bayesian model of scripts, frames and language. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Aaron Fisher and Edward H Kennedy. 2018. Visually communicating and teaching intuition for influence functions. *arXiv preprint arXiv:1810.03260*.

Tao Ge, Lei Cui, Heng Ji, Baobao Chang, and Zhifang Sui. 2016. Discovering concept-level event associations from a text stream. In *Natural Language Understanding and Intelligent Applications*, pages 413–424. Springer.

Roxana Girju. 2003. Automatic detection of causal relations for question answering. In *Proceedings of the ACL 2003 workshop on Multilingual summarization and question answering-Volume 12*, pages 76–83. Association for Computational Linguistics.

Roxana Girju and Dan Moldovan. 2002. Mining answers for causation questions.

Arthur C Graesser, Murray Singer, and Tom Trabasso. 1994. Constructing inferences during narrative text comprehension. *Psychological review*, 101(3):371.

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2003. English gigaword. *Linguistic Data Consortium, Philadelphia*, 4(1):34.

Miguel A Hernan and James M Robins. 2019. *Causal inference: What if*. CRC Boca Raton, FL;.

J Edward Hu, Rachel Rudinger, Matt Post, and Benjamin Van Durme. 2019. Parabank: Monolingual bitext generation and sentential paraphrasing via lexically-constrained neural machine translation. *arXiv preprint arXiv:1901.03644*.

Bram Jans, Steven Bethard, Ivan Vulić, and Marie Francine Moens. 2012. Skip n-grams and ranking functions for predicting script events. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 336–344. Association for Computational Linguistics.

Katherine A Keith, David Jensen, and Brendan O'Connor. 2020. Text and causal inference: A review of using text to remove confounding from causal estimates. *ACL 2020*.

Edward H Kennedy. 2016. Semiparametric theory and empirical processes in causal inference. In *Statistical causal inferences and their applications in public health research*, pages 141–167. Springer.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3294–3302. Curran Associates, Inc.

Manabu Kuroki and Judea Pearl. 2014. Measurement bias and effect restoration in causal inference. *Biometrika*, 101(2):423–437.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd*

Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pages 55–60.

Gail McKoon and Roger Ratcliff. 1986. Inferences about predictable events. *Journal of Experimental Psychology: Learning, memory, and cognition*, 12(1):82.

Gail McKoon and Roger Ratcliff. 1992. Inference during reading. *Psychological review*, 99(3):440.

Wang Miao, Zhi Geng, and Eric J Tchetgen Tchetgen. 2018. Identifying causal effects with proxy variables of an unmeasured confounder. *Biometrika*, 105(4):987–993.

Marvin Minsky. 1974. A framework for representing knowledge. MIT Laboratory Memo 306.

Ashutosh Modi. 2016. Event embeddings for semantic script modeling. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 75–83.

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, Ann Arbor, Michigan. Association for Computational Linguistics.

Raymond Mooney and Gerald DeJong. 1985. Learning schemata for natural language processing. In *Ninth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 681–687.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France. European Language Resources Association (ELRA).

John Walker Orr, Prasad Tadepalli, Janardhan Rao Doppa, Xiaoli Fern, and Thomas G Dietterich. 2014. Learning scripts as hidden markov models. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*.

Patrick Pantel and Deepak Ravichandran. 2004. Automatically labeling semantic classes. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*.

Judea Pearl. 1995. Causal diagrams for empirical research. *Biometrika*, 82(4):669–688.

Judea Pearl. 2000. *Causality: models, reasoning and inference*, volume 29. Springer.

Haoruo Peng and Dan Roth. 2016. Two discourse driven language models for semantics. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Karl Pichotta and Raymond J Mooney. 2016. Learning statistical scripts with lstm recurrent neural networks. In *AAAI*, pages 2800–2806.

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*, pages 1–40. Association for Computational Linguistics.

Victor Raskin, Sergei Nirenburg, Christian F. Hempelmann, Inna Nirenburg, and Katrina E. Triezenberg. 2003. The genesis of a script for bankruptcy in ontological semantics. In *Proceedings of the HLT-NAACL 2003 Workshop on Text Meaning*, pages 30–37.

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. Association for Computational Linguistics.

Mehwish Riaz and Roxana Girju. 2013. Toward a better understanding of causality between verbal events: Extraction and analysis of the causal power of verb-verb associations. In *Proceedings of the SIGDIAL 2013 Conference*, pages 21–30.

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.

Rachel Rudinger, Adam Teichert, Ryan Culkin, Sheng Zhang, and Benjamin Van Durme. 2018a. Neural-davidsonian semantic proto-role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 944–955, Brussels, Belgium. Association for Computational Linguistics.

Rachel Rudinger, Aaron Steven White, and Benjamin Van Durme. 2018b. Neural models of factuality. *arXiv preprint arXiv:1804.02472*.

Keisuke Sakaguchi and Benjamin Van Durme. 2018. Efficient online scalar annotation with bounded support. *ACL*.

Roger C Schank and Robert P Abelson. 1975. Scripts, plans, and knowledge. *IJCAI*.

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

Uri Shalit, Fredrik D Johansson, and David Sontag. 2017. Estimating individual treatment effect: generalization bounds and algorithms. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3076–3085. JMLR. org.

Ilya Shpitser and Judea Pearl. 2008. Complete identification methods for the causal hierarchy. *Journal of Machine Learning Research*, 9(Sep):1941–1979.

Robyn Speer and Catherine Havasi. 2012. Representing general relational knowledge in conceptnet 5. In *LREC*.

Peter Spirtes, Clark N Glymour, Richard Scheines, David Heckerman, Christopher Meek, Gregory Cooper, and Thomas Richardson. 2000. *Causation, prediction, and search*. MIT press.

Leonard Talmy. 1988. Force dynamics in language and cognition. *Cognitive science*, 12(1):49–100.

Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for computational Linguistics.

Tom Trabasso and Linda L Sperry. 1985. Causal relatedness and importance of story events. *Journal of Memory and language*, 24(5):595–611.

Tom Trabasso and Paul Van Den Broek. 1985. Causal thinking and the representation of narrative events. *Journal of memory and language*, 24(5):612–630.

Noah Weber, Niranjan Balasubramanian, and Nathanael Chambers. 2018a. Event representations with tensor-based compositions. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Noah Weber, Leena Shekhar, Niranjan Balasubramanian, and Nate Chambers. 2018b. Hierarchical quantized representations for script generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3783–3792.

Aaron Steven White, D. Reisinger, Keisuke Sakaguchi, Tim Vieira, Sheng Zhang, Rachel Rudinger, Kyle Rawlins, and Benjamin Van Durme. 2016. Universal decompositional semantics on universal dependencies. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1713–1723, Austin, Texas. Association for Computational Linguistics.

Phillip Wolff and Grace Song. 2003. Models of causation and the semantics of causal verbs. *Cognitive Psychology*, 47(3):276–332.

Zach Wood-Doughty, Ilya Shpitser, and Mark Dredze. 2018. Challenges of using text classifiers for causal inference. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, volume 2018, page 4586. NIH Public Access.

James Woodward. 2005. *Making things happen: A theory of causal explanation*. Oxford university press.

Sheng Zhang, Rachel Rudinger, Kevin Duh, and Benjamin Van Durme. 2017. Ordinal common-sense inference. *TACL*.

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 the IEEE international conference on computer vision*, pages 19–27.

A Appendix

A.1 Event Representation

To best contrast with prior work, we use the event representation of Chambers and Jurafsky (2008) and others (Jans et al., 2012; Rudinger et al., 2015). Each event is a pair (p, d), where p is the event predicate (e.g. *hit*), and d is the dependency relation (e.g. *nsubj*) between the predicate and the *protagonist* entity. The protagonist is the entity that participates in every event in the considered event chain, e.g., the 'Bob' in the chain 'Bob sits, Bob eats, Bob pays.'

A.2 Data Pre-Processing

For these experiments, we use the Toronto Books corpus (Zhu et al., 2015; Kiros et al., 2015), a collection of fiction novels spanning multiple genres. The original corpus contains 11,040 books by unpublished authors. We remove duplicate books from the corpus (by exact file match), leaving a total of 7,101 books; a distribution by genre is provided in Table 7. The books are assigned randomly to train, development, and test splits in 90%-5%-5% proportions (6,405 books in train, and 348 in development and test splits each). Each book is then sentence-split and tokenized with CoreNLP 3.8 (Manning et al., 2014); these sentence and token boundaries are observed in all downstream processing.

A.2.1 Narrative Chain Extraction Pipeline

In order to extract the narrative chains from the Toronto Books data, we implement the following

Adventure	390	Other	284
Fantasy	1,440	Romance	1,437
Historical	161	Science Fiction	425
Horror	347	Teen	281
Humor	237	Themes	32
Literature	289	Thriller	316
Mystery	512	Vampires	131
New Adult	702	Young Adult	117

Table 7: Distribution of books within each genre of the deduplicated Toronto Books corpus.

pipeline. First, we note that coreference resolution systems are trained on documents much smaller than full novels (Pradhan et al., 2012); to accommodate this limitation, we partition each novel into non-overlapping windows that are 100 sentences in length, yielding approximately 400,000 windows in total. We then run CoreNLP's universal dependency parser (Nivre et al., 2016; Chen and Manning, 2014), part of speech tagger (Toutanova et al., 2003), and neural coreference resolution system (Clark and Manning, 2016a,b) over each window of text. For each window, we select the longest coreference chain and call the entity in that chain the "protagonist," following Chambers and Jurafsky (2008).

We feed the resulting universal dependency (UD) parses into PredPatt (White et al., 2016), a rule-based predicate-argument extraction system that runs over universal dependency parses. From PredPatt output, we extract predicate-argument edges, i.e., a pair of token indices in a given sentence where the first index is the head of a predicate, and the second index is the head of an argument to that predicate. Edges with non-verbal predicates are discarded.

At this stage in the pipeline, we merge information from the coreference chain and predicate-argument edges to determine which events the protagonist is participating in. For each predicate-argument edge in every sentence, we discard it if the argument index does not match the head of a protagonist mention. Each of the remaining predicate-argument edges therefore represents an event that the protagonist participated in.

With a list of PredPatt-determined predicate-argument edges (and their corresponding sentences), we are now able to extract the narrative event representations, (p, d) For p, we take the lemma of the (verbal) predicate head. For d, we take the dependency relation type (e.g., *nsubj*) between the predicate head and argument head indices (as determined by the UD parse); if a direct arc relation does not exist, we instead take the unidirectional dependency path from predicate to argument; if a unidirectional path does not exist, we use a generic "arg" relation.

To extract a factuality feature for each narrative event (i.e. whether the event happened or not, according to the meaning of the text), we use the neural model of Rudinger et al. (2018a).As input to this model, we provide the full sentence in which the event appears, as well as the index of the event predicate's head token. The model returns a factuality score on a $[-3, 3]$ scale, which is then discretized using the following intervals: $[1, 3]$ is "positive" $(+)$, $(-1, 1)$ is "uncertain," and $[-3, -1]$ is "negative" $(-)$.

From this extraction pipeline, we yield one sequence of narrative events (i.e., narrative chain) per text window.

A.3 Training and Model Details - Causal Model

A.3.1 RNN Encoder

We use a single layer GRU based RNN encoder with a 300 dimensional hidden state and 300 dimensional input event embeddings to encode the previous events into a single 300 dimensional vector.

A.3.2 CNN Encoder

We use a CNN to encode the text into a 300 dimensional output vector. The CNN uses 4 filters with ngram windows of $(2, 3, 4, 5)$ and max pooling.

A.3.3 Training Details - Pretraining

The conditional for the Causal model is trained using Adam with a learning rate of 0.001, gradient clipping at 10, and a batch size of 512. The model is trained to minimize cross entropy loss. We train the model until loss on the validation set does not go down after three epochs, afterwhich we keep the model with the best validation performance, which in our case was epoch 4

A.3.4 Training Details - Finetuning

The model is then finetuned on our dataset of 2000 annotated examples. We use the same objective as above, training using Adam with a learning rate of 0.00001, gradient clipping at 10, and a batch size of 512. We split our 2000 samples into a train set of

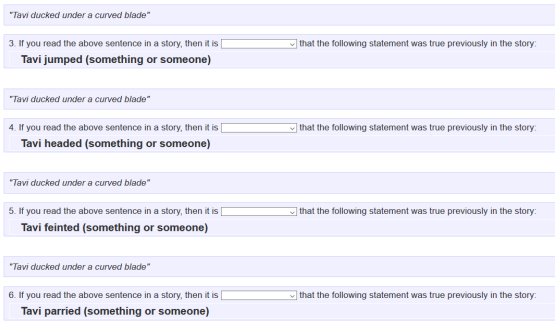

Figure 3: The annotation interface for the out-of-text events annotation.

1800 examples and a dev set of 200 examples. We train the model in a way similar to above, keeping the best validation model (at epoch 28).

A.4 Training and Model Details - LM Baseline

We use a 2 layer GRU based RNN encoder with a 512 dimensional hidden state and 300 dimensional input event embeddings as our baseline event sequence LM model.

A.4.1 Training Details

The LM model is trained using Adam with a learning rate of 0.001, gradient clipping at 10, and a batch size of 64. We found using dropout at the embedding layer and the output layers to be helpful (with dropout probability of 0.1). The model is trained to minimize cross entropy loss. We train the model until loss on the validation set does not go down after three epochs, afterwhich we keep the model with the best validation performance, which in our case was epoch 5.

A.5 Annotation Interfaces

To get an idea for about the annotation set ups used here, we also provide screen shots of the annotation suites for all three annotation experiments. The out-of-text annotation experiment of Section 3.3 is shown in Figure 3. The pairwise annotation evaluation of Section 4.3 is shown in Figure 4. The chain completion annotation evaluation of Section 4.4 is shown in Figure 5.

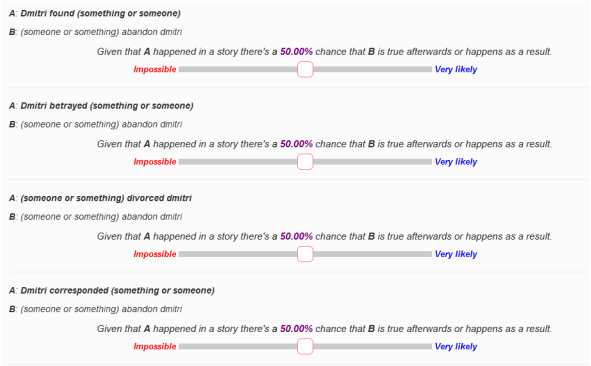

Figure 4: The annotation interface for the pairwise human evaluation annotation experiment.

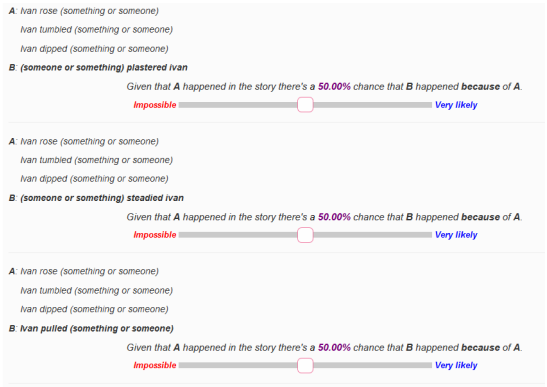

Figure 5: The annotation interface for the chain completion human evaluation annotation experiment.