Lexical Event Ordering with an Edge-Factored Model

Omri Abend, Shay Cohen and Mark Steedman School of Informatics University of Edinburgh

June 2, 2015

Temporal lexical knowledge is useful for:

- Textual entailment
- Information extraction
- Tense and modality analysis
- Knowledgebase induction
- Question answering

We study a simple problem: lexical event ordering

## **Related Work**

Temporal relations between predicates (Chklovski and Pantel, 2004; Talukdar et al., 2012; Modi and Titov, 2014)

Binary classification of permutations (Chambers and Jurfasky, 2008; Manshadi et al., 2008)

Temporal lexicons (Regneri et al., 2010)

Finding stereotypical event order (Modi and Titov, 2014)

This paper:

- Conceptually simple model and inference
- Can include rich features in the learning problem
- General model can be used for other ordering problems (causality)
- Mostly relies on lexical information

Problem definition

Getting the data

Model

Inference and Learning

Experiments

Conclusion

What is an event? predicate (arguments)

What is an event? predicate (arguments)

Example of bag of events:



What is an event? predicate (arguments)

Example of bag of events:



Example of temporal ordering:

## **Getting the Data**

Wanted to avoid annotating data

Needed text where temporal order extraction is easy

## **Getting the Data**

Wanted to avoid annotating data

Needed text where temporal order extraction is easy



# **Preparing Recipes**

Downloaded 73K recipes from the web

Parsed them using the Stanford parser

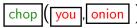
Verb with its arguments is an event

The devil is in the details. See paper

The dataset is available online: http://bit.ly/1Ge8wjj

Example:

"you should begin to chop the onion": chop (you, onion)

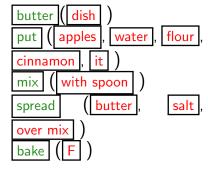


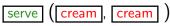
Butter a deep baking dish Put apples, water, flour, sugar and cinnamon in it

Mix with spoon

... and spread butter and salt over the apple mix

Bake at 350 degrees F until the apples are tender and the crust brown, about 30 minutes Serve with cream or whipped cream





A recipe for "Apple Crisp Ala [sic] Brigitte"

Examined 20 recipes (353 events)

13 events did not have a clear temporal ordering

Cases of mismatch mostly covered by:

• Disjunction:

"roll Springerle pin over dough, or press mold into top"

• Reverse order:

"place on greased and floured cookie sheet"

Average Kendall Tau between temporal ordering and linear one: 0.92

Represent all events in a recipe as a weighted complete graph

Each edge  $(e_1, e_2)$  is scored with a weight  $w(e_1, e_2)$ 

The larger the weight  $w(e_1, e_2)$ , the more likely event  $e_1$  to precede  $e_2$ 

A temporal ordering is a Hamiltonian path p in that graph

The score of a path:  $\label{eq:score} \mathrm{score}(p) = \sum_{(e_i,e_j) \in p} w(e_i,e_j)$ 

The edge weights are parametrized by  $\theta \in \mathbb{R}^m$ :

$$w(e_1, e_2) = \sum_{i=1}^{m} \theta_i f_i(e_1, e_2)$$

Features:

- Combinations of predicates and arguments of  $e_1$  and  $e_2$
- Combinations of their Brown clusters
- Point-wise mutual information between predicates and arguments

To do learning, we need

#### An inference algorithm

- Find the highest scoring Hamiltonian path
- An NP-hard problem
- No triangle inequality even approximation is hard
- Used Integer Linear Programming

#### An estimation algorithm for $\theta$

• Used the Perceptron algorithm

## **Integer Linear Programming Inference**

$$\begin{array}{ll} \max_{u_i \in \mathbb{Z}, z_{ij} \in \{0,1\}} & \sum_{i \neq j}^n w(e_i, e_j) z_{ij} \\ \\ \text{such that} & \sum_{\substack{i=1 \\ n}}^n z_{ij} = 1 & \forall i \\ & \sum_{\substack{j=1 \\ u_j - u_i \geq 1 - n(1 - z_{ij})} & \forall (i, j) \end{array}$$

Interpretation:

• 
$$z_{ij} - is (e_i, e_j) \in p?$$

•  $u_i$  – number of edges between start to  $e_i$  in p

- Also experimented with a conditional log-linear model
- It scores the probability  $p(e_2|e_1)$
- Induces a Markovian model over Hamiltonian paths
- Trained using log-likelihood maximization
- Greedy decoding is better than global decoding

#### Features:

- Frequency features estimated from "unlabeled" corpus
- Lexical features
- Brown cluster features
- Linkage frequency: joint occurence with temporal discourse connective
- **Evaluation:** To compare two Hamiltonian paths:
  - Count the number of "concordant pairs" (or tuples)
  - Divide by the total number of pairs
- In addition, we also checked the fraction of exact match

We used two ILP time budgets: 5 seconds and 30 seconds

4K training data

Results on dev set with perceptron:

Budget	Features	Pair-accuracy	Exact
30 secs	Frequency	68.7	31.7
	Frequency + Lexical	68.9	32.1
	Frequency + Lexical + Brown	68.4	31.8
5 secs	Frequency	65.9	30.4
	Frequency + Lexical	66.2	30.7
	Frequency + Lexical + Brown	66.3	30.4

### **Final Results**

Random baseline: 50% (0.5% exact)

Train size	Method	Pair-accuracy	Exact
4K	Perceptron (30 secs)	71.2	35.1
	Greedy Perceptron	60.8	20.4
	Greedy Log-linear	65.6	21.0
	Perceptron (5 secs)	68.9	34.4
58K	Greedy Perceptron	60.7	20.5
	Greedy Log-linear	66.3	21.3

Global model better than local log-linear model

Budget is more important than train size

PMI features were trained on 58K instances

#### Summary:

- Showed what the lexcial event temporal ordering problem is
- Described a domain in which data is easy to get
- Used structured prediction to solve the problem
- Method can be used for general ordering problems (causality, etc.)

#### Future Work:

- Future work: improved inference
- Different domains