Semantic Parsing and Information Extraction

2022–23

Ion Androutsopoulos

http://www.aueb.gr/users/ion/

These slides are partly based on material from the books:

- "Speech and Language Processing" by D. Jurafsky and J.H. Martin, 2nd edition, Prentice Hall, 2009 and 3rd edition (in preparation, https://web.stanford.edu/~jurafsky/slp3/).
- "Artificial Intelligence – A Modern Approach" by S. Russel and P. Norvig, 2nd edition, Prentice Hall, 2003.

Contents

- **Semantic parsing**:
    - Translating sentences to First-Order Predicate Logic (FOPL) using grammars.
    - Intent recognition and slot filling in dialog systems using grammars or neural models.
- **Information extraction**:
    - Named entity recognition and supervised relation extraction with neural models.

Additional optional material:

- Unsupervised relation extraction.
- Lexical semantic relations, WordNet, events, FrameNet, thematic roles, selectional restrictions.

Examples of formulae in First-Order Predicate Logic

- *All cats like milk.*

  $\forall x\ (\mathrm{IsCat}(x) \Rightarrow \mathrm{Likes}(x, \mathrm{Milk}))$

- *There is a cat that likes milk.*

  $\exists x\ (\mathrm{IsCat}(x) \land \mathrm{Likes}(x, \mathrm{Milk}))$

  – Attention: $\exists x\ (\mathrm{IsCat}(x) \Rightarrow \mathrm{Likes}(x, \mathrm{Milk}))$ says "There is an x that: (i) is not a cat; or (ii) if it is a cat, it likes milk".

- *Psita likes all dogs.*

  $\forall x\ (\mathrm{IsDog}(x) \Rightarrow \mathrm{Likes}(\mathrm{Psita}, x))$

  – Attention: $\forall x\ (\mathrm{IsDog}(x) \land \mathrm{Likes}(\mathrm{Psita}, x))$ says "Everything is a dog and Psita likes it".

- *There is a cat that likes all dogs.*

  $\exists x\ (\mathrm{IsCat}(x) \land \forall y\ (\mathrm{IsDog}(y) \Rightarrow \mathrm{Likes}(x, y)))$

Examples of formulae in First-Order Predicate Logic

- *Milos dislikes all cats.*

  $\forall x\ (IsCat(x) \Rightarrow \neg\ Likes(Milos, x))$

- *All dogs dislike all cats.*

  $\forall x\ (IsDog(x) \Rightarrow \forall y\ (IsCat(y) \Rightarrow \neg\ Likes(x, y)))$

  or:

  $\forall x\ \forall y\ ((IsDog(x) \wedge IsCat(y)) \Rightarrow \neg\ Likes(x, y))$

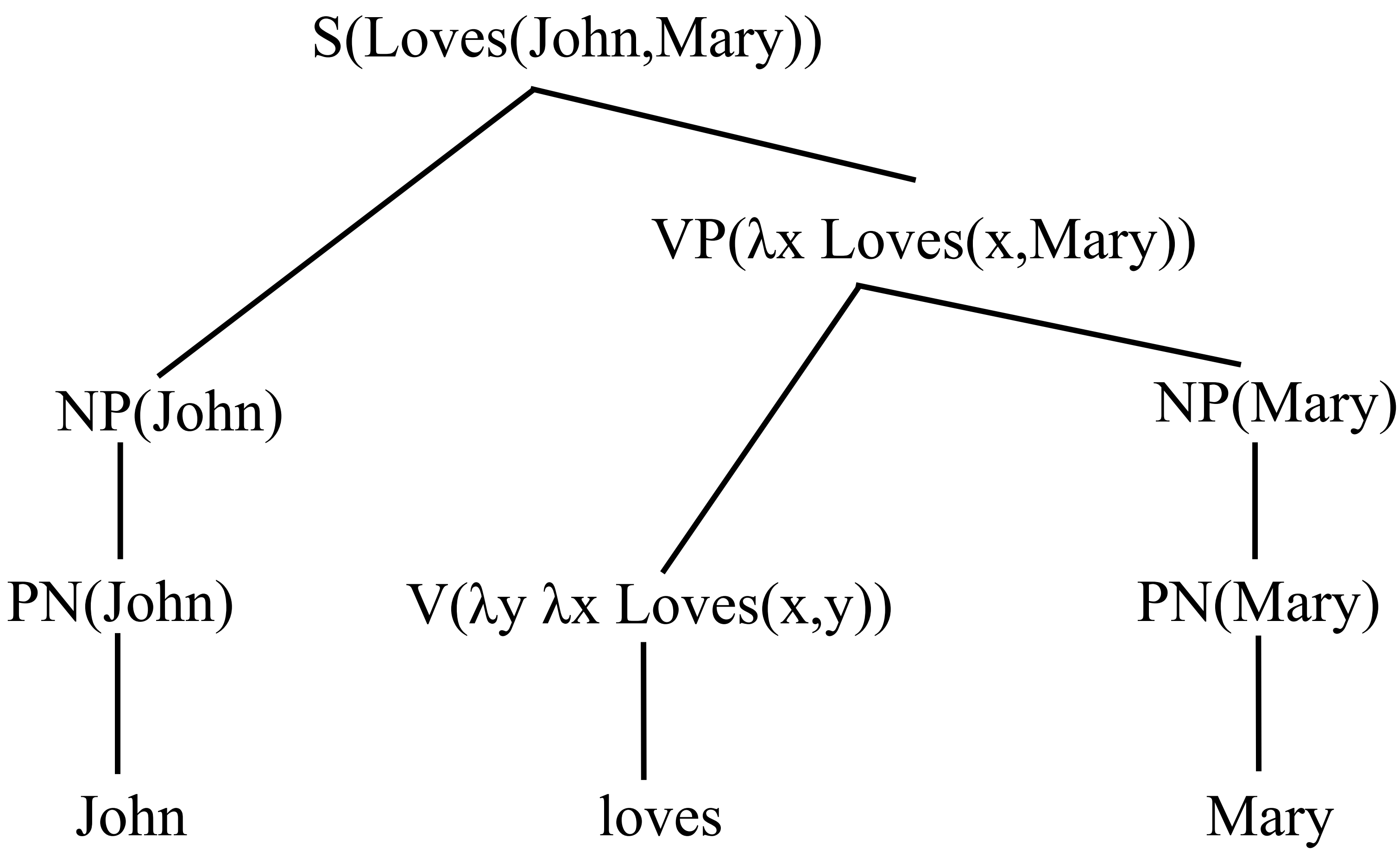
- *Every person likes his/her father.*

  $\forall x\ \forall y\ ((IsPerson(x) \wedge IsFatherOf(y, x)) \Rightarrow Likes(x, y))$

  or: $\forall x\ (IsPerson(x) \Rightarrow Likes(x, \mathbf{FatherOf(x)}))$

If we have a **large dataset** with **sentences** and the **corresponding FOPL formulae**, we can try using **neural machine translation models** to "translate" from English to FOPL. **Otherwise** one option is to use **grammars**…

Semantic parsing example



Semantics of simple sentences

$S(\sigma_2(\sigma_1)) \rightarrow NP(\sigma_1)\ VP(\sigma_2)$

$VP(\sigma_3(\sigma_4)) \rightarrow V(\sigma_3)\ NP(\sigma_4)$

$NP(\sigma) \rightarrow PN(\sigma)$

$PN(John) \rightarrow John$

$PN(Mary) \rightarrow Mary$

$V(\lambda y\, \lambda x\, Loves(x, y)) \rightarrow loves$

Compositional semantics: The semantics of each syntactic constituent is a function of the semantics of its sub-constituents.

$$\underbrace{(\lambda y\, \lambda x\, Loves(x, y))}^{\sigma_3}\underbrace{(Mary)}^{\sigma_4} \equiv \overbrace{\lambda x\, Loves(x, Mary)}^{\sigma_3(\sigma_4)}$$

$$\underbrace{(\lambda x\, Loves(x, Mary))}_{\sigma_2}\underbrace{(John)}_{\sigma_1} \equiv \underbrace{Loves(John, Mary)}_{\sigma_2(\sigma_1)}$$

Handling quantifiers

- We extend the grammar to handle sentences like:
  - "I want a flight from Athens to Thessaloniki."
  - "A customer prefers a flight to Herakleion."
- We will initially produce formulae with **quantifiers inside the arguments** of the predicates:
  - "Every customer wants a flight." will initially become:
    Wants(**$\forall$x Client(x)** , **$\exists$y Flight(y)**)
  - Not allowed in FOPL, but makes semantic parsing easier.
  - Also allows us to produce a single formula for sentences with **ambiguous quantifiers** (see below).
  - A **post-processing stage** will fix the formulae (see optional slides).

Semantics of nouns and determiners

$N(\ \lambda x_1\ Customer(x_1)\ ) \rightarrow$ customer

$Det(\ \lambda x_2\ \lambda p_1\ \forall x_2\ p_1\ ) \rightarrow$ every

$\forall x_3\ Customer(x_3)$ $\qquad\qquad$ $\lambda x_2\ \lambda p_1\ \forall x_2\ p_1$

$NP(\ \sigma_1(x_3)(\sigma_2(x_3))\ ) \rightarrow Det(\sigma_1)\ \ N(\sigma_2)$ (where x_3 is a new variable)

$\lambda p_1\ \forall x_3\ p_1$ $\qquad$ $Customer(x_3)$ $\qquad$ $\lambda x_1\ Customer(x_1)$

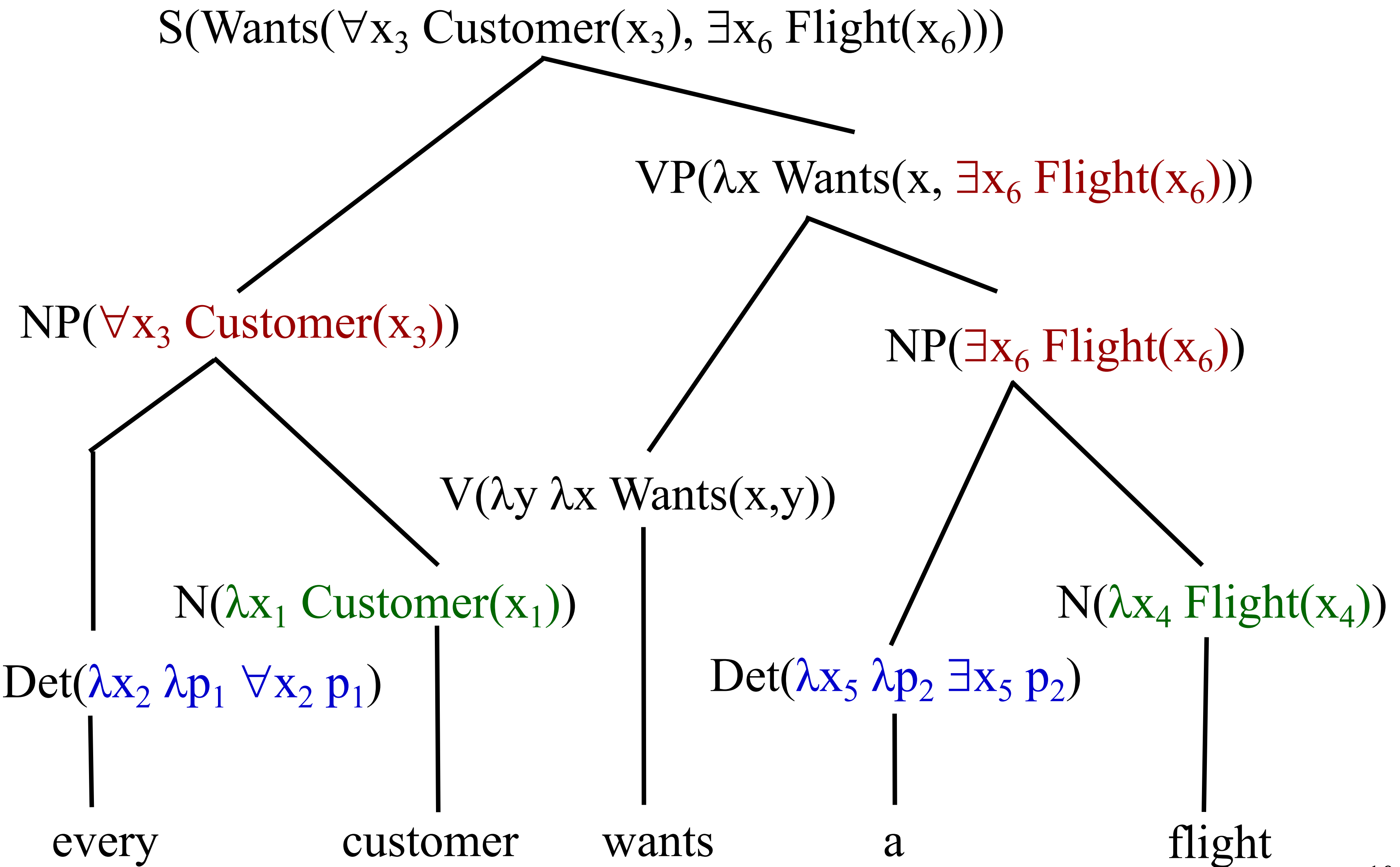
Similarly:

$N(\ \lambda x_4\ Flight(x_4)\ ) \rightarrow$ flight

$Det(\ \lambda x_5\ \lambda p_2\ \exists x_5\ p_2\ ) \rightarrow$ a

The other rules for verbs, VP, S remain unchanged.

Computing the semantics of a sentence



Converting to true FOPL

- A **post-processor** produces all the possible FOPL formulae from the intermediate formula.
  - Wants(∀x Customer(x), ∃y Flight(y)) becomes:
  - ∀x Customer(x) → (∃y Flight(y) ∧ Wants(x, y)) **or:**
  - ∃y Flight(y) ∧ (∀x Customer(x) → Wants(x,y))
- **Cut a quantifier** (and its predicate) from the intermediate formula **leaving only its variable**:
  - E.g., Wants(**x**, ∃y Flight(y))
- **Paste the quantifier** and its predicate at the **beginning of the formula**.
  - **Connect with** ⇒ if the quantifier is a ∀.
  - **Connect with** ∧ if the quantifier is a ∃.
  - **∀x Customer(x)** ⇒ Wants(x, ∃y Flight(y))

Converting to true FOPL – continued

- **Repeat** for the **other quantifier**:
  - $\forall x\ Customer(x) \Rightarrow Wants(x, y)$
  - $\exists y\ \mathbf{Flight(y)} \land (\forall x\ Customer(x) \Rightarrow Wants(x, y))$
  - There is a **single (the same for all customers) flight** that they all want.
- Cutting the existential quantifier first, leads to another reading:
  - $\forall x\ Customer(x) \Rightarrow (\exists y\ Flight(y) \land Wants(x, y))$
  - For every customer, there is a **possibly different flight** that the customer wants.
- The intermediate formula may also contain **pseudo-quantifiers representing referring expressions**.
  - "**The customer** wants a flight."
  - Wants(**The.x Customer(x)**, $\exists$y Flight(y))
  - The pseudo-quantifiers are replaced by appropriate expressions using algorithms that resolve referring expressions.

NLU in spoken dialogue systems

«Θέλω να πάω στο Ηράκλειο.» (voice)

↓

Speech recognition (ASR)

↓ [θέλω, **θα**, πάω, στο, **???**]

NL understanding (NLU)

↓ parameter-input(arrive-to, noise)

Dialogue manager ⟷ other systems

↓ parameter-ask(arrive-to)

NL generation (NLG)

↓ [πού, θέλετε, να, πάτε, ;]

Speech synthesis (TTS)

↓ *«Πού θέλετε να πάτε;»* (voice)

Dialogue systems based on frames

- Supporting **mixed initiative** is often easier with dialogue managers based on **frames**.

How may I help you?

I want to book a flight to Athens with Olympic Air tomorrow at five.

E.g., using a grammar.

requestType(booking) $\wedge$ date(tomorrow) $\wedge$ carrier(oa) $\wedge$ destination(ath) $\wedge$ departTime(17:00)

Where do you want to depart from?

requestType	booking	Do you want to book or change ticket?
carrier	OA	Which carrier do you want to fly with?
date	23/5/11	On which date do you want to fly?
departFrom		Where do you want to depart from?
destination	ATH	What is your destination?
departTime	17:00	What time do you want to depart?
…	…	…

Dialogue systems based on frames

- The **NLU** uses a **grammar** that covers **sentences** specifying the values of **any of the frame fields**.
    - Possibly also **fields** they system **hasn’t asked** about.
    - The grammar **extracts field values** from the sentences and **helps the ASR** prune unlikely word sequences.
- If a **field value** is **missing**, the **system** takes the **initiative** and **asks** its value.
    - **For each field**, the frame provides a **suitable question**.
    - The **user may or may not answer** the **particular question** and/or provide additional information, corrections etc.
- There may be **several frames** (e.g., for tickets, car, hotel).
    - There may be a **graph of frames** (e.g., book a ticket, then rent a car or book a hotel) and **classifiers activating frames**.

Instead of grammars

- **Instead of grammars**, we may rely on a **language model** and **sequence labeling** methods.
  - The **language model** helps the **ASR prune** unlikely word sequences. No grammar required, if the language model is good enough (e.g., lots of dialogue transcripts available).
  - **Sequence labeling** (e.g., with RNNs, BERT) **detects phrases** that correspond to **frame fields**. But **normalization** of the frame values needed (e.g., “to Athens” → “ATH”), possibly with **regular expressions or grammars**.

[$_{\text{ignore}}$ Good morning] [$_{\text{noise}}$ #$#$@#$] [$_{\text{ignore}}$ would like to] [$_{\text{requestType}}$ book] [$_{\text{noise}}$ @#$#$] [$_{\text{destination}}$ to Athens] [$_{\text{noise}}$ @#$@] [$_{\text{departTime}}$ at five] [$_{\text{noise}}$ #$$@#$] [$_{\text{date}}$ tomorrow] [$_{\text{noise}}$ $@##$$] [$_{\text{carrier}}$ with Olympic] [$_{\text{ignore}}$ please.]

Domain, intent, slot fillers

- In Siri, Alexa, Cortana, Google Now etc., we need to figure out the **domain** of the request, the **intent** of the user, and fill in the **slots** of the corresponding frame.

Show me morning flights from Boston to San Francisco on Tuesday

```
DOMAIN:       AIR-TRAVEL
INTENT:       SHOW-FLIGHTS
ORIGIN-CITY:  Boston
ORIGIN-DATE:  Tuesday
ORIGIN-TIME:  morning
DEST-CITY:    San Francisco
```

Examples from Jurafsky and Martin, 3rd edition.

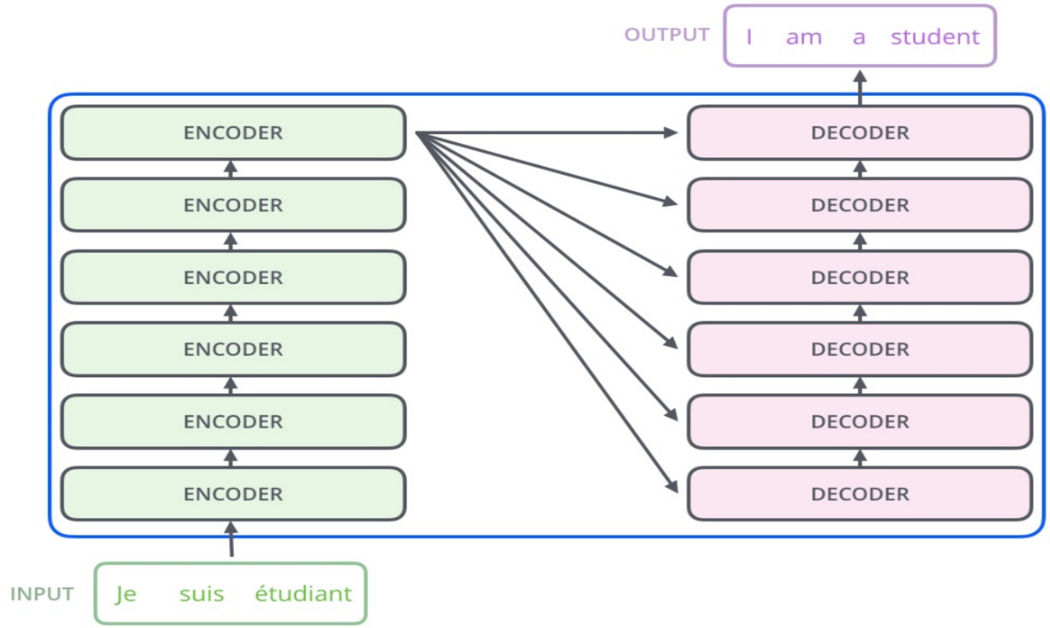
Wake me tomorrow at 6

```
DOMAIN:  ALARM-CLOCK
INTENT:  SET-ALARM
TIME:    2017-07-01 0600-0800
```

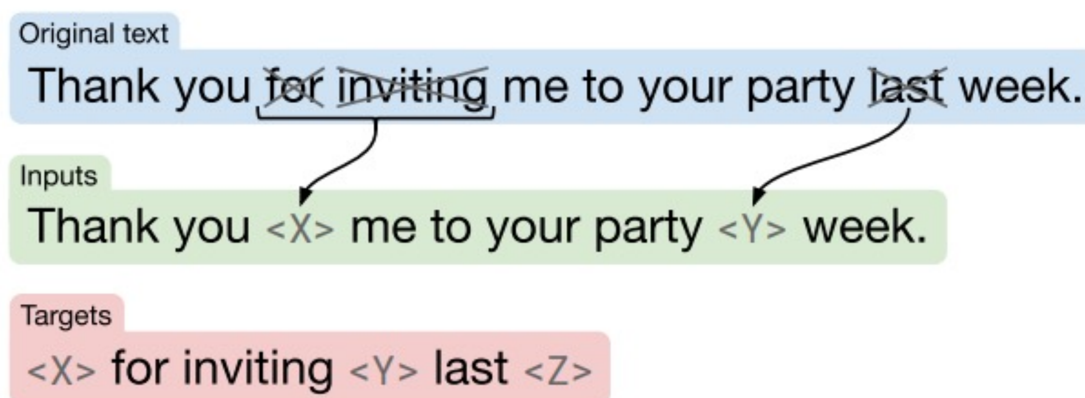
- There may be **grammars** for all these, or there may be **classifiers** predicting the **domain**, **intent**, and **sequence labeling** components extracting **slot values**.

Reminder: T5

T5 uses stacked **encoder** and stacked **decoder** Transformer layers.

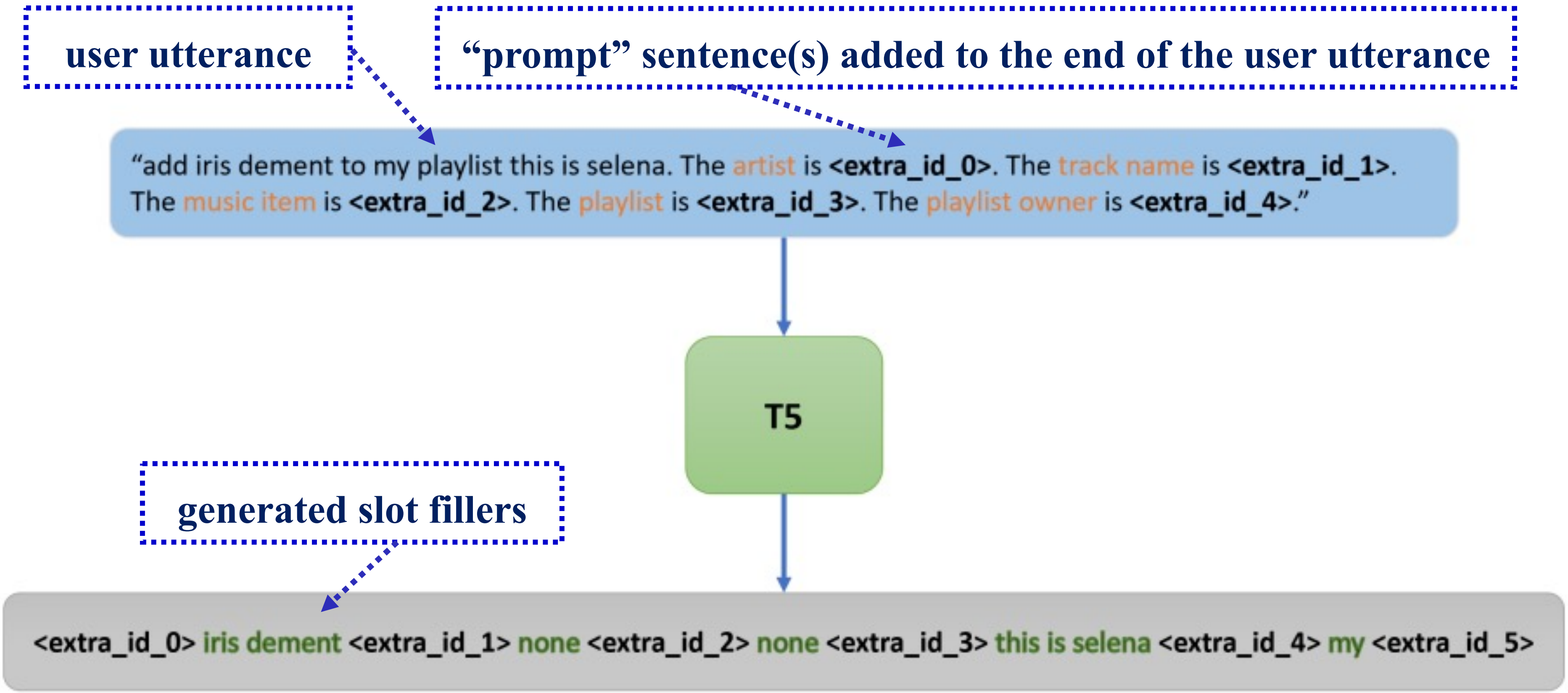


For unsupervised pre-training, T5 is trained to **recover missing/noised parts of the input**, here **masked spans**.



Top figure from J. Alammar's "The Illustrated Transformer" (https://jalammar.github.io/illustrated-transformer/). Bottom figure from the T5 paper: C. Raffel et al., "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer", JMLR 2020 (https://jmlr.org/papers/v21/20-074.html/).

T5-based slot filling (for known intent)



The model **can also learn to normalize** the fillers (e.g., “Athens” → “ATH”). And **no need to annotate** the **spans of the fillers** (no BIO tags) in the training examples.

Figure from P. Tassias, “A prompting-based encoder-decoder approach to intent recognition and slot filling”, MSc thesis in Data Science, AUEB, 2021.
http://nlp.cs.aueb.gr/theses/p_tassias_msc_thesis.pdf

Information extraction

Dec. 3, 2015: Important news from General Company Hellas, the largest Greek construction company. Yesterday GCH announced it bought 42% of Small Company Ltd, a British company that specializes in iron constructions.

⬇

buyer	GCH
bought	SCL
share	0.42
announcement date	2/12/2015

Information extraction: first stages

<s> <date norm="3/12/2015"> Dec. 3, 2015 </date> : </s>

<s> Important news from <company id="GCH"> General Company Hellas </company>, the largest Greek construction company. </s>

<s> <date norm="2/12/2015"> Yesterday </date> <company id="GCH"> GCH </company> announced it <verb base="buy"> bought </verb> <percent norm="0.42"> 42% </percent> of <company id="SCL"> Small Company Ltd </company>, a British company that specializes in iron constructions. </s>

Information extraction: first stages

- **Preprocessing**:
  - **Tokenization**, **sentence splitting**, **markup** processing etc.
- **Morphological** analysis:
  - **Part-of-speech tagging**, **lemmatization** etc.
- **Dates**, **amounts**, **percentages** etc.
  - Including **normalization** (e.g., "Dec. 3, 2015", 3/12/2015), often using simple regular expressions or grammars.
- **Named entity recognition** (e.g., persons, companies, locations, products, diseases, genes).
  - **Sequence labeling** algorithms (e.g., RNNs, BERT).

A neural NE recognizer

The **CRF** layer in effect adds a **second loss** (apart from maximizing the log-likelihood of the correct labels) that **penalizes unlikely tag sequences** (e.g. O followed by I-PER).

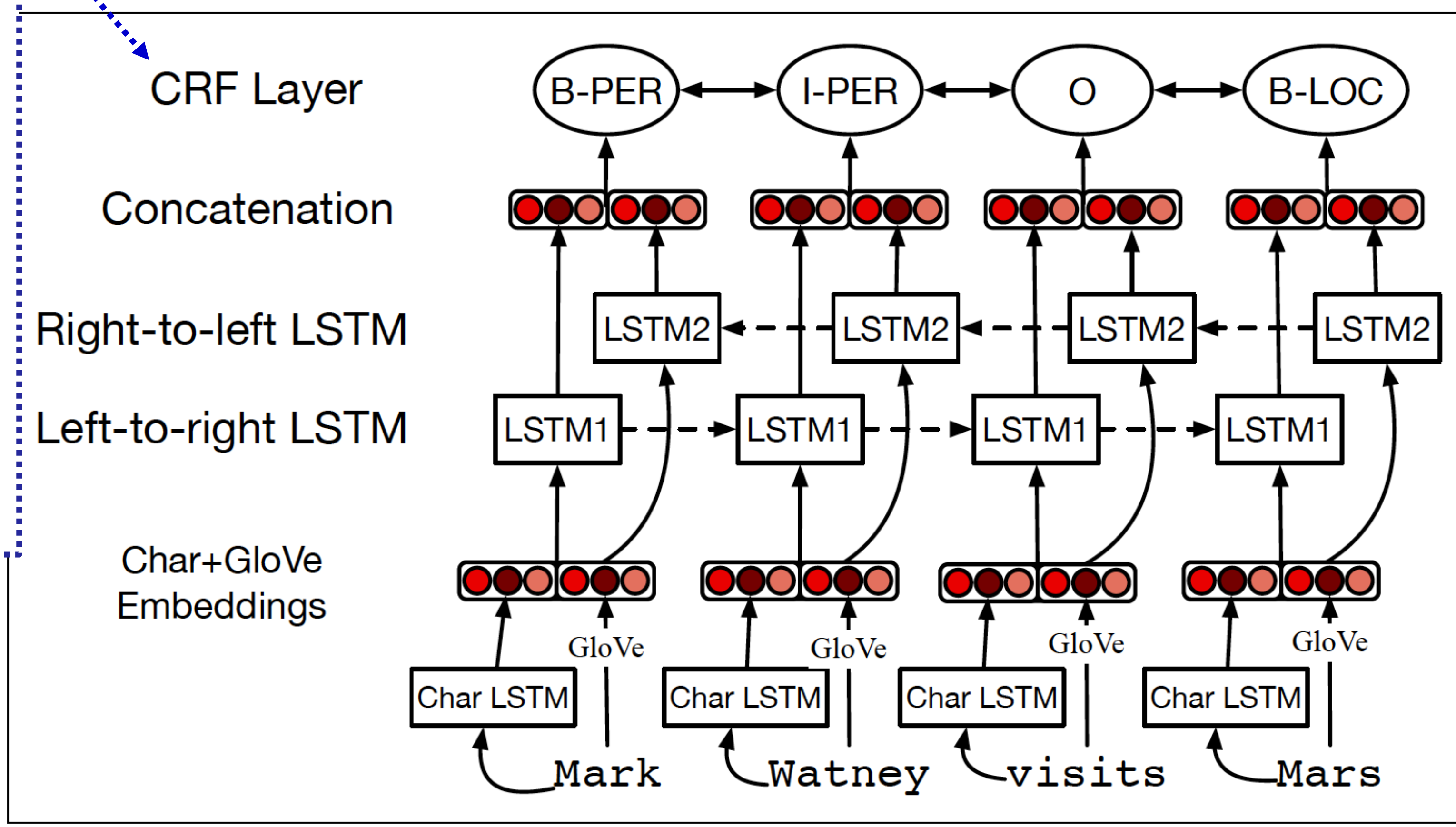


Figure 18.8 Putting it all together: character embeddings and words together in a bi-LSTM sequence model. After Lample et al. (2016).

Figure from Jurafsky and Martin's "Speech and Language Processing", 3rd edition (in preparation). https://web.stanford.edu/~jurafsky/slp3/

Information extraction: next stages

- **Coreference resolution**.
  - "**She** also said that…". "**The company** also announced that…"
  - Including **name matching** (e.g., "General Company Hellas", "GCH", "Mr. George Papandreou", "Papandreou") and possibly **linking entity mentions** to **ontology concepts** (ids).
- **Relation (more generally, event) extraction**:
  - In the simplest case, using **manually crafted rules**.
  - E.g., Acquisition(buyer:C1, bought:C2, share:P) → * Company(id:C1) * Verb(base:buy/acquire/obtain) * Percent(norm:P) * Company(id:C2) *
  - Or using **supervised machine learning**: learn to **predict the relation** (if any) between each **pair of named entities** (that do not exceed a maximum distance). **Classification** problem. One **class** per **relation type** (plus 'none').

Rule-based relation/event extraction

<s> Important news from <company id="GCH"> General Company Hellas </company>, the largest Greek construction company. </s>

<s> <date norm="2/12/2015"> Yesterday </date> **<company id="GCH">** GCH </company> announced that it **<verb base="buy">** bought</verb> **<percent norm="0.42">** 42% </percent> of **<company id="SCL">** Small Company Ltd </company>, a British company that specializes in iron constructions. </s>

Acquisition(buyer:C1, bought:C2, share:P) → *
Company(id:C1) * Verb(base:buy/acquire/obtain) *
Percent(norm:P) * Company(id:C2) *

Acquisition(buyer:GCH, bought:SCL, share:0.42)

Relation extraction via supervised learning

<s> **<company id=“GCH”>** General Company Hellas </company> **<verb base=“buy”>** bought </verb> **<percent norm=“0.42”>** 42% </percent> of **<company id=“SCL”>** Small Company Ltd </company>. </s> <s> **<company id=“LCL”>** Large Company Ltd </company> had also **<verb base=“buy”>** bought </verb> shares of **<company id=“SCL”>** Small </company> <date norm=“Y2007”> last year </date> . </s>

- E.g., for **acquisition** relations:
  - Consider **company-company** and **person-company** pairs (up to a maximum distance).
  - **Classes** (for each pair): **negative**, **positive (or type of relation**, e.g., acquisition, merger).

Relation extraction annotated dataset

Example	Entity Types & Label
Carey will succeed Cathleen P. Black, who held the position for 15 years and will take on a new role as chairwoman of Hearst Magazines, the company said.	PERSON/TITLE Relation: *per:title*
Irene Morgan Kirkaldy, who was born and reared in Baltimore, lived on Long Island and ran a child-care center in Queens with her second husband, Stanley Kirkaldy.	PERSON/CITY Relation: *per:city_of_birth*
Baldwin declined further comment, and said JetBlue chief executive Dave Barger was unavailable.	Types: PERSON/TITLE Relation: *no_relation*

Figure 17.4 Example sentences and labels from the TACRED dataset (Zhang et al., 2017).

Figure from Jurafsky and Martin's "Speech and Language Processing", 3rd edition (in preparation). https://web.stanford.edu/~jurafsky/slp3/

BERT-based relation extraction

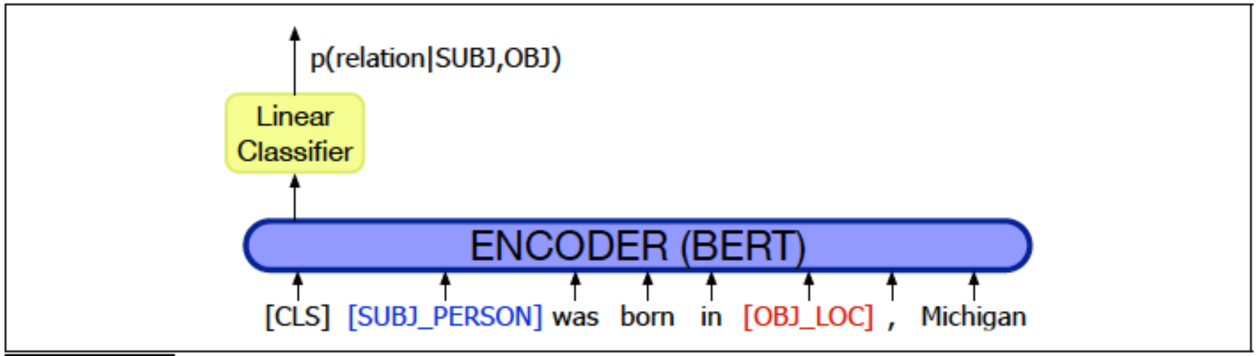


Figure 17.7 Relation extraction as a linear layer on top of an encoder (in this case BERT), with the subject and object entities replaced in the input by their NER tags (Zhang et al. 2017, Joshi et al. 2020).

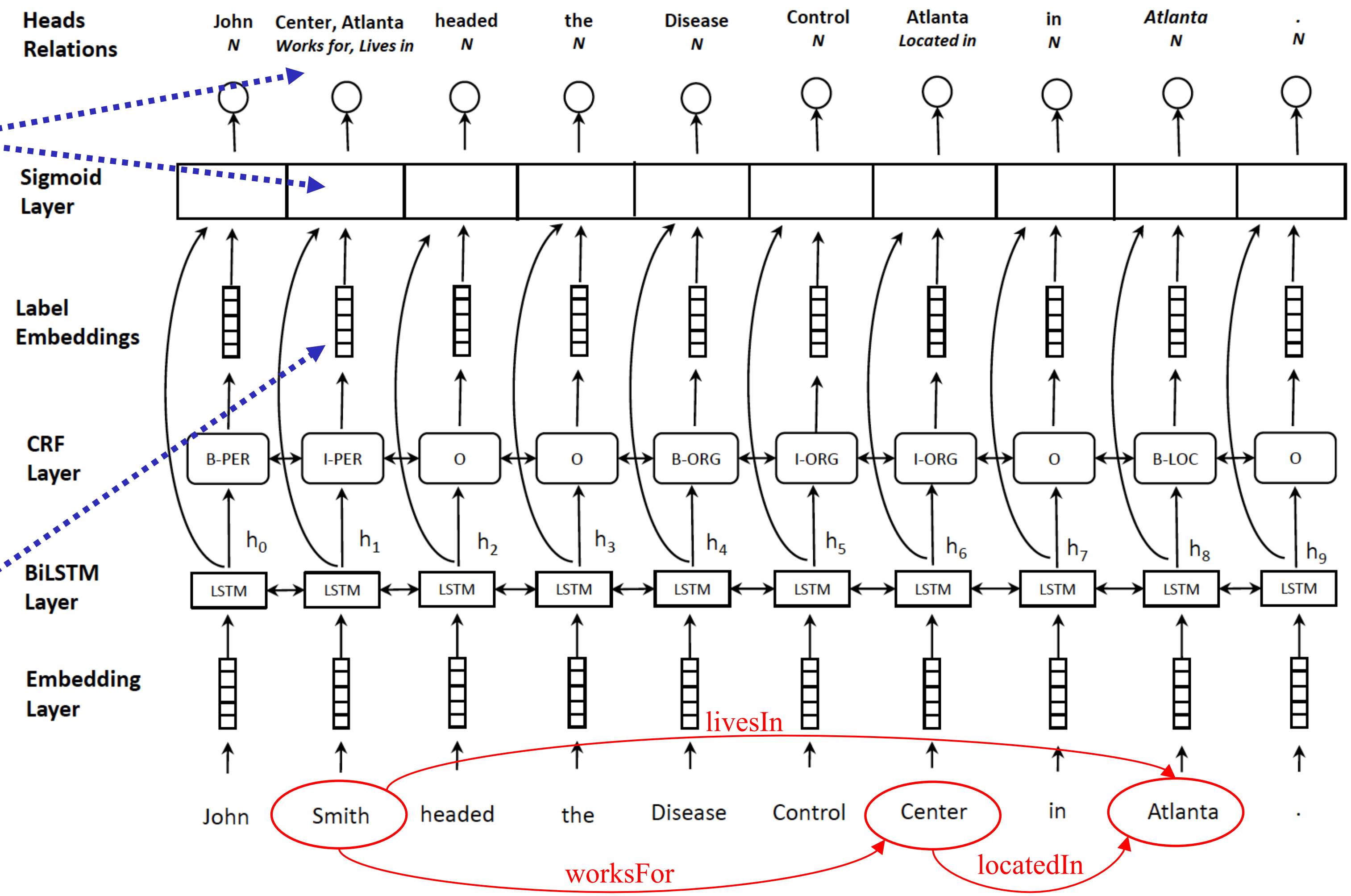
- **Replacing entity names by their types** may help the model generalize with fewer training examples.
- **Candidate entity pairs** can be **limited** to particular **syntactic relations** to consider **fewer pairs**.

○ Or just concatenate the context-aware embeddings of the **first tokens** of the **two entity names** (of each candidate pair) and pass them to an **MLP** to predict their **relation type** (if any).

Figure from Jurafsky and Martin's "Speech and Language Processing", 3rd edition (in preparation). https://web.stanford.edu/~jurafsky/slp3/

Joint NE and relation extraction

A binary **classifier scores the possible heads & relations** of each word (e.g., Smith worksFor Center)



Embedding of NER label (e.g., I-PER)

Figure from G. Bekoulis et al., “Joint entity recognition and relation extraction as a multi-head selection problem”, *Expert Systems with Applications* 114, pp. 34–45, 2018. See also G. Bekoulis et al., “Adversarial training for multi-context joint entity and relation extraction”, EMNLP 2018, https://www.aclweb.org/anthology/D18-1307/

Evaluating relation extraction

- We can use **precision**, **recall**, **F-measure** again.
  - If we have **texts manually annotated** with the **correct relations** (slide 27), count how many **correct relation mentions (in the texts)** were extracted **(true positives)**, how many were not extracted **(false negatives)**, how many wrong relation mentions were extracted **(false positives)** etc.
  - If we have a **database** with **known entity pairs per relation type**, count how many of the **known entity pairs** of **the DB** were extracted **(true positives)** from a **document collection**, how many were not extracted **(false negatives)**, how many unknown entity pairs were extracted **(false positives)** etc.
    - Such a **database** could also be used **during training** ("**distant supervision**", we have **no annotations directly on the texts**).

Extra <u>optional</u> slides.

DCG for simple arithmetic language

open open two plus four close slash open four minus one close close

((2 + 4) / (4 – 1))

```
digit --> [zero].
digit --> [one].
...
digit --> [nine].

expression --> digit.
expression --> [open], expression, [plus], expression, [close].
expression --> [open], expression, [minus], expression, [close].
expression --> [open], expression, [star], expression, [close].
expression --> [open], expression, [slash], expression, [close].
```

➢ phrase(expression, [open, open, two, plus, four, close, slash, open, four, minus, one, close, close]).

Yes.

Semantic parsing for the arithmetic language

```
digit(0) --> [zero].
digit(1) --> [one].
...
digit(9) --> [nine].
```

Inside braces we write additional constraints that need to be satisfied for the rule to be used. Here 'is' assigns the result of X1 + X2 to X. ('=' denotes unification in Prolog.)

```
expression(X) --> digit(X).
expression(X) --> [open], expression(X1) , [plus], expression(X2),
    [close], {X is X1 + X2}.
expression(X) --> [open], expression(X1) , [minus],
    expression(X2) , [close], {X is X1 - X2}.
...
```

- phrase(expression(X) , [open, open, two, plus, four, close, slash, open, four, minus, one, close, close]).

  X = 2.

Syntax of First-Order Predicate Logic

formula → *atomic_formula*
| (*formula connective formula*)
| *quantifier variable formula*
| ¬*forumla*

atomic_formula → *relation_symbol*(*term*, ...) | *term* = *term*

term → *constant* | *variable* |
function_symbol(*term*, ...)

connective → $\wedge$ | $\vee$ | $\Rightarrow$ | $\Leftrightarrow$

quantifier → $\forall$ | $\exists$

constant → A | X_1 | John | Mary | ...

variable → a | x | s | ...

relation_symbol → IsFatherOf | HasColor | IsKing | ...

function_symbol → FatherOf | LeftLeg | ...

Examples of formulae in First-Order Predicate Logic

- *Every dog that barks is afraid of a (possibly different) cat.*

  $\forall x\ ((IsDog(x) \wedge Barks(x)) \Rightarrow$

  $\exists y\ (IsCat(y) \wedge IsAfraidOf(x, y)))$

- *Every cat likes <u>exactly one</u> (possibly different) dog.*

  $\forall y\ (IsCat(y) \Rightarrow$

  $\exists x\ (IsDog(x) \wedge Likes(y, x) \wedge$

  $\forall z\ ((IsDog(z) \wedge Likes(y, z)) \Rightarrow z = x)))$

DCG for semantics of simple sentences

```
s(Predicate) --> np(X1), vp(X, Predicate), {X1 = X}.

vp(X, Predicate) --> v(Y, X, Predicate), np(Y1), {Y = Y1}.

np(Sem) --> pn(Sem).

pn(john) --> [john].
pn(mary) --> [mary].

v(Y, X, loves(X,Y)) --> [loves].
```

We require the meaning representation Y1 of the np (e.g., john) to be unified with the second argument of the logical predicate of the verb (the Y of loves(X,Y)). This causes the representation of the np to be copied into the predicate of the verb (e.g., loves(X,Y) becomes loves(X, john)).

The three arguments of v stand for the $\lambda y\ \lambda x$ Loves(x, y) of the previous slide.

DCG rules for nouns and quantifiers

```
n( X1, customer(X1) ) --> [customer].
n( X4, flight(X4) ) --> [flight].
det( X2, P1, forall(X2, P1) ) --> [every].
det( X5, P2, forsome(X5, P2) ) --> [a].
```

The NewX corresponds to the new x_3 variable of the original rules.

```
np(SemNP) --> det(XDet, P, SemDet), n(XN, SemN),
   {XDet = NewX, XN = NewX,  P = SemN, SemNP = SemDet}.
```

or more briefly:

```
np(SemDet) --> det(X, SemN, SemDet), n(X, SemN).
```

Representing events

I ate.

$\text{Eating}_1(\text{Speaker})$

I ate a souvlaki.

$\exists x\ (\text{IsSouvlaki}(x) \wedge \text{Eating}_2(\text{Speaker}, x))$

I ate a souvlaki at my office.

$\exists x\ (\text{IsSouvlaki}(x) \wedge \text{Eating}_3(\text{Speaker}, x, \text{OfficeOf}(\text{Speaker})))$

I ate a souvlaki at my office yesterday.

$\exists x\ (\text{IsSouvlaki}(x) \wedge$

$\text{Eating}_4(\text{Speaker}, x, \text{OfficeOf}(\text{Speaker}), \text{Yesterday}))$

- If we use separate $\text{Eating}_1(\ldots)$, $\text{Eating}_2(\ldots)$, $\text{Eating}_3(\ldots)$ etc. predicates, we need meaning postulates stating, for example, that when $\text{Eating}_3(\ldots)$ happens then $\text{Eating}_2(\ldots)$ also happens.

Representing events – cont.

I ate.

$\exists x \, \exists y \, \exists z \; Eating(Speaker, x, y, z)$

I ate a souvlaki.

$\exists x \, \exists y \, \exists z \; (IsSouvlaki(x) \land Eating(Speaker, x, y, z))$

I ate a souvlaki at my office.

$\exists x \, \exists z \; (IsSouvlaki(x) \land Eating(Speaker, x, OfficeOf(Speaker), z))$

I ate a souvlaki at my office yesterday.

$\exists x \; (IsSouvlaki(x) \land$

$Eating(Speaker, x, OfficeOf(Speaker), Yesterday))$

- How many arguments does Eating(…) need?
- What about “I <u>quickly</u> ate a souvlaki at my office yesterday”?
- What about “I quickly ate a souvlaki at my office yesterday <u>before leaving</u>”?

Event variables (Davidsonian semantics)

I ate.

$\exists e\ (\text{Eating}(e) \wedge \text{Eater}(e, \text{Speaker}))$

I ate a souvlaki.

$\exists e\ \exists x\ (\text{Eating}(e) \wedge \text{IsSouvlaki}(x) \wedge \text{Eater}(e, \text{Speaker}) \wedge$
$\text{Eaten}(e, x))$

I ate a souvlaki at my office.

$\exists e\ \exists x\ (\text{Eating}(e) \wedge \text{IsSouvlaki}(x) \wedge \text{Eater}(e, \text{Speaker}) \wedge$
$\text{Eaten}(e, x) \wedge \text{Location}(e, \text{OfficeOf}(\text{Speaker})))$

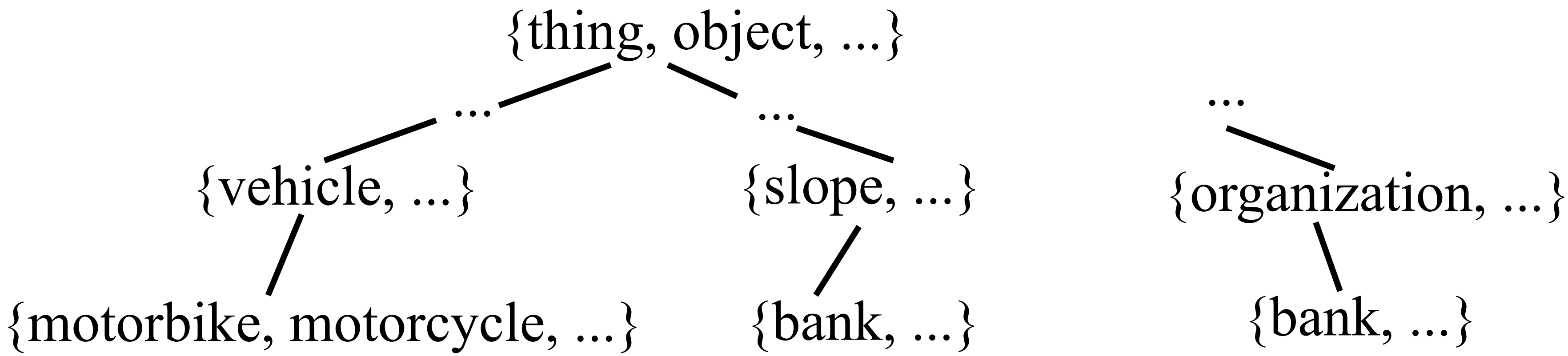
I quickly ate a souvlaki at my office before leaving.

$\exists e_1\ \exists e_2\ \exists x\ \exists i_1\ \exists i_2\ (\text{Eating}(e_1) \wedge \text{IsSouvlaki}(x) \wedge \text{Eaten}(e_1, x) \wedge$
$\text{Eater}(e_1, \text{Speaker}) \wedge \text{Location}(e_1, \text{OfficeOf}(\text{Speaker})) \wedge$
$\text{Speed}(e_1, \text{Fast}) \wedge \text{Leaving}(e_2) \wedge \text{Leaver}(e_2, \text{Speaker}) \wedge$
$\text{IntervalOf}(e_1, i_1) \wedge \text{IntervalOf}(e_2, i_2) \wedge \text{Before}(\text{End}(i_1), \text{Start}(i_2)))$

Lexical semantic relations

- **Homonyms**: same spelling, different meanings.
  - E.g., financial “bank” and “bank” of a river.
  - E.g., “letter” of the alphabet and “letter” that you post.
- **Synonyms**: different words, but can be used with (approximately) the same meaning.
  - E.g., “motorbike” and “motorcycle”, “lift” and “elevator”.
- **Hypernym – hyponym**: broader – narrower meaning.
  - E.g., “vehicle” – “car”, “organization” – “company”.
- **Antonyms**: opposite meanings.
  - E.g., “tall” – “short”, “large” – “small”.
- See J&M for more kinds of lexical semantic relatations.

WordNet (http://wordnet.princeton.edu/)



- Every **sense** is represented by a **set of synonyms** (synset) that can have that sense.
- **Hypernym – hyponym** hierarch per part of speech (nouns, adjectives, verbs, adverbs).
- Many **other relations** also included (see J&M).
  - E.g., **meronyms** (a “wheel” is part of a “bicycle”).
- Initially for **English**, now for **many languages**.

WordNet Search - 3.0 - Mozilla Firefox

Αρχείο Επεξεργασία Προβολή Ιστορικό Σελιδοδείκτες Εργαλεία Βοήθεια

http://wordnetweb.princeton.edu/ 12 ghosts clip

WordNet Search - 3.0 | eClass του Οικονομικού Πανεπιστη...

WordNet Search - 3.0 - WordNet home page - Glossary - Help

Word to search for: bank Search WordNet

Display Options: (Select option to change) Change

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Noun

- S: (n) **bank** (sloping land (especially the slope beside a body of water)) *"they pulled the canoe up on the bank"; "he sat on the bank of the river and watched the currents"*
    - *direct hyponym* / *full hyponym*
        - S: (n) riverbank, riverside (the bank of a river)
        - S: (n) waterside (land bordering a body of water)
    - *direct hypernym* / *inherited hypernym* / *sister term*
        - S: (n) slope, incline, side (an elevated geological formation) *"he climbed the steep slope"; "the house was built on the side of a mountain"*
    - *derivationally related form*
- S: (n) depository financial institution, **bank**, banking concern, banking company (a financial institution that accepts deposits and channels the money into lending activities) *"he cashed a check at the bank"; "that bank holds the mortgage on my home"*
    - *direct hyponym* / *full hyponym*
    - *member holonym*
    - ***direct hypernym*** / *inherited hypernym* / *sister term*
        - S: (n) financial institution, financial organization, financial organisation (an institution (public or private) that collects funds (from the public or other institutions) and invests them in financial assets)
    - *derivationally related form*
- S: (n) **bank** (a long ridge or pile) *"a huge bank of earth"*
- S: (n) **bank** (an arrangement of similar objects in a row or in tiers) *"he operated a bank of*

Ολοκληρώθηκε

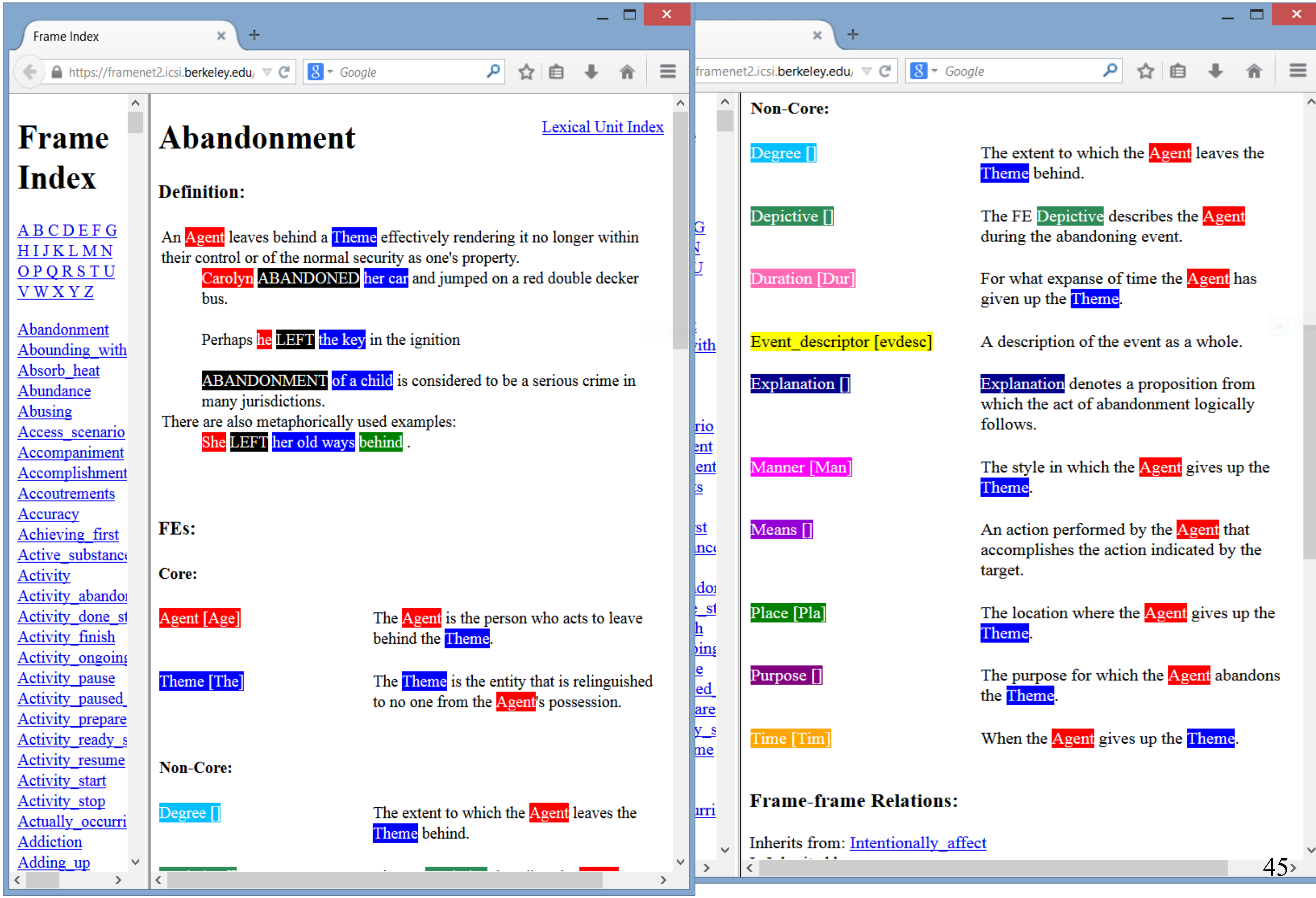
Thematic roles

I quickly ate a souvlaki at my office before leaving.

$\exists e_1 \exists e_2 \exists x \exists i_1 \exists i_2$ (Eating(e_1) $\wedge$ IsSouvlaki(x) $\wedge$ Eaten(e_1, x) $\wedge$ Eater(e_1, Speaker) $\wedge$ Location(e_1, OfficeOf(Speaker)) $\wedge$ Speed(e_1, Fast) $\wedge$ Leaving(e_2) $\wedge$ Leaver(e_2, Speaker) $\wedge$ IntervalOf(e_1, i_1) $\wedge$ IntervalOf(e_2, i_2) $\wedge$ Before(End(i_1), Start(i_2)))

- Depending on the **event type**, different **roles** are available.
  - **Every event** has an **IntervalOf** role.
  - An **Eating** event may also have **Eaten** and **Eater** roles.
  - A **Leaving** event may also have a **Leaver** role.
- We need a **taxonomy of event types**, which will define the possible **event types** and their **roles**.

FrameNet


Frame Index
https://framenet2.icsi.berkeley.edu/
Google
Frame Index
A B C D E F G
H I J K L M N
O P Q R S T U
V W X Y Z
Abandonment
Abounding_with
Absorb_heat
Abundance
Abusing
Access_scenario
Accompaniment
Accomplishment
Accoutrements
Accuracy
Achieving_first
Activity
Adding_up
Addiction
Abandonment
Lexical Unit Index
Definition:
An Agent leaves behind a Theme effectively rendering it no longer within their control or of the normal security as one's property.
Carolyn ABANDONED her car and jumped on a red double decker bus.
Perhaps he LEFT the key in the ignition
ABANDONMENT of a child is considered to be a serious crime in many jurisdictions.
There are also metaphorically used examples:
She LEFT her old ways behind .
FEs:
Core:
Agent [Age]
The Agent is the person who acts to leave behind the Theme.
Theme [The]
The Theme is the entity that is relinguished to no one from the Agent's possession.
Non-Core:
Degree []
The extent to which the Agent leaves the Theme behind.
framenet2.icsi.berkeley.edu/
Google
Non-Core:
Degree []
The extent to which the Agent leaves the Theme behind.
Depictive []
The FE Depictive describes the Agent during the abandoning event.
Duration [Dur]
For what expanse of time the Agent has given up the Theme.
Event_descriptor [evdesc]
A description of the event as a whole.
Explanation []
Explanation denotes a proposition from which the act of abandonment logically follows.
Manner [Man]
The style in which the Agent gives up the Theme.
Means []
An action performed by the Agent that accomplishes the action indicated by the target.
Place [Pla]
The location where the Agent gives up the Theme.
Purpose []
The purpose for which the Agent abandons the Theme.
Time [Tim]
When the Agent gives up the Theme.
Frame-frame Relations:
Inherits from: Intentionally_affect

FrameNet (https://framenet.icsi.berkeley.edu/)

- Particular trigger words **activate frames**, which define **thematic roles** (frame elements).
  - "... an *increase* [$_{\text{INIT\_VALUE}}$ from 20%] [$_{\text{FINAL\_VALUE}}$ to 27%]…"
  - "... *fell* [$_{\text{FINAL\_VALUE}}$ to 27%]…"
  - Here both trigger words activate the same frame.
- **FrameNet** provides a rich collection of **frames**, **trigger words**, **roles**, **inheritance** from more general to more specific frames etc.
  - Useful, for example, in **information extraction**.
  - **Semantic role labeling**: methods that "fill" the roles of active frames in each sentence, usually by employing machine learning (see J&M).

Selectional restrictions

I saw the [doctor [with the white coat]].

$\exists e_1\ \exists x_1\ \exists x_2\ \exists i_1$ (Seeing(e_1) ∧ IsDoctor(e_1, x_1) ∧ IsCoat(e_1, x_2) ∧ IsWhite(e_1, x_2) ∧ Agent(e_1, Speaker) ∧ Seen(e_1, x_1) ∧ **Wearing**(e_1, x_1, x_2) ∧ IntervalOf(e_1, i_1) ∧ Before(End(i_1), Now)))

? I saw [the doctor] [with the white coat].

$\exists e_1\ \exists x_1\ \exists x_2\ \exists i_1$ (Seeing(e_1) ∧ IsDoctor(e_1, x_1) ∧ IsCoat(e_1, x_2) ∧ IsWhite(e_1, x_2) ∧ Agent(e_1, Speaker) ∧ Seen(e_1, x_1) ∧ **ObservationInstrument**(e_1, x_2) ∧ IntervalOf(e_1, i_1) ∧ Before(End(i_1), Now)))

The 2^{nd} reading can be ruled out via **logical inference**, if we have a sufficiently rich **knowledge base**. Difficult...

$\forall e\ \forall x$ (ObservationInstrument(e, x) ⇔ (IsEyeGlasses(e, x) ∨ IsBinoculars(e, x) ∨ …))

Selectional restrictions – cont.

- Alternatively, simple **selectional restrictions** can be included in the **lexicon** and **grammar**.

  n(sense: s_{144}) → [food]. (Assuming that $\mathbf{s_{144}}$ is the **synset** for the concept of **food**.)

  n(sense: s_{138}) → [salad]. (Assuming s_{138} is a **hyponym** of s_{144}.)

  v(objSense: s_{144}) → [eat]. (The argument of v shows that the verb **requires** an **object** with sense $\mathbf{s_{144}}$ **or hyponym**.)

  vp → v(objSense: S_1), n(sense: S_2), {hypernymOf(S_1, S_2)}.

- **Similarly** (exercise…) we can **rule out**:

  I saw [the doctor] [with the white coat].

- **No parse trees** produced for **readings violating selectional restrictions**.
  - But **WordNet** may not provide exactly the **concepts we need** for some selectional restrictions.
  - Also problems with **negations** (e.g., "Do not drink gasoline.") or **metaphors** ("Time flies when you're having fun.").

Temporal expressions

- **Some kinds** of temporal expressions:
  - **Temporal locations**: e.g., "on January 18th, 2015", "in the 5th century BC".
  - **Relative temporal locations**: e.g., "yesterday", "two weeks earlier", "the following two years". Possibly related to the **publication date** or other **prominent time** (e.g., of an event).
  - **Durations, frequencies**: "the battery lasts <u>for 8 hours</u>", "payment is due <u>in three days/every January</u>".
- **Many temporal mechanisms** in languages:
  - **Tense/aspect** of verbs, temporal **adjectives/adverbs**, **clauses**…
  - Check my thesis and book if interested…

Recognizing temporal expressions

- Temporal expressions often contain **trigger words**.
  - E.g., “January”, “week”, “year”, “Sunday”.
  - But there are exceptions too (e.g., “Never on a Sunday”).
- We can use the **same supervised learning** methods as in **named entity recognition** (sequence labeling).
- For simple expressions (e.g., dates, durations) manually crafted **regular expressions** or **grammars** may suffice.
  - Reasonably easy to write for simple temporal expressions.
  - Grammars can be extended to also **normalize** the temporal expressions (e.g., “on January 18, 2015” → “18/1/2015”).

TimeML (ISO 8601)

Athens, <TIMEX3 id=“t1” type=“DATE” value=“2015-07-02” functionInDocument=“CREATION_TIME”> July 2, 2015</TIMEX3> : The reduced ticket prices that OA announced <TIMEX3 id=“t2” type=“DATE” value=“2015-W26” anchorTimeID=“t1”> last week </TIMEX3> forced …

- **TimeML**: **Annotation standard** (XML-based) for **temporal expressions**, their **normalized values**, and **events**.
  - **TimeBank**: corpus annotated according to TimeML.
  - See J&M for further details.

(almost) Unsupervised relation extraction

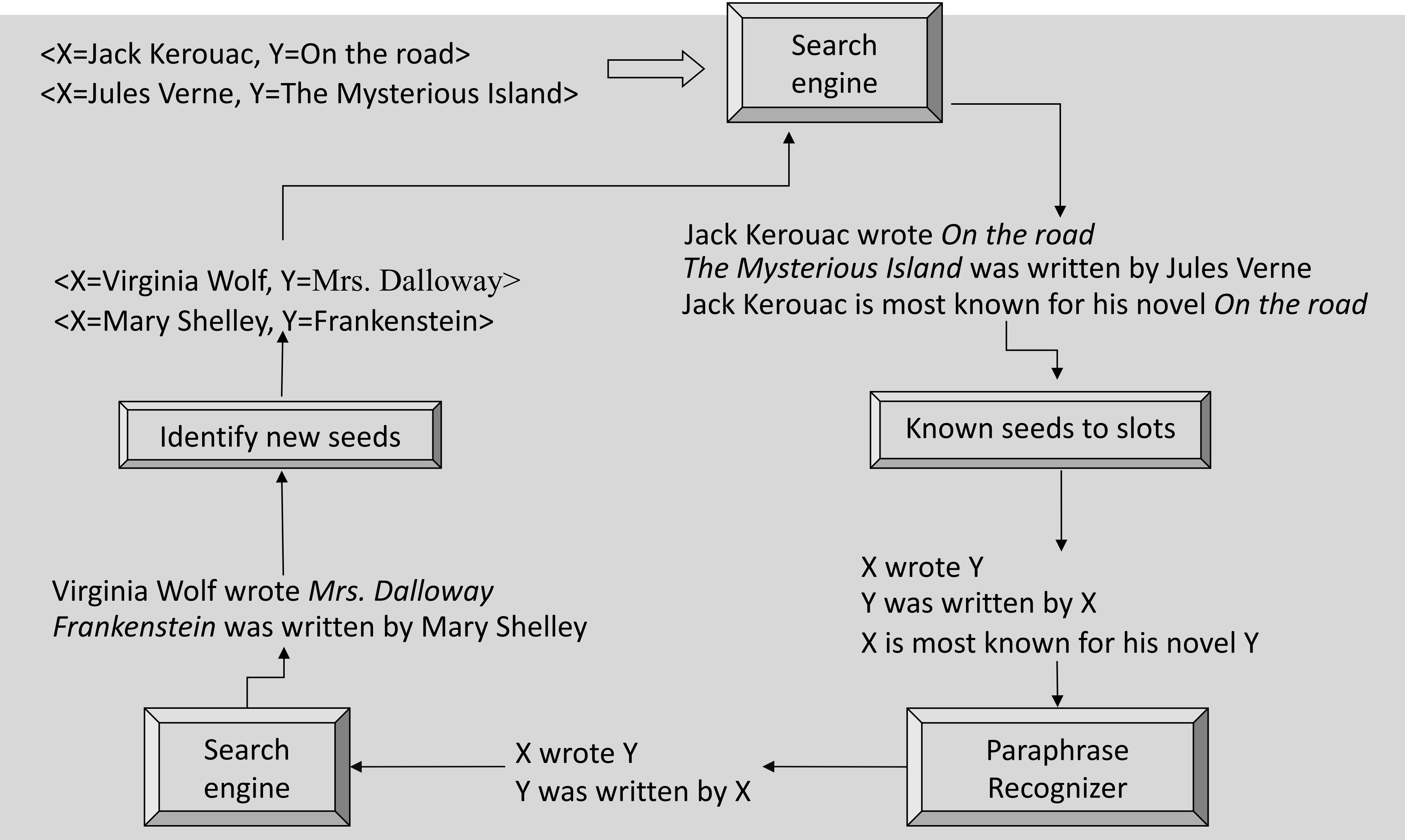


Image by Prodromos Malakasiotis.

(almost) Unsupervised relation extraction

- **For each relation type**, perform **bootstrapping**:
  - Start with **seed entity pairs** for which the relation holds.
  - **Retrieve sentences** that **contain** the **seed entity pairs**.
  - **Construct patterns** from the **retrieved sentences**, e.g., regular expression patterns or patterns operating on parse trees.
  - Retrieve **sentences** that **match** the **patters**.
  - Extract **new entity pairs** from the **retrieved sentences** etc.
- **Semantic drift** problem:
  - Soon we start obtaining **entity pairs** and **patterns** that are very **general** or **irrelevant** (e.g., “X is known for Y”).
  - A “**human in the loop**” is needed to **filter** the **entity pairs** and **patterns**, or **measures** to automatically **score them**, or **classifiers** to filter them (possibly trained on human decisions).

Scoring new patterns

- A good **candidate pattern** p will have **high precision**, but will also **fire frequently** in a corpus (high recall).
  - Patterns with **low precision** are **unreliable**.
  - Patterns that **rarely fire** will also **not help** much.

$$\mathrm{Conf}(p) = \frac{\mathrm{hits}(p)}{\mathrm{finds}(p)} \cdot \log \mathrm{finds}(p)$$

How many **known correct entity pairs** does p **extract** from the corpus?

How many **entity pairs** does p **extract** from the corpus?

Scoring new entity pairs

- Assume an **entity pair $\boldsymbol{r}$** is **wrong** if and only if **all the patterns $\boldsymbol{p} \in \boldsymbol{P}$** that **extract (support) $\boldsymbol{r}$** are **mistaken**.
  - Assume that the **mistakes of the patterns** are **independent**.
  - And that the **confidence Conf($\boldsymbol{p}$)** of each **pattern** p is (almost) a **probability**.

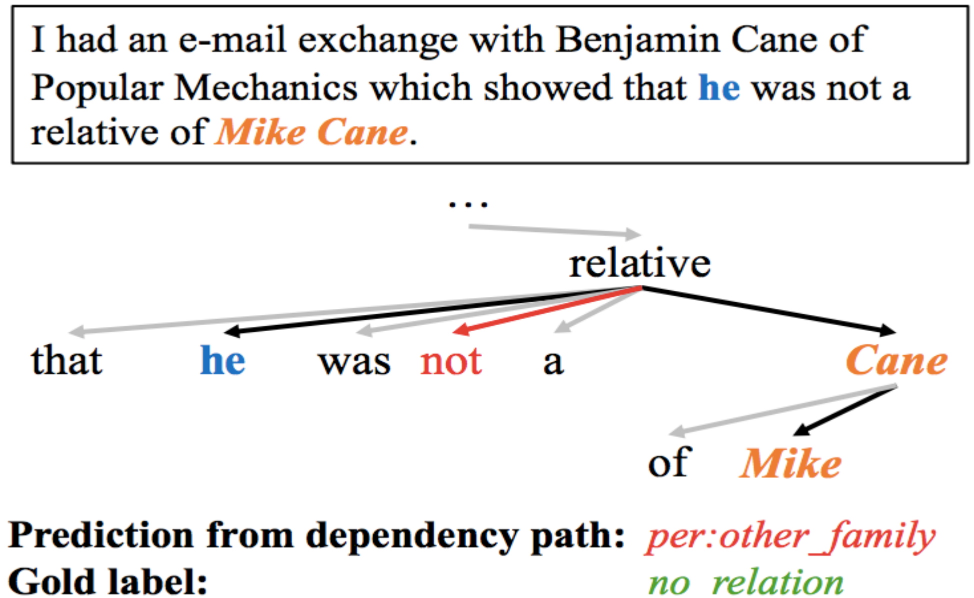
"**Probability**" that **entity pair $\boldsymbol{r}$** is **correctly extracted** (not all of the patterns that extract it are mistaken).

"**Probability**" that **all the patterns** that **extract $\boldsymbol{r}$** are **mistaken**.

$$\mathrm{Conf}(r) = 1 - \overline{\prod_{p:\, p \text{ extracts } r} \underline{(1 - \mathrm{Conf}(p))}}$$

"**Probability**" that a **pattern $\boldsymbol{p}$** that **extracts $\boldsymbol{r}$** is **mistaken**.

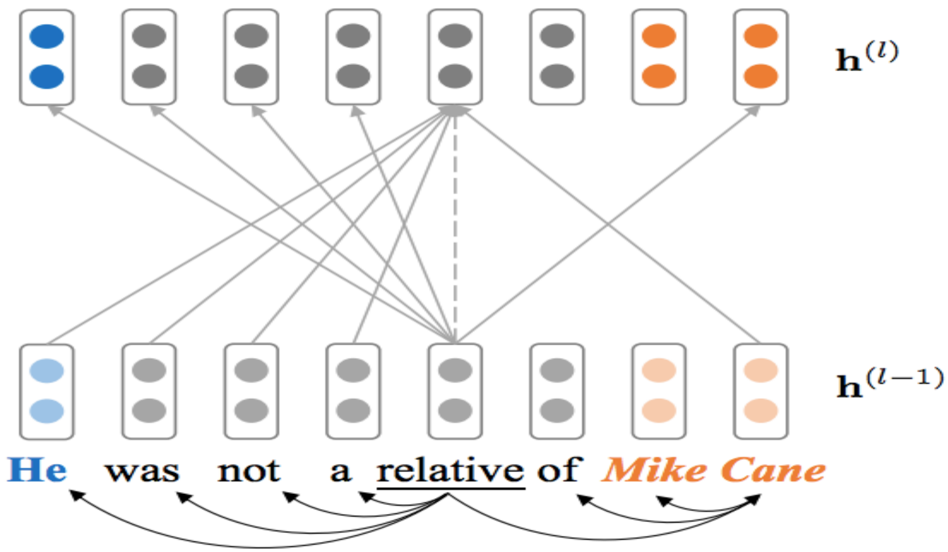
Relation extraction with graph CNNs



- We are **given** the **dependency tree** and the spans (positions in text and tree) of two **named entities** (S, O).
  - Here "he" (referring expression really) and "Mike Cane".
- We need to **predict** the **type** of their **relation** (if any).

Figure from Y. Zhang, P. Qi, C.D. Manning, "Graph Convolution over Pruned Dependency Trees Improves Relation Extraction", EMNLP 2018. http://aclweb.org/anthology/D18-1244

Relation extraction with graph CNNs (cont.)



Matrix (convolution kernel) and **bias** term of **level *l***. We could use a different kernel per dependency type (label).

$$h_i^{(l)} = \sigma\Big(\sum_{j=1}^{n} A_{ij} W^{(l)} h_j^{(l-1)} + b^{(l)}\Big)$$

$A_{i,j} = 1$ if word $\boldsymbol{w_i}$ is connected to word $\boldsymbol{w_j}$ or $i = j$, otherwise 0.

- We build **representations** of the **words** at **levels** 1, 2, ….
  - At **level one**, each **word** $\boldsymbol{w_i}$ is represented by its **embedding**.
  - At **level *l***, the representation $\boldsymbol{h_i^{(l)}}$ of each **word** $\boldsymbol{w_i}$ is a **combination** of the **lower-level representations** $\boldsymbol{h_j^{(l-1)}}$ of the words $\boldsymbol{w_j}$ is **connected** to in the **dependency tree** (incl. itself).

Figure from Y. Zhang, P. Qi, C.D. Manning, "Graph Convolution over Pruned Dependency Trees Improves Relation Extraction", EMNLP 2018. http://aclweb.org/anthology/D18-1244

Relation extraction with graph CNNs (cont.)

Max-pooling of **top-level representations** of the **words** in **entity S**, all the **sentence**, **entity O**.

$$h_{\text{final}} = \text{FFNN}\left([h_{\text{sent}}; h_s; h_o]\right)$$

Fed to a linear layer with softmax to predict the relation type (if any) between the two entities (S, O).

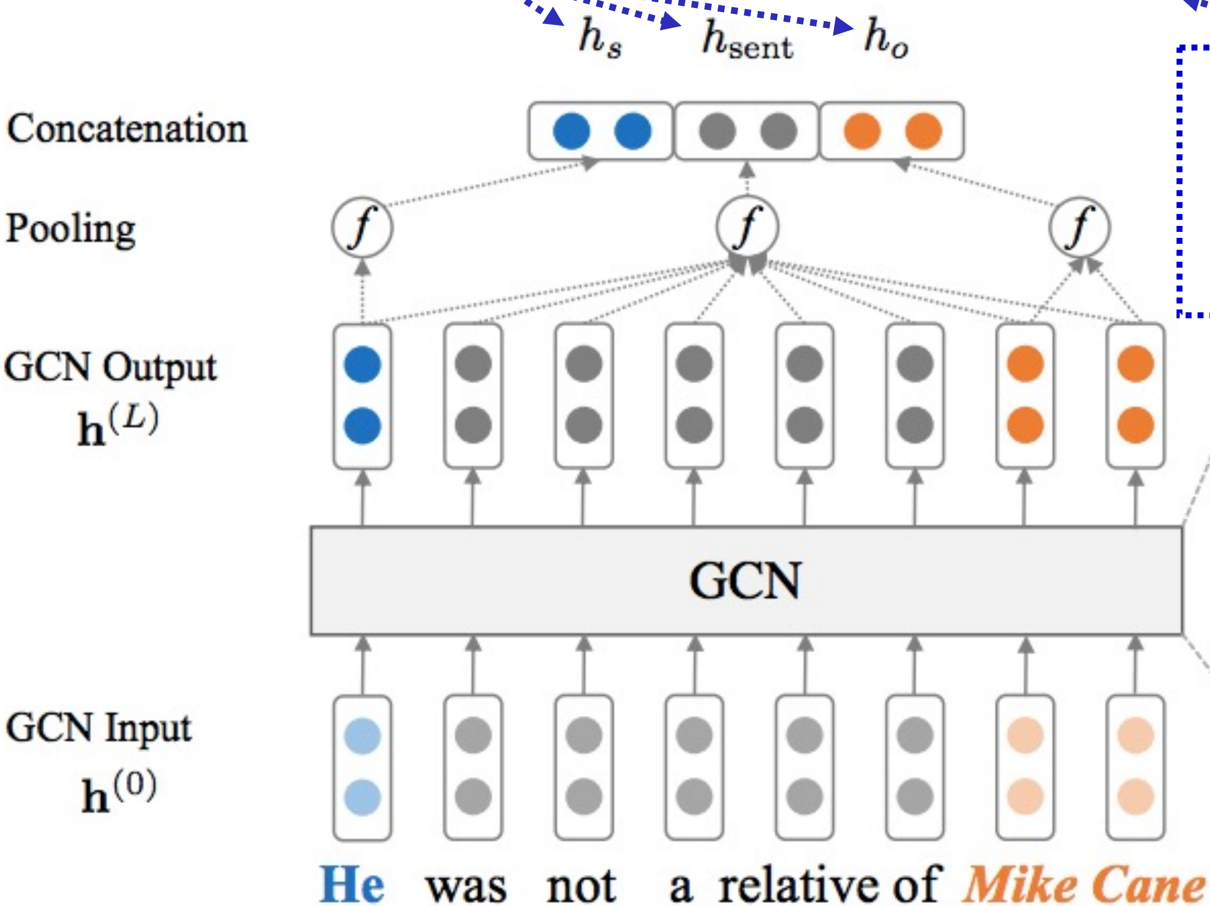
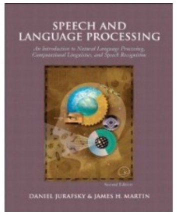


Figure from Y. Zhang, P. Qi, C.D. Manning, "Graph Convolution over Pruned Dependency Trees Improves Relation Extraction", EMNLP 2018. http://aclweb.org/anthology/D18-1244

Recommended reading

- J&M (2nd ed.): chapters 17, 18, 19, 20, 21, 22.
    - Material not covered by the slides is optional.
    - Check also the 3rd edition (in preparation): http://web.stanford.edu/~jurafsky/slp3/



- J. Eisenstein, "Natural Language Processing", MIT Press, 2019: chapters: 12, 13, 17.
    - Free draft available at https://github.com/jacobeisenstein/gt-nlp-class/blob/master/notes/eisenstein-nlp-notes.pdf
    - Material not covered by the slides is optional.