Advances in Cognitive Systems 2 (2012) 25–42 Submitted 9/2012; published 12/2012

Crowdsourcing Narrative Intelligence

Boyang Li BOYANGLI@GATECH.EDU
Stephen Lee-Urban LEE-URBAN@GATECH.EDU
Darren Scott Appling DARREN.SCOTT.APPLING@GATECH.EDU
Mark O. Riedl RIEDL@GATECH.EDU
School of Interactive Computing, Georgia Institute of Technology, Atlanta, GA 30332 USA

Abstract

Narrative intelligence is an important part of human cognition, especially in sensemaking and communicating with people. Humans draw on a lifetime of relevant experiences to explain stories, to tell stories, and to help choose the most appropriate actions in real-life settings. Manual authoring the required knowledge presents a significant bottleneck in the creation of systems demonstrating narrative intelligence. In this paper, we describe a novel technique for automatically learning script-like narrative knowledge from crowdsourcing. By leveraging human workers' collective understanding of social and procedural constructs, we can learn a potentially unlimited range of scripts regarding how real-world situations unfold. We present quantitative evaluations of the learned primitive events and the temporal ordering of events, which suggest we can identify orderings between events with high accuracy.

1. Introduction

From ancient Greek plays to modern motion pictures, from bedtime stories to Nobel-prize-winning novels, storytelling in various forms plays a pervasive role in human culture. Cognitive and psychological research suggests that the prevalence of storytelling may be explained by the use of narrative as a cognitive tool for situated understanding (Bruner, 1991; Gerrig, 1993; Graesser, Singer, & Trabasso, 1994), as a cornerstone of one's identity (Singer, 2004), and as a means of supporting early development of language (Johnston, 2008).

We consider *narrative intelligence* as an entity's ability to organize and explain experiences in narrative terms (Mateas & Sengers, 1999), to comprehend and make inferences about narratives we are told (Graesser, Singer, & Trabasso, 1994), and to produce affective responses such as empathy to narratives (Mar et al., 2011). Narrative intelligence is central to the cognitive processes employed across a range of experiences from entertainment to learning. It follows that computational systems possessing narrative intelligence may be able to interact with human users naturally because they understand collaborative contexts as emerging narrative and are able to express themselves by telling stories.

In this paper we consider the problem of creating, telling, and understanding stories that involve common procedural and social situations. Most stories are about people; these narrative intelligence tasks require the ability to recognize and act according to social and cultural norms. Furthermore, for virtual agents and robots to co-exist and cooperate with humans the ability to explain others' behaviors and to carry out common tasks involving social contexts is important.

For example, during a trip to a restaurant, an agent should know that drinks are typically ordered before the food. Likewise, when accompanying a human to the movie theatre, one should know to purchase popcorn before finding one's seats. To omit these elements or to use them at the wrong time invites failures in believability, breakdowns in communication, or increased overhead of coordination.

Akin to Fodor's (1983) notion of central cognitive processes, narrative intelligence is highly knowledge intensive. Narrative intelligence tasks such as story understanding and story generation require the ability to recognize and act according to technical procedures as well as social and cultural norms. Humans draw on a lifetime of relevant experiences from which to explain or tell stories, and to help choose the most appropriate actions in real-life setting. However, attempts to instill computational systems with narrative intelligence have been limited by the high cost of manual coding of extensive knowledge. For example, a simple model of restaurant behavior uses 87 rules (Mueller, 2007). A simulation game about attending a prom dance (McCoy et al., 2010) requires over 5,000 rules to capture the associated social dynamics. Therefore, most narrative understanding / generation systems to date are restricted to operate within limited micro-worlds for which knowledge are provided. The ability to automatically acquire knowledge may ease this bottleneck.

We propose a novel technique to learn the knowledge needed for narrative intelligence from the stories humans tell, which encode human experiences. We obtained these stories via crowdsourcing techniques. Crowdsourcing is the outsourcing of complicated tasks—typically tasks that AI cannot perform—to a large number of anonymous workers via Web services (Quinn & Bederson, 2011). A common model of crowdsourcing is to break the complex problem into many simple sub-problems that can be completed by untrained humans quickly. In our case, the knowledge-authoring task is broken into writing many simple stories about a given situation, such as visiting a restaurant or going on a date at a movie theatre. Workers write stories in simplified natural language that include typical events for that situation. Our algorithm aggregates the stories and learns a model of the given situation. Crowdsourcing provides a means for rapidly acquiring a highly specialized corpus of examples. An intelligent system uses this specialized corpus to build a general model of the situation that can be used for narrative intelligence tasks such as story understanding, story generation, or acting in the real world. In contrast, learning from less specialized corpora, such as the Penn Treebank or Wikipedia, face the challenges that (a) information about the topics of interest do not always exist and (b) satisfactory natural language processing that can yield knowledge robust enough for real-world application is still an open problem.

We employ *scripts* (Schank & Abelson, 1977) as our knowledge representation. Script is a form of procedural knowledge that describes how common situations are expected to unfold, which can capture technical knowledge as well as social and cultural norms. Scripts are specialized forms of *schemas* or *frames* and have been found to be practical means for encoding expectations of events that occur during frequently experienced situations. Although scripts are convenient descriptions of patterns of neural activations associated with procedural behaviors (Abelson, 1981), they have been found to be useful for describing human expertise (Glaser, 1984). Much research into computational narrative understanding and narrative generation has used of manually coded script-like knowledge, such as cases or plan libraries.

The contribution of this work is threefold: (1) a framework for rapidly acquiring a specialized corpus of narrative examples in simple language about specific procedural or sociocultural situations, (2) an algorithm for turning acquired chronological sequences of events into a computable model, and (3) the demonstration of crowdsourcing as an effective means for controlling the complexity of natural language, so that an intelligent system can increase its information gain from the specialized corpus. Learning good models from small corpora, our algorithm acquires knowledge needed for narrative intelligence in an accurate, economical, and just-in-time manner. A quantitative evaluation shows the quality of the models learned on two situations. By leveraging the crowd and its collective understanding of social constructs, we can learn a potentially unlimited range of scripts regarding how humans generally believe real-world situations unfold. We seek intelligent computational systems that can apply script-like knowledge to perform narrative intelligence tasks such as understanding stories, creating new stories, and coordinating activities with humans.

2. Related Work

Story understanding systems demonstrate their capabilities by automatically processing a narrative text and then answering questions that a human could answer after reading the text (Schank & Riesbeck, 1981). Story understanding tasks include identifying atypical event sequences, inferring character goals, inferring missing events, summarization, etc. Early story understanding systems processed narrative texts by comparing them to hand-coded knowledge structures that encoded common occurrences such as scripts, frames, or schemas (cf., Schank & Riesbeck, 1981; Mueller, 2007). Automated story generation systems, on the other hand, search for a novel sequence of events that meet a given communicative objective, such as to entertain or convey a message or moral. The most common approaches to story generation are planning and case-based reasoning, as described by Gervas (2009) in an overview. We note that the task of generating stories is also knowledge-intensive and many techniques treat the problem as the assembly or adaptation of chunks of schematic knowledge.

Narrative intelligence is closely associated with commonsense reasoning. Recent work on commonsense reasoning has sought to acquire propositional knowledge from a variety of sources. LifeNet (Singh & Williams, 2003) is a commonsense knowledge base about everyday experiences constructed from 600,000 propositions asserted by the general public. However, according to Singh and Williams (2003), this technique tends to yield spotty coverage. Crowdsourcing techniques promise to address some of the sparseness issues of building commonsense knowledge bases (Kuo, Hsu, & Shih, 2012). Most commonsense reasoning systems, however, do not attempt to create script-like knowledge representations.

There has been interest in learning script-like knowledge from large-scale corpora such as news corpora and other online sources of textual information (Chambers & Jurafsky, 2008; Girju, 2003; Kasch & Oates, 2010). Unlike other natural language processing techniques that learn correlations between sentences, these systems attempt to find relationships between many events. In particular, Chambers and Jurafsky (2008) attempt to identify related sentences and learn their partial ordering.

Gordon et al. (2011) describe an approach to mining causal relations from millions of personal webblog stories, under the expectation that this corpus would contain, by virtue of scale, causality information for everyday situations. They note the challenges associated with extracting causal, commonsense information from such a corpus and also note that increasing the size of the corpus from one million to ten million produced statistically insignificant improvements. Gordon et al. further suggest that causal information in stories from these sources is best left implicit, and that the ability to select between causal relations does not constitute a full solution to open-domain commonsense causal reasoning.

While large-scale corpus-based script learning can be very powerful, the results from the above researchers suggest that it suffers from four notable limitations. First, the topic of the script to be learned must be represented in the corpus. For example, one would not expect to learn the script for how to go on a date to a movie theatre from a news article corpus. Unfortunately, many existing corpora are written for human readers and lack the level of detail required by computer algorithms. Second, given a topic, a system must determine which events are relevant to the script when there may be many interleaved situations and topics. Third, corpora written for human consumption may omit canonical events under the assumption that humans are familiar with the situation. This may create a problem when one wishes to computationally learn a sociocultural norm as a script. Compared to a general-purpose corpus, crowdsourcing is beneficial in creating a highly specialized corpus that contains the exact information to be learned with reduced noise. Finally, and compounding the first three problems, extracting information from unconstrained natural language remains a challenging problem.

Our work shares similarities with other crowdsourcing techniques. Jung et al. (2010) extract procedural knowledge from eHow.com and wikiHow.com, which provide crowdsourced how-to instructions for a wide range of topics. However, these resources are written for human readers and still have poor coverage of the most common situations and use complex language that is difficult for current technologies.

In *The Restaurant Game*, Orkin and Roy (2009) use traces of people in a virtual restaurant to learn a probabilistic model of restaurant activity. Because *The Restaurant Game* is an existing virtual game, Orkin and Roy have an *a priori* known set of actions that can occur in restaurants (e.g., sit down, order, etc.). *SayAnything* (Swanson & Gordon, 2008) is a system that co-creates stories with human assistance by mining events from Weblogs and thus does not require a fixed domain model. Human players provide every other sentence, which helps to retain story coherence, whereas we believe our generalization from stories to scripts provides the necessary context to preserve coherence.

3. Crowdsourcing Narrative Examples

Whereas humans have a lifetime of experiences from which to construct correct and functional models, our system must rapidly acquire experiences and learn from them. Crowdsourcing provides a means for accessing humans' distributed memories of relevant real-world experiences. We hire anonymous workers on the crowdsourcing platform of Amazon Mechanical Turk (AMT). Given the name of a situation, e.g., going to a restaurant, going on a date to a movie theatre, or attending a wedding, our system uses a three-stage process to construct a general model of the situation, which is represented as a branching script.

1. We ask crowd workers to provide linear, natural language narratives of the given situation. This set of linear narratives may be considered as a way to rapidly experience relevant situations. For lay workers, providing step-by-step narratives is a more intuitive means to convey information than manipulating complex graphical structures.

2. We identify the events—primitive activities that comprise the script. We do not assume the existence of a set of known actions or events. Instead, we identify sentences in the

crowdsourced narratives that describe the same activities, which are extracted as primitive events in the situation.

3. We construct the script, which establishes a partial order between the events. The second and third steps together comprise a form of learning by demonstration, where the primitive actions are *a priori* unknown.

The crowdsourcing paradigm usually involves breaking a complex task into simple subtasks. In this case, we simplify knowledge authoring into writing short narratives about a given situation. Knowledge is then learned from aggregating these narratives. This is in contrast to, for example, letting workers write production rules or manipulate complex graphical models. We do this for two reasons. First, simplifying the task maximizes the number of potential participants and lowers the cost of hiring. We believe telling stories is a natural mode of communication that most people are capable of. Second, in order to learn about special or rare situations such as submarine accidents, it may be necessary to collect stories from experts. Telling stories is found to be an effective means for human experts to share *tacit knowledge* (Hedlund, Antonakis, & Sternberg 2002), which is procedural knowledge that is hard to articulate even for experts. Thus, we believe this form of data collection appeals to both ordinary workers and domain experts.

Our approach starts with publishing a number of tasks on Amazon Mechanical Turk, each of which requests a crowd worker to write a short narrative about a particular situation. After some time, a small, highly specialized corpus, containing examples of how the situation can unfold, is collected. The crowdsourced corpus facilitates subsequent learning for two reasons: One, the corpus contains highly specialized information about a specific situation. Two, we can guide the crowd workers in how best to communicate their knowledge. To reduce our reliance on immature natural language technologies, we ask crowd workers to:

- Use proper names for all the characters in the task. This allows us to avoid co-reference resolution altogether. We provide a cast of characters for common roles, e.g., for the task of going to a fast-food restaurant, we provide named characters in the role of the restaurant patron, the cashier, etc. Currently, these roles must be hand-specified, although we envision future work where the roles are extracted from commonsense knowledge bases.
- Segment the narrative into events such that each sentence contains a single activity.
- Use simple natural language such as using one verb per sentence, avoiding conditionals, complex, and compound sentences.

We refer to each segmented activity as a *step*. Figure 1 shows two narratives about the same situation.

Once a corpus of narrative examples for a specific situation is collected from the crowd, we begin the task of learning a script. Our computational script representation differs from prior work on script-based systems. We represent a script as a set of *before* relations, $B(e_1, e_2)$ for all events e_1 and e_2 signifying when e_1 must strictly occur before e_2. These relations coincide with causal and temporal precedence information, which are important for narrative comprehension (Trabasso & Sperry, 1985; Graesser, Singer, & Trabasso, 1994). A set of *before* relations allows for partial ordering, which can allow for variations in legal event sequences for the situation. Figure 2 shows a visualization of a script as a set of *before* relations between events. Note the support for variations and alternatives. When events in a script belong to different variations of

Story A	**Story B**
a. John drives to the restaurant.	a. Mary looks at the menu.
b. John stands in line.	b. Mary decides what to order.
c. John orders food.	c. Mary orders a burger.
d. John waits for his food.	d. Mary finds a seat.
e. John sits down.	e. Mary eats her burger.
f. John eats the food.	…
…	

Figure 1. Fragments of crowdsourced narratives.

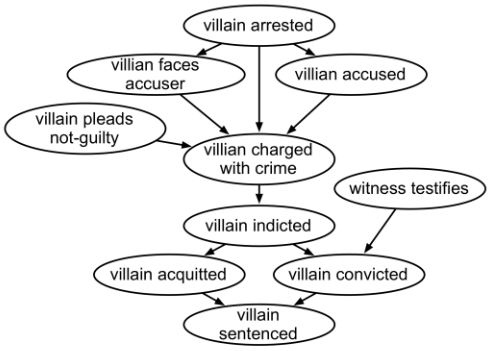

Figure 2. A criminal trial script, adapted from Chambers and Jurafsky (2009).

the same situation, underlying statistical data determines when events should never co-occur in the same narrative. The tasks of learning the main events that occur in the situation and learning the ordering of events are described in the following sections.

4. Event Learning

Event learning is a process of determining the primitive units of action to be included in the script. By working from natural language descriptions of situations, we learn the salient concepts used by a society to represent and reason about common situations. We must overcome several challenges: (a) the same step may be described in different ways; (b) some steps may be omitted by some workers; (c) a task may be performed in different ways and therefore narratives may have different steps, or the same steps but in a different order. Our approach is to automatically cluster steps from the narratives based on semantic similarity such that clusters come to represent the consensus set of events that should be part of the script. There are many possible ways to cluster sentences based on semantic similarity; below we present the technique that leverages the simple language encouraged by our crowdsourcing technique.

4.1 Sentence Similarity

Following Lintean and Rus (2010), we compute the similarity between two sentences as the similarity between their grammatical structures. We use the Stanford parser (Klein & Manning, 2003) to extract the grammatical structure of a sentence as a directed graph (by collapsing and propagating relations in the basic tree structure). Each edge on the graph describes a grammatical relation involving two words. One word is designated as the governor of the relation and the other word is the dependent. Each relation also has a type. When two grammatical relations are of different types, their similarity is zero. When the two relations belong to the same type, the similarity is the average of the word similarity between the governors and the word similarity between the dependents. The semantic similarity between two words is computed based on WordNet (Miller, 1995). Empirically, we found the Resnik (1995) word similarity function to be the most useful.

After pairwise similarities between grammatical relations from two sentences are computed, the maximum matching between the relations is found using the Hungarian algorithm. For the pairs in the best matching, we directly sum their similarity as the similarity between two sentences. More formally, sentence A and sentence B are described with a set of relations $R_A = \{a_1, a_2, \ldots, a_n\}$ and $R_B = \{b_1, b_2, \ldots, b_m\}$. A matching $M \subset R_A \times R_B$ pairs one element in A with at most one element in B. The maximum matching M^* maximizes $\sum_{(i,j)\in M} s(i,j)$, where $s(i,j)$ is the similarity between two grammatically relations. The similarity between the two sentences is thus $\sum_{(i,j)\in M^*} s(i,j)$.

Finally, we rely on event location—a step's location as the percentage of the way through a narrative—to disambiguate semantically similar steps that happen at different times, especially when a situation is highly linear with little variation. For example, when going to a movie theatre, one will "wait in line" to buy tickets and then may "wait in line" to buy popcorn. These two activities may share many grammatical similarities, but will differ in their locations in the narrative. The similarity between two sentences is a weighted sum of grammatical similarity and location similarity.

4.2 Event Clustering

We identify events that frequently occur in a given situation by clustering similar steps in collected narratives, using the similarity measure computed above. The resultant clusters are the events that can occur in the given situation. We use OPTICS (Ankerst et al., 1999) as our clustering algorithm due to its robustness under noisy input and capability to detect clusters of different shapes and density. Noise arises from human performance and from computational errors (i.e. imperfect similarity measures and imperfect grammatical parsing). The system employs density-based clustering, which is based on the intuition that a cluster is formed when a number of points are close to one another. Thus, OPTICS requires one parameter, the minimum size of a cluster C_m, which is the minimum number of points needed for a cluster to be recognized. We extract the leaf clusters from the hierarchical clustering structure produced by the system.

4.3 Experiments and Results

To evaluate our event learning algorithm, we collected two sets of narratives for the following situations: going to a fast food restaurant, and taking a date to a movie theater. While restaurant activity is a fairly standard situation for story understanding, the movie date situation is meant to be a more accurate test of the range of socio-cultural constructs that our system can learn. Our experience suggests that on AMT, we can hire a worker to write a story for roughly \$0.60. Table 1 shows the attributes of each crowdsourced corpus.

For each situation, we manually created a gold standard set of clusters against which to calculate precision and recall. Table 2 presents the results of event learning on our two crowdsourced corpora, using the MUC6 cluster scoring scheme (Vilain et al., 1995) to match actual cluster results against the gold standard. The purity of a cluster measures intra-cluster homogeneity. Higher purity indicates higher cluster quality. For a single cluster C_j, it is defined as maximum portion of sentences in C_j that actually belong to the same class:

$$purity(C_j) = \frac{1}{|C_j|} \max_i \left|\{s \in C_j | label(s) = i\}\right| ,$$

Table 1. Characteristics of the crowdsourced data sets.

Situation	**Number of stories**	**Mean number of steps**	**Unique verbs**	**Unique nouns**
Fast food restaurant	30	7.6	55	44
Movie theatre date	68	11	105	99

where i denotes the gold standard class label. The overall purity over all clusters is defined as a weighted average:

$$purity = \frac{1}{N}\sum_{j}|C_j|\, purity(C_j)\,,$$

where N is the total number of sentences.

Table 2 shows the quality of clusters in terms of precision, recall, F1 score, and purity. We compare the results obtained by using only the semantic similarity between sentences and results after the relative location of steps in crowdsourced narratives are also incorporated. It can be seen that the location information improves the clustering accuracy.

The two data sets contain some significant differences, which led to difference in performance. The movie date corpus has a significantly greater number of unique verbs and nouns, longer narratives, and greater usage of colloquial language. Interestingly, the movie date corpus contains a number of non-prototypical events about social interactions (e.g., John and Sally hug) that rarely appear. We have configured OPTICS to conservatively identify clusters, resulting in a large number of outliers. This has the effect of creating pure clusters at the expense of recall. This is more pronounced in the movie data because of the greater variation in language usage.

4.4 Improving Event Clustering with Crowdsourcing

While our event learning process achieves acceptably high accuracy rates, errors in event clustering may impact overall script learning performance. To improve event-clustering accuracy, we can adopt a technique to improve cluster quality using a second round of crowdsourcing, similar to that proposed by Boujarwah, Abowd, and Arriaga (2012). Workers are tasked with inspecting the members of a cluster and marking those that do not belong. Under sufficient agreement, a particular step can be removed from its cluster. Next, workers are tasked to identify which cluster these “un-clustered” steps should be placed into. Crowdsourcing is often used to improve on artificial intelligence results and we hypothesize that we can increase clustering accuracy to near perfect in this way. However, in the long term our goal is minimize the use of the crowd so as to speed up script acquisition and reduce costs. This stage of our framework is currently under development.

5. Script Learning

Once we have the events that can occur during a given situation, the next stage is to learn the script structure. We learn *before* relations (e.g., *before*(e_1, e_2)) between all pairs of events e_1 and e_2. See Figure 2 for an example visualization of a script as a graph. Chambers and Jurafsky train

Table 2. Precision, recall, F1, and purity scores for the restaurant and movie data sets.

Situation	Gold std. num. events	Semantic similarity				Semantics + Location			
		Precision	Recall	F1	Purity	Precision	Recall	F1	Purity
Fast food	21	0.879	0.649	0.746	0.831	**0.880**	**0.688**	**0.772**	**0.836**
Movie date	56	0.761	0.539	0.631	0.642	**0.837**	**0.587**	**0.690**	**0.724**

their models on the Timebank corpus (Pustejovsky et al., 2003), which uses temporal signal words. Because we are able to leverage a highly specialized corpus of narrative examples of the desired situation, we can probabilistically determine *before* relations between events directly from the crowdsourced narrative examples. This process produces a general model of expected event ordering for the given situation. Our process for script learning involves two procedures:

1. **Initial script construction**—a procedure that infers absolute ordering between events from observed orderings in crowdsourced narrative examples. Due to the inherent uncertainty and errors existing in human-authored narratives and clustering of sentences, this procedure may be sensitive to the selected parameters. It may overlook some relations or include wrong relations.
2. **Script improvement**—a heuristic procedure that restores missing *before* relations by analyzing the impact of each relation on the global script structure. To make the procedure robust against parameter selection, we use a high threshold, which leads to missing relations, together with this procedure to restore them. We optimize the script structure by minimizing an error metric, which accounts for differences in the script structure and crowdsouced narrative examples.

5.1 Initial Script Construction

Script construction is the process of identifying the script structure that most accurately explains the set of crowdsourced narratives. For every pair of events e_1 and e_2, we create two hypotheses *before*(e_1, e_2) and *before*(e_2, e_1). We count the amount of evidence for and against each hypothesis. Let s_1 be a step in the cluster that represents event e_1, and let s_2 be a step in the cluster that represents event e_2. If s_1 and s_2 appear in the same input narrative, and if s_1 appears before s_2 in the narrative, then we consider this as an observation in support of *before*(e_1, e_2). If s_2 appears before s_1 in the same narrative, this observation supports *before*(e_2, e_1).

A hypothesis *before*(e_1, e_2) is only accepted when we are sufficiently confident that the probability of e_1 appearing before e_2 is higher than 50%. We perform a one-tailed hypothesis testing based on the binomial distribution. The confidence of *before*(e_1, e_2) is thus defined as

$$confidence = \sum_{i=1}^{k} \binom{n}{i} \frac{1}{2^n},$$

where n is the number of observations supporting either *before*(e_1, e_2) or the opposite hypothesis *before*(e_2, e_1), and k is the observations that support *before*(e_1, e_2). We accept the hypothesis only if the confidence exceeds a threshold $T_p \in (0.5,1]$.

Graphically, a node represents an event and a directed edge represents a *before* relation, as in Figure 2. Our script representation requires the graph to be acyclic. We eliminate loops involving only two events by setting $T_p > 0.5$, which makes it impossible to accept both *before*(e_1, e_2) and *before*(e_2, e_1). We also forbid self-loops. However, the graph may contain loops that involve three or more events. For a simple loop that does not share edges with other loops, we break the loop by removing the lowest confidence edge in the loop. The general case of finding a minimum feedback edge set is NP-hard and APX-hard (Kann, 1992), which we do not tackle in this paper. When global threshold T_p is set to a sufficiently large value, complex loops will always be eliminated.

T_p and T_c apply to the entire graph and allow us to generate an initial estimate of the script structure through simple observation counts. In practice, we find that the graph quality is sensitive to the selection of parameters. Pre-selection of a set of parameters that always work for the entire graph is often impossible as different parts of the graph may respond well to different parameters. Thus, it is desirable for graph estimation to be robust against parameter selection, and to locally relax the global thresholds for some relations. We achieve these goals by using a high threshold for T_p and then restoring missing relations back into the script to minimize a measure of graph error, as described below.

5.2 Script Improvement

In this section, we describe a technique to improve the graph estimation by locally adjusting thresholds of *before* relation acceptance in order to better conform to the corpus data. Since a script encodes event ordering, we introduce an error measure based on the expected number of interstitial events between any pair of events. The error is the difference between two distance measures, $D_G(e_1, e_2)$ and $D_N(e_1, e_2)$. $D_G(e_1, e_2)$ is the number of events on the shortest path from e_1 to e_2 on the graph (e_1 excluded); this is also the minimum number of events that must occur between e_1 and e_2 in all legal totally ordered sequences consistent with the *before* relations of the script. In contrast, $D_N(e_1, e_2)$ is the normative distance from e_1 to e_2 averaged over the entire set of narratives. For each input narrative that includes sentence s_1 from the cluster representing e_1 and sentence s_2 from the cluster representing e_2, the distance (i.e. number of interstitial sentences plus one) between s_1 and s_2 is $d_N(s_1, s_2)$. $D_N(e_1, e_2)$ is thus the average of $d_N(s_1, s_2)$ over all such input narratives. Outlier sentences that do not belong to any events are not counted as interstitial sentences. The mean squared graph error (MSGE) for the entire graph is defined as

$$MSGE = \frac{1}{|P|} \sum_{e_1 \in P} \sum_{e_2 \in P} \left(D_G(e_1, e_2) - D_N(e_1, e_2) \right)^2 ,$$

where P is the set of all ordered event pairs (e_1, e_2) such that e_2 is reachable from e_1 or that they are unordered.

We utilize this error measure to improve the graph based on the belief that D_N represents the normative distance we expect between events in any narrative accepted by the script. That is, typical event sequences in the space of narratives described by the script should have $D_G(e_1, e_2) \approx D_N(e_1, e_2)$ for all events. A particularly large deviation from the norm may indicate that some edges with low confidence could be included in the graph to make it closer to user inputs and reduce the overall error.

We implement a greedy, iterative improvement procedure that reduces mean square graph error in a script (Table 3). For each pair of events (e_1, e_2) such that e_2 is reachable from e_1 in the

Table 3. The script graph improvement algorithm.

```
$Q$ := all of events $(e_1, e_2)$ such that $e_2$ is reachable from $e_1$ or unordered
**Foreach** $(e_1, e_2) \in Q$ in order of decreasing $D_N(e_1, e_2) - D_G(e_1, e_2)$ **do:**
  $E$ := the set of event $e_i$ that satisfy $D_G(e_1, e_i) = D_N(e_1, e_2) - 1$
  **Foreach** $e_i \in E$ **do:**
    **If** edge $e_i \rightarrow e_2$ is not in the graph and adding it to the graph will
    not create a cycle **do:**   (also shouldn't increase MSGE)
      Add $e_i \rightarrow e_2$ to the graph
**Return** graph
```

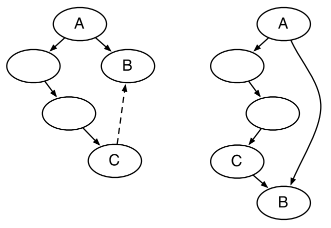

Figure 3. Compensation for errors between pairs of events. Dashed lines are low-confidence relations.

graph of directed edges, we compute a set of potential predecessor events, denoted by E. For all $e_i \in E$, if e_i were the immediate predecessor of e_2 then $D_G(e_1, e_2)$ would be equal to $D_N(e_1, e_2)$. Starting from the pairs of events with the largest deviation from the norm, computed as $D_N(e_1, e_2) - D_G(e_1, e_2)$, we check if adding an edge $e_i \rightarrow e_2$ will create any cycles or increase MSGE. If not, the edge is added to the graph. This intuition is illustrated in Figure 3 where the edge (dashed arrow) from event C to event B was originally rejected due to insufficient confidence; adding the edge to the graph creates the desired separation between events A and B. Note that adding an edge may also increase overall graph error by changing the distance between other nodes. Our improvement procedure repeats until no new changes to graph structure can be made that reduce the mean square graph error.

We find a relatively high T_p (≈ 0.7) combined with the graph improvement step leads to robust graph estimation. A conservative T_p initially discards many edges in favor of a more compact graph with many unordered events. After that, the improvement algorithm opportunistically restores the relations of different levels of evidence as long as graph error can be reduced. This effectively relaxes the threshold locally. Rare events are automatically excluded from the graphs because their relations to all other events do not meet our probability and confidence thresholds.

5.3 Experiments and Results

Figure 4 shows scripts learned for the fast food restaurant and movie theatre date situations. These plots were learned from the gold standard clusters under the assumption that a second round of crowdsourcing (as described in Section 4.4) can achieve near perfect clustering. The event labels are English interpretations of each event for presentation purposes only, based on manual inspection of the sentences in each event. For clarity, edges that do not affect the partial ordering are omitted from the figure. The asterisks in Figure 4 indicate edges that were added during graph improvement. Table 4 shows statistics for mean square graph error reduction. Over 32 sets of different parameter settings, we found that iterative graph improvement led to an average error reduction of 21.0% and 7.5% for the fast-food restaurant and movie date situations respectively. Note that it is not always possible to reduce graph errors to zero when there are plausible ordering varations between events. For example *choose menu item* and *wait in line* can happen in any order, introducing a systematic bias for any graph path across this pair.

In general, we tend to see ordered relations when we expect causal necessity, and we see unordered events when ordering variations are supported by the data. Visual inspection of the

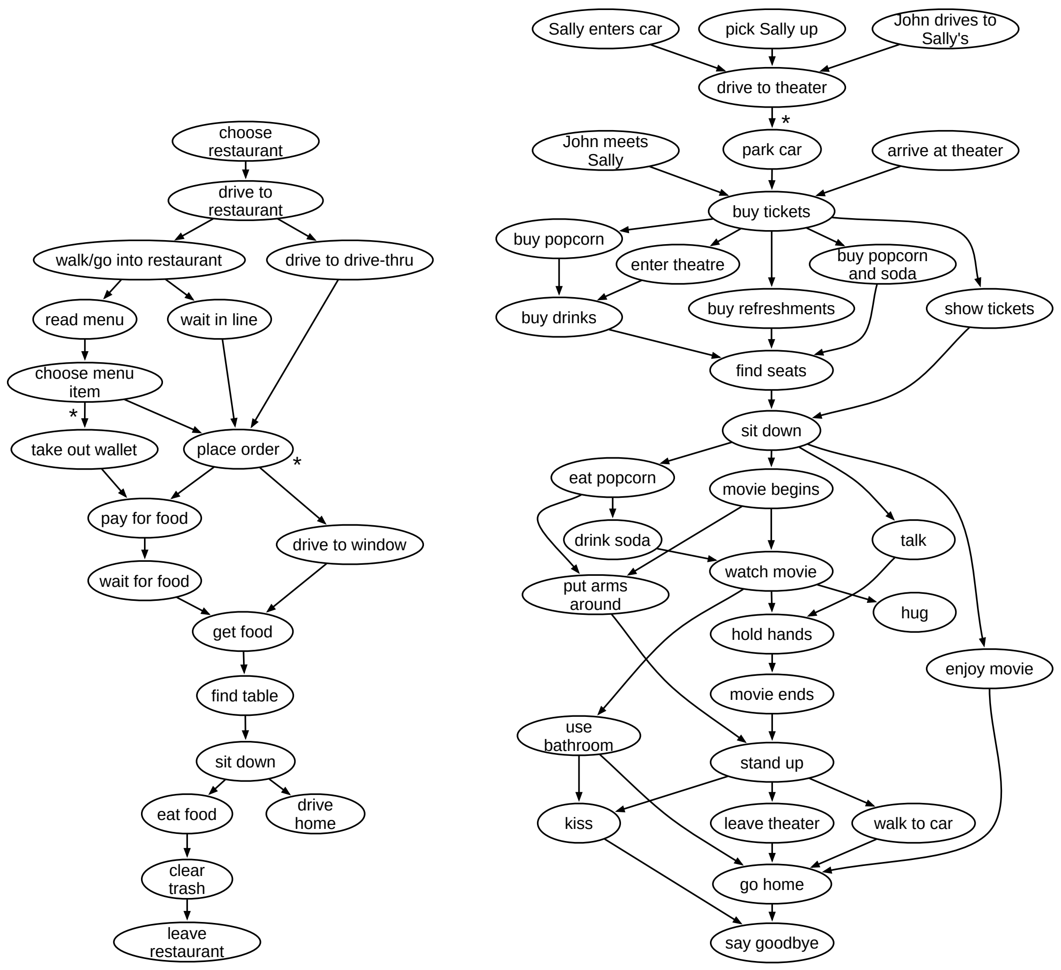

Figure 4. Scripts generated for the fast-food restaurant (left) and movie date (right) situations. Asterisks denote relations restored by the graph improvement procedure.

graphs suggests that some *before* relations are missed, especially near the beginning of the script. Our script construction algorithm errs on the side of omitting links with low probability, unless it can infer the existence of the link to reduce MSGE. We currently suffer from sparseness of data at the beginning and end of the situation because different crowd workers start and stop their examples at different points. Clustering errors can result in duplicate events.

To evaluate our script construction technique, we again crowdsource the checking of the learned script. Crowd workers are asked to check the correctedness of the learned *before* relations between events as well as the absence of such relations. For this study, we used the movie date script, shown in Figure 4 (right). We randomly sampled 30 pairs of adjacent events, i.e. events in the automatically generated script that are ordered by a *before* relation without any interstitial events. We also randomly sampled 29 pairs of parallel events, i.e. events for which the script indicates no necessary ordering relative to one another. From AMT, we recruited 144 workers. Each worker was paid $0.12-$0.20 to check seven pairs of events.

Table 4. Error reduction for the restaurant and the movie situations.

Situation	Error before Improvement		Error after Improvement		Average Error Reduction
	Avg.	Min.	Avg.	Min.	
Fast food	2.56	0.97	2.09	0.86	21.0%
Movie date	4.03	2.48	3.77	2.11	7.5%

Each worker was instructed to consider each pair of events in the context of going on a date to the move theater. Each pair of events (*A*, *B*) was presented to a worker in a randomized order (50% of workers saw *A* before *B* and 50% of workers saw the opposite) and workers were asked whether (a) it is more likely that *A* comes before *B*, (b) it is more likely that *B* comes before *A*, or (c) that they are unable to tell which should come first. In order to avoid randomly clicking behaviors, two of the seven pairs were designed as validation questions. These two pairs of events do *not* appear anywhere in the script, but were manually written and have obvious orderings. If a worker provided a wrong answer on either of two pairs, all of his or her answers were considered invalid and discarded. Each worker was allowed to participate in the study only once.

Table 5 shows the results of our study. Rows indicate subsets of the data. The first three rows show the results from all sampled pairs, all sampled adjacent pairs, and all sampled parallel pairs (the remaining rows are explained later). The columns measure accuracy—the percentage of time human workers agree with our scripts—at different levels of worker agreement. We measure human agreement on each pair of events as the entropy of their answers. The entropy for the *j*th pair of events (a_j, b_j) is defined as:

$$entropy(X_j) = -\sum_{i=1}^{3} P(x_{ji}) \ln P(x_{ji}) ,$$

where $x_{ji} \in \{before(a_j, b_j),\ before(b_j, a_j),\ parallel(a_j, b_j)\}$. The probability distribution $P(X_j)$ is observed directly from human responses for the pair (a_j, b_j). The columns of Table 5 show statistics for event pairs with increasing entropy from left to right (i.e. decreasing worker agreement). For example, the first column include only pairs where workers unanimously agree (*entropy* = 0), which are 29% of all pairs evaluted (row "All"), and of those 29%, workers agreed with the ordering in our script 76% of the time. Lowering the entropy threshold filters out pairs of events with low agreement from consideration.

We draw four sets of conclusions about our script learning algorithm in the movie situation:

- **Overall accuracy.** Our overall accuracy is greater than 53%. When we examine only pairs for which workers perfectly agree with each other our accuracy is as high as 76%, although this only accounts for about 29% of our total sampled pairs. We found that when humans could not reach consensus on a pair of events, they tend to also disagree with our system.
- **Adjacent events.** Our system is very accurate when it comes to determining when a *before* relation should exist between a pair of events. Workers agree with our *before* relations at or above 90% of the time when they can reach good consensus (*entropy* < 0.6). This suggests our algorithm is a good model of the ground truth. Accuracy remains high (≈ 0.7-0.8) even where workers tend to disagree.

Table 5. Results of the script accuracy study.

	***entropy* = 0**		***entropy* < 0.4**		***entropy* < 0.6**		***entropy* < 0.8**		***entropy* < ∞**	
	acc.	**% pairs**	**acc.**	**% pairs**	**acc.**	**% pairs**	**acc.**	**% pairs**	**acc.**	**% pairs**
All	**0.76**	0.29	0.64	0.42	0.66	0.54	0.54	0.78	0.53	1.00
Adjacent	**1.00**	0.40	0.93	0.50	0.90	0.67	0.82	0.73	0.70	1.00
Parallel	0.20	0.17	0.20	0.34	0.25	0.41	0.22	0.79	**0.28**	1.00
All-sans-ends	**0.80**	0.24	0.73	0.37	0.68	0.46	0.48	0.76	0.49	1.00
Adjacent-sans-ends	**1.00**	0.35	0.50	0.50	0.83	0.60	0.69	0.80	0.60	1.00
Parallel-sans-ends	0.33	0.14	0.40	0.24	**0.43**	0.33	0.25	0.76	0.33	1.00

- **Parallel events.** For all parallel pairs, workers agreed with our system only 28% of the time. However, workers agreement is generally lower for parallel events than adjacent events. Unanimous agreement can be reached on only 17% of all pairs, in contrast to 40% for adjacent pairs. The lack of agreement on many of these pairs suggests insufficient collective social expectation of the orderings. The reason that individual worker may prefer one ordering to another may be attributed to the way questions were asked (which ordering is more likely). Even though one ordering is likely, the other ordering may be also possible. Our results suggest that although we are missing before relations that would eliminate parallel events our system may be correctly placing events as parallel in the graph when there is very little agreement on ordering.
- **Removing events with sparse data.** The last three rows of Table 5 show the results when we remove all pairs involving events before "buy tickets" and three events at the end: "go home", "walk to car", and "say goodbye" from the data. As people start and end their example narratives at different points, data about these events are more sparse than rest of the script. This leads to lower confidence in event orderings and a high degree of parallelism between events. Our results confirm our observation: when we eliminate these events at the beginning and the end, accuracy increases 13%-20% for parallel pairs.

We further note that our system may utilize an active learning scheme similar to this evaluation methodology. To improve a script, the system can potentially seek worker feedback about ordering between events for which the system has low confidence.

6. Limitations

Most social situations contain some choices which lead to common, disparate variations of situations. Our script learning technique does not yet distinguish between alternative variations,. As a result, a script can contain events that should not occur in a single instance of the same situation. For example, our fast-food restaurant script contains events from both the drive-through situation and the eating-in situation, and one would expect that "driving up to the window" would preclude "sitting down in the restaurant". Correlation statistics, such as mutual information between unordered events, can be used to detect mutually exclusive and optional events, although our work on this is at a preliminary stage.

The kinds of stories humans find interesting are usually those that deviate from the norm. Our current approach would find it challenging to capture these uncommon variations of a situation. This is due to the requirement of statistical significance in deriving before relations. However, the learned model of ordinary situations can act as a stepping-stone for further learning of extraordinary variations. Such learning may happen in the form of querying the crowd for interesting variations to our model. For example Boujarwah et al. (2012) query the crowd for ways in which scripts can be violated. Alternatively, our model might guide the parsing of and learning from a wider, general-purpose natural text corpora that, as noted in Section 2, are more likely to naturally contain interesting script deviations.

Currently we have fairly restrictive constraints on the input narratives. Specifically, we require that (1) events are described in a strictly chronological order and (2) all stories describe the same situation. These constraints may be also relaxed by bootstrapping further learning with models learned with our approach. Advances in natural language understanding can help relax the constraints on natural language.

Closely inspecting Figure 4, we note that *before* relations sometimes appear to capture causal necessity and other times merely temporal ordering. We hypothesize that our *before* relations *approximate* causal knowledge in the same way that humans heuristically reason about causality. In the general case, causation cannot be concluded based on mere correlation, and counterfactual interventions (e.g. observing sunrise after the rooster is slaughtered) are required to strictly determine causal relations (Pearl, 2010). However, Barthes (1975), an influential narratologist, notes that *when reading a story* causal relations between events can be inferred simply by co-occurrence and the explicit temporal ordering of the events. Storytellers avoid tangential events, essentially filtering out correlations that are not also causal. This provides justification that learning from crowdsourced narrative examples can be an effective means of learning by demonstration. More causal knowledge, if needed, may be queried from crowd workers with questions about counterfactuals as similar to Trabasso and Sperry (1985).

7. Conclusions

We have demonstrated that crowdsourcing can provide an intelligent system with direct access to the rich set of experiences possessed by humans. The system we describe in this paper is able to learn from those experiences to create procedural scripts about sociocultural situations that can then be applied to narrative intelligence tasks such as understanding stories, creating new stories, or coordinating activity with humans. Crowdsourcing provides an effective means to filter irrelevant information, segment narratives into individual steps, and control the complexity of natural language. This provides an advantage over learning from general-purpose corpora. Capitalizes on these advantages, we are able to learn both the primitive events from the segmented natural language and ordering constraints on these events.

Our evaluation suggests that our system achieves high accuracy at identifying the primitive events of a situation. Further, our system is good at determining *before* relations between events, as agreed by crowd workers. While it does omit ordering constraints, there tend to be many events for which there is no collectively agreed ordering.

Script learning overcomes one of the primary bottlenecks in acquiring procedural and sociocultural knowledge required for tasks of narrative intelligence. Our approach makes it possible to extend narrative intelligence of computational systems beyond a single, handcrafted micro-world. One of the strengths of our approach is the way in which we can leverage shared social constructs acquired directly from humans. Our approach learns the events that make up common situations directly from the language people use to describe those situations; event ordering captures shared social and cultural understanding based on people's descriptions of experiences.

Acknowledgements

The authors gratefully acknowledge the support of the U.S. Defense Advanced Research Projects Agency (DARPA) for this effort. Thanks to Alexander Zook for assistance with the evaluation.

References

Abelson, R. (1981). Psychological status of the script concept. *American Psychologist*, *36*, 715–729.

Ankerst, M., Breunig, M. M., Kriegel, H.-P., & Sander, J. (1999). OPTICS: Ordering Points To Identify the Clustering Structure. *Proceedings of the ACM SIGMOD International Conference on Management of Data* (pp. 49–60).

Barthes, R. (1975). An introduction to the structural analysis of narrative. *New Literary History*, *6*, 237–272.

Boujarwah, F., Abowd, G., & Arriaga, R. (2012). Socially computed scripts to support social problem solving skills. *Proceedings of the 2012 Conference on Human Factors in Computing Systems* (pp. 1987–1996). New York, NY: ACM.

Bruner, J. (1991). The narrative construction of reality. *Critical Inquiry*, *18*, 1–21.

Chambers, N., & Jurafsky, D. (2008). Unsupervised learning of narrative event chains. *Proceedings of the Forty-Sixth Annual Meeting of the Association for Computational Linguistics: Human Language Technologies* (pp. 789–797).

Fodor, J. A. (1983). *Modularity of mind: An essay on faculty psychology*. Cambridge, MA: MIT Press.

Gerrig, R. (1993). *Experiencing narrative worlds: On the psychological activities of reading*. New Haven, CT: Yale University Press.

Gervas, P. (2009). Computational approaches to storytelling and creativity. *AI Magazine*, *30*, 49–62.

Girju, R. (2003). Automatic detection of causal relations for question answering. *Proceedings of the ACL 2003 Workshop on Multilingual Summarization and Question Answering—Machine Learning and Beyond*.

Glaser, R. (1984). Education and thinking. *American Psychologist*, *39*, 93–104.

Gordon, A., Bejan, C.A., & Sagae, K. (2011). Commonsense causal reasoning using millions of personal stories. *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*. San Francisco: AAAI Press.

Graesser, A., Singer, M. & Trabasso, T. (1994). Constructing inferences during narrative text comprehension. *Psychological Review*, *101*, 371–395.

Hedlund, J., Antonakis, J., & Sternberg, R.J. (2002). *Tacit knowledge and practical intelligence: Understanding the lessons of experience* (ARI Research Note 2003-04). United States Army Research Institute for the Behavioral and Social Sciences, Fort Belvoir, VA.

Johnston, J. (2008) Narratives: Twenty-five years later. *Topics in Language Disorders. 28*, 93–98.

Jung, Y., Ryu, J., Kim, K.-M., & Myaeng, S.-H. (2010). Automatic construction of a large-scale situation ontology by mining how-to instructions from the web. *Web Semantics: Science, Services and Agents on the World Wide Web*, *8*, 110–124.

Kasch, N., & Oates, T. (2010). Mining script-like structures from the web. *Proceedings of the NAACL/HLT 2010 Workshop on Formalism and Methodology for Learning by Reading* (pp. 34–42).

Kann, V. (1992). *On the approximability of NP-complete optimization problems*. Doctoral dissertation, Department of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm, Sweden.

Klein, D., & Manning, C. (2003). Accurate unlexicalized parsing. *Proceedings of the 41st Meeting of the Association for Computational Linguistics* (pp. 423–430).

Kuo, Y-L., Hsu, J., & Shih, F. (2012). Contextual commonsense knowledge acquisition from social content by crowd-sourcing explanations. *Proceedings of the Fourth AAAI Workshop on Human Computation* (pp. 18–24).

Lintean, M. C., & Rus, V. (2010). Paraphrase identification using weighted dependencies and word semantics. *Informatica*, *34*, 19–28.

Mar, R. A., Oatley, K., Djikic, M., & Mullin, J. (2011). Emotion and narrative fiction: Interactive influences before, during, and after reading. *Cognition & Emotion*, *25*, 818–833.

Mateas, M., & Sengers, P. (1999). Narrative intelligence. *Proceedings of the AAAI Fall Symposium on Narrative Intelligence* (pp. 1–10). North Falmouth, MA: AAAI Press.

McCoy, J., Treanor, M., Samuel, B., Tearse, B., Mateas, M., & Wardrip-Fruin, N. (2010). Comme il Faut 2: A fully realized model for socially-oriented gameplay. *Proceedings of the Third Workshop on Intelligent Narrative Technologies*.

Miller, G. (1995). WordNet: A lexical database for English. *Communications of the ACM*, *38*, 39–41.

Mueller, E. T. (2007). Modelling space and time in narratives about restaurants. *Literary and Linguistic Computing*, *22*, 67–84.

Orkin J., & Roy, D. (2009). Automatic learning and generation of social behavior from collective human gameplay. *Proceedings of the Eighth International Conference on Autonomous Agents and Multiagent Systems* (pp. 385–392).

Pearl, J. (2010). The foundations of causal inference. *Sociological Methodology*, *40*, 75–149.

Pustejovsky, J., Hanks, P. Saurí, R., See, A., Gaizauskas, R., Setzer, A. Radev, D. Sundheim, B., Day, D. Ferro, L., & Lazo, M. (2003). The TIMEBANK corpus. *Proceedings of Corpus Linguistics 2003* (pp.647–656).

Quinn, A. J., & Bederson, B. B. (2011). Human computation: A survey and taxonomy of a growing field. *Proceedings of The ACM SIGCHI Conference on Human Factors in Computing Systems* (pp. 1403–1412).

Resnik, P. (1995). Using information content to evaluate semantic similarity in a taxonomy. *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence* (pp. 448–453). Montreal, Quebec, Canada: Morgan Kaufmann Publishers Inc.

Schank, R. & Abelson, R. (1977). *Scripts, plans, goals, and understanding: An inquiry into human knowledge structures*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Schank, R. and Riesbeck, C. (Eds). (1981). *Inside computer understanding: Five programs plus miniatures*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Singer, J. A. (2004). Narrative identity and meaning making across the adult lifespan: An introduction. *Journal of Personality*, *72*, 437–460.

Singh, P., & Williams, W. (2003). LifeNet: A propositional model of ordinary human activity. *Proceedings of the Workshop on Distributed and Collaborative Knowledge Capture*. Sanibel Island, FL.

Swanson, R., & Gordon, A. (2008). Say anything: A massively collaborative open domain story writing companion. *Proceedings of the First International Conference on Interactive Digital Storytelling* (pp. 32–40).

Trabasso, T., & Sperry, L. (1985). Causal relatedness and importance of story events. *Journal of Memory and Language*, *24*, 595–611.

Vilain, M., Burger, J., Aberdeen, J., Connolly, D., & Hirschman, L. (1995). A model-theoretic coreference scoring scheme. *Proceedings of the Sixth Conference on Message Understanding* (pp. 45–52).