Sentence Semantics

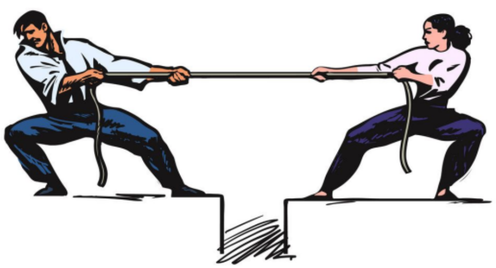
SLP3 Ch 15, 16, 19; INLP Ch 12, 13

Semantics

- Semantics studies meaning, connecting language to the real world
  - Lexical semantics: the meanings of words (last chapter)
  - Sentence semantics (this chapter)
- Implicit vs. explicit meaning representation

Implicit: vector semantics (e.g., word embedding)

Explicit: formal, symbolic representation



Vector Representation of Sentences

- Many options of modeling and learning
- Models
  - Pooling of word embeddings
  - The last hidden vector of an RNN
  - Concatenation of the last hidden vectors in two directions of a bi-RNN
  - Representation of [CLS] in BERT
  - Recursive neural networks based on parse trees
  - …
- Learning
  - Sentence-level tasks: NSP, NLI, …

Vector Representation of Sentences

- Pros:
  - Seamless integration with downstream neural models
  - Impressive performance on many NLP tasks
- Cons:
  - Blackbox: not interpretable

Symbolic Representation of Sentences

- Pros:
  - Interpretable
  - Seamless integration with symbolic knowledge bases and inference engines
- Cons:
  - Many forms of representations, unclear which one is “best”
  - Difficult to build an accurate semantic parser

Formal Meaning Representation

Meaning Representations

- **Unambiguity:** one representation should have exactly one meaning
- **Canonical form:** one meaning should have exactly one representation
- **Verifiability:** ability to ground with knowledge bases
- **Inference ability:** should be able to draw conclusions
- **Expressiveness:** should be able to handle a wide variety of subject matter

Meaning Representations

- Special-purpose representations
  - Database query
  - Robot control commands
  - …
- General-purpose representations
  - Formal logic
  - Semantic graphs

Database queries

- To facilitate data exploration and analysis, you might want to parse natural language into database queries (SQL)

which country had the highest carbon emissions last year

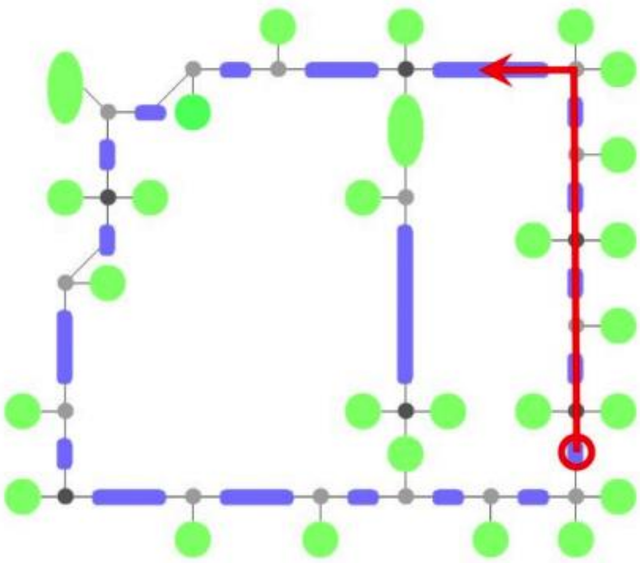
```
SELECT    country.name
FROM      country, co2_emissions
WHERE     country.id = co2_emissions.country_id
AND       co2_emissions.year = 2014
ORDER BY co2_emissions.volume DESC
LIMIT     1;
```

Robot control

- For a robot control application, you might want a custom-designed procedural language:

Go to the third junction and take a left.

```
(do-sequentially
  (do-n-times 3
    (do-sequentially
      (move-to forward-loc)
      (do-until
        (junction current-loc)
        (move-to forward-loc))))
  (turn-left))
```



Intents and arguments

- For smartphone voice commands, you might want relatively simple meaning representations, with intents and arguments:

directions to SF by train

```
(TravelQuery
  (Destination /m/0d6lp)
  (Mode TRANSIT))
```

angelina jolie net worth

```
(FactoidQuery
  (Entity /m/0f4vbz)
  (Attribute /person/net_worth))
```

text my wife on my way

```
(SendMessage
  (Recipient 0x31cbf492)
  (MessageType SMS)
  (Subject "on my way"))
```

play sunny by boney m

```
(PlayMedia
  (MediaType MUSIC)
  (SongTitle "sunny")
  (MusicArtist /m/017mh))
```

First-Order Logic (FOL)

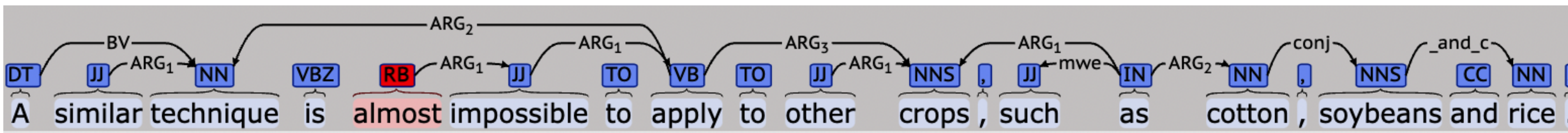
- **Term:** a constant or a variable
- **Formula:** defined recursively
  - If R is an n-ary relation and $t_1, \ldots, t_n$ are terms, then $R(t_1, \ldots, t_n)$ is a formula.
  - If ϕ is a formula, then its negation, $\neg\phi$, is a formula.
  - If ϕ and ψ are formulas, then binary logical connectives can be used to create formulas:
    - $\phi \wedge \psi$, $\phi \vee \psi$, $\phi \Rightarrow \psi$, …
  - If ϕ is a formula and v is a variable, then quantifiers can be used to create formulas:
    - Universal quantifier: $\forall v, \phi$
    - Existential quantifier: $\exists v, \phi$

Translating Between FOL and Natural Language

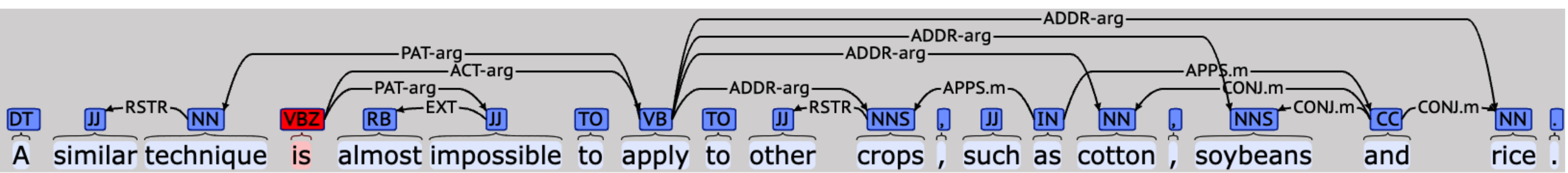
- Alice is not tall
  - $\neg Tall(a)$
- Some people like Broccoli
  - $\exists x, Human(x) \land Likes(x, br)$
- If a person likes Thai, then he isn't a friend with Donald
  - $\forall x, Human(x) \land Likes(x, th) \Rightarrow \neg Friends(x, d)$
- $\forall x, Restaurant(x) \Rightarrow \big(Longwait(x) \lor \neg Likes(a, x)\big)$
  - Every restaurant has a long wait or is disliked by Adrian
- $\forall x, \exists y, \neg Likes(x, y)$
  - Everybody has something he doesn't like
- $\exists y, \forall x, \neg Likes(x, y)$
  - There exists something that nobody likes

Semantic Graphs

- Flavor 0
  - Node: word
  - Edge: relation
  - Ex: DELPH-IN Minimal Recursion Semantics (DM)

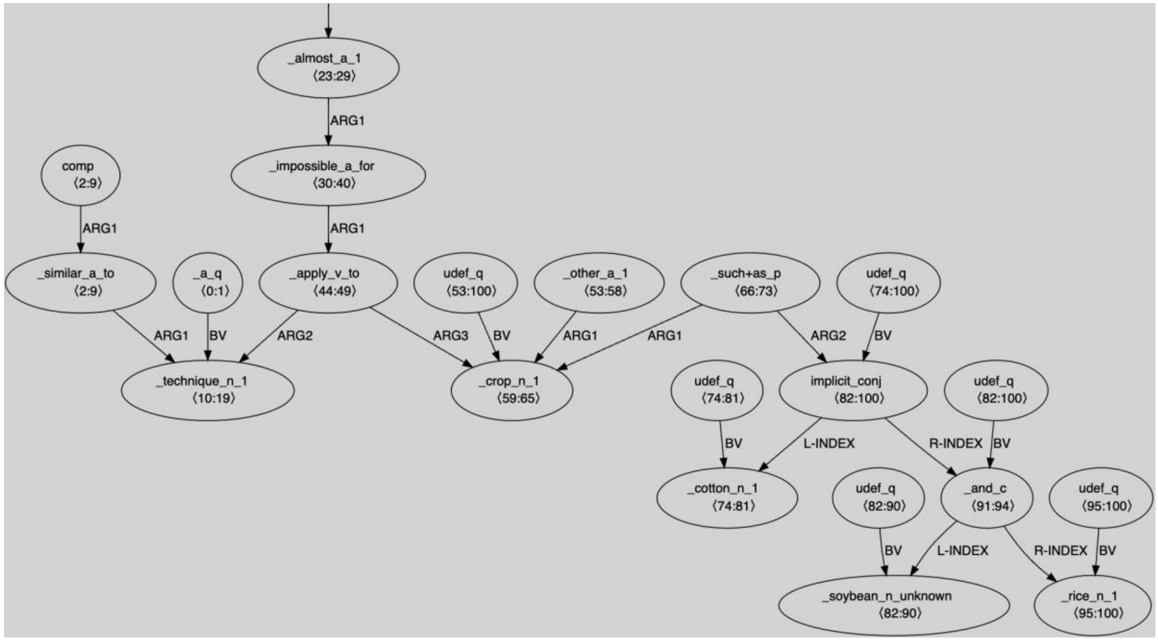


  - Ex: Prague Semantic Dependencies (PSD)



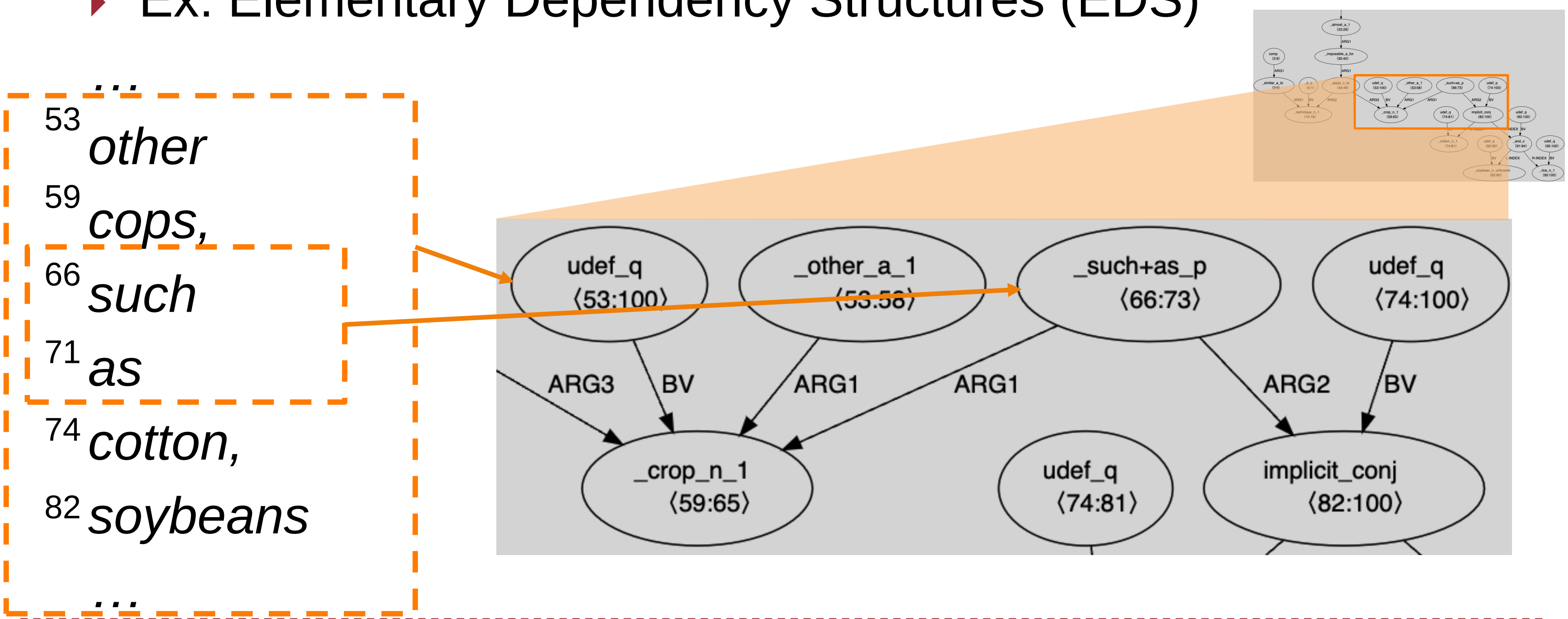
Semantic Graphs

- Flavor 1
  - Node: arbitrary part of the sentence (sub-word, multiple words, no word)
  - Edge: relation
  - Ex: Elementary Dependency Structures (EDS)



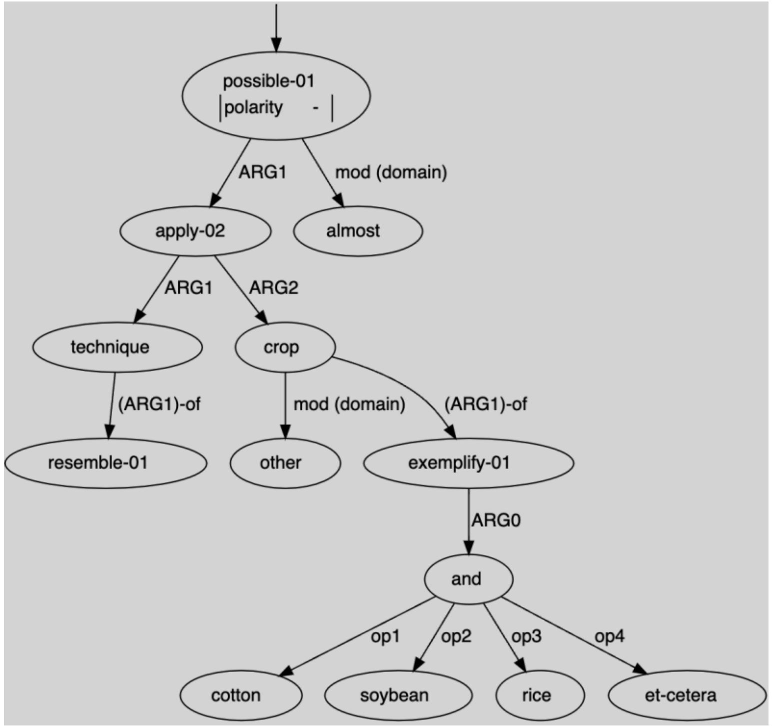
Semantic Graphs

- Flavor 1
  - Node: arbitrary part of the sentence (sub-word, multiple words, no word)
  - Edge: relation
  - Ex: Elementary Dependency Structures (EDS)



Semantic Graphs

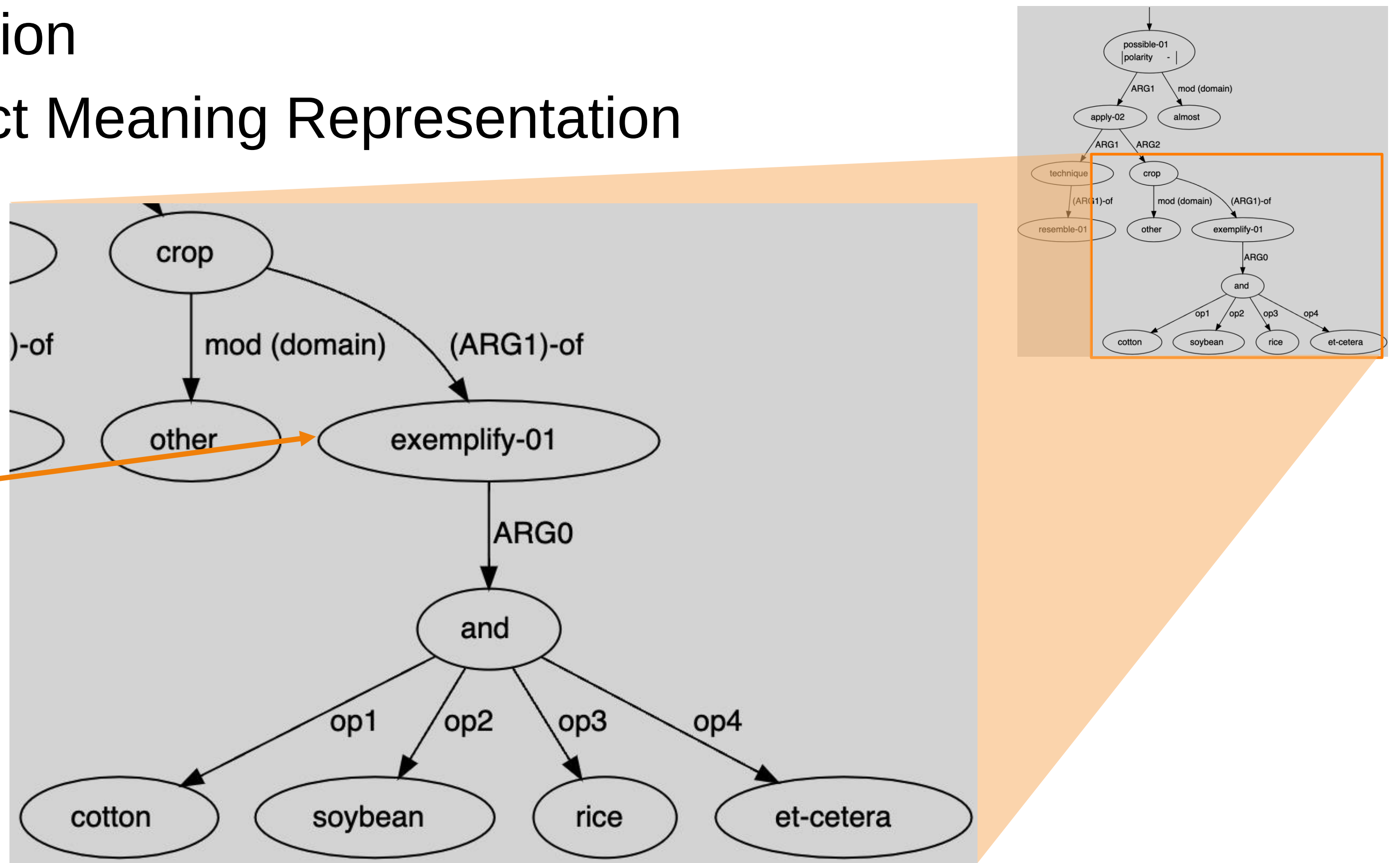
- Flavor 2
  - Node: not necessarily corresponding to words (unanchored)
  - Edge: relation
  - Ex: Abstract Meaning Representation



Semantic Graphs

- Flavor 2
  - Node: not necessarily corresponding to words (unanchored)
  - Edge: relation
  - Ex: Abstract Meaning Representation

…
other
cops,
such
as
cotton,
soybeans
…



Semantic Graphs

- Flavor 0
  - Node: word
  - Edge: relation
- Flavor 1
  - Node: arbitrary part of the sentence (sub-word, multiple words, no word)
  - Edge: relation
- Flavor 2
  - Node: not necessarily corresponding to words (unanchored)
  - Edge: relation

Semantic Parsing

- Translating a sentence to its semantic representation
  - Syntax-driven approach
  - Neural approach

Syntax-Driven Semantic Parsing

The Principle of Compositionality

- The meaning of a NL phrase is determined by the meanings of its sub-phrases.
- Phrase ⇨ sub-phrases: this is syntax!
- Syntax-driven semantic parsing
  - follow a constituency syntactic tree from bottom up
  - repeatedly compose semantics of sub-phrases together

- First of all, we need a way to express semantics of phrases
  - We've already talked about sentence meaning representations, e.g., FOL
  - But phrases are incomplete pieces of meanings

λ-Calculus

- Informally, two extensions over FOL
  - λ-abstraction
    - If ϕ is a FOL formula and v is a variable, then $\lambda v.\phi$ is a λ-term, meaning an unnamed function or map from values (of v) to formulas (usually involving v)
    - Notational conventions:

$$\lambda x.\left(\lambda y.f(x,y)\right) = \lambda x.\lambda y.f(x,y) = \lambda xy.f(x,y)$$

  - Application (or λ-reduction)
    - If we have $\lambda v.\phi$ and ψ, then $[\lambda v.\phi](\psi)$ is a formula.
    - It can be reduced by substituting every instance of v in ϕ with ψ

λ-Calculus Examples

- $\lambda x. Likes(x, NLP)$
  - A map of someone to a statement that he likes NLP
  - $[\lambda x. Likes(x, NLP)](a)$ reduces to $Likes(a, NLP)$
- $\lambda x. \lambda y. Friends(x, y)$
  - A map of thing x to a map of thing y to a statement that x and y are friends
  - $[\lambda x. \lambda y. Friends(x, y)](a)$ reduces to $\lambda y. Friends(a, y)$
  - $\big[[\lambda x. \lambda y. Friends(x, y)](a)\big](b)$ reduces to $[\lambda y. Friends(a, y)](b)$, which reduces to $Friends(a, b)$
- $\lambda f. f(a, b)$
  - A map of relation f to a statement that a and b have relation f
  - $[\lambda f. f(a, b)](\lambda x. \lambda y. Friends(x, y))$ reduces to $[\lambda x. \lambda y. Friends(x, y)](a, b)$, which reduces $Friends(a, b)$

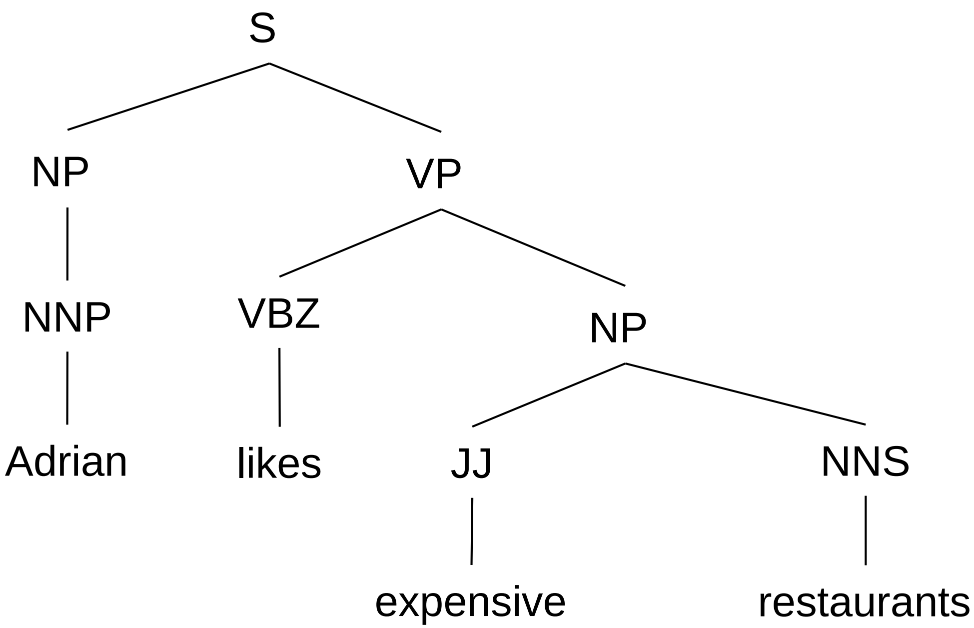
Example CFG

- NNP → Adrian
- VBZ → likes
- JJ → expensive
- NNS → restaurants
- NP → NNP
- NP → JJ NNS
- VP → VBZ NP
- S → NP VP

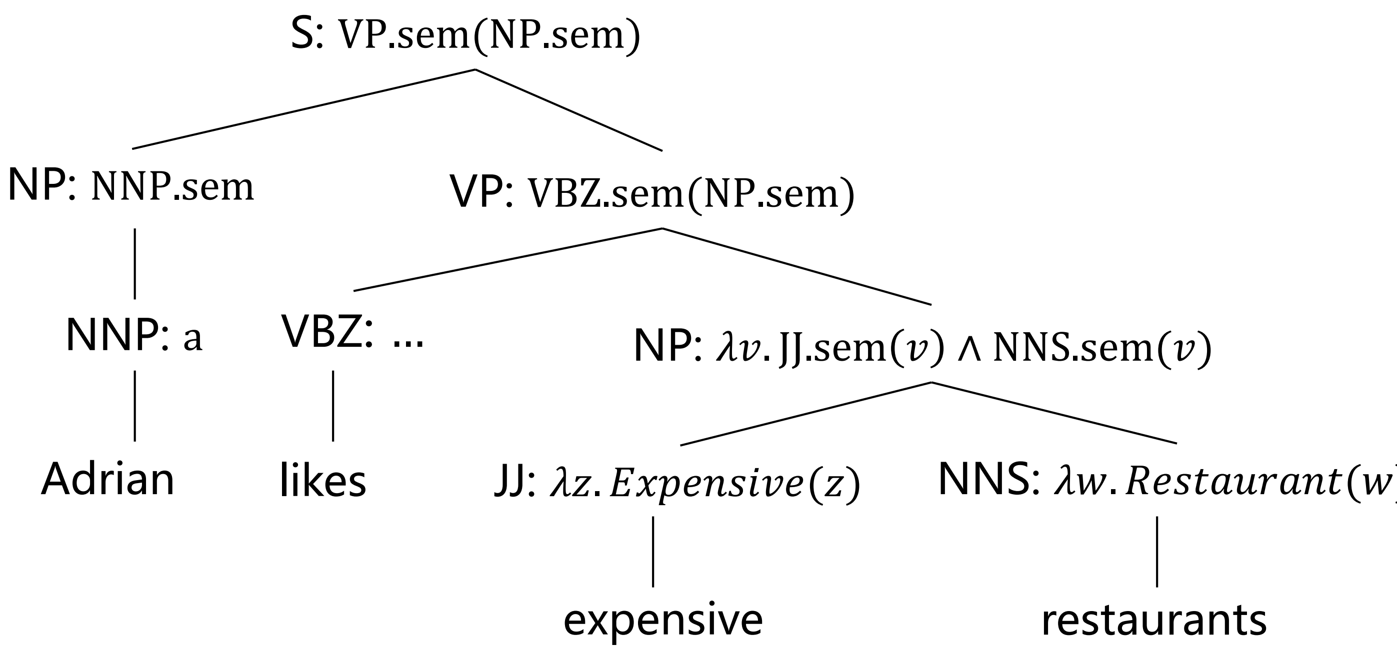
Semantic Attachments to CFG

- $\text{NNP} \rightarrow \text{Adrian}\ \{a\}$
- $\text{VBZ} \rightarrow \text{likes}\ \{\lambda f y. \forall x f(x) \Rightarrow Likes(y, x)\}$
- $\text{JJ} \rightarrow \text{expensive}\ \{\lambda x. Expensive(x)\}$
- $\text{NNS} \rightarrow \text{restaurants}\ \{\lambda x. Restaurant(x)\}$
- $\text{NP} \rightarrow \text{NNP}\ \{\boxed{\text{NNP.sem}}\}$ an undetermined formula of NNP
- $\text{NP} \rightarrow \text{JJ NNS}\ \{\lambda x. \text{JJ.sem}(x) \wedge \text{NNS.sem}(x)\}$
- $\text{VP} \rightarrow \text{VBZ NP}\ \{\text{VBZ.sem}(\text{NP.sem})\}$
- $\text{S} \rightarrow \text{NP VP}\ \{\text{VP.sem}(\text{NP.sem})\}$

Example

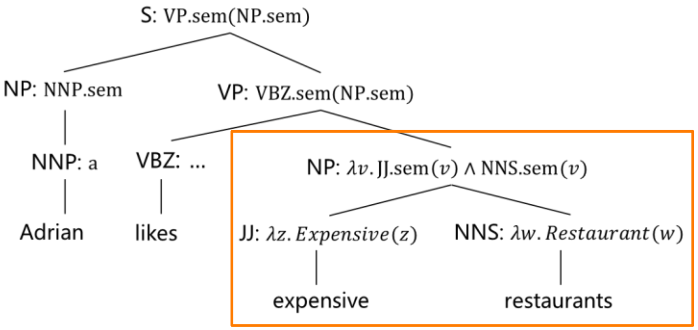


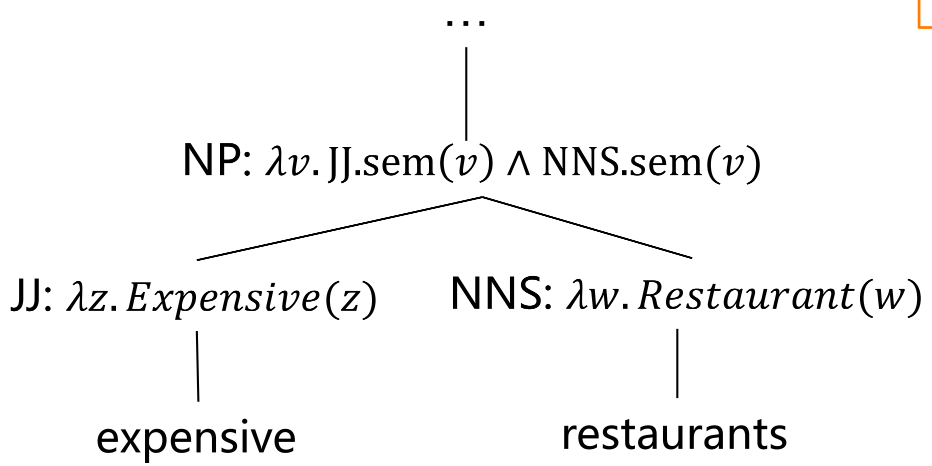
Example



S: $\mathrm{VP.sem}(\mathrm{NP.sem})$
NP: $\mathrm{NNP.sem}$
VP: $\mathrm{VBZ.sem}(\mathrm{NP.sem})$
NNP: a
VBZ: …
NP: $\lambda v.\,\mathrm{JJ.sem}(v) \wedge \mathrm{NNS.sem}(v)$
Adrian
likes
JJ: $\lambda z.\,Expensive(z)$
NNS: $\lambda w.\,Restaurant(w)$
expensive
restaurants

Example

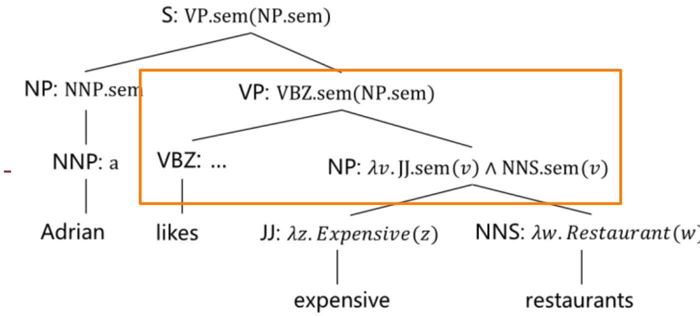


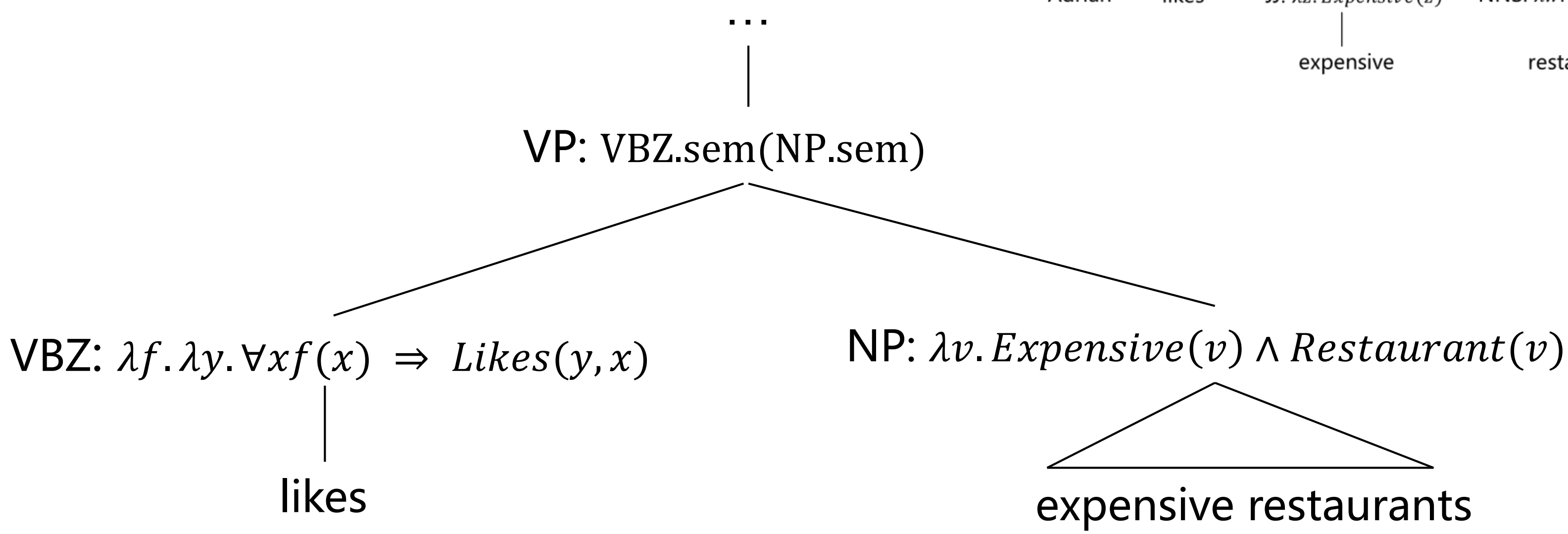


$$\lambda v.\,[\underbrace{\lambda z.\,Expensive(z)}_{\text{JJ.sem}}](v) \wedge [\underbrace{\lambda w.\,Restaurant(w)}_{\text{NNS.sem}}](v)$$

$$\lambda v.\,Expensive(v) \wedge Restaurant(v)$$

Example



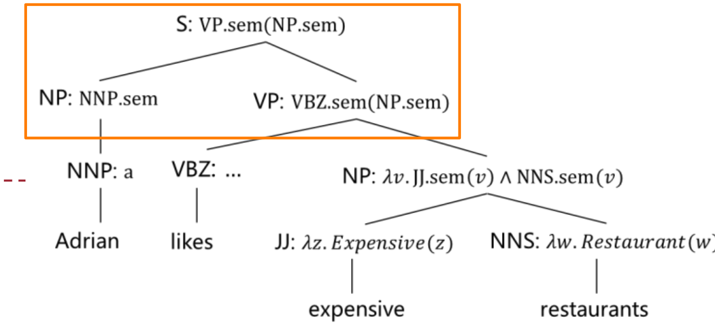


$$\underbrace{[\lambda f.\lambda y.\forall x f(x) \Rightarrow Likes(y,x)]}_{\text{VBZ.sem}}\underbrace{(\lambda v. Expensive(v) \wedge Restaurant(v))}_{\text{NP.sem}}$$

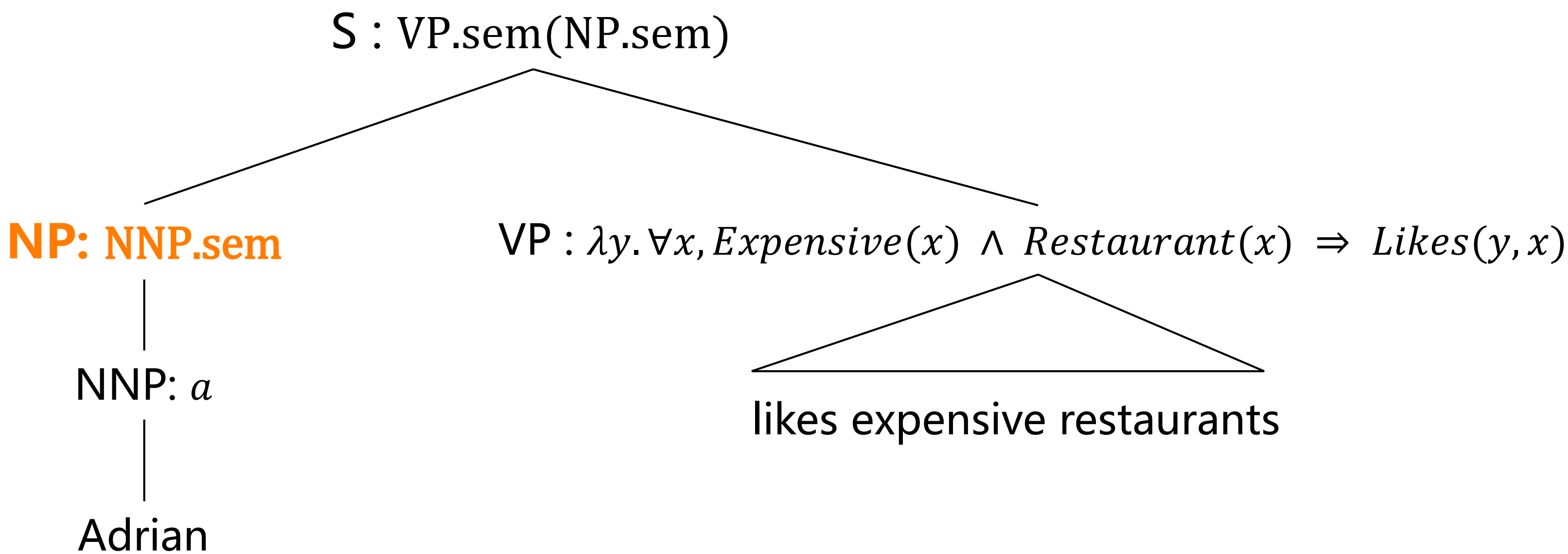
$$\lambda y.\forall x\, [\lambda v. Expensive(v) \wedge Restaurant(v)]\,(x) \Rightarrow Likes(y,x)$$

$$\lambda y.\forall x, Expensive(x) \wedge Restaurant(x) \Rightarrow Likes(y,x)$$

Example

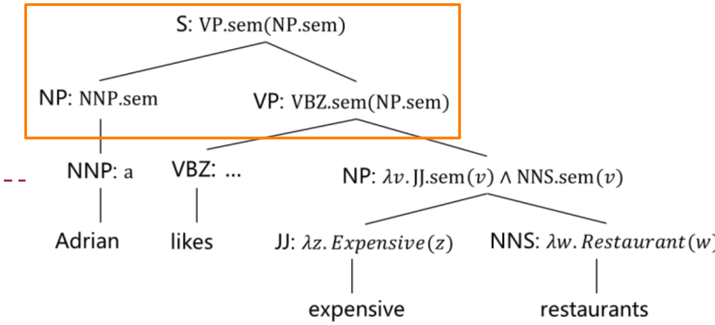


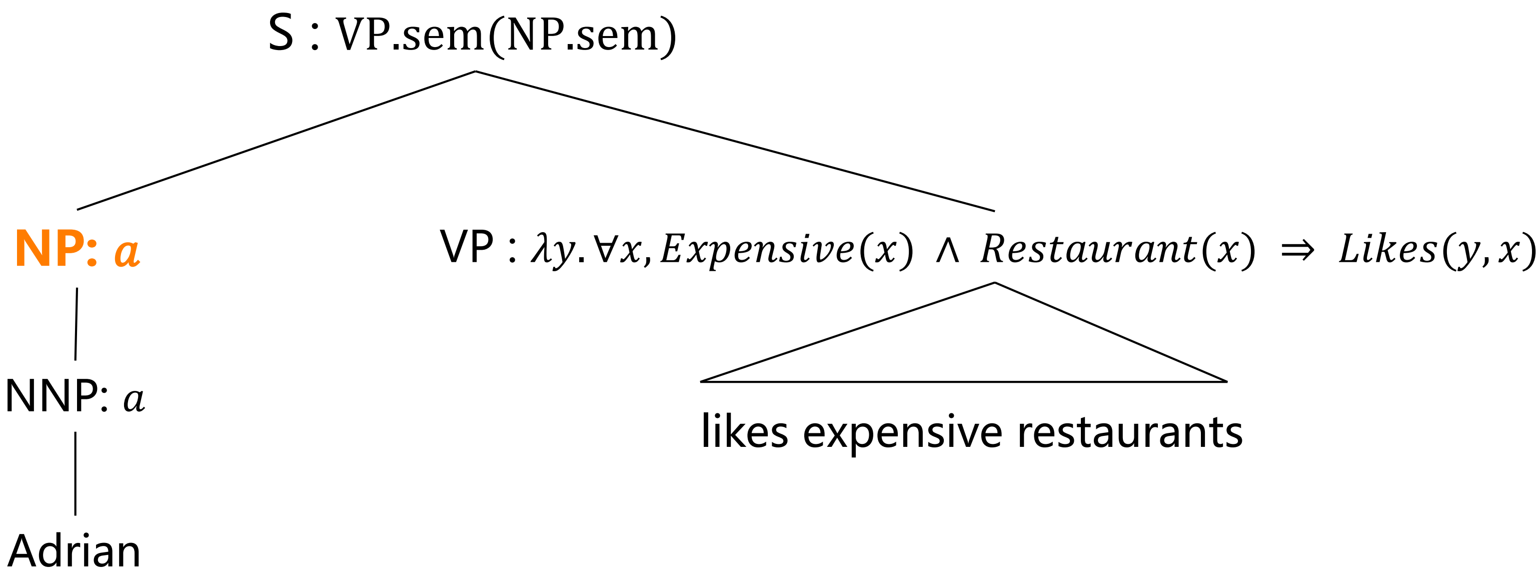
S: VP.sem(NP.sem)
NP: NNP.sem
VP: VBZ.sem(NP.sem)
NNP: a
VBZ: ...
NP: $\lambda v.$ JJ.sem$(v) \wedge$ NNS.sem(v)
Adrian
likes
JJ: $\lambda z. Expensive(z)$
NNS: $\lambda w. Restaurant(w)$
expensive
restaurants



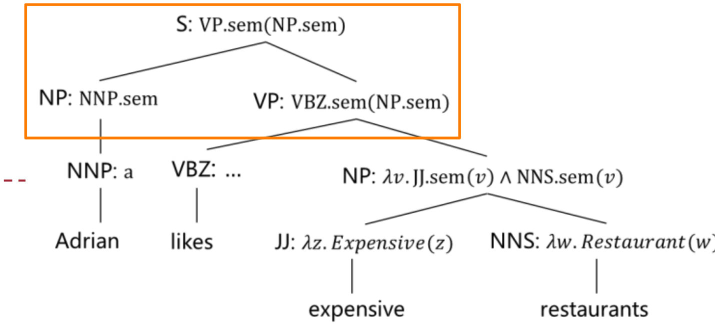
S : VP.sem(NP.sem)
NP: NNP.sem
VP : $\lambda y. \forall x, Expensive(x) \wedge Restaurant(x) \Rightarrow Likes(y, x)$
NNP: a
likes expensive restaurants
Adrian

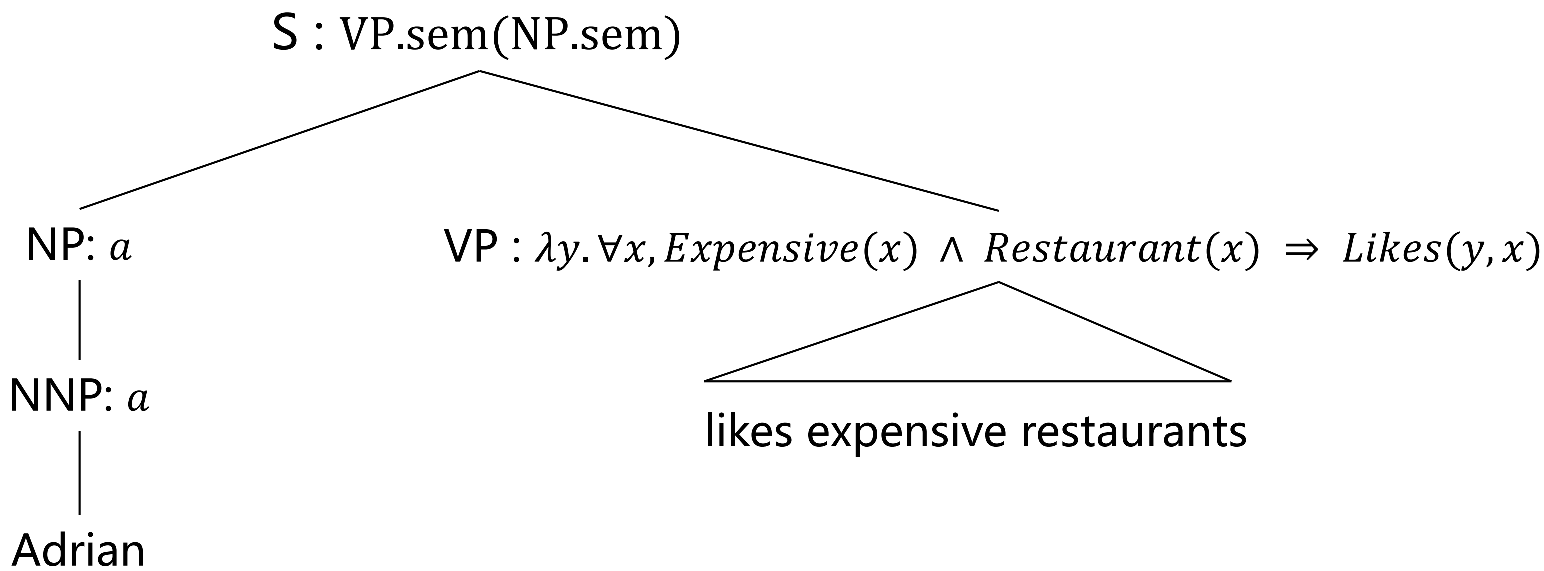
Example


S: VP.sem(NP.sem)
NP: NNP.sem
VP: VBZ.sem(NP.sem)
NNP: a
VBZ: ...
NP: $\lambda v.$ JJ.sem$(v) \wedge$ NNS.sem(v)
Adrian
likes
JJ: $\lambda z. Expensive(z)$
NNS: $\lambda w. Restaurant(w)$
expensive
restaurants


S : VP.sem(NP.sem)
NP: a
VP : $\lambda y. \forall x, Expensive(x) \wedge Restaurant(x) \Rightarrow Likes(y, x)$
NNP: a
likes expensive restaurants
Adrian

Example





$$\underbrace{[\lambda y. \forall x, Expensive(x) \wedge Restaurant(x) \Rightarrow Likes(y, x)]}_{\text{VP.sem}}\underbrace{(\ a\ )}_{\text{NP.sem}}$$

$$\forall x, Expensive(x) \wedge Restaurant(x) \Rightarrow Likes(a, x)$$

Neural Semantic Parsing

Neural Models

- Sequence-to-sequence (to be introduced later)
  - Input: sentence
  - Output:
    - Logic formula
    - Linearized semantic graph (e.g., depth-first traversal)

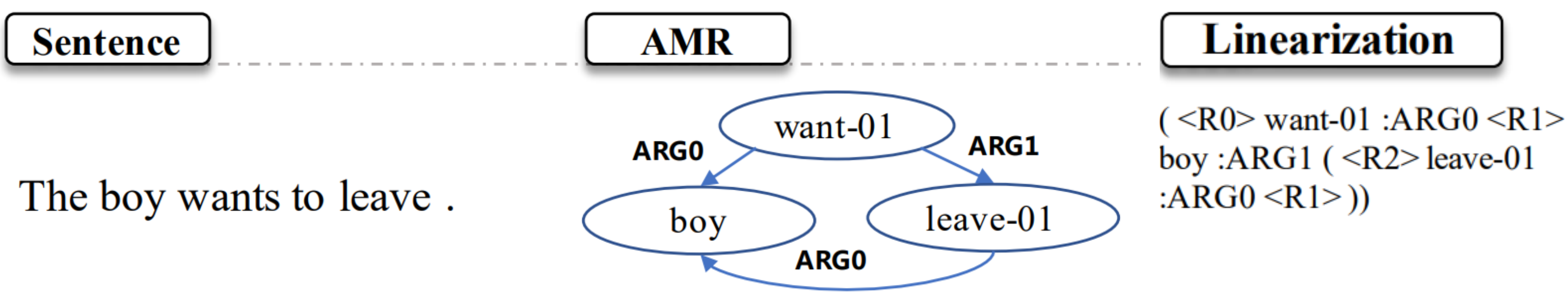


Image from Chen et al., AMRize, then Parse! Enhancing AMR Parsing with PseudoAMRs. 2022

Neural Models

- Parsing to semantic graph
  - Transition-based method
    - Similar to transition-based parsing, but with actions that build a graph instead of a tree

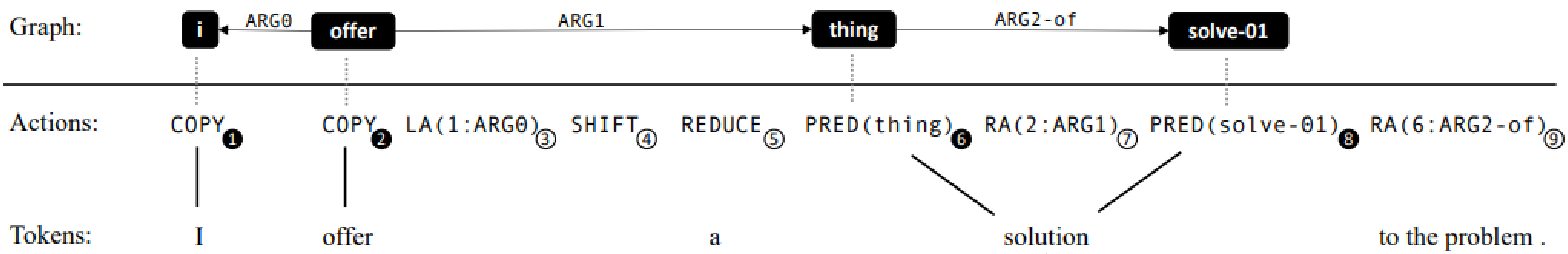


Image from Jiawei et al. AMR Parsing with Action-Pointer Transformer. 2021.

  - Graph-based method
    - First generate a set of nodes (using seq2seq or seq2set) and then predict edges between them (like dependency parsing)
    - Or, generate nodes and edges alternately

Neural Models

- A variety of other methods…
- Many ongoing researches

Learning

- Supervised learning
  - Challenge: manual annotation of semantic representations is difficult and costly
- Weakly supervised learning
  - Correct semantic representation not available
  - But in many scenarios, semantic representation is executable and we know the correct outcomes
    - Ex. The correct SQL for a NL question is not known, but the correct answer is known
      - “What is the capital of France?” ⇨ Paris
  - Supervised learning with latent variables, reinforcement learning

Semantic Role Labeling

Semantic Role Labeling (SRL)

- Semantic parsing produces the complete meaning representation of a sentence
- SRL only identifies predicate-argument structures in a sentence
  - A shallow semantic representation
  - No fine-grained meaning representation inside each argument

Who **did what** **to whom** **at where?**

The police officer detained the suspect at the scene of the crime

Agent Predicate Theme Location

Examples: Who did What to Who(m)?

- **XYZ corporation bought the stock**.
- **They sold the stock** to **XYZ corporation**.
- **The stock** was **bought** by **XYZ corporation**.
- The **purchase** of **the stock** by **XYZ corporation**...
- **The stock purchase** by **XYZ corporation**...

- Predicates (bought, sold, purchase) represent an event
- Semantic roles express the abstract role that arguments of a predicate can take in the event

Two widely used semantic role specifications

- **FrameNet**
  - more roles
  - define roles specific to a group of predicates
- **PropBank**
  - fewer roles
  - define generalized semantic roles (prototypes)

<u>XYZ corporation</u> bought the stock

More specific ⟷ More general

buyer **agent** **proto-agent**

The volitional causer of an event

PropBank

- Data resource: annotated on top of the Penn Treebank (so arguments are always constituents).
- Each verb sense has a specific set of roles.
  - I.e., semantic roles in PropBank are verb-sense specific.
- These roles are given numbers rather than names (e.g., Arg0, Arg1).

PropBank Roles

- Arg0: PROTO-AGENT
  - Volitional involvement in event or state
  - Sentience (and/or perception)
  - Causes an event or change of state in another participant
  - Movement (relative to position of another participant)
- Arg1: PROTO-PATIENT
  - Undergoes change of state
  - Causally affected by another participant
  - Stationary relative to movement of another participant

PropBank Roles

- Arg0: PROTO-AGENT
- Arg1: PROTO-PATIENT
- *Arg2-5 are not really that consistent*
  - Arg2: usually: benefactive, instrument, attribute, or end state
  - Arg3: usually: start point, benefactive, instrument, or attribute
  - Arg4: usually: the end point

PropBank Roles

- Arg-M: modifiers or adjuncts of the predicate

ArgM-TMP	when?	yesterday evening, now
LOC	where?	at the museum, in San Francisco
DIR	where to/from?	down, to Bangkok
MNR	how?	clearly, with much enthusiasm
PRP/CAU	why?	because ... , in response to the ruling
REC		themselves, each other
ADV	miscellaneous	
PRD	secondary predication	...ate the meat raw

fall.01 (move downward)

- Arg1: logical subject, patient, thing falling
- Arg2: extent, amount fallen
- Arg3: starting point
- Arg4: ending point
- ArgM-LOC: medium

Examples:

- Sales fell to $251.2 million from $278.8 million.
- The average junk bond fell by 4.2%.
- The meteor fell through the atmosphere, crashing into Palo Alto.

fall.01 (move downward)

- Arg1: logical subject, patient, thing falling
- Arg2: extent, amount fallen
- Arg3: starting point
- Arg4: ending point
- ArgM-LOC: medium

Examples:

- Sales fell to $251.2 million from $278.8 million.
- The average junk bond fell by 4.2%.
- The meteor fell through the atmosphere, crashing into Palo Alto.

fall.01 (move downward)

- Arg1: logical subject, patient, thing falling
- Arg2: extent, amount fallen
- Arg3: starting point
- Arg4: ending point
- ArgM-LOC: medium

Examples:

- Sales fell to $251.2 million from $278.8 million.
- The average junk bond fell by 4.2%.
- The meteor fell through the atmosphere, crashing into Palo Alto.

fall.01 (move downward)

- Arg1: logical subject, patient, thing falling
- Arg2: extent, amount fallen
- Arg3: starting point
- Arg4: ending point
- ArgM-LOC: medium

Examples:

- Sales fell to $251.2 million from $278.8 million.
- The average junk bond fell by 4.2%.
- The meteor fell through the atmosphere, crashing into Palo Alto.

fall.01 (move downward)

- Arg1: logical subject, patient, thing falling
- Arg2: extent, amount fallen
- Arg3: starting point
- Arg4: ending point
- ArgM-LOC: medium

Examples:

- Sales fell to $251.2 million from $278.8 million.
- The average junk bond fell by 4.2%.
- The meteor fell through the atmosphere, crashing into Palo Alto.

fall.01 (move downward)

- Arg1: logical subject, patient, thing falling
- Arg2: extent, amount fallen
- Arg3: starting point
- Arg4: ending point
- ArgM-LOC: medium

Examples:

- Sales fell to $251.2 million from $278.8 million.
- The average junk bond fell by 4.2%.
- The meteor fell through the atmosphere, crashing into Palo Alto.

fall.08 (fall back, rely on in emergency)

- Arg0: thing falling back
- Arg1: thing fallen back on

Example:

- World Bank president Paul Wolfowitz has fallen back on his last resort.

fall.08 (fall back, rely on in emergency)

- Arg0: thing falling back
- Arg1: thing fallen back on

Example:

- World Bank president Paul Wolfowitz has fallen back on his last resort.

fall.08 (fall back, rely on in emergency)

- Arg0: thing falling back
- Arg1: thing fallen back on

Example:

- World Bank president Paul Wolfowitz has fallen back on his last resort.

fall.10 (fall for a trick; be fooled by)

- Arg1: the fool
- Arg2: the trick

Example:

- Many people keep falling for the idea that lowering taxes on the rich benefits everyone.

fall.10 (fall for a trick; be fooled by)

- Arg1: the fool
- Arg2: the trick

Example:

- Many people keep falling for the idea that lowering taxes on the rich benefits everyone.

fall.10 (fall for a trick; be fooled by)

- Arg1: the fool
- Arg2: the trick

Example:

- Many people keep falling for the idea that lowering taxes on the rich benefits everyone.

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

- Roles are specific to a frame.
- Frames can be any content word (verb, noun, adjective, adverb)
- About 1,000 frames, each with its own roles
- Both frames and roles are hierarchically organized
- Annotated without syntax, so that arguments can be anything

The “Change position on a scale” Frame

- This frame consists of words that indicate the change of an ITEM’s position on a scale (the ATTRIBUTE) from a starting point (INITIAL VALUE) to an end point (FINAL VALUE)
- It consists of the following words:
  - **Verbs**: advance, climb, decline, decrease, diminish, dip, double, drop, dwindle, edge, explode, fall, fluctuate, gain, grow, increase, jump, move, mushroom, plummet, reach, rise, rocket, shift, skyrocket, slide, soar, swell, swing, triple, tumble
  - **Nouns**: decline, decrease, escalation, explosion, fall, fluctuation, gain, growth, hike, increase, rise, shift, tumble
  - **Adverb**: increasingly

The “Change position on a scale” Frame

- **Item**: entity that has a position on the scale
- **Attribute**: scalar property that the Item possesses
- **Difference**: distance by which an Item changes its position
- **Final state**: Item’s state after the change
- **Final value**: position on the scale where Item ends up
- **Initial state**: Item’s state before the change
- **Initial value**: position on the scale from which the Item moves
- **Value range**: portion of the scale along which values of Attribute fluctuate
- **Duration**: length of time over which the change occurs
- **Speed**: rate of change of the value
- **Group**: the group in which an Item changes the value of an Attribute

The "Change position on a scale" Frame

- Examples

[$_{\text{ITEM}}$ Oil] *rose* [$_{\text{ATTRIBUTE}}$ in price] [$_{\text{DIFFERENCE}}$ by 2%].

[$_{\text{ITEM}}$ It] has *increased* [$_{\text{FINAL\_STATE}}$ to having them 1 day a month].

[$_{\text{ITEM}}$ Microsoft shares] *fell* [$_{\text{FINAL\_VALUE}}$ to 7 5/8].

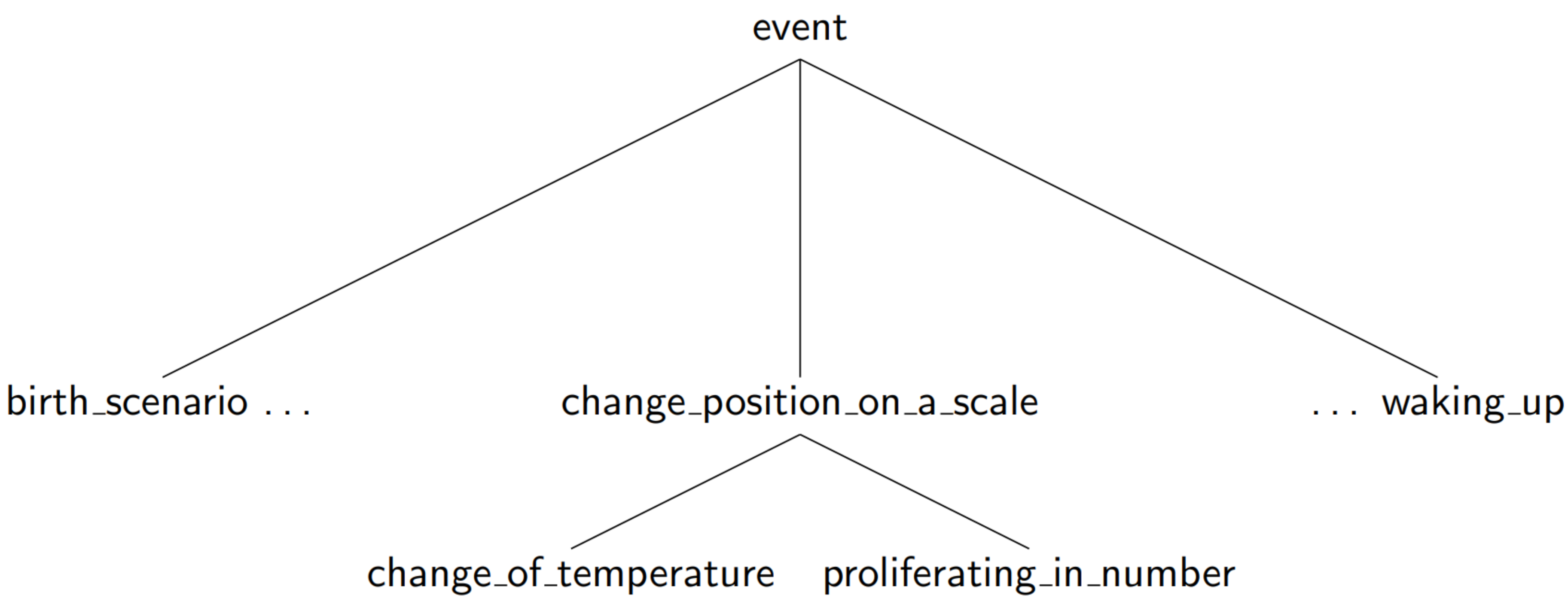
[$_{\text{ITEM}}$ Colon cancer incidence] *fell* [$_{\text{DIFFERENCE}}$ by 50%] [$_{\text{GROUP}}$ among men].

a steady *increase* [$_{\text{INITIAL\_VALUE}}$ from 9.5] [$_{\text{FINAL\_VALUE}}$ to 14.3] [$_{\text{ITEM}}$ in dividends]

a [$_{\text{DIFFERENCE}}$ 5%] [$_{\text{ITEM}}$ dividend] *increase*...

The “Change position on a scale” Frame

- Hierarchical organization



Semantic Role Labeling

- Input: a sentence x
- Output:
  - A collection of **predicates**, each consisting of:
    - A label, sometimes called the **frame**
    - A span
    - A set of arguments, each consisting of:
      - A label, usually called the **role**
      - A span

In some settings, predicates are given.

Example:

$$\overbrace{\text{You}}^{\text{ARG0}}\ \text{can't}\ \overbrace{\text{blame}}^{\text{TARGET}}\ \overbrace{\text{the program}}^{\text{ARG1}}\ \overbrace{\text{for being unable to identify it}}^{\text{ARG2}}$$

$$\text{You can't blame}\ \overbrace{\text{the program}}^{\text{ARG0}}\ \text{for being unable to}\ \overbrace{\text{identify}}^{\text{TARGET}}\ \overbrace{\text{it}}^{\text{ARG1}}$$

Methods

- As sequence labeling
  - Use the BIO scheme to represent predicate and argument spans and labels
  - One tag sequence for each predicate

Example

$$\overbrace{\text{You}}^{\text{ARG0}}\quad \text{can't}\quad \overbrace{\text{blame}}^{\text{TARGET}}\quad \overbrace{\text{the program}}^{\text{ARG1}}\quad \overbrace{\text{for being unable to identify it}}^{\text{ARG2}}$$

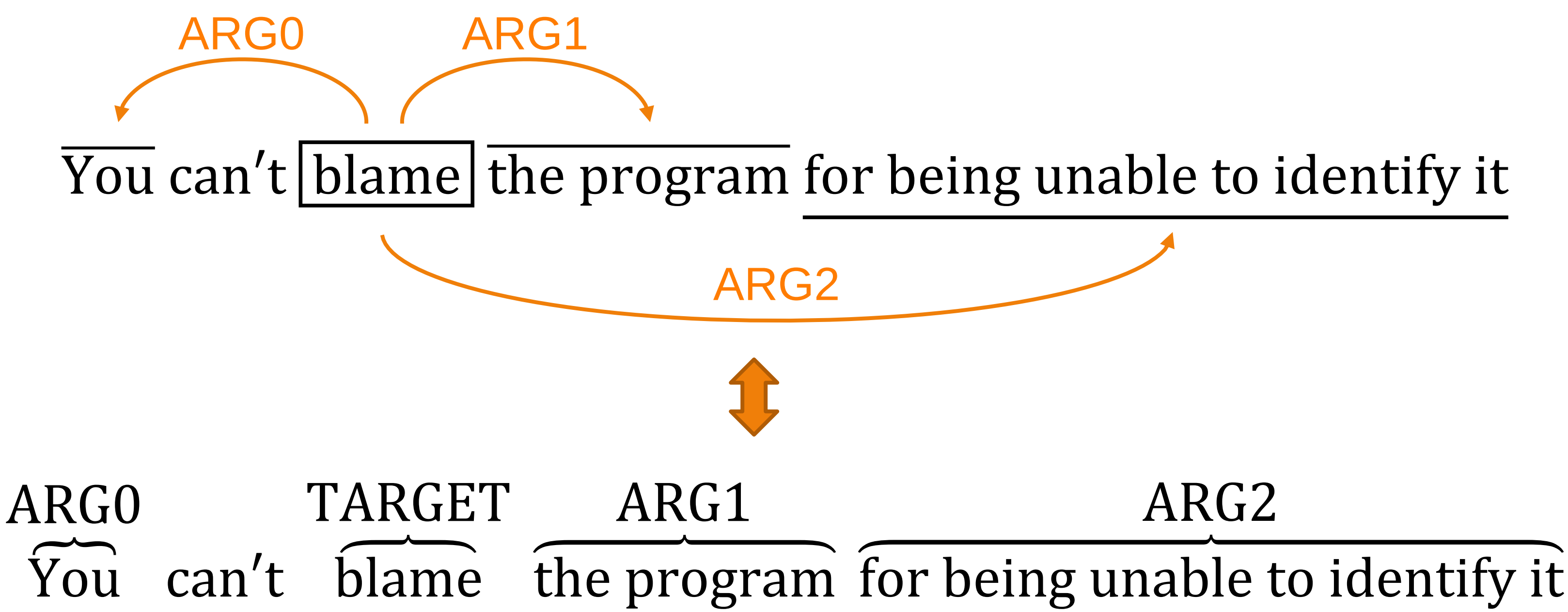
$$\text{You can't blame}\ \overbrace{\text{the program}}^{\text{ARG0}}\ \text{for being unable to}\ \overbrace{\text{identify}}^{\text{TARGET}}\ \overbrace{\text{it}}^{\text{ARG1}}$$

You	can't	blame	the	program	for	being	unable	to	identify	it
B-ARG0	O	B-V	B-ARG1	I-ARG1	B-ARG2	I-ARG2	I-ARG2	I-ARG2	I-ARG2	I-ARG2
O	O	O	B-ARG0	I-ARG0	O	O	O	O	B-V	B-ARG1

Methods

- Graph-based methods
  - First predict predicate and argument spans
  - Then predicate roles as dependency arcs

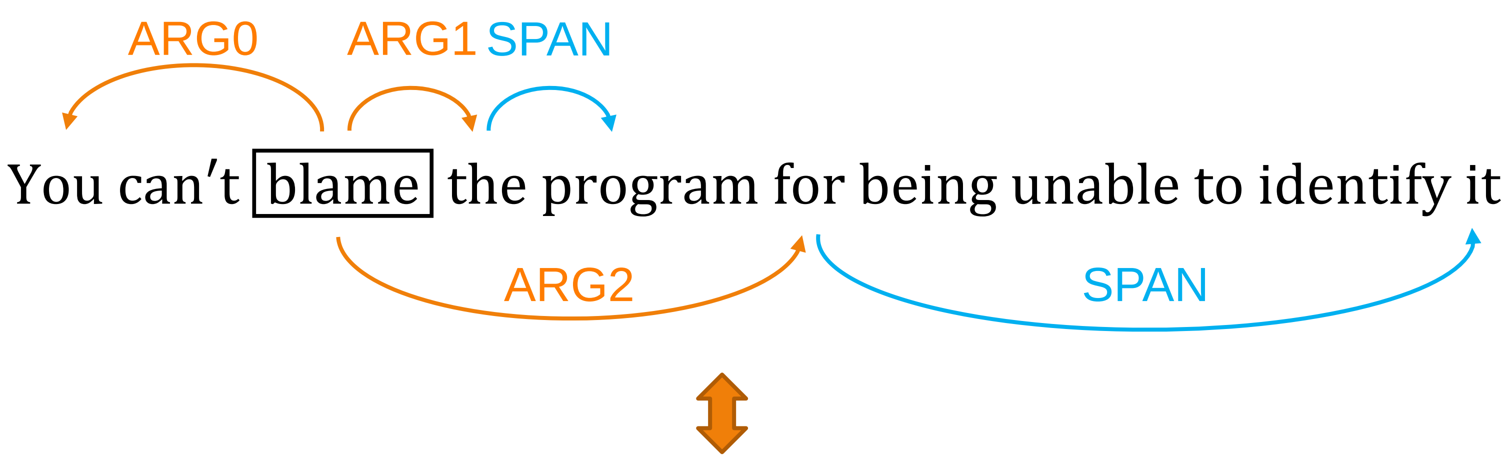
Example



Methods

- Graph-based methods
  - Predict both spans and roles as dependency arc

Example



$$\overbrace{\text{You}}^{\text{ARG0}}\ \text{can't}\ \overbrace{\text{blame}}^{\text{TARGET}}\ \overbrace{\text{the program}}^{\text{ARG1}}\ \overbrace{\text{for being unable to identify it}}^{\text{ARG2}}$$

Summary

Sentence Semantics

- Vector vs. symbolic representation of sentences
- Formal Meaning Representation
  - Special-purpose representations
  - General-purpose representations: formal logic, semantic graphs
- Syntax-Driven Semantic Parsing
  - λ-Calculus, Semantic Attachments to CFG
- Neural Semantic Parsing
  - Seq2seq, parsing to graph, ...
- Semantic Role Labeling
  - PropBank, FrameNet
  - Methods: sequence labeling, graph-based methods