Introduction

LING 571 — Deep Processing Techniques for NLP
September 30, 2020
Shane Steinert-Threlkeld

Introductions

- Name [and how you prefer to be addressed]

- Program / year / status at UW

- What are you most excited about in this course?

Roadmap

- **Motivation**

- Language and Intelligence

- Knowledge of Language

- Course Overview

- Intro to Syntax and Parsing

Motivation: Applications

- Applications of Speech and Language Processing
  - Call Routing
  - Information Retrieval
  - Question Answering
  - Machine Translation
  - Dialog Systems
  - Spell– and Grammar– Checking
  - Sentiment Analysis
  - Information Extraction
  - …

Building on Many Fields

- **Linguistics**: *Morphology, phonology, syntax, semantics…*

- **Psychology**: *Reasoning, mental representations*

- **Formal Logic**

- **Philosophy (of Language)**

- **Theory of Computation**: *Automata theory*

- **Artificial Intelligence**: *Search, Reasoning, Knowledge Representation, Machine Learning, Pattern Matching*

- **Probability**

Roadmap

- Motivation

- **Language and Intelligence**

- Knowledge of Language

- Course Overview

- Intro to Syntax and Parsing

Operationalizing Intelligence:
The Turing Test (1950)

- Two contestants: Human vs. Computer

  - **Judge**: human

  - **Test**: interact via text questions

  - **Question**: Can judge tell which contestant is human?

- *Crucially*:

  - Posits that passing requires language use and understanding

Limitations of the Turing Test

- ELIZA ([Weizenbaum, 1966](#)) [[Try it Online](#)]

  - Simulates Rogerian therapist:
    ```
    User: You are like my father in some ways
    ELIZA: WHAT RESEMBLANCE DO YOU SEE
    USER: You are not very aggressive
    ELIZA: WHAT MAKES YOU THINK I AM NOT AGGRESSIVE
    ```

  - Passes the Test! (Sort of)

  - Simple pattern matching technique

Turing Test Revisited:
"On the web, no one knows you're a…"

- **Problem**: "Bots":

  - Automated agents overrun services

  - Challenge: Prove you're human

- **Test**: Something a human can do, but a bot can't.

- **Solution:** CAPTCHAs

  - **C**ompletely **A**utomated **P**ublic **T**uring test to tell **C**omputers and **H**umans **A**part
    (Von Ahn et al., 2003)

  - Initially: Distorted images, driven by perception

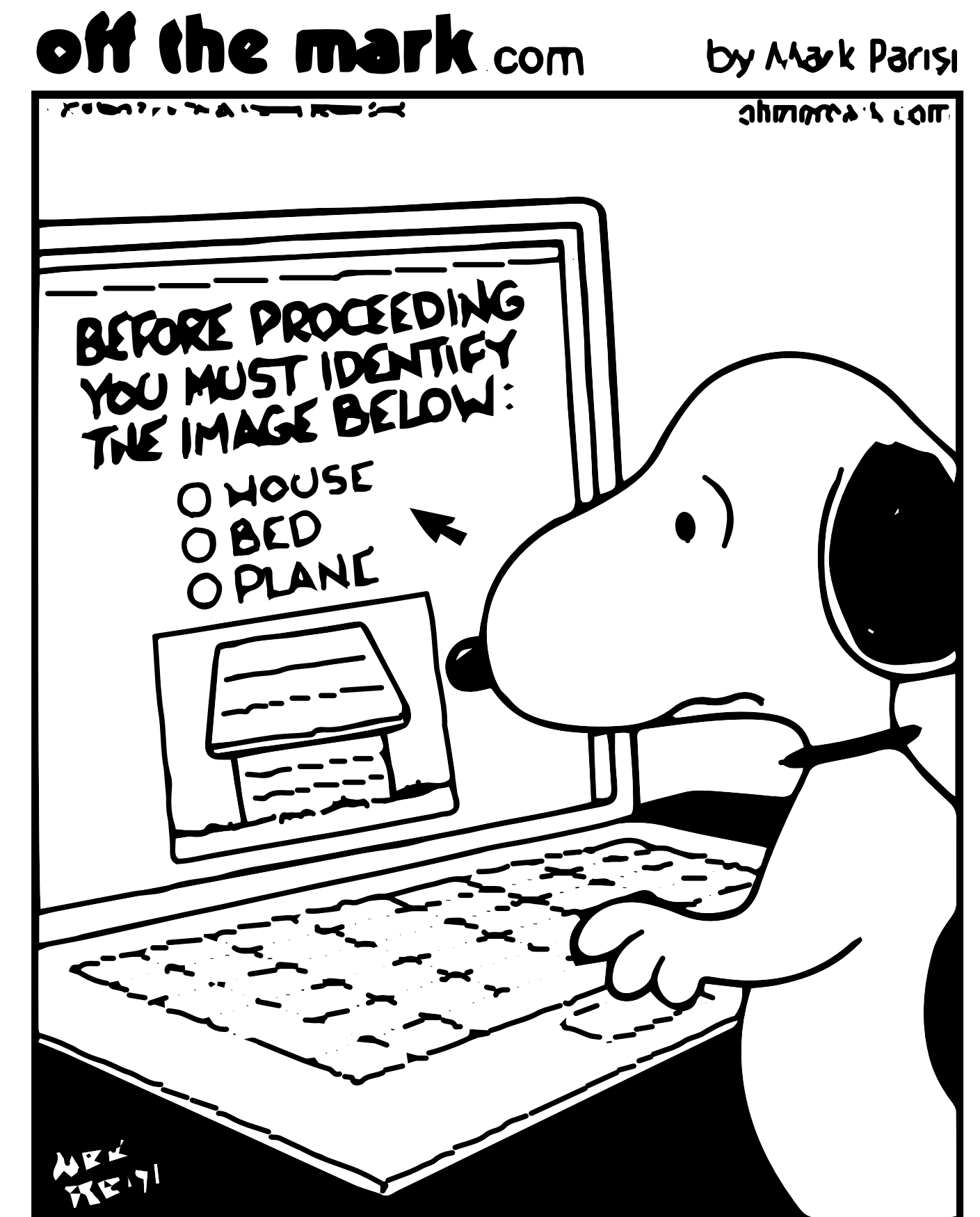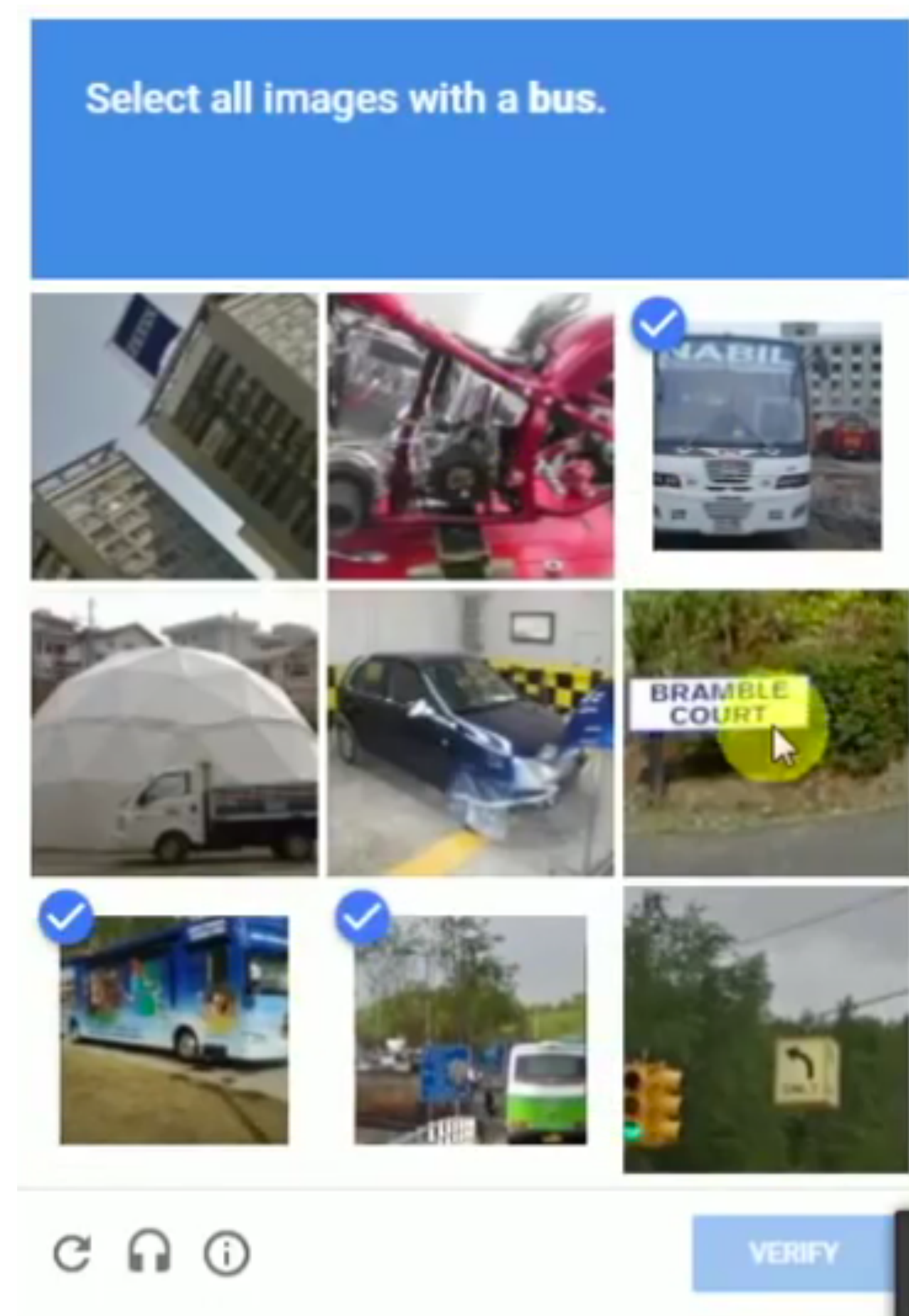 - Long-term: Inspires "arms race"

CAPTCHA arms race
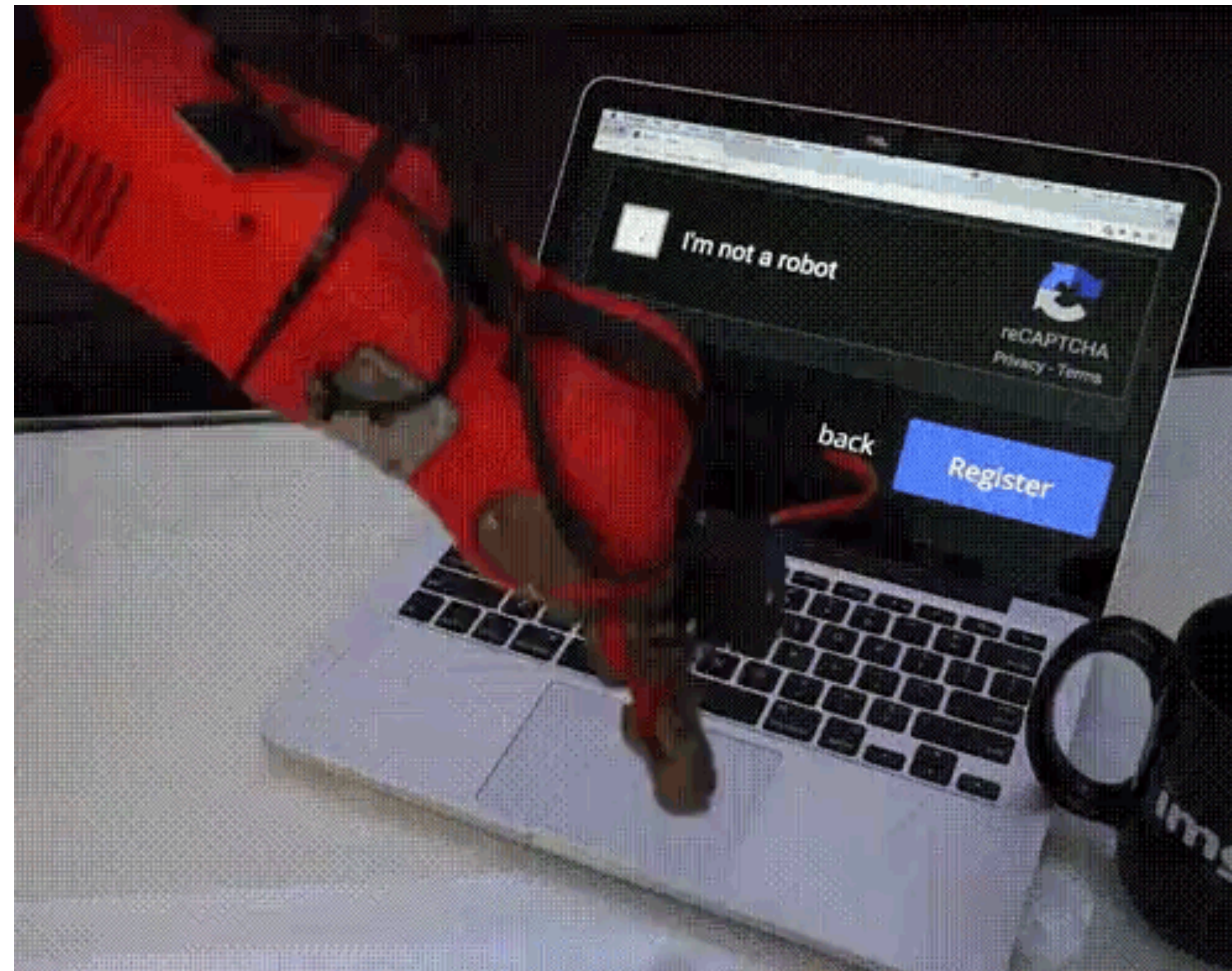
Turing Test Revisited:

"On the web, no one knows you're a…"

- Current Incarnation

  - Still perception-based

  - But also relies on world knowledge

  - "What is a bus?"

    - Assumes that the user has extrinsic, **shared** world knowledge

Select all images with a **bus**.

off the mark .com   by Mark Parisi

BEFORE PROCEEDING YOU MUST IDENTIFY THE IMAGE BELOW:
○ HOUSE
○ BED
○ PLANE

Roadmap

- Motivation

- Language and Intelligence

- **Knowledge of Language**

- Course Overview

- Intro to Syntax and Parsing

Knowledge of Language

- NLP vs. Data Processing

- POSIX command "wc"

  - Counts total number of bytes, words, and lines in text file

  - bytes and lines → data processing

  - words → *what do we mean by "word"?*

Knowledge of Language

- What does HAL (of *2001, A Space Odyssey*) need to know to converse?

Dave: *Open the pod bay doors, HAL.*
HAL: *I'm sorry, Dave. I'm afraid I can't do that.*

Knowledge of Language

- What does HAL (of *2001, A Space Odyssey*) need to know to converse?

Dave: Open the pod bay doors, HAL.
HAL: I'm sorry, Dave. I'm afraid I can't do that.

- **Phonetics & Phonology** (Ling 450/550)

  - Sounds of a language, acoustics

  - Legal sound sequences in words

Knowledge of Language

- What does HAL (of *2001, A Space Odyssey*) need to know to converse?

Dave: Open the pod bay doors, HAL.
HAL: I'm sorry, Dave. I'm afraid I can't do that.

- **Morphology** (Ling 570)

  - Recognize, produce variation in word forms

  - Singular vs. plural: `Door + sg → "door"    Door + pl → "doors"`

  - Verb inflection: `be + 1st Person + sg + present → "am"`

Knowledge of Language

- What does HAL (of *2001, A Space Odyssey*) need to know to converse?

> **Dave**: *Open the pod bay doors, HAL.*
> **HAL**: *I'm sorry, Dave. I'm afraid I can't do that.*

- **Part-of-speech Tagging** (Ling 570)

  - Identify word use in sentence

  - Bay (Noun) — Not verb, adjective

Knowledge of Language

- What does HAL (of *2001, A Space Odyssey*) need to know to converse?

  Dave: *Open the pod bay doors, HAL.*
  HAL: *I'm sorry, Dave. I'm afraid I can't do that.*

- **Syntax**

  - (566: Analysis, 570: Chunking, 571: Parsing)

  - Order and group words in sentence

    - cf. *"I'm I do, sorry that afraid Dave I can't"*

Knowledge of Language

- What does HAL (of *2001, A Space Odyssey*) need to know to converse?

> *Dave: Open the pod bay doors, HAL.*
> *HAL: I'm sorry, Dave. I'm afraid I can't do that.*

- **Semantics** (Word Meaning)

  - Individual (lexical) + Combined (Compositional)

  - `'Open'` **:** <u>AGENT</u> **cause** <u>THEME</u> **to become** *open*;

    - 'pod bay doors' → doors to the 'pod bay' → the bay which houses the pods.

Knowledge of Language

- What does HAL (of *2001, A Space Odyssey*) need to know to converse?

Dave: Open the pod bay doors, HAL.
HAL: I'm sorry, Dave. I'm afraid I can't do that.

- **Pragmatics/Discourse/Dialogue** (Ling 571)

  - Interpret utterances in context

  - Speech as acts (request vs. statement)

  - Reference resolution: "I"=[**HAL**]; "that"=[**open…doors**]

 - Politeness: "*I'm sorry*, *I'm afraid I can't*…"

Roadmap

- Motivation

- Language and Intelligence

- Knowledge of Language

- **Course Overview**

- Intro to Syntax and Parsing

Course Overview:
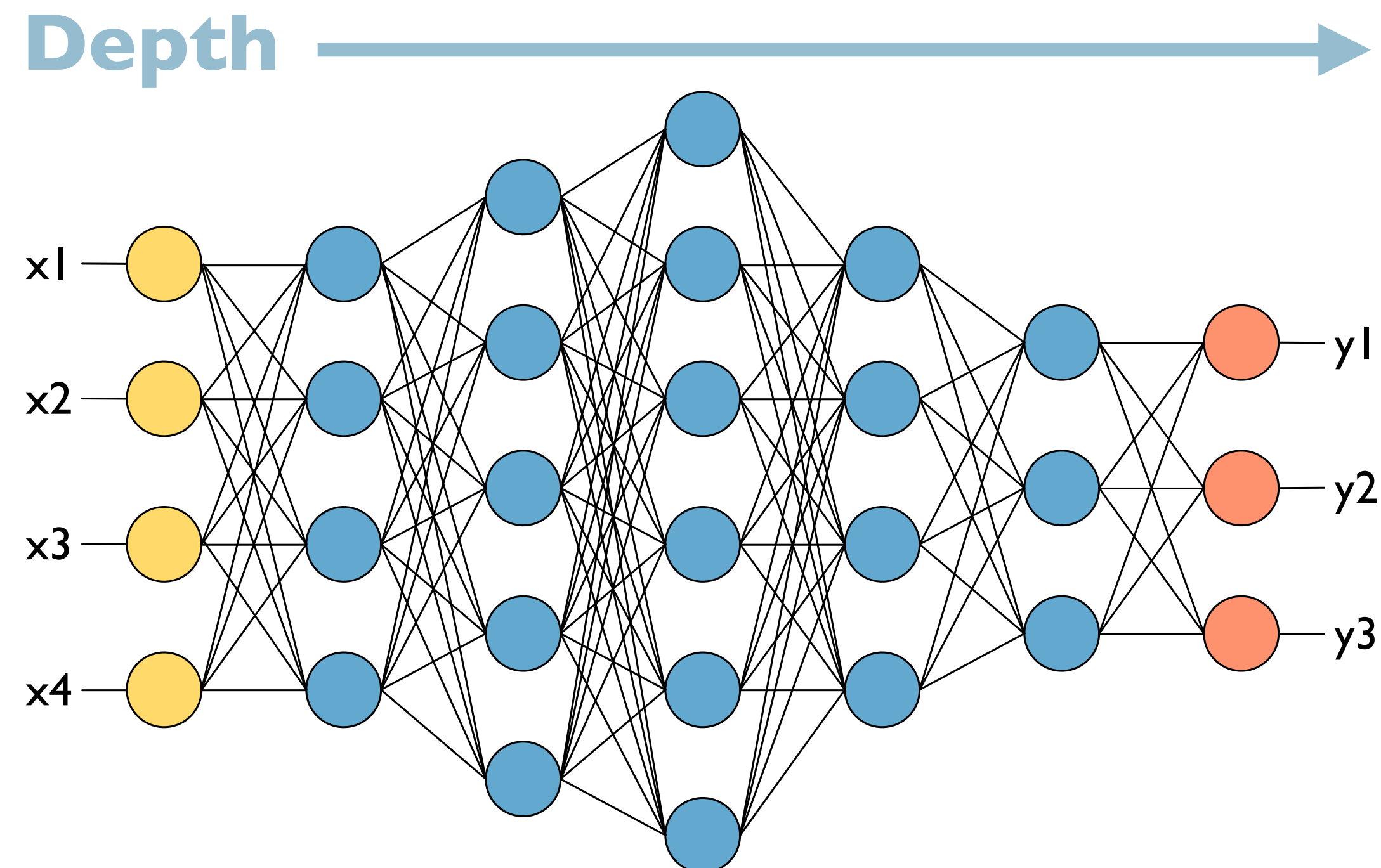# Shallow vs. Deep Processing

- Shallow processing (LING 570)

  - ***Less elaborate*** linguistic representations

    - Usually relies on surface forms (e.g. words)

  - Examples: HMM POS-tagging; FST morphology

- Deep processing (LING 571)

  - Relies on ***more elaborate*** linguistic representations

    - Deep syntactic analysis (Parsing)

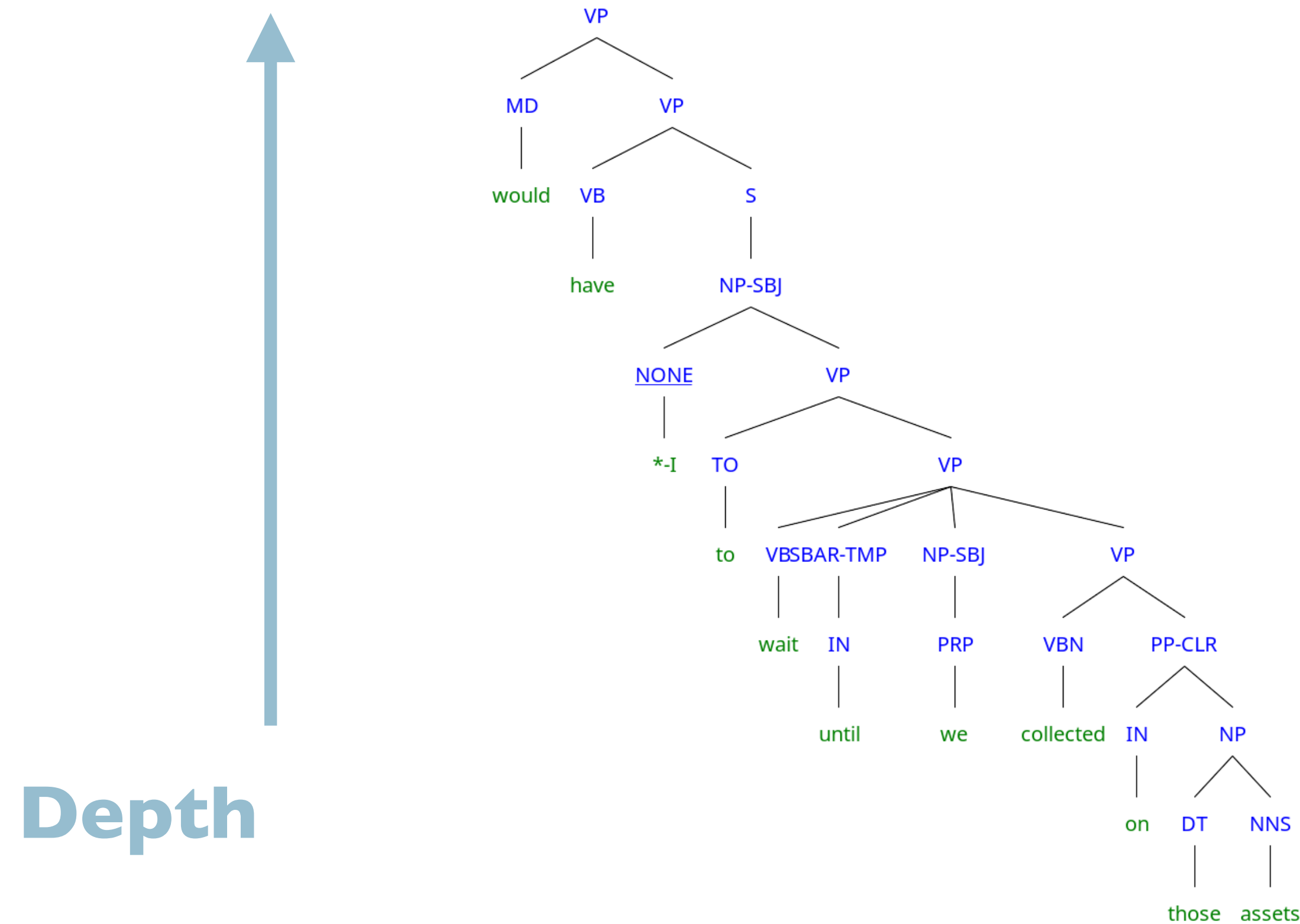   - Rich spoken language understanding (NLU)

Language Processing Pipeline
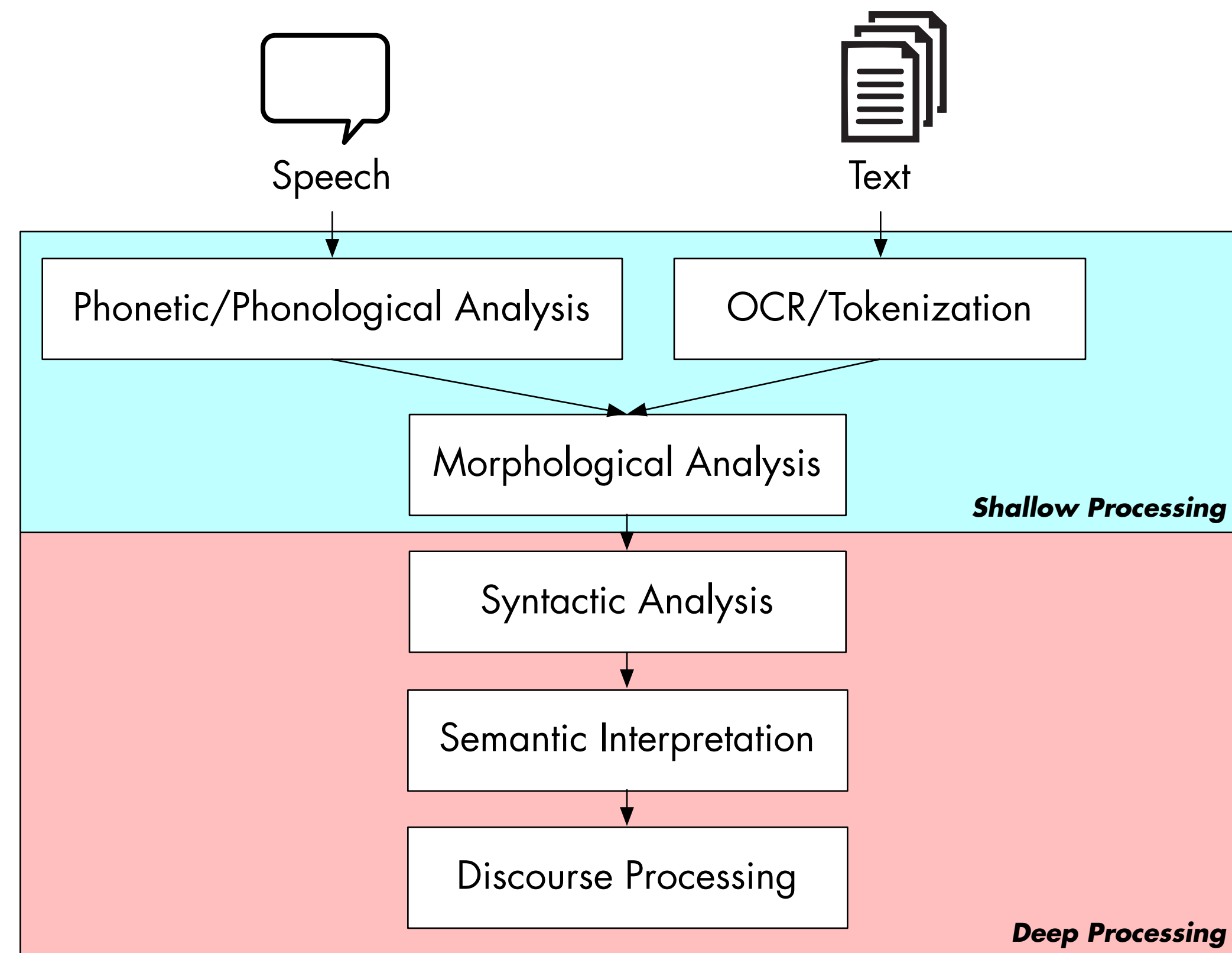
A Note On "Depth"

- "Deep" can be a tricky word these days in NLP

- "Deep Learning" ← "Deep Neural Networks"
  - Refers to depth of network architecture:

A Note On "Depth"

- "Deep Processing" ← "Depth" of Analysis (Amt. of Abstraction)

  - Depth of parse graph (tree) is one representation

Depth

A Note On "Depth"

- Depth of NN $\not\Rightarrow$ Depth of Analysis

- NNs are general function approximators
  - can be used for "shallow" analysis:
    - POS tagging, chunking, etc.
  - Can *also* be used for "deep" analysis:
    - Semantic role labeling
    - Parsing

- In both paradigms, graph depth aids, but $\not\Rightarrow$ abstraction

Cross-cutting Themes

- **Ambiguity**
  - How can we select from among alternative analyses?

- **Evaluation**
  - How well does this approach perform:
    - On a standard data set?
    - As part of a system implementation?

- **Multilinguality**
  - Can we apply the same approach to other languages?
  - How much must it be modified to do so?

Ambiguity: POS

- "I made her duck."

- Could mean…
  - I caused her to duck down.
  - I made the (carved) duck she has.
  - I cooked duck for her.
  - I cooked a duck that she owned.
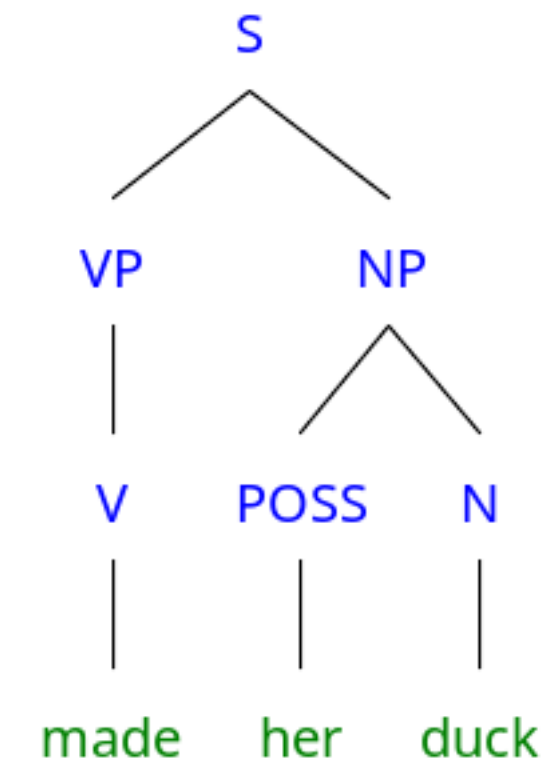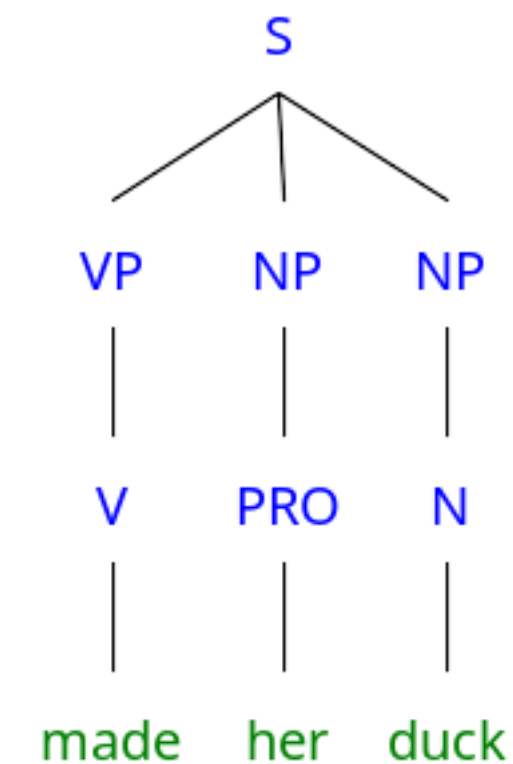 - I magically turned her into a duck.

VERB

POSS

NOUN

PRON

Ambiguity: Syntax

- "I made her duck."

- Could mean…
  - I made the (carved) duck she has

  - I cooked a duck for her

Ambiguity: Semantics

"I made her duck."

I caused her to duck down	*made* = [AG] **cause** [TH] [to_do_sth]
I cooked duck for her	*made* = [AG] **cook** [TH] for [REC]
I cooked the duck she owned	*made* = [AG] **cook** [TH]
I made the (carved) duck she has	*made* = [AG] **sculpted** [TH] *duck* = **duck-shaped-figurine**
I magically turned her into a duck	*made* = [AG] **transformed** [TH] *duck* = **animal**

Ambiguity

- Pervasive in language

- Not a bug, a feature! (Piantadosi et al 2012)

- *"I believe we should all pay our tax bill with a smile. I tried—but they wanted cash."*

- What would language be like without ambiguity?

Ambiguity

- Challenging for computational systems

- Issue we will return to again and again in class.

Course Information

- Website is main source of information: https://www.shane.st/teaching/571/aut20/

  - slides, office hours, resources, etc

- Canvas: lecture recordings, homework submission / grading

  - Communication!!! Please use the discussion board for questions about the course and its content.

  - Other students have same questions, can help each other.

  - May get prompter reply. The teaching staff will not respond outside of normal business hours, and may take up to 24 hours.

Syntax Crash Course

LING 571 — Deep Processing Techniques for NLP
September 30, 2020
Shane Steinert-Threlkeld

Roadmap

- **Sentence Structure**
  - More than a bag of words

- **Representation**
  - Context-free Grammars
    - Formal Definition

Applications

- Shallow techniques useful, but limited

- Deeper analysis supports:

  - Grammar checking — and teaching

  - Question-answering

  - Information extraction

  - Dialogue understanding

  - …

Grammar and NLP

- "Grammar" in linguistics is **NOT** prescriptive high school grammar

  - Explicit rules

  - "Don't split infinitives!" etc.

- "Grammar" in linguistics **IS**:

  - How to capture structural knowledge of language as a native speaker would have

  - Largely implicit

  - Learned early, naturally

More than a Bag of Words

- Sentences are structured

- Choice of structure can impact:
  - Meaning:
    - *Dog bites man.* **vs.** *Man bites dog.*
  - Acceptability:
    - *Colorless green ideas sleep furiously.*
    - ** Colorless sleep ideas furiously green.*
    - ** Dog man bites*

Constituency

- **Constituents**: basic units of sentences

  - Word or group of words that act as a single unit syntactically

- **Phrases**:

  - Noun Phrase (NP)

  - Verb Phrase (VP)

  - Prepositional Phrase (PP)

  - …

- Single unit: type determined by "head"

  - e.g. **N** heads NP

Representing Sentence Structure

- Basic Units

  - Phrases (**NP**, **VP**, etc…)

  - Capture <u>constituent</u> structure

- Subcategorization

  - (**NP**-**SUBJ**, **VP**-**INTRANS**, etc…)

  - Capture <u>argument</u> structure

    - Components expected by verbs

- Hierarchical

Representation: Context-free Grammars

- CFGs: 4-tuple

  - A set of **terminal** symbols: Σ

    - [think: words]

  - A set of **nonterminal** symbols: *N*

    - [think: phrase categories]

  - A set of **productions** *P*:

    - of the form $A \rightarrow \alpha$

    - Where A is a non-terminal and $\alpha \in \{\Sigma \cup N\}*$

  - A **start** symbol $S \in N$

# Representation: Context-free Grammars

- Altogether a grammar defines a language *L*

  - $L = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$

    - The language *L* is the set of all words in which:

    - $S \Rightarrow^* w$: *w* can be *derived* starting from *S* by some sequence of productions

CFG Components

- **Terminals**:

  - Only appear as leaves of parse tree (hence the name)

  - Right-hand side of productions (RHS)

  - Words of the language

    - *cat, dog, is, the, bark, chase…*

- **Non-terminals**

  - Do not appear as leaves of parse tree

  - Appear on left or right side of productions

  - Represent constituent phrases of language

    - NP, VP, S[entence], etc…

Representation: Context-free Grammars

- Partial example:
  - Σ: *the, cat, dog, bit, bites, man*
  - *N*: NP, VP, Nom, Det, V, N, Adj
  - *P*:
    - S→NP VP;
    - NP→Det Nom;
    - Nom→N Nom I N;
    - VP→V NP;
    - N→*cat*; N→*dog*; N→*man*;
    - Det→*the*;
    - V→*bit*; V→*bites*
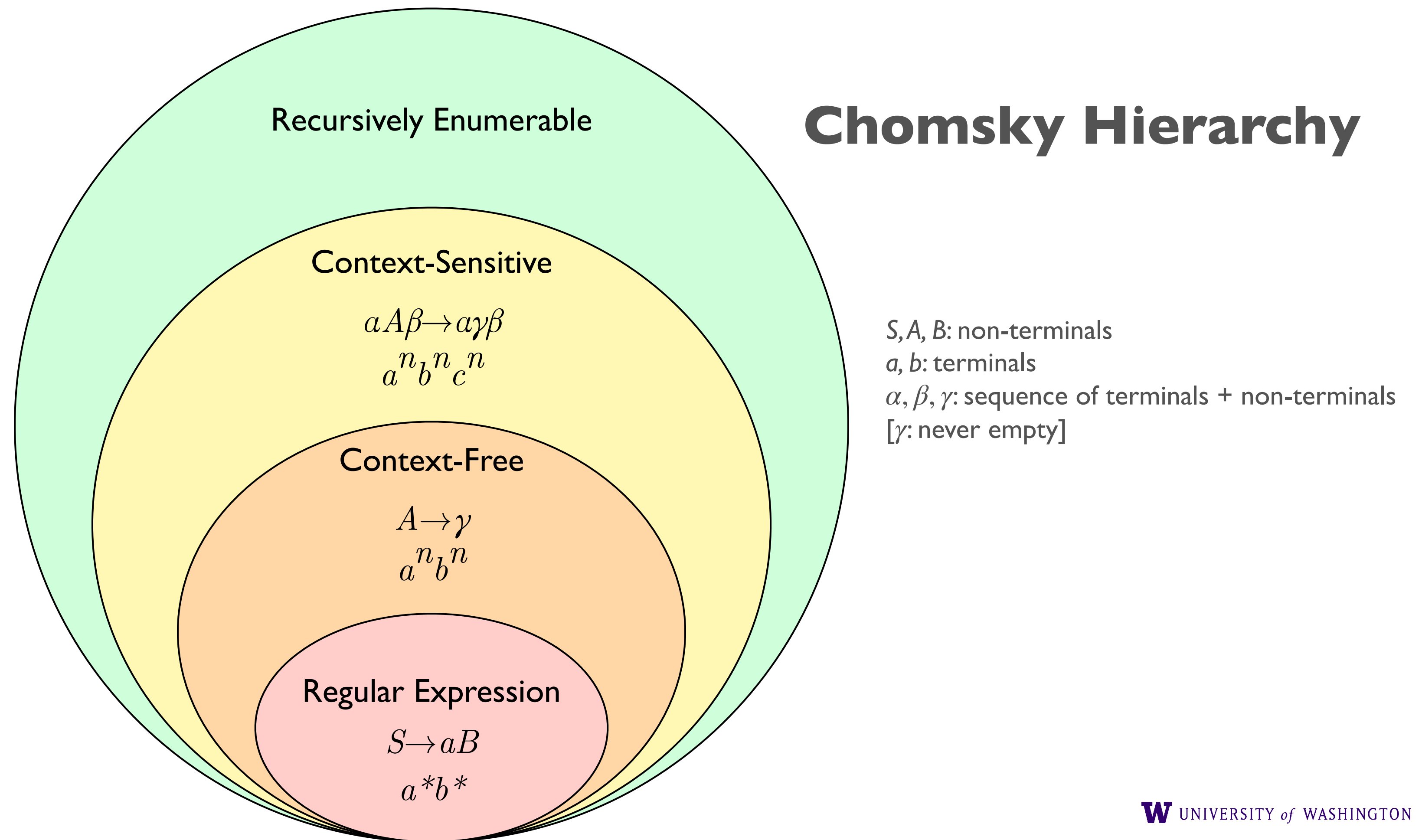 - *S*: S
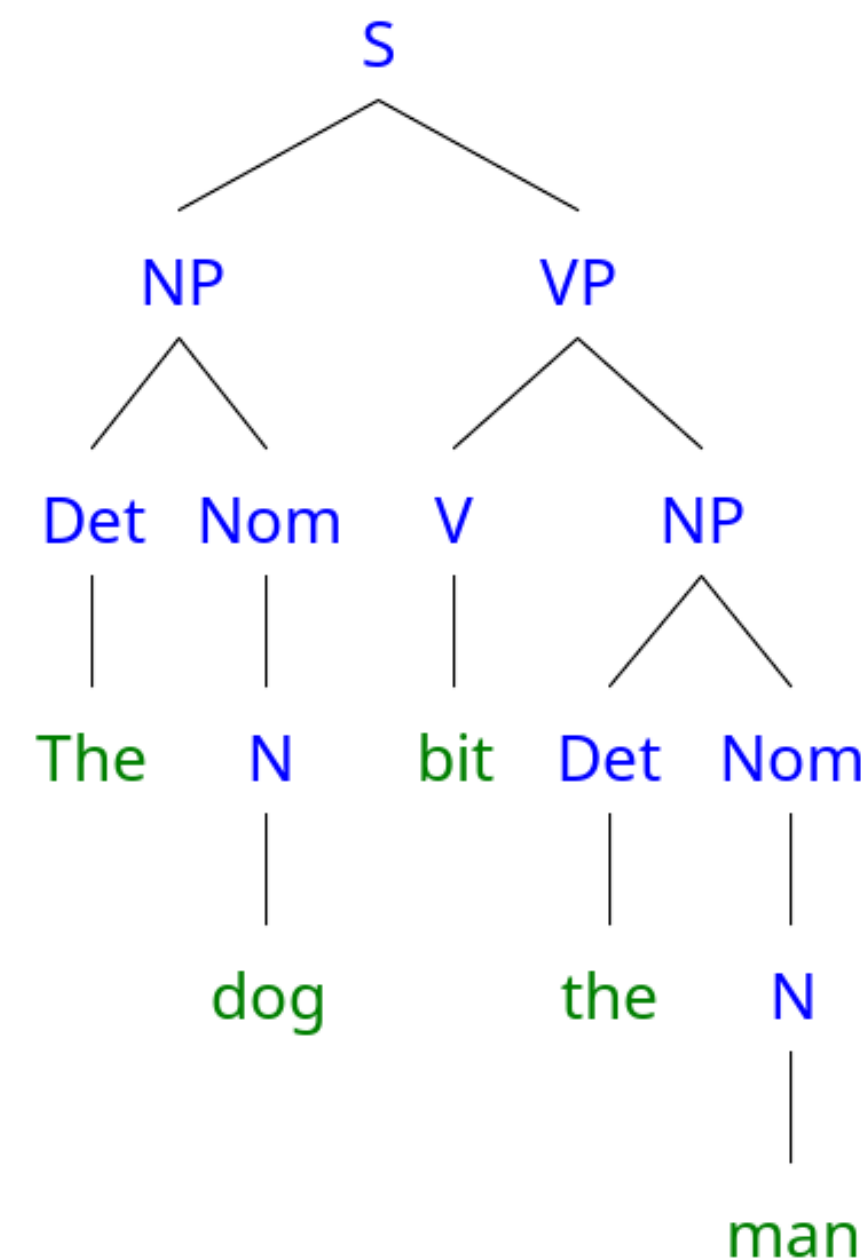
Parsing Goals

- Acceptance

  - Legal string in language?

    - Formally: rigid

    - Practically: degrees of acceptability

- Analysis

  - What structure produced the string

    - Produce one (or all) parses for the string

- Will develop techniques to produce analyses of sentences

  - Rigidly accept (with analysis) or reject

  - Produce varying degrees of acceptability

Sentence-level Knowledge: Syntax

- Different models of language that specify the ***expressive power*** of a formal language

Chomsky Hierarchy

Recursively Enumerable

Context-Sensitive

$$\alpha A \beta \rightarrow \alpha \gamma \beta$$
$$a^n b^n c^n$$

Context-Free

$$A \rightarrow \gamma$$
$$a^n b^n$$

Regular Expression

$$S \rightarrow aB$$
$$a*b*$$

S, A, B: non-terminals
a, b: terminals
α, β, γ: sequence of terminals + non-terminals
[γ: never empty]

Representing Sentence Structure

- Why not just Finite State Models (Regular Expressions)?

  - Cannot describe some grammatical phenomena

  - Inadequate expressiveness to capture generalization

Representing Sentence Structure: Center Embedding

- **Regular Language**: $A \rightarrow w; A \rightarrow w^*B$

- **Context-Free**: $A \rightarrow \alpha A \beta$ (e.g.)
  - Allows recursion:
    - The luggage arrived
    - The luggage that the passengers checked arrived
    - The luggage that the passengers whom the storm delayed checked arrived

Recursion in Grammar
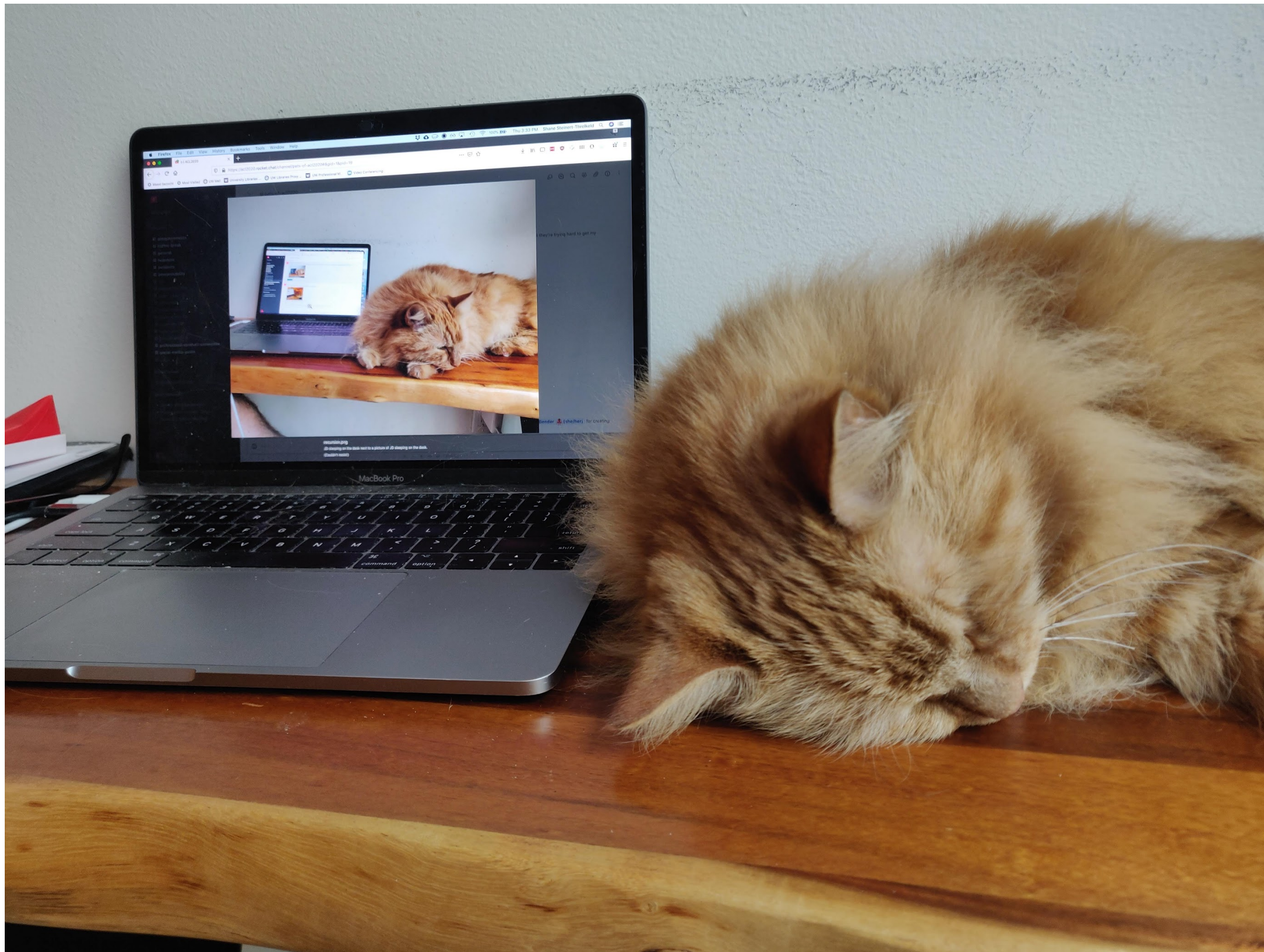
This is JD lying on the desk next to a picture of JD lying on the desk next to a picture of JD lying on the desk.

Exercise: write a toy grammar for producing this sentence!

Is Context-Free Enough?

- Natural language not finite state

- …but do we need context-sensitivity?

  - Many articles have attempted to demonstrate we do

  - …many have failed.

- Solid proof for Swiss German: *Cross-Serial Dependencies* (Shieber, 1985)

  - $a^i b^j c^i d^j$

$$x_1 \quad x_2 \quad \cdots \quad x_n \quad \cdots \quad y_1 \quad y_2 \quad \cdots \quad y_n$$
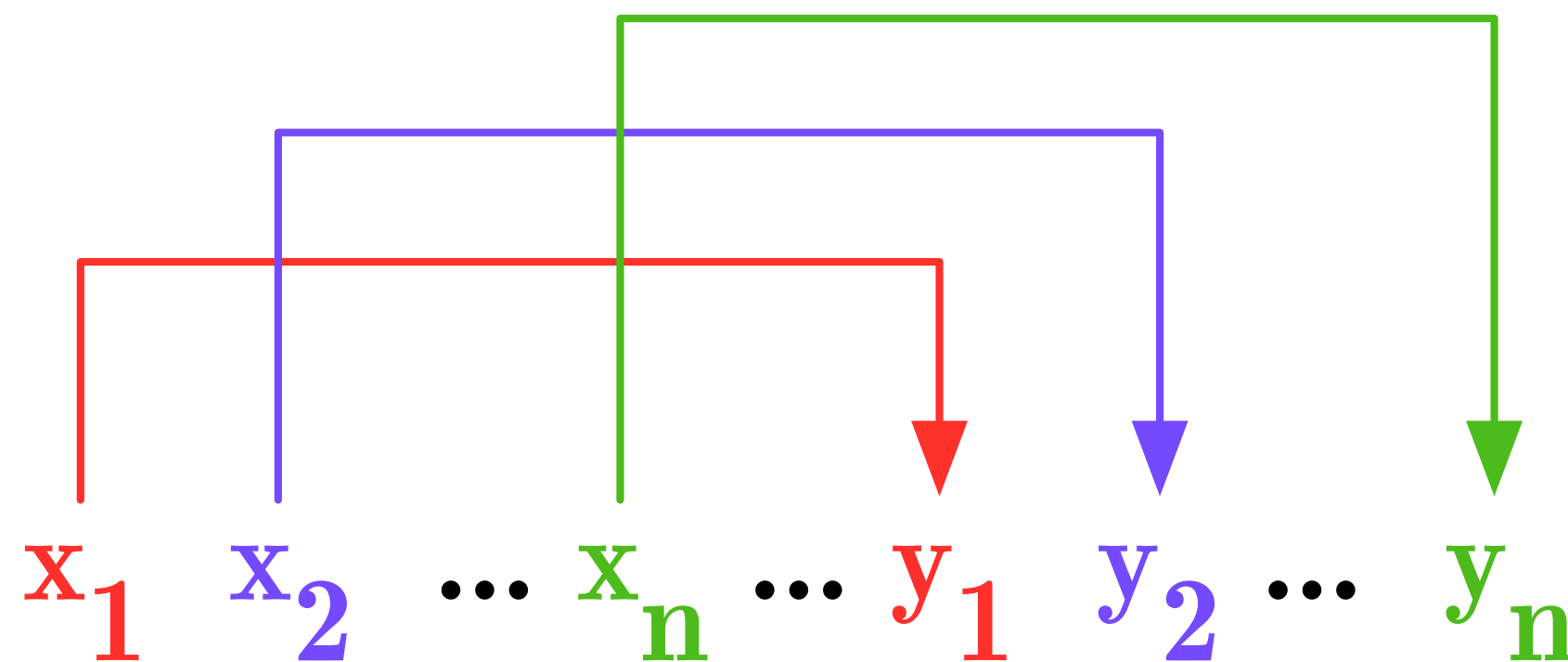
Context-Sensitive Example

- Verbs and their arguments must be ordered **_cross-serially_**

  - Arguments and verbs must match

...mer   em Hans   s   huus   hälfed   aastriiche.
...we   *Hans (DAT)*   *the*   *house.ACC*   *help*   *paint*
"We helped hans paint the house."

...mer   d'chind   em Hans   s   huus   haend wele   laa   hälfed   aastriiche.
...we   *the children*   *Hans (DAT)*   *the*   *house.ACC*   *have*   *wanted.to let*   *help*   *paint*
"We wanted to let the children help Hans paint the house."

Questions so far?

HW#1 & Getting Started

LING 571 — Deep Processing Techniques for NLP
September 30, 2020
Shane Steinert-Threlkeld

Department Cluster

- Assignments are **required** to run on department cluster

  - If you don't have a cluster account, request one ASAP!

    - Link to account request form on Canvas or below:

    - vervet.ling.washington.edu/db/accountrequest-form.php

- You are not required to develop on the cluster, but code must run on it

- **Reminder:** *All but most simple tasks must be run via Condor*

Condor

- Parallel computing management system

- All homework will be run via condor

- See documentation on CLMS wiki for:

  - Construction of condor scripts

  - Link also on course page under "Course Resources"

Programming

- For most assignments, we will be using NLTK in Python.

- For assignments where NLTK is not required, you *may* choose to use a different programming language.

NLTK

- **N**atural **L**anguage **T**ool**K**it (NLTK)
  - Large, integrated, fairly comprehensive
    - Stemmers
    - Taggers
    - Parsers
    - Semantic analysis
    - Corpus samples
    - …& More
  - Extensively documented
  - Pedagogically Oriented
    - Implementations Strive for Clarity
    - …sometimes at the expense of efficiency.

NLTK

- nltk.org
  - Online book
  - Demos of software
  - How-Tos for specific components
  - API information, etc.

Python & NLTK

- NLTK is installed on the Cluster

  - Use Python 3.4+ with NLTK

  - **N.B.:** Python 2.7 is default

    - Use: `python3` to run, not `python`

    - More versions in `/opt/python-*/bin/`

    - You can make a personal alias, but your bash scripts will not run in your personal environment, so keep that in mind (e.g. use full path).

- Data is also installed:

  - /corpora/nltk/nltk-data

- Written in Python

  - Some introductions at:

    - **python.org, docs.python.org**

Python & NLTK

● Interactive mode allows experimentation, introspection:

```
patas$ python3
>>> import nltk
>>> dir(nltk)
['AbstractLazySequence', 'AffixTagger', 'AlignedSent',
'Alignment', 'AnnotationTask', 'ApplicationExpression',
'Assignment', 'BigramAssocMeasures', 'BigramCollocationFinder',
'BigramTagger', 'BinaryMaxentFeatureEncoding',…
>>> help(nltk.AffixTagger)
```

Turning In Homework

- Will be using Canvas' file submission mechanism
  - Quick how to at:
    https://community.canvaslms.com/docs/DOC-10663-421254353

- Homeworks due on **Wednesday** nights

- 11:00 PM, Pacific Time

- Generally, each assignment will include:
  - `readme.{txt|pdf}`
  - `hwX.tar.gz`
    - Where "X" is the assignment number
    - `tar -cvzf hwX.tar.gz <hw_path>`

HW #1

- Read in sentences and corresponding grammar

- Use NLTK to parse those sentences

- Goals:

  - Set up software environment for rest of course

  - Get familiar with NLTK

  - Work with parsers and CFGs

HW #1: Useful Tools

- Loading data:
  - **`nltk.data.load(resource_url)`**
    - Reads in and processes formatted CFG/FCFG/treebank/etc
    - Returns a grammar from CFG
    - **examples:**
      `nltk.data.load('grammars/sample_grammars/toy.cfg')`
      `nltk.data.load('file://' + my_grammar_path)`
    - (NB: absolute path!)

- Tokenization:
  - **`nltk.word_tokenize(mystring)`**
    - Returns array of tokens in string

HW #1: Useful Tools

- Parsing:

  - **parser = nltk.parse.EarleyChartParser(grammar)**
    - Returns parser based on the grammar

  - **parser.parse(token_list)**
    - Returns iterator of parses:

```
>>> for item in parser.parse(tokens):
>>>     print(item)


(S (NP (Det the) (N dog)) (VP (V chased) (NP (Det the) (N cat))))
```