Learning to predict script events from domain-specific text

Rachel Rudinger[1], **Vera Demberg**[3], **Ashutosh Modi**[3],
Benjamin Van Durme[1,2], **Manfred Pinkal**[3]
Center for Language and Speech Processing[1]
Human Language Technology Center of Excellence[2]
Johns Hopkins University, Baltimore, MD USA
MMCI Cluster of Excellence[3]
Saarland University, Saarbrücken, Germany
rudinger@jhu.edu, {vera,pinkal}@coli.uni-saarland.de,
amodi@mmci.uni-saarland.de, vandurme@cs.jhu.edu

Abstract

The automatic induction of scripts (Schank and Abelson, 1977) has been the focus of many recent works. In this paper, we employ a variety of these methods to learn Schank and Abelson's canonical *restaurant script*, using a novel dataset of restaurant narratives we have compiled from a website called "Dinners from Hell." Our models learn *narrative chains*, script-like structures that we evaluate with the "narrative cloze" task (Chambers and Jurafsky, 2008).

1 Introduction

A well-known theory from the intersection of psychology and artificial intelligence posits that humans organize certain kinds of general knowledge in the form of *scripts*, or common sequences of events (Schank and Abelson, 1977). Though many early AI systems employed hand-encoded scripts, more recent work has attempted to induce scripts with automatic and scalable techniques. In particular, several related techniques approach the problem of script induction as one of learning *narrative chains* from text corpora (Chambers and Jurafsky, 2008; Chambers and Jurafsky, 2009; Jans et al., 2012; Pichotta and Mooney, 2014). These statistical approaches have focused on open-domain script acquisition, in which a large number of scripts may be learned, but the acquisition of any particular set of scripts is not guaranteed. For many specialized applications, however, knowledge of a few relevant scripts may be more useful than knowledge of many irrelevant scripts. With this scenario in mind, we attempt to learn the famous "restaurant script" (Schank and Abelson, 1977) by applying the aforementioned narrative chain learning methods to a specialized corpus of dinner narratives we compile from the website "Dinners from Hell." Our results suggest that applying these techniques to a domain-specific dataset may be reasonable way to learn domain-specific scripts.

2 Background

Previous work in the automatic induction of scripts or script-like structures has taken a number of different approaches. Regneri et al. (2010) attempt to learn the structure of specific scripts by eliciting event sequence descriptions (ESDs) from humans to which they apply multiple sequence alignment (MSA) to yield one global structure per script. (Orr et al. (2014) learn similar structures in a probabilistic framework with Hidden Markov Models.) Although Regneri et al. (2010), like us, are concerned with learning pre-specified scripts, our approach is different in that we apply unsupervised techniques to scenario-specific collections of natural, pre-existing texts.

Note that while the applicability of our approach to script learning may appear limited to domains for which a corpus conveniently already exists, previous work demonstrates the feasibility of assembling such a corpus by automatically retrieving relevant documents from a larger collection. For example, Chambers and Jurafsky (2011) use information retrieval techniques to gather a small number of bombing-related documents from the Gigaword corpus, which they successfully use to learn a MUC-style (Sundheim, 1991) information extraction tem-

*Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics (*SEM 2015)*, pages 205–210,
Denver, Colorado, June 4–5, 2015.

plate for bombing events.

Following the work of Church and Hanks (1990) in learning word associations via mutual information, and the DIRT system introduced by Lin and Pantel (2001), Chambers and Jurafsky (2008) propose a PMI-based system for learning script-like structures called *narrative chains*. Several follow-up papers introduce variations and improvements on this original model for learning narrative chains (Chambers and Jurafsky, 2009; Jans et al., 2012; Pichotta and Mooney, 2014). It is from this body of work that we borrow techniques to apply to the Dinners from Hell dataset.

As defined by Chambers and Jurafsky (2008), a narrative chain is "a partially ordered set of narrative events that share a common actor," where a *narrative event* is "a tuple of an event (most simply a verb) and its participants, represented as typed dependencies." To learn narrative chains from text, Chambers and Jurafsky extract chains of narrative events linked by a common coreferent within a document. For example, the sentence "John drove to the store where he bought some ice cream." would generate two narrative events corresponding to the protagonist John: (DRIVE, *nsubj*) followed by (BUY, *nsubj*). Over these extracted chains of narrative events, pointwise mutual information (PMI) is computed between all pairs of events. These PMI scores are then used to predict missing events from such chains, i.e. the *narrative cloze* evaluation.

Jans et al. (2012) expand on this approach, introducing an ordered PMI model, a bigram probability model, skip n-gram counting methods, coreference chain selection, and an alternative scoring metric (recall at 50). Their bigram probability model outperforms the original PMI model on the narrative cloze task under many conditions. Pichotta and Mooney (2014) introduce an extended notion of narrative event that includes information about subjects and objects. They also introduce a competitive "unigram model" as a baseline for the narrative cloze task.

To learn the restaurant script from our dataset, we implement the models of Chambers and Jurafsky (2008) and Jans et al. (2012), as well as the unigram baseline of Pichotta and Mooney (2014). To evaluate our success in learning the restaurant script, we perform a modified version of the narrative cloze task, predicting only verbs that we annotate as "restaurant script-relevant" and comparing the performance of each model. Note that these annotations are not used for training.

3 Methods

This section provides an overview of each of the different methods and parameter settings we employ to learn narrative chains from the Dinners from Hell corpus, starting with the original model (Chambers and Jurafsky, 2008) and extending to the modifications of Jans et al. (2012). As part of this work, we are releasing a program called NaChos, our integrated Python implementation of each of the methods for learning narrative chains described in this section.[1]

3.1 Counting methods for PMI

Formally, a narrative event, $e := (v, d)$, is a verb, v, paired with a typed dependency (De Marneffe et al., 2006), d, defining the role a "protagonist" (coreference mention) plays in an event (verb). The main computational component of learning narrative chains in Chambers and Jurafsky's model is to learn the pointwise mutual information for any pair of narrative events:

$$pmi(e_1, e_2) := \log \frac{C(e_1, e_2)}{C(e_1, *)C(*, e_2)} \quad (1)$$

where $C(e_1, e_2)$ is the number of times that narrative events e_1 and e_2 "co-occur" and

$$C(e, *) := \sum_{e'} C(e, e') \quad (2)$$

Chambers and Jurafsky define $C(e_1, e_2)$ as "the number of times the two events e_1 and e_2 had a coreferring entity filling the values of the dependencies d_1 and d_2." This is a symmetric value with respect to e_1 and e_2.

We implement the following counting variants:

Skip N-gram By default, $C(e_1, e_2)$ is incremented if e_1 and e_2 occur anywhere within the same chain of events derived from a single coreference chain (**skip-all**); we also implement an option to restrict the distance between e_1 and e_2 to 0 though 5 intervening events (**skip-0** through **skip-5**). (Jans et al., 2012)

[1] www.clsp.jhu.edu/people/rachel-rudinger

Coreference Chain Length The original model counts co-occurrences in **all** coreference chains; we include Jans et al. (2012)'s option to count over only the **longest** chains in each document, or to count only over chains of length 5 or greater (**long**).

Count Threshold Because PMI favors low-count events, we add an option to set $C(e_1, e_2)$ to zero for any e_1, e_2 for which $C(e_1, e_2)$ is below some threshold, T, up to 5.

3.2 Predictive Models for Narrative Cloze

In order to perform the narrative cloze task, we need a model for predicting the missing narrative event, e, from a chain of observed narrative events, $e_1 \dots e_n$, at insertion point k. The original model, proposed by Chambers and Jurafsky (2008), predicts the event that maximizes **unordered pmi**,

$$\hat{e} = \arg\max_{e \in V} \sum_{i=1}^{n} pmi(e, e_i) \quad (3)$$

where V is the set of all observed events (the vocabulary) and $C(e_1, e_2)$ is symmetric. Two additional models are introduced by Jans et al. (2012) and we use them here, as well. First, the **ordered pmi** model,

$$\hat{e} = \arg\max_{e \in V} \sum_{i=1}^{k} pmi(e_i, e) + \sum_{i=k+1}^{n} pmi(e, e_i) \quad (4)$$

where $C(e_1, e_2)$ is asymmetric, i.e., $C(e_1, e_2)$ counts only cases in which e_1 occurs *before* e_2. Second, the **bigram probability** model:

$$\hat{e} = \arg\max_{e \in V} \prod_{i=1}^{k} p(e|e_i) \prod_{i=k+1}^{n} p(e_i|e) \quad (5)$$

where $p(e_2|e_1) = \frac{C(e_1,e_2)}{C(e_1,*)}$ and $C(e_1, e_2)$ is asymmetric.

Discounting For each model, we add an option for discounting the computed scores. In the case of the two PMI-based models, we use the discount score described in Pantel and Ravichandran (2004) and used by Chambers and Jurafsky (2008). For the bigram probability model, this PMI discount score would be inappropriate, so we instead use absolute discounting.

Document Threshold We include a document threshold parameter, D, that ensures that, in any narrative cloze test, any event e that was observed during training in fewer than D distinct documents will receive a worse score (i.e. be ranked behind) any event e' whose count meets the document threshold.

4 Dataset: Dinners From Hell

The source of our data for this experiment is a blog called "Dinners From Hell"[2] where readers submit stories about their terrible restaurant experiences. For an example story, see Figure 1. To process the raw data, we stripped all HTML and other non-story content from each file and processed the remaining text with the Stanford CoreNLP pipeline version 3.3.1 (Manning et al., 2014). Of the 237 stories obtained, we manually filtered out 94 stories that were "off-topic" (e.g., letters to the webmaster, dinners not at restaurants), leaving a total of 143 stories. The average story length is 352 words.

4.1 Annotation

For the purposes of evaluation only, we hired four undergraduates to annotate every non-copular verb in each story as either corresponding to an event "related to the experience of eating in a restaurant" (e.g., *ordered* a steak), "unrelated to the experience of eating in a restaurant" (e.g., *answered* the phone), or uncertain. We used the WebAnno platform for annotation (Yimam et al., 2013).

A total of 8,202 verb (tokens) were annotated, each by three annotators. 70.3% of verbs annotated achieved 3-way agreement; 99.4% had at least 2-way agreement. After merging the annotations (simple majority vote), 30.7% of verbs were labeled as restaurant-script-related, 68.6% were labeled as restaurant-script-unrelated, and the remaining 0.7% as uncertain.

Corresponding to the 8,202 annotated verb tokens, there are 1,481 narrative events at the type level. 580 of these narrative event types were annotated as script-relevant in at least one token instance.

[2] www.dinnersfromhell.com

"A long time ago when I was still in college, my family decided to take me out for pizza on my birthday. We **decided** to try the new location for a favorite pizza chain of ours. It was all adults and there were about 8 of us, so we **ordered** 3 large pizzas. We **got** to chatting and soon realized that the pizzas should've been ready quite a bit ago, so we **called** the waitress over and she went to check on our pizzas. She did not come back. We **waited** about another 10 minutes, then called over another waitress, who went to check on our pizzas and waitress. It now been over an hour. About 10 minutes later, my Dad goes up to the check-out and asks the girl there to send the manager to our table. A few minutes later the manager comes out. He **explains** to us that our pizzas got stuck in the oven and burned. They were out of large pizza dough bread, so they were making us 6 medium pizzas for the price of 3 large pizzas. We **had** so many [pizzas] on our table we barely **had** [room] to eat! Luckily my family is pretty easy going so we just **laughed** about the whole thing. We did **tell** the manager that it would have been nice if someone, anyone, had **said** something earlier to us, instead of just disappearing, and he agreed. He even said it was his responsibility, but that he had been busy trying to fix what caused the pizzas to jam up in the oven. He went so far as to **give** us 1/2 off our bill, which was really nice. It was definitely a memorable birthday!"

Figure 1: Example story from Dinners from Hell corpus. Bold words indicate events in the "we" coreference chain (the longest chain). Boxed words (blue) indicate best narrative chain of length three (see Section 5.2); underlined words (orange) are corresponding subjects and bracketed words (green) are corresponding objects.

MODEL	AVGRNK	MRR	R@50
unigram model (baseline)	298.13	0.062	0.50
1. unordered pmi; avgrnk	**276.88**	**0.063**	0.36
2. unordered pmi; mrr	376.25	0.058	0.33
3. unordered pmi; R@50	400.36	0.050	0.50
4. ordered pmi; avgrnk	**284.68**	0.061	0.32
5. ordered pmi; mrr	381.44	0.054	0.25
6. ordered pmi; R@50	401.69	0.047	0.50
7. bigram; avgrnk	**281.07**	**0.077**	0.38
8. bigram; mrr	378.06	**0.066**	0.30
9. bigram; R@50	**271.78**	**0.084**	0.43
10. bigram disc; avgrnk	**283.01**	**0.077**	0.38
11. bigram disc; mrr	378.10	**0.067**	0.30
12. bigram disc; R@50	**271.62**	**0.089**	0.43

Figure 2: Narrative cloze evaluation. Shaded blue cells indicate which scoring metric that row's parameter settings have been optimized to. Bold numbers indicate a result that beats the baseline. Row 12 representes the best model performance overall.

ROW	SKIP	T	D	COREF	PMI DISC	ABS DISC
1	0	1	3	all	yes	N/A
2	1	3	5	long	no	N/A
3	1	5	4	longest	yes	N/A
4	0	1	3	all	yes	N/A
5	3	5	5	long	no	N/A
6	0	3	4	longest	yes	N/A
7	all	1	3	all	N/A	no
8	3	5	5	long	N/A	no
9	all	1	5	all	N/A	no
10	all	1	3	all	N/A	yes
11	3	5	5	long	N/A	yes
12	all	1	5	all	N/A	yes

Figure 3: Parameter settings corresponding to each model in Fig 2.

5 Evaluation

5.1 Narrative Cloze

We evaluate the various models on the narrative cloze task. What is different about our version of the narrative cloze task here is that we limit the cloze tests to only "interesting" events, i.e., those that have been identified as relevant to the restaurant script by our annotators (see Section 4.1).

Because our dataset is small (143 documents), we perform leave-one-out testing at the document level, training on 133 folds total. (Ten documents are excluded for a development set.) For each fold of training, we extract all of the narrative chains (mapped directly from coreference chains) in the held out test document. For each test chain, we generate one narrative cloze test per "script-relevant" event in that chain. For example, if a chain contains ten events, three of which are "script-relevant," then three cloze tests will be generated, each containing nine "observed" events. Chains with fewer than two events are excluded. In this way, we generate a total of 2,273 cloze tests.

Scoring We employ three different scoring metrics: average rank (Chambers and Jurafsky, 2008), mean reciprocal rank, and recall at 50 (Jans et al., 2012).

Baseline The baseline we use for the narrative cloze task is to rank events by frequency. This is the "unigram model" employed by Pichotta and Mooney (2014), a competitive baseline on this task.

For each model and scoring metric, we perform a complete grid search over all possible parameter settings to find the best-scoring combination on a cloze tests from a set-aside development set of ten documents. The parameter space is defined as the Cartesian product of each of the following possible parameter values: skip-n (all,0-5), coreference chain length (all, long, longest), count threshold (T=1-5), document threshold (D=1-5), and discounting (yes/no). Bigram probability with and without discounting are treated as two separate models.

Figure 2 reports the results of the narrative cloze evalutation. Each of the four models (unordered pmi, ordered pmi, bigram, and bigram with discounting) outperform the baseline on the average rank metric when the parameters are optimized for that metric. Both bigram models beat the baseline on mean reciprocal rank not only for MRR-optimized parameter settings, but for the average-rank- and recall-at-50-optimized settings. None of the parameter settings are able to ouperform the baseline on recall at 50, though both PMI models tie the baseline. Overall, the model that performs the best is the bigram probability model with discounting (row 12 of Figure 2) which has the following parameter settings: skip-all, coref-all, T=1, and D=5.

The fact that several model settings outperform an informed baseline on average rank and mean reciprocal rank indicates that these methods may in general be applicable to smaller, domain-specific corpora. Furthermore, it is apparent from the results that the bigram probability models perform better overall than PMI-based models, a finding also reported in Jans et al. (2012). This replication is futher evidence that these methods do in fact transfer.

5.2 Qualitative Example

To get a qualitative sense of the narrative events these models are learning to associate from this data, we use the conditional probabilities learned in the bigram model (Fig 2, row 12) to select the highest probability narrative chain of length three out of the 12 possible events in the "we" coreference chain in Figure 1 (bolded). The three events selected are boxed and highlighted in blue. The bigram model selects the "deciding" event (selecting restaurant) and the "having" event (having pizza), both reasonable components of the restaurant script. The third event selected is "having room," which is not part of the restaurant script. This mistake illustrates a weakness of the narrative chains model; without considering the verb's object, the model is unable to distinguish "have pizza" from "have room." Incorporating object information in future experiments, as in Pichotta and Mooney (2014), might resolve this issue, although it could introduce sparsity problems.

6 Conclusion

In this work, we describe the collection and annotation of a corpus of natural descriptions of restaurant visits from the website "Dinners from Hell." We use this dataset in an attempt to learn the restaurant script, using a variety of related methods for learning narrative chains and evaluating on the narrative cloze task. Our results suggest that it may be possible in general to use these methods on domain-specific corpora in order to learn particular scripts from a pre-specified domain, although further experiments in other domains would help bolster this conclusion. In principle, a domain-specific corpus need not come from a website like Dinners from Hell; it could instead be sub-sampled from a larger corpus, retrieved from the web, or directly elicited. Our domain-specific approach to script learning is potentially useful for specialized NLP applications that require knowledge of only a particular set of scripts.

One feature of the Dinners from Hell corpus that bears further inspection in future work is the fact that its stories contain many violations of the restaurant script. A question to investigate is whether these violations impact how the restaurant script is learned. Other avenues for future work include incorporating object information into event representations and applying domain adaptation techniques in order to leverage larger general-domain corpora.

Acknowledgments

This research was sponsored in part by the NSF under grant 0530118 (PIRE) and with additional support from the Allen Institute for Artificial Intelligence (AI2). The authors would also like to thank Michaela Regneri, Annemarie Friedrich, Stefan Thater, Alexis Palmer, Andrea Horbach, Diana Steffen, Asad Sayeed, and Frank Ferraro for their insightful contributions.

References

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*, pages 789–797, Columbus, Ohio. Association for Computational Linguistics.

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

Nathanael Chambers and Dan Jurafsky. 2011. Template-based information extraction without the templates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 976–986, Portland, Oregon, USA. Association for Computational Linguistics.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.

Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454.

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, Avignon, France. Association for Computational Linguistics.

Dekang Lin and Patrick Pantel. 2001. Dirt - discovery of inference rules from text. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 323–328. ACM.

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.

J 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 Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 321–328, Boston, Massachusetts, USA. Association for Computational Linguistics.

Karl Pichotta and Raymond Mooney. 2014. Statistical script learning with multi-argument events. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 220–229, Gothenburg, Sweden. Association for Computational Linguistics.

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

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

Beth M. Sundheim. 1991. Third message understanding evaluation and conference (muc-3): Phase 1 status report. In *Proceedings of the Message Understanding Conference*.

Seid Muhie Yimam, Iryna Gurevych, Richard Eckart de Castilho, and Chris Biemann. 2013. Webanno: A flexible, web-based and visually supported system for distributed annotations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 1–6, Sofia, Bulgaria. Association for Computational Linguistics.