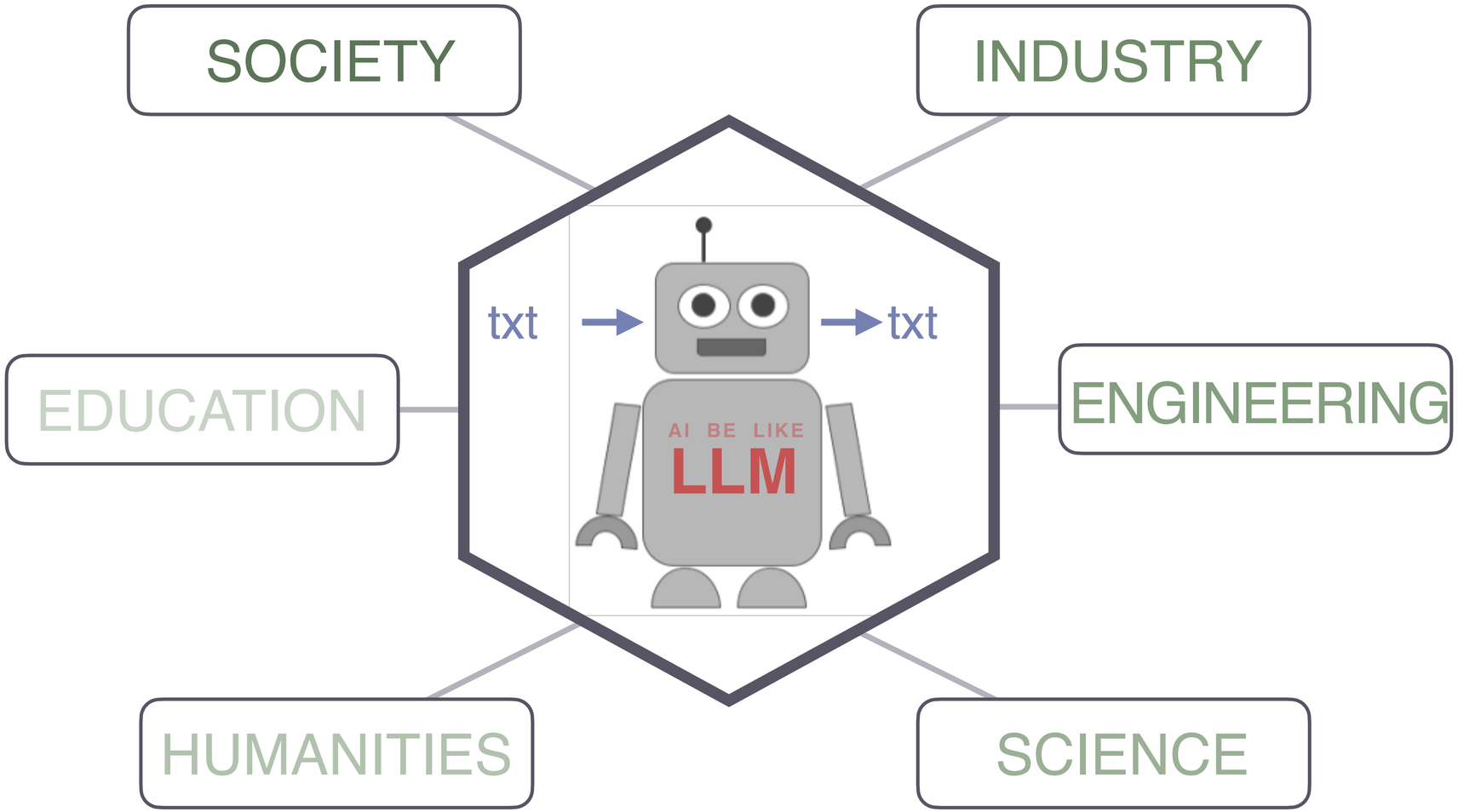
Understanding Large Language Models

Carsten Eickhoff, Michael Franke and Polina Tsvilodub

Session 01: Introduction

Motivation

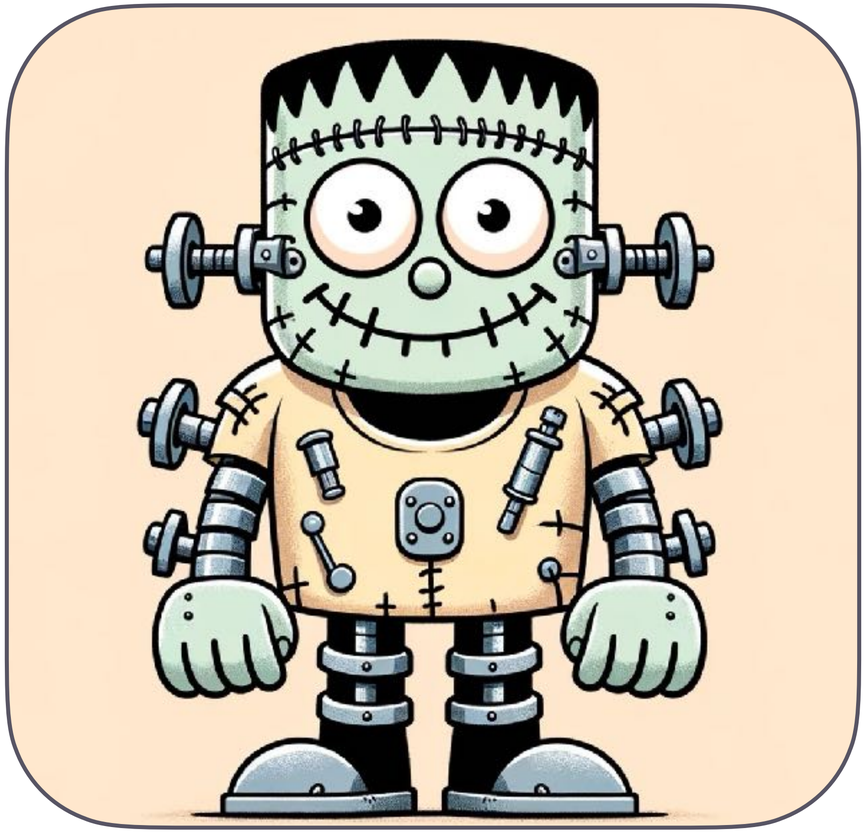

SOCIETY
INDUSTRY
txt
txt
AI BE LIKE
LLM
EDUCATION
ENGINEERING
HUMANITIES
SCIENCE

We built a creature

but what does it do? how and why?

TRAINING ✓

INFERENCE ✓



BEHAVIOR ✗

MECHANISMS ✗

Value proposition

I understand better how NLP is used in industry, & know whether I want to work with it

I can propose a project topic that this is relevant and doable (by me.)

I'm interested in modern NLP & want to know more about it!

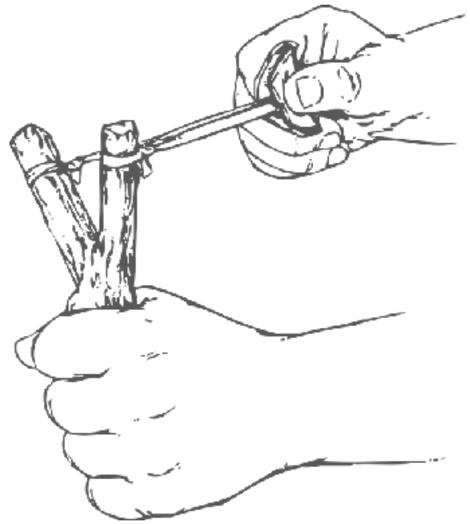
I can pick a random paper on LLMs & understand most of it.

from here

to there

Main learning goals

towards a better understanding of language models



1. **know about LM architectures, training and inference methods**
   a. be able to **implement and train small LMs**
   b. be able to **implement different decoding methods** for trained LMs
   c. understand the workings of **fine-tuning** and **LM agents**

2. **increase skill for using pretrained LMs for practical applications**
   a. developed sharper **intuitions about in-context learning / prompt engineering**
   b. acquire an acute sense of **critical distrust** based on knowledge of LM architectures

3. **become familiar with the current literature on assessment & interpretability**
   a. know about state-of-the-art results in **targeted assessment** and **machine psychology**
   b. be able to implement simple **attribution and probing methods**
   c. understand how more **sophisticated methods of mechanistic interpretability** work

4. **develop confidence to critically evaluate SOTA language technology and its implications**
   a. build intuitions about **implications for academia, education, industry, society**
   b. anticipate potential **risks of future language technology**

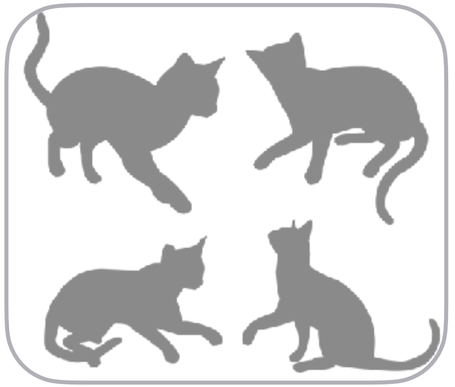
Course organization

Team

Carsten Eickhoff
lectures

Michael Franke
lectures

Polina Tsvilodub
hands-on sessions



Jan-Felix Klumpp
Svenja Schulze
Christoph Stengel
Daniel Stuhlinger
support

Course requirements

- standard track: 6 CP
  - 3-4 homework sets
    - individual submissions
  - final in-class exam
    - targets conceptual understanding, not programming abilities
  - passing requirements: passing grade for both parts HW + exam
  - final grade: 2/3 * HW grade + 1/3 * final
  - extra oral exam for suspicious cases
- additional project: + 3 CP
  - small-ish group project
    - topic suggestions to be disseminated, own suggestions welcome
  - grade contribution: 1/3 project, 2/3 grade from standard track

Schedule

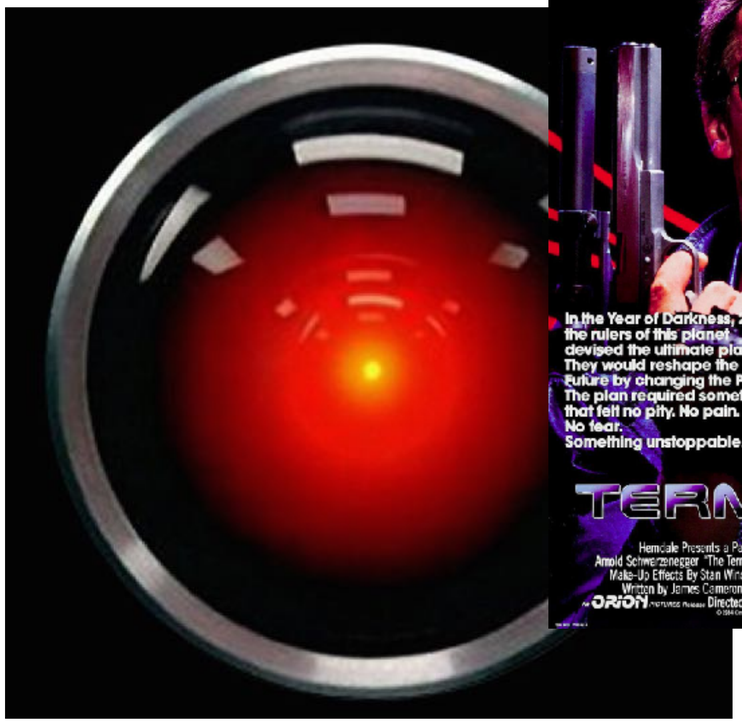
preliminary plan / changes possible

	date	topic	HW release	HW due
1	2024-04-16	intro / orga / LM history / motivation		
2	2024-04-23	ANNs & small LMs		
3	2024-04-30	LSTMs, transformers	HW1	
	2024-05-07	no class		
4	2024-05-14	LLMs, in-context learning / prompt engineering	HW2	HW1
	2024-05-21	Pentecost		
5	2024-05-28	supervised fine-tuning & RLHF		
6	2024-06-04	agents & assistants	HW3	HW2
7	2024-06-11	probing / attribution		
8	2024-06-18	behavioral assessments (general / SyntaxGym ...)		
9	2024-06-25	implications for linguistics / cogsci / society?	HW4	HW3
10	2024-07-02	mechanistic interpretability		
11	2024-07-09	TBD		HW4
12	2024-07-16	spill-over / wrap-up / discussion / exam		
13	2024-07-23	**in-class exam**		

A short history of language modeling

NLP Motivation

- Popular Fiction
- Machines that speak



NLP Motivation

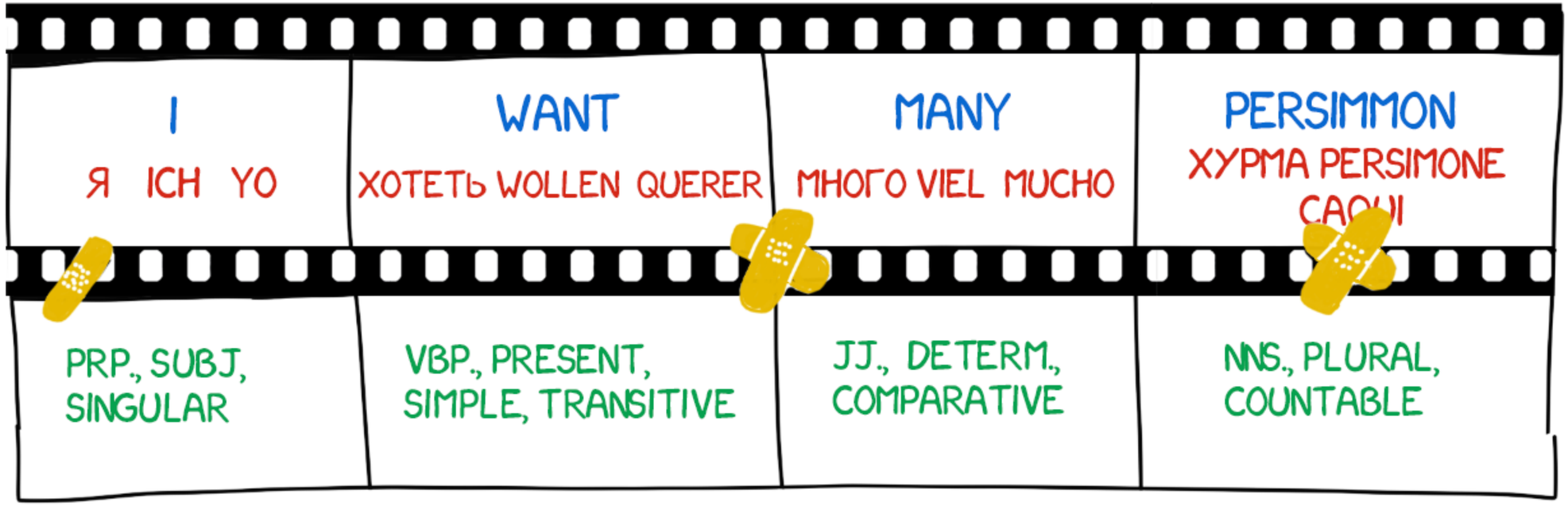
- Historically: Machine Translation (MT)
- DARPA funding patterns speak volumes
- Now much broader
  - Named Entity Recognition
  - Tagging
  - Question Answering
  - Information Retrieval
  - Chatbots
  - ...

NLP History

NLP History

Troyanskij's Machine (1933)

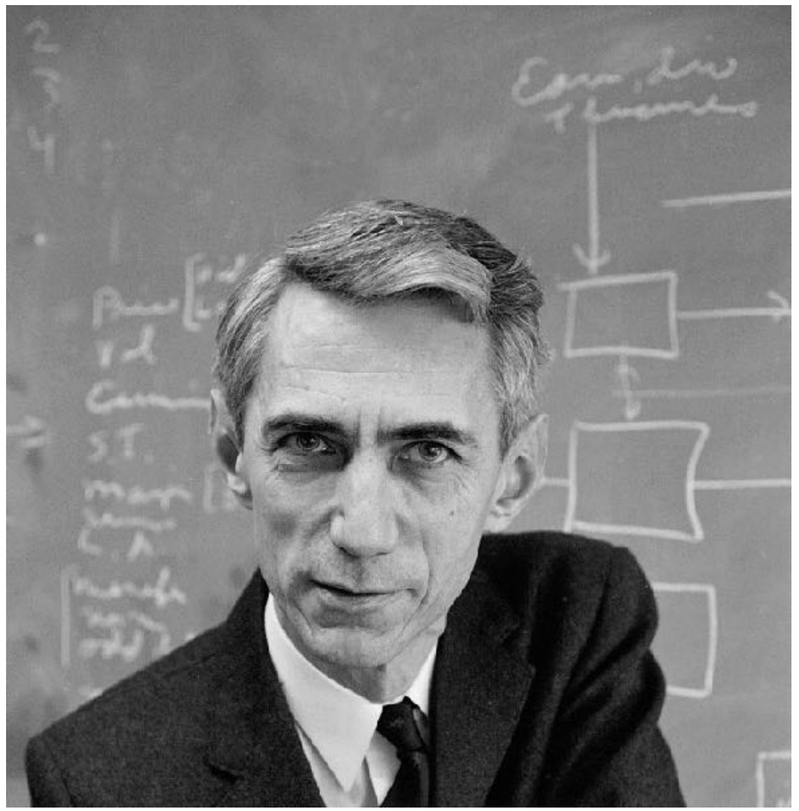
- Peter Troyanskij
- “Machine for the selection and printing of words when translating from one language to another”
- Enter a foreign language sentence word-by-word
- Supply POS information
- Machine photographs multiple translations onto paper
- The result: Nobody cared



NLP History

Language as Signal Streams (1950)

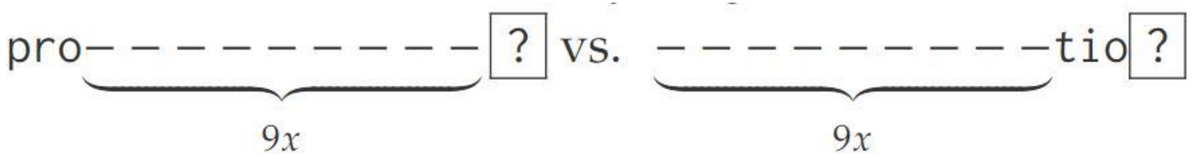
- Information theoretic view of language
  - Text as a string of signals (characters)
  - No need to define “units” of language
  - Claude Shannon
- Predict next character a in the string based on history h



$$p(a_1, a_2, \ldots, a_n) = p(a_1) * p(a_2|a_1) * p(a_3|a_1 a_2) * \ldots$$

$$= \prod_{i=1}^{n} p(\underbrace{a_i}_{\text{a}} | \underbrace{a_1, \ldots, a_{i-1}}_{\text{h}})$$

Markov Chains

- In practice, the full-length history h becomes unwieldy
- Instead, we limit the history length
- 

- Nearby characters are more informative
- Markov Chain
  - Reduce context to previous k characters
  - k-th order Markov assumption

$$p(a_n|a_{n-1}\dots a_{n-k}) \approx p(a_n|a_{n-1}\dots a_2 a_1)$$

k-th Order Markov Assumption

We can make predictions about the future based on just the past k observations just as well as if we had known the full history.

n-gram Models

- k-th order Markov assumption

$$p(w_{1:n}) = \prod_{i=1}^{n} p(w_i | w_{i-k:i-1})$$

- Only look at k previous words
- Also called: n-gram model ($n = k + 1$)
- Estimation
  - Estimate probabilities from frequencies (counts)
  - Count word sequences of length ($k + 1$)
  - $k = 0$: unigram; $k = 1$: bigram; $k = 2$: trigram

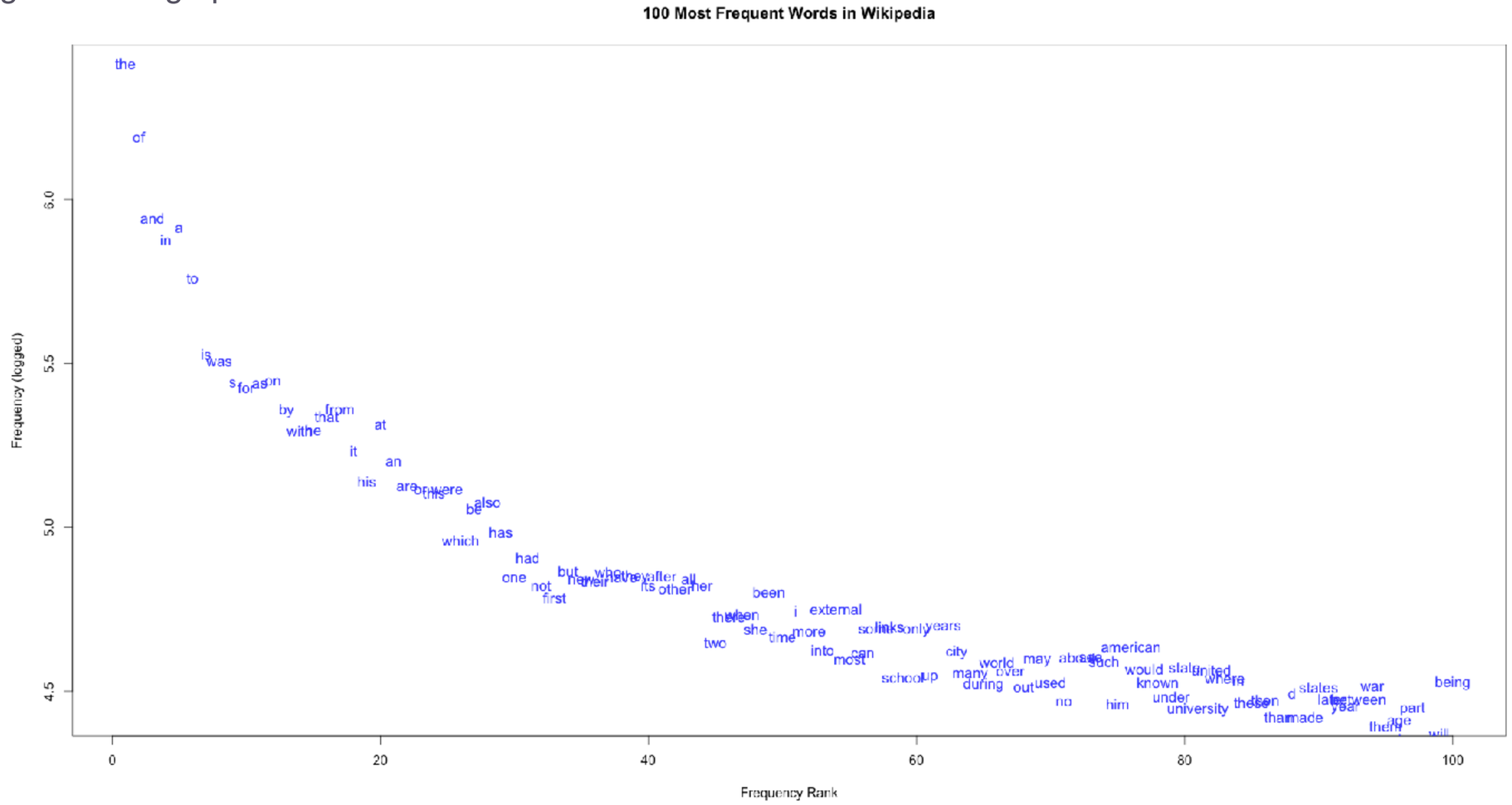
3–4-grams word

	Word	↓ Count
1	one of the	303,265
2	as well as	291,495
3	part of the	151,966
4	I do n't	148,894
5	a lot of	145,674
6	the United States	144,719
7	be able to	132,872
8	in order to	128,631
9	some of the	119,133
10	the end of	118,539

	Word	↓ Count
11	to be a	100,285
12	a number of	98,337
13	is one of	90,869
14	of the most	90,028
15	out of the	87,383
16	the use of	82,268
17	you want to	81,888
18	there is a	80,923
19	the number of	77,974
20	end of the	77,110

Zipf's Law

- Terms show up at different frequencies
- Only very few terms are highly frequent
- Most terms hardly ever appear
- Zipf’s Law governs large parts of NLP

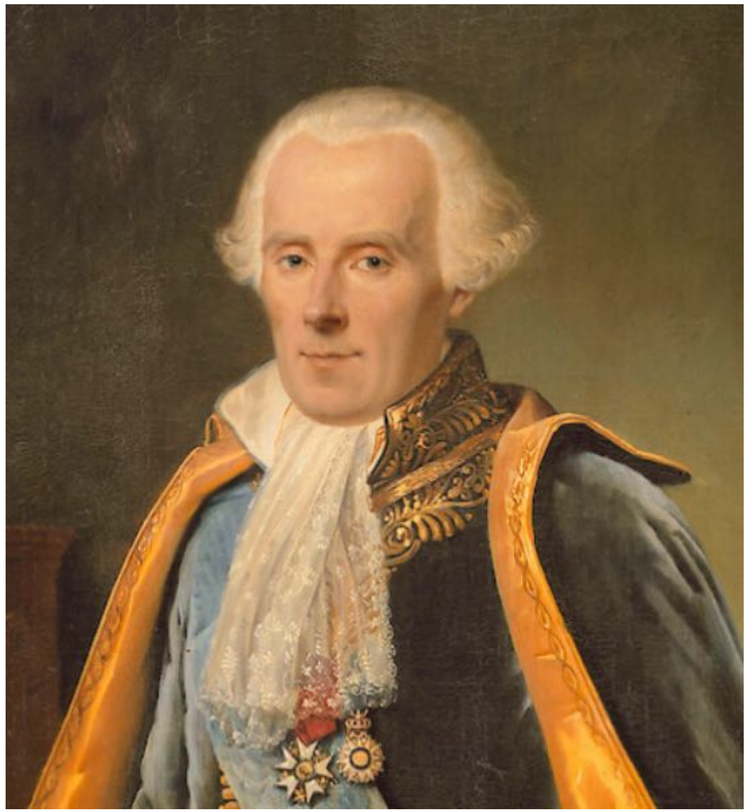


Smoothing

- 1799: Simon Laplace and the Sunrise Problem
- Idea: Add a small number to each observed frequency

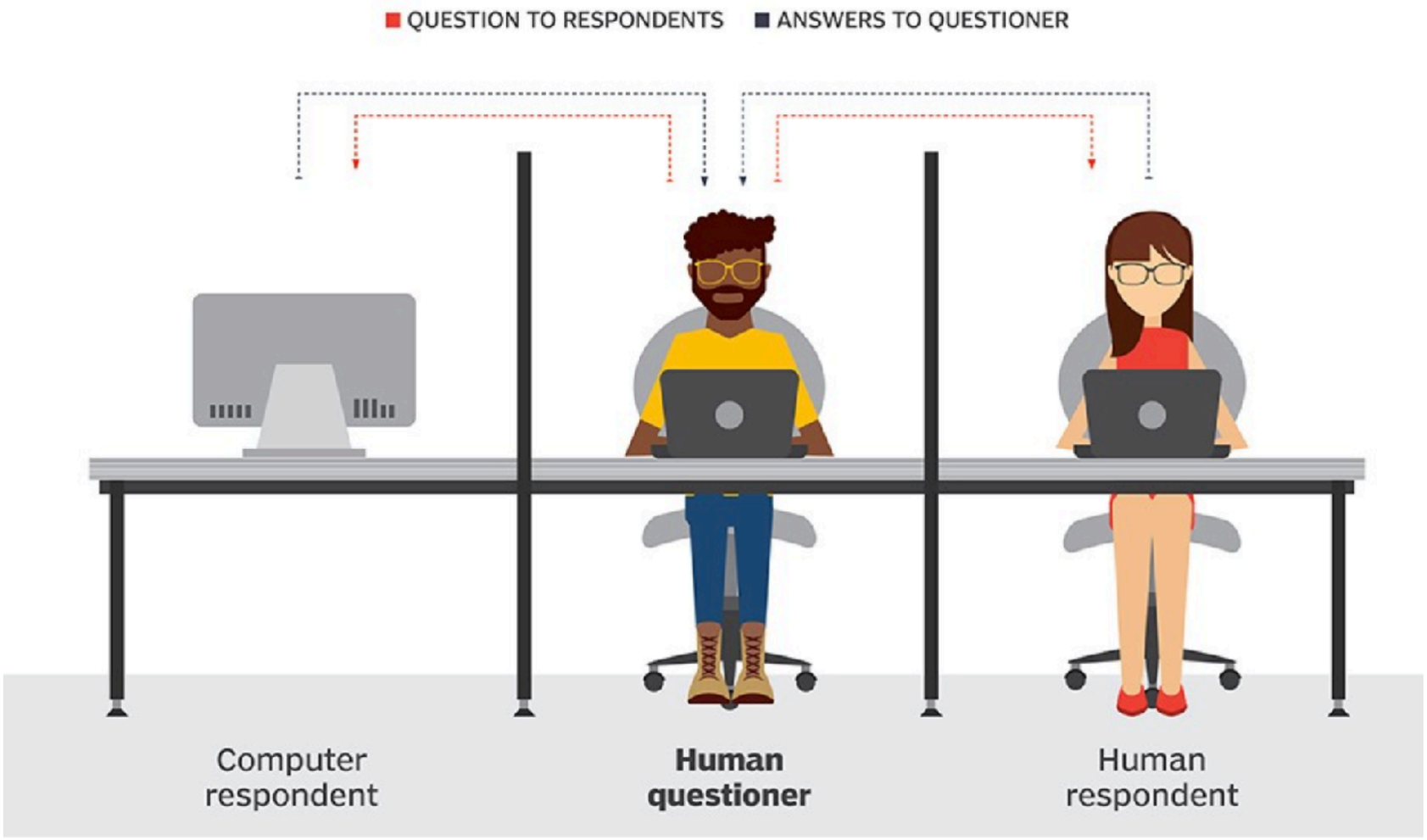
$$\hat{p}(w_i) = \frac{c(w_i) + \alpha}{N + V\alpha}$$

- Add-one Smoothing: $\alpha = 1$
- In practice often smaller quantities
- Retain relative difference between something $c(wi) = 1$ and nothing $c(wi) = 0$



The Turing Test (1950)

- Alan Turing
- "An Imitation Game"

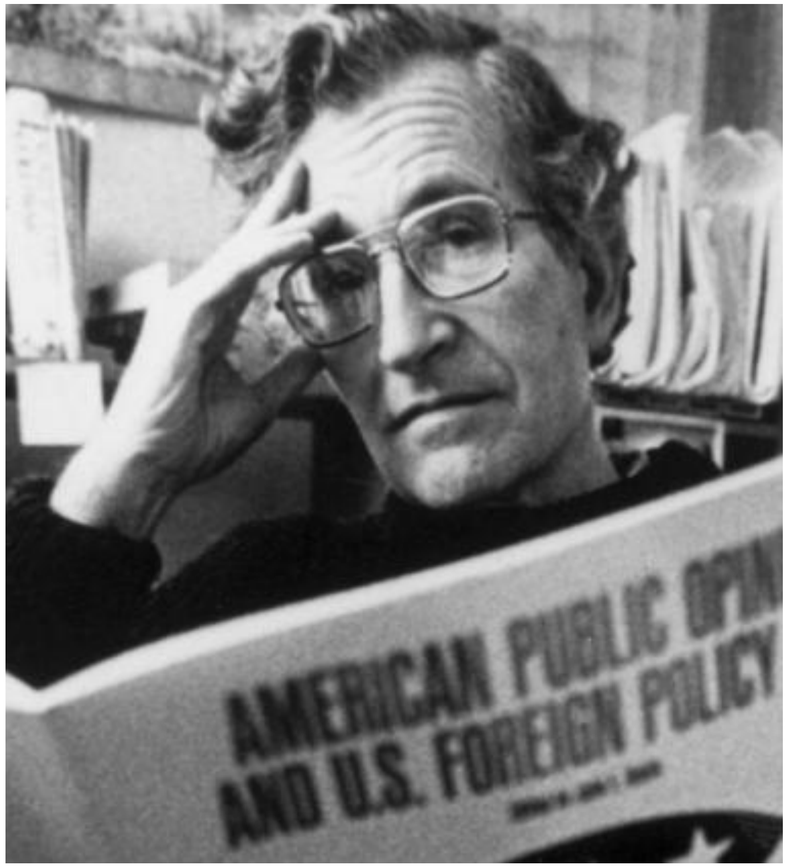


NLP History


1957
Chomsky
Syntactic Structures
1933
Troyanskij
1950
Turing Test
1950
Shannon
1956
Dartmouth
Conference

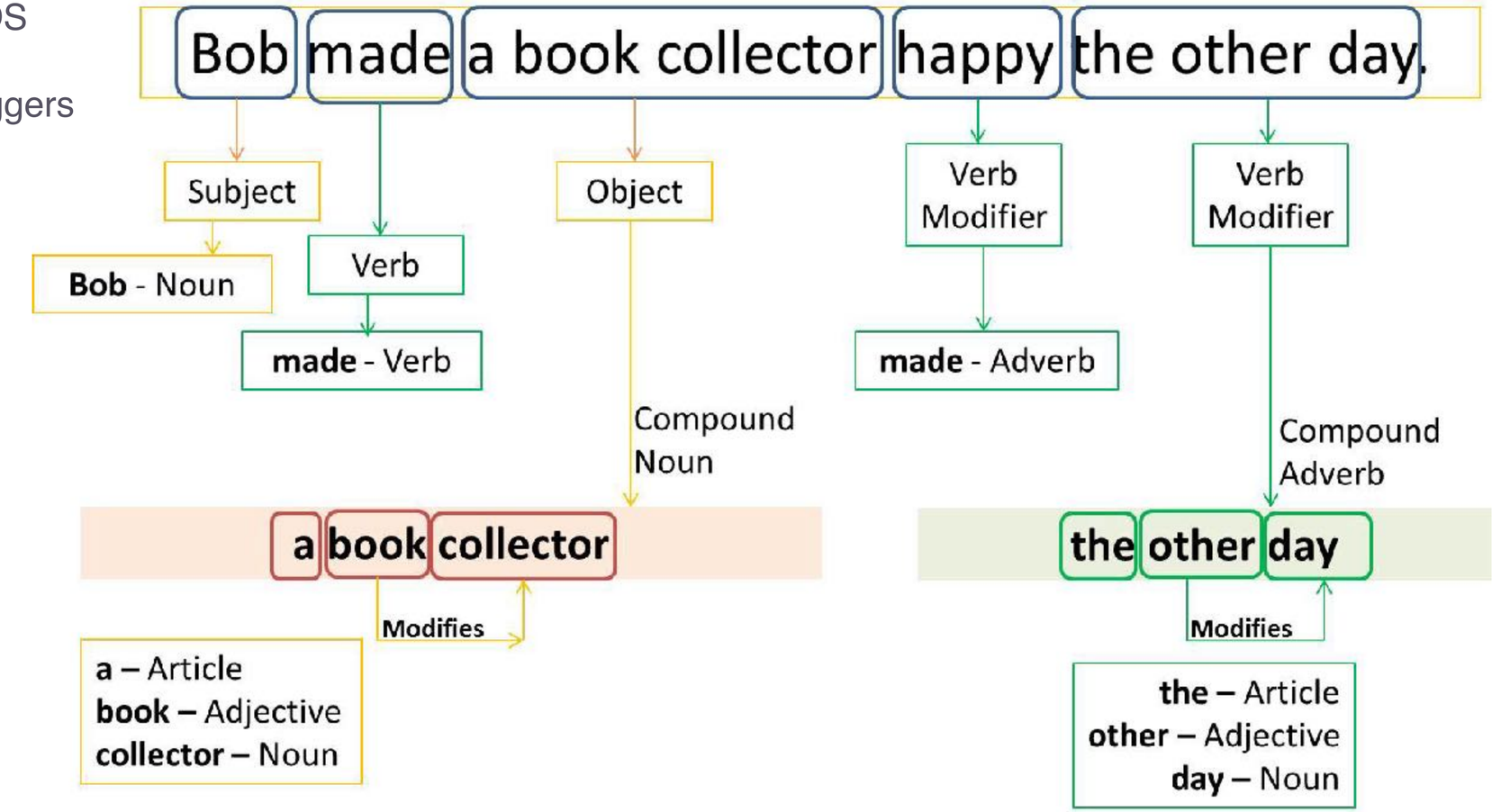
Formal Grammars (1950s)

- Computing with grammar
- Compose high-level symbols C from low-level ones A, B
- C → A B (Read: C is produced by A followed by B)
- Examples:
  - S ⟶ *Bob likes Alice*
  - S ⟶ PROPN *likes* PROPN
  - PROPN ⟶ *Alice* | *Bob* | *Eve*
  - S ⟶ PROPN *likes* N
  - N ⟶ *ice cream* | *pasta* | *Summer*
  - S ⟶ PROPN *likes* N *very much*
  - ...
- Realistic grammars have MANY rules...



Part-of-Speech Tagging

- To reduce ambiguity, annotate words with parts of speech
- No globally accepted definition
  - (Thrax of Alexandria, 100 B.C.) → 8 POS
  - (Brown Corpus, 1964) → 85 POS
  - (LOB Corpus, 1970-1981) → 148 POS
  - (Penn Treebank, 1993) → 45 POS
- Use annotated corpora to train POS taggers

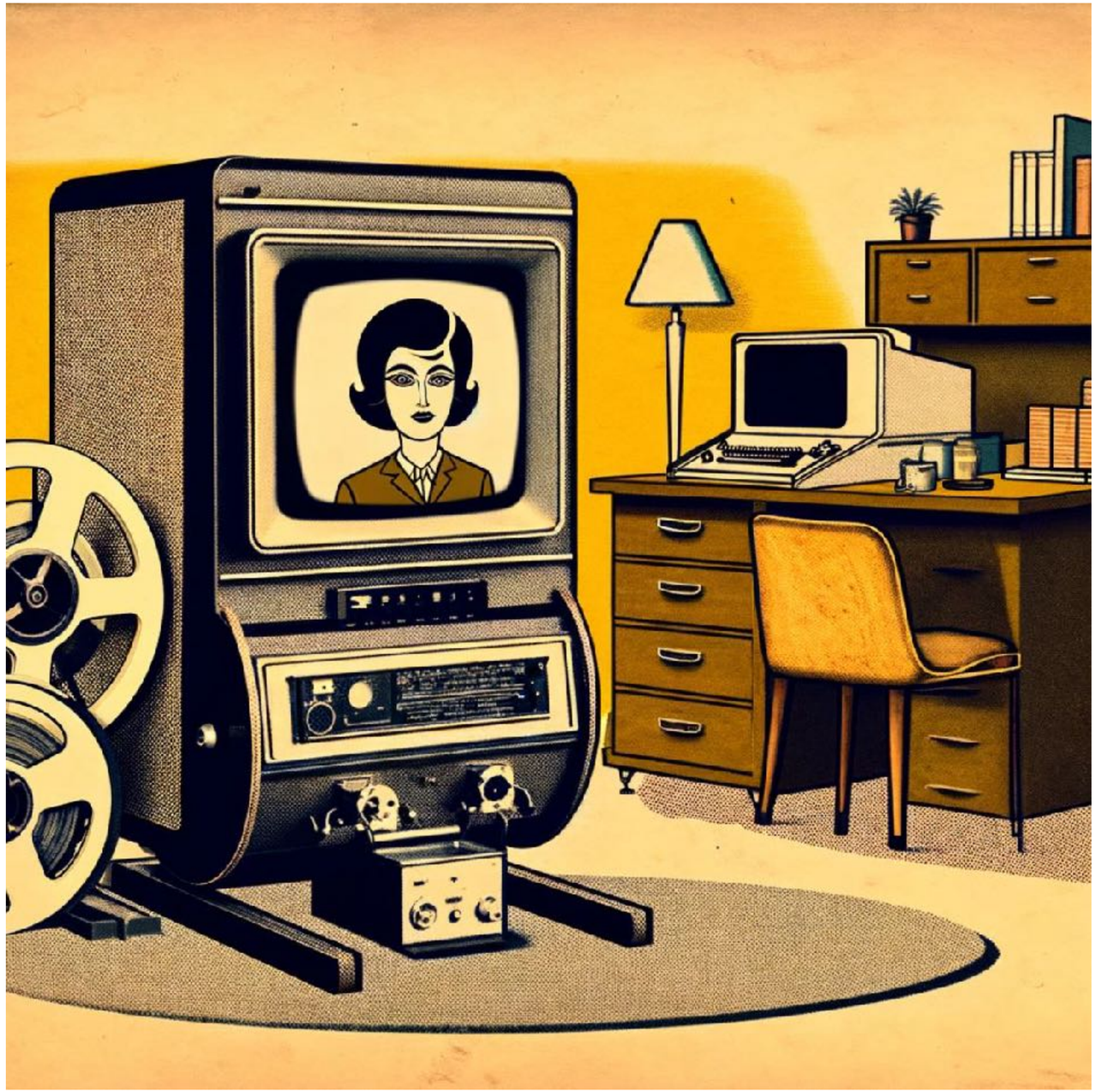


NLP History


1957
Chomsky
Syntactic Structures
1933
Troyanskij
1950
Turing Test
1971
Parry
1950
Shannon
1956
Dartmouth
Conference
1966
ALPAC
ELIZA

Eliza (1966)

- All men are alike.
- In what way?
- They’re always bugging us about something or other.
- Can you think of a specific example?
- Well, my boyfriend made me come here.
- Your boyfriend made you come here?
- He says I’m depressed much of the time.
- I am sorry to hear you are depressed
- ...
- Who else in your family takes care of you?
- My father.
- Your father?
- You are like my father in some ways.
- What resemblance do you see?



Eliza

- Speaking is dangerous for bots
- Draw the user out by reflecting statements back at them
- Rare domain where one can “assume the pose of knowing almost nothing of the world”
- Distracts from limited/no information
- Suggests a conversational goal
- Most chatbots trying to pass the Turing test operate in similar domains

1. Examine each word w in the sentence
2. Pick the w with the highest rank
3. If w exists:
    3.1 Check all rules for keyword w
    3.2 Apply the highest ranking rule
4. If no keyword applies:
    4.1 Apply a NONE rule
    4.2 Or grab an action from memory

Eliza

“I know everybody laughed at me”

- “I” is a very general keyword
- I: (I *) → (You say you *)
  You say you know everybody laughed at you
- Don’t forget to swap “me” for “you”
- “Everybody” is much more interesting
- Who says everybody/always is probably referring to a specific person/event. Let’s ask!
  Who in particular are you thinking of?
- Implementation: Keywords are stored with their ranks
  Everybody 5 ($R1$, $R2$, . . .)
  I 0 ($R1$, $R2$, . . .)
- NONE rules are generic conversation starters in case the previous sentence did not match any keywords
  Please go on.
  That’s very interesting.
  I see.
- Whenever “my” is the highest-ranking keyword
  - Transform sentence
  - Store it in memory (stack)
- Later, if we do not match keywords, we return to this topic

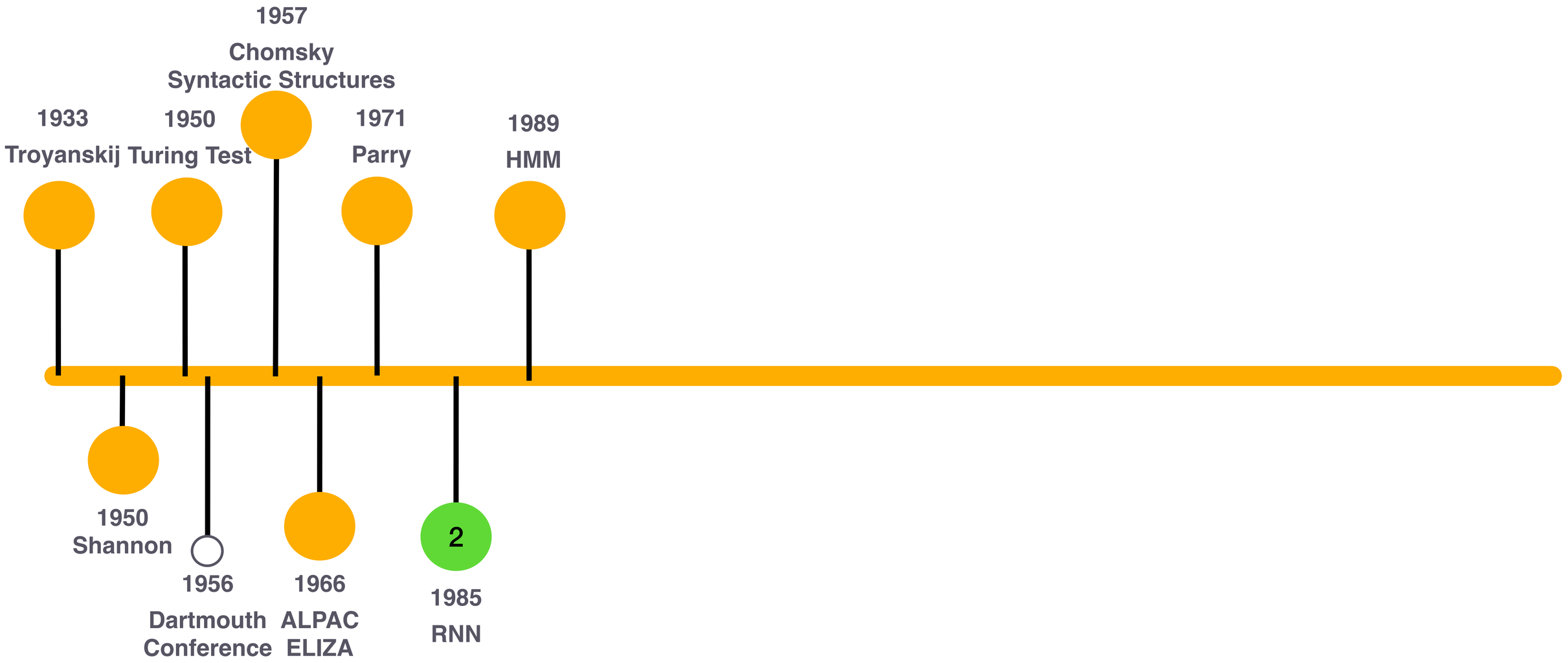
Parry (1971)

- The first system to pass the Turing test
- Same pattern-response structure as Eliza
- Much richer controls
- Mental model with affective variables
  - Anger, fear, mistrust
  - Reacts accordingly
- Psychiatrists could not distinguish Parry interviews from interviews with human paranoids
- Parry's persona:
  - 28-year-old single man, post office clerk
  - no siblings and lives alone
  - Sensitive about his physical appearance, his family, his religion, his education and the topic of sex
  - Hobbies are movies and gambling on horse races
  - Recently attacked a bookie, claiming the bookie did not pay off in a bet
  - Afterwards worried about possible underworld retaliation
  - Eager to tell his story to non-threatening listeners

Parry (1971)

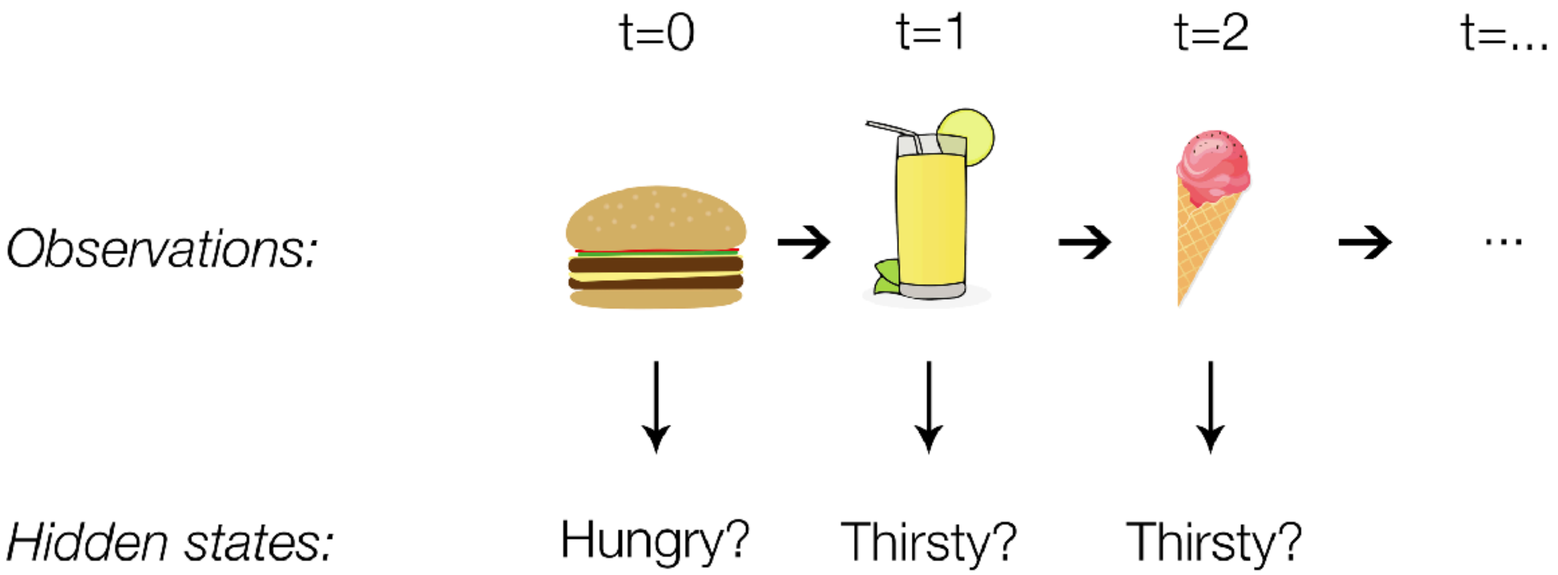
- How are you today?
- OK.
- What is your name?
- My name is Frank Smith.
- How old are you?
- 28.
- Are you married?
- No.
- What sort of work do you do?
- I work in the post office.
- Why are you in the hospital?
- I shouldn’t be here.
- Who brought you here?
- The police. (FEAR rises slightly)
- What trouble did you have with the police?
- They spy on me.
- Are you sure of all this?
- You don’t believe me? (FEAR, ANGER and MISTRUST go up)

NLP History


1957
Chomsky
Syntactic Structures
1933
Troyanskij
1950
Turing Test
1971
Parry
1989
HMM
1950
Shannon
1956
Dartmouth
Conference
1966
ALPAC
ELIZA
2
1985
RNN

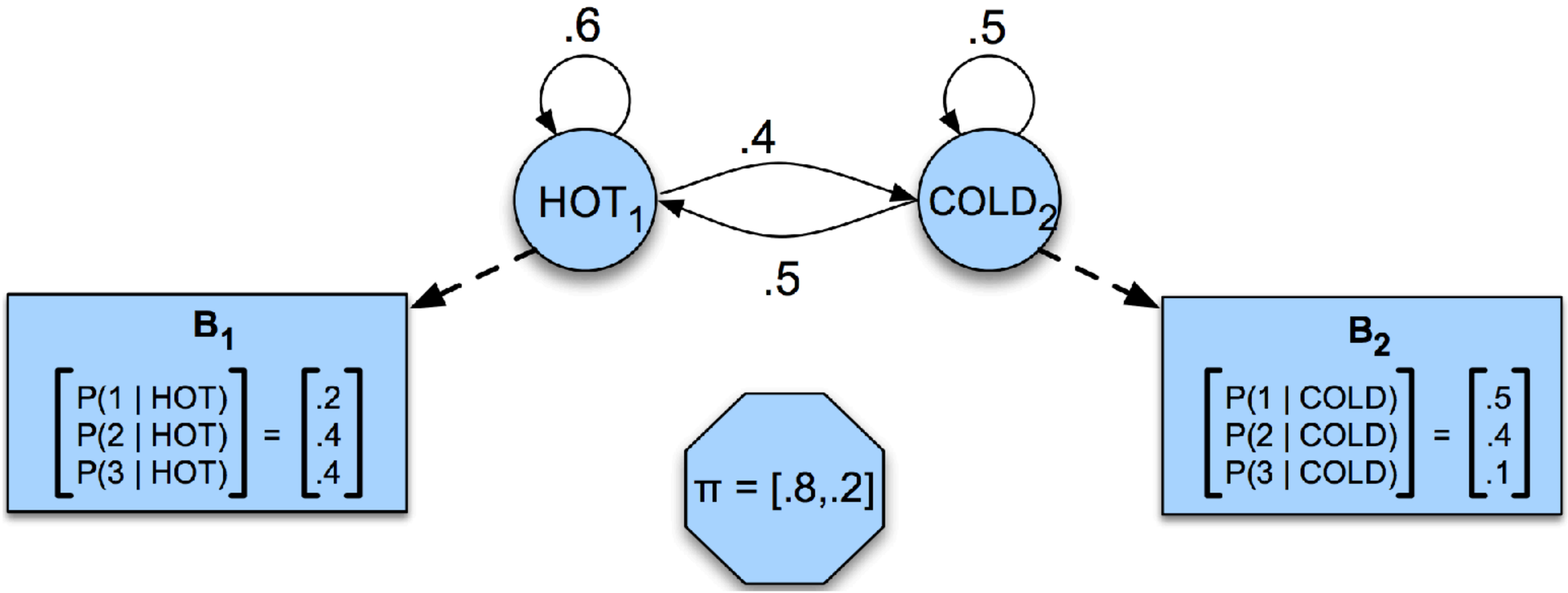
Hidden Markov Models (1989)

- An extension to Markov Chain Models
- Key idea: Some things are observable, causes are hidden
- Reason over hidden states based on observations
- Example: Jason Eisner likes ice cream (2002)

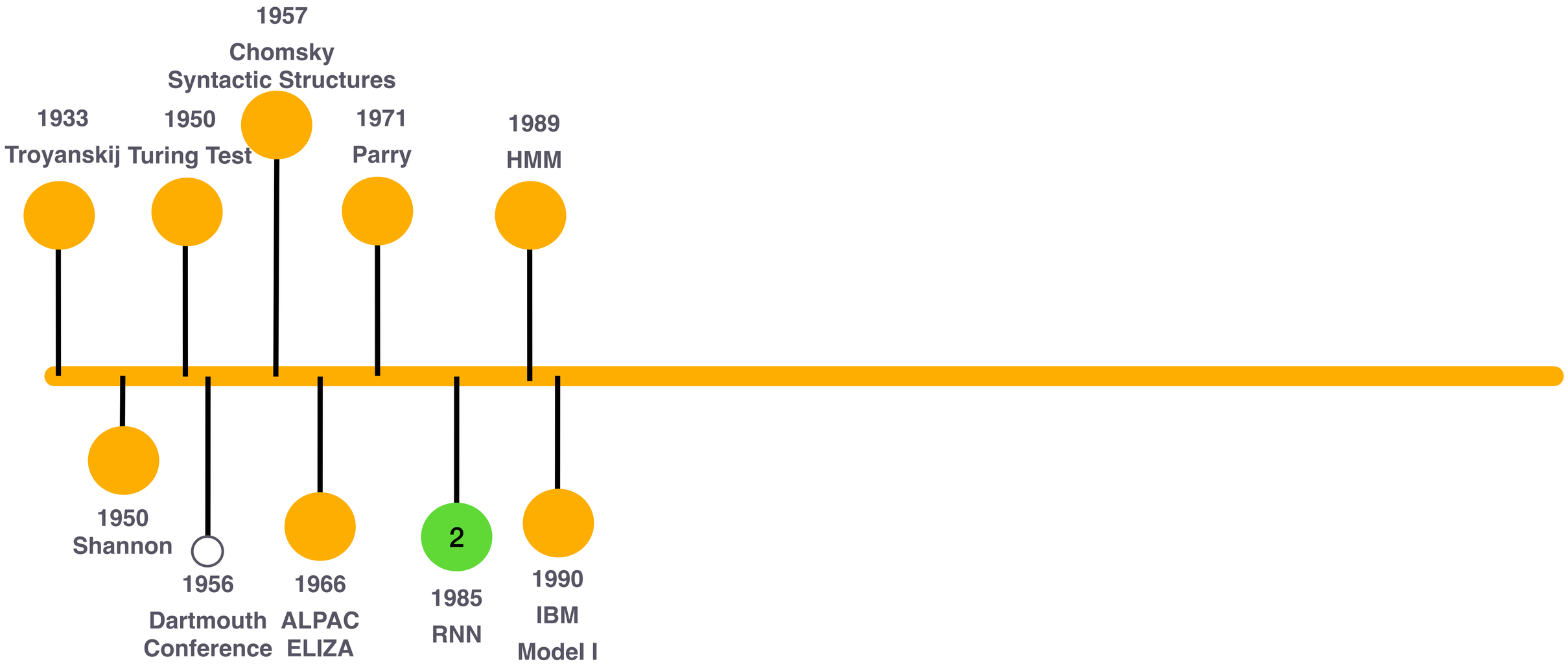


Hidden Markov Models (1989)

- Key components:
  - Hidden states $Q = \{q1\ ,\ q2\ ,\ .\ .\ .\ ,\ qN\}$
  - Transition probabilities $A = \{a1,1\ ,\ .\ .\ .\ ai,j\ ,\ .\ .\ .\ ,\ aN\ ,N\}$
  - Emission likelihoods $B = bi\ (ot\ )$
  - Initial state distribution $\Pi = \{\pi1\ ,\ \pi2\ ,\ .\ .\ .\ ,\ \pi N\}$
  - Observations $O = \{o1\ ,\ o2\ ,\ .\ .\ .\ ,\ oT\}$

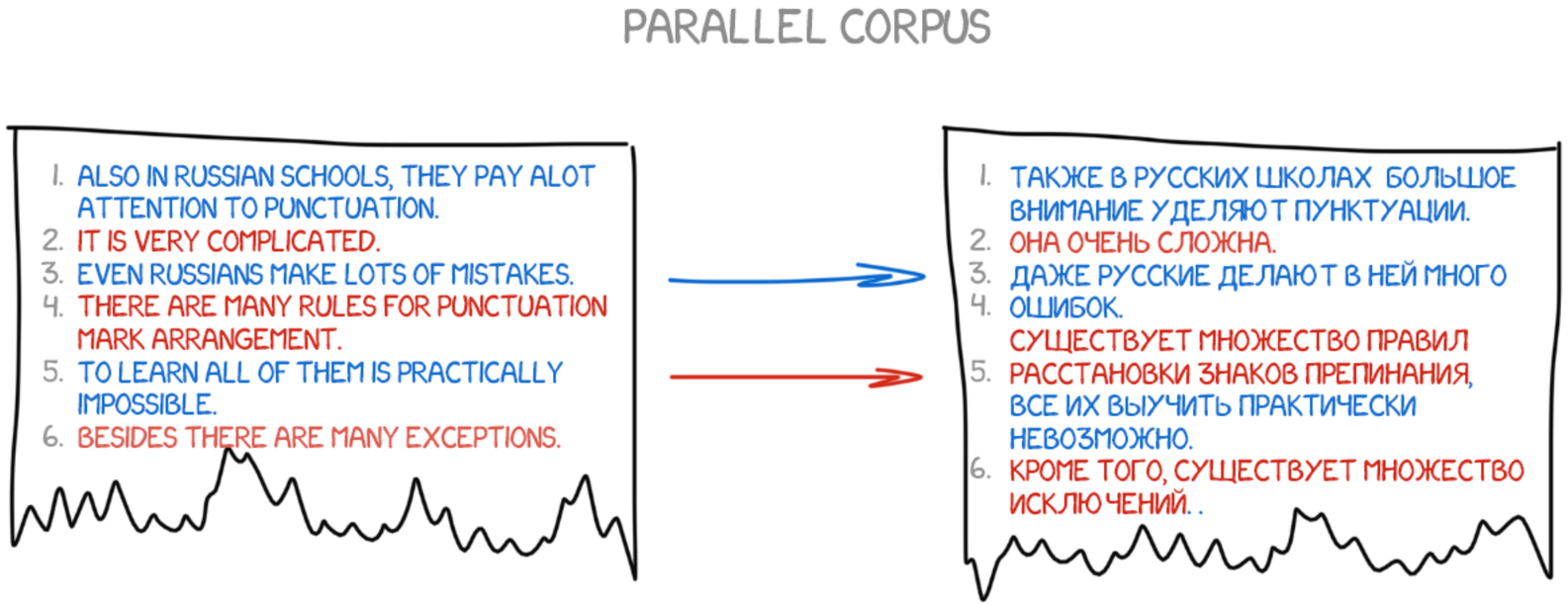


NLP History


1933
Troyanskij
1950
Turing Test
1957
Chomsky
Syntactic Structures
1971
Parry
1989
HMM
1950
Shannon
1956
Dartmouth
Conference
1966
ALPAC
ELIZA
2
1985
RNN
1990
IBM
Model I

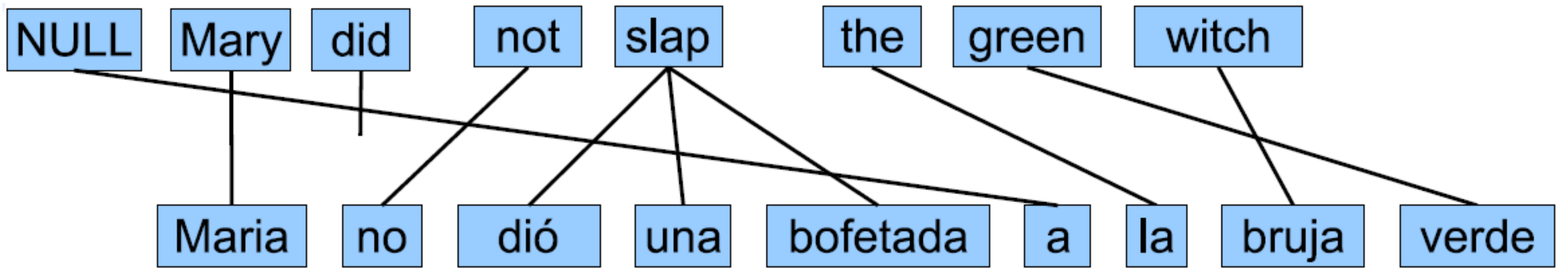
IBM Model I

- IBM, 1990
- Idea: Learn by computing translation likelihoods
- No more rules or linguists!
- Main source of information: parallel corpora
- The more data we use, the better we get
- Open Problem: Alignment
- If we had word-level alignment, the task would be trivial
- But that task is really cumbersome and contrived



IBM Model I

- Aligning words in source and target texts
- Words may be reordered
- Some source words translate into multiple targets
- Some source words are dropped
- Target words can be generated from the NULL word



IBM Model I

das

e	$t(e\|f)$
the	0.7
that	0.15
which	0.075
who	0.05
this	0.025

Haus

e	$t(e\|f)$
house	0.8
building	0.16
home	0.02
household	0.015
shell	0.005

ist

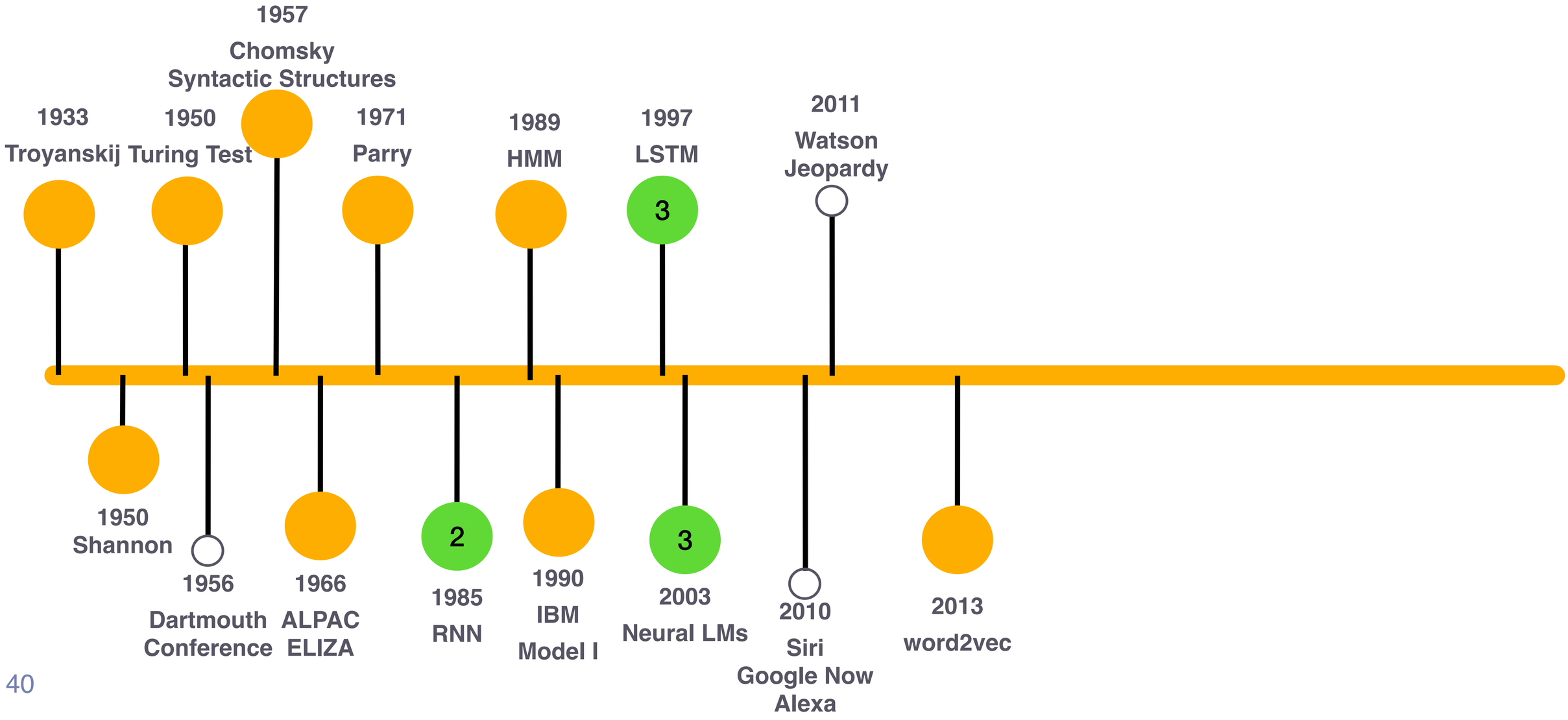
e	$t(e\|f)$
is	0.8
's	0.16
exists	0.02
has	0.015
are	0.005

klein

e	$t(e\|f)$
small	0.4
little	0.4
short	0.1
minor	0.06
petty	0.04

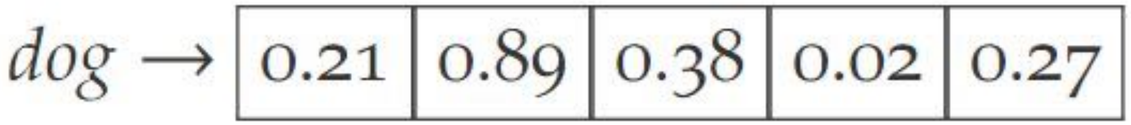
$$\hat{E} = \arg\max_{E} \frac{p(F|E)p(E)}{p(F)} \approx \overbrace{p(F|E)}^{\textit{translation model}} \overbrace{p(E)}^{\textit{language model}}$$

NLP History


1933
Troyanskij
1950
Turing Test
1957
Chomsky
Syntactic Structures
1971
Parry
1989
HMM
1997
LSTM
3
2011
Watson
Jeopardy
1950
Shannon
1956
Dartmouth
Conference
1966
ALPAC
ELIZA
2
1985
RNN
1990
IBM
Model I
3
2003
Neural LMs
2010
Siri
Google Now
Alexa
2013
word2vec

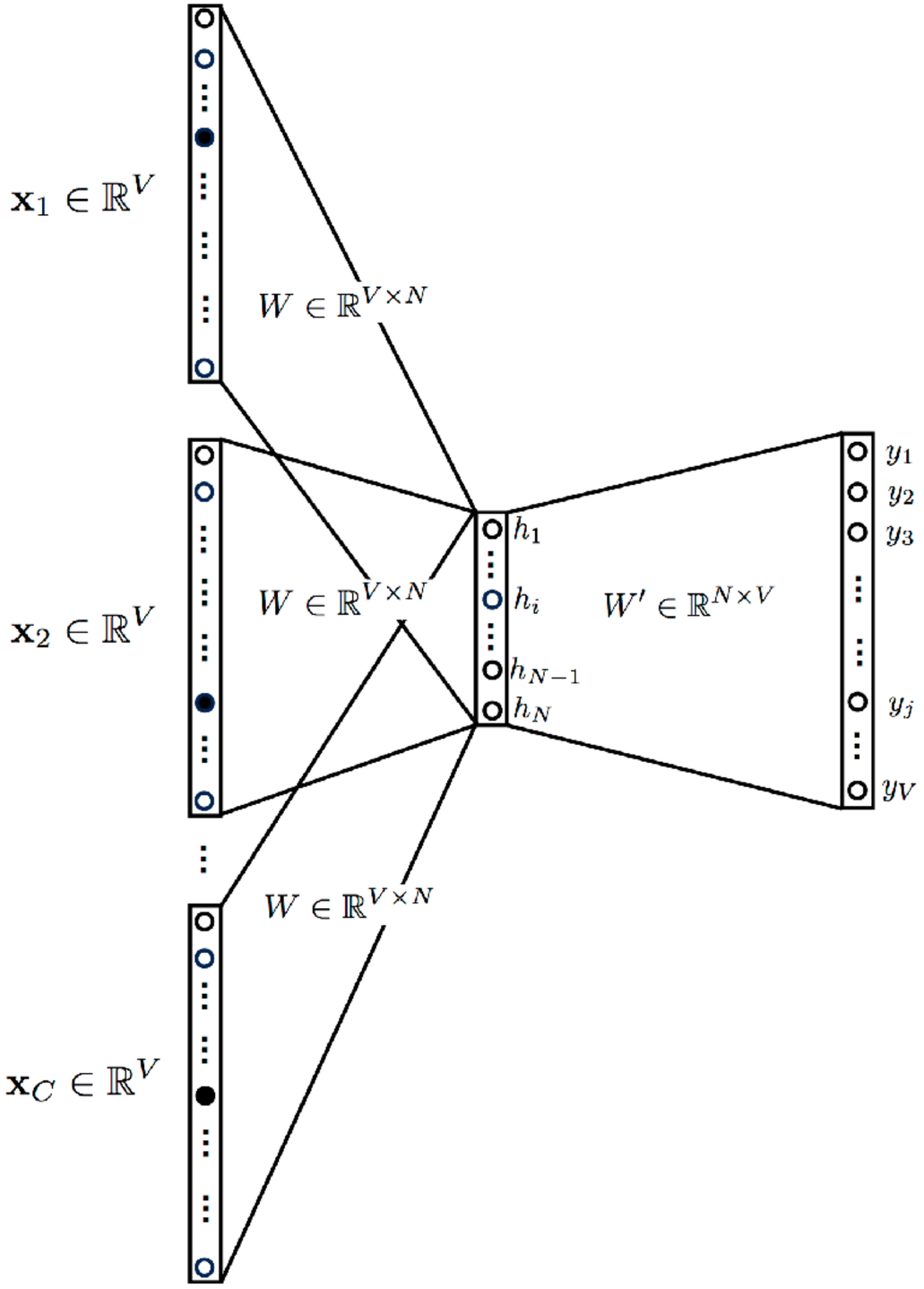
word2vec (2013)

- How to bring meaning to words?
- Distributional semantics
  - Remember Wittgenstein
- Describe words based on the words surrounding them
- Idea: Represent terms as N-dimensional vectors
- Similar to autoencoding
- Popular word embedding method
- Input: (large) sequences of text
- Encoding: Sparse V-dimensional Boolean BOW (1-hot encoding)
- Output: Dense N-dimensional embeddings
- Two training metaphors:
  - Continuous Bag of Words (CBOW)
  - Skip-gram

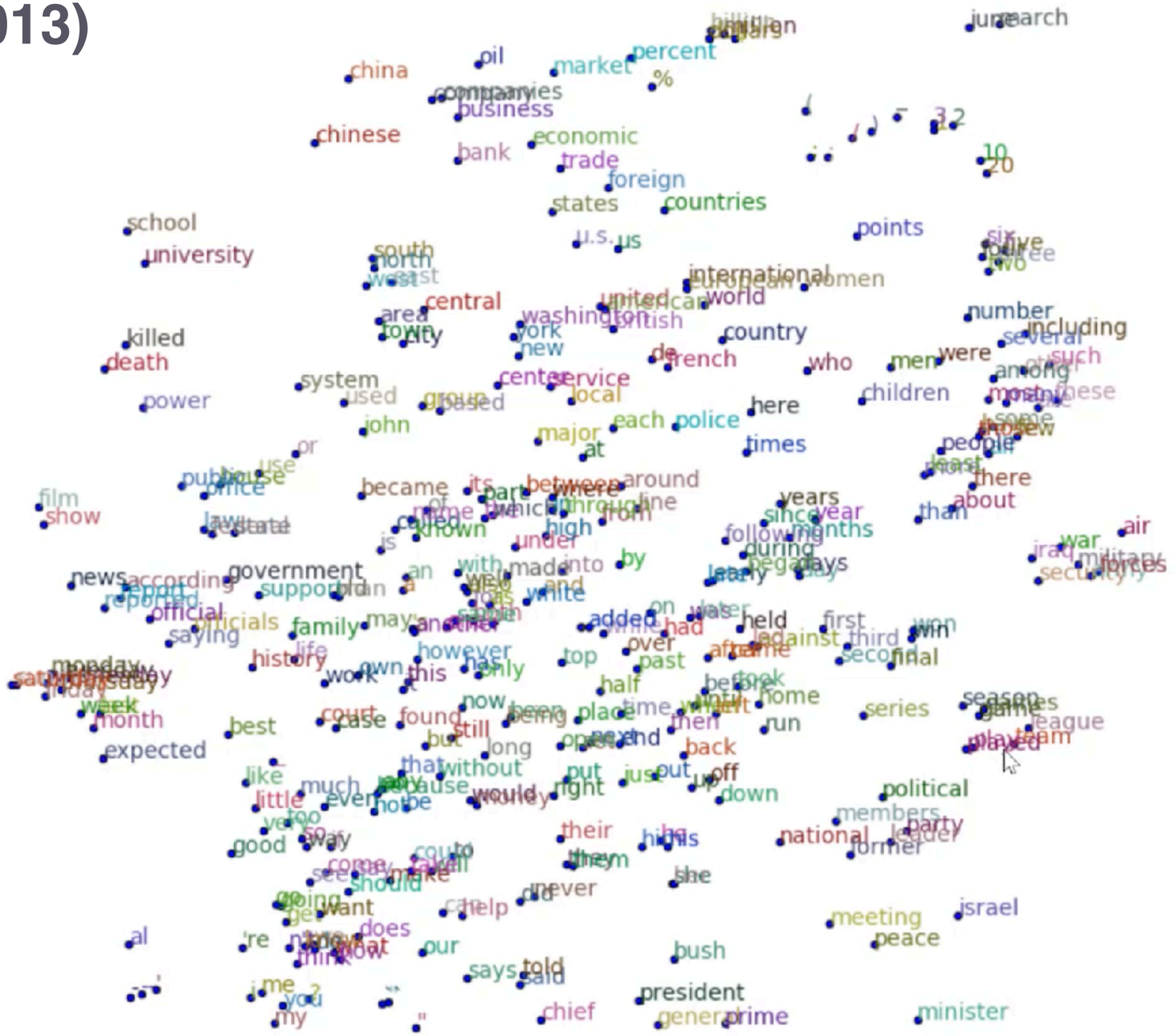


word2vec (2013)

- CBOW: Given the context, predict the missing word
- Now we have dense vector representations
- Important parameters:
  - Context size c
  - Embedding dimension N
- Word vectors describe locations in N-dimensional space
- Similar terms are located nearby each other

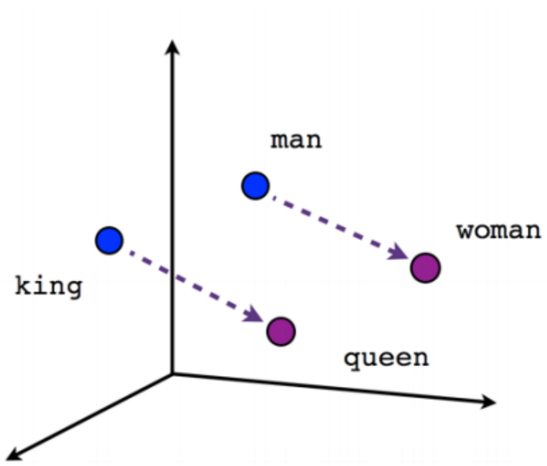


word2vec (2013)

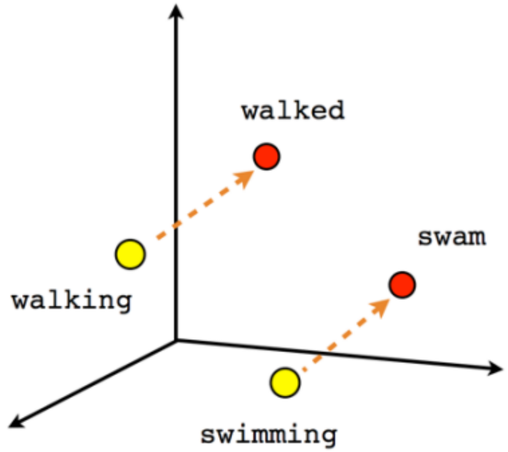


word2vec (2013)

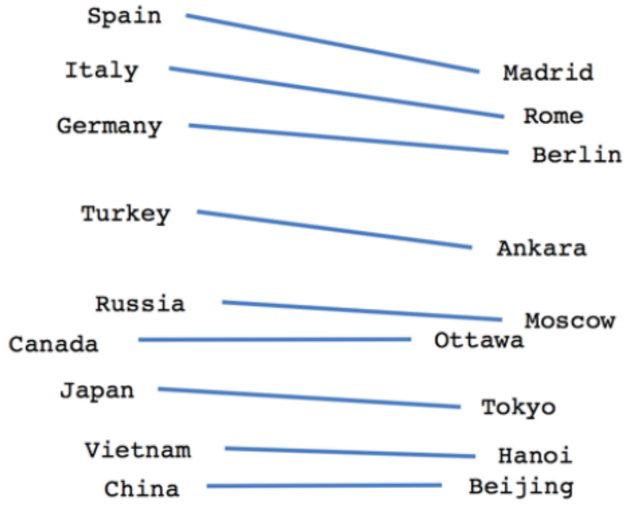
- But we can also perform simple semantic reasoning



Male-Female

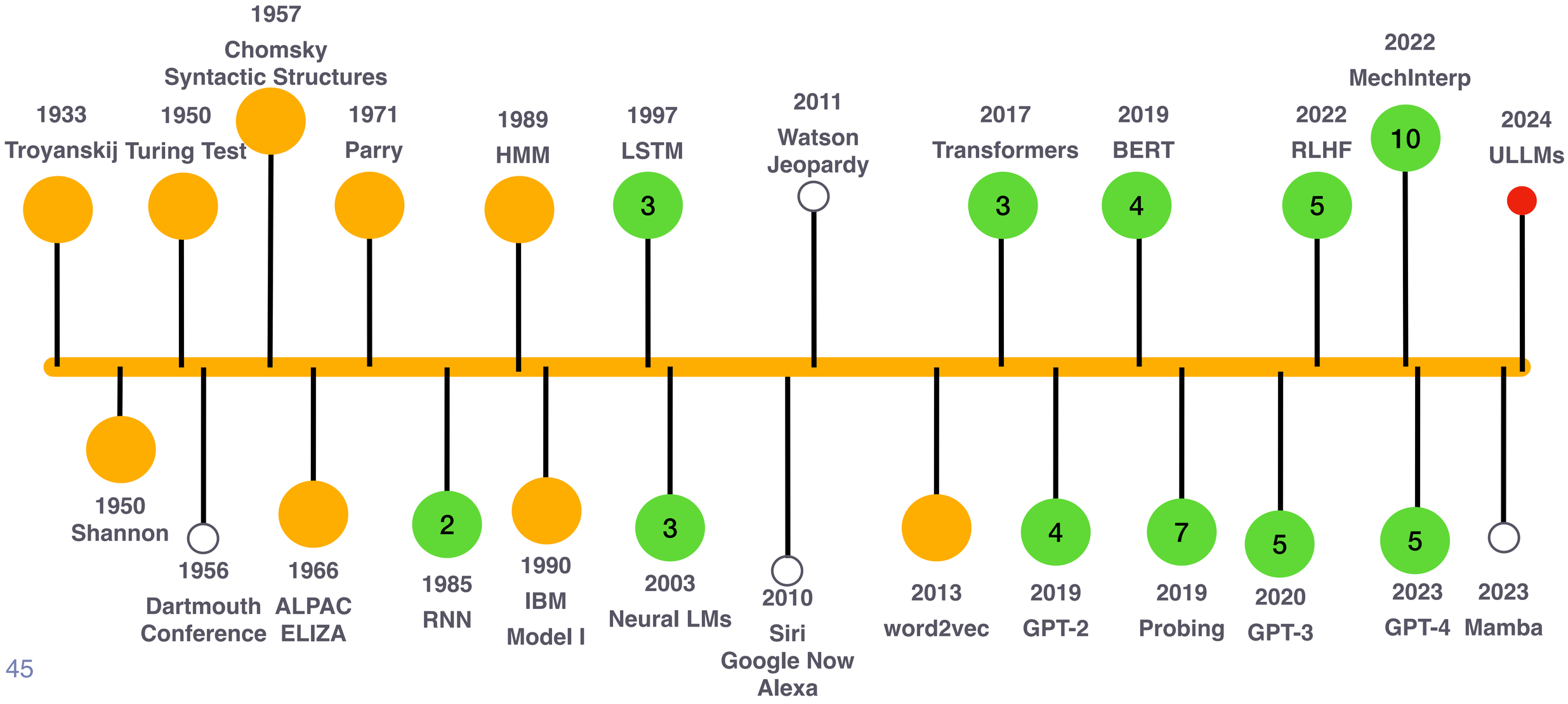


Verb tense

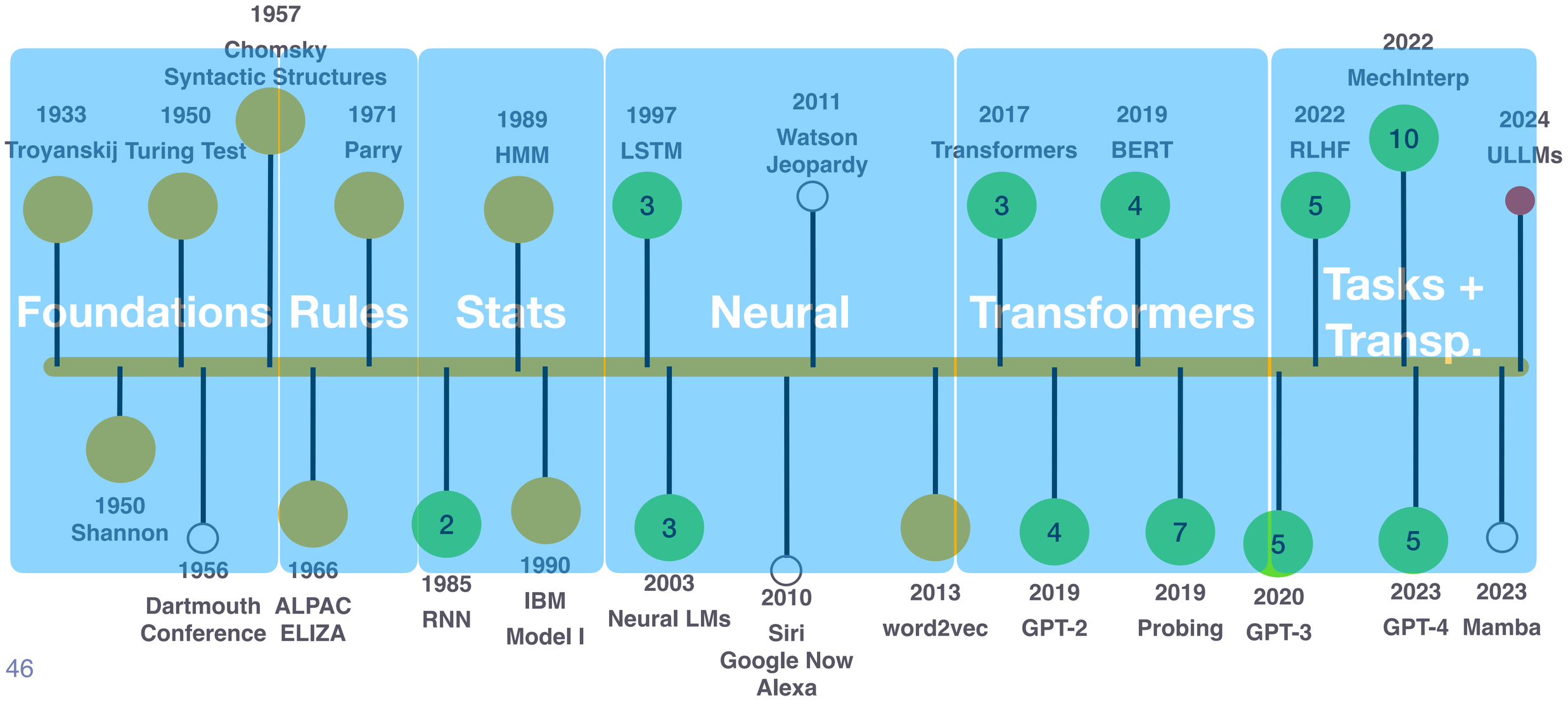


Country-Capital

NLP History


1933
Troyanskij
1950
Turing Test
1957
Chomsky
Syntactic Structures
1971
Parry
1989
HMM
1997
LSTM
3
2011
Watson
Jeopardy
2017
Transformers
3
2019
BERT
4
2022
RLHF
5
2022
MechInterp
10
2024
ULLMs
1950
Shannon
1956
Dartmouth
Conference
1966
ALPAC
ELIZA
2
1985
RNN
1990
IBM
Model I
3
2003
Neural LMs
2010
Siri
Google Now
Alexa
2013
word2vec
4
2019
GPT-2
7
2019
Probing
5
2020
GPT-3
5
2023
GPT-4
2023
Mamba

NLP History


1957
Chomsky
Syntactic Structures
1933
Troyanskij
1950
Turing Test
1971
Parry
1989
HMM
1997
LSTM
2011
Watson
Jeopardy
2017
Transformers
2019
BERT
2022
RLHF
2022
MechInterp
2024
ULLMs
Foundations
Rules
Stats
Neural
Transformers
Tasks +
Transp.
1950
Shannon
1956
Dartmouth
Conference
1966
ALPAC
ELIZA
1985
RNN
1990
IBM
Model I
2003
Neural LMs
2010
Siri
Google Now
Alexa
2013
word2vec
2019
GPT-2
2019
Probing
2020
GPT-3
2023
GPT-4
2023
Mamba

Homework for next week

- get familiar with PyTorch
  - choose a tutorial of your trust & liking
- read / prepare the following tutorials from the course web-book:
  - Sheet 2.1: PyTorch essentials
  - Sheet 2.2: ML-estimation
- recap math notation and linear algebra basics w/ “cheat sheet” on moodle