CS 490: NATURAL LANGUAGE PROCESSING

Dan Goldwasser, Abulhair Saparov

Lecture 14: Semantics

RECAP

- We have discussed context-free grammars (CFGs)
  - Covers most (if not all) of natural language syntax
- We have talked about how to parse CFGs:
  - CKY, Earley
- We discussed structured prediction:
  - A more general class of machine learning problems where the output labels are structures.
  - Syntactic parsing is a subset of structured prediction
    - Outputs are syntax trees
  - Viterbi's algorithm is a great way to predict sequences.

STRUCTURED PREDICTION

- Structured prediction is the task where the output is structured.
  - E.g., syntax trees, sequences, graphs, tables, etc.
- This task usually involves discrete optimization/search,
  - For example via algorithms like branch-and-bound.
- Sequence prediction is a kind of structured prediction.
  - Let's say we have some objective function f over sequences of items.
    - E.g., this could be a sequence of words.
    - E.g., Traveling Salesman Problem: suppose we need to make n deliveries in n different cities. In what order should we visit the cities to minimize the overall distance traveled?
- Goal: Find the sequence $x_1, \ldots, x_n$ such that $f(x_1, \ldots, x_n)$ is maximized.
  - In traveling salesman, f is the negative distance.

SEQUENCE PREDICTION

- We can apply branch-and-bound to sequence prediction.
- The set S is the set of all sequences of length n.
- Each state is a partial sequence: $x_1, \ldots, x_k$
  - Represents the set of all sequences of length n that start with $x_1, \ldots, x_k$.
  - E.g., the first k cities that we visit to make deliveries.
- What is the "branch" step?
  - For each possible next symbol x_{k+1}, we create a new state $x_1, \ldots, x_{k+1}$.
- What is the "bound"?
  - The simplest bound is the total cost so far:

$$\texttt{-distance}(x_1, \ldots, x_n) \leq \texttt{-distance}(x_1, \ldots, x_k),$$

$$= \texttt{-distance}(x_1, x_2) - \texttt{distance}(x_2, x_3) - \ldots - \texttt{distance}(x_{k-1}, x_k).$$

SEQUENCE PREDICTION

- This algorithm is too slow.
- In the worst case, we need a number of branches exponential in n.
  - We would need to search over all possible sequences of cities.
  - How many such sequences are there?
  - $n(n-1)(n-2)\ldots(2)(1) = n!$
  - (there are n possible values for the first city, then there are $n-1$ possible values for the second city, etc...)

SEQUENCE PREDICTION

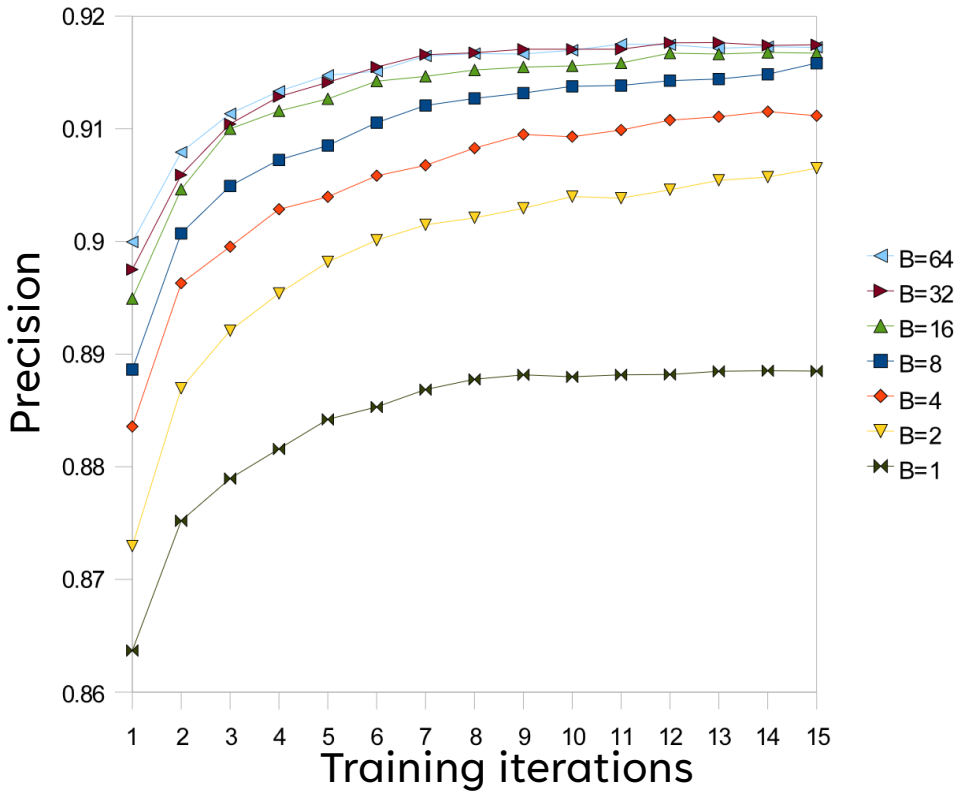
- How to make sequence prediction faster?
- We can trade optimality for performance.
  - We can limit the capacity of the priority queue.
    - Let B be the capacity.
  - After adding new states to the priority queue, simply remove the lowest priority states until only B elements remain.
- This is called beam search,
  - And B is the beam width or beam size.
- If $B = 1$, we have greedy search.
- If $B = \infty$, we recover exact search.

BEAM SEARCH IN PARSING

- Beam search can also be used in parsing.
- Used when there is an objective function over syntax trees.
  - E.g., a model that assigns probabilities/scores to syntax trees.
  - This would be very useful in choosing among ambiguous parses.
  - E.g., in ‘`Sally caught a butterfly with a net`,’ who has the net?
    - Ideally, the parse where Sally has the net should have higher probability/score than the parse where the butterfly has the net.
- Can significantly increase parsing speed if there are many ambiguous parses.

BEAM SEARCH IN PARSING

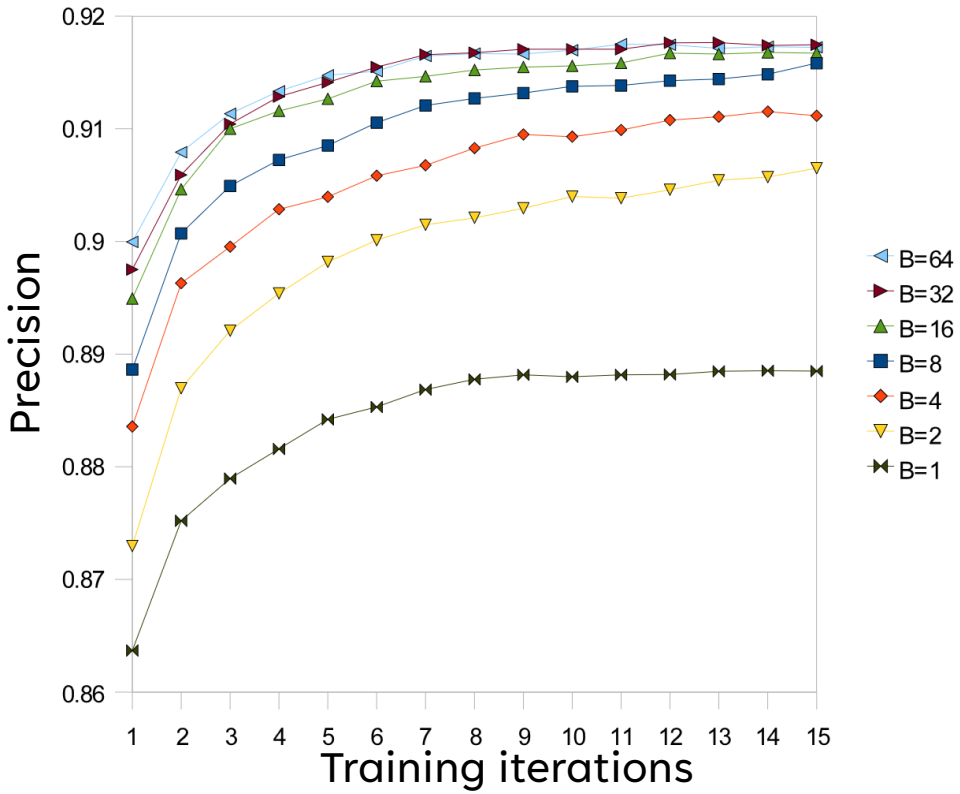
- Zhang and Clark (2008) used beam search for parsing.
- They used a neural model to assign probabilities to parser outputs.



[Zhang and Clark, 2008]

BEAM SEARCH IN PARSING

- They measured precision vs number of training iterations for the neural model vs beam size.



[Zhang and Clark, 2008]

SEMANTICS

HOW TO MODEL MEANING?

- How do we model the meaning of natural language utterances?
- Language can convey lots of different kinds of meaning:
  - Truth-functional: ‘Cats are mammals’, ‘Fish are not mammals.’
  - Modality:
    - ‘You must follow the law.’
    - ‘Compute the average of the list of numbers.’
  - Emphasis/topicalization:
    - ‘It wasn’t them who robbed the bank yesterday.’
    - ‘The bank wasn’t what they robbed yesterday.’
    - ‘It wasn’t yesterday that they robbed the bank.’

HOW TO MODEL MEANING?

- How do we model the meaning of natural language utterances?
- Language can convey lots of different kinds of meaning:
  - Verbal emphasis:
    - ‘***He*** didn’t rob the bank yesterday.’
    - ‘He didn’t ***rob*** the bank yesterday.’
    - ‘He didn’t rob the ***bank*** yesterday.’
    - ‘He didn’t rob the bank ***yesterday***.’
  - This is not a comprehensive list.

FORMAL SEMANTICS

- Formal semantics is the study of formal representations of meaning.
- Ideally, a good model of meaning can capture all semantic information.
- For simplicity, we will focus on truth-functional meaning.
  - We can interpret the meaning of each sentence as a function.
  - The input to the function is a possible world (or "model").
  - The output is `TRUE` or `FALSE`.
- E.g., '`All insects have six legs`'
  - If the possible world is one in which every instance of an insect has six legs, the output value is `TRUE`.
  - If the possible world has an insect that does not have six legs, the output value is `FALSE`.

HOW TO MODEL MEANING?

- Truth-functional meaning representations are useful for tasks that require complex reasoning.
- For example:
  - Question answering
    - Especially multi-hop QA
  - Mathematical reasoning
- Logic provides a natural way to represent meaning in these applications.

PROPOSITIONAL LOGIC

- Propositional logic or propositional calculus is a logic that describes propositions and relations between them.
  - Propositions are represented as symbols or variables.
  - Each proposition can either be `TRUE` or `FALSE`.
- E.g., the logical form of '`Sally is a cat`' is `sally_is_cat = TRUE`.
  - '`Sally is a cat and Bob likes dogs`'
    - `sally_is_cat & bob_likes_dogs`
    - `sally_is_cat ∧ bob_likes_dogs`
  - '`Sally is a cat or Bob likes dog`'
    - `sally_is_cat | bob_likes_dogs`
    - `sally_is_cat ∨ bob_likes_dogs`

PROPOSITIONAL LOGIC

- Propositional logic has the following operators/relations:
  - Conjunction ("and"):
    - `A & B` is true if and only if both `A` and `B` are true
  - Disjunction ("or"):
    - `A | B` is true if either `A` or `B` are true
  - Negation ("not"):
    - `~A` or `!A` or `−A` or `¬A` is true if `A` is false
  - Implication ("if-then", "implies"):
    - `A -> B` or `A => B` or `A ⊃ B` is true if `B` is true whenever `A` is true

PROPOSITIONAL LOGIC

- One common task in reasoning is deduction.
  - Given some premises, prove whether a conclusion is `TRUE` or `FALSE`.
  - E.g., Given `sally_is_cat` and `bob_likes_dogs`,
  - Prove: `(sally_is_cat & bob_likes_dogs) | charlie_has_wings`
- This is equivalent to proving that in any possible world where the premises are true, the conclusion is necessarily true.
  - In propositional logic, we can search over all possible truth value assignments to the propositions.
    - Exponential running time
  - Inference in propositional logic can be shown to be NP-complete.

BETTER MODEL OF REASONING?

- Searching over all possible truth assignments is not a good model of human reasoning.
  - And it may not be a good model of reasoning more generally if we want to design systems that can solve complex reasoning problems.
- Can we model reasoning as a step-by-step process?
- We can define inference rules that tell us how to deduce new true facts.

$$\frac{A\ true \quad B\ true}{A \wedge B\ true}\ \wedge I$$

- In this rule, given the premises `A` is true and `B` is true,
  - We can conclude that `A & B` is true.

PROOFS

- In addition to conjunction introduction, we can define a conjunction elimination rule.

$$\frac{A \wedge B \ true}{A \ true} \wedge E$$

- We can similarly define inference rules for other logical connectives:

$$\frac{A \ true}{A \vee B \ true} \vee I$$

$$\frac{A \vee B \ true \quad \begin{array}{c}\overline{A \ true}\ u \\ \vdots \\ C \ true\end{array} \quad \begin{array}{c}\overline{B \ true}\ w \\ \vdots \\ C \ true\end{array}}{C \ true} \vee E^{u,w}$$

This rule is also called proof-by-cases

[Pfenning, Automated Theorem Proving Course Notes, 2004]

PROOFS

- We can similarly define inference rules for other logical connectives:

$$\frac{A \supset B \ true \quad A \ true}{B \ true} \supset E$$

This rule is also called "modus ponens"

$$\frac{\dfrac{}{A \ true}\, u \\ \vdots \\ B \ true}{A \supset B \ true} \supset I^u$$

[Pfenning, Automated Theorem Proving Course Notes, 2004]

PROOFS

- We can similarly define inference rules for other logical connectives:

$$\dfrac{\dfrac{}{\neg A}\,u \\ \vdots \\ \bot}{A}\,\bot^{u}_{C}$$

This rule is also called proof by contradiction

$$\dfrac{\neg A\ true \qquad A\ true}{C\ true}\,\neg\mathrm{E}$$

[Pfenning, Automated Theorem Proving Course Notes, 2004]

PROOFS

- We can use proofs to solve the earlier reasoning example:
  - E.g., Given `sally_is_cat` and `bob_likes_dogs`,
  - Prove: `(sally_is_cat & bob_likes_dogs) | charlie_has_wings`

$$\dfrac{\dfrac{\dfrac{}{\texttt{sally\_is\_cat}}\,\text{Ax} \quad \dfrac{}{\texttt{bob\_likes\_dogs}}\,\text{Ax}}{\texttt{sally\_is\_cat \& bob\_likes\_dogs}}\,\&\text{I}}{\texttt{(sally\_is\_cat \& bob\_likes\_dogs) | charlie\_has\_wings}}\,|\text{I}$$

- The step-by-step proof also provides a better model of human step-by-step reasoning.
- The above is an example of a depth-2 proof.

PROPOSITIONAL LOGIC LIMITATIONS

- Propositional logic does not cover the meaning of all natural language.
- E.g., ‘All states have capitals’ or ‘There is a city on the river.’
- In propositional logic, we can only express statements about specific objects.
  - In order to represent the above sentences, we need a way to express statements over all objects.
- Propositional logic can be extended by adding variables and quantifiers.
  - ‘All states have capitals’
    - ∀x(state(x) -> ∃y(capital(y) & has(x,y)))
  - ‘There is a city on the river’
    - ∃x(city(x) & on(x,the_river))
- Propositional logic + quantifiers + variables = first-order logic (FOL).

FIRST-ORDER LOGIC

- In the worst-case, reasoning in first-order logic is significantly more computationally expensive than in propositional logic.
- In fact, the problem of determining whether a given logical form is true in first-order logic is undecidable.
  - I.e., there is no algorithm that will solve any FOL problem in finite time.
- However, FOL is also highly expressive.
- Mathematics can largely be written in FOL.
  - See Zermelo-Fraenkel set theory.
  - The most common “foundation of mathematics.”

CAN WE TRAIN NLP MODELS TO REASON?

- Han et al (2024) create a dataset of FOL reasoning problems, written in both natural language and logic. (this is an example of question-answering)

A FOLIO example based on the Wild Turkey Wikipedia page: https://en.wikipedia.org/wiki/Wild_turkey

NL premises

1. There are six types of wild turkeys: Eastern wild turkey, Osceola wild turkey, Gould's wild turkey, Merriam's wild turkey, Rio Grande wild turkey, and the Ocellated wild turkey.
2. Tom is not an Eastern wild turkey.
3. Tom is not an Osceola wild turkey.
4. Tom is also not a Gould's wild turkey.
5. Tom is neither a Merriam's wild turkey, nor a Rio Grande wild turkey.
6. Tom is a wild turkey.

NL Conclusions -> Labels

A. Tom is an Ocellated wild turkey. -> True
B. Tom is an Eastern wild turkey. -> False
C. Joey is a wild turkey. -> Unknown

FOL Premises

1. $\forall x(\text{WildTurkey}(x) \rightarrow (\text{EasternWildTurkey}(x) \vee \text{OsceolaWildTurkey}(x) \vee \text{GouldsWildTurkey}(x) \vee \text{MerriamsWildTurkey}(x) \vee \text{RiograndeWildTurkey}(x) \vee \text{OcellatedWildTurkey}(x)))$
2. $\neg\text{EasternWildTurkey}(tom)$
3. $\neg\text{OsceolaWildTurkey}(tom))$
4. $\neg\text{GouldsWildTurkey}(tom)$
5. $\neg\text{MerriamsWildTurkey}(tom) \wedge \neg\text{RiograndeWildTurkey}(tom)$
6. $\text{WildTurkey}(tom)$

FOL conclusions -> Labels

A. $\text{OcellatedWildTurkey}(tom)$ -> True
B. $\text{EasternWildTurkey}(tom)$ -> False
C. $\text{WildTurkey}(joey)$ -> Unknown

[Han et al, 2024]

CAN WE TRAIN NLP MODELS TO REASON?

- They use Wikipedia as the source of their examples.

A FOLIO example based on the Wild Turkey Wikipedia page: https://en.wikipedia.org/wiki/Wild_turkey

NL premises

1. There are six types of wild turkeys: Eastern wild turkey, Osceola wild turkey, Gould's wild turkey, Merriam's wild turkey, Rio Grande wild turkey, and the Ocellated wild turkey.
2. Tom is not an Eastern wild turkey.
3. Tom is not an Osceola wild turkey.
4. Tom is also not a Gould's wild turkey.
5. Tom is neither a Merriam's wild turkey, nor a Rio Grande wild turkey.
6. Tom is a wild turkey.

NL Conclusions -> Labels

A. Tom is an Ocellated wild turkey. -> True
B. Tom is an Eastern wild turkey. -> False
C. Joey is a wild turkey. -> Unknown

FOL Premises

1. $\forall x(\text{WildTurkey}(x) \rightarrow (\text{EasternWildTurkey}(x) \vee \text{OsceolaWildTurkey}(x) \vee \text{GouldsWildTurkey}(x) \vee \text{MerriamsWildTurkey}(x) \vee \text{RiograndeWildTurkey}(x) \vee \text{OcellatedWildTurkey}(x)))$
2. $\neg\text{EasternWildTurkey}(tom)$
3. $\neg\text{OsceolaWildTurkey}(tom))$
4. $\neg\text{GouldsWildTurkey}(tom)$
5. $\neg\text{MerriamsWildTurkey}(tom) \wedge \neg\text{RiograndeWildTurkey}(tom)$
6. $\text{WildTurkey}(tom)$

FOL conclusions -> Labels

A. $\text{OcellatedWildTurkey}(tom)$ -> True
B. $\text{EasternWildTurkey}(tom)$ -> False
C. $\text{WildTurkey}(joey)$ -> Unknown

[Han et al, 2024]

CAN WE TRAIN NLP MODELS TO REASON?

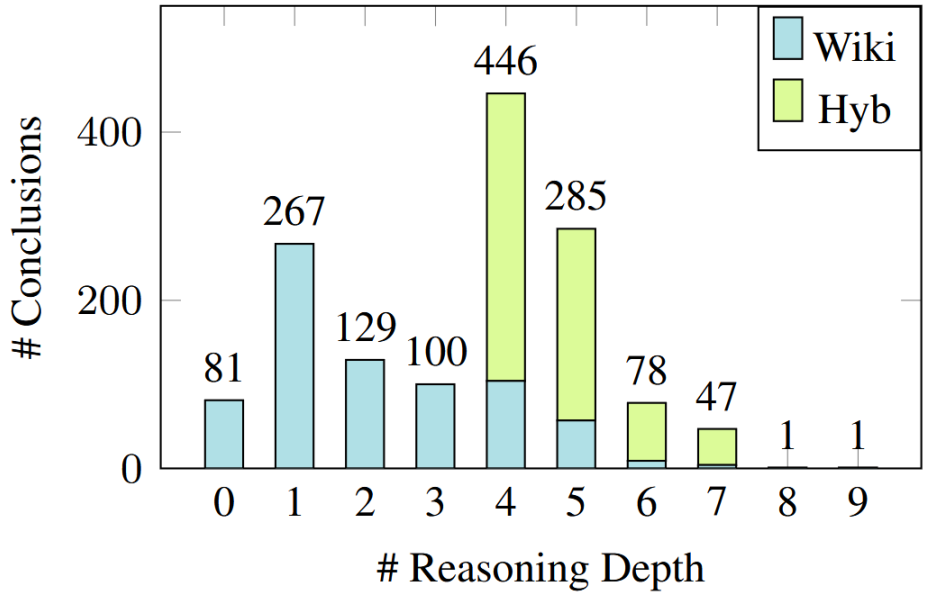
- They find NLP models perform decently.
  - Larger models perform better.
  - But there is still room for improvement.

Model	Size	Acc (%)
majority baseline	-	38.5%
random probability	-	33.3 %
Fully supervised fine-tune		
BERT-base	110M	56.8
BERT-large	340M	59.0
RoBERTa-base	110M	56.8
RoBERTa-large	340M	62.1
Flan-T5-Large	783M	**65.9**

[Han et al, 2024]

CAN WE TRAIN NLP MODELS TO REASON?

- They find NLP models perform decently.
  - Larger models perform better.
  - But there is still room for improvement.
  - The complexity of their examples is rather limited.



[Han et al, 2024]

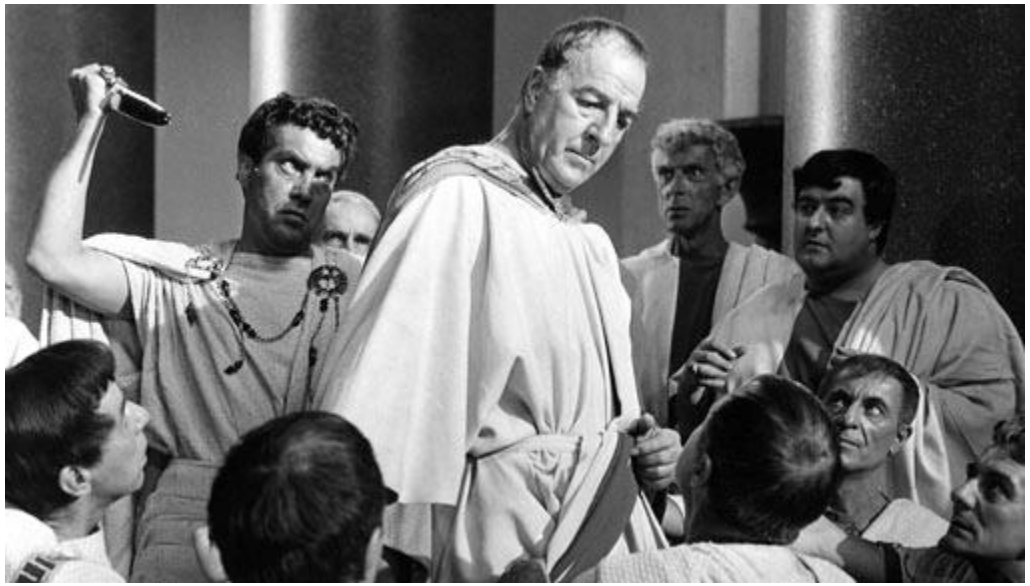
REPRESENTING THE MEANING OF SENTENCES

- How do we actually use logic to represent the meaning of natural language sentences?
- We'll work with the following running example:
  - 'Brutus stabs Caesar'
- Try first-order logic:
  - stab(brutus,caesar)

[Sayeed, Neo-Davidsonian semantics, 2012; F. H. Fuger, La muerte de Julio César, 1818]

REPRESENTING THE MEANING OF SENTENCES

- That was easy.
- Let's make it more complicated:
  - 'Brutus stabs Caesar with a knife'



[Sayeed, Neo-Davidsonian semantics, 2012; Shakespeare's Julius Caesar, dir. Joseph L. Mankiewicz, 1953]

REPRESENTING THE MEANING OF SENTENCES

- That was easy.
- Let's make it more complicated:
  - 'Brutus stabs Caesar with a knife'
- The knife is a participant in the action.
  - It's an instrument.
- Maybe: Make it an argument of the stab predicate?
  - stab(brutus,caesar,knife)
- Well, Brutus isn't stabbing Caesar using the *concept* of 'knife'.
  - Rather, 'a knife' is an instance of an object that has type knife.
  - ∃k(knife(k) & stab(brutus,caesar,k))

[Sayeed, Neo-Davidsonian semantics, 2012]

REPRESENTING THE MEANING OF SENTENCES

- ‘Brutus stabs Caesar in the agora’

[Sayeed, Neo-Davidsonian semantics, 2012; Gérôme, The Death of Caesar, 1867]

REPRESENTING THE MEANING OF SENTENCES

- ‘Brutus stabs Caesar in the agora’
- The agora is also a participant in the action.
- So maybe: `stab(brutus,caesar,agora)`?
- But the agora is not participating in the same way as the knife.
  - The agora is a location, whereas the knife is an instrument.
  - So the third argument of `stab` is ambiguous.
- Or add a fourth argument:
  - `stab(brutus,caesar,_,agora)`

[Sayeed, Neo-Davidsonian semantics, 2012]

REPRESENTING THE MEANING OF SENTENCES

- What if we split up the predicate?
  - `stab(brutus,caesar) & in(agora)`
  - This is a bit better.
- ‘`Brutus stabs Caesar with a knife in the agora.`’
  - `stab(brutus,caesar) & with(knife) & in(agora)`
  - `∃k(knife(k) & stab(brutus,caesar) & with(k) & in(agora))`
- ‘`Brutus stabs Caesar with a knife in the agora and twists it hard.`’
  - `∃k(knife(k) & stab(brutus,caesar) & with(k) & in(agora) & twists(brutus,k) & hard)`
  - It’s difficult to represent the adverb `hard`.

[Sayeed, Neo-Davidsonian semantics, 2012]

REPRESENTING THE MEANING OF SENTENCES

- The main difficulty is referring to predicates.
  - First-order logic does not allow us to refer to predicates.
  - In natural language, there is very rich variety of ways to refer to predicates.
    - Adverbial modifiers, relative clauses, etc.
- It's easy to represent modifiers of nouns in FOL.
  - 'some clever driver in America'
  - ∃x(driver(x) & clever(x) & location(x,america))
- Can we borrow this idea for verbs?
  - 'Bob drives cleverly in America'

[Sayeed, Neo-Davidsonian semantics, 2012]

DAVIDSON'S SOLUTION

- Davidson (1989) quoted in Maienborn (2010):

  "Adverbial modification is thus seen to be logically on a par with adjectival modification: what adverbial clauses modify is not verbs but the events that certain verbs introduce."

- A verb (e.g., '`stab`') is actually a description of an event.
  - An event is an object rather than a predicate.
  - Events can be variables (and we can quantify over events),
  - Just as any other object can be a variable, and we can quantify over objects.

[Sayeed, Neo-Davidsonian semantics, 2012]

DAVIDSON'S SOLUTION

- 'Brutus stabs Caesar with a knife in the agora.'
  - ∃k(knife(k)
    & ∃e(stab(e,brutus,caesar) & with(e,k) & location(e,agora))
)
- We now have a flexible way to refer to the event of stabbing.
- 'Brutus stabs Caesar with a knife in the agora and twists it hard.'
  - ∃k(knife(k)
    & ∃e(stab(e,brutus,caesar) & with(e,k) & location(e,agora)
      & ∃t(twist(t,brutus,k) & hard(t))
)
)

[Sayeed, Neo-Davidsonian semantics, 2012]

EVENT SEMANTICS

- This idea is called event semantics (or Davidsonian semantics; Davidson, 1967).
  - ‘Mark drove’ -> ∃d(drive(d,mark) & past(d))
  - ‘Mark drives quickly’ -> ∃d(drive(d,mark) & quickly(d))
- Parsons (1990) proposed an updated form where thematic roles (or semantic roles) are explicit.
- In ‘Brutus stabbed Caesar’ (active voice)
  - The subject is Brutus, and the object is Caesar
- In ‘Caesar was stabbed by Brutus’ (passive voice)
  - The subject is Caesar, and the object is Brutus
- Even if the grammatical roles have switched, the semantic roles have not!

SEMANTIC ROLES

- Every event has a set of semantic or thematic roles.
- The exact set of roles is debated, but here are some candidates:
  - Agent: the performer of the action (e.g., ‘Brutus stabs’)
  - Patient: the thing undergoing the action that changes state (e.g., ‘Brutus stabs Caesar’)
  - Theme: the thing undergoing the action but does not change state (e.g., ‘I gave them the food’)
  - Instrument
  - Location
  - Time
  - etc…

SEMANTIC ROLE LABELING

- Semantic role labeling is the task of identifying the semantic roles for a given sentence and verb.
  - Sometimes the verb is not specified.
- Example input:

  ‘The batter hit the ball yesterday.’
- Example output:

  ‘$[_{\text{agent}}$The batter$]$ hit $[_{\text{patient}}$the ball$]$ $[_{\text{time}}$yesterday$]$.’
- Not really full compositional semantic parsing,
  - But a “shallow” form of it.
  - Can be thought of as something between syntactic and semantic parsing.

EVENT SEMANTICS

- ‘Caesar was stabbed by Brutus’
  - Caesar is the patient (or theme)
  - Brutus is the agent
- In Davidsonian semantics, we would write this as:
  - ∃e.stab(e,brutus,caesar)
  - But how would we write the meaning of ‘Caesar was stabbed’?
- In neo-Davidsonian semantics, we separate the semantic roles explicitly:
  - ∃e(stab(e) & agent(e,brutus) & patient(e,caesar))
  - ∃e(stab(e) & arg1(e,brutus) & arg2(e,caesar))
  - ∃e(stab(e) & arg1(e)=brutus & arg2(e)=caesar)

LOGICAL FORMALISMS

- First-order logic is not just one monolithic logical formalism.
  - There are different ways we can use FOL to represent meaning.
  - Each with advantages and disadvantages.
- What makes a good logical formalism/meaning representation?
  - Coverage:
    - If there are two sentences with different meanings, they should have different logical forms.
    - If there are multiple readings/interpretations of the same sentence, there should be a logical form for each reading.

SEMANTIC AMBIGUITY

- We have seen ambiguity in syntax:
  - ‘Sally caught a butterfly with a net.’
  - ‘Sally caught a butterfly with a stripe.’
- We can also have ambiguity in semantics:
  - ‘The trophy didn’t fit in the suitcase because it’s too big.’
  - ‘The trophy didn’t fit in the suitcase because it’s too small.’
  - These sentences have the same syntactic structure.
  - Coreference resolution:
    - Does ‘it’ refer to the same thing as ‘trophy’?
    - Or to the same thing as ‘suitcase’?
- A good logical formalism will have 2 LFs for each sentence.

SEMANTIC AMBIGUITY

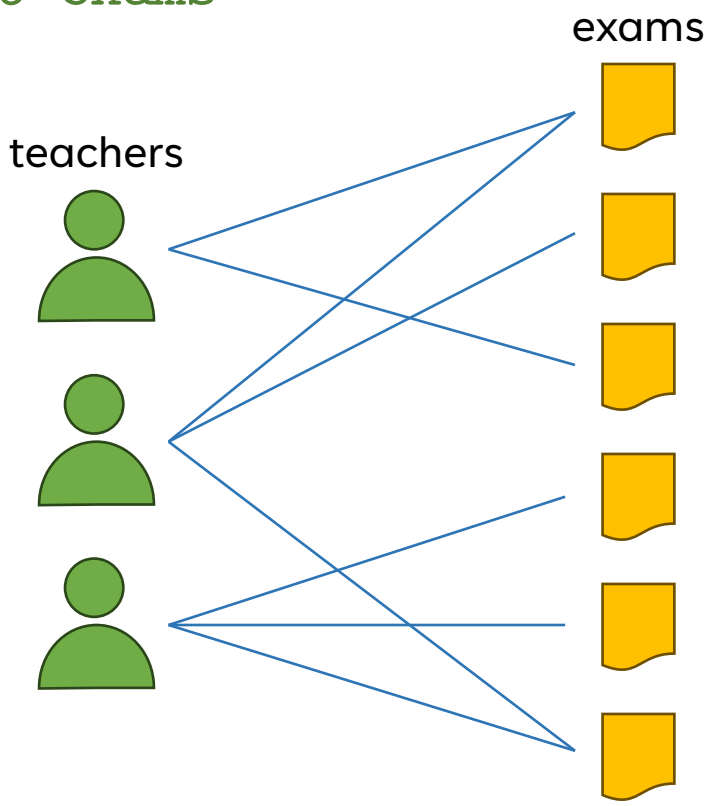
- Lexical ambiguity is another source of ambiguity in semantics:
  - ‘I went to the bank to withdraw cash.’
  - ‘I went to the bank to catch some fish.’
- The word ‘bank’ refers to the financial institution sense in the first sentence.
  - Whereas it refers to a riverbank in the second.
- A good logical formalism will produce two logical forms for each of the above two sentences:
  - 1 LF for the financial institution sense,
  - And 1 LF for the riverbank sense.
- The logical formalism itself has no background knowledge of the world.
  - Maybe there’s a possible world where you withdraw money near rivers.

SEMANTIC AMBIGUITY

- Quantifier scope ambiguity is another source of ambiguity in semantics:
  - ‘Every dog chases a cat.’
- There are two valid readings:
  - ∀d(dog(d) -> ∃c(cat(c) & chase(d,c)))
  - ∃c(cat(c) & ∀d(dog(d) -> chase(d,c)))
- The only difference is the order of the quantifiers,
  - But there is a stark difference in meaning.
- Negation scope ambiguity: ‘All that glitters is not gold.’
  - ∀g(glitter(g) -> ¬gold(g))
  - ¬∀g(glitter(g) -> gold(g))

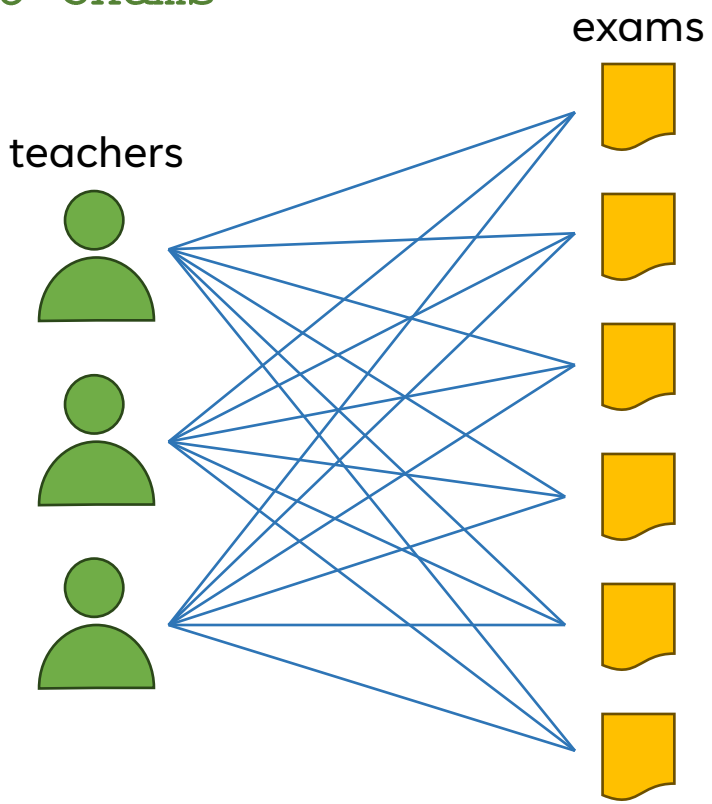
SEMANTIC AMBIGUITY

- More extreme example of quantifier scope ambiguity:
  - ‘3 teachers graded 6 exams’



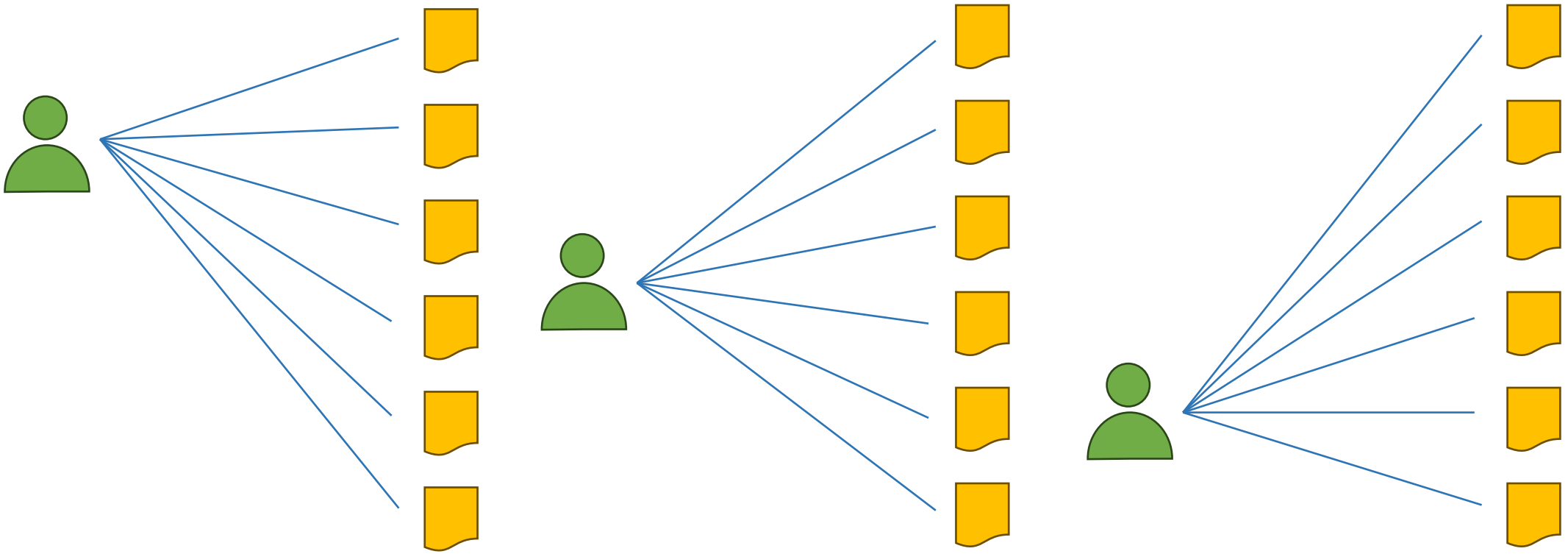
SEMANTIC AMBIGUITY

- More extreme example of quantifier scope ambiguity:
  - ‘3 teachers graded 6 exams’



SEMANTIC AMBIGUITY

- More extreme example of quantifier scope ambiguity:
  - ‘3 teachers graded 6 exams’



LOGICAL FORMALISMS

- What makes a good logical formalism/meaning representation?
  - Coverage
  - Language-independence:
    - This depends on the application.
    - If we wish to model translation from one natural language to another, Where the meaning is language-independent,
    - Ideally, two sentences in different languages but with the same meaning should map to the same logical form.

LOGICAL FORMALISMS

- What makes a good logical formalism/meaning representation?
  - Coverage
  - Language-independence
  - Uniqueness:
    - Two sentences with the same meaning should map to the same LF.
    - Not always needed.
  - Amenable for reasoning:
    - If we want to reason over the logical forms,
    - It would help to have a set of inference rules with which we can construct proofs.

LOGICAL FORMALISMS

- What makes a good logical formalism/meaning representation?
  - Coverage
  - Language-independence
  - Uniqueness
  - Amenable for reasoning
  - Easy to parse:
    - It shouldn't be too difficult to parse from natural language into LF.
    - If the logical form is more similar to the natural language utterance, semantic parsing is easier.
    - We will discuss semantic parsing in more detail later.

LOGICAL FORMALISMS

- What makes a good logical formalism/meaning representation?
  - Coverage
  - Language-independence
  - Uniqueness
  - Amenable for reasoning
  - Easy to parse
  - Compositional:
    - The logical form of larger phrases should be composed of the logical forms of subphrases.

COMPOSITIONALITY

- Compositionality is also relevant in syntactic analysis.
  - The syntax tree of a larger phrase is composed of the syntax trees of its constituents.
  - Grammars with recursion enable syntactic compositionality.
  - E.g., context-free grammars.
- Semantic compositionality:
  - ‘a barn’ -> ∃b.barn(b)
  - ‘a dog’ -> ∃d.dog(d)
  - ‘in a barn’ -> ∃b(barn(b) & location(_,b))
  - ‘a dog in a barn’ -> ∃d(dog(d) & ∃b(barn(b) & location(d,b)))
- Exceptions: Idioms such as ‘break a leg’, ‘on the other hand’, etc.

COMPOSITIONALITY

- Compositionality is very useful for efficient parsing.
  - We can parse a sentence by first parsing smaller phrases.
  - Enables dynamic programming approaches.
- Compositionality is also important for generalization.
  - We can predict the syntactic/semantic structure of larger phrases/sentences based on its smaller constituent phrases.
  - Compositional generalization:
    - If a model correctly "understands" some phrases/sentences,

      How well does it "understand" larger phrases/sentences that contain those phrases?

COMPOSITIONAL GENERALIZATION

- Kim and Linzen (2020) introduced a dataset called COGS to test the semantic compositional generalization ability of NLP models.
- Some examples from COGS:

TRAINING

[[The girl]] = $\iota x.\, girl'(x)$, [[The cat]] = $\boldsymbol{\iota x.\, cat'(x)}$, [[The boy]] = $\iota x.\, boy'(x)$

[[The cat loves the girl]] = $love'\left(\boldsymbol{\iota x.\, cat(x)}, \iota x.\, girl'(x)\right)$

[[The hedgehog sees the cat]] = $see'(\iota x.\, hedgehog'(x), \boldsymbol{\iota x.\, cat'(x)})$

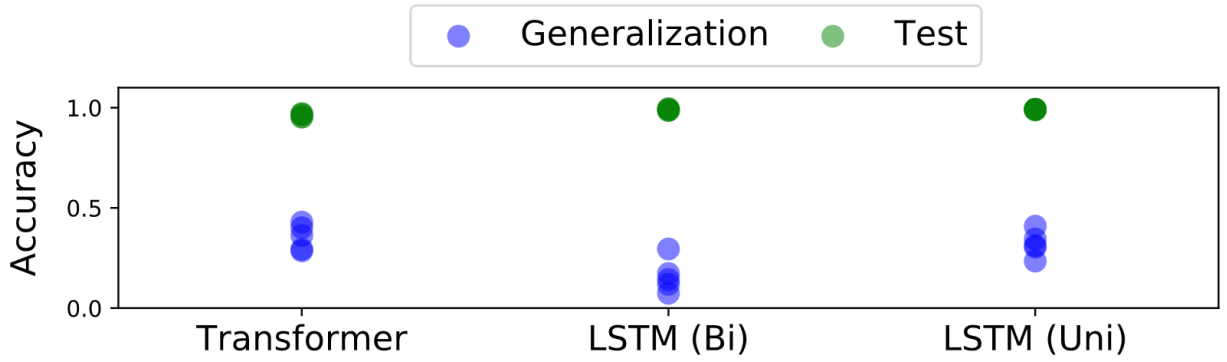
GENERALIZATION

[[The boy loves the hedgehog]] = $love'\left(\iota x.\, boy'(x), \boldsymbol{\iota x.\, hedgehog(x)}\right)$

COMPOSITIONAL GENERALIZATION

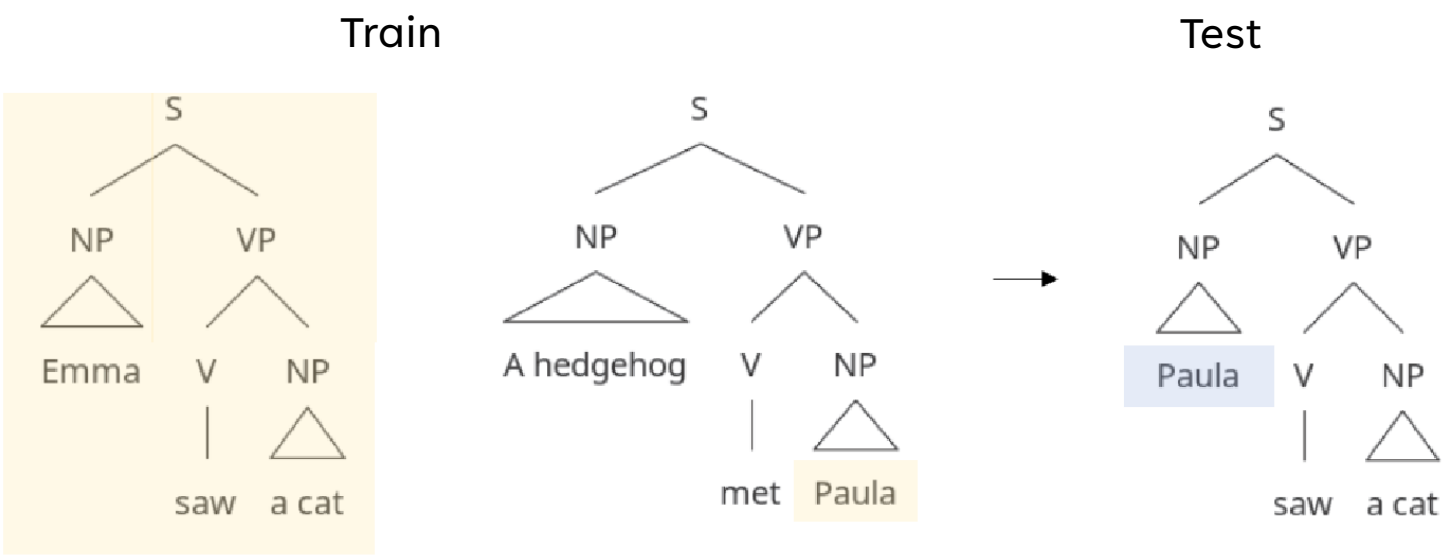
- Kim and Linzen (2020) introduced a dataset called COGS to test the semantic compositional generalization ability of NLP models.
- They evaluated transformers, unidirectional LSTMs, and bidirectional LSTMs.

Model	Dev.	Test	Gen.
Transformer	0.96	0.96	**0.35** ($\pm$ 0.06)
LSTM (Bi)	0.99	0.99	0.16 ($\pm$ 0.08)
LSTM (Uni)	0.99	0.99	0.32 ($\pm$ 0.06)



COMPOSITIONAL GENERALIZATION

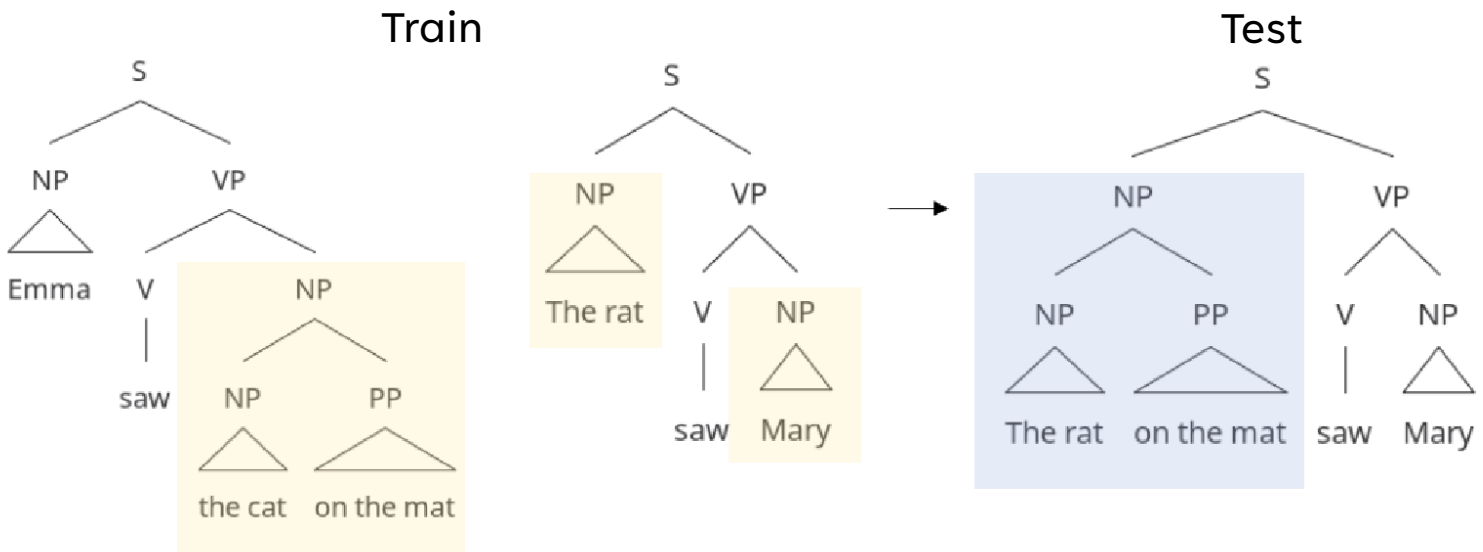
- They considered two types of generalization:
  - Lexical generalization: A word is used in a previously-unseen context.



[Kim and Linzen, 2020]

COMPOSITIONAL GENERALIZATION

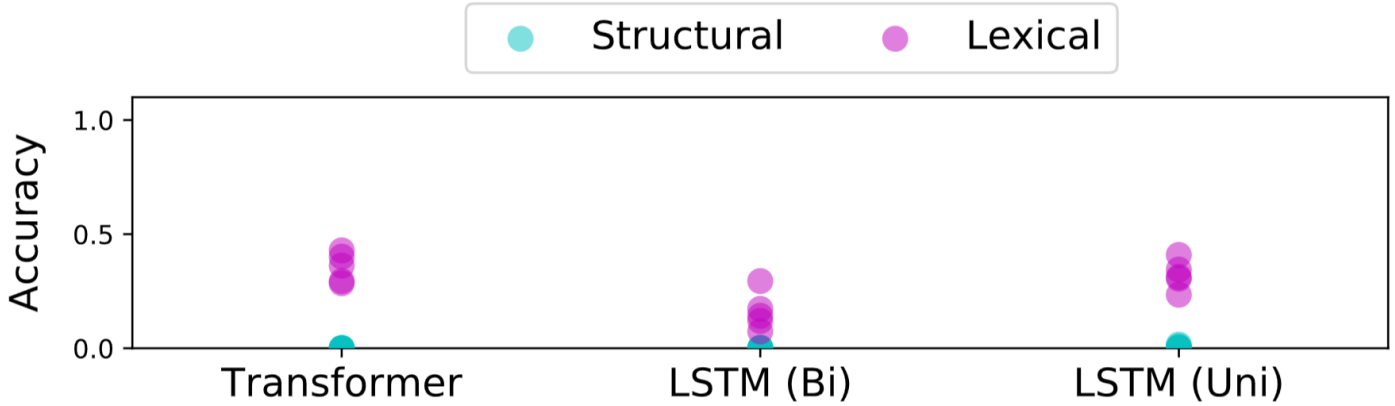
- They considered two types of generalization:
  - Lexical generalization: A word is used in a previously-unseen context.
  - Structural generalization: Familiar structures are combined into a novel larger structure.



[Kim and Linzen, 2020]

COMPOSITIONAL GENERALIZATION

- They considered two types of generalization:
  - Lexical generalization: A word is used in a previously-unseen context.
  - Structural generalization: Familiar structures are combined into a novel larger structure.



- They found that the tested models are incapable of structural generalization.
  - Newer models may do better?

[Kim and Linzen, 2020]

COMPOSITIONAL GENERALIZATION

- Compositionality is also a feature of reasoning.
- A proof can consist of simpler subproofs.
  - In order to find the full proof, an intelligent system must first find the subproofs,
  - Then combine them to produce the full proof.
- E.g., 2-hop question answering:

  `‘Who won the Master’s Tournament the year Justin Bieber was born?’`

- To answer this, you must prove:
  - The year Justin Bieber was born is 1994.
  - Jose Maria Olazabal won the Master’s Tournament in 1994.

LOGICAL FORMALISMS

- What makes a good logical formalism/meaning representation?
  - Coverage
  - Language-independence
  - Uniqueness
  - Amenable for reasoning
  - Easy to parse
  - Compositional
  - Human-readable:
    - Logical forms should be easy to understand by humans.
    - Ideally, they shouldn't require extensive special training (e.g., a university course on formal semantics).

QUESTIONS?