Mining Script-Like Structures from the Web

Niels Kasch
Department of Computer Science
and Electrical Engineering
University of Maryland,
Baltimore County
Baltimore, MD 21250, USA
nkasch1@umbc.edu

Tim Oates
Department of Computer Science
and Electrical Engineering
University of Maryland,
Baltimore County
Baltimore, MD 21250, USA
oates@cs.umbc.edu

Abstract

This paper presents preliminary work to extract script-like structures, called events and event sets, from collections of web documents. Our approach, contrary to existing methods, is topic-driven in the sense that event sets are extracted for a specified topic. We introduce an iterative system architecture and present methods to reduce noise problems with web corpora. Preliminary results show that LSA-based event relatedness yields better event sets from web corpora than previous methods.

1 Introduction

In this paper, we present a preliminary system to extract script-like structures in a goal-directed fashion from the web. For language processing purposes, humans appear to have knowledge of many stylized situations, such as what typically happens when going to a restaurant or riding a bus. This knowledge is shared among a large part of the population and lets us predict the next step in a sequence in a familiar situation, allows us to act appropriately, and enables us to omit details when communicating with others while ensuring common ground is maintained between communication partners. It seems we have such knowledge for a vast variety of situations and scenarios, and thus natural language processing systems need access to equivalent information if they are to understand, converse, or reason about these situations.

These knowledge structures, comparable to scripts (Schank and Abelson, 1977) or narrative chains (Chambers and Jurafsky, 2008), describe typical sequences of events in a particular context. Given the number of potential scripts, their development by hand becomes a resource intensive process. In the past, some work has been devoted to automatically construct script-like structures from compiled corpora (Fujiki et al., 2003) (Chambers and Jurafsky, 2008). Such approaches, however, only produce scripts that are directly related to the topics represented in such corpora. Therefore, newspaper corpora (e.g., the Reuters Corpus) are likely to contain scripts relating to government, crime and financials, but neglect other subject areas. We present a system that extracts scripts from the web and removes the constraints of specialized corpora and domain limitations. We hope our iterative technique will produce scripts for a vast variety of topics, and has the potential to produce more complete scripts.

Another drawback of existing approaches lies with their passive extraction mechanisms. A user/system does not have the ability to obtain scripts for a specific topic, but rather is bound to obtain the most prevalent scripts for the underlying corpus. Furthermore, scripts derived in this fashion lack an absolute labeling or description of their topics. This can be problematic when a user/system is looking for specific scripts to apply to a given scenario. In contrast, our system facilitates the search for scripts given a topic. This goal oriented approach is superior in that (1) scripts are labeled by a descriptive topic and can be organized, accessed and searched by topic, (2) scripts can be constructed by topic and are not reliant on existing and potentially limiting corpora and (3) script coarseness and de-

Proceedings of the NAACL HLT 2010 First International Workshop on Formalisms and Methodology for Learning by Reading, pages 34–42,
Los Angeles, California, June 2010. ©2010 Association for Computational Linguistics

tail can be be controlled through iterative script improvement and augmentation based on additional information retrieved from the web.

2 Related Work

Lin and Pantel describe an unsupervised algorithm for discovering inference rules from text (DIRT) (Lin and Pantel, 2001a) (Lin and Pantel, 2001b). Inference rules are derived from paths in dependency trees. If two paths occur in similar contexts (i.e., the words/fillers of their slots are distributionally similar) then the meaning of the paths is similar. VerbOcean (Chklovski and Pantel, 2004) is a resource of strongly associated verb pairs and their semantic relationship. Verbs are considered strongly associated if DIRT deems dependency paths, which contain the verbs, as being similar. A form of mutual information between verb pairs and lexico-syntactic patterns indicative of semantic relationship types is used to categorize the verb pairs according to similarity, strength, antonymy, happens-before and enablement.

(Fujiki et al., 2003) describe a method to extract script knowledge from the first paragraph of Japanese newspaper articles. The first paragraph of such articles is assumed to narrate its contents in temporal order. This circumvents the need to order events as they can be extracted in presumed order. Events are defined in terms of actions, where an action consists of a tuple composed of a transitive verb and its subject and object. The method's goal is to find sequences of pairs of actions by (1) using co-occurrence of subjects and objects in neighboring sentences, (2) locating sentences where two verbs share the same subject and (3) identifying sentences where two verbs share the same object. Once pairs of events are extracted, their subject and objects are generalized into semantic entities similar to semantic roles.

(Chambers and Jurafsky, 2008) attempt to identify narrative chains in newspaper corpora. They utilize the notion of protagonist overlap or verbs sharing co-referring arguments to establish semantic coherence in a story. Co-referring arguments are taken as indicators of a common discourse structure. This assumption is used to find pairwise events in an unsupervised fashion. Point wise mutual information (PMI) is used to indicate the relatedness between event pairs. A global narrative score, aiming to maximize the PMI of a set of events is utilized to generate a ranked list of events most likely to participate in the narrative chain. Temporal order is established by labeling events with temporal attributes and using those labels, along with other linguistic features, to classify the relationship (before or other) between two events.

For the purposes of our work, finding documents related to a term and identifying similar terms is an important step in the script creation process. (Deerwester et al., 1990) describe Latent Semantic Analysis/Indexing (LSA) as a technique superior to term matching document retrieval. LSA aims to facilitate document retrieval based on the conceptual content of documents, thereby avoiding problems with synonomy and polysemy of individual search terms (or in documents). LSA employs singular-value-decomposition (SVD) of a term-by-document matrix to construct a "semantic space" in which related documents and terms are clustered together.

3 Approach

In this work we aim to extract scripts from the web. We define a script as a collection of typically related events that participate in temporal relationships amongst each other. For example, e_1 `happens-before` e_2 denotes a relationship such that event e_1 occurs prior to event e_2. An event is defined as a tuple consisting of a verb, a grammatical function and a set of arguments (i.e., words) which act out the grammatical function in relation to the verb. Figure 1 shows the structure of events.

$$e\ [\text{verb, grammatical function, } \{\text{set of arguments}\}]$$

Figure 1: The structure of an event. An event is a tuple consisting of a verb, a grammatical function and a set of arguments (i.e., instances of words filling the grammatical function).

The set of arguments represents actual instances found during the script extraction process. Figure 2 illustrates an incomplete script for the topic *eating at a restaurant*.

3.1 The Task

We define the task of goal driven script extraction as: (1) given a topic, compile a "relevant" corpus of doc-

e_1 [enter, nsubj, {customer, John})]
e_2 [enter, dobj, {restaurant}]
e_3 [order, nsubj, {customer}]
e_4 [order, dobj, {food}]
e_5 [eat, nsubj, {customer}]
e_6 [eat, dobj, {food}]
e_7 [pay, nsubj, {customer}]
e_8 [pay, dobj, {bill}]
e_9 [leave, nsubj, {customer}]
e_{10} [leave, dobj, {restaurant}]

Temporal Ordering = $e_1 < e_2 < e_3 < e_4$
$e_4 < e_5 < e_6 < e_7 < e_8 < e_9 < e_{10}$

Figure 2: An excerpt of a script for the topic *eating at a restaurant*. The script denotes the stylized actions of a customer dining at a restaurant. The $<$ relation denotes event e_i `happens_before` event e_j.

uments from a subset of documents on the web, (2) extract events relevant for the topic, (3) (optional) refine the topic and restart at 1, and (4) establish a temporal ordering for the events.

We currently impose restrictions on the form of acceptable topics. For our purposes, a topic is a short description of a script, and contains at least a verb and a noun from the script's intended domain. For example, the topic for a passenger's typical actions while using public transportation (i.e. a bus) can be described by the topic *riding on a bus*.

3.2 System Architecture

The script extraction system consists of a variety of modules where each module is responsible for a certain task. Modules are combined in a mixed fashion such that sequential processing is combined with an iterative improvement procedure. Figure 3 illustrates the system architecture and flow of information between modules. The following sections describe each module in detail.

3.2.1 Document Retrieval

Our system utilizes the web as its underlying information source to circumvent domain limitations of fixed corpora. However, using the entire web to extract a script for a specific topic is, on one hand, infeasible due to the size of the web and, on the other hand, impractical in term of document relevancy. Since only a subset of pages is potentially relevant to a given topic, the web needs to be filtered such that mostly relevant web pages are retrieved. The document retrieval module makes use of existing search engines for this purpose.[1]

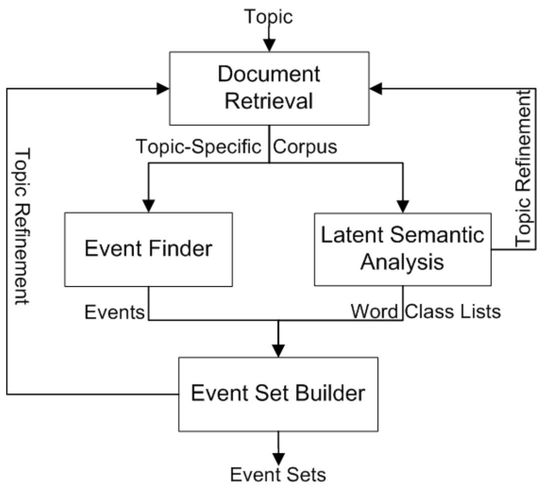


Figure 3: System Architecture and flow of information.

The document retrieval module is presented with the topic for a script and issues this topic as a query to the search engines. The search engines produce a relevancy ranked list of documents/URLs (Brin and Page, 1998) which, in turn, are downloaded. The number of downloaded pages depends on the current iteration number of the system (i.e., how often the *retrieval-analysis* cycle has been executed for a given topic[2]).

The document retrieval module is also responsible for cleaning the documents. The cleaning process aims to remove "boilerplate" elements such as navigational menus and advertising from web pages while preserving content elements.[3] The collection of cleaned documents for a given topic is considered to be a *topic-specific corpus*.

3.2.2 Latent Semantic Analysis (LSA)

The aim of the LSA module is to identify words (verbs, nouns and adjectives) that are closely related

[1]The Google and Yahoo API's are used to establish communication with these search engines.

[2]At the first iteration, we have arbitrarily choosesn to retrieve the first 1000 unduplicated documents.

[3]The Special Interest Group of the ACL on Web as Corpus (SIGWAC) is interested in web cleaning methods for corpus construction. Our web page cleaner uses a support vector machine to classify blocks of a web page as content or non-content. The cleaner achieves $\approx$ 85% F1 on a random set of web pages.

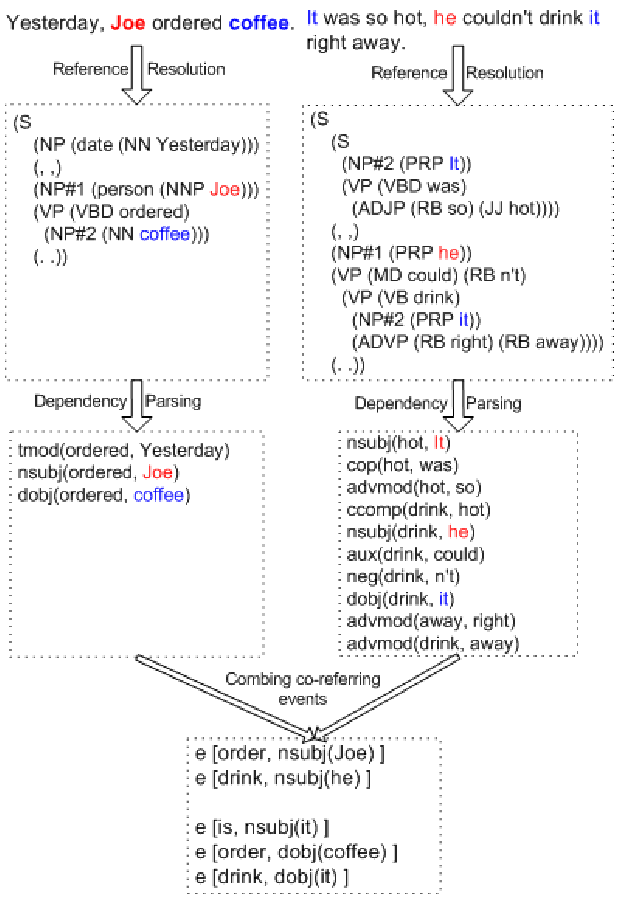
to the topic presented to the document retrieval module. To find such words, the topic-specific corpus is (1) part-of-speech tagged and (2) transformed into a term-document matrix. Each cell represents the log-entropy for its respective term in a document. Note that we consider a term to be a tuple consisting of a word and its POS. The advantage of using word-POS tag combinations over words only is the ability to query LSA's concept space for words by their word class. A concept space is created by applying SVD to the term-document matrix, reducing the dimensionality of the scaling matrix and reconstructing the term-document matrix using the reduced scaling matrix.

Once the concept space is constructed, the space is searched for all terms having a high correlation with the original topic. Terms from the original topic are located in the concept space (i.e., term vectors are located) and other term vectors with high cosine similarity are retrieved from the space. A list of $n = 50$ terms[4] for each word class $\in$ {verb, noun, adjective} is obtained and filtered using a stop list. The stop list currently contains the 100 most common words in the English language. The idea behind the stop list is to remove low content words from the list. The resulting set of words is deemed to have high information content with respect to the topic. This set is used for two purposes: (1) to augment the original topic and to restart the document collection process using the augmented topic and (2) identify event pairs constructed by the event finding module which contain these highly correlated terms (either as events or event arguments). The first purpose aims to use an iterative process to construct a higher quality topic-specific corpus. A new corpus created in this fashion presumably represents documents that are richer and more relevant to the augmented topic. The second purpose steers the extraction of events towards events containing those constituents judged most relevant. This fact can be incorporated into a maximization calculation based on pointwise mutual information to find highly correlated events.

[4]The number was chosen experimentally and is based on the correlation score (cosine similarity) between word vectors. After about 50 words, the correlation score begins to drop significantly indicating weaker relatedness.

3.2.3 Event Finding

The collection of documents (or topic-specific corpus) is then processed to facilitate finding event pairs. Finding event pairs involves the notion of verb argument overlap using the assumption that two events in a story are related if they share at least one semantic argument across grammatical functions. This virtue of discourse structure of coherent stories has been described in (Trabasso et al., 1984) and applied by (Fujiki et al., 2003) as subject and object overlap and by (Chambers and Jurafsky, 2008) as following a common protagonist in a story. For example, in the sentences "John ordered a drink. He enjoyed it very much." we can establish that events *order* and *enjoy* are part of a common theme because the arguments (or loosely semantic roles) of *order* and *enjoy* refer to the same entities, that is *John* = *He* and *a drink* = *it*.



Yesterday, Joe ordered coffee.
It was so hot, he couldn't drink it right away.
Reference Resolution
Reference Resolution
Dependency Parsing
Dependency Parsing
Combing co-referring events

```
(S
  (NP (date (NN Yesterday)))
  (, ,)
  (NP#1 (person (NNP Joe)))
  (VP (VBD ordered)
    (NP#2 (NN coffee)))
  (. .))
```

```
(S
  (S
    (NP#2 (PRP It))
    (VP (VBD was)
      (ADJP (RB so) (JJ hot))))
  (, ,)
  (NP#1 (PRP he))
  (VP (MD could) (RB n't)
    (VP (VB drink)
      (NP#2 (PRP it))
      (ADVP (RB right) (RB away))))
  (. .))
```

```
tmod(ordered, Yesterday)
nsubj(ordered, Joe)
dobj(ordered, coffee)
```

```
nsubj(hot, It)
cop(hot, was)
advmod(hot, so)
ccomp(drink, hot)
nsubj(drink, he)
aux(drink, could)
neg(drink, n't)
dobj(drink, it)
advmod(away, right)
advmod(drink, away)
```

```
e [order, nsubj(Joe) ]
e [drink, nsubj(he) ]

e [is, nsubj(it) ]
e [order, dobj(coffee) ]
e [drink, dobj(it) ]
```

Figure 4: Example processing of the sentences "Yesterday, Joe ordered coffee. It was so hot, he couldn't drink it right away". The output after dependency parsing, reference resolution and event finding is a set of event pairs.

To identify such pairs, the topic specific corpus is (1) co-reference resolved[5] and (2) dependency parsed[6]. Sentences containing elements referring to the same mention of an element are inspected for verb argument overlap. Figure 4 illustrates this procedure for the sentences "Yesterday, Joe ordered coffee. It was so hot, he couldn't drink it right away".

Co-reference resolution tells us that mention *he* refers to *Joe* and mention(s) *it* refer to *coffee*. By our previous assumption of discourse coherence, it is possible to deduce that events *was* and *drink* are associated with event *order*. In a similar fashion, event *drink* is associated with event *was*. This is due to the fact that all events share at least one argument (in the case of events *order* and *drink* two arguments are shared). For each pair of events sharing arguments in a particular grammatic function, an event pair is generated indicating where the overlap occurred.

3.2.4 Constructing Event Sets

Sets of events representing script-like structures are constructed through the use of pointwise mutual information in combination with the lists of related words found by Latent Semantic Analysis. We utilize the definition of PMI described in (Chambers and Jurafsky, 2008). For two events e_1 and e_2

$$pmi(e1, e2) = log\frac{P(e_1, e_2)}{P(e_1)P(e_2)} \tag{1}$$

where

$$P(e_1, e_2) = \frac{C(e_1, e_2)}{\sum_{i,j} C(e_i, e_j)} \tag{2}$$

and $C(e_1, e_2)$ is the number of times events e_1 and e_2 had coreferring arguments.

We extend the definition of PMI between events to assign more weight to events whose constituents are contained in the list of words (verbs, nouns and adjectives) judged by Latent Semantic Analysis to be most relevant to the topic. For notational purposes, these lists are denoted L. Thus, we can calculate the weighted LSA PMI $LP(e_1, e_2)$ as follows:

$$LP(e_1, e_2) = P(e_1, e_2) + LSA(e_1, e_2) \tag{3}$$

where

$$LSA(e_1, e_2) = \alpha * (E(e_1) + E(e_2)) \tag{4}$$

$$\alpha = \begin{cases} 2 & \text{if } e_{1_{verb}} \in L \text{ and } e_{2_{verb}} \in L \\ 1 & \text{otherwise} \end{cases} \tag{5}$$

$$\begin{aligned} E(e) &= (||e_{verb} \cap L|| + 1) \\ &* \ (\tfrac{||e_{Args} \cap L||}{||e_{Args}||} + 1) \end{aligned} \tag{6}$$

To construct the set of n events related to the topic, the LP scores are first calculated for each event pair in the corpus. The set can then be constructed by maximizing:

$$\max_{i<k\leq n} \sum_{i=0}^{k-1} LP(e_i, e_k) \tag{7}$$

Therefore, events that share a larger number of constituents with the LSA relevancy list are preferred for inclusion in the event set. This practice distributes the relatedness weight among the frequency of events and LSA. The noisy nature of our proposed corpus generation method makes such a technique essential as we will see in section 4.3.

3.2.5 Ordering Events

At this time, we only establish a naive temporal ordering on the events. The ordering process simply assumes that an event appearing in the corpus prior to another event also occurs earlier in time. We realize that this assumption does not always hold, but delay a more sophisticated ordering process as future work.

4 Experiments

This section describes experimental results, obstacles we have encountered, various approaches to overcome these obstacles and lessons learned from our work. Unless mentioned otherwise, the results pertain to the topic *eating at a restaurant*. This topic has been chosen for our investigation since previous work (Schank and Abelson, 1977) establishes a comprehensive reference as to what a script for this domain may entail.

4.1 Domain Richness

The first step in our work was to confirm the notion that the web can be used as the underlying information source for topic-specific script extraction.

[5]OpenNLP's Co-Reference Engine is utilized `http://opennlp.sourceforge.net/`.

[6]The Stanford Parser is utilized `http://nlp.stanford.edu/software/lex-parser.shtml`

The overall goal was to investigate whether a topic-specific corpus contains sufficiently useful information which is conducive to the script extraction task.

Latent Semantic Analysis was performed on the Part-of-Speech tagged topic specific corpus. The semantic space was queried using the main constituents of the original topic. Hence, this resulted in two queries, namely *eating* and *restaurant*. For each query, we identified the most related verbs, nouns and adjectives/adverbs and placed them in respective lists. Lists are then combined according to word class, lemmatized, and pruned. Auxiliary verbs such as *be*, *do* and *have* consistently rank in the top 10 most similar words in the un-pruned lists. This result is expected due to the frequency distribution of auxiliaries in the English language. It is a natural conclusion to exclude auxiliaries from further consideration since their information content is relatively low. Furthermore, we extend this notion to exclude the 100 most frequent words in English from these lists using the same justification. By the inverse reasoning, it is desirable to include words in further processing that occur infrequently in natural language. We can hypothesize that such words are significant to a given script because their frequency appears to be elevated in the corpus. Table 1 (left) shows the resulting word class lists for both queries. Duplicates (i.e., words with identical lemma) have been removed.

The table reveals that some words also appear in the restaurant script as suggested by (Schank and Abelson, 1977). In particular, **bold** verbs resemble Schank's *scenes* and **bold** nouns resemble his *props*. We can also see that the list of adverbs/adjectives appear to not contribute any significant information. Note that any bold words have been hand selected using a human selector's subjective experience about the *eating at a restaurant* domain. Furthermore, while some script information appears to be encoded in these lists, there is a significant amount of noise, i.e., normal font words that are seemingly unimportant to the script at hand.

For our purposes, we aim to model this noise so that it can be reduced or removed to some degree. Such a model is based on the notion of overlap of noisy terms in the LSA lists derived from independent topic related corpora for the main constituents of the original topic. For example, for the topic *eating at a restaurant*, we obtain two additional corpora using the method described in Section 3.2.1, i.e., one corpus for constituent *eating* and another for the second main constituent of the original topic, *restaurant*. Both corpora are subjected to LSA analysis from which two (one for each corpus) LSA word lists are obtained. Each list was created using the respective corpus query as the LSA query. The assumption is made that words which are shared (pairwise) between all three lists (i.e., the two new LSA lists and the LSA list for topic *eating at a restaurant*) are noisy due to the fact that they occur independent of the original topic.

eating at a restaurant			**Overlap removed**		
Verbs	Nouns	A&A	Verbs	Nouns	A&A
keep	home	own	**order**	home	**fry**
need	place	still	set	hand	amaze
help	**table**	last	expect	**bowl**	green
dine	lot	open	**share**	**plate**	**grill**
love	part	full	**drink**	**cook**	chain
order	hand	off	try	**fish**	**diet**
feel	reason	long	cut	**soup**	clean
avoid	course	fat	**decide**	**service**	smart
add	side	right	watch	break	total
let	number	down	process	**drink**	relate
stay	experience	busy	save	**cheese**	worst
include	**water**	fast	**offer**	**rice**	black
tend	point	single	provide	**serve**	fit
set	**dish**	low	hear	chance	light
tell	**bowl**	free	**fill**	**portion**	exist
found	**plate**	white	forget	body	empty
bring	**bite**	wrong	write	party	live
locate	**cook**	ready	follow	rest	
eat	**fish**	true	travel	cream	
leave	**soup**	close		**taste**	

Table 1: (Left) 20 most relevant terms (after pruning) for LSA queries *eating* and *restaurant* on the *eating at a restaurant* corpus. (Right) Terms remaining after noise modeling and overlap removal. **Bold** terms in the table were manually judged by a human to be relevant.

Table 1 (right) illustrates the LSA list for topic *eating at a restaurant* after removing overlapping terms with the two other LSA lists. Bold words were judged by a human selector to be relevant to the intended script. From the table we can observe that:

1. A significant amount of words have been removed. The original table contains 50 words for each word class. The overlap reduced table contains only 19 verbs, 29 nouns, and 17 adjectives, a reduction of what we consider noise by

$\approx 57\%$

2. More words (bold) were judged to be related to the script (e.g., 6 vs. 5 relevant verbs, 12 vs. 9 nouns, and 3 vs. 0 adjectives/adverbs)
3. More relevant words appear in the top part of the list (words in the list are ordered by relevancy)
4. Some words judged to be relevant were removed (e.g., dine, bring, eat).

Using the information from the table (left and right) and personal knowledge about *eating at a restaurant*, a **human** could re-arrange the verbs and nouns into a partial script-like format of the form[7]:

e_1 [offer, nsubj, {**waiter**})]
e_2 [offer, dobj, {drink})]
Example: waiter offers a drink

e_2 [order, nsubj, {**customer**}]
e_3 [order, dobj, {fish, soup, rice}]
Example: customer orders fish

e_4 [serve/bring, nsubj, {**waiter**}]
e_5 [serve/bring, nsubj, {bowl, plate}]
Example: waiter serves/bings the bowl, plate

e_6 [eat, nsubj, {**customer**}]
e_7 [eat, dobj, {portion, cheese}]
Example: customer eats the portion, cheese

e_8 [leave, nsubj, {customer}]
e_9 [leave, dobj, {table}]
Example: customer leaves the table

Note that this script-like information was not obtained by direct derivation from the information in the table. It is merely an illustration that some script information is revealed by LSA. Table 1 neither implies any ordering nor suggest semantic arguments for a verb. However, the analysis confirms that the web contains information that can be used in the script extraction process.

4.2 Processing Errors

As mentioned in section 3.2.3, events with co-referring arguments are extracted in a pairwise fashion. In the following section we describe observations about the characteristics of events extracted this way. However, we note that each step in our system architecture is imperfect, meaning that errors are introduced in each module as the result of processing. We have already seen such errors in the form of words with incorrect word class in the LSA lists as the result of incorrect POS tagging. Such errors are amplified through imprecise parsing (syntactic and dependency parsing). Other errors, such as omissions, false positives and incorrect class detection, are introduced by the named entity recognizer and the co-reference module. With this in mind, it comes as no surprise that some extracted events, as seen later, are malformed. For example, human analysis reveals that the verb slot of these events are sometimes "littered" with words from other word classes, or that the arguments of a verb were incorrectly detected. A majority of these errors can be attributed to ungrammatical sentences and phrases in the topic-specific corpus, the remainder is due to the current state of the art of the parsers and reference resolution engine.

4.3 Observations about events

To compare relations between events, we looked at three different metrics. The first metric $M1$ simply observes the frequency counts of pairwise events in the corpus. The second metric $M2$ utilizes point wise mutual information as defined in (Chambers and Jurafsky, 2008). The third metric $M3$ is our LSA based PMI calculation as defined in section 3.2.4.

$M1$ reveals that uninformative event pairs tend to have a high number of occurrences. These pairs are composed of low content, frequently occurring events. For example, event pair [e [have, nsubj,{}], e [say, nsubj, {}]] occurred 123 times in our topic-specific corpus. More sophisticated metrics, such as $M2$, consider the frequency distributions of individual events and allocate more weight to co-occurring events with lower frequency counts of their individual events.

In this fashion, $M2$ is capable of identifying strongly related events. For example, Table 2 lists the five pairwise events with highest PMI for our topic-specific corpus.

From Table 2, it is apparent that these pairs participate in mostly meaningful (in terms of human comprehensibility) relationships. For example, it does

[7]Bold terms do not appear in the LSA lists, but were added for readability.

Event Pairs	
e[sack, dobj, {the, employees}]	e[reassign, dobj, {them}]
e[identify, nsubj, {we, Willett}]	e[assert, nsubj, {Willett}]
e[pour, dobj, {a sweet sauce}]	e[slide, dobj, {the eggs}]
e[walk, nsubj, {you, his sister}]	e[fell, nsubj, {Daniel}]
e[use, nsubj, {the menu}]	e[access, dobj, {you}]

Table 2: Pairwise events with highest PMI according to Equation 1.

not require a leap of faith to connect that *sack*ing employees is related to *reassign*ing them in the context of a corporate environment.

e(eat, nsubj), e(gobble, nsubj), e(give, nsubj),
e(live, nsubj), e(know, nsubj), e(go, nsubj),
e(need, nsubj), e(buy, nsubj), e(have, nsubj),
e(make, nsubj), e(say, nsubj), e(work, nsubj),
e(try, nsubj), e(like, nsubj), e(tell, dobj),
e(begin, nsubj), e(think, nsubj), e(tailor, nsubj),
e(take, nsubj), **e(open, nsubj)**,e(be, nsubj)

Figure 5: An event set obtained through metric $M2$ (using the PMI between events). Temporal ordering is not implied. Event arguments are omitted. Bold events indicate subjectively judged strong relatedness to the *eating at a restaurant* topic.

Figure 5 shows a set of events for our topic. The set was created by greedily adding event e_n such that for all events $e_1, e_2, ...e_{n-1}$ already in the set $\forall_i^{n-1} pmi(e_i, e_n)$ is largest (see (Chambers and Jurafsky, 2008)). The topic constituent *eating* (i.e., *eat*) was used as the initial event in the set. If this set is intended to approximate a script for the *eating at a restaurant* domain, then it is easy to see that virtually no information from Schank's *restaurant* reference script is represented. Furthermore, by human standards, the presented set appears to be incoherent. From this observation, we can conclude that $M2$ is unsuitable for topic-specific script extraction.

Figure 6 illustrates an event set constructed using metric $M3$. Note that the sets in Figures 5 and 6 do not imply any ordering on the events. The bold events indicate events that appear in our reference script or were judged by a human evaluator to be logically coherent with the *eating at a restaurant* domain. The evaluation was conducted using the evaluators personal experience of the domain. In the future, we intend to formalize this evaluation process.

Figure 6 signifies an improvement of the results of

e(eat, nsubj), **e(wait, dobj)**, e(total, nsubj)
e(write, dobj), e(place, dobj), e(complete, dobj)
e(exist, nsubj), e(include, dobj), e(top, nsubj)
e(found, dobj), e(keep, dobj), e(open, dobj)
e(offer, dobj), e(average, nsubj), **e(fill, dobj)**
e(taste, nsubj), **e(drink, dobj)**, **e(cook, dobj)**
e(read, dobj), **e(enjoy, dobj)**,e(buy, dobj)

Figure 6: An event set obtained through metric $M3$ (weighing PMI and LSA between events). Temporal ordering is not implied. Event arguments are omitted. Bold events indicate subjectively judged strong relatedness to the *eating at a restaurant* topic.

Figure 5 in terms of the number of events judged to belong to the restaurant script. This leads us to the conclusion that a metric based on scaled PMI and LSA appears more suitable for the web based topic driven script extraction task. Once a temporal order is imposed on the events in Figure 6, these events could, by themselves, serve as a partial event set for their domain.

5 Discussion and Future Work

We have presented preliminary work on extracting script-like information in a topic driven fashion from the web. Our work shows promise to identify script knowledge in a topic-specific corpus derived from an unordered collection of web pages. We have shown that while web documents contain significant amounts of noise (both boilerplate elements and topic unrelated content), a subset of content can be identified as script-like knowledge.

Latent Semantic Analysis allows for the filtering and pruning of lists of related words by word classes. LSA furthermore facilitates noise removal through overlap detection of word class list elements between independent corpora of topic constituents. Our method of weighted LSA and PMI for event relatedness produces more promising partial event sets than existing metrics.

For future work, we leave the automated evaluation of partial sets and the establishing of temporal relations between events in a set. Our system architecture features an iterative model to event set improvement. We hope that this approach will allow us to improve upon the quality of event sets by using extracted sets from one iteration to bootstrap a new iteration of event extraction.

References

Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.*, 30(1-7):107–117.

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

Timothy Chklovski and Patrick Pantel. 2004. VerbOcean: Mining the Web for Fine-Grained Semantic Verb Relations. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, pages 33–40, Barcelona, Spain, July. Association for Computational Linguistics.

Scott Deerwester, T. Susan Dumais, W. George Furnas, K. Thomas Landauer, and Richard Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41:391–407.

Toshiaki Fujiki, Hidetsugu Nanba, and Manabu Okumura. 2003. Automatic acquisition of script knowledge from a text collection. In *EACL '03: Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics*, pages 91–94, Morristown, NJ, USA. Association for Computational Linguistics.

Dekang Lin and Patrick Pantel. 2001a. DIRT - Discovery of Inference Rules from Text. In *KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 323–328, New York, NY, USA. ACM.

Dekang Lin and Patrick Pantel. 2001b. Discovery of Inference Rules for Question-Answering. *Nat. Lang. Eng.*, 7(4):343–360.

Roger C. Schank and Robert P. Abelson. 1977. *Scripts, Plans, Goals and Understanding*. Lawrence Erlbaum Associates, Hillsdale, NJ.

Tom Trabasso, T. Secco, and Paul van den Broek. 1984. Causal Cohesion and Story Coherence. In *Heinz Mandl, N.L. Stein and Tom Trabasso (eds). Learning and Comprehension of Text.*, pages 83–111, Hillsdale, NJ, USA. Lawrence Erlbaum Associates.