Narrative Text Understanding

John was hungry. He went into Goldstein's and ordered a pastrami sandwich. It was served to him quickly. He paid the check and left the waitress a big tip.

- What is Goldstein's?
- What did John eat?
- Who made the sandwich?
- Who took John's order?
- Who served the sandwich?
- Why did John leave a big tip?

Scripts

- A **script** is a stereotypical sequence of events associated with a common activity.
- A script is a *conceptual knowledge structure* that was proposed as a means of memory organization.
- People usually omit many events in a stereotypical event sequence when telling a story.
- Scripts are essential to story understanding as a mechanism for filling in details that were not explicitly mentioned but would be commonly (almost universally) inferred.

Example: Restaurant Script

$RESTAURANT
1. enter restaurant
2. be seated
3. read menu
4. order food
5. served food
6. eat food
7. pay for meal
8. leave restaurant

Each individual event is called a *scene*. A scene may have its own script.

Script-based Inferencing

John was hungry. He went into Goldstein's and ordered a pastrami sandwich. It was served to him quickly. He paid the check and left the waitress a big tip.

1. go to a restaurant	→ *[John]* ***went into Goldstein's***
2. be seated	
3. read menu	
4. order food	→ *[John]* ***ordered a pastrami sandwich***
5. served food	→ *[The sandwich]* ***was served*** *[to John]*
6. eat food	
7. pay for meal	→ *[John]* ***paid the check*** *[for sandwich]*
8. leave the restaurant	

Script-based Inferencing

John was hungry. He went into Goldstein's and ordered a pastrami sandwich. It was served to him quickly. He paid the check and left the waitress a big tip.

1. go to a restaurant → *[John]* ***went into Goldstein's***
2. be seated → John sat down
3. read menu → John read a menu
4. order food → *[John]* ***ordered a pastrami sandwich***
5. served food → *[The sandwich]* ***was served*** *[to John]*
6. eat food → John ate the sandwich
7. pay for meal → *[John]* ***paid the check*** *[for sandwich]*
8. leave the restaurant → John left Goldstein's

Script Tracks

A script may have several *tracks* representing different variations of the situation.

For example, the restaurant script may have tracks for:

– Cafeteria: seat yourself, no waiter or waitress, no tipping

– Fast food: seat yourself, no waiter or waitress, no tipping, pay after ordering but before eating, may eat inside or outside the restaurant

– Fine dining: several waiters and waitresses, wine list, additional courses

Example: Subway Script

$SUBWAY
1. enter subway station
2. go to turnstile
3. put token in turnstile
4. go through turnstile
5. go to train platform
6. wait for train
7. enter train
8. find a seat
9. exit train at destination

Script Roles

Each script has a cast of characters specified by **script roles**.

For example, $SUBWAY has the script roles:

&PATGRP = (patron) group of subway riders

&CASHIER = cashier

&CONDUCTOR = train conductor

&DRIVER = person driving the train

&SUBORG = subway organization (e.g., UTA)

Props

Each script has a set of associated objects called **props**.

For example, $SUBWAY has the props:

&TOKEN subway token
&FARE money paid for a token
&TURNSTILE turnstile
&PLATSEAT seat on a platform
&TRAIN the train
&TRAIN CAR a car of the train
&CAR SEAT a seat on a car
&STRAP strap for rider to grasp
&ENTERGATE the gate from the origin to the platform
&EXITGATE the gate from the platform leading to the destination

Settings

Each script has a list of places where the events take place, called settings.

For example, $SUBWAY has three main settings:

– the origin station

– inside the train

– the destination station

Together, the roles, props, and settings make up the **script variables**. These variables are instantiated with information from the story, or default values, when the script is applied.

Applications for Scripts

- Coreference Resolution
- Semantic Interpretation
- Inferencing
- Summarization
- Text Generation
- Question Answering
- Deep Natural Language Understanding!

Question Answering

John went to a restaurant. He ordered a hot dog. The waiter said they didn't have any. He asked for a hamburger. When the hamburger came it was burnt. He left the restaurant.

Q1: Did John sit down at the restaurant?
A1: Probably.

Q2: Did John order a hot dog?
A2: Yes.

Q3: Did John eat a hot dog?
A3: No, the waiter told John the management was unable to give it to him.

Q4: What did the waiter serve John?
A4: The waiter served John a hamburger.

Q5: Why didn't John eat the hamburger?
A5: Because the hamburger was overdone.

Q6: Did John pay the check?
A6: No, John was angry because the hamburger was overdone so he left the restaurant.

Narrative Event Chains

A **narrative event chain** is a partially ordered set of events related by a common protagonist [Chambers & Jurafsky, ACL 2008]. For example:

Firing of Employee	*Executive Resigns*
accused X	X joined
X claimed	X served
X argued	X oversaw
dismissed X	X resigned

Narrative event chains represent structured background knowledge that can support inferencing of subevents.

The Protagonist

- A key assumption in this work is that most narrative texts have a central actor, which they call the **protagonist**.

  protagonist = the leading character, hero, or heroine in a drama of other literary work. (From dictionary.com)

- A narrative chain is structured by the grammatical roles of the protagonist in events.

  ***Assumption of Narrative Coherence**: verbs that share coreferring arguments are semantically connected by virtue of narrative discourse structure.*

- A narrative chain therefore represents one perspective (the protagonist's) on the events in the story.

Event Representation

- A **narrative event** is represented as a tuple consisting of a verb (and optional particle) and a typed dependency relation linking to the protagonist.

  For example: (*rescued*, Subject) or (*saved*, PP_by)

- A **narrative chain** is a set of narrative events (e_1, e_2, … , e_N) and a *before relation* B(e_i, e_j) that is true if e_i occurs strictly before e_j in time.

- A narrative chain is a partial ordering, so some *before relations* may be unknown, or the ordering between some events may not matter.

Learning Narrative Relations

The first step is to identify events that are related by virtue of narrative discourse structure.

For each document:

1. Apply a dependency parser.
2. Apply a coreference resolver to produce entity clusters.
3. Compute the pointwise mutual information (PMI) between pairs of narrative events to measure their relatedness.

Identifying Related Events

Pointwise mutual information (PMI) is used to measure the degree of relatedness between two narrative events.

$$\text{PMI}\,(e(w,d), e(v,g)) = \log \frac{P(e(w,d), e(v,g))}{P(e(w,d)) * P(e(v,g))}$$

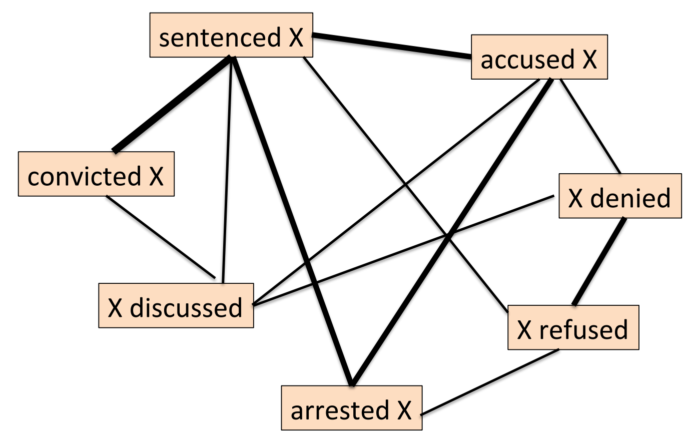
e(w,d) and e(v,g) are verbs (w & v) paired with a dependency (d & g)

The numerator is defined as:

$$P(e(w,d), e(v,g)) = \frac{\text{Count}(e(w,d), e(v,g))}{\sum_{w,v} \sum_{d,g} \text{Count}(e(w,d), e(v,g))}$$

The Count function is the number of times that the two events had a coreferring expression as the dependency values.

Event Relatedness Example



Global Narrative Score

- Given a set of narrative events in a document, the next most likely event to occur is identified by maximizing:

$$\max_{0<j<m} \sum_{i=0}^{n} \text{PMI}(e_i, f_j)$$

where n is the number of events in the chain (e_i is the i^{th} event), and m is the number of events in the training corpus.

Intuition: given a set of N events that have occurred, what other events are likely to occur?

Example of Predicted Events

Known Events:

(pleaded, subj) (admits, subj) (convicted, obj)

Likely Additional Events:

(sentenced, obj) .89
(paroled, obj) .76
(fired, obj) .75
(indicted, obj) .74
(fined, obj).73
(denied, subj) .73

Narrative Cloze Evaluation Metric

- The **Cloze Task** [Taylor, 1953] is used to evaluate language proficiency by removing a random word from a sentence and asking a person/system to replace it. For example:

  I forgot to _____ the waitress for the good service.

- **Narrative cloze task**: given a sequence of narrative events from which one event has been removed, fill in the missing event.

  NOTE: They do not claim this is solveable, but use it as a comparative measure.

> *McCann threw two interceptions early. Toledo pulled McCann aside and told him he'd start. McCann quickly completed his first two passes.*

(*threw* Subject), (*pulled*, Object), (*told*, Object), (*start*, Subject), (*completed*, Subject)

Remove one event and use the others to assign it a score!

Narrative Cloze Experiment

- **Training Data**: ~1 million documents from the Gigaword corpus
  - parsed to identify verbs (with optional particle) linked to Subject, Object, or PP dependencies.
  - entity clusters generated by a coreference resolver.
- **Development Data**: 10 documents
- **Testing Data**: 69 documents, which contained 740 events

For each document, the entity involved in the most events was selected as the protagonist.

Evaluation used leave-one-out cross validation, so 740 tests. Results were averaged over these 740 tests.

Sample Test Document

New York Times Editorial

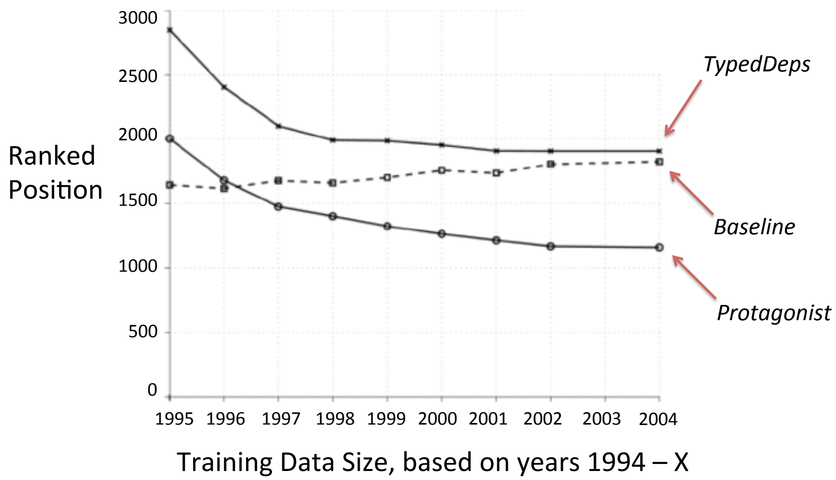
Protagonist = *President Bush*

(*occupied*, Subject)	(*brought*, Subject)	(*rejecting*, Subject)
(*projects*, Subject)	(*met*, Subject)	(*appeared*, Subject)
(*offered*, Subject)	(*voted*, PP_for)	(*offer*, Subject)
(*thinks*, Subject)		

Baseline for Narrative Cloze Test

- A **baseline system** was designed to learn relatedness simply using verb co-occurrence in the same document with PMI.
- Since all pairs are expensive to compute, pairs occurring < 10 times in the training data were discarded.
- Another **Protagonist model** ignores typed dependencies but computes PMI only across verbs that share arguments.
- The full narrative model that includes grammatical dependencies is called the **Typed Deps model**.

Experimental Results



Temporal Classifier

- Created a temporal classifier using the Timebank corpus as supervised data.
- **Stage 1:** train an SVM to assign temporal attributes to each event, including tense, grammatical aspect, and aspectual class. Features include neighboring POS tags, auxiliaries and modals, and WordNet synsets.
- **Stage 2:** train an SVM to classify the temporal relation between two events as *before/immediately-before* or *other*. Features included the assigned temporal attributes and event-event properties such as syntactic dominance, bigrams of tense, aspect, and class, and whether they occur in the same sentence.
- Performance was 72% for the *before* vs. *other* classification task.

Temporal Ordering of Narrative Events

- Each pair of events that shares a coreferring argument is assigned a temporal order.
- For each pair, all instances are classified by the *before* vs. *other* temporal classifier.
- A table is constructed for each pair (E_1, E_2) with the counts *before*(E_1, E_2) and *before*(E_2, E_1).
- A before link is assigned to (E_1, E_2) if *before*(E_1, E_2) > *before*(E_2, E_1). The **confidence score** of a before link is the logarithm of:

$$D(E_1, E_2) = |\ \textit{before}(E_1, E_2) - \textit{before}(E_1, E_2)\ |$$

Temporal Evaluation

The **coherence score** for a narrative chain is the sum of the confidence scores for each decision, with a positive weight for correct decisions and negative weight for incorrrect decisions.

$$\sum_{E:x,y} \begin{cases} log(D(x,y)) & \text{if } x\beta y \text{ and } B(x,y) > B(y,x) \\ -log(D(x,y)) & \text{if } x\beta y \text{ and } B(y,x) > B(x,y) \\ -log(D(x,y)) & \text{if } !x\beta y \ \&\ !y\beta x \ \&\ D(x,y) > 0 \\ 0 & \text{otherwise} \end{cases}$$

where E is the set of all event pairs, $B(i,j)$ is how many times we classified events i and j as *before* in Gigaword, and $D(i,j) = |B(i,j) - B(j,i)|$. The relation $i\beta j$ indicates that i is temporally before j.

Temporal Evaluation Results

- For evaluation, the narrative events found in 63 test documents were manually labeled with a correct partial ordering.
- For each chain, up to 300 random orderings of the same events were also generated.
- The coherence score of each random ordering was compared to the score for the true ordering, to see which one was higher.

	All	$\geq$ **6**	$\geq$ **10**
correct	8086 **75%**	7603 **78%**	6307 **89%**
incorrect	1738	1493	619
tie	931	627	160

Figure 5: Results for choosing the correct ordered chain. ($\geq$ 10) means there were at least 10 pairs of ordered events in the chain.

Example of Different Orderings

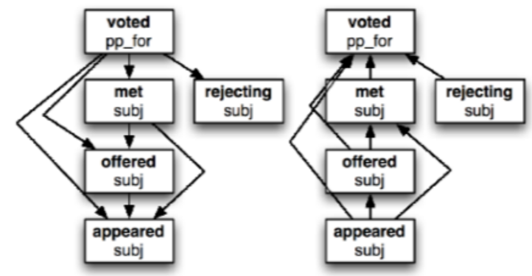


Figure 4: A narrative chain and its reverse order.

Discrete Narrative Event Chains

- Finally, the event generation component and the temporal ordering classifier were combined to see if script-like structures could be produced.
- The PMI scores were used in an agglomerative clustering algorithm, and then the ordering relations were applied to produce a directed graph.
- For example, choose an initial event, such as (arrested, Object). Then use PMI to find the next most likely event. Then another. Etc.

(*arrested*, Object)
(*arrested*, Object) (*accused*, Object)
(*arrested*, Object) (*accused*, Object) (*testified*, Subject)

Finally, apply the temporal classifier to assign *before* links.

Prosecution Chain Example

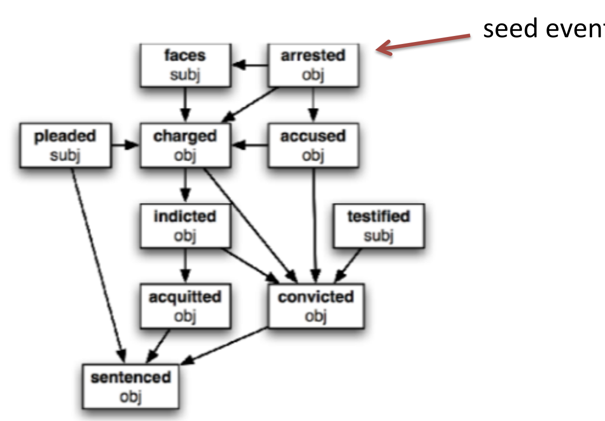


Figure 6: An automatically learned Prosecution Chain. Arrows indicate the *before* relation.

Employment Chain Example

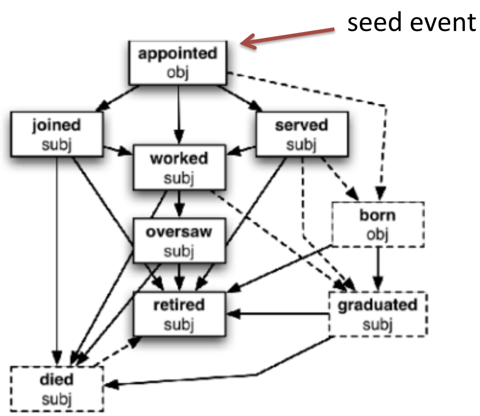


Figure 7: An Employment Chain. Dotted lines indicate incorrect *before* relations.

Conclusions

- This research on narrative event chains was one of the first attempts to automatically learn common sequences of related events.
- This is an exciting research direction! But this work was a very simple first step.
  - event representation very limited
  - one protagonist per story is not realistic
  - unclear how good the temporal orderings are
- Fundamentally, scripts should be **stereotypical sequences of events**.
  - By definition, they are so stereotypical that they can be omitted! How do we learn events that are implicit?
  - How to distinguish important from unimportant events?