Unsupervised Learning of Narrative Chains and Schemata

(Chambers and Jurafsky, 2008, 2009)

Dominikus Wetzel
dwetzel@coli.uni-sb.de

Seminar: Recent Developments in Computational Semantics (SS 2010)
Department of Computational Linguistics
Saarland University

June 28, 2010

1. Introduction
2. (Untyped) Narrative Chains
3. Typed Narrative Chains and Narrative Schemata
4. Conclusion

Introduction

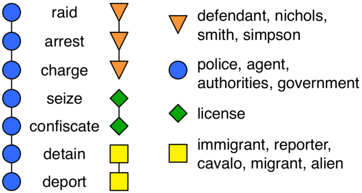


Figure: Narrative schema

- event slot, narrative chain, protagonist, types, narrative schema

The narrative chain model

Narrative chain: (L, O)

- L is a set of *event slots*
- event slot: a tuple $\langle v, d\rangle$ – also represented by e
- v is an event (represented by the verb)
- d is a typed dependency, s.t. $d \in \{subject, object, preposition\}$
  $\longrightarrow$ the protagonist: a central actor
- O is a partial order over L in time:
  $\longrightarrow O(e_i, e_j)$ is true if e_i is strictly before e_j

Example of a narrative chain

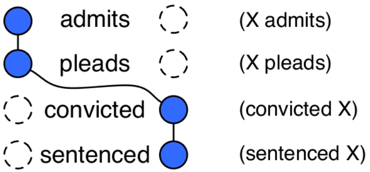


Figure: Narrative chain with protagonist X

Narrative chain (L, O):

- $L = \{\langle admits, subj\rangle, \langle pleads, subj\rangle, \langle convicted, obj\rangle, \langle sentenced, obj\rangle\}$
- $O = \{(pleads, convicted), (convicted, sentenced), \ldots\})$

Narrative coherence

Assumption of narrative coherence:

Verbs sharing coreferring arguments are semantically connected by virtue of narrative discourse structure.

- verbs with shared arguments are more likely to be part of the same narrative chain

Overview of the procedure

- parse text (dependency parser, e.g. Stanford Parser)
- record verbs with subj, obj or prepositional dependencies
- resolve coreference (OpenNLP)
- record verb pairs with coreferring arguments
- the protagonist is: the entity involved in the most events
- extract chains by agglomerative clustering
- determine partial order

A similarity measure

A similarity measure between two event slots:

$$pmi(\langle w,d\rangle,\langle v,g\rangle) = log\frac{P(\langle w,d\rangle,\langle v,g\rangle)}{P(\langle w,d\rangle)P(\langle v,g\rangle)}$$

Numerator is calculated by:

$$P(\langle w,d\rangle,\langle v,g\rangle) = \frac{\#(\langle w,d\rangle,\langle v,g\rangle)}{\sum_{x,y}\sum_{h,f}\#(\langle x,h\rangle,\langle y,f\rangle)}$$

The #-function is defined as:

$$\#(\langle a,k\rangle,\langle b,m\rangle)$$

is the count where event a and b have coreferring arguments of dependency type k and m, respectively

The most likely next event

The chainsim function:

$$chainsim(C, \langle v, g \rangle) = \sum_{i=1}^{n} pmi(\langle w, d \rangle_i, \langle v, g \rangle)$$

where

- i is an index for existing chain members
- n is the size of the existing chain, i.e. $|C|$

The most likely next event:

$$\max_{j:0<j<m} chainsim(C, \langle x, h \rangle_j)$$

where

- j is an index for the event slots in the training corpus
- m is the number of event slots in the training corpus

Evaluation - The Narrative Cloze

- a sequence of event slots in a text from which one event slot has been removed
- task: predict the missing event slot

Example:

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

- $\langle threw, subj \rangle$, $\langle pulled, obj \rangle$, $\langle told, obj \rangle$, $\langle start, subj \rangle$, $\langle completed, subj \rangle$

Baselines

Two baselines:

$$pmi(w,v) = \frac{P(w,v)}{P(w)P(v)}$$

1. Verb-only baseline:
   - the numerator is defined w.r.t to the count when both verbs occur together in a document
2. Protagonist:
   - the numerator is defined w.r.t to the count when both verbs have shared arguments in a document (irrespective of dependency type)

Training, development and test data

All documents are from the Gigaword Corpus:

- training: maximally ca. 1 mio documents (years 1994-2004)
- development: 10 manually selected documents (year 1994)
- testing: 69 random documents (year 2001) - they contain at least 5 events

Results

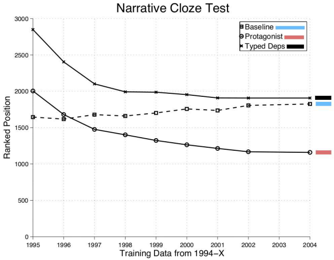


Figure: Narrative cloze test results

- y-axis: average ranked position
- the higher the rank, the better the performance
- for Protagonist and Typed Deps: the more training data, the better
- in this figure: Typed Deps not really comparable to baselines
  $\longrightarrow$ set of possible event slots is larger than set of possible events

The temporal ordering[1]

- only *before* and *other* relations
- two-stage supervised classification approach
- first stage: classifier using temporal features of one event
- second stage: classifier using event-event features including labels from first stage

[1] Based on previous work (Chambers et al., 2007)

The first stage

Classifier using temporal features of one event:

- labels: tense, grammatical aspect, aspectual class
- features: neighboring POS tags, neighboring auxiliaries and modals, WordNet synsets
- training: SVM trained on Timebank Corpus

The second stage

Classifier using event-event features
$\longrightarrow$ only event pairs with coreferring arguments:

- labels: before or other
- features: syntactic properties (e.g. dominance relation), combined bigram features of first stage (“present past”), same or different sentence
- training: ca. 37,000 relations from Timebank Corpus

Temporal Evaluation

- testing: the same 69 Gigaword documents as before
  $\longrightarrow$ with hand identified and labeled narrative chains (only "before")
- coherence score: sum of matching relations to gold standard, each weighted by a confidence score

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

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

- highest accuracy for the 24 documents containing $\geq$ 10 event pairs

Problems with untyped narrative chains

Two shortcomings:

- only one entity is treated in a narrative chain (i.e. the protagonist)
- the type of the entity is not considered

Motivation for typed narrative chains

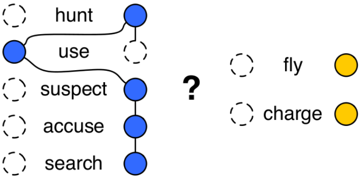


Figure: fly vs. charge

- fly is one of the top scoring events
  $\longrightarrow$ it is observed during training with all five events slots
- but charge would fit much better
  $\longrightarrow$ it shares more types with the other event slots
- Example:
  types: criminal, suspect; other slots: accuse, search, suspect

The typed protagonist

Typed narrative chain: (L, P, O)

- L and O as before
- P set of possible "argument types":
  e.g. lexical units (head words), noun clusters, other semantic representations

Example:

- $L = \{\langle arrest, subj \rangle, \langle charge, subj \rangle, \langle raid, subj \rangle, \langle confiscate, subj \rangle, \langle detain, subj \rangle, \langle deport, subj \rangle\}$
- $P = \{police, agent, authority, government\}$
- $O = \ldots$

Learning the argument type

- build the referential set of coreferring arguments
- identify the most salient one in the referential set (pronouns are ignored, named entities are mapped to “PERSON”)
- identify any event pair which has shared argument types from the referential set
  $\longrightarrow$ update count of (e, f, a), where a is the most salient argument

Example:

But for a growing proportion of [U.S. workers], the troubles really **set in** when [they] **apply** for unemployment benefits. Many [workers] **find** [their] benefits challenged.

- $\langle set\_in, prep\rangle, \langle apply, subj\rangle, \langle find, subj\rangle$
- $\{workers = 2, they = N/A, their = N/A\}$

Untyped and typed similarity measures

In (Chambers and Jurafsky, 2008):

$$sim_{untyped}(\langle e,d\rangle,\langle f,g\rangle) = pmi(\langle e,d\rangle,\langle f,g\rangle)$$

In (Chambers and Jurafsky, 2009):

$$\begin{aligned} sim_{typed}(\langle e,d\rangle,\langle f,g\rangle,a) &= pmi(\langle e,d\rangle,\langle f,g\rangle) \\ &\quad +\lambda freq(\langle e,d\rangle,\langle f,g\rangle,a) \end{aligned}$$

where

- λ is a constant weight
- $freq(e,f,a)$ is the count of the coreferring event slots e and f having an argument type from the same referential set a
- the more often e and f share argument types, the higher the similarity

A weighting function

a score function is defined, for a particular chain C and argument a:

$$score(C,a) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} sim_{typed}(\langle e,d \rangle_i, \langle f,g \rangle_j, a)$$

where

- all permutations of e_i and e_j where e_i is strictly before e_j
- this is a weight of how much a referential set a, i.e. the type, contributes to the chain

chainsim measures

The new chainsim for a given chain C and a new event slot:

$$chainsim_{typed}(C, \langle f, g \rangle) =$$

$$\max_{a} \left[score(C, a) + \sum_{i=1}^{n} sim_{typed}(\langle e, d \rangle_i, \langle f, g \rangle, a) \right]$$

Compare to (Chambers and Jurafsky, 2008):

$$chainsim_{untyped}(C, \langle f, g \rangle) = \sum_{i=1}^{n} sim_{untyped}(\langle e, d \rangle_i, \langle f, g \rangle)$$

Narrative Schema

Narrative schema: (E, C)

- E: set of events: $\langle v, D_v \rangle$ where v is a verb and $D_v \subseteq \{subject, object, prep\}$
- C: set of typed narrative chains
- Each $\langle v, d \rangle$ (where $d \in D_v$) belongs to a chain $c \in C$
- models all actors in a set of events
- Example: both $\langle push, obj \rangle$ and $\langle push, subj \rangle$ are in two (distinct) chains of the same schema

Example/Motivation

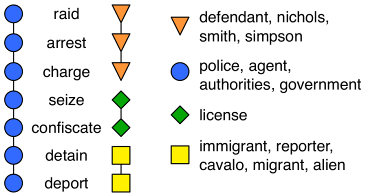


Figure: Narrative schema

- *pull_over* and *search* could be in the schema
- however, the first only shares *subj* dependency with the containing chains, e.g. circle types
- the second additionally shares *obj* dependency, e.g. triangle types
  $\Rightarrow$ favor this because it shares more arguments with containing chains

Learning Narrative Schemata

- a verb is added to a narrative schema, if all its arguments (D_v) are assigned to a chain $c \in C$ with high confidence:

$$narsim(N, v) = \sum_{d \in D_v} max \left[\beta, \max_{c \in C_N} (chainsim_{typed}(c, \langle v, d \rangle)) \right]$$

where

- β score to decide upon creation of new chain

Building Narrative Schemata

Schemata are built incrementally by adding the event from the training data that maximizes the *narsim* function:

$$\max_{j:0<j<|v|} narsim(N, v_j)$$

where

- $|v|$ is the number of observed events in the training data
- v_j is the j-th verb

Compare to building narrative chains in (Chambers and Jurafsky, 2008):

$$\max_{j:0<j<m} chainsim(c, \langle v, g \rangle_j)$$

where

- m is the number of observed *event slots* in the training data
- $\langle v, g \rangle_j$ is the j-th event slot

Evaluation - Comparison to FrameNet

- top 20 scoring narrative schemata were compared to FrameNet frames
- after automatic mapping (at least 2 matching verbs), only 13 schemata remained

Evaluation - Narrative cloze

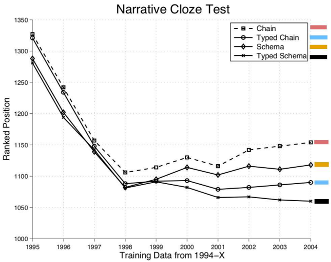
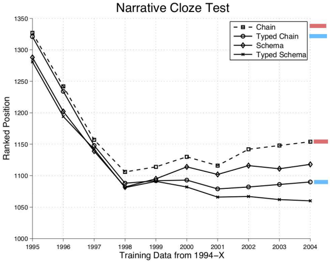


Figure: Narrative cloze test results

- same training, development and testing data as in (Chambers and Jurafsky, 2008)

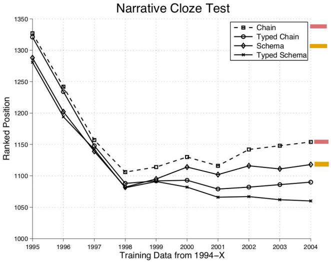
Evaluation - Narrative cloze



Untyped vs. Typed chains:

- used $chainsim_{untyped}$ and $chainsim_{typed}$ scores, respectively
- 6.9% improvement (at 2004)
- both show long-term improvement the more training data is added

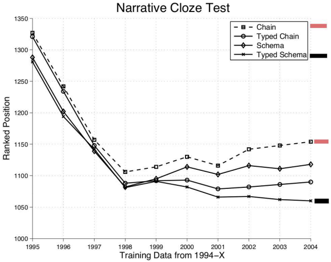
Evaluation - Narrative cloze



Untyped chains vs. untyped schemata:

- both use untyped chains ($chainsim_{untyped}$)
- 3.3% improvement (at 2004)
- again, both show long-term improvement the more training data is added

Evaluation - Narrative cloze



(full) schemata:

- typed chains outperform, untyped chains
- untyped schemata outperform, untyped chains
- combination, i.e. full schemata shows 10.1% improvement (at 2004)
- but: long-term improvement with more training data?

Conclusion

- *unsupervised* method to learn narrative chains and schemata
- key idea for chain learning: use *coreference* for event similarity
- temporal classifier: two-stage supervised method
- typed chains: using set of types for protagonist enhances chain learning
- types as semantic roles: event similarity helps learning thereof
- narrative schemata: considering all dependencies of a verb increases performance even more
- “narratives” because script information is not explicit in text

The End

Thank you! Let us discuss.

References

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

Chambers, N. and Jurafsky, D. (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.

Chambers, N., Wang, S., and Jurafsky, D. (2007). Classifying temporal relations between events. In *ACL '07: Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 173–176, Morristown, NJ, USA. Association for Computational Linguistics.