LEARNING COMPOSITIONALITY

Ignacio Cases with Christopher Potts
Stanford University

@ignaciocases

nlp.stanford.edu

github.com/stanfordnlp/CoreNLP

Understanding is needed

Compositional Semantics

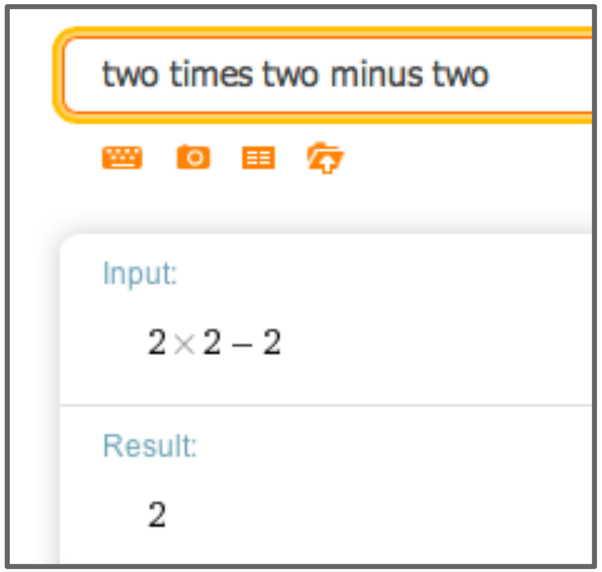
A growing proportion of queries require semantic interpretation.

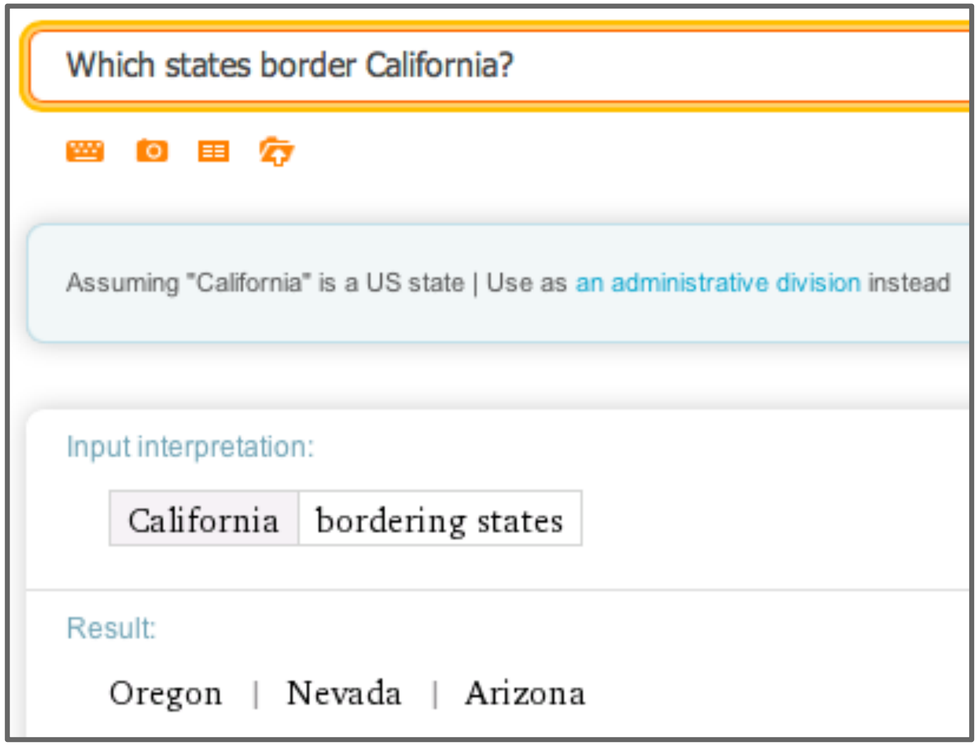
Conventional keyword-based retrieval does not suffice!

Slide adapted from Bill McCartney

Wolfram Alpha

Compositional Semantics

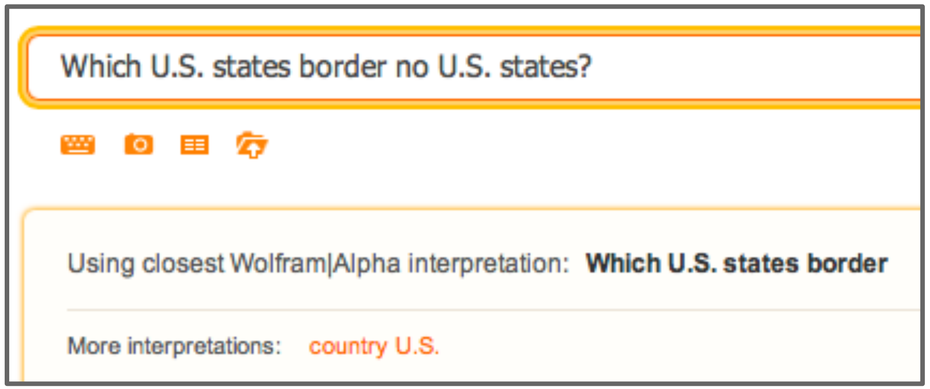




Slide adapted from Bill McCartney

Wolfram Alpha

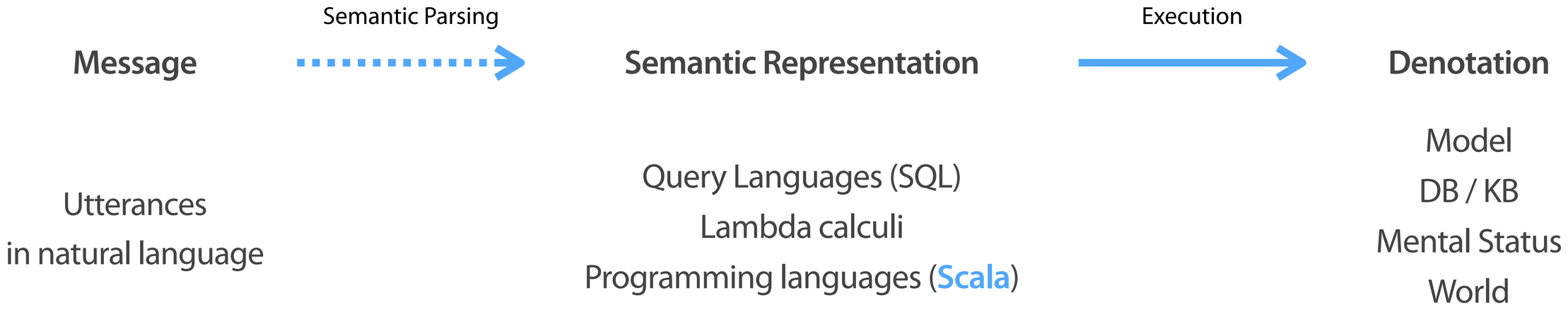
Compositional Semantics



Slide adapted from Bill McCartney

Meaning and Semantic Representation

Compositional Semantics



Meaning and Semantic Representation

Compositional Semantics

Semantic Parsing

Execution

Message	Semantic Representation	Denotation
one plus two	Add(1, 2)	
minus two plus three	Add(Neg(2), 3) Neg(Add(2, 3))	
crane	Noun(crane, bird) Noun(crane, machine)	

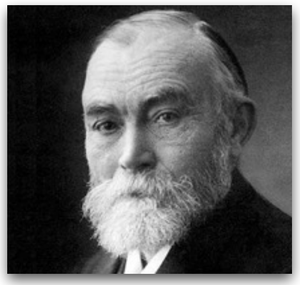
Meaning and Semantic Representation

Compositional Semantics

Message	Semantic Representation	Denotation
`two is less than three`	`LessThan(2,3)`	`T`
`the sum of one to four`	`(1 to 4).` `foldRight(zero)(Add(_,_))`	`10`

Compositionality Principle

Compositional Semantics



Gottlob Frege

Barbara Partee

"The meaning of a sentence is a function of the meanings of the parts and of the way they are syntactically combined."

Partee (1995)

Compositionality Principle

Compositional Semantics

old linguists and engineers

minus two plus three

Compositionality Principle

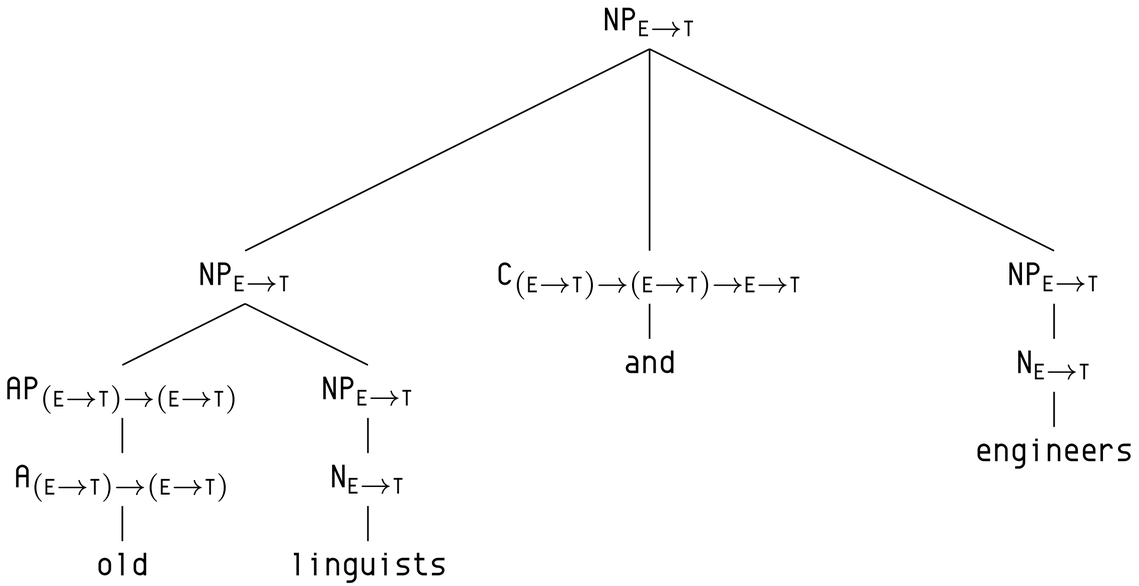
Compositional Semantics

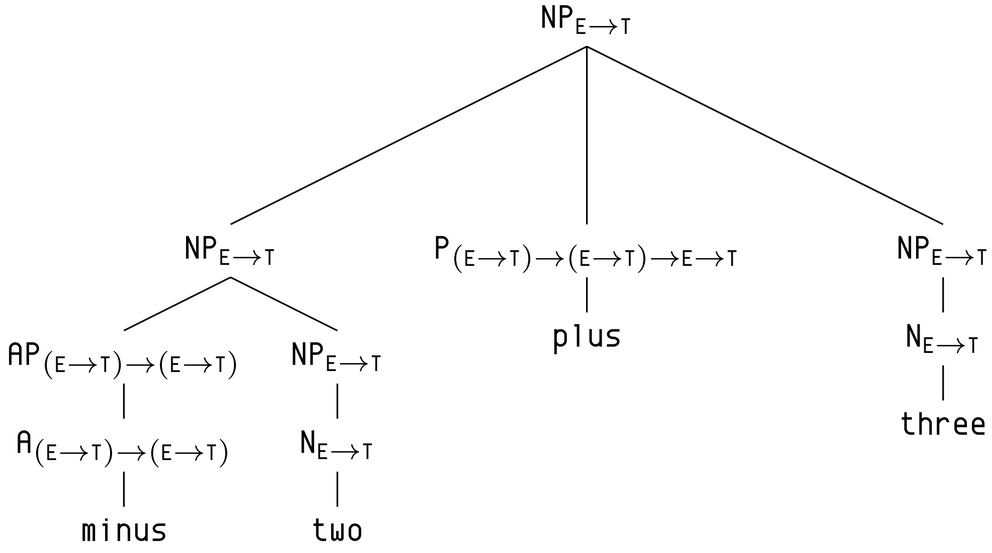
```
old linguists and engineers
```

```
minus two plus three
```

Compositionality Principle

Compositional Semantics



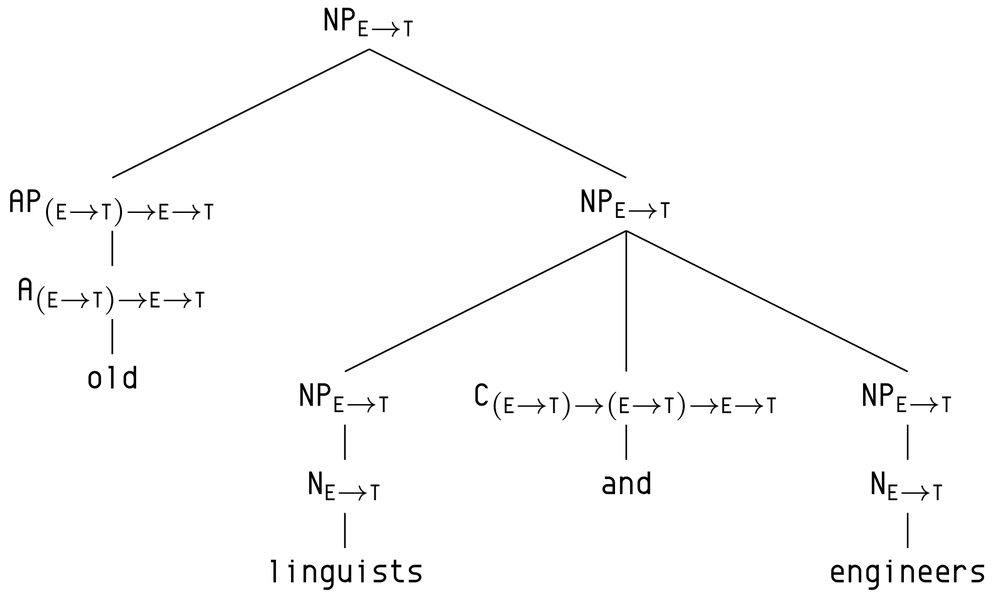


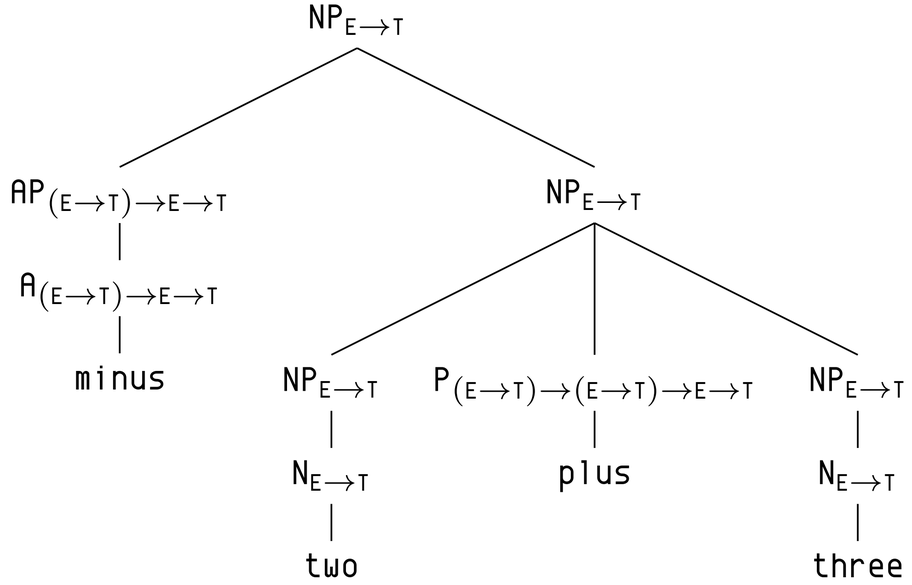
(old linguists) and engineers

(minus two) plus three

Compositionality Principle

Compositional Semantics





`old (linguists and engineers)`

`minus (two plus three)`

Learning Compositionality

Compositional Semantics

"compositionality characterizes the recursive nature of the linguistic ability required to generalize to a creative capacity, and learning details the conditions under which such an ability can be acquired from data."

Liang and Potts (2015: 356)

Learning Challenge

Learning Compositionality

Learning Dependency-Based Compositional Semantics

Percy Liang*
University of California, Berkeley

Michael I. Jordan**
University of California, Berkeley

Dan Klein†
University of California, Berkeley

Suppose we want to build a system that answers a natural language question by representing its semantics as a logical form and computing the answer given a structured database of facts. The core part of such a system is the semantic parser that maps questions to logical forms. Semantic parsers are typically trained from examples of questions annotated with their target logical forms, but this type of annotation is expensive.

Our goal is to instead learn a semantic parser from question–answer pairs, where the logical form is modeled as a latent variable. We develop a new semantic formalism, dependency-based compositional semantics (DCS) and define a log-linear distribution over DCS logical forms. The model parameters are estimated using a simple procedure that alternates between beam search and numerical optimization. On two standard semantic parsing benchmarks, we show that our system obtains comparable accuracies to even state-of-the-art systems that do require annotated logical forms.

1. Introduction

One of the major challenges in natural language processing (NLP) is building systems that both handle complex linguistic phenomena and require minimal human effort. The difficulty of achieving both criteria is particularly evident in training semantic parsers, where annotating linguistic expressions with their associated logical forms is expensive but until recently, seemingly unavoidable. Advances in learning latent-variable models, however, have made it possible to progressively reduce the amount of supervision

* Computer Science Division, University of California, Berkeley, CA 94720, USA. E-mail: pliang@cs.stanford.edu.
** Computer Science Division and Department of Statistics, University of California, Berkeley, CA 94720, USA. E-mail: jordan@cs.berkeley.edu.
† Computer Science Division, University of California, Berkeley, CA 94720, USA. E-mail: klein@cs.berkeley.edu.

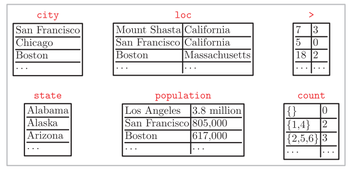
Submission received: 12 September 2011; revised submission received: 19 February 2012; accepted for publication: 18 April 2012.

doi:10.1162/COLI.a.00127

Percy Liang

Learning Challenge

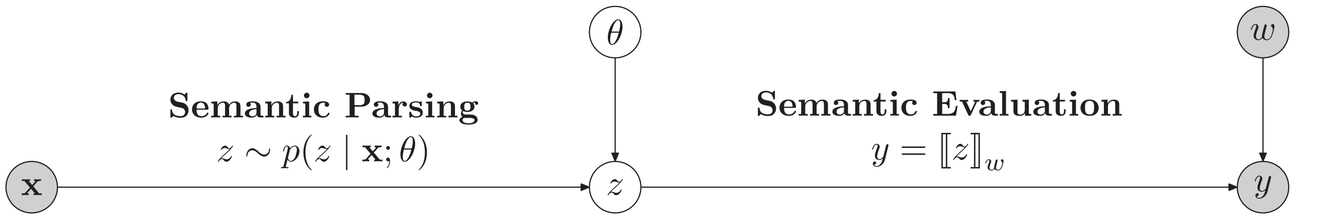
Learning Compositionality



$[0.3, -0.7, 4.5, 1.2, \dots]$

(parameters)

(world)

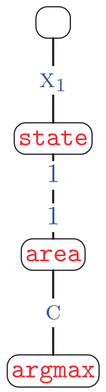


(utterance)

(logical form)

(answer)

state with the largest area



Alaska

Liang et al. (2013: 391)

Structured Prediction

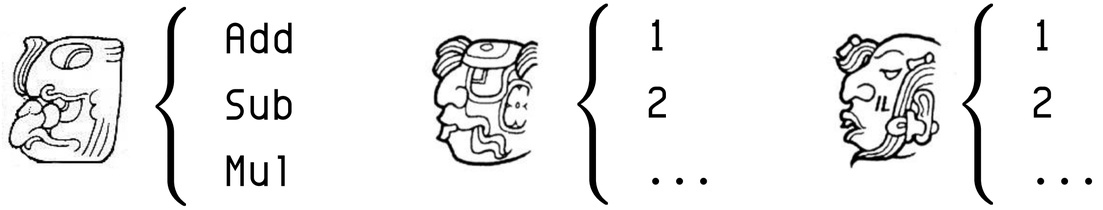
Learning Compositionality

Four pieces

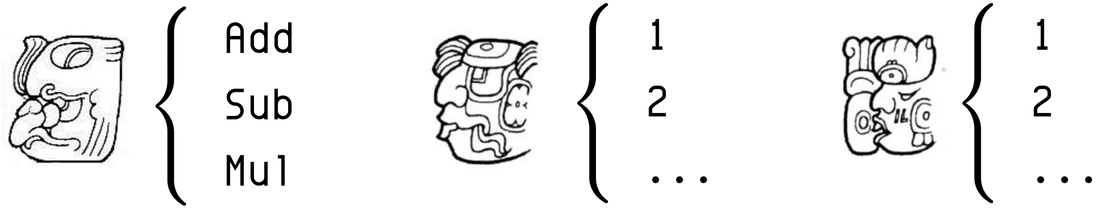
- initial grammar (refined during the learning)

Structured Prediction

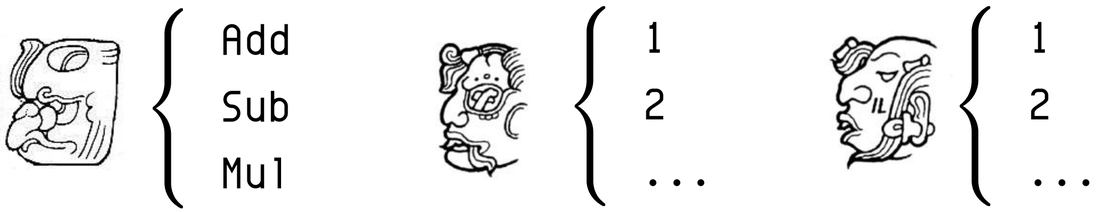
Learning Compositionality



5



6



7

Structured Prediction

Learning Compositionality



Add

4

1
5



Add

4

2
6



Add

6

1
7

Structured Prediction

Learning Compositionality

Four pieces

- initial grammar (refined during the learning)
- a feature representation of the data
- an objective function
- an algorithm for optimizing the objective function

Liang and Potts (2015)

Initial grammar

Synthesis Framework

```
lazy val expr: Parser[Expr] = (
     expr ~ "plus" ~ expr  ^^ { (e1, _, e2) => Add(e1, e2, "plus") }
   | expr ~ "plus" ~ expr  ^^ { (e1, _, e2) => Mul(e1, e2, "plus") }
   | expr ~ "plus" ~ expr  ^^ { (e1, _, e2) => Sub(e1, e2, "plus") }
   ...
   | term
 )

val numbers = List("one", "two", "three", … , "nine")

val terms = for {
  n <- numbers
  x <- 1 to 9
} yield (n ^^ { _ => IntLit(x, n) })

val term = terms.reduceLeft(_ | _)
```

Algebraic Data Types

Synthesis Framework

```
sealed trait Expr

trait BinExpr extends Expr
trait UnExpr extends Expr

case class Add(e1: Expr, e2: Expr, label: String) extends BinExpr
case class Sub(e1: Expr, e2: Expr, label: String) extends BinExpr
case class Mul(e1: Expr, e2: Expr, label: String) extends BinExpr
case class Neg(e: Expr, label: String) extends UnExpr

case class IntLit(i: Int, label: String) extends Expr
```

Denotations as Catamorphisms

Synthesis Framework

"compositionality outlines a recursive interpretation process in which the lexical items are listed as base cases and the recursive clauses define the modes of combination."

Liang and Potts (2015: 359)

"Fold is a generic transform for any algebraic data type"

Noel Welsh (2015)

Fold provides the appropriate abstraction over structural recursion

Featurization with Catamorphisms

Synthesis Framework

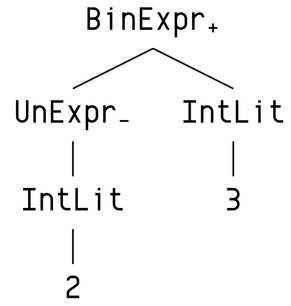
```
def foldt[A,B](f: Expr => Int, c: Int, g: Expr => (Int => Int => Int))(t: Expr): Int =
  t match {
    case IntLit(a, l) => f(IntLit(a, l))
    case u: UnExpr => g(u)(c)(foldt(f, c, g)(u.e))
    case b: BinExpr => g(b)(foldt(f, c, g)(b.e1))(foldt(f, c, g)(b.e2))
  }
```

Algebraic Data Types

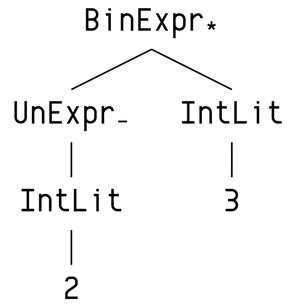
Synthesis Framework

```
minus two plus three
```

1

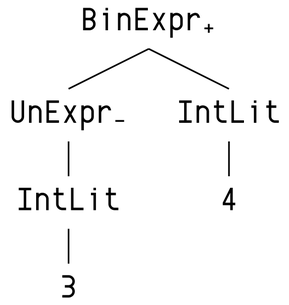


1

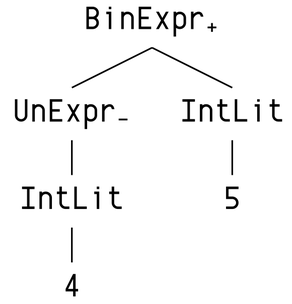


-6

...

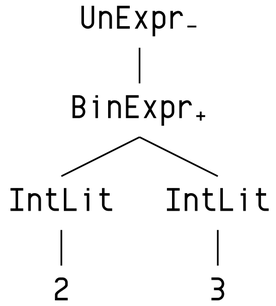


1

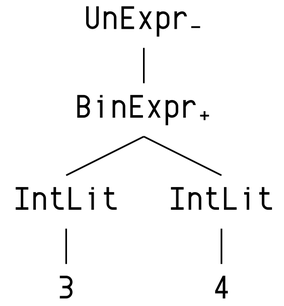


1

...



-5



-7

...

Polymorphic implementation of SGD

Synthesis Framework

```
def latentSGD[U,S,D](dataset: List[(U, S, D)], phi: Phi[U,S,D], numIter: Int,
   eta: Double, classes: U => List[S], transform: S => D,
   cost: (S, S) => Double): Weight[U,D] = {
   ...
   for (t <- 0 until numIter) {
     for ((x, ys, d) <- shuffledData) {
       val monadicClass = (z: U) => for {
         zd <- classes(z)
         if transform(zd) == d
       } yield zd
       // Get the best viable candidate given the current weights:
       val y = predict(x, phi, w, monadicClass, transform)
       // Get all (score, y') pairs:
       val scores = for(yAlt <- classes(x)) yield (score(x, yAlt, phi, w) + cost(y, yAlt), yAlt)
       // Get the maximal score
       // Get all the candidates with the max score and chose one randomly
       // Update the weights
       ...
     }
   }
   w
 }
```

Results

Synthesis Framework

```
INTERPRETIVE
================================================
 Feature function: Catamorphism
------------------------------------------------
 Learned feature weights
 Training:

   Input:      one plus one
   Gold:       1 + 1
   Prediction: 1 + 1
   Correct:    true

   Input:      one plus two
   Gold:       1 + 2
   Prediction: 1 + 2
   Correct:    true

   Input:      one plus three
   Gold:       1 + 3
   Prediction: 1 + 3
   Correct:    true
```

Results

Synthesis Framework

```
Test:

  Input:      minus three
  Gold:       -3
  Prediction: -3
  Correct:    true

  Input:      three plus two
  Gold:       3 + 2
  Prediction: 3 + 2
  Correct:    true

  Input:      minus four
  Gold:       -4
  Prediction: -1
  Correct:    false
```

CONTINUOUS REPRESENTATIONS

Distributional hypothesis

Continuous Representations

Firth (1957)

- "You shall know a word by the company it keeps."
- "the complete meaning of a word is always contextual, and no study of meaning apart from context can be taken seriously."

Harris (1954)

- "distributional statements can cover all of the material of a language without requiring support from other types of information."

Turney and Pantel (2010)

- "If units of text have similar vectors in a text frequency matrix, then they tend to have similar meanings."

Distributed Representations

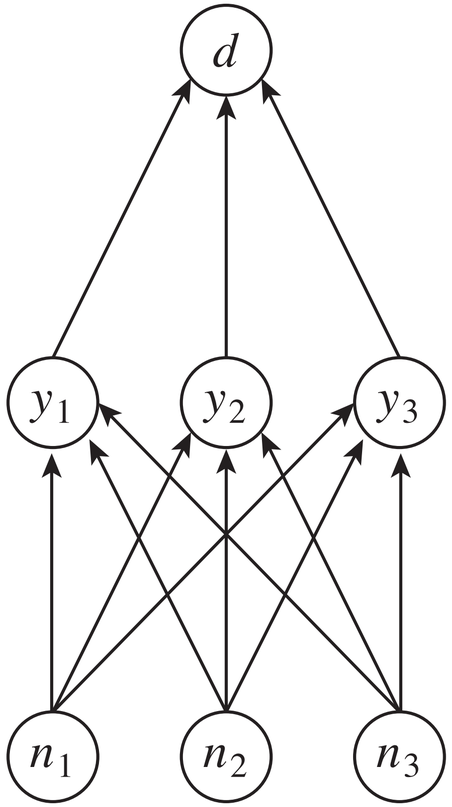
Continuous Representations

Syntax	Representation
N → *rollercoaster*	$[1.0\ 1.0\ 1.0]^\top$
N → *airplane*	$[1.0\ 0.0\ 1.0]^\top$
N → *website*	$[0.0\ 1.0\ 1.0]^\top$
N → *movie*	$[0.0\ 0.0\ 1.0]^\top$
A → *unpredictable*	$\begin{bmatrix} -6.3 & -6.3 & 2.5 \\ -4.4 & -4.4 & 6.5 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
N → A N	$\sigma(\ulcorner \mathrm{A} \urcorner\ \ulcorner \mathrm{N} \urcorner)$

Liang and Potts (2015)

Distributed Representations

Continuous Representations



$$d = [\![y]\!] = \sigma(\delta y)$$

$$\delta = [9.0 \;\; -8.8 \;\; 5.6]$$

$$y = \sigma(\ulcorner \mathrm{A} \urcorner \; \ulcorner \mathrm{N} \urcorner)$$

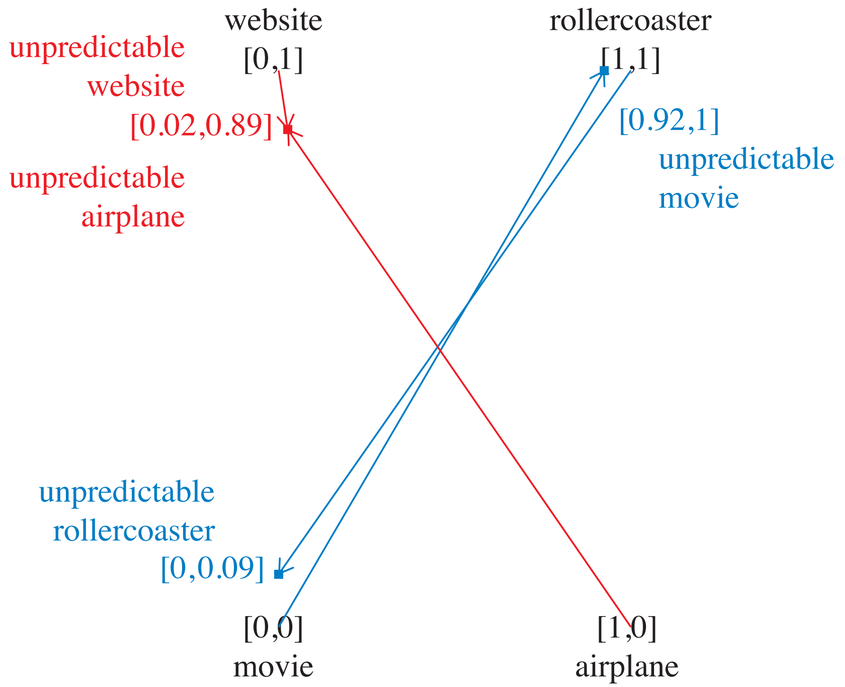
$$\ulcorner \mathrm{A} \urcorner$$

$$\ulcorner \mathrm{N} \urcorner$$

Liang and Potts (2015)

Distributed Representations

Continuous Representations



Liang and Potts (2015)

Architectures

Continuous Representations

Different architectures for composing distributed representations

- Algebraic composition

Hermann (2014)

Architectures

Continuous Representations

Different architectures for composing distributed representations

- Algebraic composition

Operation	Function
Additive	$\mathbf{p} = \alpha\mathbf{u} + \beta\mathbf{v}$
General Additive	$\mathbf{p} = A\mathbf{u} + B\mathbf{v}$
Multiplicative	$\mathbf{p} = \mathbf{u} \odot \mathbf{v}$
Tensor	$\mathbf{p} = \mathbf{u} \otimes \mathbf{v}$
Dilation	$\mathbf{p} = (\mathbf{u} \cdot \mathbf{u})\,\mathbf{v} + (\lambda - 1)\,(\mathbf{u} \cdot \mathbf{v})\,\mathbf{u}$

Hermann (2014)

Architectures

Continuous Representations

Different architectures for composing distributed representations

- Algebraic composition
- Parameterized composition
  - Coecke et al. 2010

Hermann (2014)

Architectures

Continuous Representations

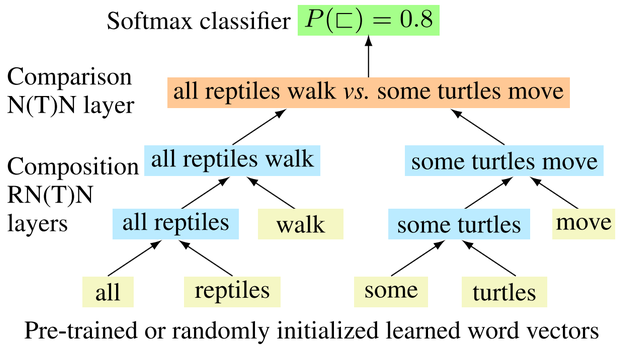
Different architectures for composing distributed representations

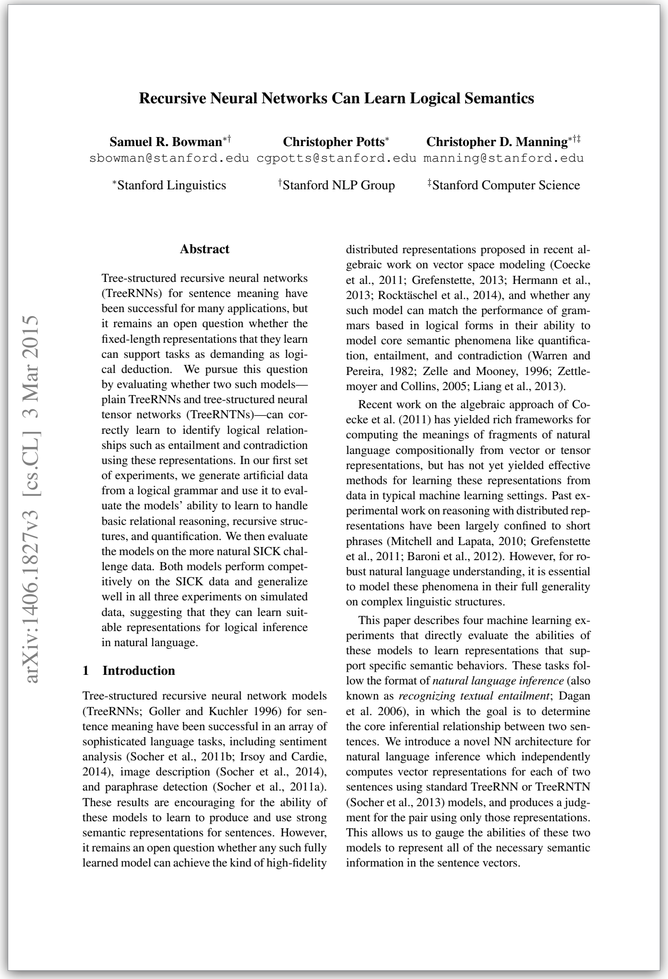
- Algebraic composition
- Parameterized composition
- Recursive composition
  - Recursive Neural Networks
  - Recurrent Neural Networks
  - Long-Short Term Memory (LSTM)
  - Convolutional Neural Networks

Hermann (2014)

Neural Network Architectures for Semantics

Continuous Representations





Recursive Neural Networks Can Learn Logical Semantics

Samuel R. Bowman[*†] **Christopher Potts**[*] **Christopher D. Manning**[*‡†]
sbowman@stanford.edu cgpotts@stanford.edu manning@stanford.edu

[*]Stanford Linguistics [†]Stanford NLP Group [‡]Stanford Computer Science

Abstract

Tree-structured recursive neural networks (TreeRNNs) for sentence meaning have been successful for many applications, but it remains an open question whether the fixed-length representations that they learn can support tasks as demanding as logical deduction. We pursue this question by evaluating whether two such models—plain TreeRNNs and tree-structured neural tensor networks (TreeRNTNs)—can correctly learn to identify logical relationships such as entailment and contradiction using these representations. In our first set of experiments, we generate artificial data from a logical grammar and use it to evaluate the models' ability to learn to handle basic relational reasoning, recursive structures, and quantification. We then evaluate the models on the more natural SICK challenge data. Both models perform competitively on the SICK data and generalize well in all three experiments on simulated data, suggesting that they can learn suitable representations for logical inference in natural language.

1 Introduction

Tree-structured recursive neural network models (TreeRNNs; Goller and Kuchler 1996) for sentence meaning have been successful in an array of sophisticated language tasks, including sentiment analysis (Socher et al., 2011b; Irsoy and Cardie, 2014), image description (Socher et al., 2014), and paraphrase detection (Socher et al., 2011a). These results are encouraging for the ability of these models to learn to produce and use strong semantic representations for sentences. However, it remains an open question whether any such fully learned model can achieve the kind of high-fidelity distributed representations proposed in recent algebraic work on vector space modeling (Coecke et al., 2011; Grefenstette, 2013; Hermann et al., 2013; Rocktäschel et al., 2014), and whether any such model can match the performance of grammars based in logical forms in their ability to model core semantic phenomena like quantification, entailment, and contradiction (Warren and Pereira, 1982; Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Liang et al., 2013).

Recent work on the algebraic approach of Coecke et al. (2011) has yielded rich frameworks for computing the meanings of fragments of natural language compositionally from vector or tensor representations, but has not yet yielded effective methods for learning these representations from data in typical machine learning settings. Past experimental work on reasoning with distributed representations have been largely confined to short phrases (Mitchell and Lapata, 2010; Grefenstette et al., 2011; Baroni et al., 2012). However, for robust natural language understanding, it is essential to model these phenomena in their full generality on complex linguistic structures.

This paper describes four machine learning experiments that directly evaluate the abilities of these models to learn representations that support specific semantic behaviors. These tasks follow the format of *natural language inference* (also known as *recognizing textual entailment*; Dagan et al. 2006), in which the goal is to determine the core inferential relationship between two sentences. We introduce a novel NN architecture for natural language inference which independently computes vector representations for each of two sentences using standard TreeRNN or TreeRNTN (Socher et al., 2013) models, and produces a judgment for the pair using only those representations. This allows us to gauge the abilities of these two models to represent all of the necessary semantic information in the sentence vectors.

Sam Bowman

Thanks!