Chatbots from First Principles

Data Day Seattle

October 20, 2017

Jonathan Mugan

@jmugan

Talk outline

What conversation is

The current state of chatbots

- Purposeless mimicry agents
- Intention-based agents
- Conversational agents

Other plumbing

Additional resources

Conclusion

Talk outline

What conversation is

The current state of chatbots

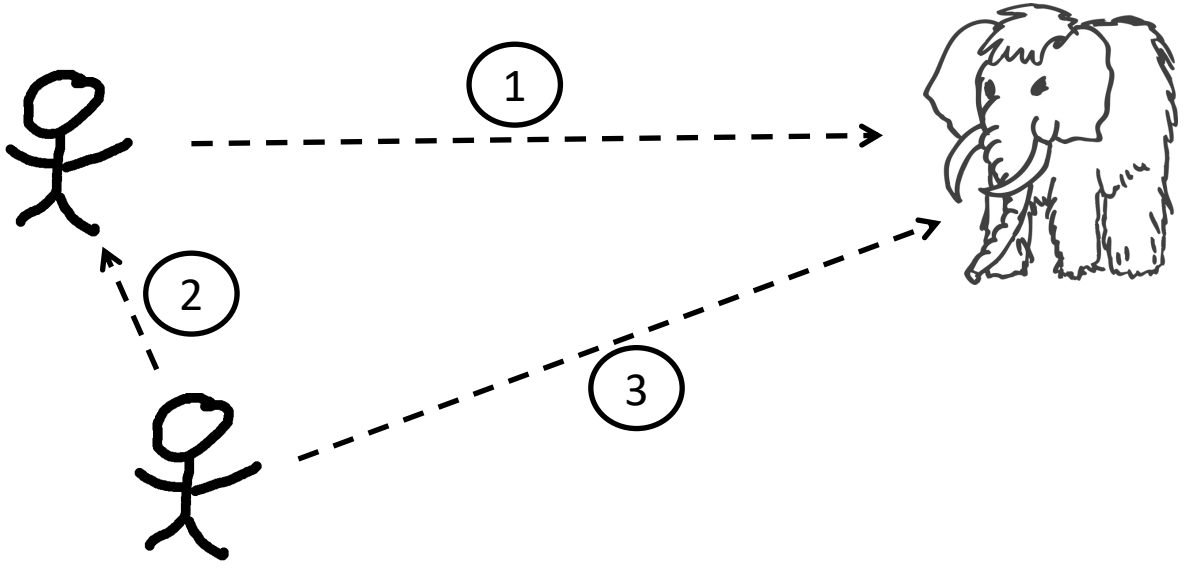
- Purposeless mimicry agents
- Intention-based agents
- Conversational agents

Other plumbing

Additional resources

Conclusion

Conversation requires shared reference



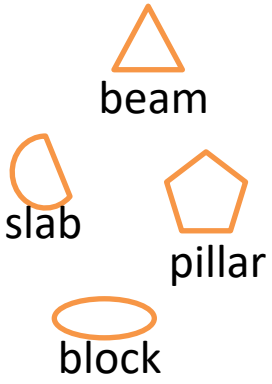
Language begins with shared attention by pointing to things in the world. Words then point to shared ideas in our minds [Gärdenfors, 2014].

Language as shared convention

Wittgenstein and his language games,
Philosophical Investigations, 1958

A

Two people building something.
"Bring me a beam."

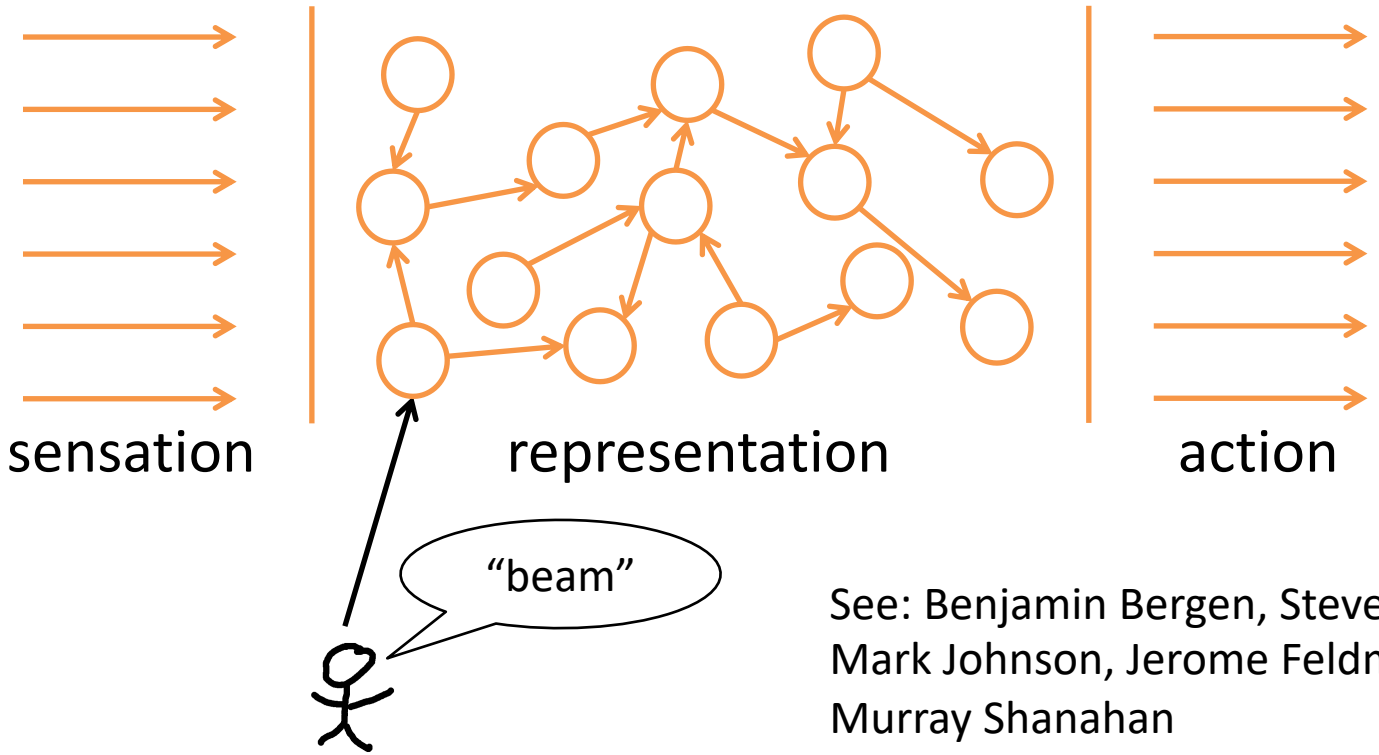


B

Eventually turns into a shared convention for a community

Our brains map community conventions to personal sensations and actions

This is what it means for language to be grounded.

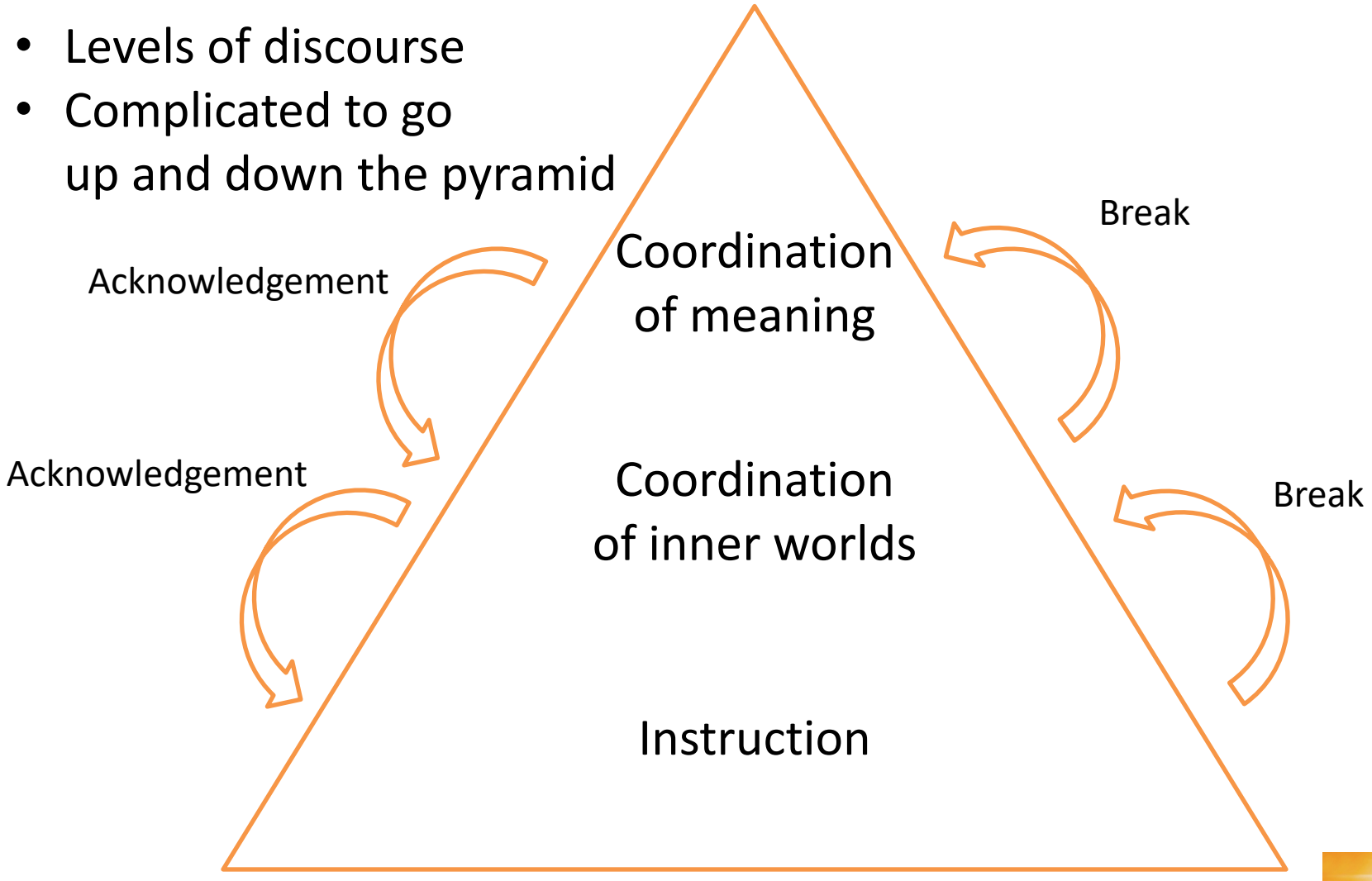


See: Benjamin Bergen, Steven Pinker, Mark Johnson, Jerome Feldman, and Murray Shanahan

- When someone says "beam," we map that to our experience with beams.
- We understand each other because we have had similar experiences.

We negotiate language and meaning as we go

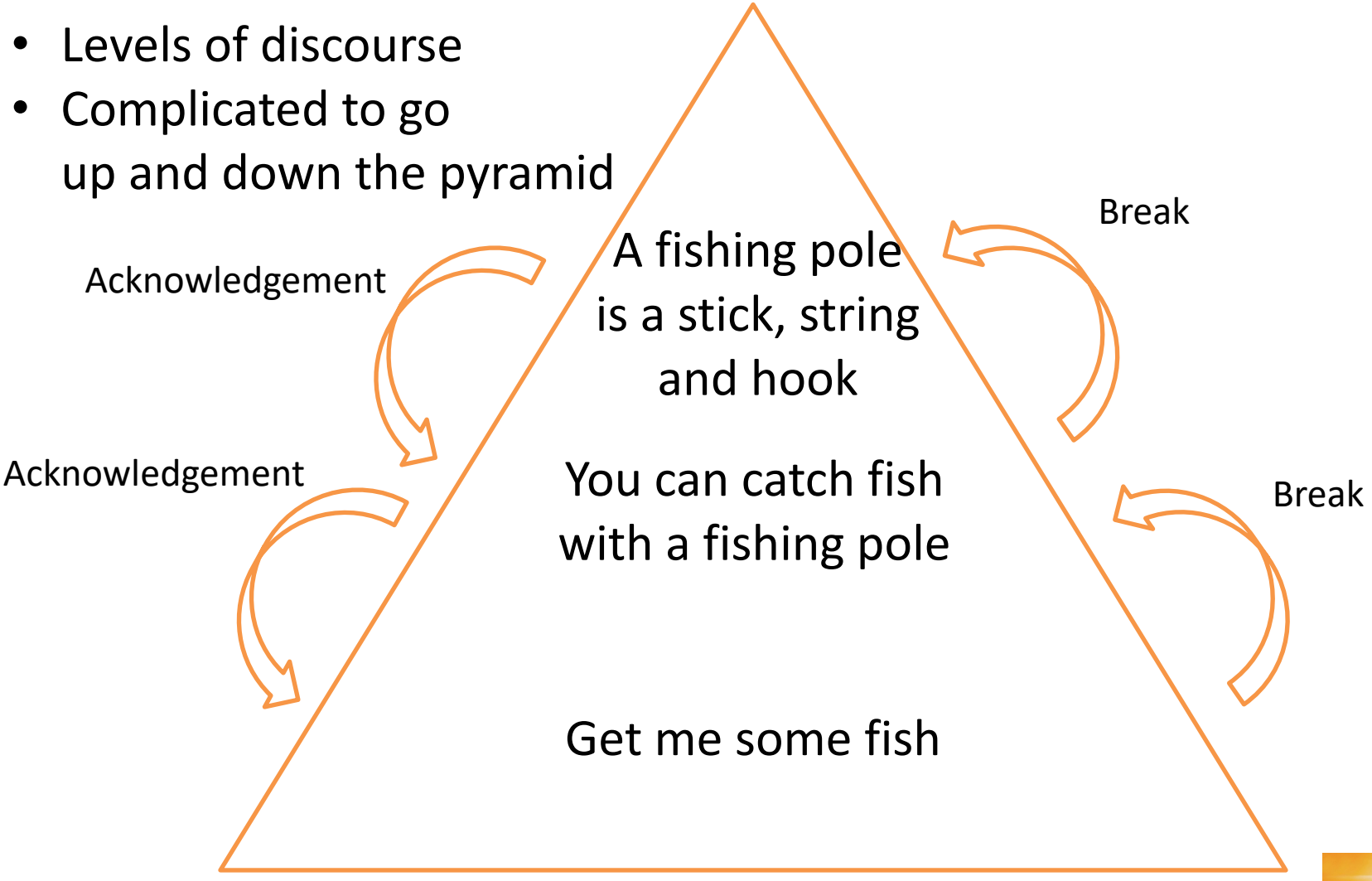
- Levels of discourse
- Complicated to go up and down the pyramid



Modified from Gärdenfors (2014), which was based on Winter (1998)

We negotiate language and meaning as we go

- Levels of discourse
- Complicated to go up and down the pyramid



Modified from Gärdenfors (2014), which was based on Winter (1998)

Conversation has its own rules (pragmatics)

- Conversational maxims: Grice (1975, 1978)
- Breaking these rules is a way to communicate more than the meaning of the words.

Maxim of Quantity: Say only what is not implied.
Yes: "Bring me the block."
No: "Bring me the block by transporting it to my location."
What did she mean by that?

Maxim of Quality: Say only things that are true.
Yes: "I hate carrying blocks."
No: "I love carrying blocks, especially when they are covered in fire ants."
She must be being sarcastic.

Maxim of Relevance: Say only things that matter.
Yes: "Bring me the block."
No: "Bring me the block and birds sing."
What did she mean by that?

Maxim of Manner: Speak in a way that can be understood.
Yes: "Bring me the block."
No: "Use personal physical force to levitate the block and transport it to me."
What did she mean by that?

Sometimes meaning isn't even in the words but how they are spoken (prosody)

``

1. You know. *I* don't. [So don't ask me.]
2. You know. I *don't*. [As a matter of fact, I really don't.]
3. You *know* I don't. [You know that I don't.]

''

Example from *Voice User Interface Design*
by Giangola and Balogh [2004]

Talk outline

What conversation is

The current state of chatbots

- Purposeless mimicry agents
- Intention-based agents
- Conversational agents

Other plumbing

Additional resources

Conclusion

Talk outline

What conversation is

The current state of chatbots

- Purposeless mimicry agents
- Intention-based agents
- Conversational agents

Other plumbing

Additional resources

Conclusion

We've all heard of Eliza

Simple substitutions to mimic a psychologist from the 1960s

"My mother wants me to buy a bazooka." …
"Tell me why your mother wants you to buy a bazooka."

Python implementation at https://www.smallsurething.com/implementing-the-famous-eliza-chatbot-in-python/

You can extend it as much as you want

```
[r'I want to buy (.*)',
 ["How much does {0} cost?",
  "You ain't gonna buy {0}. Not on my watch.",
  "Will {0} finally fill that hole in your soul?"]],
```

I want to buy a car. → How much does a car cost?

Modern mimicry agents are example based

Take a bunch of dialogs and use machine learning to predict what the next statement will be given the last statement.

- Movie and TV subtitles
  - OpenSubtitles
    http://opus.lingfil.uu.se/OpenSubtitles.php
- Can mine Twitter
  - Look for replies to tweets using the API
- Ubuntu Dialog Corpus
  - Dialogs on people wanting technical support
    https://arxiv.org/abs/1506.08909

Sequence-to-sequence model

The sequence-to-sequence (seq2seq) model can encode sequences of tokens, such as sentences, into single vectors.

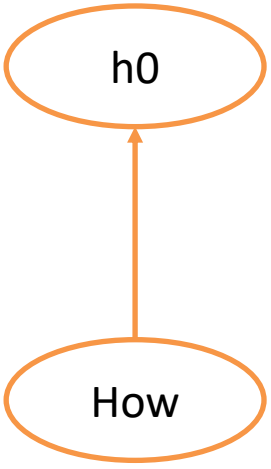
It can then decode these vectors into other sequences of tokens.

Both the encoding and decoding are done using recurrent neural networks (RNNs).

The initial big application for this was machine translation. For example, where the source sentences are English and the target sentences are Spanish.

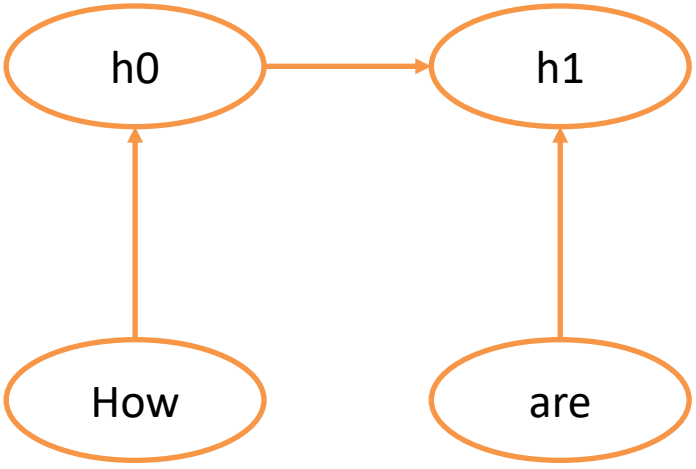
Encoding sentence meaning into a vector

"How are you?"



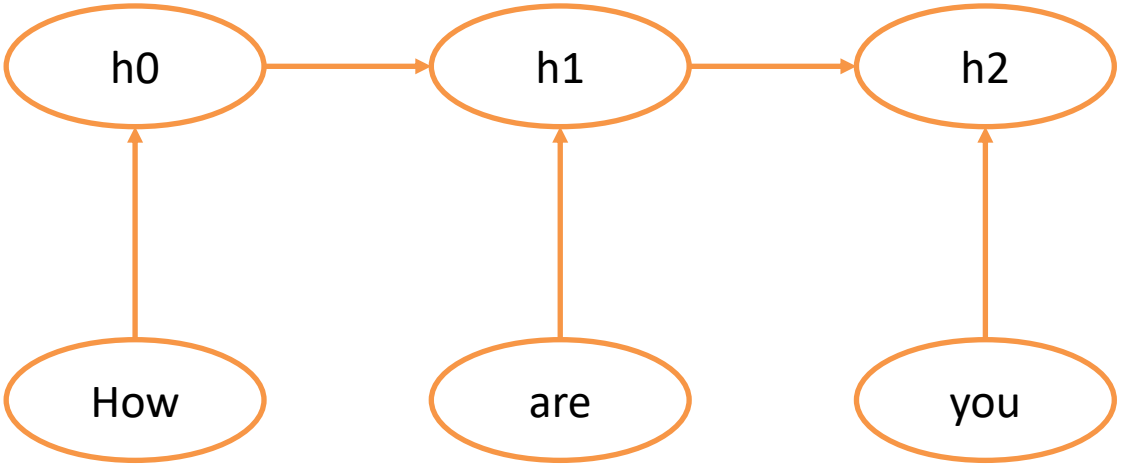
Encoding sentence meaning into a vector

"How are you?"



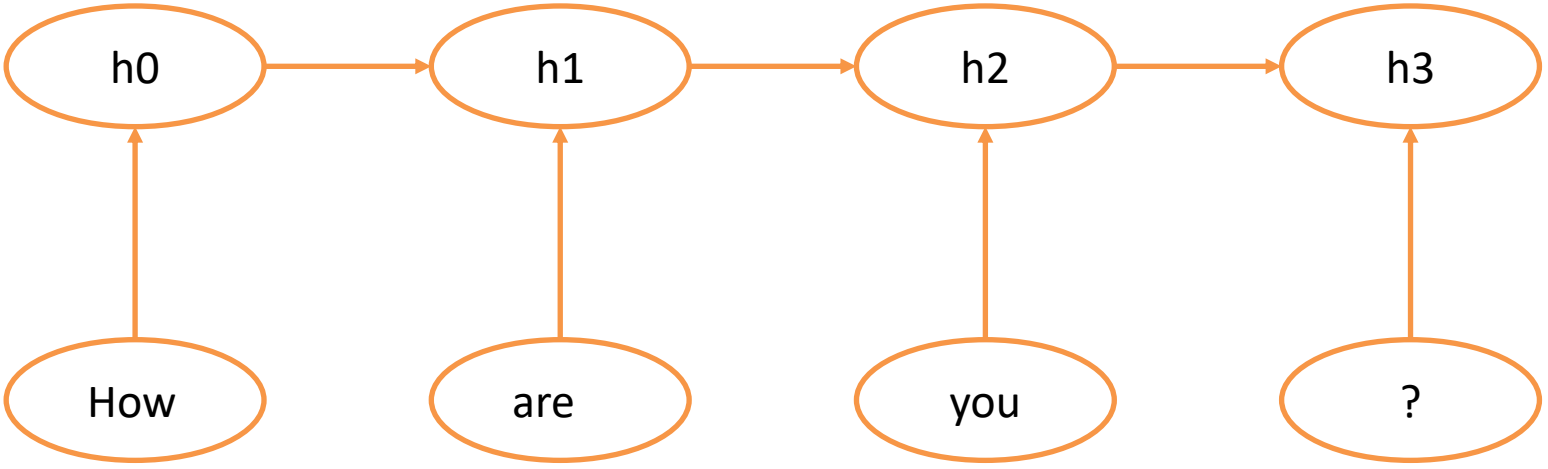
Encoding sentence meaning into a vector

"How are you?"

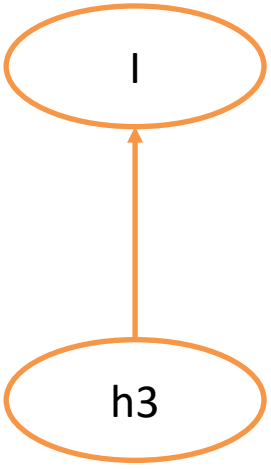


Encoding sentence meaning into a vector

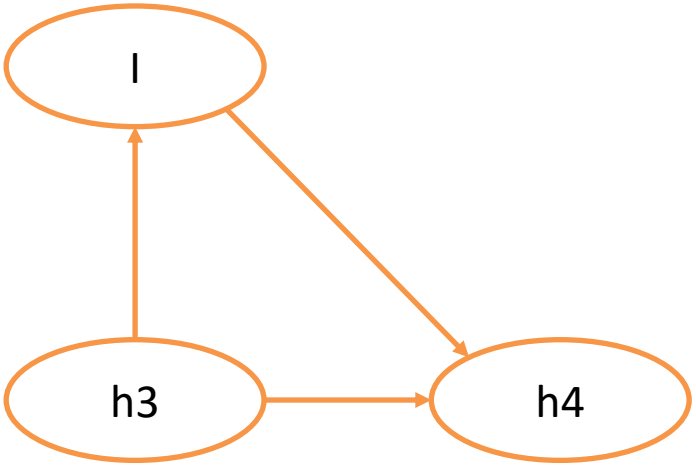
"How are you?"



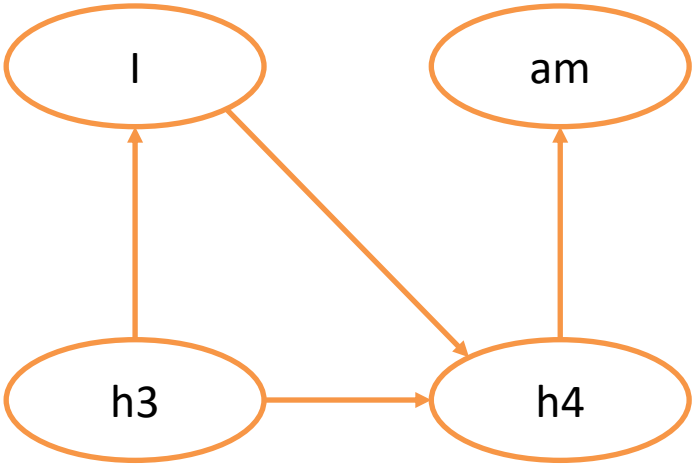
Decoding sentence meaning



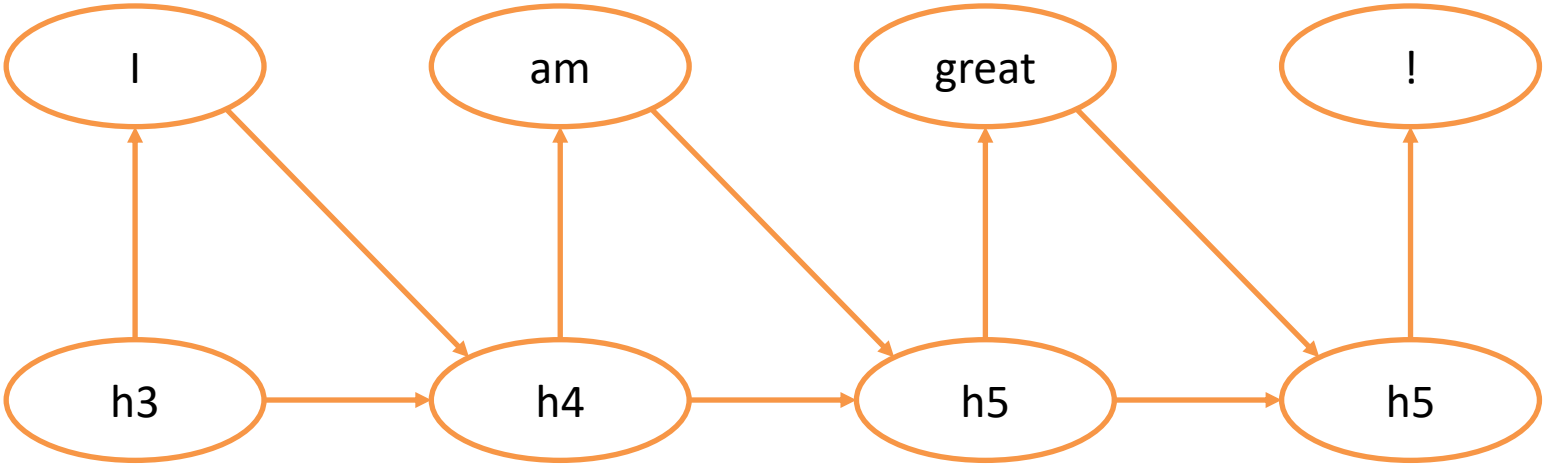
Decoding sentence meaning



Decoding sentence meaning



Decoding sentence meaning



[Cho et al., 2014]

It keeps generating until it generates a stop symbol. Note that the lengths don't need to be the same.

- Treats this task like it is devoid of meaning.
- Great that this can work on just about any kind of seq2seq problem, but this generality highlights its limitation for use as language understanding.

Seq2seq implementation

Enterprise ready seq2seq implementation
https://google.github.io/seq2seq/

Code is here
https://github.com/google/seq2seq

Generative adversarial networks (GANs)

- GANs have a discriminator that tries to tell if the generated response was from a real conversation or generated by the model
- All the rage in image processing
- Still a work in progress for language

Adversarial Learning for Neural Dialogue Generation, Li et al., 2017.
https://arxiv.org/pdf/1701.06547.pdf

Code in Torch (not Python!) for neural dialog generation
https://github.com/jiweil/Neural-Dialogue-Generation

Talk outline

What conversation is

The current state of chatbots

- Purposeless mimicry agents
- Intention-based agents
- Conversational agents

Other plumbing

Additional resources

Conclusion

Talk outline

What conversation is

The current state of chatbots

- Purposeless mimicry agents
- Intention-based agents
- Conversational agents

Other plumbing

Additional resources

Conclusion

Intention-based agents

Examples: Amazon Echo, Google Assistant, Siri, Cortana

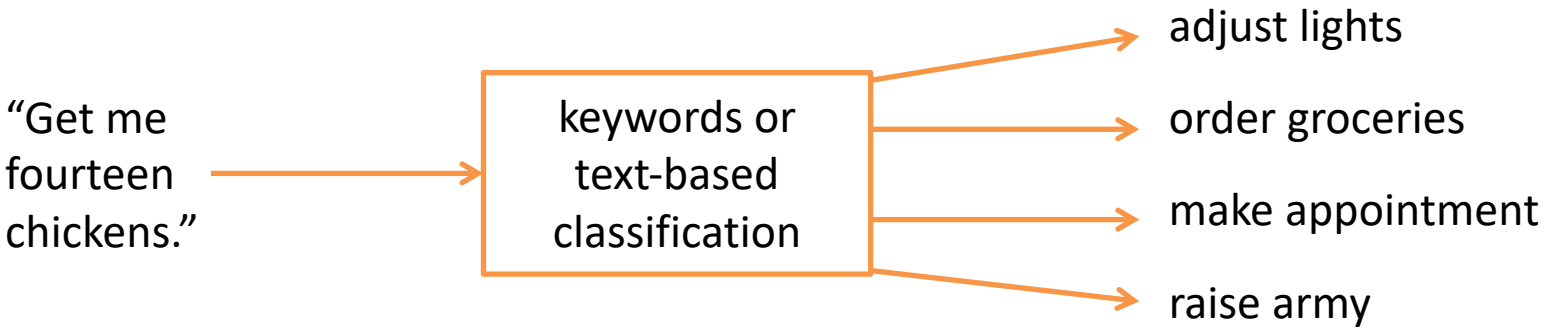
Like a command language

1. Identify what the user wants the machine to do (the “intent”)
2. Figure out the details of the intent so the machine can take action

Amazon Echo

How to determine the intent?

Use keywords or text-based classification

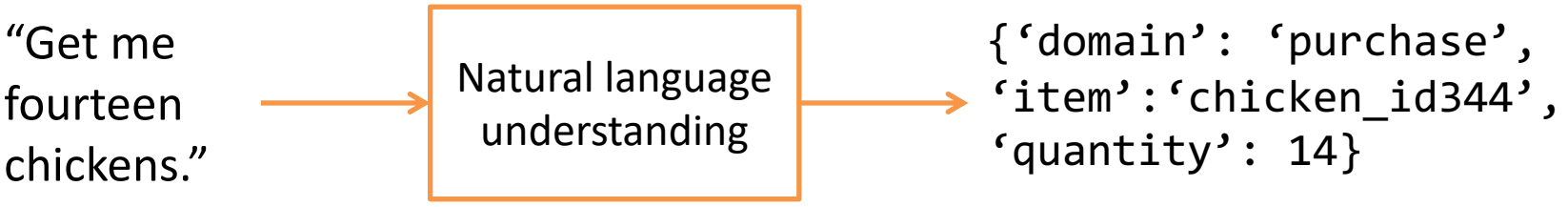


With a bag-of-words in scikit-learn (library for Python)
http://scikit-learn.org/stable/auto_examples/applications/plot_out_of_core_classification.html

Using deep learning with a convolutional neural network (CNN) in TensorFlow (use this if you have a lot of data)
https://github.com/dennybritz/cnn-text-classification-tf

What to do with the intent

Convert the squishiness of language into a Python dictionary



Called a frame and slot semantics

Natural language understanding with context free grammars (compositional semantics)

s[0] is the semantic value of the first item on rule right hand side
s[1] is the semantic value of the second item on rule right hand side

```
$Order → $Purchase $ItemAmount, dunion(s[0],s[1])
$Purchase → is_purchase(tokens), {‘domain’: ’purchase’}
$ItemAmount → $Amount $Item, dunion(s[0],s[1])
$Amount → is_number(tokens), {‘amount’: get_number(tokens)}
$Item → is_item(tokens), {‘item’: get_item(tokens)}
```

```
def is_purchase(tokens):
    return tokens in [‘get me’, ‘buy’, ‘grab me some’]

get_item(‘chickens’) → ‘chicken_id344’
get_item(‘soap’) → ‘soap_id143’
get_number(fourteen) → 14

dunion: dictionary union
```

Natural language understanding with context free grammars (compositional semantics)

s[0] is the semantic value of the first item on rule right hand side
s[1] is the semantic value of the second item on rule right hand side

```
$Order → $Purchase $ItemAmount, dunion(s[0],s[1])
$Purchase → is_purchase(tokens), {‘domain’: ’purchase’}
$ItemAmount → $Amount $Item, dunion(s[0],s[1])
$Amount → is_number(tokens), {‘amount’: get_number(tokens)}
$Item → is_item(tokens), {‘item’: get_item(tokens)}
```

```
def is_purchase(tokens):
    return tokens in [‘get me’, ‘buy’, ‘grab me some’]

get_item(‘chickens’) → ‘chicken_id344’
get_item(‘soap’) → ‘soap_id143’
get_number(fourteen) → 14

dunion: dictionary union
```

Natural language understanding with context free grammars (compositional semantics)

s[0] is the semantic value of the first item on rule right hand side
s[1] is the semantic value of the second item on rule right hand side

```
$Order → $Purchase $ItemAmount, dunion(s[0],s[1])
$Purchase → is_purchase(tokens), {‘domain’: ’purchase’}
$ItemAmount → $Amount $Item, dunion(s[0],s[1])
$Amount → is_number(tokens), {‘amount’: get_number(tokens)}
$Item → is_item(tokens), {‘item’: get_item(tokens)}
```

```
def is_purchase(tokens):
    return tokens in [‘get me’, ‘buy’, ‘grab me some’]

get_item(‘chickens’) → ‘chicken_id344’
get_item(‘soap’) → ‘soap_id143’
get_number(fourteen) → 14

dunion: dictionary union
```

Natural language understanding with context free grammars (compositional semantics)

`s[0]` is the semantic value of the first item on rule right hand side
`s[1]` is the semantic value of the second item on rule right hand side

```
$Order → $Purchase $ItemAmount, dunion(s[0],s[1])
$Purchase → is_purchase(tokens), {‘domain’: ’purchase’}
$ItemAmount → $Amount $Item, dunion(s[0],s[1])
$Amount → is_number(tokens), {‘amount’: get_number(tokens)}
$Item → is_item(tokens), {‘item’: get_item(tokens)}
```

```
def is_purchase(tokens):
    return tokens in [‘get me’, ‘buy’, ‘grab me some’]

get_item(‘chickens’) → ‘chicken_id344’
get_item(‘soap’) → ‘soap_id143’
get_number(fourteen) → 14

dunion: dictionary union
```

Natural language understanding with context free grammars (compositional semantics)

s[0] is the semantic value of the first item on rule right hand side
s[1] is the semantic value of the second item on rule right hand side

```
$Order → $Purchase $ItemAmount, dunion(s[0],s[1])
$Purchase → is_purchase(tokens), {‘domain’: ’purchase’}
$ItemAmount → $Amount $Item, dunion(s[0],s[1])
$Amount → is_number(tokens), {‘amount’: get_number(tokens)}
$Item → is_item(tokens), {‘item’: get_item(tokens)}
```

```
def is_purchase(tokens):
    return tokens in [‘get me’, ‘buy’, ‘grab me some’]

get_item(‘chickens’) → ‘chicken_id344’
get_item(‘soap’) → ‘soap_id143’
get_number(fourteen) → 14

dunion: dictionary union
```

Natural language understanding with context free grammars (compositional semantics)

`s[0]` is the semantic value of the first item on rule right hand side
`s[1]` is the semantic value of the second item on rule right hand side

```
$Order → $Purchase $ItemAmount, dunion(s[0],s[1])
$Purchase → is_purchase(tokens), {‘domain’: ’purchase’}
$ItemAmount → $Amount $Item, dunion(s[0],s[1])
$Amount → is_number(tokens), {‘amount’: get_number(tokens)}
$Item → is_item(tokens), {‘item’: get_item(tokens)}
```

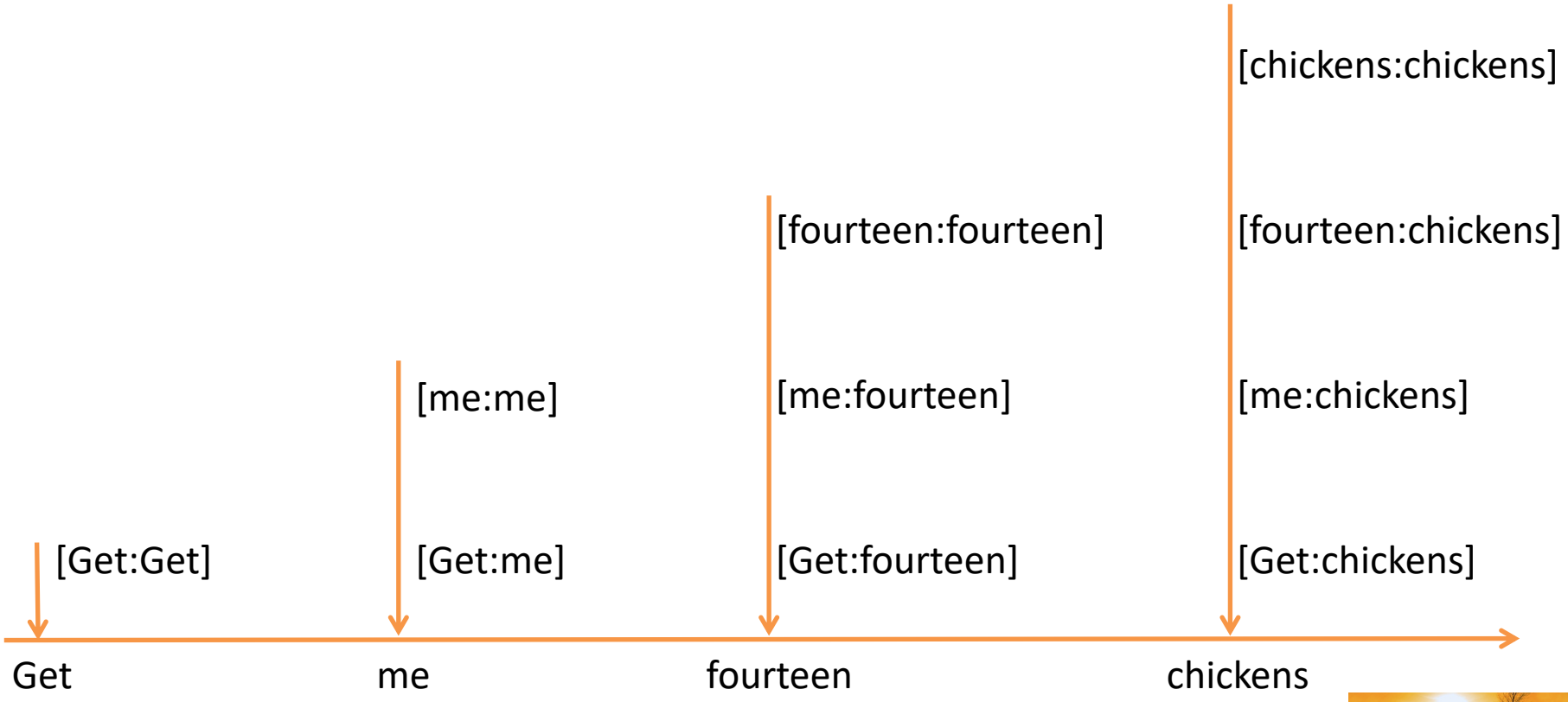
```
def is_purchase(tokens):
    return tokens in [‘get me’, ‘buy’, ‘grab me some’]

get_item(‘chickens’) → ‘chicken_id344’
get_item(‘soap’) → ‘soap_id143’
get_number(fourteen) → 14

dunion: dictionary union
```

Parsing: applying rules to text to get semantics

- Parsing is done using bottom-up dynamic programming.
- For each box, it loops over all n possible midpoints. E.g., (1,4) -> (1,2),(2,4) and (1,3),(3,4)
- This is why the rules have to have two non-terminals on RHS. Not a problem. You can always put a context free grammar in this form (called Chomsky Normal Form).



Parsing: applying rules to text to get semantics

- Parsing is done using bottom-up dynamic programming.
- For each box, it loops over all n possible midpoints. E.g., (1,4) -> (1,2),(2,4) and (1,3),(3,4)
- This is why the rules have to have two non-terminals on RHS. Not a problem. You can always put a context free grammar in this form (called Chomsky Normal Form).

			7. `$Item`
		4. `$Amount`	8. `$ItemAmount`
	2. ‘me’	5. `$Amount`	9. `$Order`
1. ‘get’	3. `$Purchase`	6. `$Amount`	10. `$Order`

Get me fourteen chickens

Parsing: applying rules to text to get semantics

- Parsing is done using bottom-up dynamic programming.
- For each box, it loops over all n possible midpoints. E.g., (1,4) -> (1,2),(2,4) and (1,3),(3,4)
- This is why the rules have to have two non-terminals on RHS. Not a problem. You can always put a context free grammar in this form (called Chomsky Normal Form).

			7. {'item': 'chicken_id344'}
		4. {'amount':14}	8. {'item': 'chicken_id344', 'amount':14}
	2. 'me'	5. {'amount':14}	9. {'domain': 'purchase', 'item': 'chicken_id344', 'amount':14,}
1. 'get'	3. {'domain': 'purchase'}	6. {'amount':14}	10. {'domain': 'purchase', 'item': 'chicken_id344', 'amount':14,}

Get me fourteen chickens

Python code for natural language understanding

Code is available in Python in the SippyCup library
https://github.com/wcmac/sippycup

There can be many valid parses, and a scoring algorithm can be used to choose which one to use

Talk outline

What conversation is

The current state of chatbots

- Purposeless mimicry agents
- Intention-based agents
- Conversational agents

Other plumbing

Additional resources

Conclusion

Talk outline

What conversation is

The current state of chatbots
- Purposeless mimicry agents
- Intention-based agents
- Conversational agents

Other plumbing

Additional resources

Conclusion

Conversational agents

Extended, meaningful conversations. Have to be able to keep track of the state of the conversation and know when the person wants to talk about something else.

Blog post I wrote, You and Your Bot: A New Kind of Lifelong Relationship
https://chatbotsmagazine.com/you-and-your-bot-a-new-kind-of-lifelong-relationship-6a9649feeb71

A teacher for the young

- Starts out as a cell phone app for a child
- Child can talk to its cartoon face and show it her toys
- Teaches her things
  - "If you have 7 giraffes and bought 3 more, how many would you have?"
  - "If you eat your chocolate bear today, how are you going to feel tomorrow when you no longer have it?"

A guide for the adult

- As the child grows, so does the app, so it becomes her operating system
- Knows what you know. Can give turn-by-turn directions on fixing the washing machine

An advocate for the old

- Could advocate for us and guide us through government and healthcare bureaucracies
- Could help us live independently longer.
  - Imagine it sees you confused. Could talk you through making coffee.

How could a machine manage such conversations?

A dialog manager needs to

Take the person through all the things it wants to talk about
- Teach her math
- Then talk about planets

Recognize when she wants to talk about something else and queue the current topics for later

RavenClaw from CMU is probably the best known dialog manager

Principles of RavenClaw

RavenClaw dialog manager.
Part of CMU Olympus system http://wiki.speech.cs.cmu.edu/olympus/index.php/Olympus

Dialog agents
Little programs organized hierarchically that correspond to different bits of conversation
E.g., general dialog agent about cooking, wants to discuss multiple things

Dialog stack
Stack of dialog agents to keep track of all the things the machine wants to talk about

Expectation agenda
Data structure of dialog agents to keep track of what the machine expects to hear

Execution phase
Take top dialog agent off dialog stack and let it talk. Also, set up what the machine expects to hear in the expectation agenda

Listening phase
Take what the person said and see which dialog agent can respond

Hierarchy of dialog agents

Dialog Stack

Expectation Agenda

Talk about toys always there.

Hierarchy of dialog agents

Robot just finished asking, “What is 4 + 5?”
Child says: “Do you like Mr. Fluffles?”

Dialog Stack

Expectation Agenda

Hierarchy of dialog agents

Robot just finished asking, “What is 4 + 5?”
Child says: “Do you like Mr. Fluffles?”
Robot responds: “Very nice. Is he your favorite?”

Dialog Stack

Expectation Agenda

Hierarchy of dialog agents

Robot just finished asking, “What is 4 + 5?”
Child says: “Do you like Mr. Fluffles?”
Robot responds: “Very nice. Is he your favorite?”

Now we are talking about toys.

Dialog Stack

Expectation Agenda

Hierarchy of dialog agents

Robot just finished asking, “What is 4 + 5?”
Child says: “Do you like Mr. Fluffles?”
Robot responds: “Very nice. Is he your favorite?”

Child says, “Of course!”
Update database
with new favorite toy.

Now we are talking about toys.

Dialog Stack

Expectation Agenda

State-based dialog agents

Markov decision process
Set of **states**
Set of **actions**
Get a **reward** for being in a state s and taking an action a

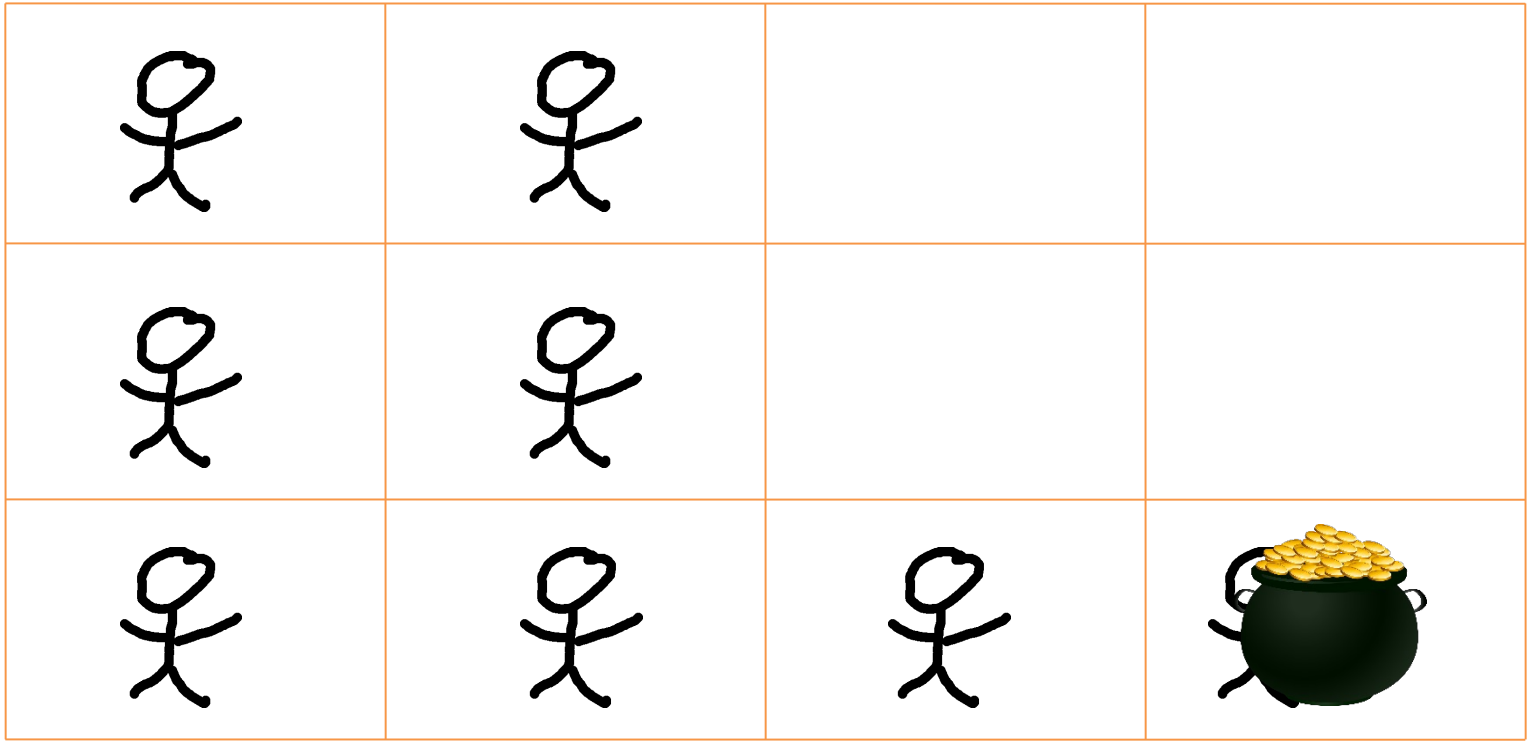
States are things such as what the bot knows (questions it has answered), the last thing the bot said, and the last thing the user said.

Actions are making particular statements.

Reward comes from meeting a goal state, such as child giving correct answer to a math problem or successfully completing a travel reservation.

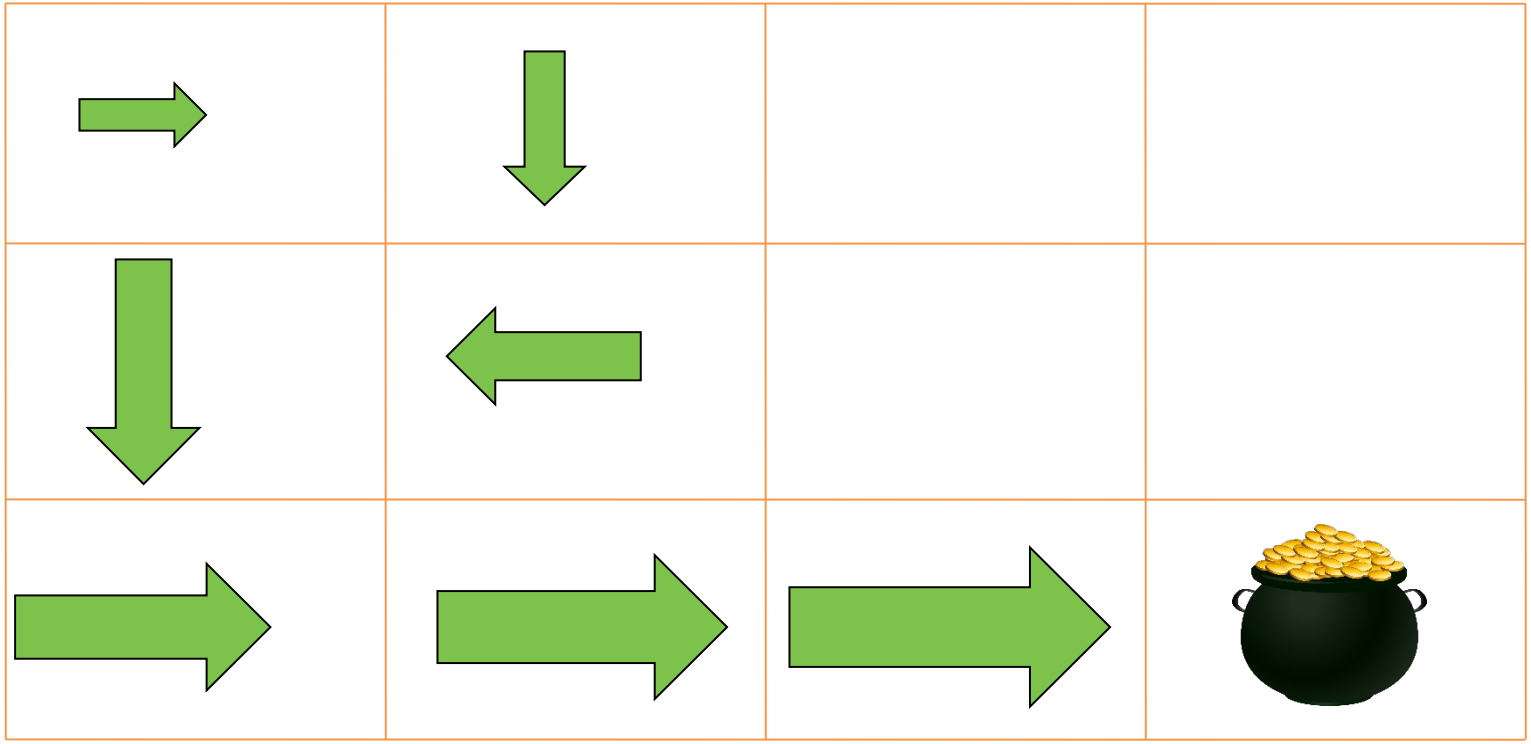
Use **reinforcement learning** to learn a policy that gives the best action a for being in state s.

Beginning with random exploration



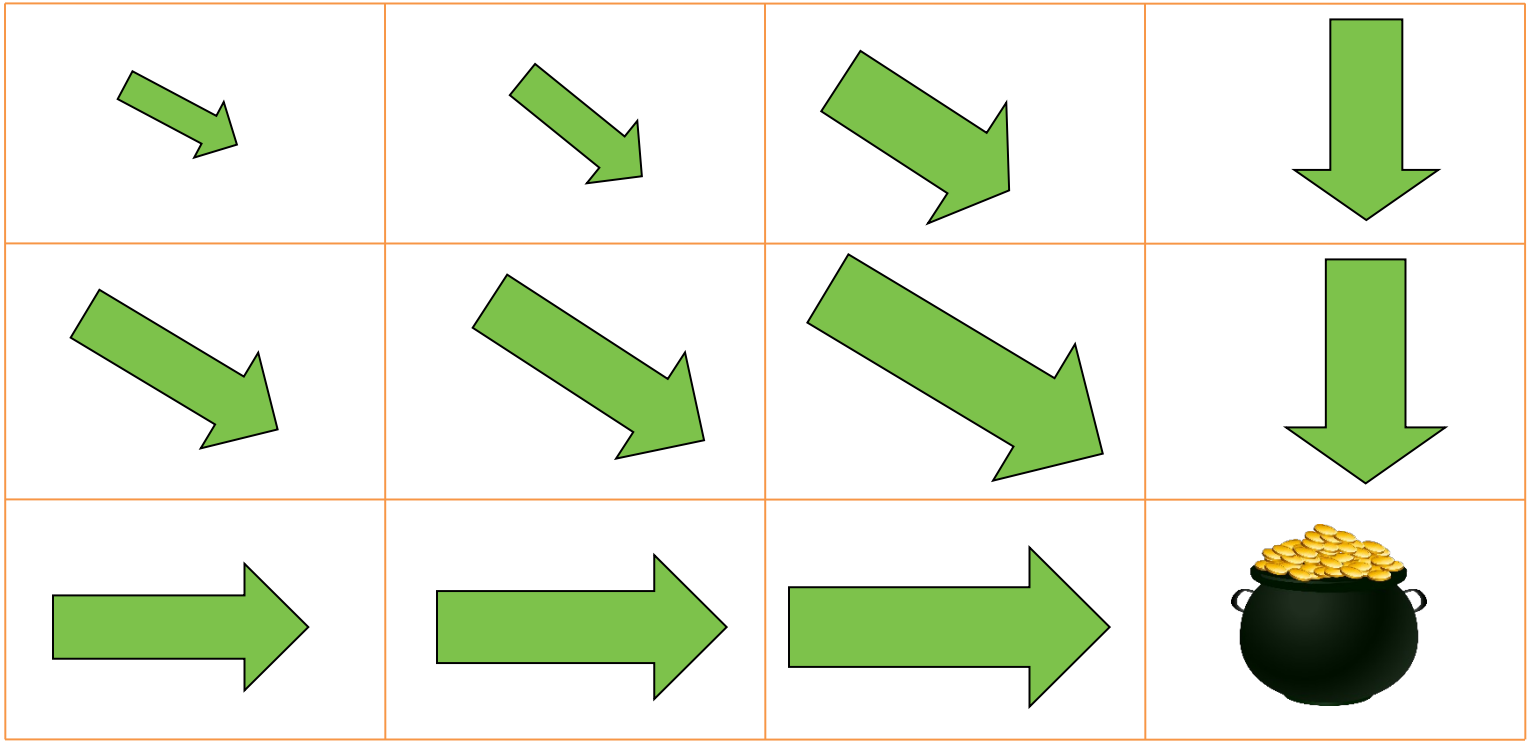
In reinforcement learning, the agent begins by randomly exploring until it reaches its goal.

Reaching the goal



- When it reaches the goal, credit is propagated back to its previous states.
- The agent learns the function $Q(s,a)$, which gives the cumulative expected discounted reward of being in state s and taking action a and acting according to the policy thereafter.

Learning the behavior



Eventually, the agent learns the value of being in each state and taking each action and can therefore always do the best thing in each state.

But there is limited observability

- Since speech-to-text and natural language understanding are error prone, you actually don't know for sure which state you are in.
- Does she want to talk about her toys, or is she telling me that I should be fluffier?

- Partially Observable Markov Decision Process (POMDP)
- As you can guess, these need a lot of training data. Generally have to create a simulation of a person for the bot to talk to.

Talk outline

What conversation is

The current state of chatbots

- Purposeless mimicry agents
- Intention-based agents
- Conversational agents

Other plumbing

Additional resources

Conclusion

Talk outline

What conversation is

The current state of chatbots
- Purposeless mimicry agents
- Intention-based agents
- Conversational agents

Other plumbing

Additional resources

Conclusion

Basic working with text

spaCy is good for general text processing.

O'Reilly video course I made on doing NLP in Python (not free)
http://shop.oreilly.com/product/0636920061007.do

Free video I made on tokenizing text in spaCy
https://www.oreilly.com/learning/how-can-i-tokenize-a-sentence-with-python

https://demos.explosion.ai/displacy

```
from spacy.en import English
parser = English()
text = "Get me fourteen chickens."
tokens = parser(text)
tokens = [t.orth_ for t in tokens if not t.orth_.isspace()]
print(tokens)
```

Tokenization result:

```
['Get', 'me', 'fourteen', 'chickens', '.']
```

Speech-to-Text in Python

Use SpeechRecognition 3.6.5 Python library
https://pypi.python.org/pypi/SpeechRecognition/

Good blog post by Gurwinder Gulati here
https://ggulati.wordpress.com/2016/02/24/coding-jarvis-in-python-3-in-2016/

Sphinx doesn't work well for me. I use Google Cloud Speech API
https://cloud.google.com/speech/

Look at:
https://github.com/Uberi/speech_recognition/blob/master/speech_recognition/__init__.py

My modified version of Gurwinder's blog post

```
with open('my_json_cred.json', 'r') as json_data:
    json_cred = json_data.read()
# other stuff ...
text = recognizer.recognize_google_cloud(audio,
                                         credentials_json=json_cred,
                                         language="en-US",
                                         preferred_phrases=None,
                                         show_all=False)
```

Can also use WaveNet. TensorFlow implementation: https://github.com/buriburisuri/speech-to-text-wavenet
WaveNet paper: https://arxiv.org/abs/1609.03499

Text-to-Speech in Python

```
def speak(text):
    # Have to use triple quotes because sometimes
    # text has single quotes in it.
    command = """say "{}" """.format(text)
    os.system(command)
    print("done speaking")
```

Can also use WaveNet https://github.com/ibab/tensorflow-wavenet

Talk outline

What conversation is

The current state of chatbots

- Purposeless mimicry agents
- Intention-based agents
- Conversational agents

Other plumbing

Additional resources

Conclusion

Talk outline

What conversation is

The current state of chatbots
- Purposeless mimicry agents
- Intention-based agents
- Conversational agents

Other plumbing

Additional resources

Conclusion

Additional resources not previously mentioned

Detailed resources are surprisingly hard to find

- Jurafsky https://web.stanford.edu/class/cs124/lec/chatbot.pdf
- Jurafsky and Martin, *Speech and Language Processing,* Ch. 24, 2009
  - New edition in progress https://web.stanford.edu/~jurafsky/slp3/
- Cathy Pearl, Designing Voice User Interfaces, 2016

Talk outline

What conversation is

The current state of chatbots

- Purposeless mimicry agents
- Intention-based agents
- Conversational agents

Other plumbing

Additional resources

Conclusion

Talk outline

What conversation is

The current state of chatbots
- Purposeless mimicry agents
- Intention-based agents
- Conversational agents

Other plumbing

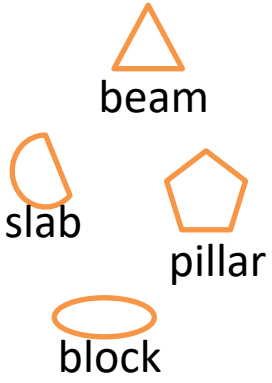
Additional resources

Conclusion

Conclusion (1 of 3)

Computers need to know our conventions

A

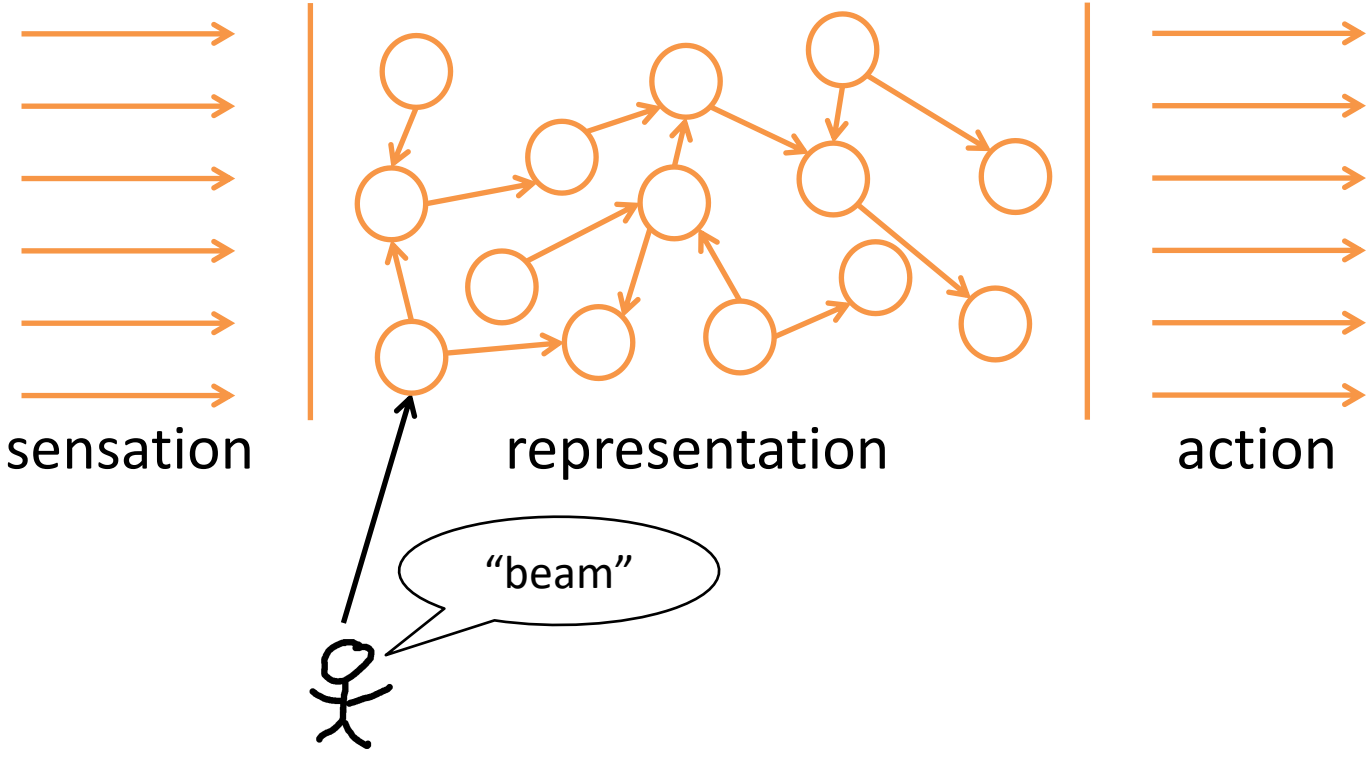


B

We can program this in

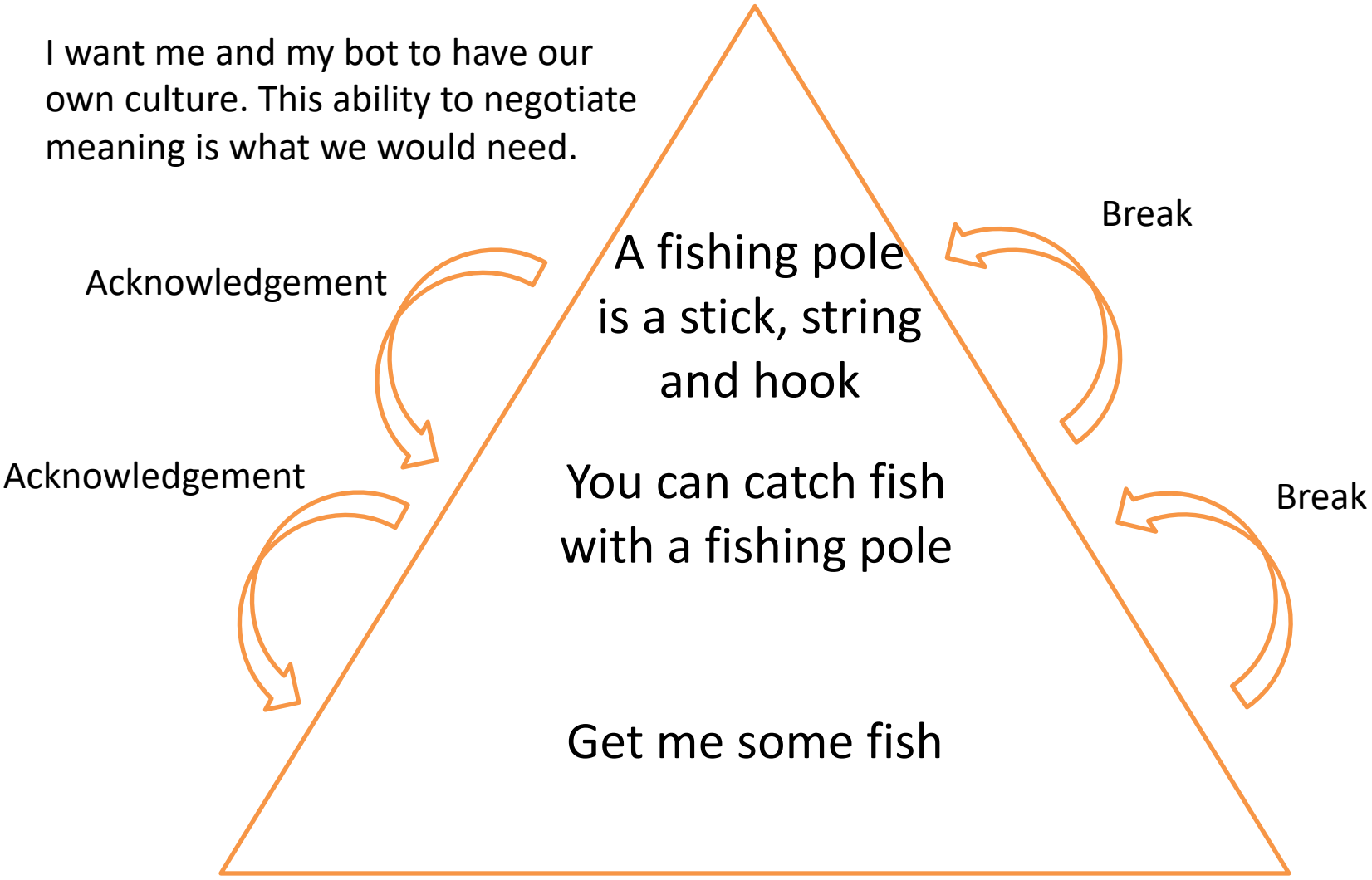
Conclusion (2 of 3)

- Conventions need to be grounded in a system of sensation and action
- Hard when machines don't have our bodies and experience
- Blog post of wrote in using physics simulations for natural language understanding https://chatbotslife.com/computers-could-understand-natural-language-using-simulated-physics-26e9706013da



Conclusion (3 of 3)

I want me and my bot to have our own culture. This ability to negotiate meaning is what we would need.



Machines also need to know the rules of conversation, so they can understand meanings beyond words. And they need to understand meanings from how words are said.

Thanks for listening

Questions?

Jonathan Mugan
@jmugan