Information Extraction

Natalie Parde, Ph.D.

Department of Computer Science

University of Illinois at Chicago

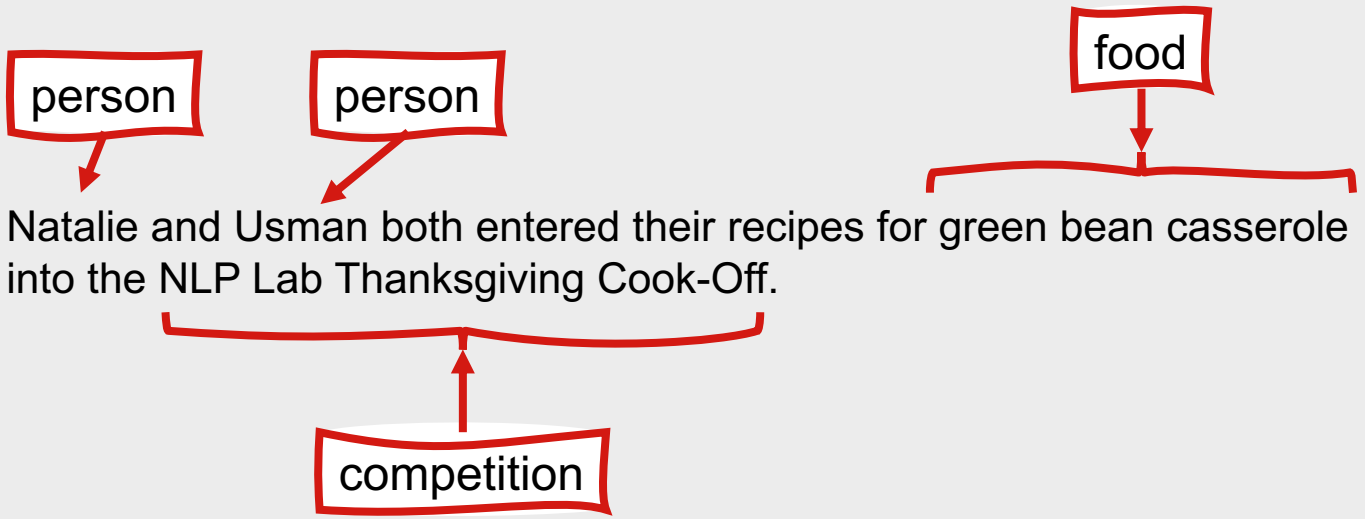
CS 421: Natural Language Processing

Fall 2019

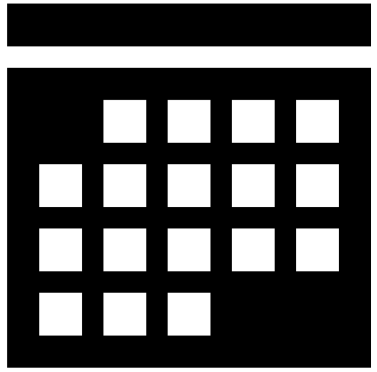
Many slides adapted from Jurafsky and Martin (https://web.stanford.edu/~jurafsky/slp3/).

What is information extraction?

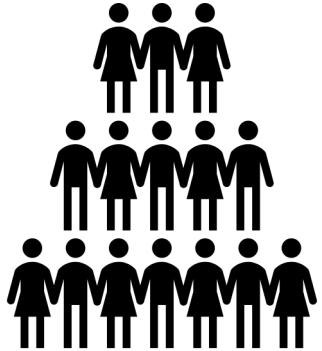
- The process of extracting **structured information** from **unstructured text**



Structured information can come in many different forms.



- Names
- Dates
- Amounts
- Relations
- Events



What are some of the most common types of information extraction?

Named entity recognition

Relation detection

Event extraction

Temporal analysis

Named Entity Recognition

- What are **named entities**?
  - Entities that can be referred to with proper names (e.g., proper nouns)
    - Natalie
    - Natalie Parde
    - Chicago
    - University of Illinois at Chicago
- **Named Entity Recognition:** The process of (a) **finding spans of text** that constitute proper names, and (b) **assigning entity type labels to those spans** of text

Generic Named Entity Types

Type	Tag	Sample Categories
People	PER	Individuals, fictional characters, small groups
Organization	ORG	Companies, agencies, political parties, religious groups, sports teams
Location	LOC	Physical extents, mountains, lakes, seas
Geo-Political Entity	GPE	Countries, states, provinces, counties
Facility	FAC	Bridges, buildings, airports
Vehicles	VEH	Planes, trains, automobiles

Generic Named Entity Types

Type	Tag	Sample Categories
People	PER	Individuals, fictional characters, small groups
Or		
Location	LOC	Physical extents, mountains, lakes, seas
Geo-Political Entity	GPE	Countries, states, provinces, counties
Facility	FAC	Bridges, buildings, airports
Vehicles	VEH	Planes, trains, automobiles

Natalie couldn't stop thinking about cheesy potatoes and green bean casserole as she prepared for the Thanksgiving holiday.

Generic Named Entity Types

Type	Tag	Sample Categories
People	PER	Individuals, fictional characters, small groups
Organization	ORG	Companies, agencies, political parties, religious groups, sports teams
Lo		s,
Ge		es,
Facility	FAC	Bridges, buildings, airports
Vehicles	VEH	Planes, trains, automobiles

She also couldn't wait to enjoy an entire day of traditional TV programming: the **Macy**'s Thanksgiving Day Parade, followed by the **American Kennel Club**'s national dog show, followed by the big **Dallas Cowboys** game.

Generic Named Entity Types

Type	Tag	Sample Categories
People	PER	Individuals, fictional characters, small groups
Organization	ORG	Companies, agencies, political parties, religious groups, sports teams
Location	LOC	Physical extents, mountains, lakes, seas
Geo-Political Entity	GPE	Countries, states, provinces,
		orts
Vehicles	VEH	Planes, trains, automobiles

To make sure she didn't feel too guilty about eating and watching TV for an entire day (or four), Natalie went on a healthy jog around **Millennium Park**.

Generic Named Entity Types

Type	Tag	Sample Categories
People	PER	Individuals, fictional characters, small groups
Organization	ORG	Companies, agencies, political parties, religious groups, sports teams
Location	LOC	Physical extents, mountains, lakes, seas
Geo-Political Entity	GPE	Countries, states, provinces, counties
Facility	FAC	Bridges, buildings, airports
Ve		s

Then she packed her suitcase for her journey from **Illinois** to **Texas**.

Generic Named Entity Types

Type	Tag	Sample Categories
People	PER	Individuals, fictional characters, small groups
Organization	ORG	Companies, agencies, political parties, religious groups, sports teams
Location	LOC	Physical extents, mountains, lakes, seas
G		ces,
Facility	FAC	Bridges, buildings, airports
Vehicles	VEH	Planes, trains, automobiles

Although she was excited about her trip, she was not excited to be flying from **O'Hare Airport** to **DFW Airport** on the busiest travel day of the year.

Generic Named Entity Types

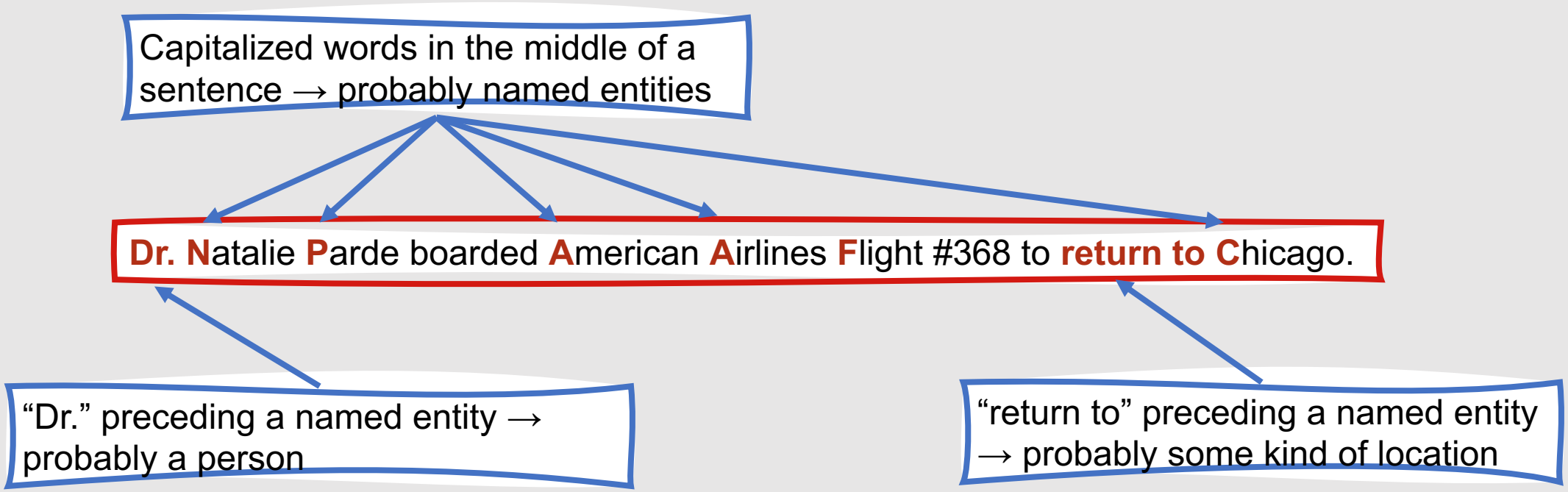
Type	Tag	Sample Categories
People	PER	Individuals, fictional characters, small groups
Organization	ORG	Companies, agencies, political parties, religious groups, sports teams
Location	LOC	Physical extents, mountains, lakes, seas
Geo-Political Entity	GPE	Countries, states, provinces,
Fac		
Vehicles	VEH	Planes, trains, automobiles

Still, the food would be worth it, so she crowded her way onto **Blue Line #143** to begin her Thanksgiving vacation.

Different systems may require different entity types!

- Commercial products
- Works of art
- Types of food
- Common traits that all systems will be interested in:
  - The (fixed) set of relevant named entity types
  - The ways that individual instances of those types are signaled in unstructured text

What are some ways named entities might be signaled in unstructured text?



Often treated the same as named entities….


Dates


Times


Named events


Measurements

Counts


Prices

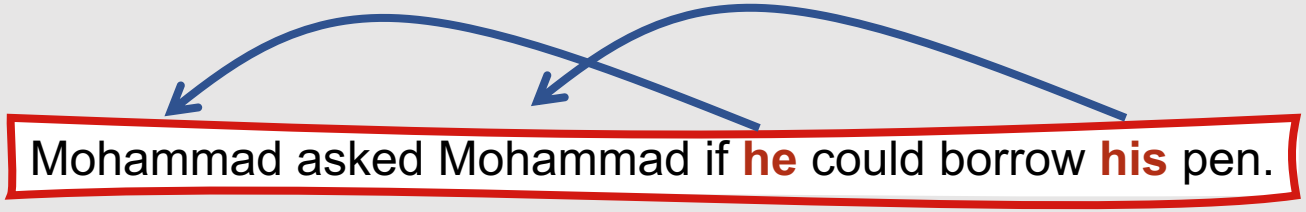
Ambiguity in Named Entity Recognition

Same name can sometimes refer to different entities of the same type

- Mohammad
- John
- George Bush

Essentially a coreference resolution issue

- **Coreference Resolution:** The task of identifying all expressions in a sample of text that refer to the same entity (even if the expressions differ from one another)



Ambiguity in Named Entity Recognition

Same name can sometimes refer to entities of different types

- DFW (airport or metropolitan area?)
- Paris (Hilton or France?)
- ACL (Association for Computational Linguistics or Austin City Limits or ligament in your knee?)

Sometimes these ambiguities are coincidental

- ACL

Other times, they are cases of metonymy

- **Metonymy:** The act of substituting a part for a whole
  - "What did the White House do today?"

Named Entity Recognition as Sequence Labeling

- How is named entity recognition typically approached?
  - Sequence labeling task
  - Assigned tags capture both (a) the boundary and (b) the type of detected named entities
- For each token:
  - Is it the beginning of an entity span?
    - If so, what type of entity is beginning?
  - Is it inside an entity span?
  - Is it outside any entity span?

What features are typically useful for named entity recognition?

Feature	Description
Lexical items	The words themselves
Stemmed lexical items	Stemmed versions of those words
Shape	The orthographic pattern of the words (word case, presence/position of numbers and punctuation, etc.)
Character affixes	Prefixes, suffixes, etc.
Part of speech	Part-of-speech tags for the words
Syntactic chunk labels	Constituency type
Gazetteer or name list	Presence of the word in a predefined list of names belonging to some entity type
Predictive tokens	Presence of words that often precede/proceed specific named entity types
Bag of words/Bag of n-grams	Words and/or n-grams in the surrounding context

Different features are useful for different applications.

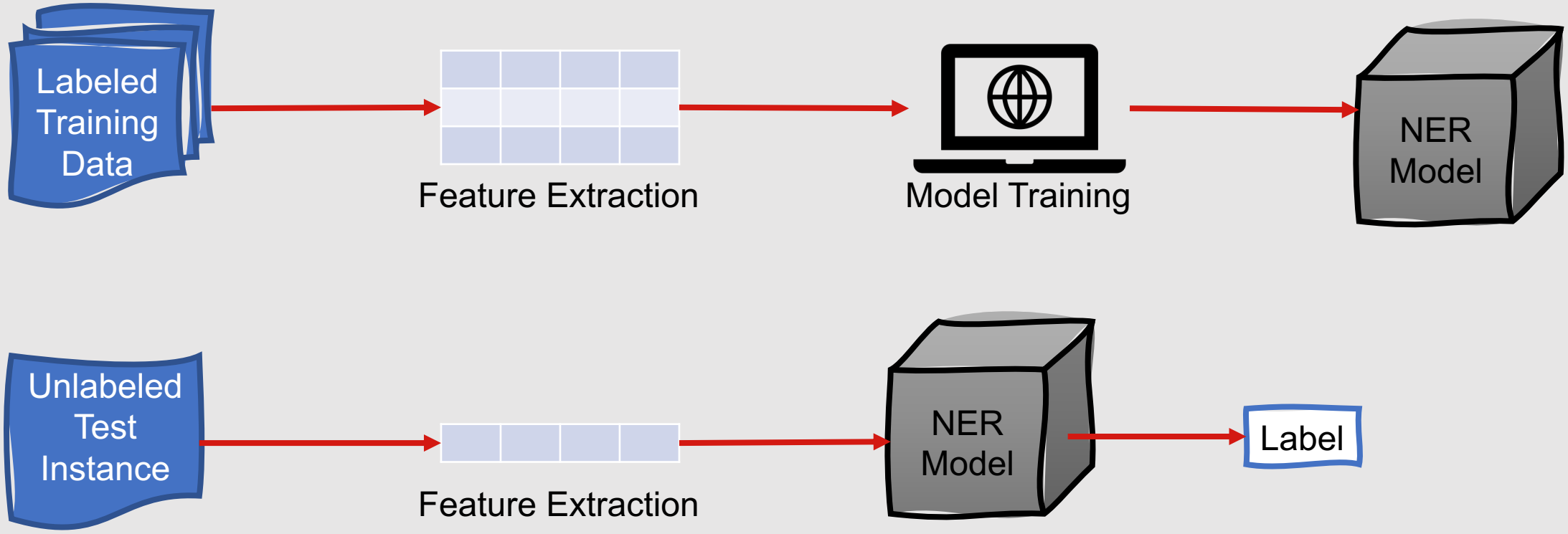
Shape features:

- Very useful for English news texts
- Not useful for transcribed speech
- Not useful for languages for which case information isn't available

Name list features:

- Useful for entity types that are relatively closed
  - Names of geographic landmarks
- Not useful for entity types that are relatively open
  - Names of people or organizations

What does an overall named entity recognition system look like?



How are commercial NER systems generally structured?

Combination of **rules** (tend to have very high precision but low recall) and **probabilistic methods** (tend to have lower precision but higher recall)

First pass: Use high-precision rules to **tag unambiguous entity mentions**

Next: Use probabilistic string matching techniques to **find substring matches of previously detected names**

Then: **Consult application-specific name lists** to identify likely named entity mentions for the application domain

Finally: Apply **probabilistic sequence labeling techniques** that make use of these and other features

Precision

Recall

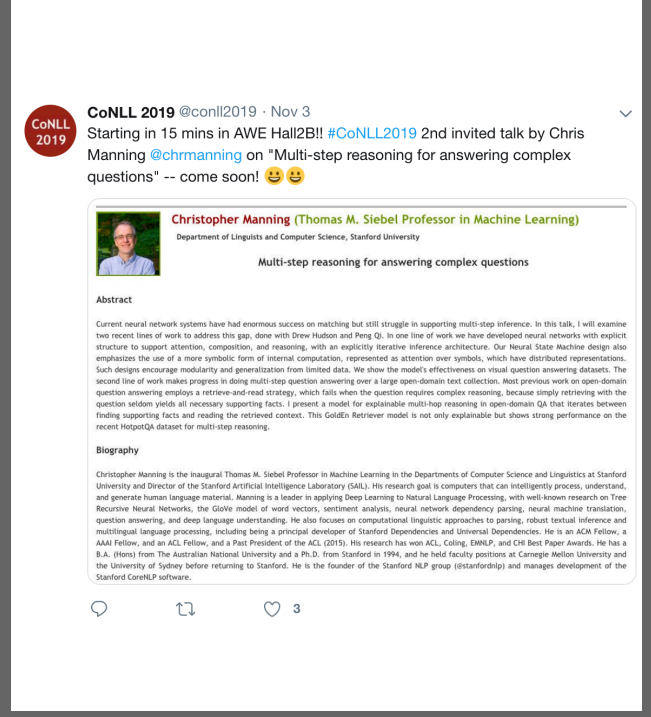
F_1

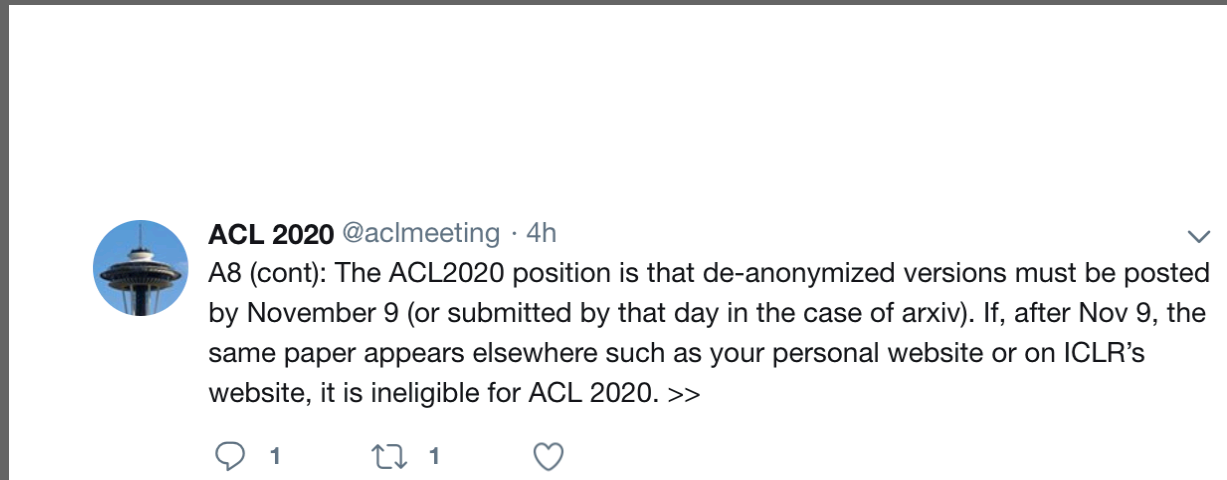
How are named entity recognition models evaluated?

In-Class Exercise

- Given the following named entity types, assign named entity labels to the specified tweets.
  - Person
  - Organization
  - Location
  - Geo-Political Entity
  - Facility
  - Vehicle

- https://www.google.com/search?q=timer





NLTK: https://www.nltk.org/

```
import nltk
from nltk.tokenize import word_tokenize
from nltk.tag import pos_tag
text = "The ACL2020 position is that de-
 anonymized versions must be posted by
 November 9."
text = nltk.word_tokenize(text)
text = nltk.pos_tag(text)
entities = nltk.chunk.ne_chunk(text)
entities
Tree('S', [('The', 'DT'),
 Tree('ORGANIZATION', [('ACL2020',
 'NNP')]), ('position', 'NN'), ('is',
 'VBZ'), ('that', 'IN'), ('de-
 anonymized', 'JJ'), ('versions', 'NNS'),
 ('must', 'MD'), ('be', 'VB'), ('posted',
 'VBN'), ('by', 'IN'), ('November',
 'NNP'), ('9', 'CD'), ('.', '.')])
```

spaCy: https://spacy.io/

```
import spacy
nlp = spacy.load("en")
text = nlp("The ACL2020 position is that
 de-anonymized versions must be posted by
 November 9.")
for i in range(0,len(text)):
print([text[i].text, text[i].ent_type_])
['The', ''], ['ACL2020', 'PERSON'],
 ['position', ''], ['is', ''], ['that',
 ''], ['de', ''], ['-', ''],
 ['anonymized', ''], ['versions', ''],
 ['must', ''], ['be', ''], ['posted',
 ''], ['by', ''], ['November', 'DATE'],
 ['9', 'DATE'], ['.', '']
```

Popular Tools for Named Entity Recognition

Relation Detection

- What are **semantic relations**?
  - Labels indicating how named entities are related to one another
    - Chicago is **northeast of** Dallas
    - Natalie **works at** University of Illinois at Chicago
- **Relation Detection:** The process of automatically **determining which relationships exist between the named entities** detected in a text

Generic Semantic Relations

Relations		Named Entity Types
Affiliations		
	Personal	PER → PER
	Organizational	PER → ORG
	Artifactual	(PER | ORG) → ART
Geospatial		
	Proximity	LOC → LOC
	Directional	LOC → LOC
Part-Of		
	Organizational	ORG → ORG
	Political	GPE → GPE

Generic Semantic Relations

Relations		Named Entity Types
Affiliations		
sister of	Personal	PER → PER
president of	Organizational	PER → ORG
owns	Artifactual	(PER | ORG) → ART
Geospatial		
	Proximity	LOC → LOC
	Directional	LOC → LOC
Part-Of		
	Organizational	ORG → ORG
	Political	GPE → GPE

Generic Semantic Relations

Relations		Named Entity Types
Affiliations		
	Personal	PER → PER
	Organizational	PER → ORG
	Artifactual	(PER | ORG) → ART
Geospatial		
near	Proximity	LOC → LOC
northeast of	Directional	LOC → LOC
Part-Of		
	Organizational	ORG → ORG
	Political	GPE → GPE

Generic Semantic Relations

Relations		Named Entity Types
Affiliations		
	Personal	PER → PER
	Organizational	PER → ORG
	Artifactual	(PER | ORG) → ART
Geospatial		
	Proximity	LOC → LOC
	Directional	LOC → LOC
Part-Of		
unit of	Organizational	ORG → ORG
annexed	Political	GPE → GPE

How can we automatically identify semantic relations?

- Similar to many of the NLP tasks explored thus far …**supervised classification** 👩‍💻
- Typical approaches break relation detection into two subtasks:
  - Detecting when a relation is present between two entities
  - Classifying any detected relations

How is this done?

Subtask #1 (Is a relationship present?)

- Extract positive samples of related named entities from a corpus
- Generate negative samples from within-sentence entity pairs that are not annotated with a relation
- Train a classifier to make a binary decision based on this data

Subtask #2 (What types of relationships are present?)

- Set one label as the positive class, and all others as the negative class
- Repeat to create a bunch of one-versus-many classifiers
- Pass the test instance to each of those classifiers
- Choose the label from the classifier with the highest confidence

What types of features are useful for this?

Characteristics of the named entities themselves

- Named entity types
- Words included in the named entities

Other words in the text

- Text between the two named entities
- Text before the first named entity
- Text after the last named entity
- Distance between the named entities
- Number of other named entities between the two that are specified

Syntactic structure

- Constituent parse
- Dependency parse

Lightly Supervised Approaches to Relation Analysis

- In order for supervised machine learning to work effectively, it requires a large amount of training data
- For real-world scenarios, assuming the availability of large amounts of relation-annotated text may be impractical
- How else can we extract training samples?
  - **Regular expression patterns** that match text segments likely to contain expressions of the relation(s) in which we are interested

Imagine we want to find all the hub cities that different airlines utilize

We design the following pattern:

.+ has a hub at .+

Then, we search for sentences matching the pattern in a large text corpus

Extracting Samples Using Regular Expression Patterns

We would end up getting (hopefully) plenty of good results….

American Airlines **has a hub at** Philadelphia International Airport

American, Wilmington's biggest airline, **has a hub at** Miami

Rather than fighting at a disadvantage against American Airlines, which **has a hub at** Reagan Airport

Its main base is Ben Gurion Airport, serving Tel Aviv, and it **has a hub at** Liège Airport (Belgium).

US Airways **has a hub at** Phoenix Sky Harbor International Airport

New Mexico Airlines (which **has a hub at** ABQ)

Atlanta-based Delta Air Lines has its largest hub at Hartsfield-Jackson and also **has a hub at** Detroit Metro

United, which **has a hub at** Bush Intercontinental (IAH)

… but also (probably) some bad ones.

Specifically, we find that: (1) incumbents lower price by more when the potential entrant **has a hub at** one or both market endpoints;

The spindle **has a hub at** each end to keep the roll in position.

Who **has a hub** at CVG?

Suppose that airline *X* is a hub carrier that **has a hub at** *A*

We'd also have another issue.

- Are we certain that we've found *all* the hubs for *all* airlines?
- Assuming that we'll find them all using a single pattern may be overly optimistic
  - Minor variations would cause the original pattern to fail!

Although UPS **has had a hub at** Louisville since 1980

How do we solve this problem?

- Option 1: **Generalize our pattern** so that it captures additional similar expressions
  - For example, allow matches that skip parts of the candidate text
- Pros:
  - **Easy to implement**
- Cons:
  - Will probably **introduce more false positives**

He **has** previously purchased **a** USB **hub at** their Chicago location.

How do we solve this problem?

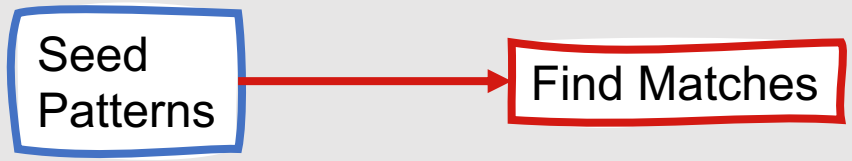
- Option 2: **Expand the set of specific patterns**
- Given a large and diverse document collection, an expanded set of patterns should be able to capture more information of interest
- How can we expand the set of specific patterns?
  - Manually create them
  - Automatically induce new patterns by **bootstrapping** them from a small set of **seed patterns**

Bootstrapping -based Pattern Extraction

- Given a seed pattern, find all of the text matches
- Extract tuples (e.g., related named entities) from those matches
- Find all of the texts containing those tuples
- Extract new patterns from those texts
- Repeat using those patterns as new seed patterns

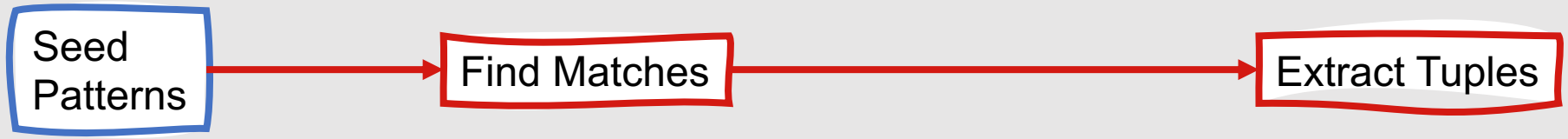
Bootstrapping-based Pattern Extraction

.* has a hub at .*

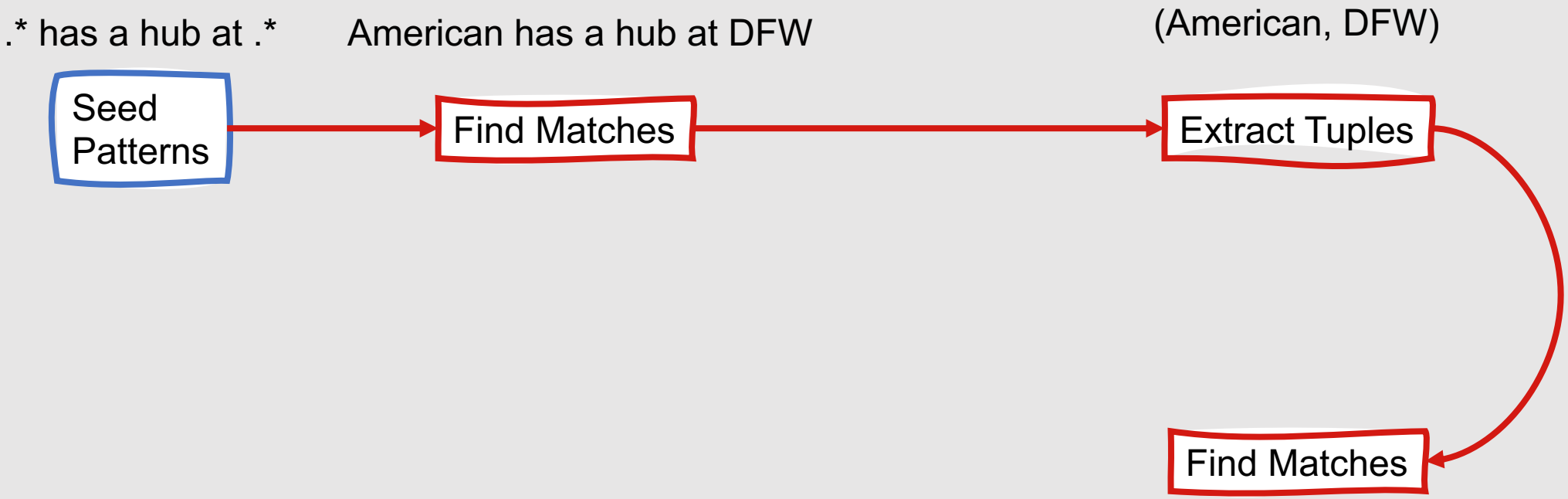


Bootstrapping-based Pattern Extraction

.* has a hub at .* American has a hub at DFW

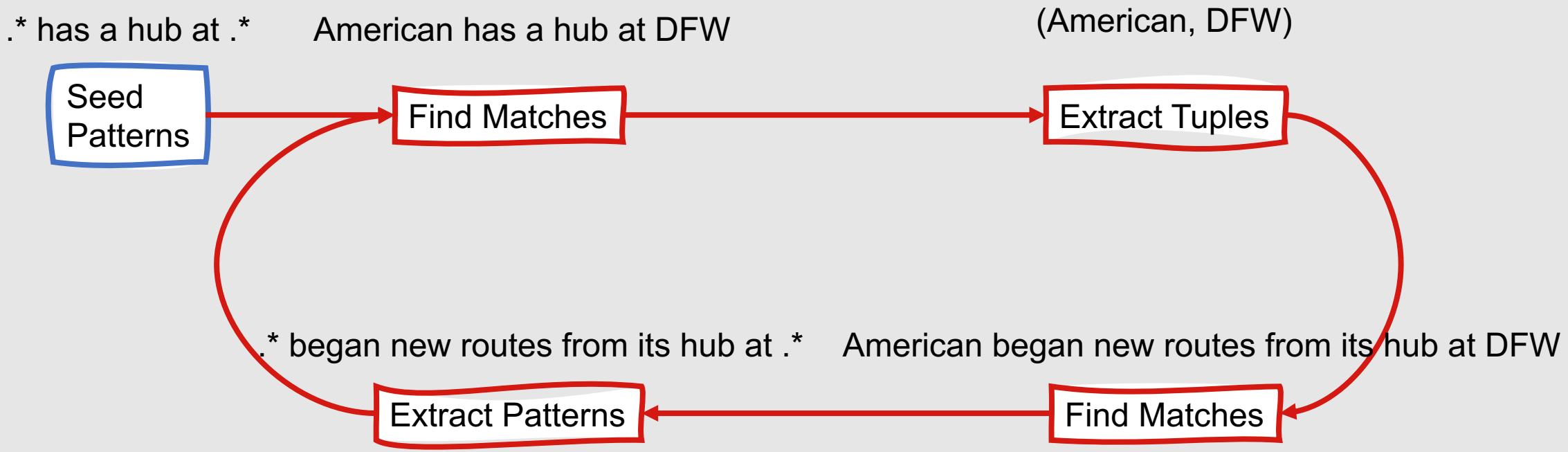


Bootstrapping-based Pattern Extraction



Bootstrapping-based Pattern Extraction

Bootstrapping-based Pattern Extraction



What should the extracted patterns look like?

- Patterns should typically be represented in a way that captures the following factors:
  - Context before the first entity mention
  - Context between the entity mentions
  - Context after the last entity mention
  - The order of the entities in the pattern
- These patterns are usually represented using **regular expressions** or **feature vectors**
  - Regular expressions tend to be more specific, leading to **higher precision**
  - Feature vectors tend to be more general, leading to **higher recall**

How do we determine the quality of extracted patterns?

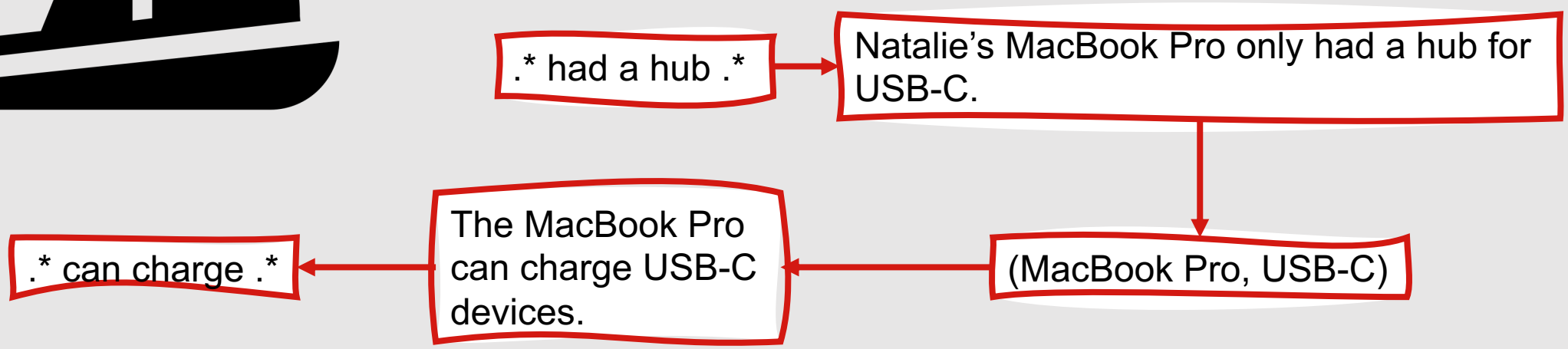
Remember, we don't have access to labeled data telling us whether our patterns are good!

We need to ensure that:

Our seed sets (for which we do have labeled data) are high quality	We don't permit significant **semantic drift** to occur as we're learning new patterns and tuples

Semantic Drift

- Occurs when bad pattern matches lead to the introduction of bad tuples, which in turn, lead to the creation of even worse patterns



Avoiding Semantic Drift

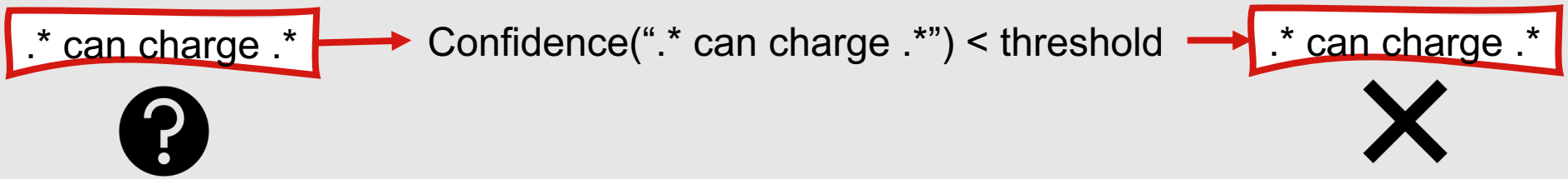
- To avoid semantic drift, we need to assess two factors for a proposed new pattern:
  - The pattern's performance with respect to the current set of tuples
  - The pattern's productivity in terms of the number of matches it produces in the document collection

Avoiding Semantic Drift

- We can refer to these as:
  - **Hits:** The number of unique tuples in *T* (the current set of tuples) that a proposed pattern *p* matches when searching through a document collection *D*
  - **Finds:** The total number of unique tuples (not only those in *T*) that *p* finds in *D*
- We can then compute the following equation (Riloff and Jones, 1999) to balance these considerations:
  - $\text{Confidence}(p) = \frac{hits_p}{finds_p} \times \log finds_p$

Avoiding Semantic Drift

- We can then assess the confidence in proposed tuples and patterns as the bootstrapping process iterates
- If confidence is greater than a predefined threshold, we can accept the pattern/tuple; otherwise, we can ignore it



Evaluating Relation Detection Systems

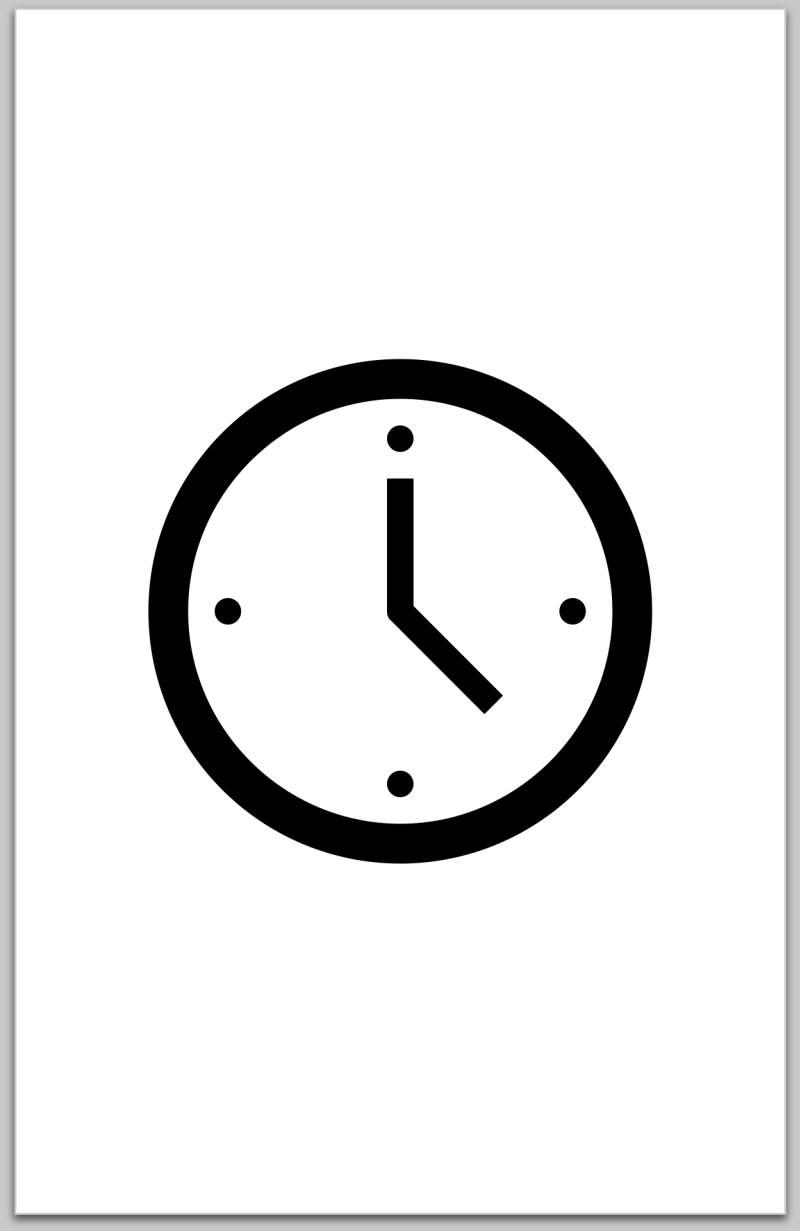
- Supervised methods:
  - Precision
  - Recall
  - F1
- Unsupervised methods:
  - Human annotators can label predicted relations (allowing precision to be computed), but there is no easy way to measure recall
  - Some solutions:
    - Compute precision at different levels of recall
      - Precision for top 1000 new relations
      - Precision for top 10,000 new relations
    - Evaluate recall based on large resources with comprehensive lists of correct answers
      - However, the external resource may be more comprehensive than the text collections used by the relation extraction system

Temporal and Event Processing

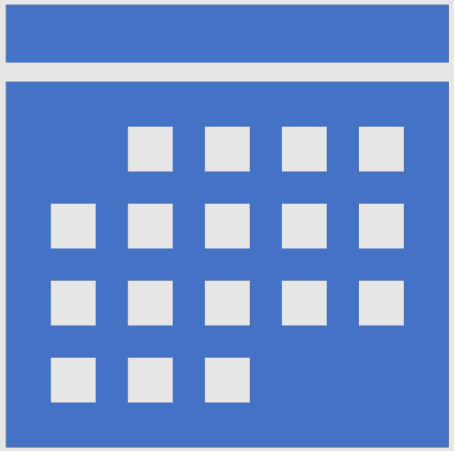
- Entities are often introduced in the course of describing the **events** in which they take part
- Temporal and event processing involves:
  - **Identifying events**
  - **Determining how they relate to one another in time**
- Particularly useful for question answering and summarization!

When **is** Thanksgiving?

Three weeks from today, Thanksgiving **will commence**!



Temporal Expression Recognition



- **Temporal Expressions:** Expressions that refer to:
  - **Absolute points in time**
    - Can be mapped directly to calendar dates and/or times of day
  - **Relative times**
    - Map to particular times through some other reference point
  - **Durations**
    - Spans of time at varying levels of granularity

Example Temporal Expressions

Absolute	Relative	Durations
November 8, 2019	Tomorrow	One day
Spring 2020	Next semester	Six weeks
9:30 a.m.	An hour ago	An hour
The third quarter of 2019	Last quarter	The last three quarters

Temporal Expressions

- Temporal expressions are generally triggered by specific nouns, adjectives, and adverbs
  - Morning, night, November, Ramadan, recent, annual, hourly, yearly
- Recognizing temporal expressions involves **finding the starting and ending points of text spans** corresponding to them, often based heavily on the locations of known **lexical triggers**
- Methods to do so can be implemented using:
  - Rule-based approaches
  - Sequence classifiers
  - Constituent-based classifiers

Rule-based Temporal Expression Recognition

- Cascades of automata that recognize patterns at increasing levels of complexity
  - Tokens are part-of-speech tagged
  - Larger and larger chunks of text are recognized from earlier stages, based on patterns containing trigger words

Sequence-based Temporal Expression Recognition

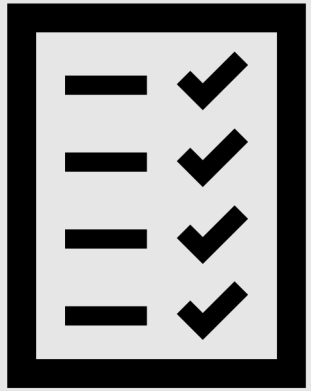
- Similar to other sequence labeling tasks:
  - Features are extracted from the context surrounding a token to be tagged
  - A statistical sequence labeler predicts a label based on those features as well as the previous decision(s)

Constituent-based Temporal Expression Recognition

- Constituent parse is automatically extracted
- Nodes in the resulting tree are classified based on whether or not they contain a temporal expression
  - Supervised, binary classification task

Evaluating Temporal Expression Recognizers

- As usual:
  - Precision
  - Recall
  - F-Measure

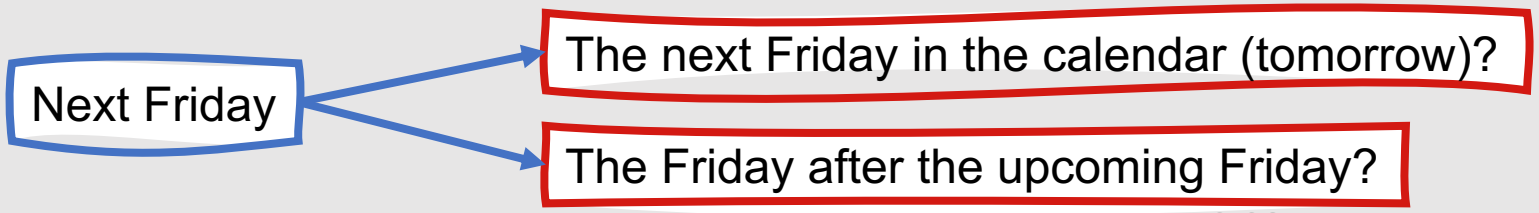


Temporal Normalization

- **Temporal Normalization:** The process of mapping a temporal expression to either:
  - A specific point in time
  - A duration
- Most current approaches are **rule-based**
- The meaning of a temporal expression is determined based on its constituent parts
  - **Fully qualified** temporal expressions
  - **Absolute** temporal expressions
  - **Relative** temporal expressions
  - **Durations**

Temporal Normalization

- Fully qualified temporal expressions
  - Contain a year, month, and day in some conventional form
  - Relatively rare (most temporal expressions are incomplete and/or implicitly anchored)
- **Temporal Anchor:** The time to which temporal expressions refer
  - Tomorrow → One day after today
  - Next week → One week after today
  - The day after Thanksgiving → One day after Thanksgiving
- Plenty of room for ambiguity!

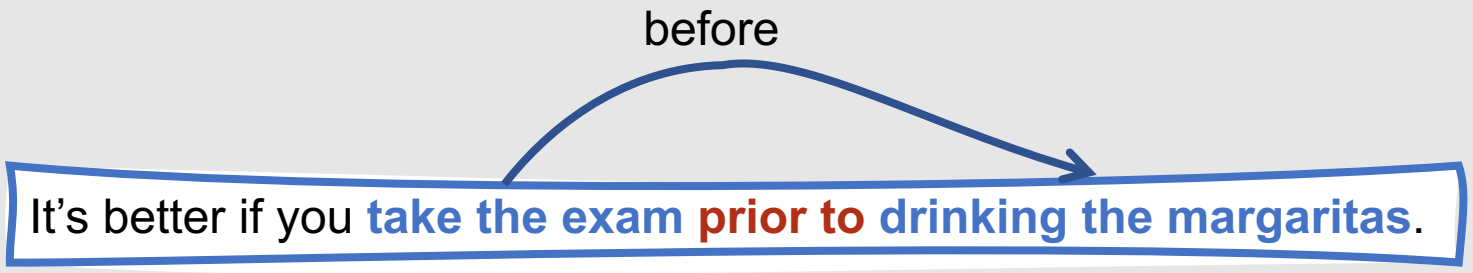


Event Detection and Analysis

- **Event Detection:** Identifying mentions of events in text, and assigning those events to specific classes
- What is an **event mention**?
  - Any expression denoting an event or state that can be assigned to a particular point, or interval, in time
  - In English, event mentions tend to be verbs
- Typically modeled using supervised machine learning

Temporal Ordering

- Once events and temporal expressions have been detected, they can (in theory) be used to create a **timeline**
  - Very challenging, open problem in NLP!
- Most methods currently seek to create **partial orderings** of events instead
  - Before, after, during



Template Filling

- Many sequences of events correspond to typical situations in the world, which can be characterized as **scripts** (Schank and Abelson, 1977)
- **Scripts generally contain prototypical sequences** of:
  - Sub-events
  - Participants
  - Roles
  - Props
- Scripts can be very useful for **drawing inferences** that fill in things that have been left unsaid

How are scripts represented?

- **Templates** consisting of fixed sets of **slots** that are filled with **values** belonging to particular classes
- **Template Filling:** The task of finding documents that invoke particular scripts, and then filling the slots in the associated templates with information extracted from the text
  - **Text segments** extracted directly from text
  - **Concepts** inferred through some additional processing

Template Filling

- Generally modeled by training **two separate supervised systems**
  - Template recognition
  - Role-filler extraction
- **Template recognition** system decides whether a given template is present in a particular sentence
- **Role-filler extraction** system contains separate classifiers to decide whether each possible role in the template is present

Natalie asked the waiter to bring another serving of green bean casserole.

Ordering at a restaurant
Customer: Natalie
Choice of Food: green bean casserole

Summary: Information Extraction

- **Information extraction** is the process of extracting **structured information** from **unstructured text**
  - Named entities
  - Semantic relations
  - Temporal expressions
  - Events
- **Named entities** are spans of text that can be referred to using proper names of various categories
- **Semantic relations** indicate how named entities are related to one another
- Semantic relation patterns can be extracted using **bootstrapping**
  - When doing so, you should be careful to avoid introducing **semantic drift**
- **Temporal expressions** refer to absolute or relative points in time, or durations
- **Event mentions** are any expressions (usually verbs) denoting events or states that can be assigned to a particular point or interval in time
- Extracted information can be used to construct **scripts**, which can in turn be used for drawing inferences in a variety of NLP tasks