intel®

AI DEVCON 2018

AI DEVCON 2018

SEMANTIC PARSING: NATURAL LANGUAGE UNDERSTANDING IN PYTHON

Milind S. Pandit

Machine Learning Solutions Manager, AI Products Group

THINGS TO KEEP IN MIND

1. You'll get access to the information covered in this session after the conference

2. Visit the Intel® AI Academy for additional resources, training materials and videos related to today's presentation.
software.intel.com/AI

3. Try the Intel Distribution for Python! (https://software.intel.com/en-us/distribution-for-python)

4. Check out more examples of Intel AI/Movidius NCS/Intel AI DevCloud in action on DevMesh – Intel's Developer Network
https://devmesh.intel.com/

REFERENCES

- Stanford CS 224U: Natural Language Understanding
- Liang, Percy and Potts, Christopher, Bringing machine learning and compositional semantics together. *Annual Review of Linguistics* 1(1): 355–376, 2014.
- Original SippyCup Github Repository
- Fork for this class: https://github.com/mspandit/sippycup

SEMANTIC PARSING

A computation which takes a linguistic expression and returns as output a structured, machine-interpretable representation of its meaning, known as the *semantic representation*

https://github.com/mspandit/sippycup

EXAMPLE: QUESTION ANSWERING APPLICATION

"How tall is Obama?"

→

`(/person/height /m/02mjmr)`

EXAMPLE: QUESTION ANSWERING APPLICATION

Linguistic Expression
"How tall is Obama?"
→ Semantic Parser
→ Semantic Representation
(/person/height /m/02mjmr)
→ Executor
→ Denotation
1.85 m

https://github.com/mspandit/sippycup

WHY SEMANTIC PARSING IS HARD

- Multiple linguistic expressions can have the same meaning
  - Example: “nyc population,” “How many people live in New York City?”
  - Canonicalization: Same meaning → Same semantic representation
- A single linguistic expression can have multiple meanings—depending on the context
  - Example: “How big is New York?” (area, population) X (city, state)
  - Ambiguity resolution: Different meanings → Different semantic representations

WHY SEMANTIC PARSING IS HARD

- Linguistic expressions can be messy with typos, misspellings, loose syntax: “where r u”
- Internationalization compounds the problem
- Scale of the problem demands machine learning

NATURAL LANGUAGE ARITHMETIC

THE PROBLEM

- Interpret natural language arithmetic expressions
  - "one plus one"
  - "minus three minus two" (lexical ambiguity)
  - "three plus three minus two"
  - "two times two plus three” (syntactic ambiguity)
- Small, closed vocabulary
- Limited variety of syntactic structures

https://github.com/mspandit/sippycup

intel AI

SEMANTIC REPRESENTATION: BINARY EXPRESSION TREES

one plus one	`('+', 1, 1)`
minus three minus two	`('-', ('~', 3), 2)`
three plus three minus two	`('-', ('+', 3, 3), 2)`
two times two plus three	`('+', ('*', 2, 2), 3)`

executor.py

CONSTITUENCY STRUCTURE

How we group words into larger and larger phrases.

SYNTACTIC PARSING

Build a tree structure (a *parse*) over the input which describes its constituency structure. Assign *categories* to each word and phrase.

https://github.com/mspandit/sippycup

EXAMPLE INPUT AND DENOTATION

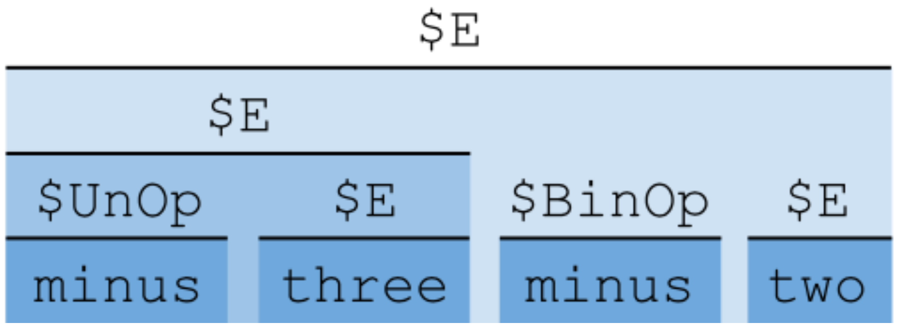
minus three minus two			-5
minus three minus two			-1

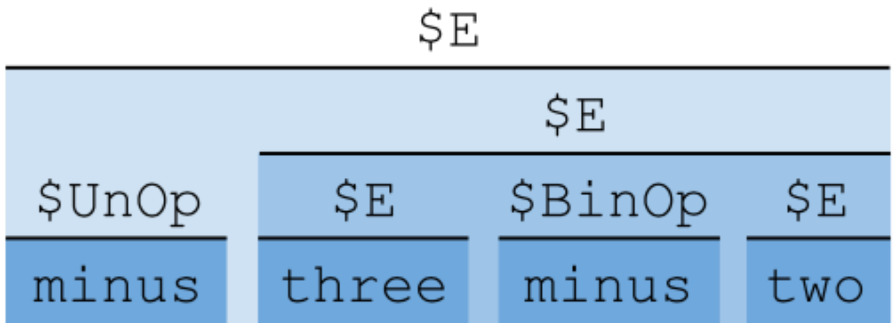
EXAMPLE WITH WORDS GROUPED

minus three minus two	((minus three) minus two)		-5
minus three minus two	(minus (three minus two))		-1

EXAMPLE WITH CATEGORIES ASSIGNED

minus three minus two	((minus three) minus two)	($E ($E ($UnOp minus) ($E three)) ($BinOp minus) ($E two))	-5
minus three minus two	(minus (three minus two))	($E ($UnOp minus) ($E ($E three) ($BinOp minus) ($E two)))	-1

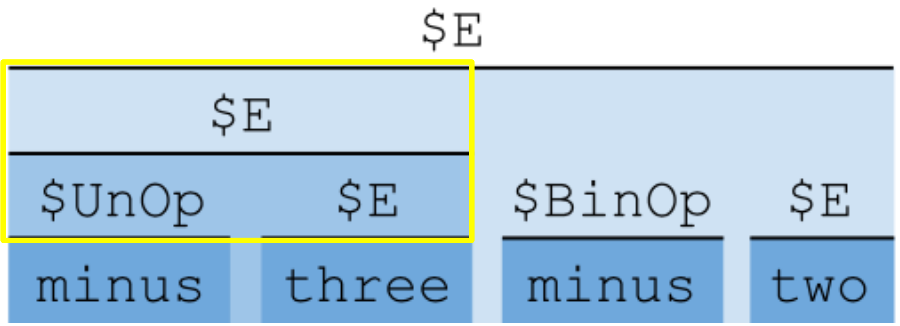


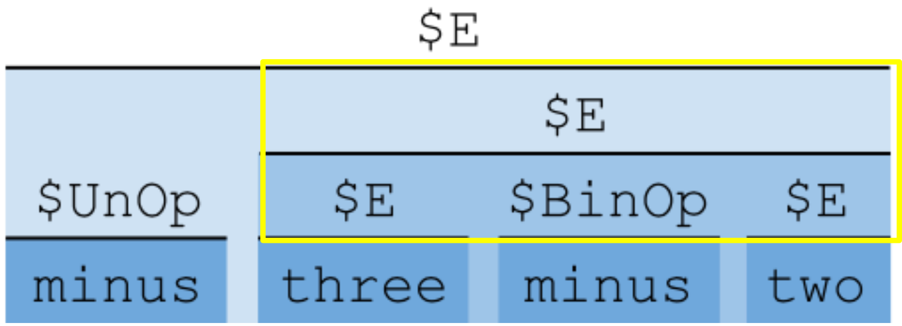


Category	Definition
$E	Expression
$UnOp	Unary Operator
$BinOp	Binary Operator

EXAMPLE WITH LOCAL SUBTREES HIGHLIGHTED

minus three minus two	((minus three) minus two)	($E ($E ($UnOp minus) ($E three)) ($BinOp minus) ($E two))	-5
minus three minus two	(minus (three minus two))	($E ($UnOp minus) ($E ($E three) ($BinOp minus) ($E two)))	-1





Category	Definition
$E	Expression
$UnOp	Unary Operator
$BinOp	Binary Operator

PARTIAL CONTEXT FREE GRAMMAR RULES

Left Hand Side	Right Hand Side
$E	two
$E	three
$UnOp	minus
$BinOp	minus
$E	$UnOp $E
$E	$E $BinOp $E

https://github.com/mspandit/sippycup

COMPLETE CONTEXT FREE GRAMMAR RULES

Left Hand Side	Right Hand Side
`$E`	`one`
`$E`	`two`
`$E`	`three`
`$E`	`four`
`$UnOp`	`minus`
`$BinOp`	`minus`
`$BinOp`	`plus`
`$BinOp`	`times`
`$E`	`$UnOp $E`
`$E`	`$E $BinOp $E`

https://github.com/mspandit/sippycup

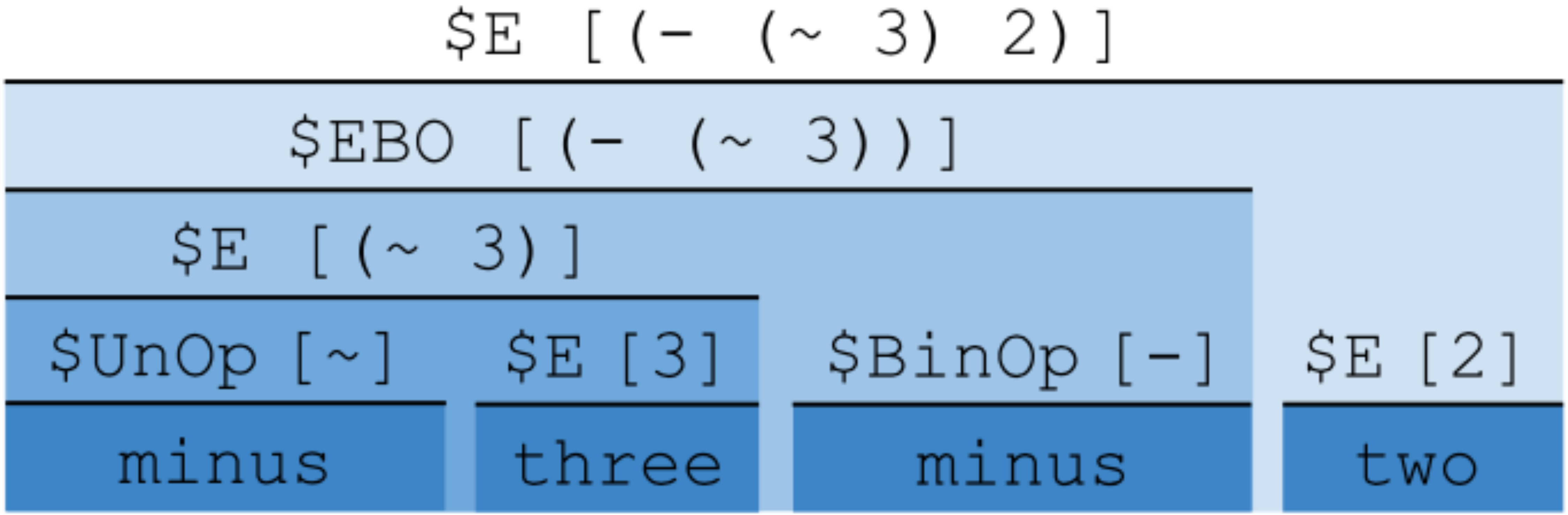
CHOMSKY NORMAL FORM (BINARIZED) CFG RULES

Left Hand Side	Right Hand Side
`$E`	`one`
`$E`	`two`
`$E`	`three`
`$E`	`four`
`$UnOp`	`minus`
`$BinOp`	`minus`
`$BinOp`	`plus`
`$BinOp`	`times`
`$E`	`$UnOp $E`
`$EBO`	**`$E $BinOp`**
`$E`	**`$EBO $E`**

https://github.com/mspandit/sippycup

unit1_tests.py

SEMANTICS



https://github.com/mspandit/sippycup

THE PRINCIPLE OF COMPOSITIONALITY

The meaning of a compound expression is a function of the meanings of its parts and the manner of their combination.

unit1_tests.py

STATUS

✅ In every example, we produced some correct parse

❌ In three examples, the parse at position 0 was incorrect

👉 **Conclusion: Rank candidate parses so that correct parses are likely to appear higher in the list.**

LINEAR SCORING FUNCTION

- Define multiple *feature functions* $\phi_i(p)$, each taking a parse p as input and returning a real number as output.
- Store a *weight* w_i for each feature function.
- For parse p:
- $score(p) = \sum_i w_i \cdot \phi_i(p)$

unit1_tests.py

LINEAR SCORING FUNCTION

- Define multiple *feature functions* $\phi_i(p)$, each taking a parse p as input and returning a real number as output.
- Store a *weight* w_i for each feature function.
- For parse p:
- $score(p) = \sum_i w_i \cdot \phi_i(p)$
- **What if there are many features? Learn weights from training data!**

unit1_tests.py

NATURAL LANGUAGE ARITHMETIC—SUMMARY

- Grammar with rules in Chomsky Normal (Binarized) Form
- Semantic representation derived from syntactic parses
- Feature functions for parses
- Machine learning of feature weights from semantics or denotation
- Performance improvement on ranking parses

TRAVEL QUERIES

THE PROBLEM

- Interpret natural language travel queries
  - “birmingham al distance from indianapolish in” (misspelling)
  - “directions from washington to canada” (ambiguity: which Washington?)
  - “discount travel flights to Austin texas”
- Much larger vocabulary, potentially unbounded
- Large variety of syntactic structures
- Accommodate misspellings, bad syntax
- Flat—not recursive nested—semantic structure: destination, origin, mode, etc.

DATASET

- Pass, Greg; Chowdhury, Abdur; Torgeson, Cayley; *A Picture of Search*
- Start: ~10 million unique search queries issued by ~650 thousand AOL users in 2006
- Selected queries containing one of the 600 locations named in Geobase (1M queries).
- Selected queries containing "from" or "to" (23K queries).
- Selected queries containing one of about 60 travel terms, or containing both "from" and "to" (6,588 queries).
- Many misspellings.

SEMANTIC REPRESENTATION: NESTED KEY-VALUE PAIRS

driving directions to Willamsburg, VA	{'domain': 'travel', 'type': 'directions', 'mode': 'car', 'destination': {'id': 4793846, 'name': 'Williamsburg, VA, US'}}
travel time by bus from Atlantic City to NYC	{'domain': 'travel', 'type': 'duration', 'mode': 'bus', 'origin': {'id': 4500546, 'name': 'Atlantic City, NJ, US'}, 'destination': {'id': 5128581, 'name': 'New York City, NY, US'}}
airfare from Newark to Charleston, SC	{'domain': 'travel', 'type': 'cost', 'mode': 'air', 'origin': {'id': 5101798, 'name': 'Newark, NJ, US'}, 'destination': {'id': 4574324, 'name': 'Charleston, SC, US'}}

- Resolves ambiguity and canonicalizes
- (No executor in this domain)

TRAINING DATA

Examples (travel_examples.py)

- travel boston to fr. myers fla
- how do i get from tulsa oklahoma to atlantic city. new jersey by air
- airbus from boston to europe
- cheap tickets to south carolina
- birmingham al distance from indianapolish in
- transportation to the philadelphia airport
- one day cruise from fort lauderdale florida
- directions from washington to canada
- flights from portland or to seattle wa
- honeymoon trip to hawaii

Roles

- Destination
- Origin
- Mode
- Type of information sought
- “Optional” words
- **Ordering of phrases isn’t important**

PHRASE BAG GRAMMAR

- Query elements can appear in any order
  - Travel locations (to, from)
  - Travel arguments (mode, trigger, request type)
- Optionals can appear anywhere
- Annotators: modules for assigning categories and semantics to specific types of phrases
- Unary compositional rules
- N-ary rules
  - Rule('$City', 'new york city')
  - Rule('$RouteQuery', '$FromLocation $ToLocation $TravelMode')
  - Optionals

https://github.com/mspandit/sippycup

intel AI

unit2_tests.py

TRAVEL QUERIES—SUMMARY

- Larger, more realistic dataset
- Annotators to “automate” rule definitions
- Rules for phrase-bag grammar with optionals
- Parsing with n-ary rules (CNF not required)

GEOGRAPHY QUERIES

THE PROBLEM

- "which states border texas?"
- "how many states border the largest state?"
- "what is the size of the capital of texas?"

- Large vocabulary
- Lexical and syntactic ambiguity
- Adhere to conventional rules for spelling and syntax
- Semantics with arbitrarily complex compositional structure
- **Isomorphic with other domains!**

https://github.com/mspandit/sippycup

DATASET

- Questions and answers
- Geo880 corpus: http://www.cs.utexas.edu/users/ml/geo.html
- “Standard evaluation” for semantic parsing systems
- (Not representative of web search queries)

geo880.py

KNOWLEDGE BASE

- Small knowledge base covering Geo880 queries
- states: capital, area, population, major cities, neighboring states, highest and lowest points and elevations
- cities: containing state and population
- rivers: length and states traversed
- mountains: containing state and height
- roads: states traversed
- lakes: area, states traversed

https://github.com/mspandit/sippycup

geobase.py

SEMANTIC REPRESENTATION: QUERIES FOR GRAPH-STRUCTURED KNOWLEDGE BASE

capital of texas	('/state/texas', 'capital')
rivers that traverse utah	('.and', 'river', ('traverses', '/state/utah'))
tallest mountain	('.argmax', 'height', 'mountain')

https://github.com/mspandit/sippycup

graph_kb.py

graph_kb_tests.py

unit3_tests.py

CURRENT STATE

- In cases where we put the wrong parse at the top, the top parse had nonsensical semantics with an empty denotation.

👉 Downweight parses with empty denotations

unit3_tests.py

GEOGRAPHY QUERIES—SUMMARY

- Semantic representation: queries for graph-structured knowledge base

SO… WHAT'S NEXT?

1. Visit the Intel® AI Academy for additional resources, training materials and videos related to today's presentation. software.intel.com/AI

2. Try the Intel Distribution for Python! (https://software.intel.com/en-us/distribution-for-python)

3. Build useful chatbots and voice interfaces using semantic parsers!

4. Check out more examples of Intel AI/Movidius NCS/Intel AI DevCloud in action on Intel's Developer Network https://devmesh.intel.com/

DISCLAIMER

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with your system manufacturer or retailer or learn more at https://www.intel.com/.

Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit http://www.intel.com/benchmarks.

Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether referenced data are accurate.

intel® AI