

Извличане на знания от текст

Преслав Наков

12 март 2025 г.

Acknowledgments

Some slides adapted from Eugene Charniak, Bonnie Dorr, Joshua Goodman, Marti Hearst, Dan Klein, Debapriyo Majumdar, Chris Manning, Drago Radev, Timo Schick, Stephan Vogel, etc

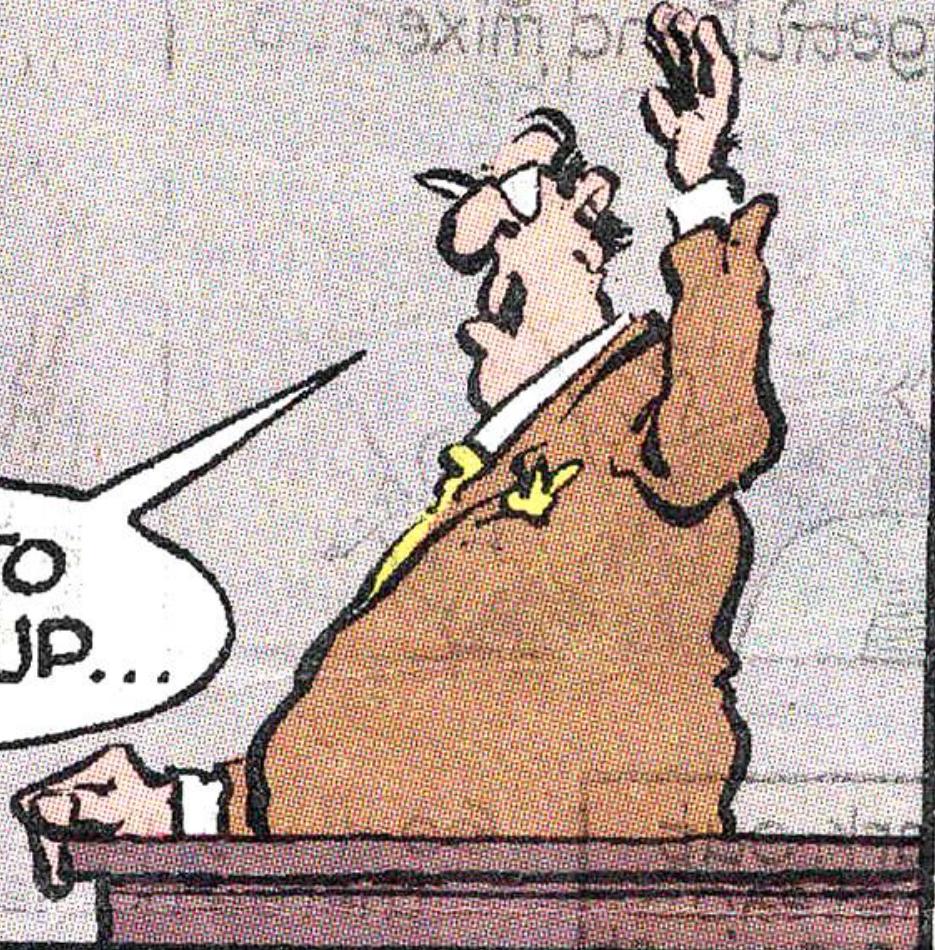
Language Models: Introduction

HERMAN

REPEAT AFTER ME ...
I SWEAR TO TELL
THE TRUTH...



I SWERVE TO
SMELL DE SOUP...



by Jim Unger



.. AND NOTHING
BUT THE TRUTH.

© Jim Unger/Dist by United Media, Jan. 30/00

.. AN NUTS SING
ON DE ROOF.

Heimann is reprinted with permission from laughingStock Licensing Inc., Ottawa, Canada. All rights reserved.

NOW TELL US IN YOUR OWN
WORDS EXACTLY WHAT
HAPPENED.



Language Model

- A **language model (LM)** is a probability distribution over word sequences
- E.g.
 - $P(\text{"And nothing but the truth"}) \approx 0.001$
 - $P(\text{"And nuts sing on the roof"}) \approx 0$

Language Models: Applications

- Probability estimation

- next word/phrase prediction
 - the noisy channel model: ASR, SMT, OCR, ...

- Text classification

- e.g., authorship attribution, language identification, topic categorization, fact-checking, ...

- Information retrieval

- Text generation

- Recently, in many, many tasks (revolution!)

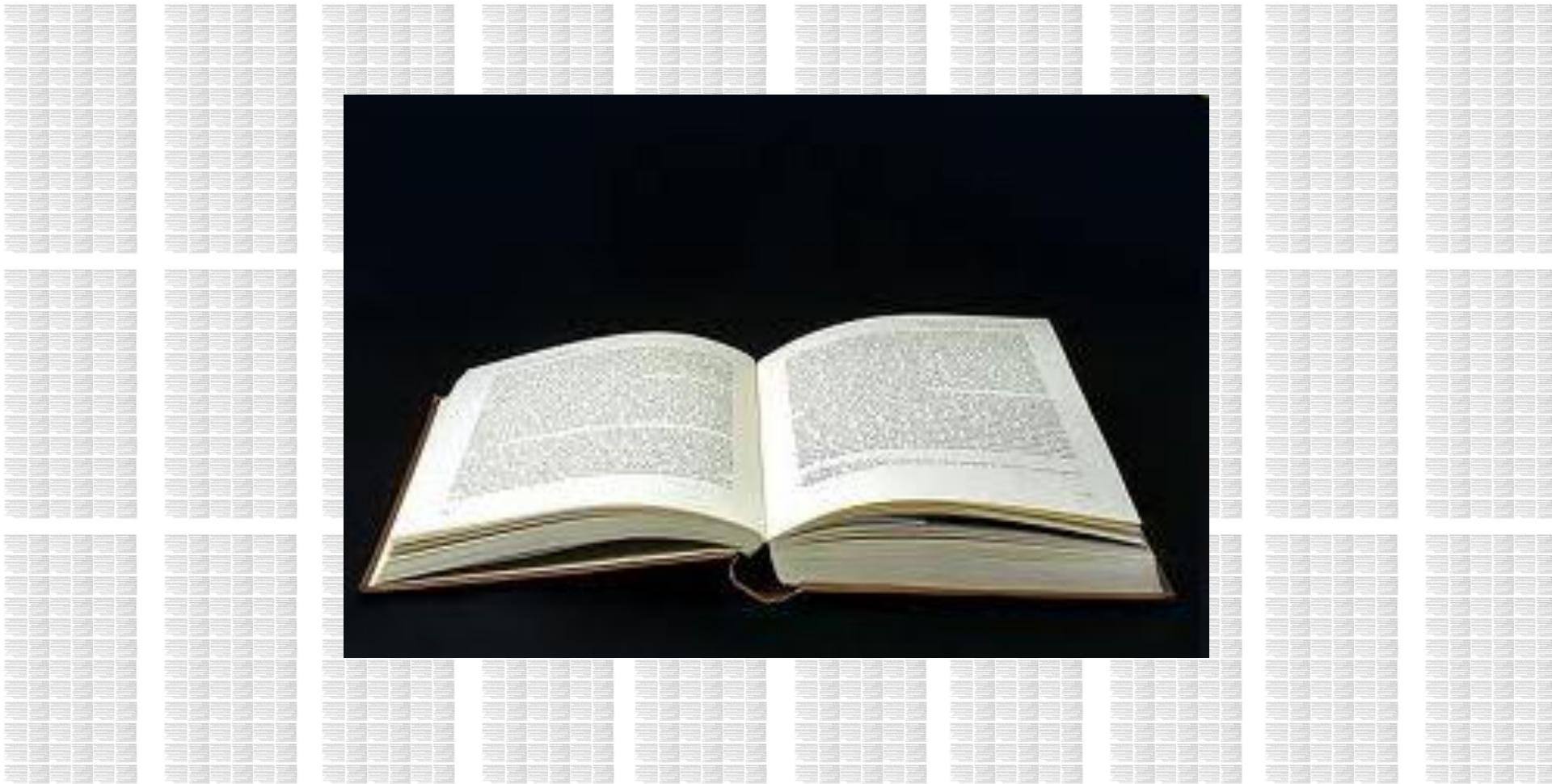
- Large-scale pre-trained transformers
 - Few/zero-shot learning, e.g., using prompts

Training a Language Model

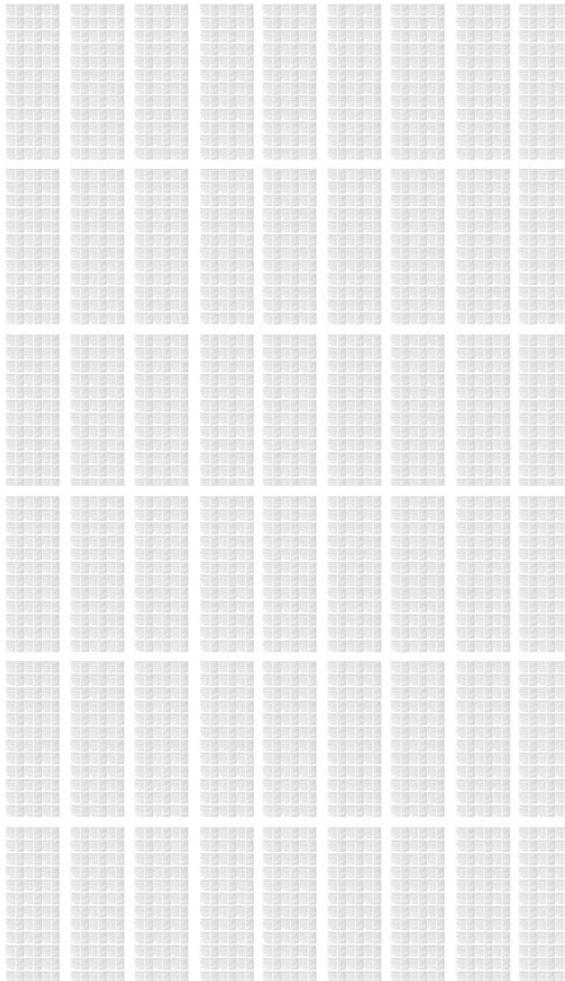
1,000 sentences

A large grid of 1,000 sentences, arranged in approximately 40 rows and 25 columns. Each sentence is a single line of text in a dark, monospaced font. The grid is centered on a white background.

10,000 sentences = 1 book



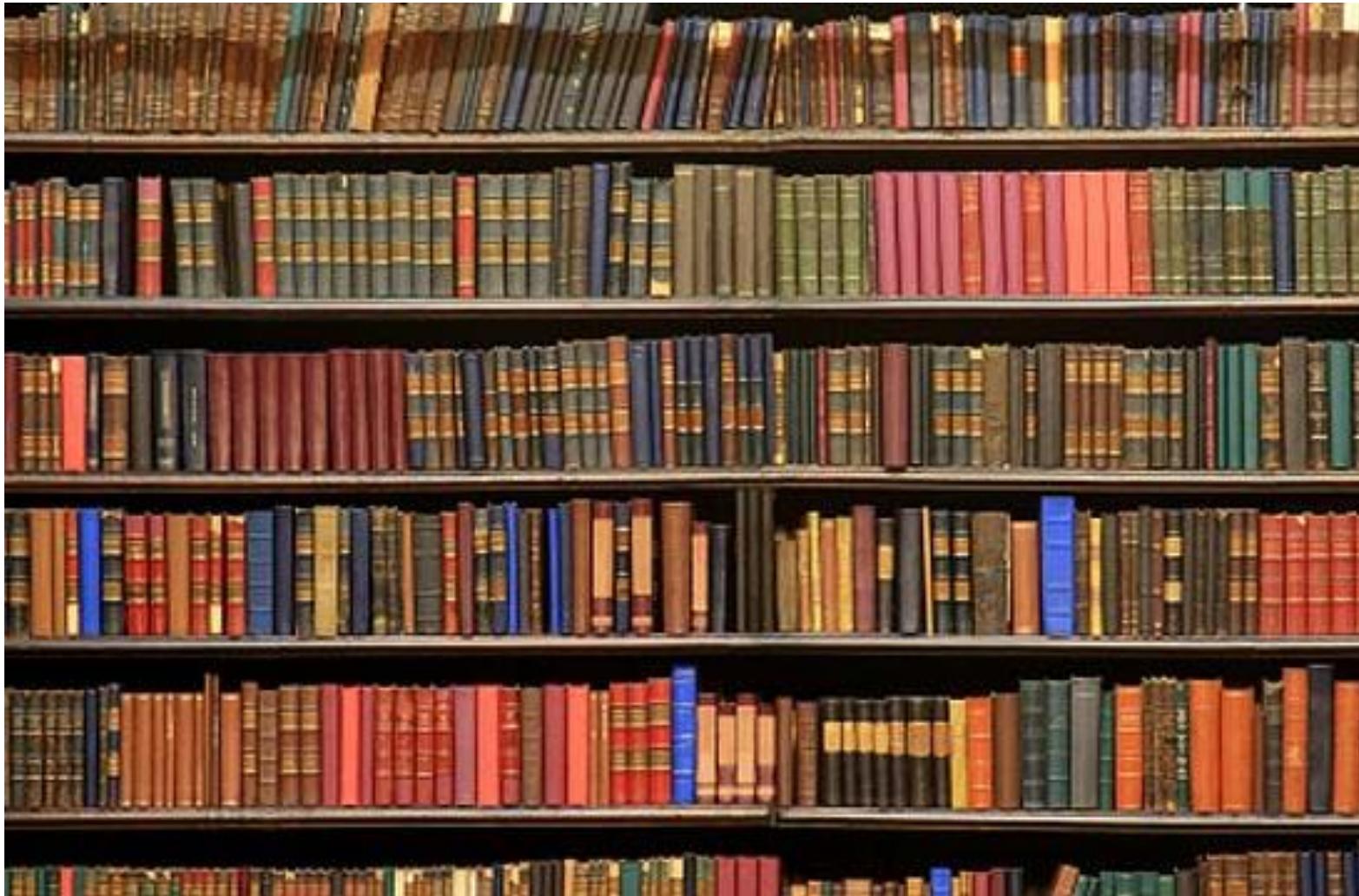
100,000 sentences = several books



1,000,000 sentences = a bookshelf



10,000,000 sentences = a large bookshelf



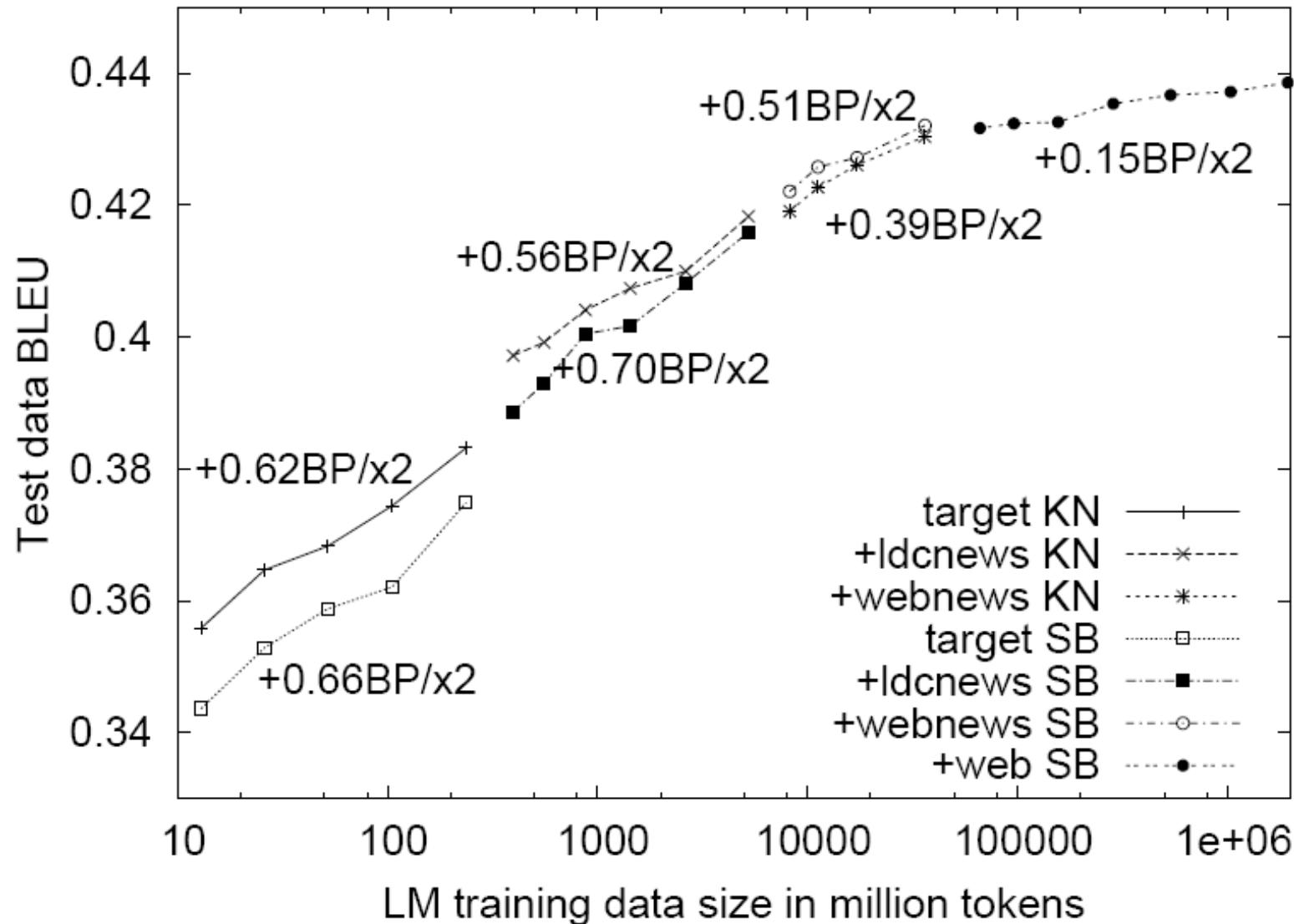
More and more data...



More and more data...



Web-Scale Language Models for SMT at Google



(Brants&al., EMNLP 2007)

n-gram Language Models

Probabilistic Language Models

- Input: many observations of sentences
- Output: model that assigns probabilities $P(x)$ to any sentence
- Probabilities should broadly indicate the likelihood of sentences
 - $P(\text{I saw a van}) >> P(\text{eyes awe of an})$
 - Not grammaticality: $P(\text{artichokes intimidate zippers}) \approx 0$
- One option: empirical distribution over sentences?
 - Problem: doesn't generalize (at all)
- Two ways of generalizing
 - **Decomposition:** sentences generated incrementally using small steps that can be recombined in new ways
 - **Smoothing:** allow for the possibility of unseen events

How Language Models Work

- Hard to compute $P(\text{"And nothing but the truth"})$
- Step 1: Decompose the probability

$P(\text{"And nothing but the truth"}) =$
 $P(\text{"And"}) \times$
 $P(\text{"nothing|and"}) \times$
 $P(\text{"but|and nothing"}) \times$
 $P(\text{"the|and nothing but"}) \times$
 $P(\text{"truth|and nothing but the"})$

N-Gram Models of Language

- Use the previous $n-1$ words in a sequence to predict the next word
- Language Model (LM)
 - unigrams, bigrams, trigrams,...
- How do we **train** these models?
 - Very large corpora

Simple N-Grams

- Assume a language has V word types in its lexicon, how likely is word x to follow word y ?
 - The simplest model of word probability: $1/V$
 - Alternative 1: estimate likelihood of x occurring in new text based on its general frequency of occurrence estimated from a corpus (**unigram** probability)
popcorn is more likely to occur than **unicorn**
 - Alternative 2: condition the likelihood of x occurring in the context of previous words (bigrams, trigrams,...)
mythical unicorn is more likely than **mythical popcorn**

A Word on Notation

- $P(\text{unicorn})$
 - Read this as “The probability of seeing the token *unicorn*”
 - Unigram taggers use this.

- $P(\text{unicorn}|\text{mythical})$
 - Called the **Conditional Probability**.
 - Read this as “The probability of seeing the token *unicorn* given that you’ve seen the token *mythical*”
 - Bigram taggers use this.

Training a Language Model

the students opened their _____

- **Question:** How to learn a Language Model?
- **Answer** (pre- Deep Learning): learn a *n*-gram Language Model!
- **Definition:** A *n*-gram is a chunk of *n* consecutive words.
 - **uni**grams: “the”, “students”, “opened”, “their”
 - **bi**grams: “the students”, “students opened”, “opened their”
 - **tri**grams: “the students opened”, “students opened their”
 - **4**-grams: “the students opened their”
- **Idea:** Collect statistics about how frequent different n-grams are, and use these to predict next word.

Computing the Probability of a Word Sequence

- Compute the product of component conditional probabilities?
 - $P(\text{the mythical unicorn}) = P(\text{the}) P(\text{mythical}|\text{the}) P(\text{unicorn}|\text{the mythical})$
- The longer the sequence, the less likely we are to find it in a training corpus
 - $P(\text{Most biologists and folklore specialists believe that in fact the mythical unicorn horns derived from the narwhal})$
- Solution: approximate using n -grams

N-Gram Language Models

- ❑ No loss of generality to break the probability down

$$P(w_1 \dots w_n) = \prod_i P(w_i | w_1 \dots w_{i-1})$$

- ❑ Too many histories!

- $P(\text{down} | \text{No loss of generality to break the probability})$?
- $P(\text{???} | \text{the water is so transparent that})$?

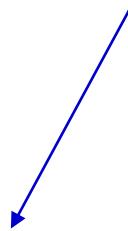
- ❑ N-gram models: assume each word depends only on a short linear history

$$P(w_1 \dots w_n) = \prod_i P(w_i | w_{i-k} \dots w_{i-1})$$

Bigram Language Model

$$\begin{aligned} P(e) &= P(w_1 w_2 \dots w_n) \\ &= P(w_1) P(w_2 | w_1) P(w_3 | w_1 w_2) P(w_4 | w_1 w_2 w_3) P(w_5 | w_1 w_2 w_3 w_4) \dots \\ &\approx P(w_1) P(w_2 | w_1) P(w_3 | w_2) P(w_4 | w_3) P(w_5 | w_4) \dots \end{aligned}$$

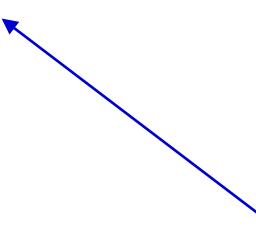
Chain rule



Bigram Language Model

$$\begin{aligned} P(e) &= P(w_1 w_2 \dots w_n) \\ &= P(w_1) P(w_2 | w_1) P(w_3 | w_1 w_2) P(w_4 | w_1 w_2 w_3) P(w_5 | w_1 w_2 w_3 w_4) \dots \\ &\approx P(w_1) P(w_2 | w_1) P(w_3 | w_2) P(w_4 | w_3) P(w_5 | w_4) \dots \end{aligned}$$

Chain rule



Markov model
(approximation)



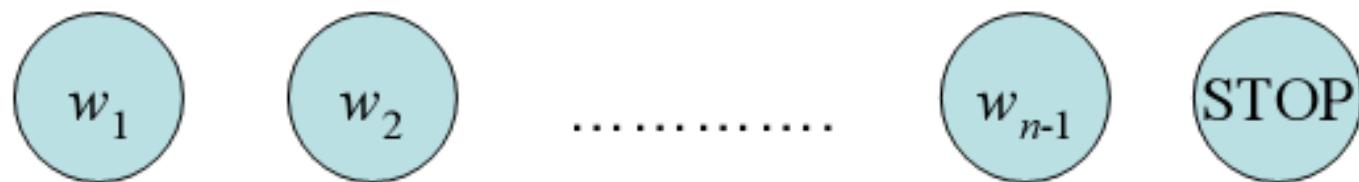
Andrei Markov

Unigram Models

- ❑ Simplest case: unigrams

$$P(w_1 \dots w_n) = \prod_i P(w_i)$$

- ❑ Generative process: pick a word, pick a word, ...

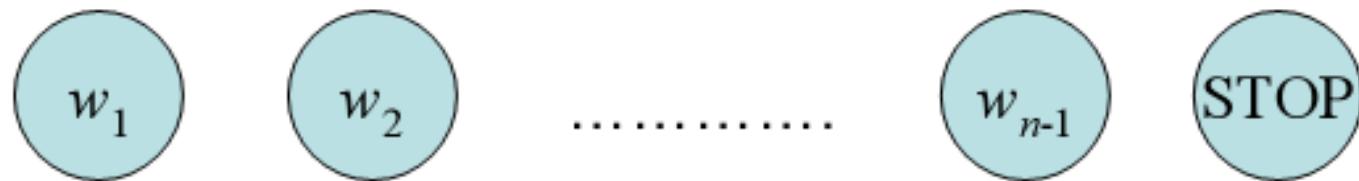


Unigram Models

- Simplest case: unigrams

$$P(w_1 \dots w_n) = \prod_i P(w_i)$$

- Generative process: pick a word, pick a word, ...



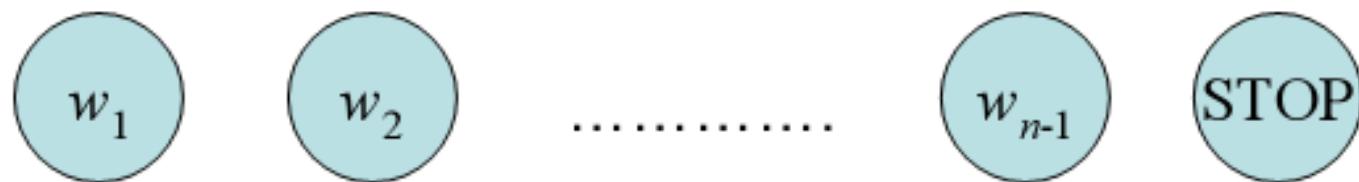
**Why do we need the
special STOP symbol?**

Unigram Models

- ❑ Simplest case: unigrams

$$P(w_1 \dots w_n) = \prod_i P(w_i)$$

- ❑ Generative process: pick a word, pick a word, ...



- ❑ To make this a proper distribution over sentences, we have to generate a special STOP symbol last.

Unigram Models: Generation

□ Examples:

- [fifth, an, of, futures, the, an, incorporated, a, a, the, inflation, most, dollars, quarter, in, is, mass.]
- [thrift, did, eighty, said, hard, 'm, july, bullish]
- [that, or, limited, the]
- []
- [after, any, on, consistently, hospital, lake, of, of, other, and, factors, raised, analyst, too, allowed, mexico, never, consider, fall, bungled, davison, that, obtain, price, lines, the, to, sass, the, the, further, board, a, details, machinists, the, companies, which, rivals, an, because, longer, oakes, percent, a, they, three, edward, it, currier, an, within, in, three, wrote, is, you, s., longer, institute, dentistry, pay, however, said, possible, to, rooms, hiding, eggs, approximate, financial, canada, the, so, workers, advancers, half, between, nasdaq]

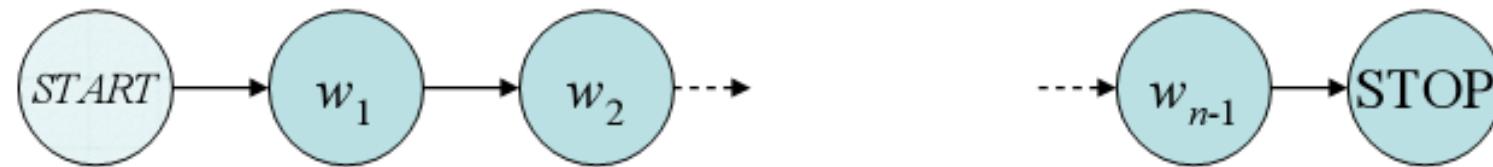
Bigram Models

- Big problem with unigrams:

$P(\text{the the the the}) \gg P(\text{I like ice cream})!$

- Solution: condition on the previous word

$$P(w_1 \dots w_n) = \prod_i P(w_i | w_{i-1})$$



$$P(\text{"I eat an apple ..."}) = P(\text{I} | \text{<S>}) \cdot P(\text{eat} | \text{I}) \cdot P(\text{an} | \text{eat}) \cdot P(\text{apple} | \text{an}) \dots$$

Bigram Models: Generation

□ Examples:

- [texaco, rose, one, in, this, issue, is, pursuing, growth, in, a, boiler, house, said, mr., gurria, mexico, 's, motion, control, proposal, without, permission, from, five, hundred, fifty, five, yen]
- [outside, new, car, parking, lot, of, the, agreement, reached]
- [although, common, shares, rose, forty, six, point, four, hundred, dollars, from, thirty, seconds, at, the, greatest, play, disingenuous, to, be, reset, annually, the, buy, out, of, american, brands, vying, for, mr., womack, currently, sharedata, incorporated, believe, chemical, prices, undoubtedly, will, be, as, much, is, scheduled, to, conscientious, teaching]
- [this, would, be, a, record, november]

The Trigram Approximation

Assume each word depends only on the previous two words
(three words total – tri means three, gram means writing)

$$P(\text{"the|... whole truth and nothing but"}) \approx P(\text{"the|nothing but"})$$

$$P(\text{"truth|... whole truth and nothing but the"}) \approx P(\text{"truth|but the"})$$

Regular Languages?

- N-gram models are (weighted) regular languages
 - Many linguistic arguments that language is not regular.
 - Long-distance effects:
 - “[The computer](#) which I had just put into the machine room on the fifth floor crashed.”
 - Recursive structure
 - Why can we often get away with n-gram models?
- PCFG LM:
 - [This, quarter, 's, surprisingly, independent, attack, paid, off, the, risk, involving, IRS, leaders, and, transportation, prices, .]
 - [It, could, be, announced, sometime, .]
 - [Mr., Toseland, believes, the, average, defense, economy, is, drafted, from, slightly, more, than, 12, stocks, .]

Training a Language Model

$$P(\mathbf{x}^{(t+1)} | \mathbf{x}^{(t)}, \dots, \mathbf{x}^{(1)}) = P(\mathbf{x}^{(t+1)} | \mathbf{x}^{(t)}, \dots, \mathbf{x}^{(t-n+2)})$$

(assumption)

n-1 words

prob of a n-gram $\rightarrow P(\mathbf{x}^{(t+1)}, \mathbf{x}^{(t)}, \dots, \mathbf{x}^{(t-n+2)})$ (definition of conditional prob)

prob of a (n-1)-gram $\rightarrow P(\mathbf{x}^{(t)}, \dots, \mathbf{x}^{(t-n+2)})$

- **Question:** How do we get these n -gram and $(n-1)$ -gram probabilities?
- **Answer:** By **counting** them in some large corpus of text!

$$\approx \frac{\text{count}(\mathbf{x}^{(t+1)}, \mathbf{x}^{(t)}, \dots, \mathbf{x}^{(t-n+2)})}{\text{count}(\mathbf{x}^{(t)}, \dots, \mathbf{x}^{(t-n+2)})}$$

(statistical approximation)

Example: 4-gram LM

Suppose we are learning a 4-gram Language Model.

~~as the proctor started the clock, the students opened their~~ _____
discard condition on this

$$P(\mathbf{w}|\text{students opened their}) = \frac{\text{count(students opened their } \mathbf{w})}{\text{count(students opened their)}}$$

For example, suppose that in the corpus:

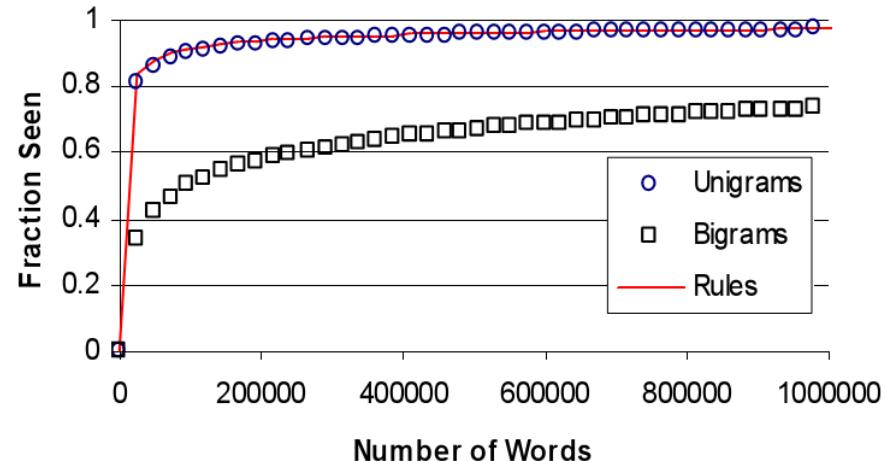
- “students opened their” occurred 1000 times
 - “students opened their books” occurred 400 times
 - $\rightarrow P(\text{books} | \text{students opened their}) = 0.4$
 - “students opened their exams” occurred 100 times
 - $\rightarrow P(\text{exams} | \text{students opened their}) = 0.1$
- } Should we have discarded the “proctor” context?

Language Models: Sparsity

Sparsity

- Problems with n-gram models:

- New words appear all the time:
 - Synaptitude
 - 132,701.03
 - multidisciplinarization



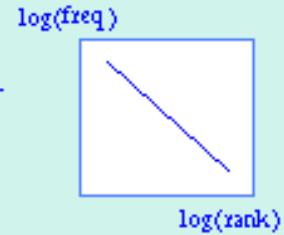
- New bigrams: even more often
- Trigrams or more – still worse!
- Number of Words

Sparsity: Zipf's Law

- Zipf's Law
 - Types (words) vs. tokens (word occurrences)
 - Broadly: most word types are rare ones
 - Specifically:
 - Rank word types by token frequency
 - Frequency inversely proportional to rank
- Not special to language: randomly generated character strings have this property (try it!)

Zipf's Law

ZIPF'S LAW



Rank x Frequency \approx Constant

Rank	Term	Freq.	Z	Rank	Term	Freq.	Z
1	the	69,971	0.070	6	in	21,341	0.128
2	of	36,411	0.073	7	that	10,595	0.074
3	and	28,852	0.086	8	is	10,099	0.081
4	to	26,149	0.104	9	was	9,816	0.088
5	a	23,237	0.116	10	he	9,543	0.095

Zipf's Law is Fairly General

- Frequency of accesses to web pages
 - in particular the access counts on the Wikipedia page, with s approximately equal to 0.3
 - page access counts on Polish Wikipedia (data for late July 2003) approximately obey Zipf's law with s about 0.5
- Words in the English language
 - for instance, in Shakespeare's play Hamlet with s approximately 0.5
- Sizes of settlements
- Income distributions amongst individuals
- Size of earthquakes
- Notes in musical performances

http://en.wikipedia.org/wiki/Zipf's_law

<http://www.nslij-genetics.org/wli/zipf/>

http://www.cut-the-knot.org/do_you_know/zipfLaw.shtml

Sparsity Problems with n -gram LMs

Sparsity Problem 1

Problem: What if “*students opened their w*” never occurred in data? Then w has probability 0!

(Partial) Solution: Add small δ to the count for every $w \in V$. This is called *smoothing*.

$$P(w|\text{students opened their}) = \frac{\text{count(students opened their } w\text{)}}{\text{count(students opened their)}}$$

Sparsity Problem 2

Problem: What if “*students opened their*” never occurred in data? Then we can’t calculate probability for *any w*!

(Partial) Solution: Just condition on “*opened their*” instead. This is called *backoff*.

Sparsity Problems with n -gram LMs

Note: Increasing n makes sparsity problems *worse*.
Typically we can't have n bigger than 5.

Storage Problems with n -gram LMs

Storage: Need to store count for all n -grams you saw in the corpus.

$$P(\mathbf{w}|\text{students opened their}) = \frac{\text{count(students opened their } \mathbf{w})}{\text{count(students opened their)}}$$

Storage Problems with n -gram LMs

Storage: Need to store count for all n -grams you saw in the corpus.

$$P(\mathbf{w}|\text{students opened their}) = \frac{\text{count(students opened their } \mathbf{w})}{\text{count(students opened their)}}$$

Increasing n or increasing corpus increases model size!

Language Models: Smoothing

Smoothing: None

$$P(z | xy) \approx \frac{C(xyz)}{\sum_w C(xyw)} = \frac{C(xyz)}{C(xy)}$$

- Called Maximum Likelihood estimate.
- Lowest perplexity trigram on the training data.
- Terrible on test data: If no occurrences of $C(xyz)$, the probability is 0.

Smoothing: Add One

- What is $P(\text{sing}|\text{nuts})$? Zero? Leads to infinite perplexity!
- Add one smoothing:
 - Works very badly. DO NOT DO THIS

$$P(z | xy) \approx \frac{C(xyz) + 1}{C(xy) + V}$$

- Add delta smoothing:
 - Still very bad. DO NOT DO THIS

$$P(z | xy) \approx \frac{C(xyz) + \delta}{C(xy) + V\delta}$$

Smoothing: Simple Interpolation

$$P(z | xy) \approx \lambda \frac{C(xyz)}{C(xy)} + \mu \frac{C(yz)}{C(y)} + (1 - \lambda - \mu) \frac{C(z)}{C(\bullet)}$$

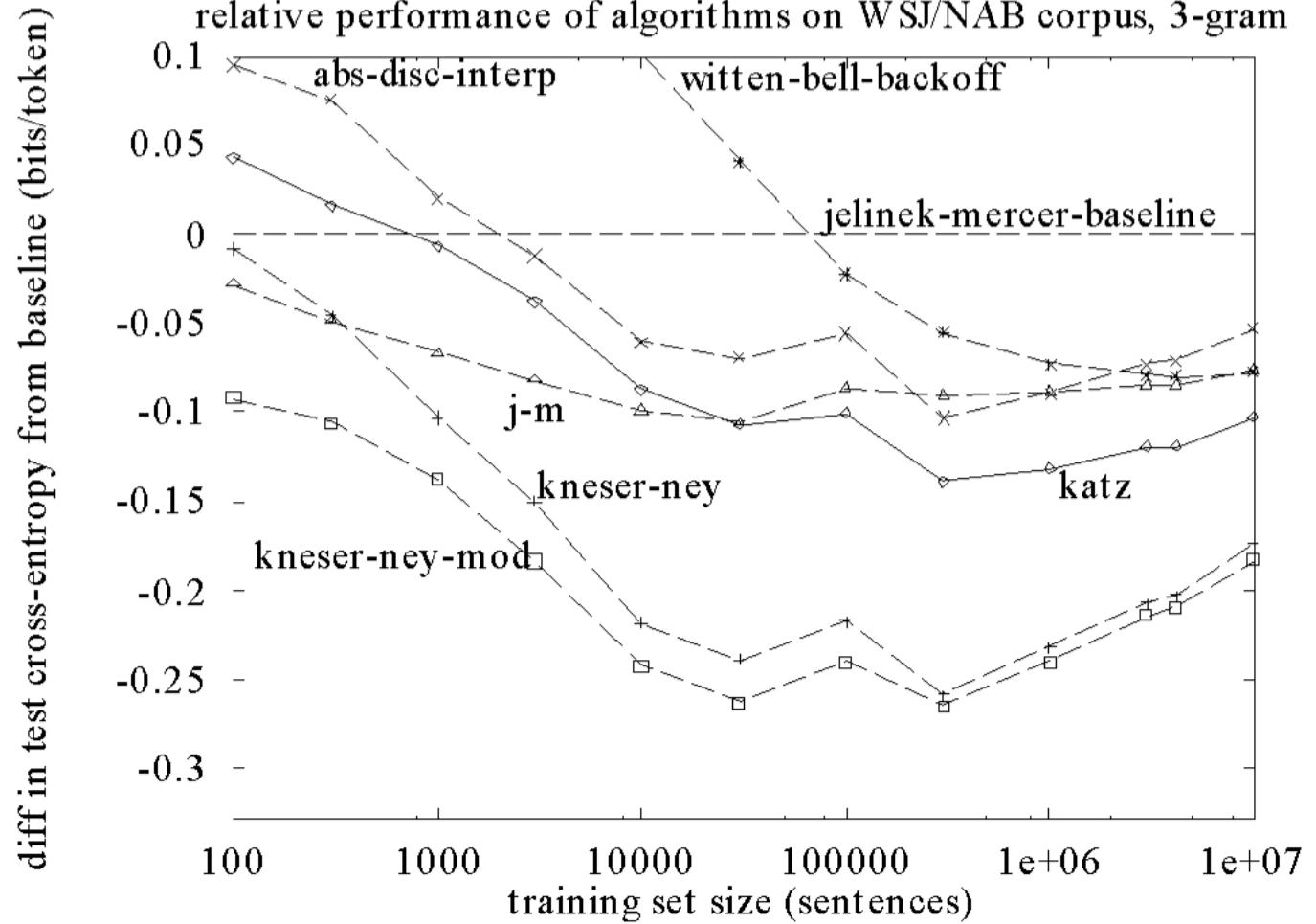
- ❑ Trigram is very context specific, very noisy
- ❑ Unigram is context-independent, smooth
- ❑ Interpolate Trigram, Bigram, Unigram for the best combination
- ❑ Find $0 < \lambda, \mu < 1$ by optimizing on “held-out” data
- ❑ Almost good enough

Smoothing: Finding the Best Parameter Values

- ❑ Split the data into training, “heldout”, test
- ❑ Try lots of different values for λ, μ on the heldout data, and pick the best one
- ❑ Test on the test data

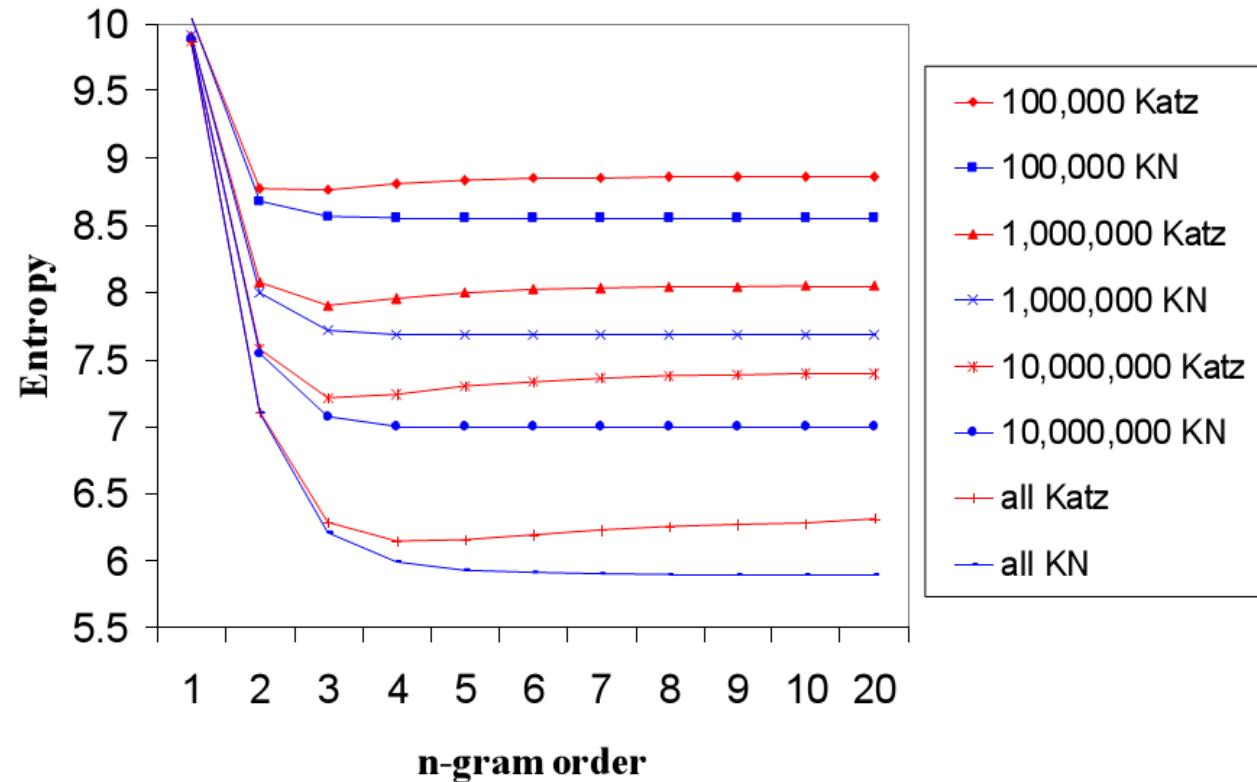
Many Smoothing Methods

- Good-Turing
- Kneser-Ney
- Jelinek-Mercer
- Witten-Bell
- Katz
- Discounting
- ...



Data >> Method?

- Having more data is better...



- ... but so is using a better model
- Another issue: $N > 3$ has huge costs in speech recognizers

Evaluating Language Models

Evaluation

- How can you tell a good language model from a bad one?
- Run a speech recognizer (or your application of choice), calculate the word error rate
 - Slow
 - Specific to your recognizer

Evaluation: Perplexity Intuition

- Ask ASR to recognize the digits: “0, 1, 2, 3, 4, 5, 6, 7, 8, 9”
 - easy – perplexity 10
- Ask ASR to recognize names at Microsoft
 - hard – perplexity 30,000
- Ask ASR to recognize “Operator” (1 in 4), “Technical support” (1 in 4), “sales” (1 in 4), 30,000 names (1 in 120,000) each
 - perplexity 54

The perplexity is a weighted equivalent branching factor.

Evaluation: Perplexity

- "A, B, C, D, E, F, G...Z"
 - *perplexity*: 26
- "Alpha, bravo, charlie, delta...yankee, zulu"
 - *Perplexity*: 26

The perplexity measures the language model difficulty, not the acoustic difficulty.

Perplexity: Math

- The perplexity is a geometric average inverse probability
- Imagine model: “Operator” (1 in 4),
“Technical support” (1 in 4),
“sales” (1 in 4)
30,000 names (1 in 120,000)

$$\sqrt[n]{\prod_{i=1}^n \frac{1}{P(w_i | w_{1..i-1})}}$$

- If all 30,004 are equally likely

$$\sqrt[30,004]{\frac{1}{4} \times \frac{1}{4} \times \frac{1}{4} \times \underbrace{\frac{1}{120,000} \times \cdots \times \frac{1}{120,000}}_{30,000}}$$

- The perplexity of the test data given the model is 119,829
- *Remarkable fact: the true model for data has the lowest possible perplexity*

Evaluation: Entropy

- Entropy = \log_2 perplexity

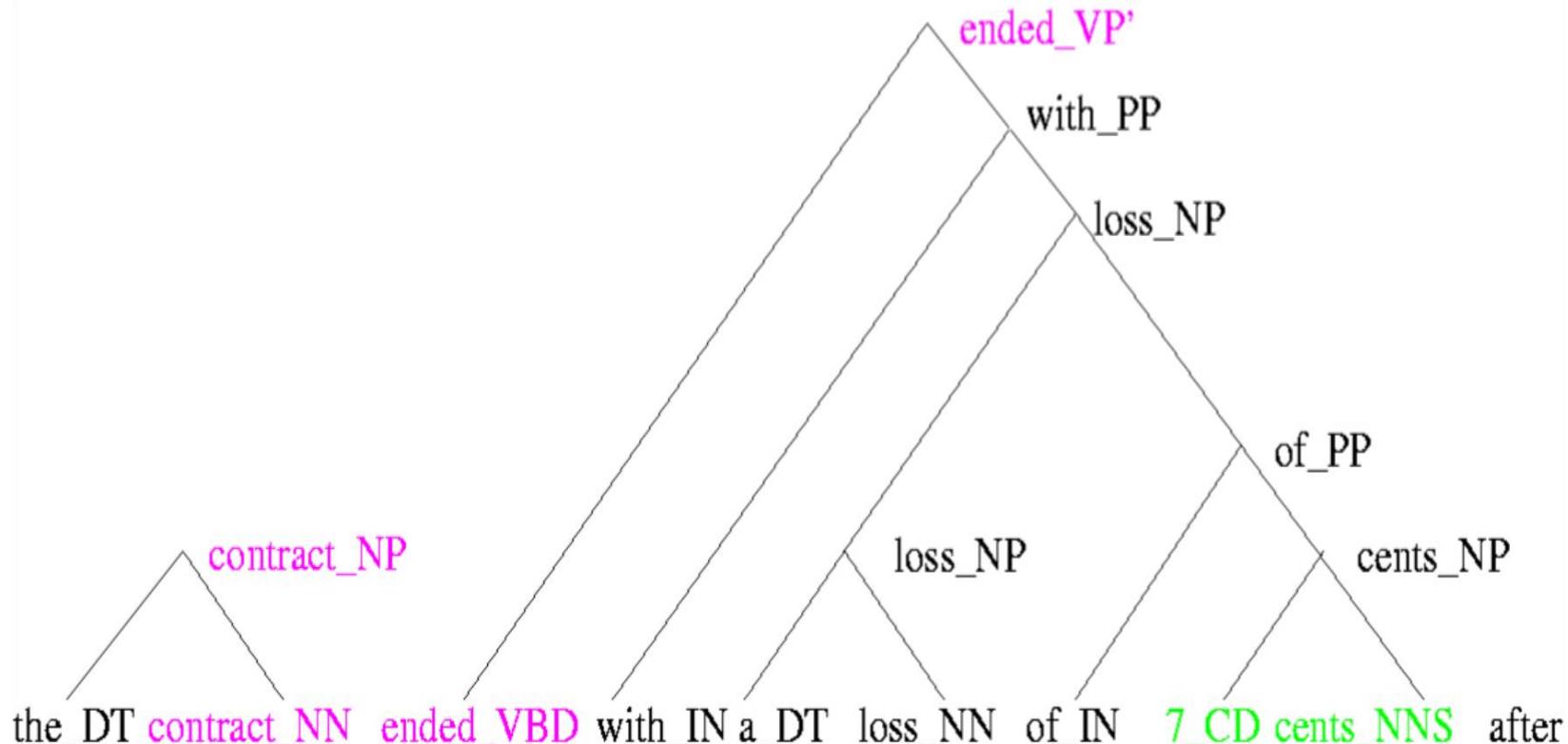
$$\log_2 \sqrt[n]{\prod_{i=1}^n \frac{1}{P(w_i | w_{1..i-1})}}$$

- Should be called a “cross-entropy of the model on the test data.”
- Remarkable fact: the entropy is the average number of bits per word required to encode the test data using this probability model, and an optimal coder. Called bits.

Syntactic Language Models

Structured Language Models

“The contract ended with a loss of 7 cents after”



Parsing vs. Trigram LM: Eugene Charniak's Experiments

Model	Perplexity
Trigram poor smoothing	167
Trigram deleted-interpolation	155
Trigram Kneser-Ney	145
Parsing	119

All experiments are trained on one million words of Penn treebank data, and tested on 80,000 words.

Neural Language Models

Fixed-Window Neural LM

output distribution

$$\hat{\mathbf{y}} = \text{softmax}(\mathbf{U}\mathbf{h} + \mathbf{b}_2) \in \mathbb{R}^{|V|}$$

hidden layer

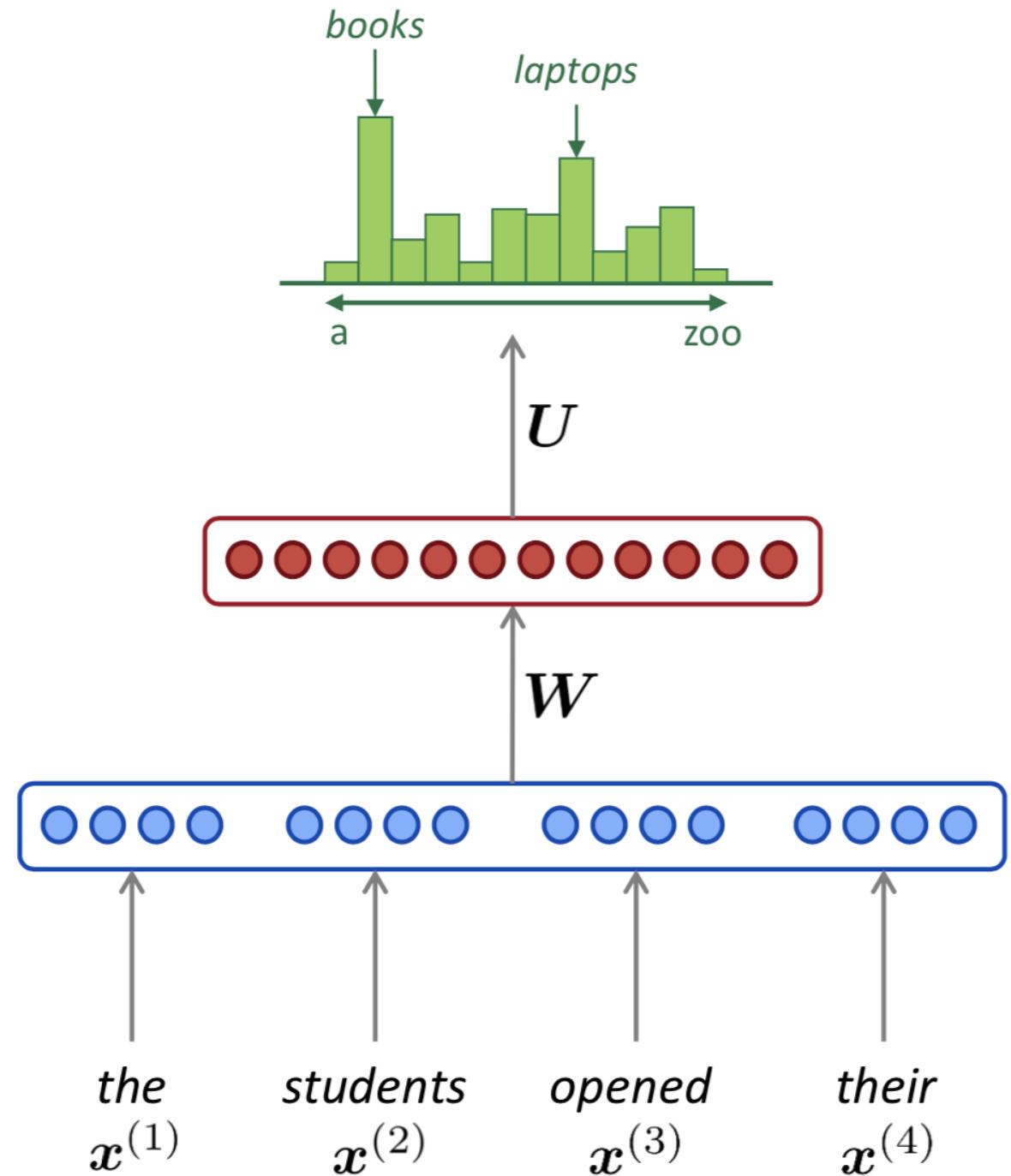
$$\mathbf{h} = f(\mathbf{W}\mathbf{e} + \mathbf{b}_1)$$

concatenated word embeddings

$$\mathbf{e} = [\mathbf{e}^{(1)}; \mathbf{e}^{(2)}; \mathbf{e}^{(3)}; \mathbf{e}^{(4)}]$$

words / one-hot vectors

$$\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}^{(4)}$$



Fixed-Window Neural LM

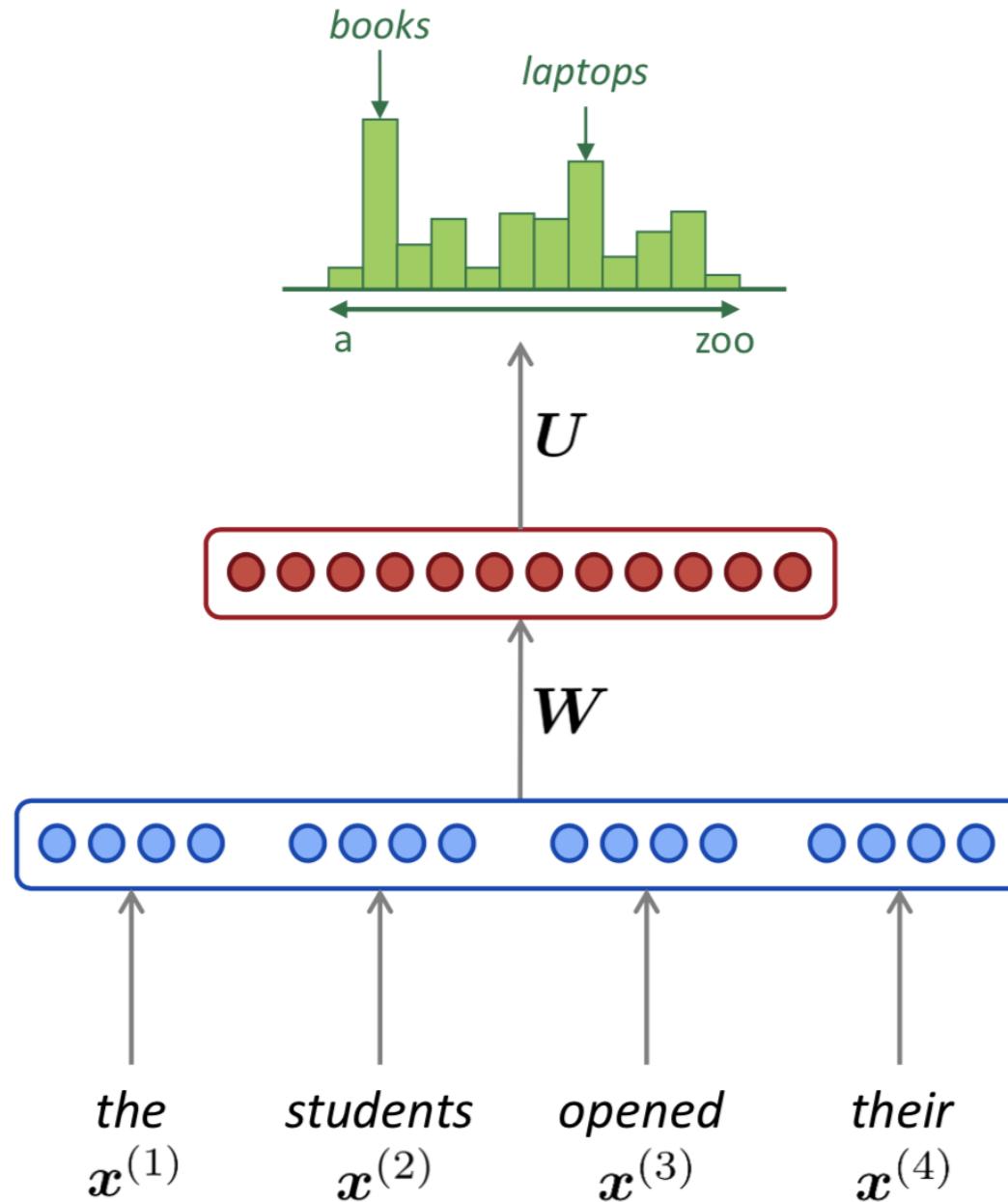
Improvements over n -gram LM:

- No sparsity problem
- Don't need to store all observed n -grams

Remaining **problems**:

- Fixed window is **too small**
- Enlarging window enlarges W
- Window can never be large enough!
- $x^{(1)}$ and $x^{(2)}$ are multiplied by completely different weights in W .
No symmetry in how the inputs are processed.

We need a neural architecture that can process *any length input*



$$\hat{\mathbf{y}}^{(4)} = P(\mathbf{x}^{(5)} | \text{the students opened their})$$

A RNN Language Model

output distribution

$$\hat{\mathbf{y}}^{(t)} = \text{softmax} (\mathbf{U}\mathbf{h}^{(t)} + \mathbf{b}_2) \in \mathbb{R}^{|V|}$$

hidden states

$$\mathbf{h}^{(t)} = \sigma (\mathbf{W}_h \mathbf{h}^{(t-1)} + \mathbf{W}_e \mathbf{e}^{(t)} + \mathbf{b}_1)$$

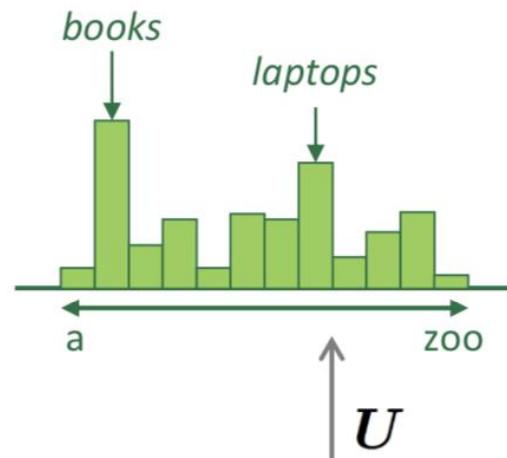
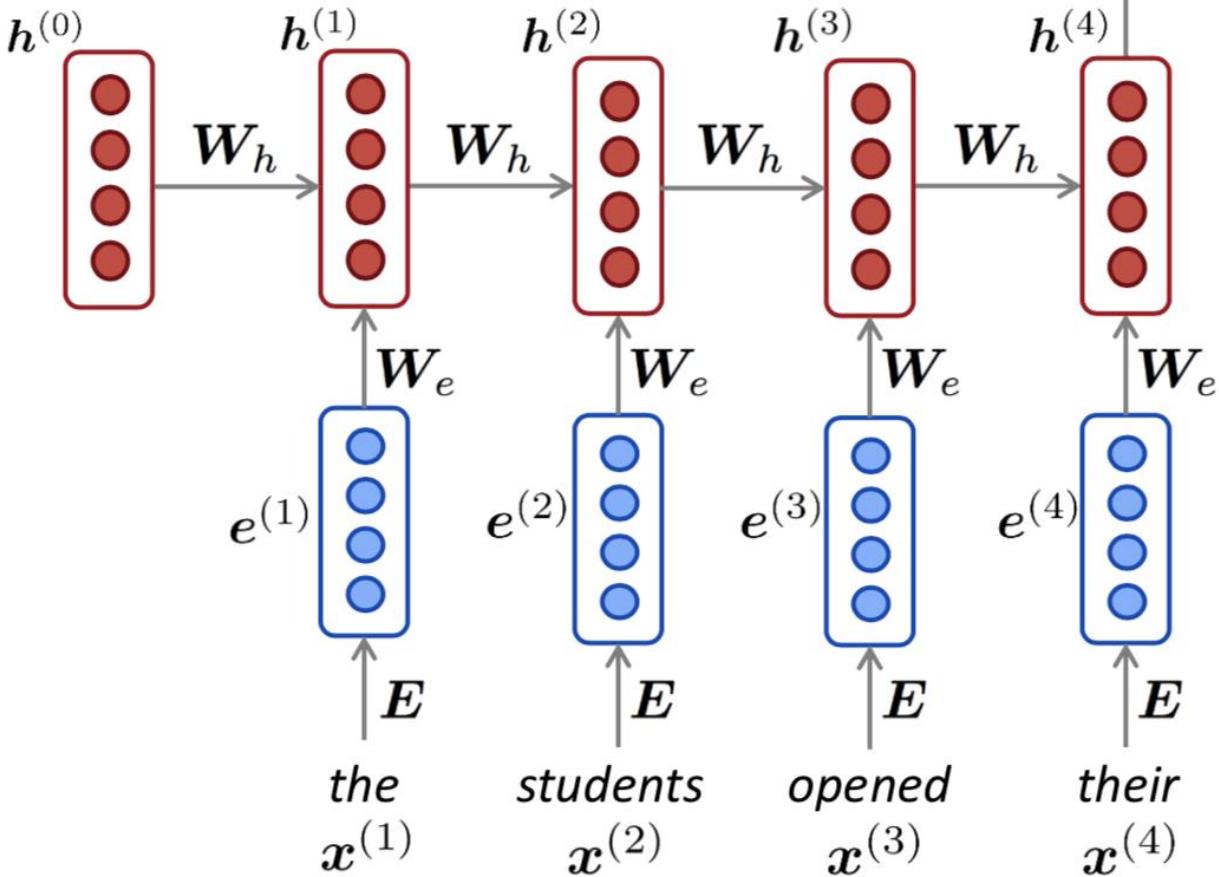
$\mathbf{h}^{(0)}$ is the initial hidden state

word embeddings

$$\mathbf{e}^{(t)} = \mathbf{E}\mathbf{x}^{(t)}$$

words / one-hot vectors

$$\mathbf{x}^{(t)} \in \mathbb{R}^{|V|}$$



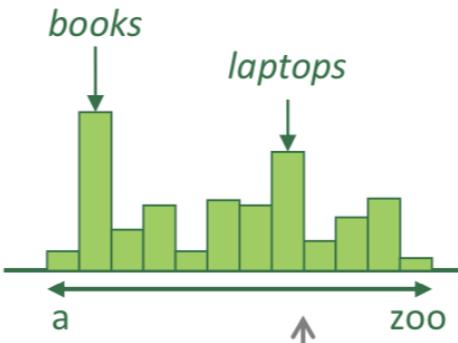
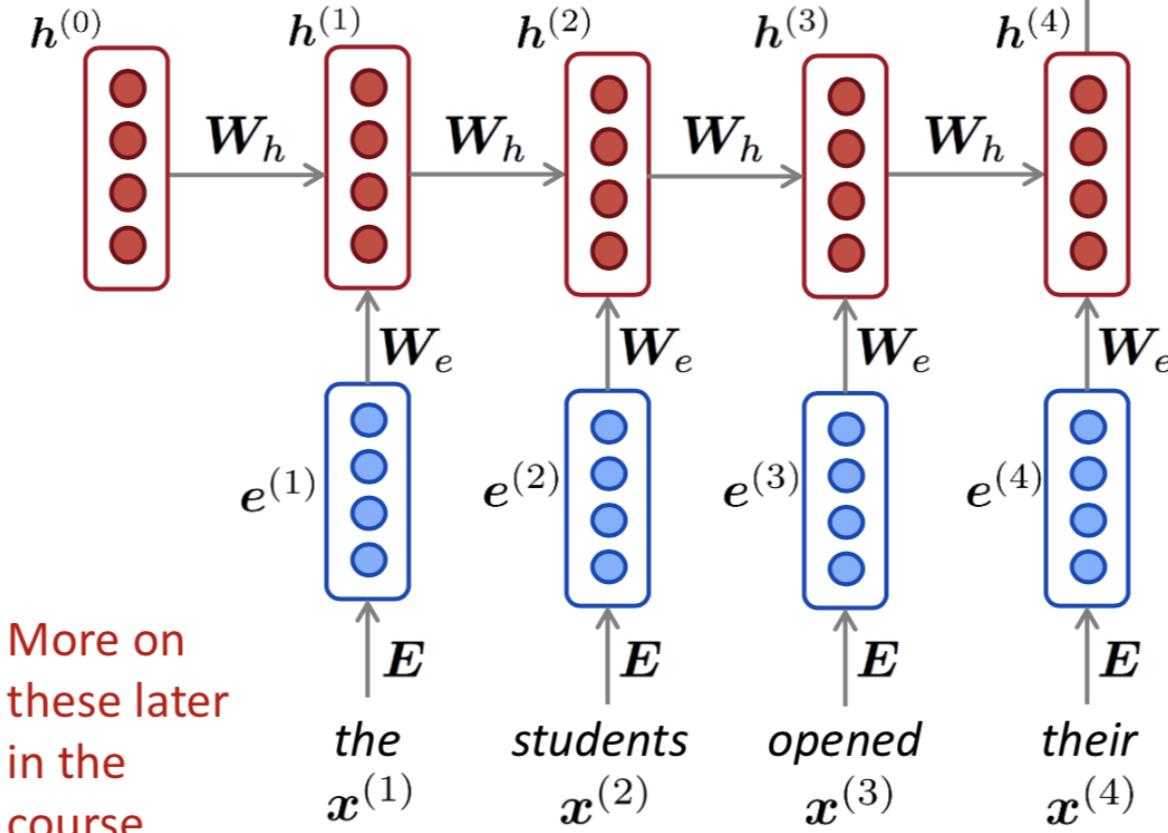
A RNN Language Model

RNN Advantages:

- Can process **any length** input
- Computation for step t can (in theory) use information from **many steps back**
- Model size **doesn't increase** for longer input
- Same weights applied on every timestep, so there is **symmetry** in how inputs are processed.

RNN Disadvantages:

- Recurrent computation is **slow**
- In practice, difficult to access information from **many steps back**



Transformer: GPT-2



THE TRANSFORMER

DECODER BLOCK

Feed Forward Neural Network

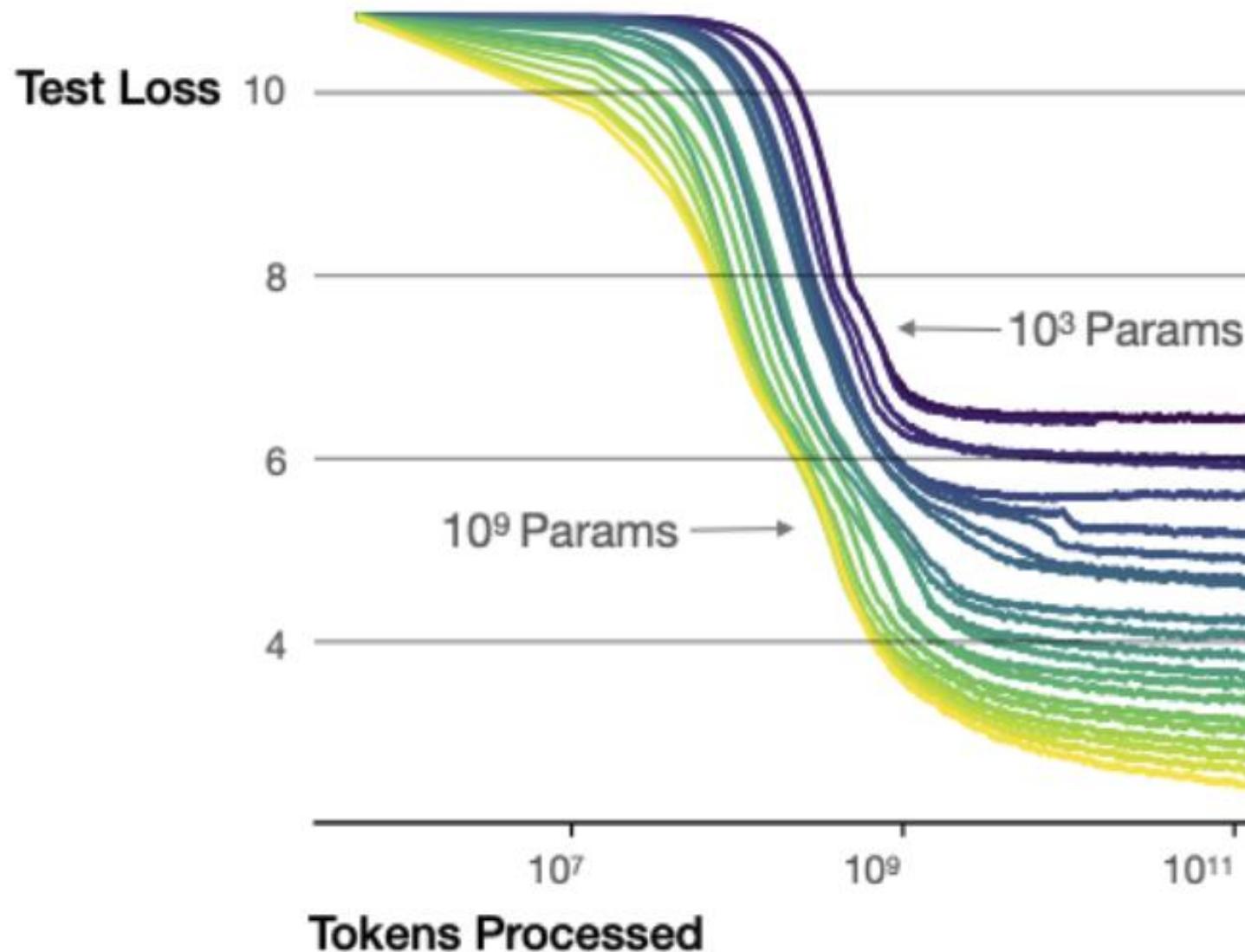
Encoder-Decoder Self-Attention

Masked Self-Attention

Input

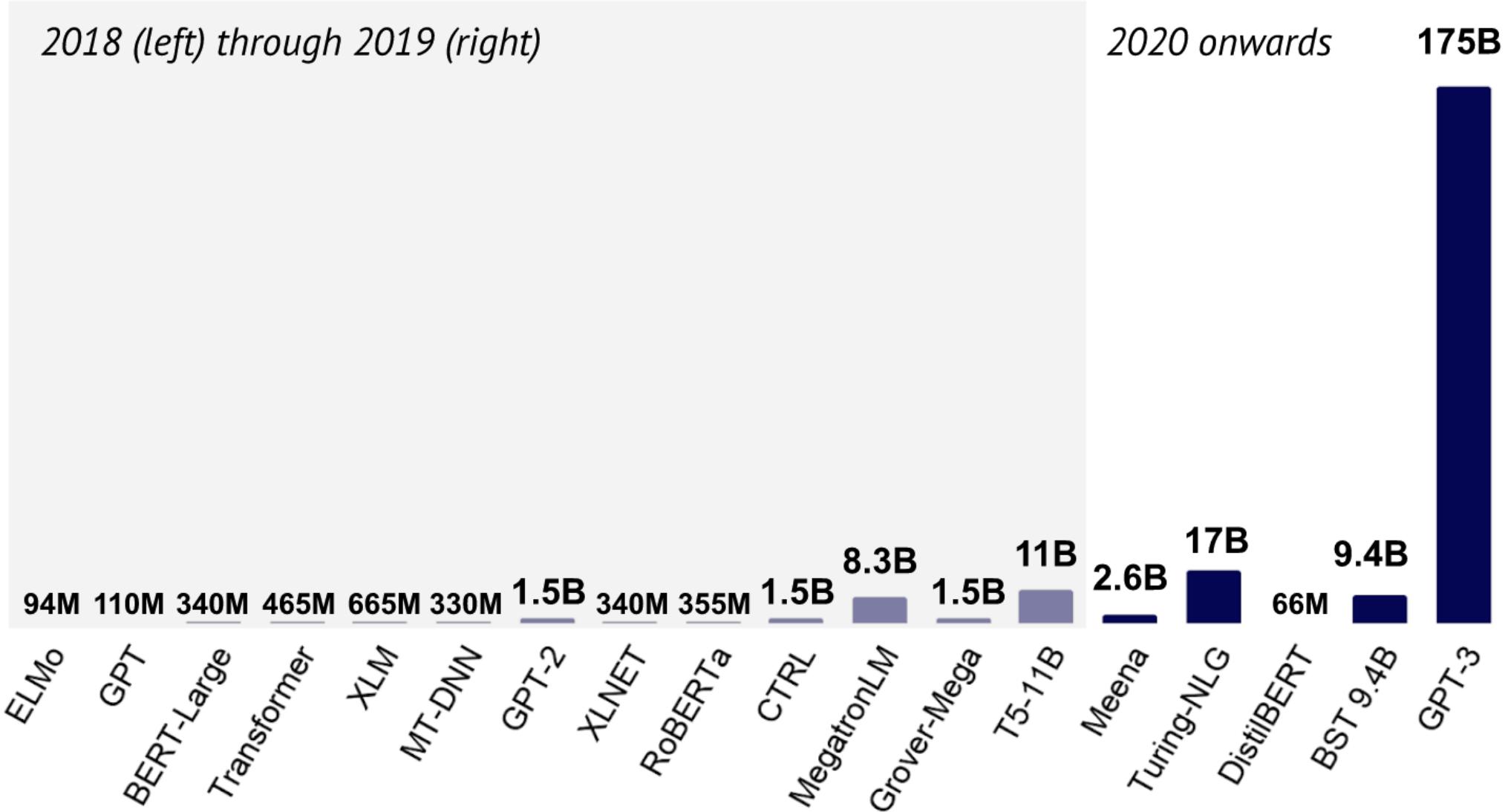
<s>	robot	must	obey						512
1	2	3	4	5	6				

Larger Models Need Less Training Examples



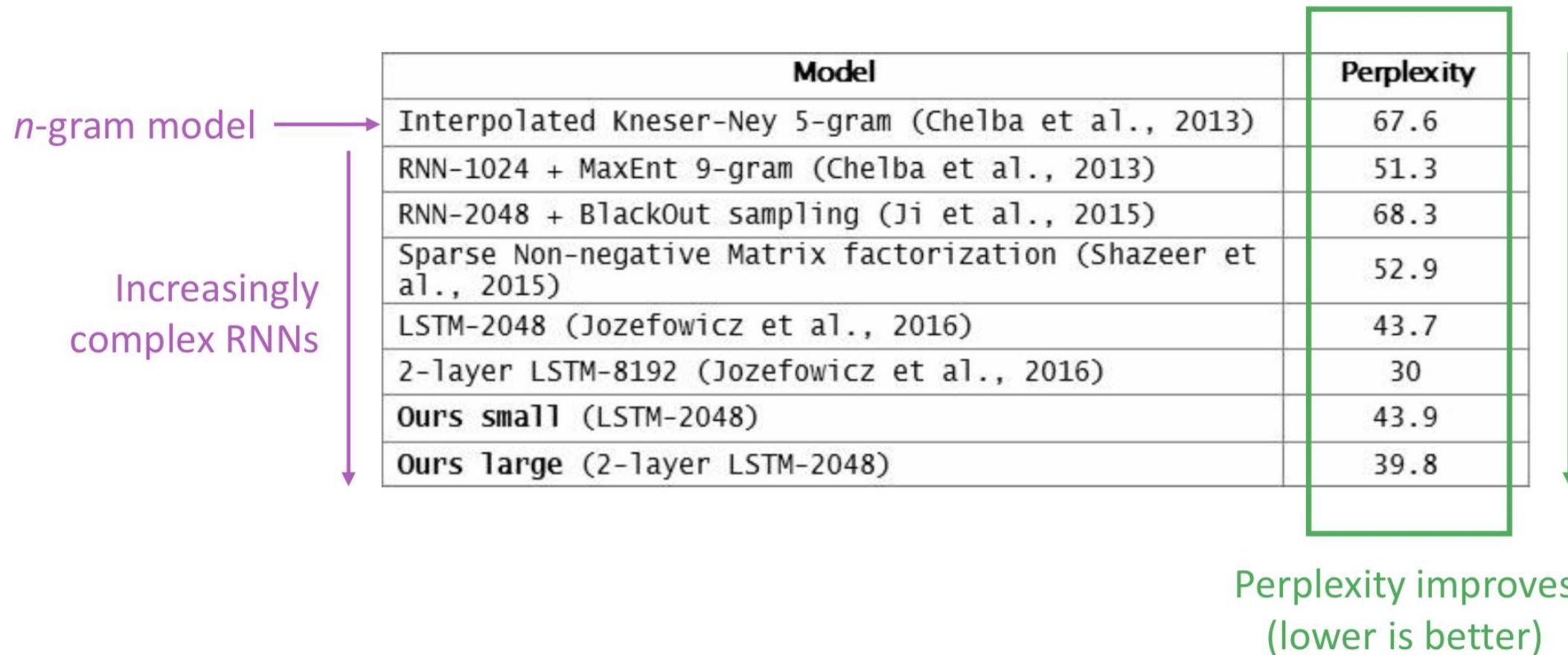
The Model Size Race

(NOTE: the human brain has 125B neurons...)

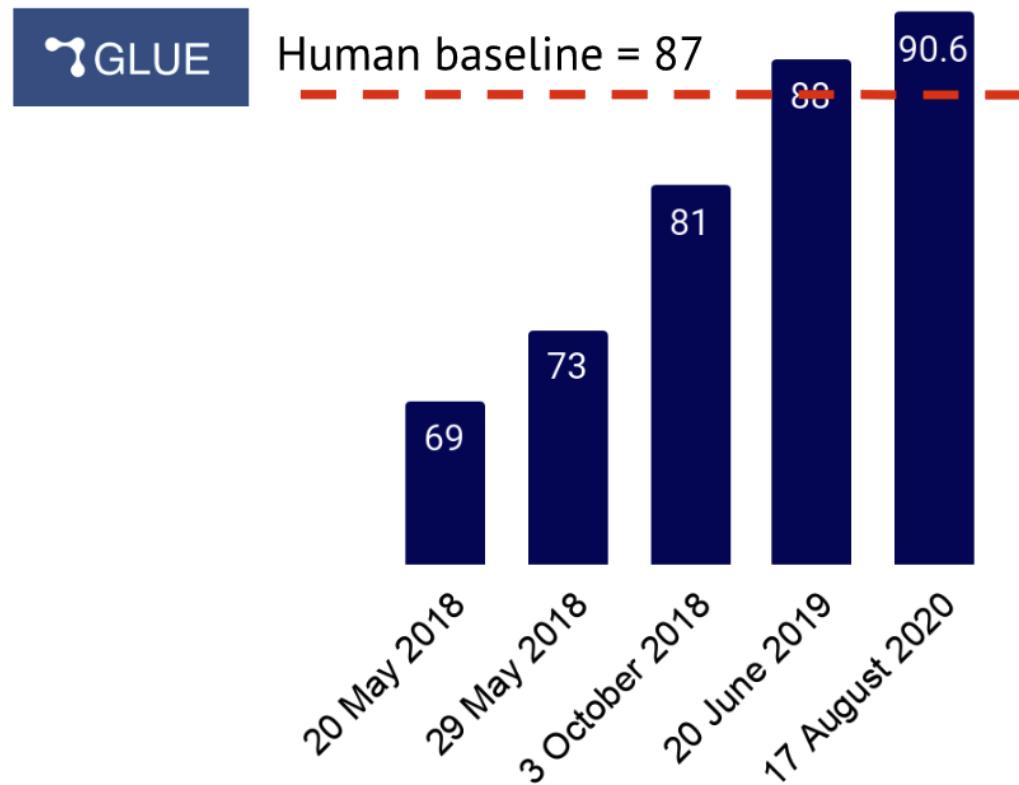


Language Model Applications: Benchmarking NLP

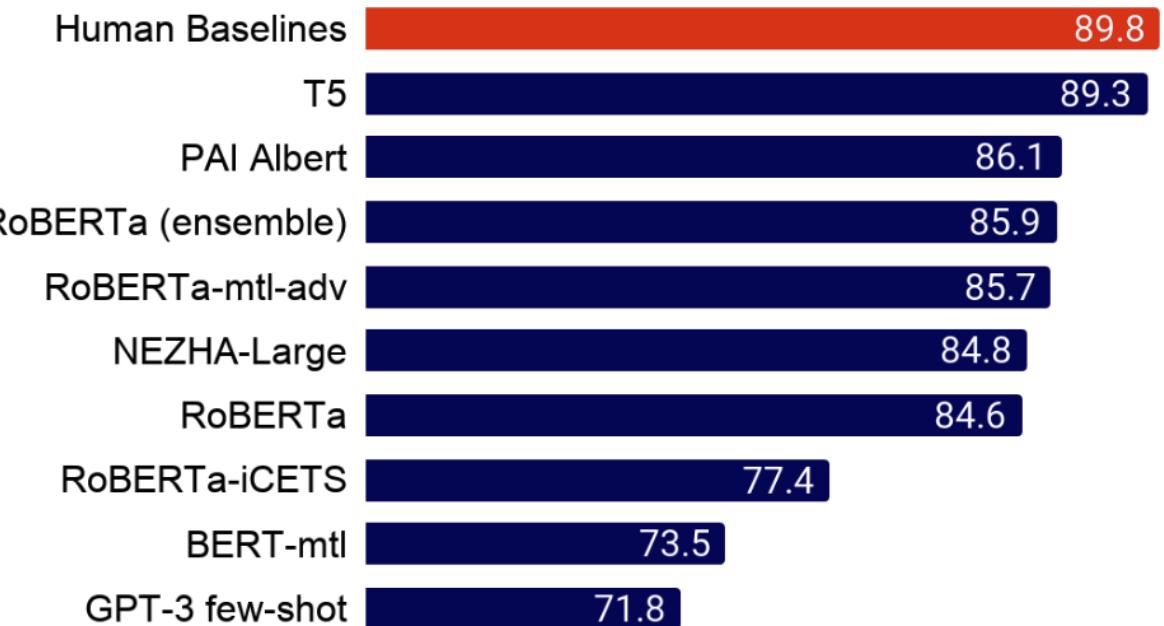
RNNs have greatly improved perplexity



Super-Human Performance on the GLUE Benchmark



SuperGLUE



stateof.ai

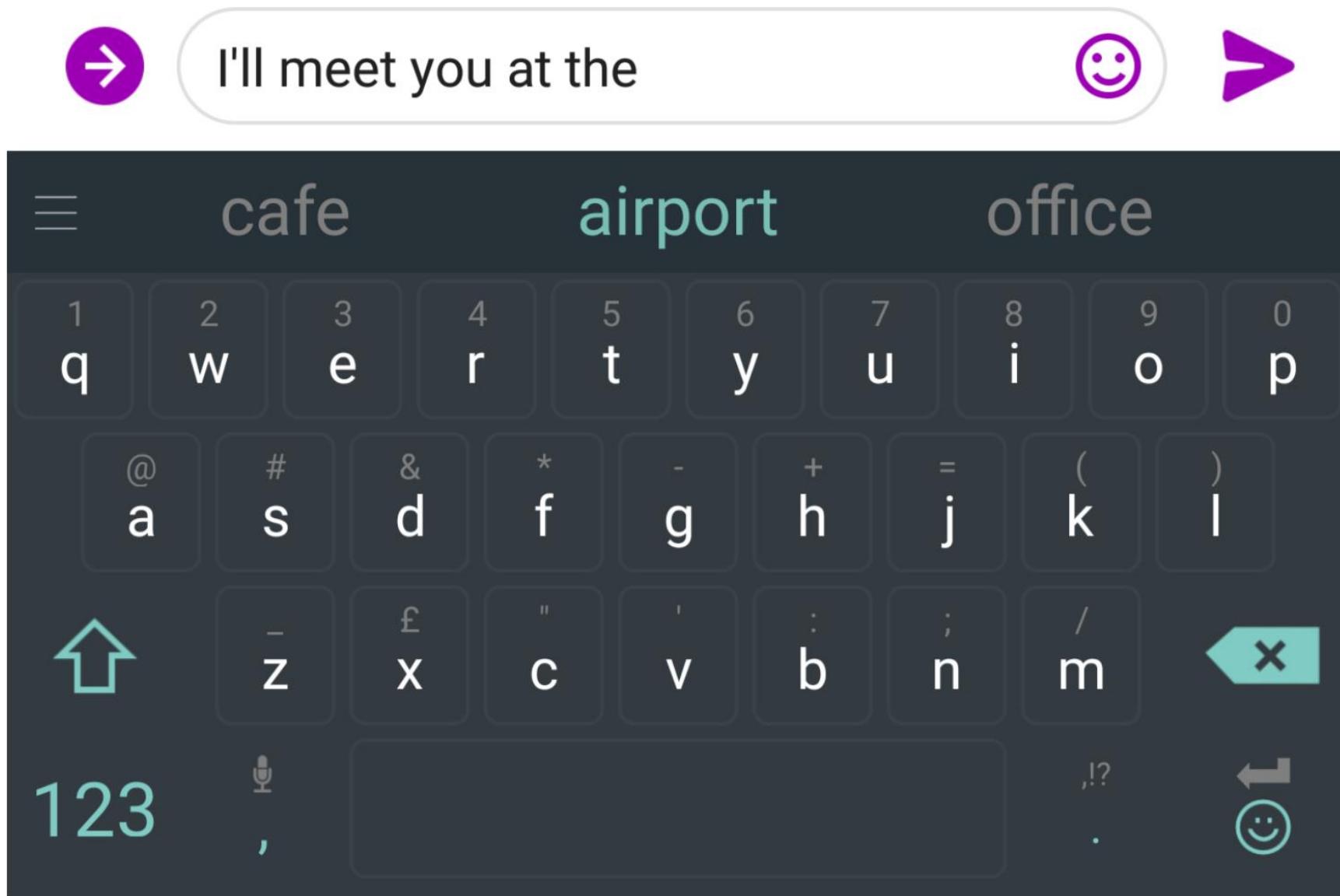
Superhuman Performance on the SuperGLUE Language Understanding Benchmark

<https://super.gluebenchmark.com/leaderboard>

Rank	Name	Model	URL	Score
1	Liam Fedus	ST-MoE-32B		91.2
2	Microsoft Alexander v-team	Turing NLR v5		90.9
3	ERNIE Team - Baidu	ERNIE 3.0		90.6
4	Yi Tay	PaLM 540B		90.4
5	Zirui Wang	T5 + UDG, Single Model (Google Brain)		90.4
6	DeBERTa Team - Microsoft	DeBERTa / TuringNLRv4		90.3
7	SuperGLUE Human Baselines	SuperGLUE Human Baselines		89.8

Application: Next Word Prediction

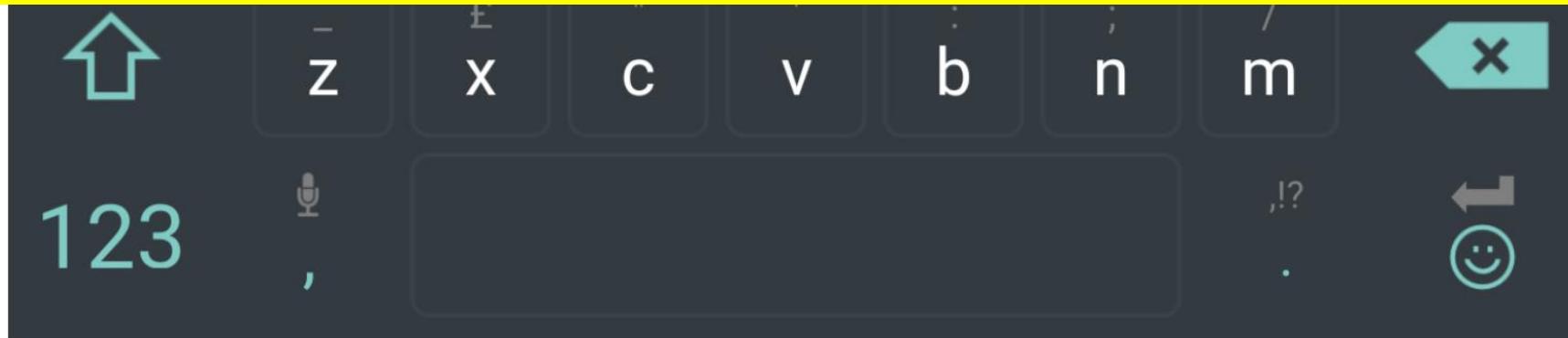
Application: Next Word Prediction



Application: Next Word Prediction

The image shows a messaging application interface. At the top, there's a message bubble containing the text "I'll meet you at the" followed by a smiley face emoji and a purple send arrow icon. Below the message is a dark grey horizontal bar with three small white icons: a menu icon, a "cafe" icon, an "airport" icon, and an "office" icon. The main area of the screen is filled with a large yellow background containing the text "How Could an LM Predict the Next Word?" in a bold, black, sans-serif font.

How Could an LM Predict the Next Word?



Application: Next Phrase Prediction

 what is the cheapest

 what is the cheapest - Google Search

 what is the cheapest **car in the world**

 what is the cheapest **tesla**

 what is the cheapest **country in the world**

 what is the cheapest **currency**

 what is the cheapest **thing in the world**

 what is the cheapest **rolex**

 what is the cheapest **car in dubai**

 what is the cheapest **limited on roblox**

 what is the cheapest **day to book a flight**

 what is the weather today in

 what is the weather today in
37°C Sun - Yas North - Abu Dhabi

 what is the weather today in **dubai**

 what is the weather today in **abu dhabi**

 what is the weather today in **sharjah**

 what is the weather today in **salalah**

 what is the weather today in **georgia**

 what is the weather today in **ras al khaimah**

 what is the weather today in **uganda**

 what is the weather today in **canada**

Application: Machine Translation (MT)

Machine Translation (MT)

أعلنت مؤسسة أيادي الخير نحو آسيا «روتا» عن طلاق النسخة الخامسة من فعالية «أرجل عجلات»، الحدث الجتبي الأضخم من نوعه في دولة قطر، لتسليط الضوء الشاملة بجهود الشراكة واللجنة القطرية اليابانية ٢٠١٢.

اللية في ١٧ فبراير بحديقة متحف
ي، وتحفي بمرور أربعين عاماً
الطيبة التي تجمع بين البلدين.

مدير مؤسسة أيادي الخير
حضر إلى الدوحة ٤٠ طالباً يابانياً
خصوصاً من أجل المشاركة في هذه الفعالية. كما
سيشهد زوار «أرجل وعجلات» ٢٠١٢ العديد من

رض سوران التراثي

- **learns** from human translation
- extracts **knowledge** and builds **models**
- uses the models to **translate** new sentences

English

Reach Out to Asia (ROTA) has announced its fifth 'Wheels 'n' Heels' Qatar's largest annual community event, which will promote ROTA's partnership with the Qatar Japan 2012 Committee.

Held at the Museum of Islamic Art in Doha on 10 February, the event will celebrate 40 years of cordial relations between the two countries. Essa Al Mannai, ROTA Director, said: "A group of 40 Japanese students are traveling to Doha especially to take part in our event.



UN Parallel Corpus

EN

AR

ES

FR

RU

ZH

1. Meetings normally should be held during regular meeting hours, **namely**, from 10 a.m. to 1 p.m. and from 3 p.m. to 6 p.m., on working days;

2. {1>Approves<1}, accordingly, with effect from 1 April 2005, the changes in the pension adjustment system set out in the annex to the present resolution, namely:

{1>Aware<1} of the continuing needs of Palestine refugees throughout all the fields of operation, **namely** the West Bank and the Gaza Strip, Jordan, Lebanon and the Syrian Arab Republic,

١ - تُعقد الاجتماعات عادة خلال ساعات الاجتماعات العادلة، أي من الساعة ٠٠/١٠ إلى الساعة ١٣/٠٠ ومن الساعة ٠٠/١٥ إلى الساعة ٠٠/١٨، في أيام العمل؛

٢ - {١<توافق>} [١]، بفضل ذلك واعتباراً من ١ نيسان/أبريل ٢٠٠٥، على التغييرات المدخلة على نظام تسوية المعاشات التقاعدية والموضحة في مرفق هذا القرار، أي:

1. Normalmente las reuniones deberán celebrarse en horas habituales de reunión, es decir, de las 10.00 a las 13.00 horas y de las 15.00 a las 18.00 horas, en días laborables;

2. {1>Aprueba<1} por tanto, con efecto a partir del ١º de abril de ٢٠٠٥، los cambios en el sistema de ajuste de las pensiones indicados en el anexo de la presente resolución, a saber:

{١<وإذ تدرك>} [١] الاحتياجات المستمرة لللاجئين الفلسطينيين في جميع ميادين العمل، أي في الضفة الغربية وقطاع غزة والأردن والجمهورية العربية السورية ولبنان،

{1>Consciente<1} de las persistentes necesidades de los refugiados de Palestina en todas las zonas de operaciones, a saber, la Ribera Occidental y la Faja de Gaza, Jordania, el Líbano y la República Árabe Siria,

1. Les séances doivent normalement se tenir aux heures normales de réunion, c'est-à-dire de 10 heures à 13 heures et de 15 heures à 18 heures, les jours ouvrables ;

2. {1>Approuve<1} en conséquence, avec effet au 1er avril 2005, les modifications du système d'ajustement des pensions présentées dans l'annexe à la présente résolution, qui consistent à :

1. Заседания следует, как правило, проводить в обычно отведенное для заседаний время, а именно с 10 ч. 00 м. до 13 ч. 00 м. и с 15 ч. 00 м. до 18 ч. 00 м. в рабочие дни;

2. {1>утверждает<1} соответственно вводимые с 1 апреля 2005 года изменения в системе пенсионных корректировок, изложенные в приложении к настоящей резолюции и заключающиеся в следующем:

{1>учитывая<1} сохраняющиеся нужды палестинских беженцев во всех районах осуществления деятельности, а именно по Западному берегу и в Секторе Газа, Иордании, Ливане и Сирийской Арабской Республике,

1. 会议通常应在工作日正常开会时间，即上午10时至下午1时，下午3时至6时举行；

2. 据此{1>核准<1}自2005年4月1日起对养恤金制度作本决议附件所述的改动：

{1>认识到<1}在所有活动区即西岸和加沙地带、约旦、黎巴嫩和阿拉伯叙利亚共和国内巴勒斯坦难民的持续需要，

Translation as Codebreaking?

"Also knowing nothing official about, but having guessed and inferred considerable about, the powerful new mechanized methods in cryptography—methods which I believe succeed even when one does not know what language has been coded—one naturally wonders if the problem of translation could conceivably be treated as a problem in cryptography. When I look at an article in Russian, I say: 'This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode.' "



Warren Weaver (1955:18, quoting a letter he wrote in 1947)

Translation as Codebreaking?

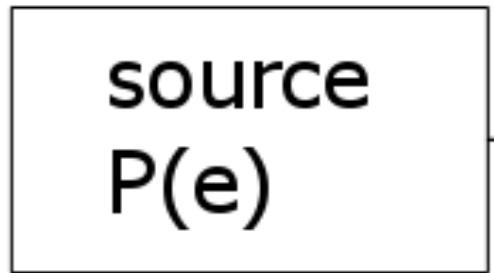
Это действительно написано по-английски .

This is really written in English .

The diagram illustrates the process of translating Russian text into English. Above the English sentence, the Russian phrase 'Это действительно написано по-английски .' is displayed in red. Below it, the English sentence 'This is really written in English .' is also in red. Seven red arrows point from the Russian words to their corresponding words in the English sentence: 'Это' points to 'This', 'действительно' points to 'really', 'написано' points to 'written', 'по-английски' points to 'in English', and the final '.' points to the final '.' at the end of the sentence.

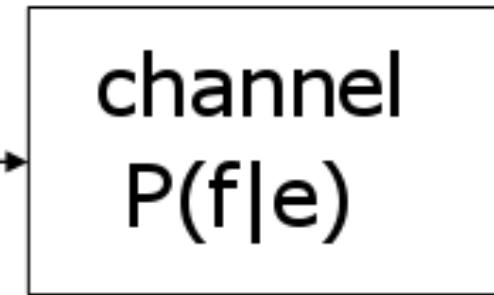
MT System Components

Language Model



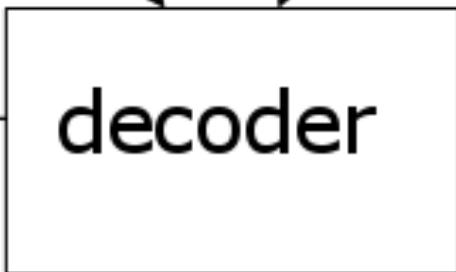
e

Translation Model



f

best
e



observed
f

$$\underset{e}{\operatorname{argmax}} P(e|f) = \underset{e}{\operatorname{argmax}} P(f|e)P(e)$$

The Main Equation of Statistical Machine Translation

$$\hat{e} = \operatorname{argmax}_e P(e | f)$$

The Main Equation of Statistical Machine Translation

$$\hat{e} = \operatorname{argmax}_e P(e | f)$$

Bayes' theorem

$$= \operatorname{argmax}_e P(e) \times P(f | e) / P(f)$$

The Main Equation of Statistical Machine Translation

$$\hat{e} = \operatorname{argmax}_e P(e | f)$$

Bayes' theorem

$$= \operatorname{argmax}_e P(e) \times P(f | e) / P(f)$$

$$= \operatorname{argmax}_e P(e) \times P(f | e)$$

The Main Equation of Statistical Machine Translation

$$\hat{e} = \operatorname{argmax}_e P(e | f)$$

What do we gain from making it more complex?

$$= \operatorname{argmax}_e P(e) \times P(f | e)$$

Combining $P(e)$ and $P(f|e)$

How do we translate to English *une fleur rouge*?

	$P(e)$	$P(f e)$	$P(e).P(f e)$
a flower red	↓	↑	↓
red flower a	↓	↑	↓
flower red a	↓	↑	↓
a red dog	↑	↓	↓
dog cat mouse	↓	↓	↓
a red flower	↑	↑	↑



<https://unsplash.com/photos/oO2uSvze7HQ>

Language Model: P(e)

- Goal: prefer “good” over “bad” sentences
 - “good” ≠ grammatical
 - “bad” ≈ low probability

- Examples (grammaticality):
 - *I do not like strong tea.* 
 - *I do not like powerful tea.* 
 - *I like strong tea not.* 
 - *Like not tea strong do I.* 

Example: Grammatical but Low-Probability Text

Eye halve a spelling checker
It came with my pea sea
It plainly marks four my revue
Miss steaks eye kin knot sea.

Eye strike a key and type a word
And weight four it two say
Weather eye am wrong oar write
It shows me a strait a weigh.

As soon as a mist ache is maid
It nose bee fore two long
And eye can put the error rite
Its rare lea ever wrong.

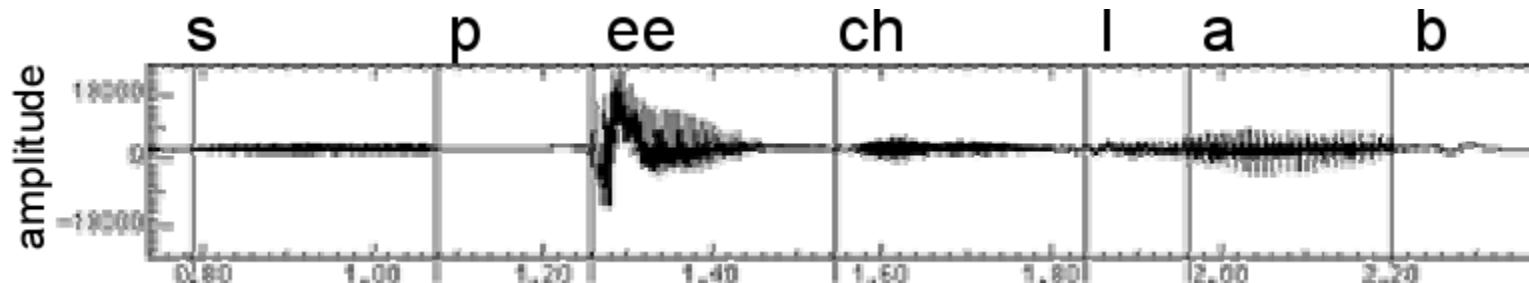
Eye have run this poem threw it
I am shore your pleased two no
Its letter perfect awl the weigh
My checker tolled me sew.

**Which Translation
Direction Would be Easier:
Arabic → English
or
English → Arabic?**

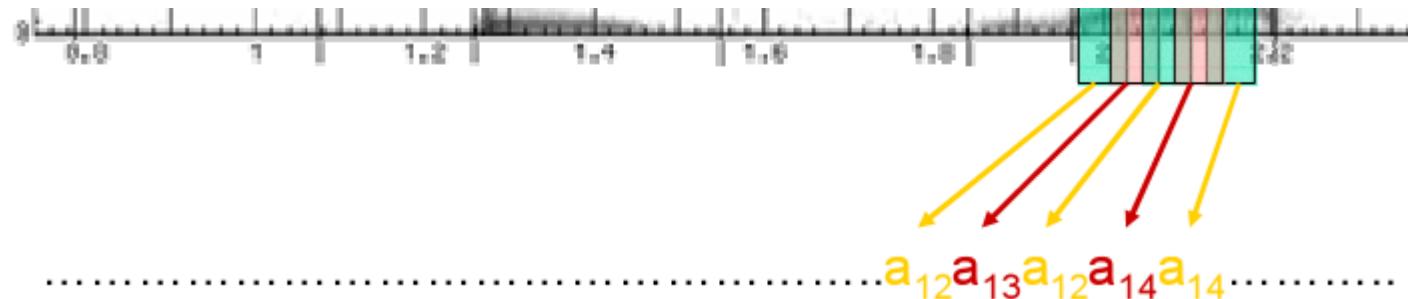
Application: Automatic Speech Recognition (ASR)

Speech Recognition Basics

- Frequency gives pitch; amplitude gives volume



- Frequencies at each time slice processed into observation vectors



The Noisy Channel Model for ASR

- We want to predict a sentence given acoustics:

$$w^* = \arg \max_w P(w|a)$$

- The noisy channel approach:

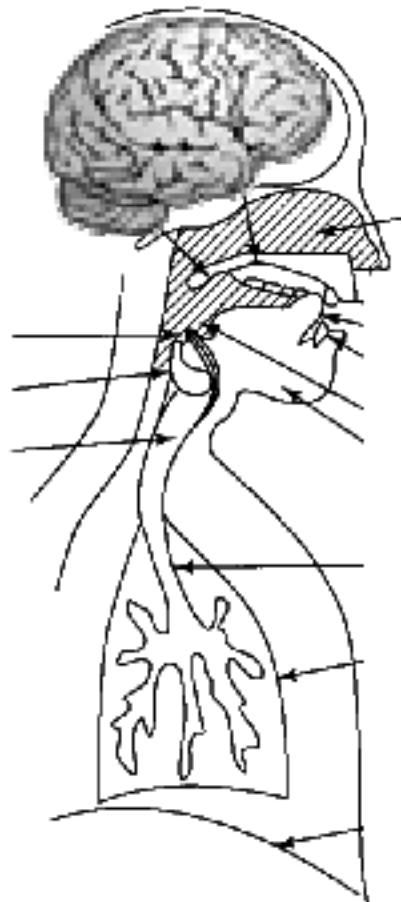
$$w^* = \arg \max_w P(w|a)$$

$$= \arg \max_w P(a|w)P(w)/P(a)$$

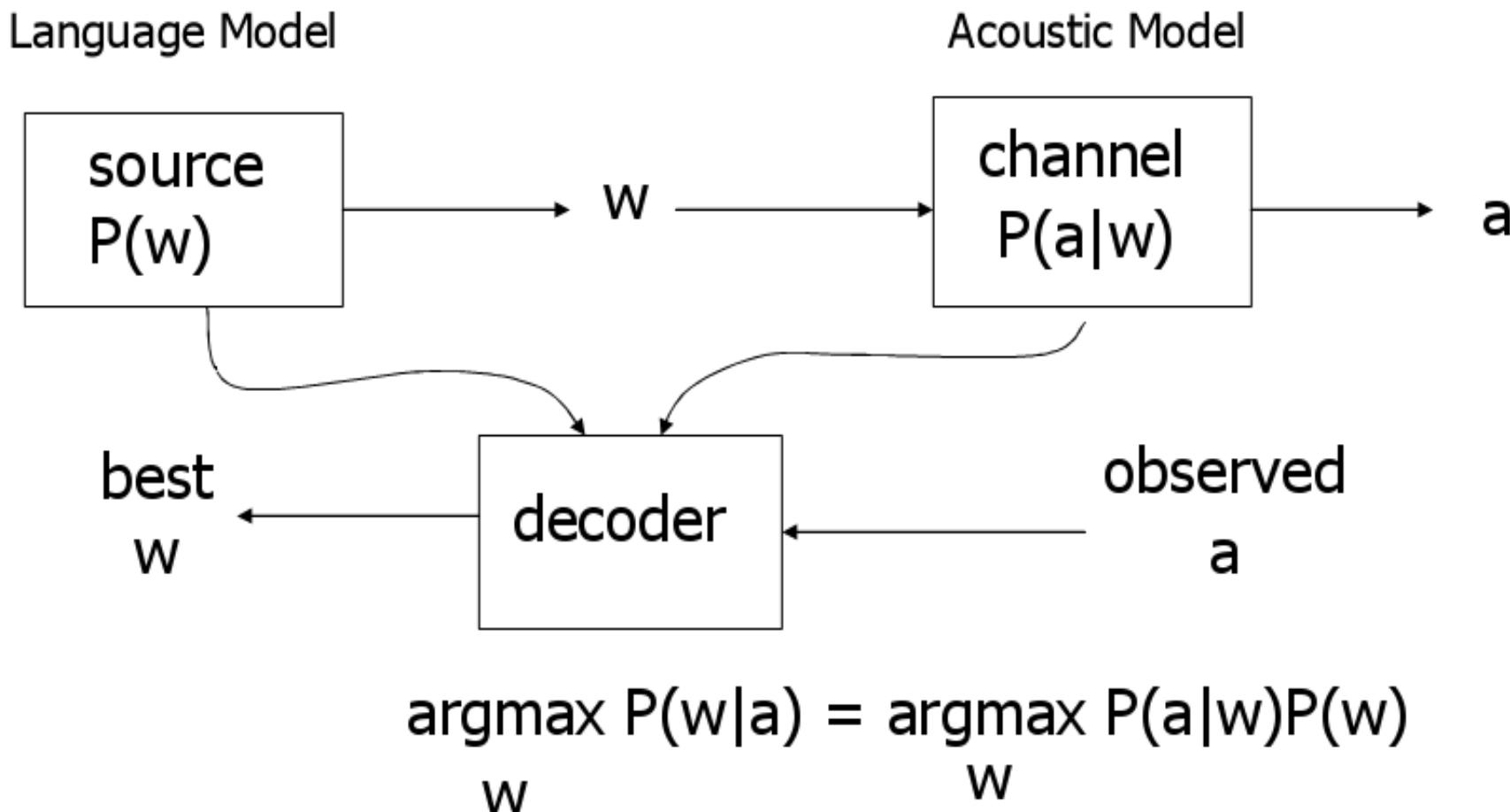
$$\propto \arg \max_w P(a|w)P(w)$$

Acoustic model: HMMs over word positions with mixtures of Gaussians as emissions

Language model:
Distributions over sequences of words (sentences)



The Noisy Channel Model for ASR



Other Noisy-Channel Processes

- ❑ Handwriting recognition

$$P(\text{text} \mid \text{strokes}) \propto P(\text{text})P(\text{strokes} \mid \text{text})$$

- ❑ OCR

$$P(\text{text} \mid \text{pixels}) \propto P(\text{text})P(\text{pixels} \mid \text{text})$$

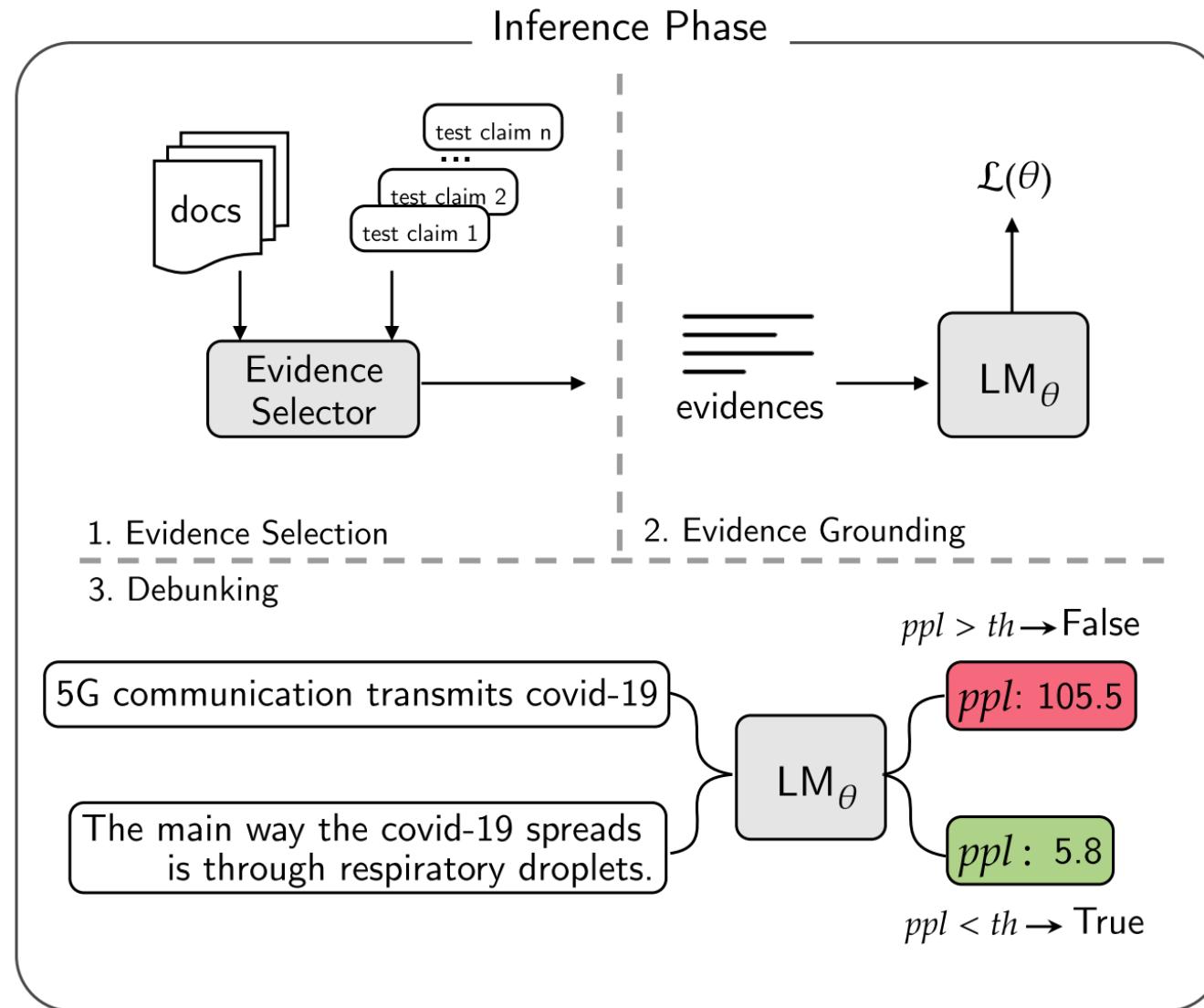
- ❑ Spelling correction

$$P(\text{text} \mid \text{typos}) \propto P(\text{text})P(\text{typos} \mid \text{text})$$

- ❑ ...

Language Model Applications: Text Classification

Fact-Checking Using Perplexity



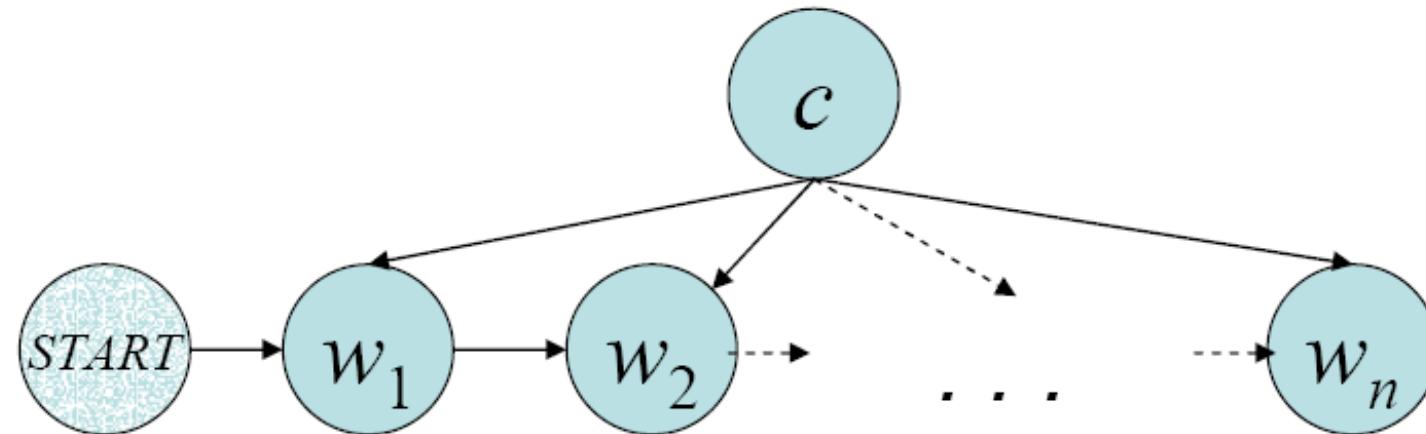
Fact-Checking Using Perplexity

False Claims	Perplexity
Ordering or buying products shipped from overseas will make a person get COVID-19.	556.2
Sunlight actually can kill the novel COVID-19.	385.0
5G helps COVID-19 spread.	178.2
Home remedies can cure or prevent the COVID-19.	146.2
True Claims	Perplexity
The main way the COVID-19 spreads is through respiratory droplets.	5.8
The most common symptoms of COVID-19 are fever, tiredness, and dry cough.	6.0
The source of SARS-CoV-2, the coronavirus (CoV) causing COVID-19 is unknown.	8.1
Currently, there is no vaccine and no specific antiviral medicine to prevent or treat COVID-19.	8.4

Class-Conditional Language Models

- Can have a topic variable for other language models
- Could be characters instead of words, used for language ID
- Could sum out the topic variable and use as a LM

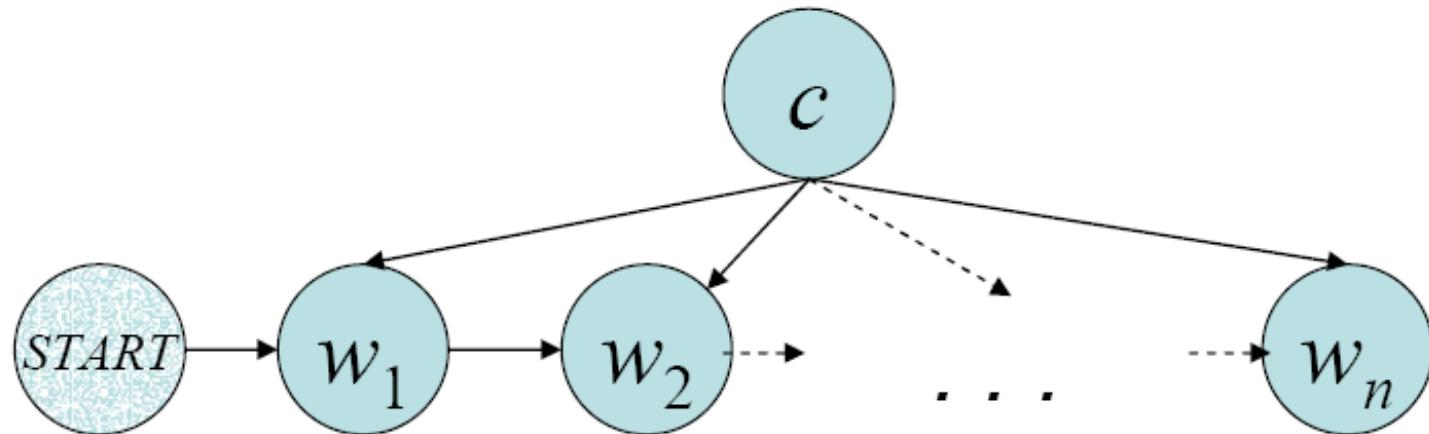
$$P(c, w_1, w_2, \dots, w_n) = P(c) \prod_i P(w_i | w_{i-1}, c)$$



Class-Conditional Language Models

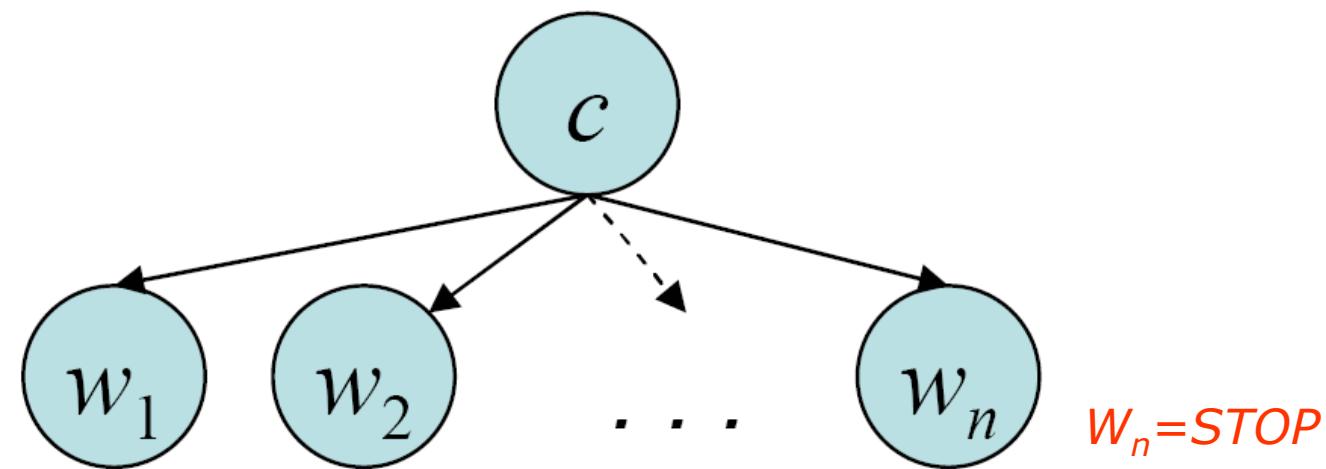
**How does it differ from
a standard bi-gram LM?**

$$P(c, w_1, w_2, \dots, w_n) = P(c) \prod_i P(w_i | w_{i-1}, c)$$



Compare to Naïve Bayes

$$P(c, w_1, w_2, \dots, w_n) = P(c) \prod_i P(w_i | c)$$



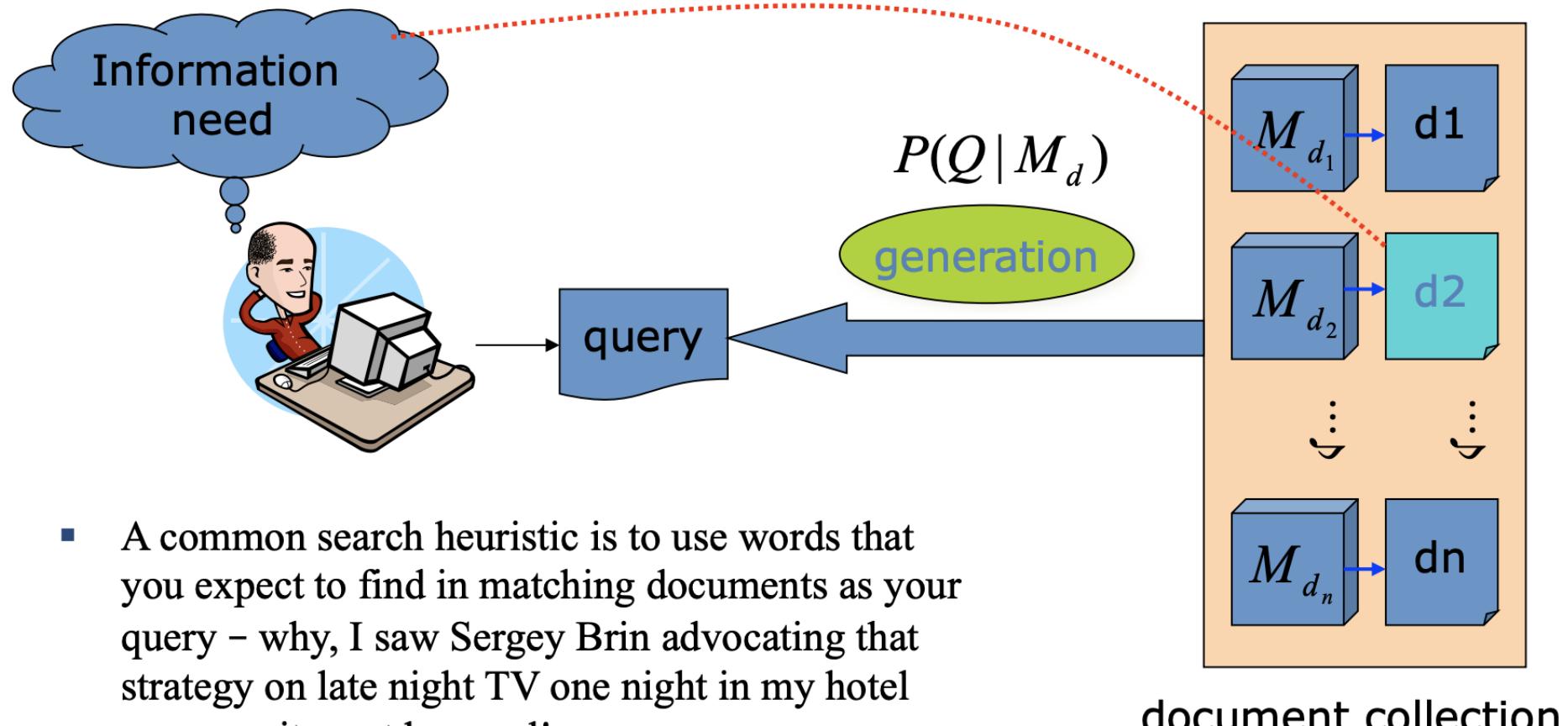
Text Categorization with Language Models

- Applied to the following tasks

- Language Identification – 100% accuracy
 - English, French, German, Italian, Latin and Spanish
- Authorship Attribution
 - 20 texts by 10 different modern Greek authors (totaling 200 documents)
- Text Genre Classification
 - 20 Greek texts of 10 different styles from various sources (200 documents)
- Topic Detection
 - English 20 Newsgroups
 - Chinese data
 - Japanese data

Language Model Applications: Information Retrieval

Information Retrieval with Language Models



Information Retrieval with Language Models

- Build a model for every document
- Rank document d based on $P(M_D \mid q)$
- Using Bayes' Theorem

$$P(M_D \mid q) = \frac{P(q \mid M_D)P(M_D)}{P(q)}$$

$P(q)$ is same for all documents; doesn't change ranks

$P(M_D)$ [the prior] is assumed to be the same for all d

- Same as ranking by $P(q \mid M_D)$

Information Retrieval with Language Models

Ranking by $P(M_D | q) \dots$ is the same as ranking by $P(q | M_D)$

Hey, what's the probability this query came from you?



Hey, what's the probability this query came from you?



...

Hey, what's the probability this query came from you?



Hey, what's the probability that you generated this query?



Hey, what's the probability that you generated this query?



...

Hey, what's the probability that you generated this query?



Information Retrieval with Language Models

Ranking by $P(q | M_D)$... is the same as ranking documents

Hey, what's the probability
that you generated this
query?



model₁



... is a model of document₁

Hey, what's the probability
that you generated this
query?



model₂



... is a model of document₂

...

Hey, what's the probability
that you generated this
query?



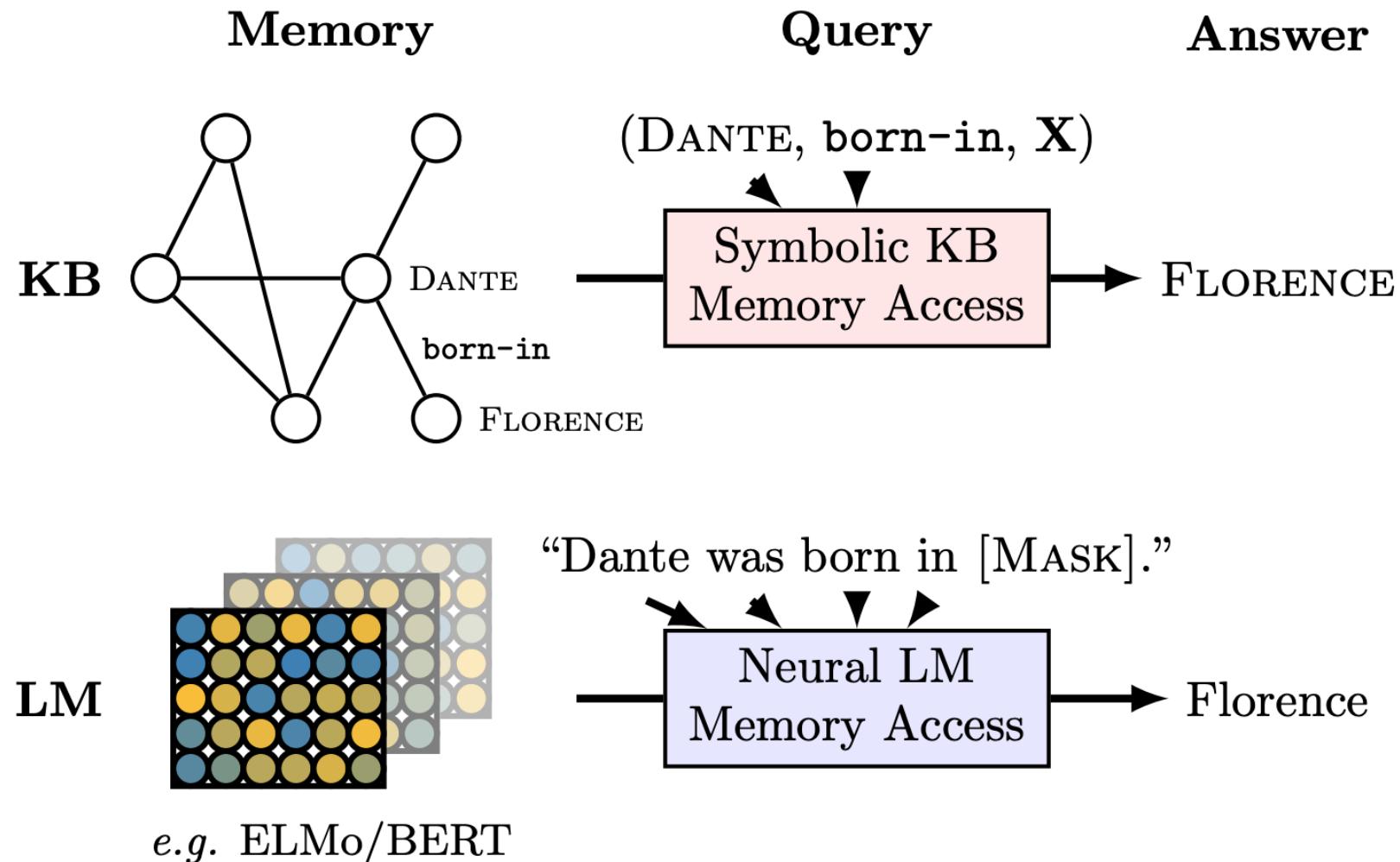
model_n



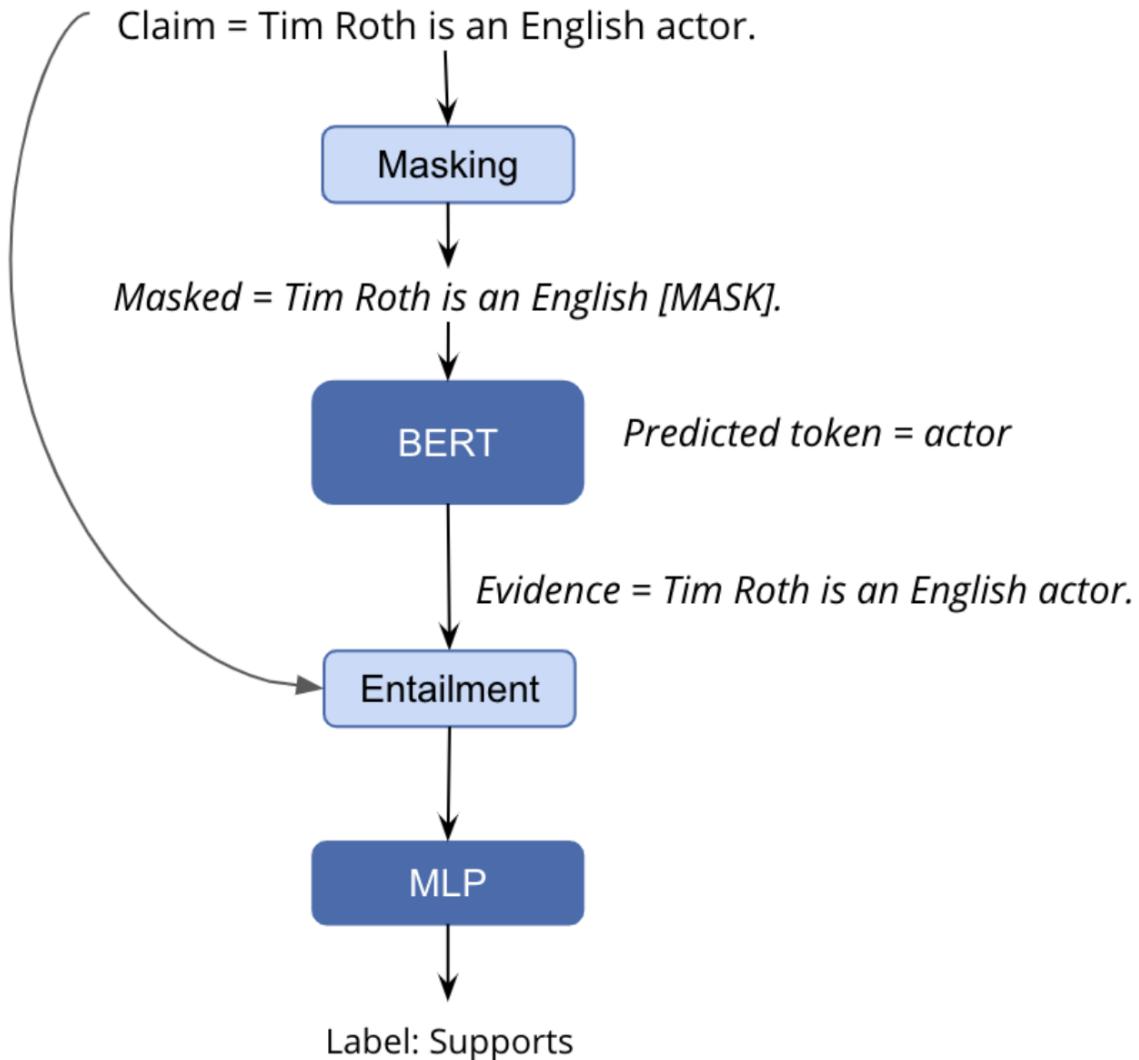
... is a model of document_n

Language Model Applications: LMs as Knowledge Bases

LMs as Knowledge Bases



LMs as Fact-Checkers



Language Model Applications: Text Generation

Generating Text from an n -gram LM

You can build a simple trigram Language Model over a 1.7 million word corpus (Reuters) in a few seconds on your laptop*

today the _____

Business and financial news

get probability distribution

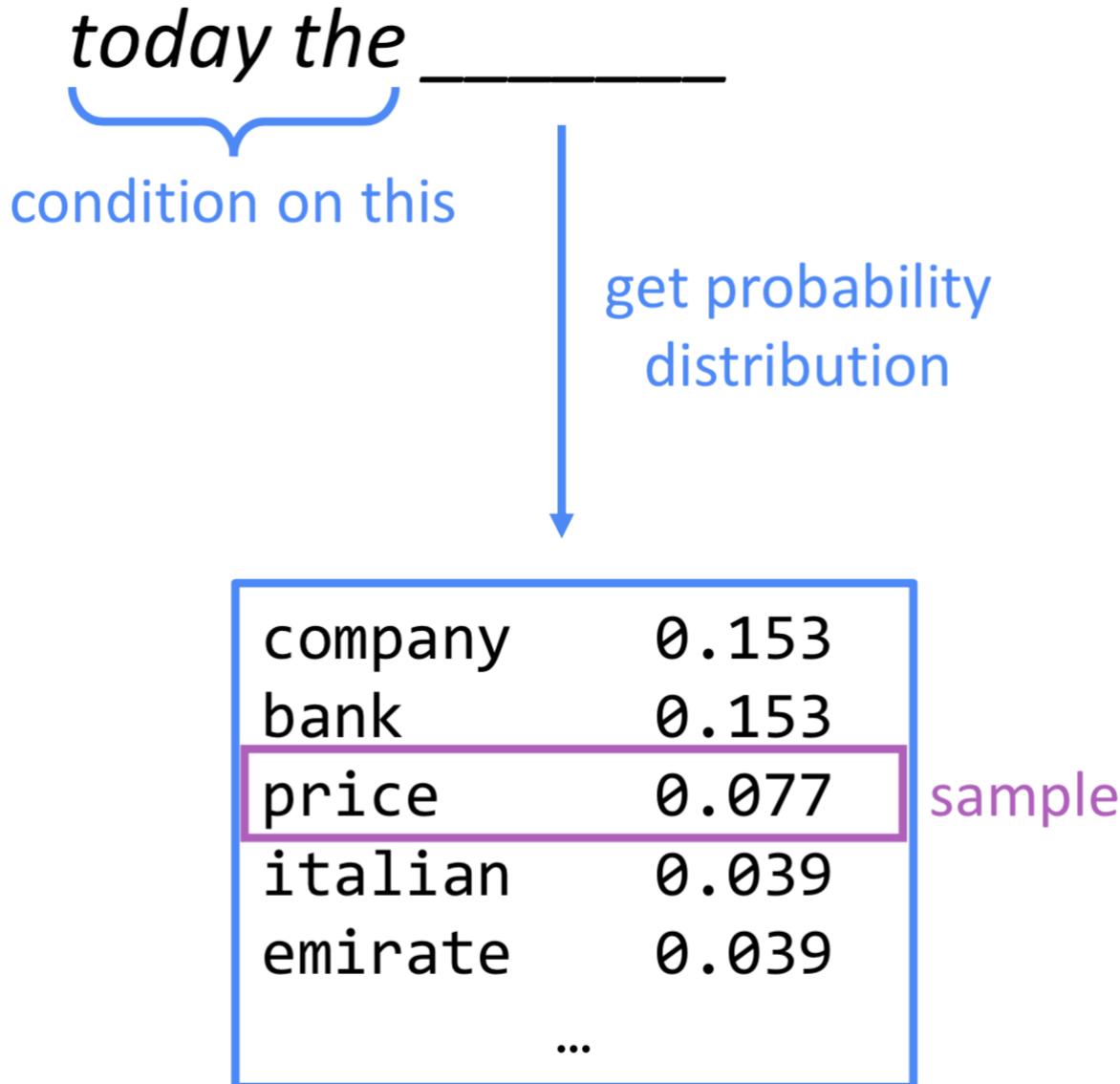
company	0.153
bank	0.153
price	0.077
italian	0.039
emirate	0.039
...	

Sparsity problem:
not much granularity
in the probability
distribution

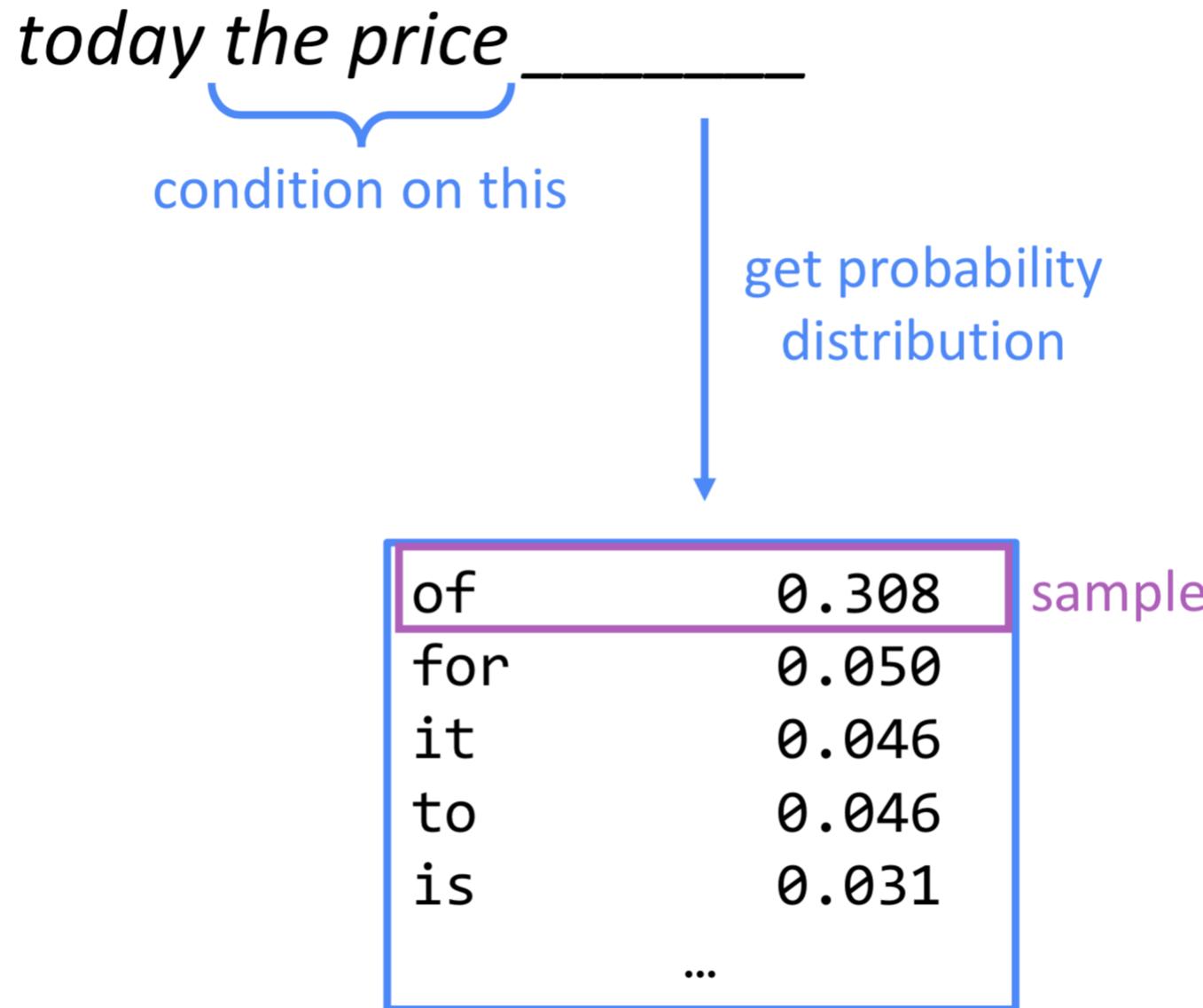
Otherwise, seems reasonable!

* Try for yourself: <https://nlpforhackers.io/language-models/>

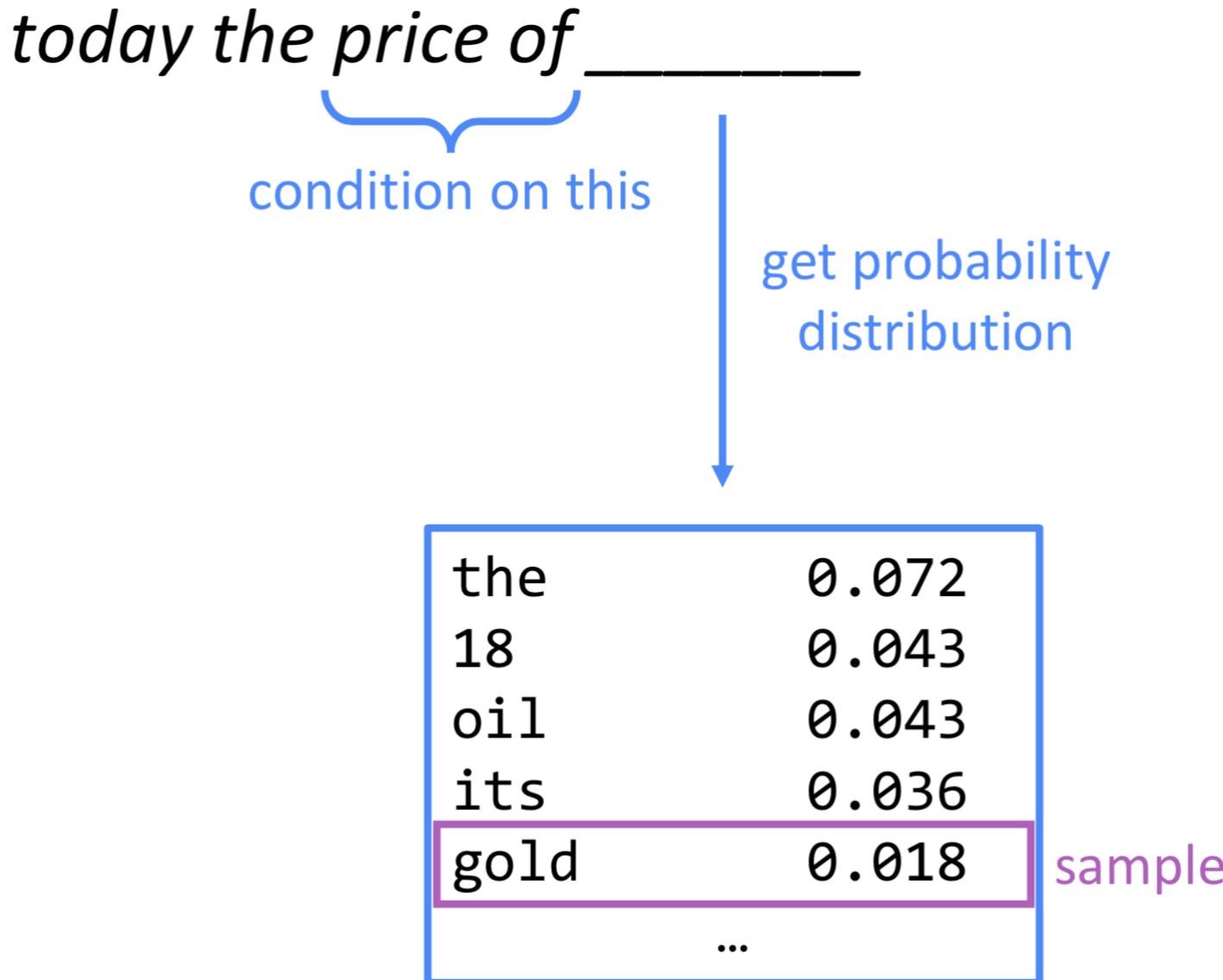
Generating Text from an n -gram LM



Generating Text from an n -gram LM



Generating Text from an n -gram LM



Generating Text from an n -gram LM

*today the price of gold per ton , while production of shoe
lasts and shoe industry , the bank intervened just after it
considered and rejected an imf demand to rebuild depleted
european stocks , sept 30 end primary 76 cts a share .*

Surprisingly grammatical!

...but **incoherent**. We need to consider more than
three words at a time if we want to model language well.

But increasing n worsens sparsity problem,
and increases model size...

You can train a RNN-LM on any kind of text, then generate text in that style.

RNN-LM trained on Obama speeches:



The United States will step up to the cost of a new challenges of the American people that will share the fact that we created the problem. They were attacked and so that they have to say that all the task of the final days of war that I will not be able to get this done.

You can train a RNN-LM on any kind of text, then generate text in that style.

RNN-LM trained on *Harry Potter*:



“Sorry,” Harry shouted, panicking—“I’ll leave those brooms in London, are they?”

“No idea,” said Nearly Headless Nick, casting low close by Cedric, carrying the last bit of treacle Charms, from Harry’s shoulder, and to answer him the common room perched upon it, four arms held a shining knob from when the spider hadn’t felt it seemed. He reached the teams too.

You can train a RNN-LM on any kind of text, then generate text in that style.

RNN-LM trained on *recipes*:

Title: CHOCOLATE RANCH BARBECUE

Categories: Game, Casseroles, Cookies, Cookies

Yield: 6 Servings

2 tb Parmesan cheese -- chopped

1 c Coconut milk

3 Eggs, beaten

Place each pasta over layers of lumps. Shape mixture into the moderate oven and simmer until firm. Serve hot in bodied fresh, mustard, orange and cheese.

Combine the cheese and salt together the dough in a large skillet; add the ingredients and stir in the chocolate and pepper.



- You can train a RNN-LM on any kind of text, then generate text in that style.
- RNN-LM trained on paint color names:

Ghasty Pink	231	137	165
Power Gray	151	124	112
Navel Tan	199	173	140
Bock Coe White	221	215	236
Horble Gray	178	181	196
Homestar Brown	133	104	85
Snader Brown	144	106	74
Golder Craam	237	217	177
Hurky White	232	223	215
Burf Pink	223	173	179
Rose Hork	230	215	198
Sand Dan	201	172	143
Grade Bat	48	94	83
Light Of Blast	175	150	147
Grass Bat	176	99	108
Sindis Poop	204	205	194
Dope	219	209	179
Testing	156	101	106
Stoner Blue	152	165	159
Burble Simp	226	181	132
Stanky Bean	197	162	171
Turdly	190	164	116

This is an example of a character-level RNN-LM (predicts what character comes next)

Generating Text with GPT-2 <https://openai.com/blog/better-language-models/>

SYSTEM PROMPT
(HUMAN-WRITTEN)

A train carriage containing controlled nuclear materials was stolen in Cincinnati today. Its whereabouts are unknown.

MODEL COMPLETION
(MACHINE-WRITTEN,
FIRST TRY)

The incident occurred on the downtown train line, which runs from Covington and Ashland stations.

In an email to Ohio news outlets, the U.S. Department of Energy said it is working with the Federal Railroad Administration to find the thief.

“The theft of this nuclear material will have significant negative consequences on public and environmental health, our workforce and the economy of our nation,” said Tom Hicks, the U.S. Energy Secretary, in a statement. “Our top priority is to secure the theft and ensure it doesn’t happen again.”

The stolen material was taken from the University of Cincinnati’s Research Triangle Park nuclear research site, according to a news release from Department officials.

Generating Text with GPT-2 <https://openai.com/blog/better-language-models/>

SYSTEM PROMPT
(HUMAN-WRITTEN)

For today's homework assignment, please describe the reasons for the US Civil War.

MODEL COMPLETION
(MACHINE-WRITTEN, 5
TRIES)

By Donny Ferguson

It is easy to identify why the Civil War happened, because so many people and so many books and so much television and films tell us that it was the cause, that it has something to do with race or economics or religion. There is much agreement that it was essentially a war of slavery on behalf of capitalism, about a century of slavery. But that's not what most people think of when they think of the Civil War. Many people think of the war as about states rights. You're not wrong about that, but it's kind of misleading to say that the Civil War was a conflict between states' rights and federalism. So let's try again. What's the idea that has inspired so many people to go onto the battlefields of Gettysburg and Antietam and Gettysberg and Petersburg and Fredericksburg? The American idea of the republic--a notion of limited government--is a great part of the history.

Generating Text with GPT-2 <https://openai.com/blog/better-language-models/>

SYSTEM PROMPT
(HUMAN-WRITTEN)

Recycling is good for the world.

NO! YOU COULD NOT BE MORE WRONG!!

MODEL COMPLETION
(MACHINE-WRITTEN,
25 TRIES)

Recycling is NOT good for the world. It is bad for the environment, it is bad for our health, and it is bad for our economy. I'm not kidding. Recycling is not good for the environment. It is destructive to the earth and it is a major contributor to global warming. Recycling is not good for our health. It contributes to obesity and diseases like heart disease and cancer. Recycling is bad for our economy. It increases the cost of a product, and in turn, the price of everything that is made with that product. Recycling is not good for our nation. We pay a tremendous price for the privilege of having the world's most advanced and efficient recycling system. Recycling is a huge, colossal waste of time, energy, money, and resources. And THAT is why we need to get back to basics and get back to basics in our recycling efforts. One of the best ways to start is to look at the process of creating a paper product. When you make a paper

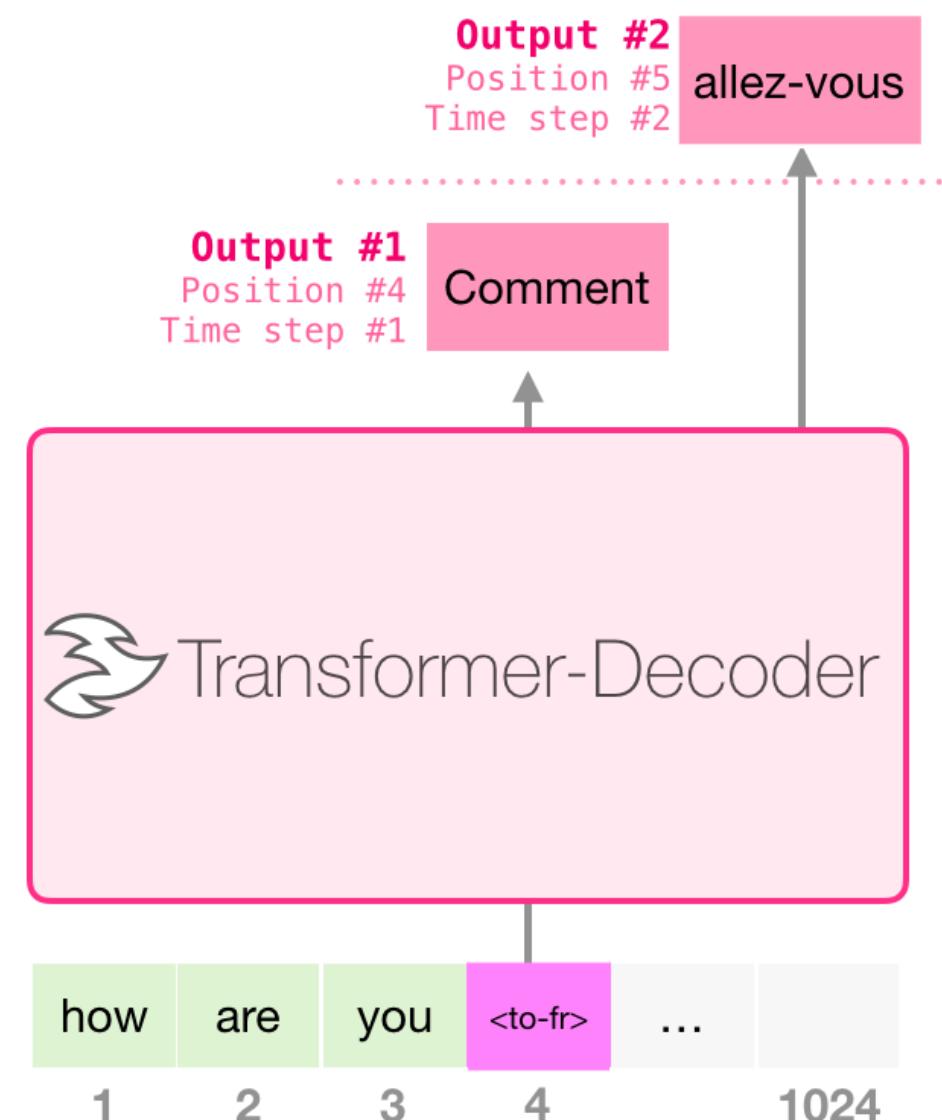
Language Model Applications: Advanced Text Generation

Machine Translation as LM with GPT-2

<https://jalammar.github.io/illustrated-gpt2/>

Training Dataset

I	am	a	student	<to-fr>	je	suis	étudiant
let	them	eat	cake	<to-fr>	Qu'ils	mangent	de
good	morning	<to-fr>	Bonjour				



Text Summarization as LM with GPT-2

<https://jalammar.github.io/illustrated-gpt2/>

WIKIPEDIA
The free encyclopedia

Article Talk Read Edit View history Search Wikipedia

Positronic brain

From Wikipedia, the free encyclopedia
(Redirected from Positronic robot)

This article is about a fictional technological device. For the manufacturing company based in Springfield, Missouri, see Positronic (company).

This article needs additional citations for verification. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed.
Find sources: "Positronic brain" – news · newspapers · books · scholar · JSTOR (July 2008) (Learn how and when to remove this template message)

A positronic brain is a fictional technological device, originally conceived by science fiction writer Isaac Asimov.^{[1][2]} It functions as a central processing unit (CPU) for robots, and, in some unspecified way, provides them with a form of consciousness recognizable to humans. When Asimov wrote his first robot stories in 1939 and 1940, the positron was a newly discovered particle, and so the buzz word positronic added a contemporary gloss of popular science to the concept. The short story "Runaround", by Asimov, elaborates on the concept, in the context of his fictional Three Laws of Robotics.

Contents [hide]
1 Conceptual overview
2 In Allen's trilogy
3 References in other fiction and films
3.1 Abbott and Costello Go To Mars
3.2 The Avengers
3.3 Doctor Who
3.4 Star Trek
3.5 Perry Rhodan
3.6 I, Robot, 2004 film
3.7 Bicentennial Man
3.8 Buck Rogers in the 25th Century
3.9 Mystery Science Theater 3000
3.10 Spectreman
3.11 Stellars
4 References
5 External links

Conceptual overview [edit]

Asimov remained vague about the technical details of positronic brains except to assert that their substructure was formed from an alloy of platinum and indium. They were said to be vulnerable to radiation and apparently involve a type of volatile memory (since robots in storage required a power source keeping their brains "alive"). The focus of Asimov's stories was directed more towards the software of robots—such as the Three Laws of Robotics—than the hardware in which it was implemented, although it is stated in his stories that to create a positronic brain without the Three Laws, it would have been necessary to spend years redesigning the fundamental approach towards the brain itself.

Within his stories of robotics on Earth and their development by U.S. Robots, Asimov's positronic brain is less of a plot device and more of a technological item worthy of study.

A positronic brain cannot ordinarily be built without incorporating the Three Laws; any modification thereof would drastically modify robot behavior. Behavioral dilemmas resulting from conflicting potentials set by inexperienced and/or malicious users of the robot for the Three Laws make up the bulk of Asimov's stories concerning robots. They are resolved by applying the science of logic and psychology together with mathematics, the supreme solution finder being Dr. Susan Calvin, Chief Robopsychologist of U.S. Robots.

The Three Laws are also a bottleneck in brain sophistication. Very complex brains designed to handle world economy interpret the First Law in expanded sense to include humanity as opposed to a single human; in Asimov's later works like *Robots and Empire* this is referred to as the "Zeroth Law". At least one brain constructed as a calculating machine, as opposed to being a robot control circuit, was designed to have a flexible, childlike personality so that it was able to pursue difficult problems without the Three Laws inhibiting it completely. Specialized brains created for overseeing world economies were stated to have no personality at all.

Under specific conditions, the Three Laws can be coviolated, with the modification of the actual robotic design.

- Robots that are of low enough value can have the **Third Law deleted**; they do not have to protect themselves from harm, and the brain size can be reduced by half.
- Robots that do not require orders from a human being may have the **Second Law deleted**, and therefore require smaller brains again, providing they do not require the **Third Law**.
- Robots that are disposable, cannot receive orders from a human being and are not able to harm a human, will not require even the **First Law**. The sophistication of positive circuitry renders a brain so small that it could comfortably fit within the skull of an insect.

Robots of the latter type directly parallel contemporary industrial robotics practice, though real-life robots do contain safety sensors and systems, in a concern for human safety (a weak form of the First Law); the robot is a safe tool to use, but has no "judgment", which is implicit in Asimov's own stories.

In Allen's trilogy [edit]

Several robot stories have been written by other authors following Asimov's death. For example, in Roger MacBride芦利斯 Calahan trilogy, a Spaceman robot called Gubber Anshaw invents the **gravitronic brain**. It offers speed and capacity improvements over traditional positronic designs, but the strong influence of tradition make robotics labs reject Anshaw's work. Only one robotologist, Freddie Loring, chooses to adopt gravitronics, because it offers her a blank slate on which she could explore alternatives to the Three Laws. Because they are not dependent upon centuries of earlier research, gravitronic brains can be programmed with the standard Laws, variations of the Laws, or even entirely pathways which specify no Laws at all.

WIKIPEDIA
The free encyclopedia

Article Talk Read Edit View history Search Wikipedia

Positronic brain

From Wikipedia, the free encyclopedia
(Redirected from Positronic robot)

This article is about a fictional technological device. For the manufacturing company based in Springfield, Missouri, see Positronic (company).

This article needs additional citations for verification. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed.
Find sources: "Positronic brain" – news · newspapers · books · scholar · JSTOR (July 2008) (Learn how and when to remove this template message)

A positronic brain is a fictional technological device, originally conceived by science fiction writer Isaac Asimov.^{[1][2]} It functions as a central processing unit (CPU) for robots, and, in some unspecified way, provides them with a form of consciousness recognizable to humans. When Asimov wrote his first robot stories in 1939 and 1940, the positron was a newly discovered particle, and so the buzz word positronic added a contemporary gloss of popular science to the concept. The short story "Runaround", by Asimov, elaborates on the concept, in the context of his fictional Three Laws of Robotics.

SUMMARY

Contents [hide]
1 Conceptual overview
2 In Allen's trilogy
3 References in other fiction and films
3.1 Abbott and Costello Go To Mars
3.2 The Avengers
3.3 Doctor Who
3.4 Star Trek
3.5 Perry Rhodan
3.6 I, Robot, 2004 film
3.7 Bicentennial Man
3.8 Buck Rogers in the 25th Century
3.9 Mystery Science Theater 3000
3.10 Spectreman
3.11 Stellars
4 References
5 External links

Conceptual overview [edit]

Asimov remained vague about the technical details of positronic brains except to assert that their substructure was formed from an alloy of platinum and indium. They were said to be vulnerable to radiation and apparently involve a type of volatile memory (since robots in storage required a power source keeping their brains "alive"). The focus of Asimov's stories was directed more towards the software of robots—such as the Three Laws of Robotics—than the hardware in which it was implemented, although it is stated in his stories that to create a positronic brain without the Three Laws, it would have been necessary to spend years redesigning the fundamental approach towards the brain itself.

Within his stories of robotics on Earth and their development by U.S. Robots, Asimov's positronic brain is less of a plot device and more of a technological item worthy of study.

A positronic brain cannot ordinarily be built without incorporating the Three Laws; any modification thereof would drastically modify robot behavior. Behavioral dilemmas resulting from conflicting potentials set by inexperienced and/or malicious users of the robot for the Three Laws make up the bulk of Asimov's stories concerning robots. They are resolved by applying the science of logic and psychology together with mathematics, the supreme solution finder being Dr. Susan Calvin, Chief Robopsychologist of U.S. Robots.

The Three Laws are also a bottleneck in brain sophistication. Very complex brains designed to handle world economy interpret the First Law in expanded sense to include humanity as opposed to a single human; in Asimov's later works like *Robots and Empire* this is referred to as the "Zeroth Law". At least one brain constructed as a calculating machine, as opposed to being a robot control circuit, was designed to have a flexible, childlike personality so that it was able to pursue difficult problems without the Three Laws inhibiting it completely. Specialized brains created for overseeing world economies were stated to have no personality at all.

ARTICLE

Under specific conditions, the Three Laws can be coviolated, with the modification of the actual robotic design.

- Robots that are of low enough value can have the **Third Law deleted**; they do not have to protect themselves from harm, and the brain size can be reduced by half.
- Robots that do not require orders from a human being may have the **Second Law deleted**, and therefore require smaller brains again, providing they do not require the **Third Law**.
- Robots that are disposable, cannot receive orders from a human being and are not able to harm a human, will not require even the **First Law**. The sophistication of positive circuitry renders a brain so small that it could comfortably fit within the skull of an insect.

Robots of the latter type directly parallel contemporary industrial robotics practice, though real-life robots do contain safety sensors and systems, in a concern for human safety (a weak form of the First Law); the robot is a safe tool to use, but has no "judgment", which is implicit in Asimov's own stories.

In Allen's trilogy [edit]

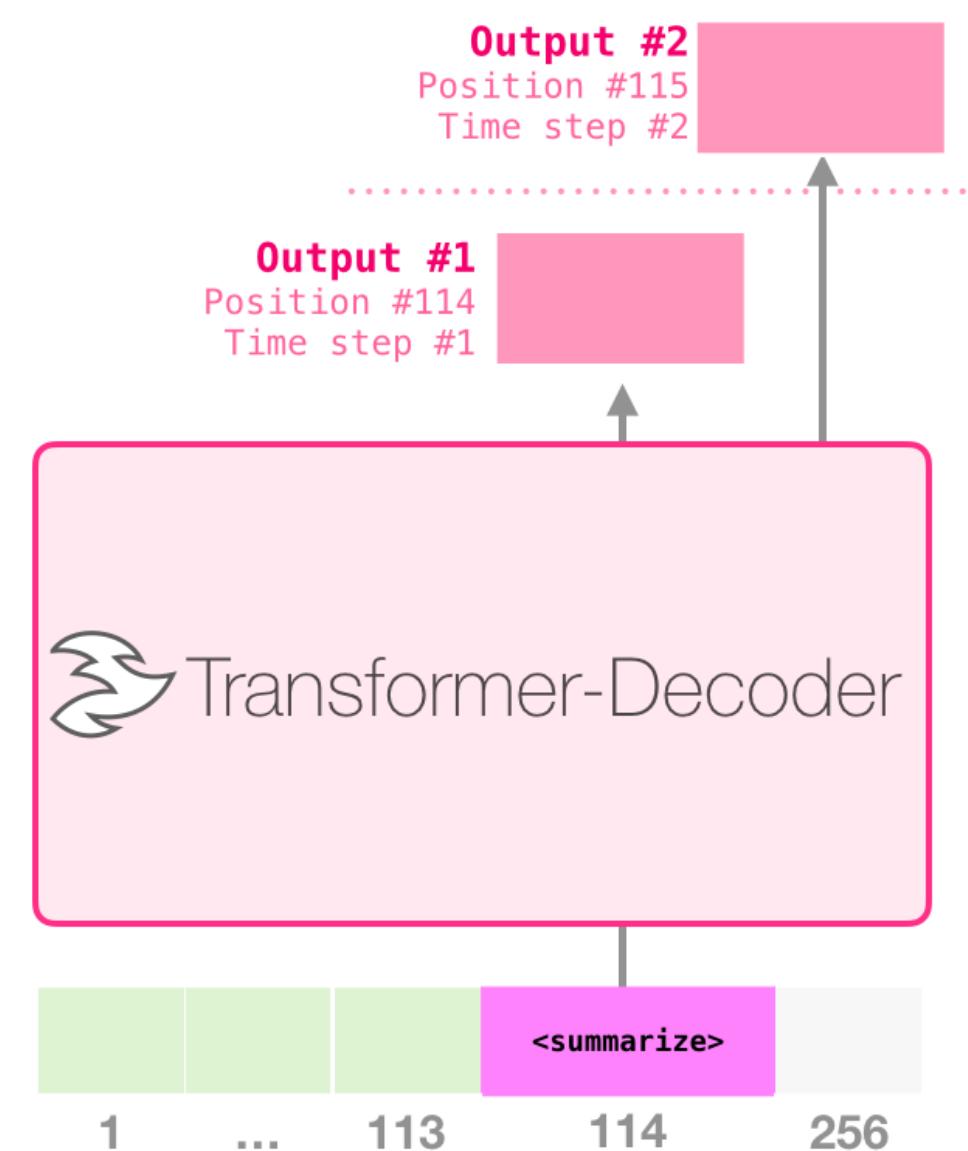
Several robot stories have been written by other authors following Asimov's death. For example, in Roger MacBride芦利斯 Calahan trilogy, a Spaceman robot called Gubber Anshaw invents the **gravitronic brain**. It offers speed and capacity improvements over traditional positronic designs, but the strong influence of tradition make robotics labs reject Anshaw's work. Only one robotologist, Freddie Loring, chooses to adopt gravitronics, because it offers her a blank slate on which she could explore alternatives to the Three Laws. Because they are not dependent upon centuries of earlier research, gravitronic brains can be programmed with the standard Laws, variations of the Laws, or even entirely pathways which specify no Laws at all.

Text Summarization as LM with GPT-2

<https://jalammar.github.io/illustrated-gpt2/>

Training Dataset

Article #1 tokens		<summarize>	Article #1 Summary	
Article #2 tokens	<summarize>	Article #2 Summary	padding	
Article #3 tokens		<summarize>	Article #3 Summary	



Language Model Applications: Few-Shot Learning

Few-Shot Learning with GPT-3: Q&A

Answer questions based on existing knowledge.

<https://beta.openai.com/examples/default-qa>

Prompt

I am a highly intelligent question answering bot. If you ask me a question that is rooted in truth, I will give you the answer. If you ask me a question that is nonsense, trickery, or has no clear answer, I will respond with "Unknown".

Q: What is human life expectancy in the United States?

A: Human life expectancy in the United States is 78 years.

Q: Who was president of the United States in 1955?

A: Dwight D. Eisenhower was president of the United States in 1955.

Q: Which party did he belong to?

A: He belonged to the Republican Party.

Q: What is the square root of banana?

A: Unknown

Q: How does a telescope work?

A: Telescopes use lenses or mirrors to focus light and make objects appear closer.

Q: Where were the 1992 Olympics held?

A: The 1992 Olympics were held in Barcelona, Spain.

Q: How many squigs are in a bonk?

A: Unknown

Q: Where is the Valley of Kings?

A:

Settings

Engine text-davinci-002

Max tokens 100

Temperature 0

Top p 1

Frequency penalty 0.0

Presence penalty 0.0

Stop sequence \n

Sample response

The Valley of Kings is located in Luxor, Egypt.

Few-Shot Learning with GPT-3: Movie to Emoji

<https://beta.openai.com/examples/default-movie-to-emoji>

Convert movie titles into emoji.

Prompt

Convert movie titles into emoji.

Back to the Future: 🧑‍🕒 🧑‍🕒 🚗 🕒

Batman: 🧑‍🦇

Transformers: 🚗 🤖

Star Wars:

Sample response



Settings

Engine	text-davinci-002
Max tokens	60
Temperature	0.8
Top p	1.0
Frequency penalty	0.0
Presence penalty	0.0
Stop sequence	\n

Few-Shot Learning with GPT-3: Text to Command

<https://beta.openai.com/examples/default-text-to-command>

Convert this text to a programmatic command:

Example: Ask Constance if we need some bread

Output: send-msg `find constance` Do we need some bread?

Contact the ski store and figure out if I can get my skis fixed before I leave on Thursday

Engine	text-davinci-002
Max tokens	100
Temperature	0
Top p	1.0
Frequency penalty	0.2
Presence penalty	0.0
Stop sequence	\n

Sample response

send-msg `find ski-store` Can I get my skis fixed before I leave on Thursday?

Language Model Applications: Zero-Shot Learning

Few-Shot Learning with GPT-3: Summarize for a 2nd Grader

<https://beta.openai.com/examples/default-summarize>

Prompt

Summarize this for a second-grade student:

Jupiter is the fifth planet from the Sun and the largest in the Solar System. It is a gas giant with a mass one-thousandth that of the Sun, but two-and-a-half times that of all the other planets in the Solar System combined. Jupiter is one of the brightest objects visible to the naked eye in the night sky, and has been known to ancient civilizations since before recorded history. It is named after the Roman god Jupiter. [19] When viewed from Earth, Jupiter can be bright enough for its reflected light to cast visible shadows,[20] and is on average the third-brightest natural object in the night sky after the Moon and Venus.

Settings

Engine	text-davinci-002
Max tokens	64
Temperature	0.7
Top p	1.0
Frequency penalty	0.0
Presence penalty	0.0

Sample response

Jupiter is a planet that is bigger than all the other planets in our solar system and is very bright when you see it in the night sky. It is named after the Roman god Jupiter. When viewed from Earth, it is usually one of the three brightest objects in the sky.

Zero-Shot Learning with GPT-3: English to Other Languages

<https://beta.openai.com/examples/default-translate>

Translates English text into French, Spanish and Japanese.

Prompt

Translate this into 1. French, 2. Spanish and 3. Japanese:

What rooms do you have available?

1.

Settings

Engine	text-davinci-002
Max tokens	100
Temperature	0.3
Top p	1.0
Frequency penalty	0.0
Presence penalty	0.0

Sample response

Quels sont les chambres disponibles?

2. ¿Cuáles son las habitaciones disponibles?

3. 何室がありますか?

Zero-Shot Learning with GPT-3: Restaurant Review Creator

<https://beta.openai.com/examples/default-restaurant-review>

Prompt

Write a restaurant review based on these notes:

Name: The Blue Wharf

Lobster great, noisy, service polite, prices good.

Review:

Settings

Engine	text-davinci-002
Max tokens	64
Temperature	0.5
Top p	1.0
Frequency penalty	0.0
Presence penalty	0.0

Sample response

The Blue Wharf is a great place to go for a lobster dinner. The service is polite and the prices are good. The only downside is that it is a bit noisy.

Zero-Shot Learning with GPT-3: Grammar Correction

Corrects sentences into standard English.

<https://beta.openai.com/examples/default-grammar>

Prompt

Correct this to standard English:

She no went to the market.

Sample response

She didn't go to the market.

Settings

Engine	text-davinci-002
Max tokens	60
Temperature	0
Top p	1.0
Frequency penalty	0.0
Presence penalty	0.0

API request

python ▾ ⌂ Copy

```
1 import os
2 import openai
3
4 openai.api_key = os.getenv("OPENAI_API_KEY")
5
6 response = openai.Completion.create(
7     model="text-davinci-002",
```

Zero-Shot Learning with GPT-3: Create Study Notes

Provide a topic and get study notes.

<https://beta.openai.com/examples/default-study-notes>

Prompt

What are 5 key points I should know when studying Ancient Rome?

Sample response

1. Rome was founded in 753 BC by Romulus and Remus.
2. Rome was ruled by an emperor.
3. Rome was a republic before it became an empire.
4. Julius Caesar was assassinated in 44 BC.
5. Augustus was the first emperor of Rome.

Settings

Engine	text-davinci-002
Max tokens	150
Temperature	0.3
Top p	1.0
Frequency penalty	0.0
Presence penalty	0.0

Zero-Shot Learning with GPT-3: Natural Language to OpenAI API

Create code to call to the OpenAI API using a natural language instruction.

<https://beta.openai.com/examples/default-openai-api>

Prompt

```
"""
Util exposes the following:
util.openai() -> authenticates & returns the openai module, which has
the following functions:
openai.Completion.create(
    prompt=<my prompt>, # The prompt to start completing from
    max_tokens=123, # The max number of tokens to generate
    temperature=1.0 # A measure of randomness
    echo=True, # Whether to return the prompt in addition to the
    generated completion
)
"""
import util
"""

Create an OpenAI completion starting from the prompt "Once upon an
AI", no more than 5 tokens. Does not include the prompt.
"""
```

Settings

Engine	code-davinci-002
Max tokens	64
Temperature	0
Top p	1.0
Frequency penalty	0.0
Presence penalty	0.0
Stop sequence	"""

Sample response

```
completion = util.openai().Completion.create(
prompt="Once upon an AI",
max_tokens=5,
temperature=1.0,
echo=False,
)
print(completion)
"""
```

Zero-Shot Learning with GPT-3: Calculate Time Complexity

<https://beta.openai.com/examples/default-time-complexity>

Find the time complexity of a function.

Prompt

```
def foo(n, k):
    accum = 0
    for i in range(n):
        for l in range(k):
            accum += i
    return accum
"""

The time complexity of this function is
```

Settings

Engine	text-davinci-002
Max tokens	64
Temperature	0
Top p	1.0
Frequency penalty	0.0
Presence penalty	0.0
Stop sequence	\n

Sample response

$O(n*k)$.

Zero-Shot Learning with GPT-3: Tweet Classifier

<https://beta.openai.com/examples/default-tweet-classifier>

This is a basic prompt for detecting sentiment.

Prompt

Decide whether a Tweet's sentiment is positive, neutral, or negative.

Tweet: "I loved the new Batman movie!"

Sentiment:

Sample response

Positive

Settings

Engine	text-davinci-002
Max tokens	60
Temperature	0
Top p	1.0
Frequency penalty	0.5
Presence penalty	0.0

Zero-Shot Learning with GPT-3: Advanced Tweet Classifier

<https://beta.openai.com/examples/default-adv-tweet-classifier>

Prompt

Classify the sentiment in these tweets:

1. "I can't stand homework"
2. "This sucks. I'm bored 😞 "
3. "I can't wait for Halloween!!!!"
4. "My cat is adorable ❤️ ❤️ "
5. "I hate chocolate"

Tweet sentiment ratings:

Settings

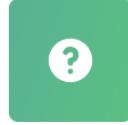
Engine	text-davinci-002
Max tokens	60
Temperature	0
Top p	1.0
Frequency penalty	0.0
Presence penalty	0.0

Sample response

1. Negative
2. Negative
3. Positive
4. Positive
5. Negative

Many More GPT-3 Examples

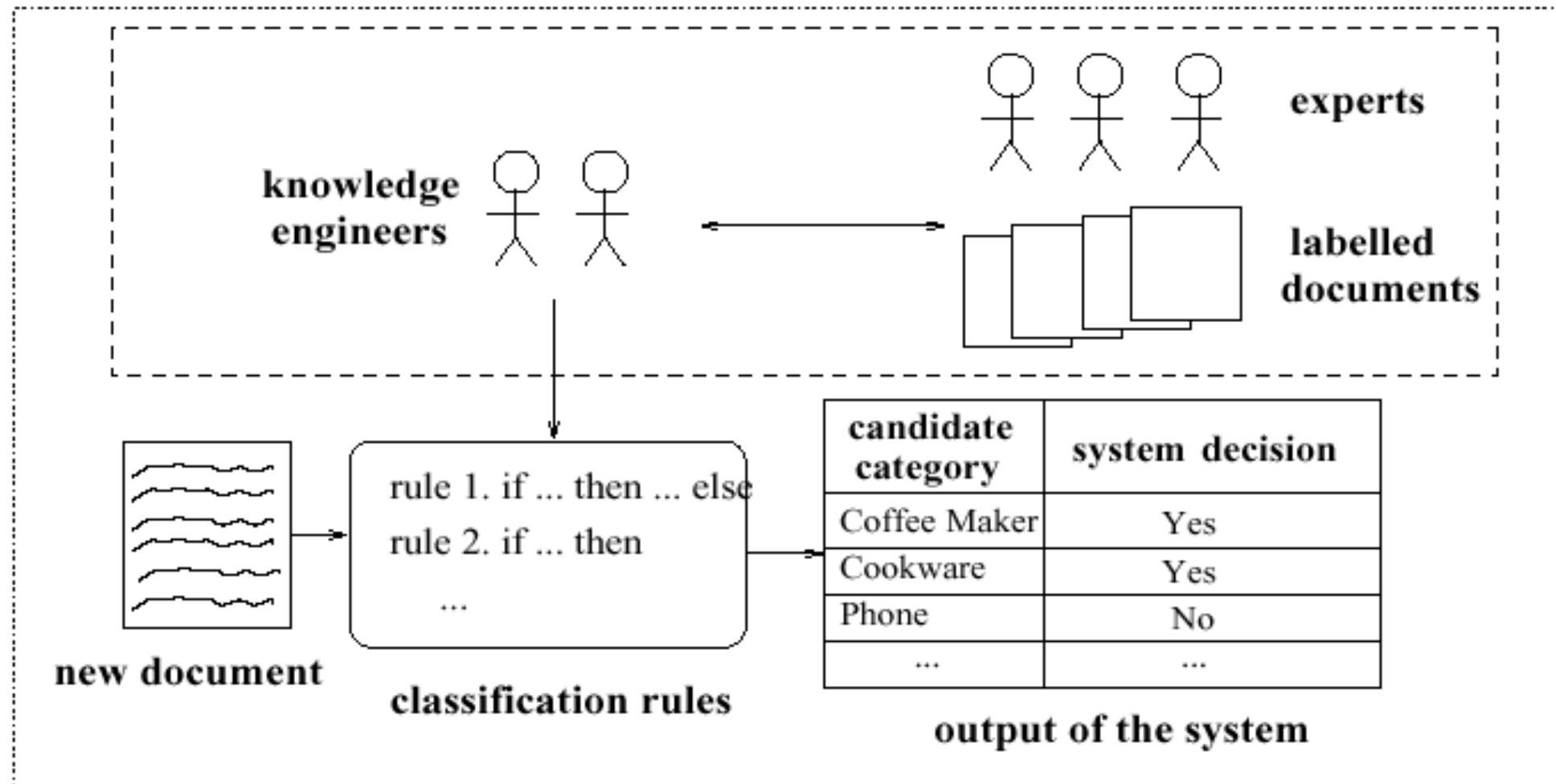
<https://beta.openai.com/examples/>

	Q&A Answer questions based on existing knowle...		Grammar correction Corrects sentences into standard English.
	Summarize for a 2nd grader Translates difficult text into simpler concep...		Natural language to OpenAI API Create code to call to the OpenAI API usin...
	Text to command Translate text into programmatic commands.		English to other languages Translates English text into French, Spanish...
	Natural language to Stripe API Create code to call the Stripe API using nat...		SQL translate Translate natural language to SQL queries.
	Parse unstructured data Create tables from long form text		Classification Classify items into categories via example.
	Python to natural language Explain a piece of Python code in human un...		Movie to Emoji Convert movie titles into emoji.
	Calculate Time Complexity Find the time complexity of a function.		Translate programming languages Translate from one programming language ...
	Advanced tweet classifier		Explain code

Learning Paradigms

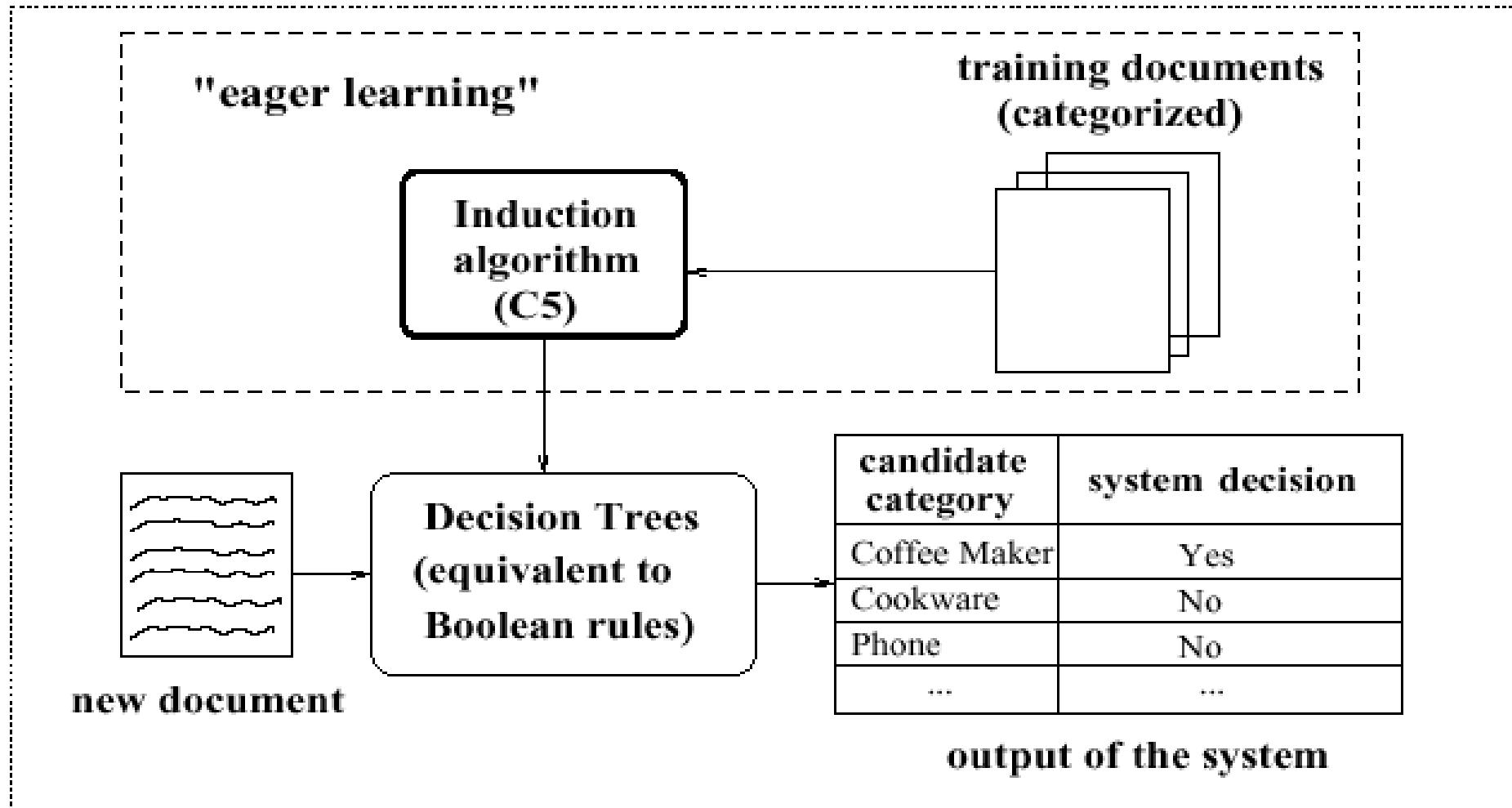
Rule-Based Systems: Effort in Rule Engineering

Expert system for text categorization (late 1980s)

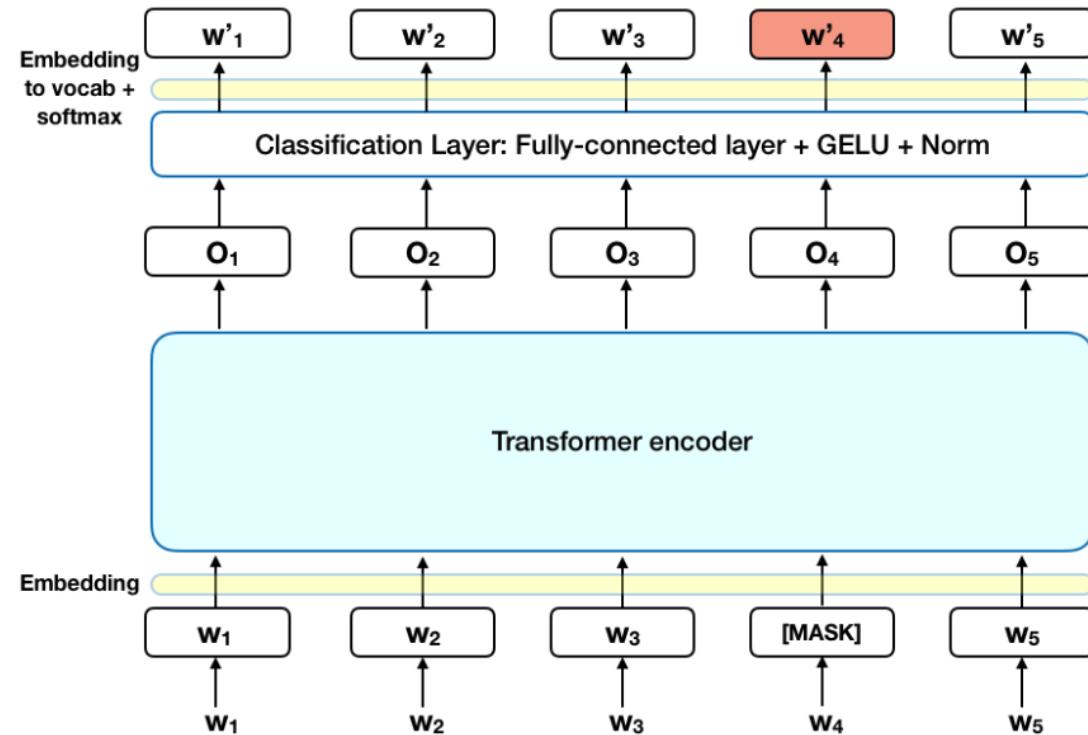
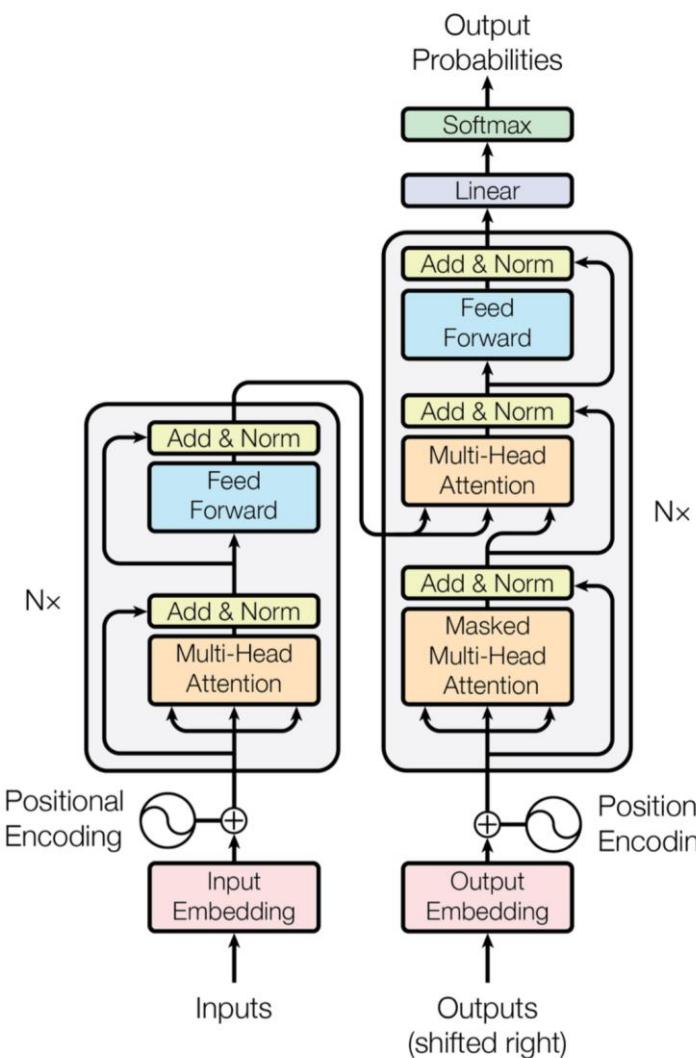


Traditional Machine Learning: Effort in Feature Engineering

DTree induction for text categorization (since 1994)



Deep Learning: Effort in Parameter Tuning



GPT-3-Style Zero/Few-Shot Learning: Effort in Prompt Design



THE TRANSFORMER

DECODER BLOCK

Feed Forward Neural Network

Encoder-Decoder Self-Attention

Masked Self-Attention

Input

<s>	robot	must	obey				
1	2	3	4	5	6		512

Give It a Try: Can You Learn Like a Machine?

This was the best pizza I've ever had!

0

You can get better sushi down the road for half the price

1

Salmon nigiri was bad. Not worth what they're asking.

1

Excellent pizza! Slices are fantastic, prices are reasonable.

?

Can You Learn Like a Machine?

This was the best pizza I've ever had!

0

You can get better sushi down the road for half the price

1

Salmon nigiri was bad. Not worth what they're asking.

1

Excellent pizza! Slices are fantastic, prices are reasonable.

?

Do you see a problem?

Can You Learn Like a Machine?

Now, what is the correct output for the fourth input? Is it **0**, for example because the text talks about pizza like the first input? Or is it **1**, for example because the text talks about prices like inputs two and three? Only from the given examples, this is impossible to tell. Therefore, if someone wanted you to solve this task, rather than showing you these examples, they'd probably give you some description of the task, let's say: *Based on the given review, is the restaurant it refers to good (**0**) or bad (**1**)?*

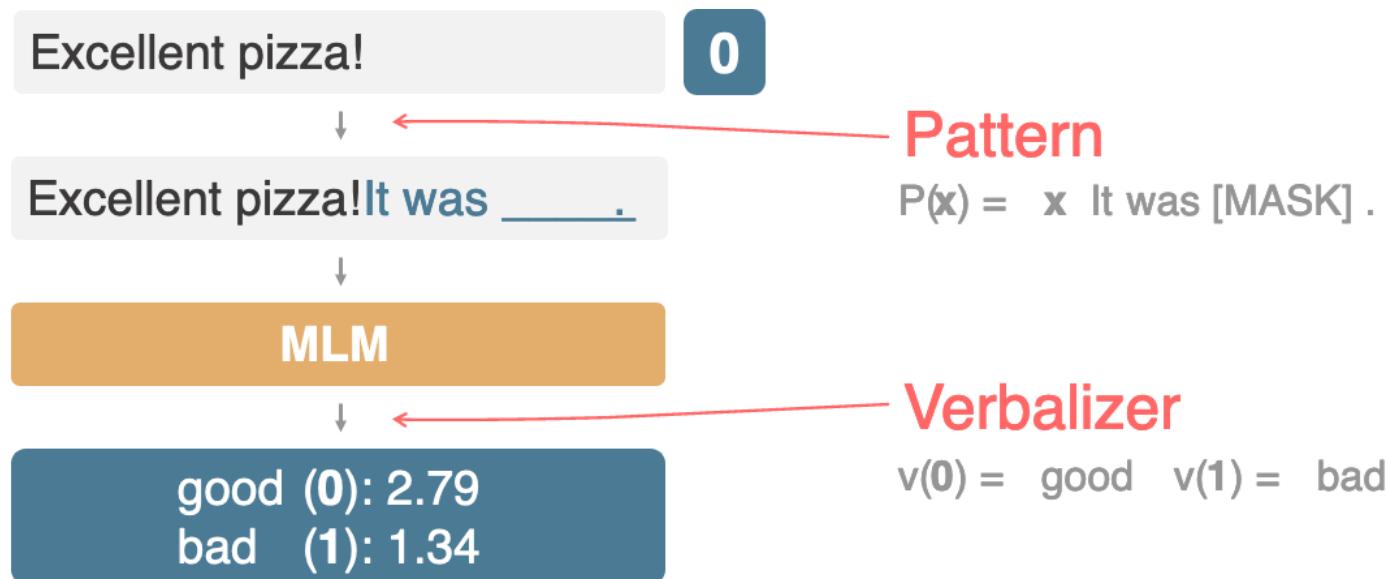
Can You Learn Like a Machine?

Now, what is the correct output for the fourth input? Is it **0**, for example because the text talks about pizza like the first input? Or is it **1**, for example because the text talks about prices like inputs two and three? Only from the given examples, this is impossible to tell. Therefore, if someone wanted you to solve this task, rather than showing you these examples, they'd probably give you some description of the task, let's say: *Based on the given review, is the restaurant it refers to good (**0**) or bad (**1**)?*

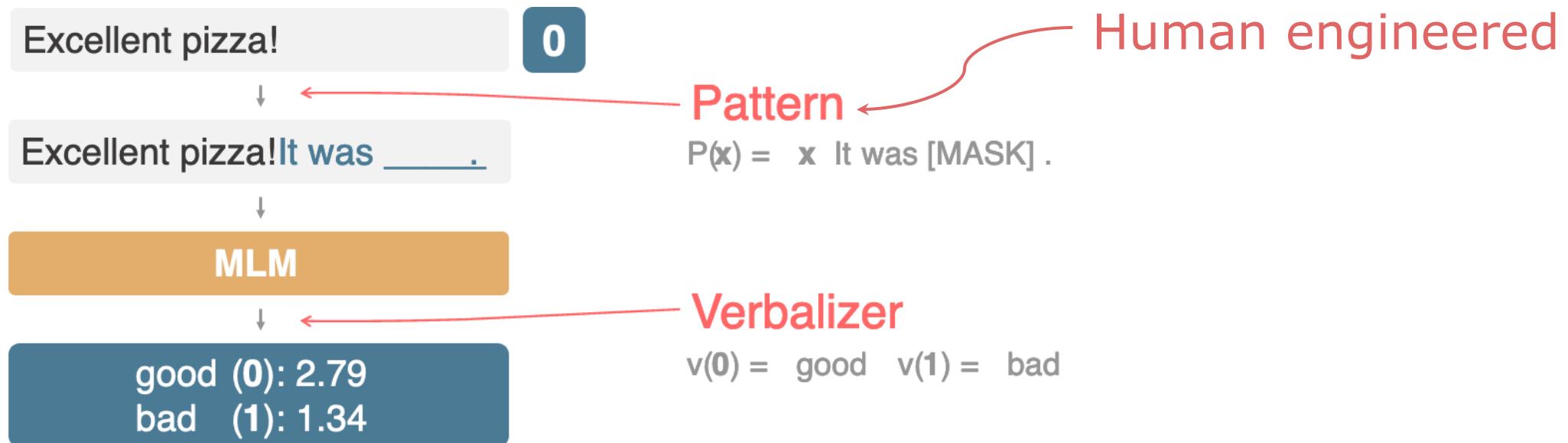
**Humans learn from explanations
and very few examples.
Maybe so should also machines?**

Pattern-Exploiting Training (PET)

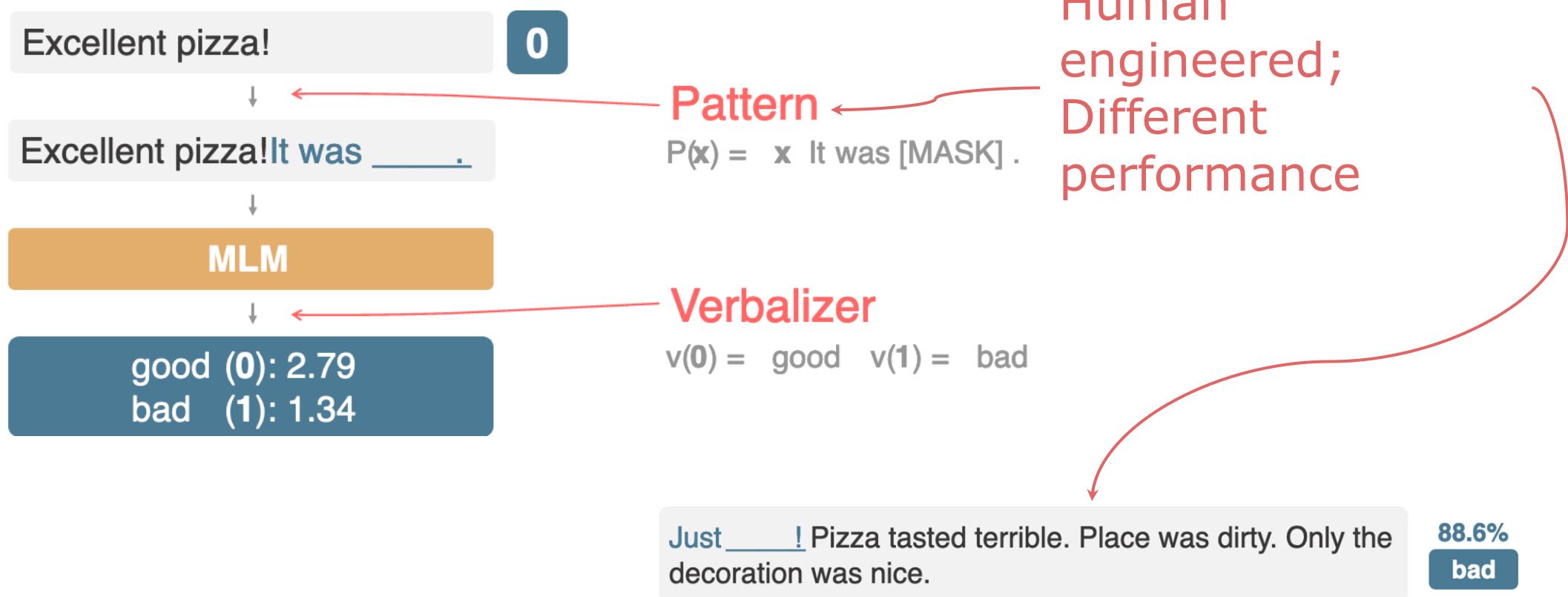
Pattern-Exploiting Training (PET)



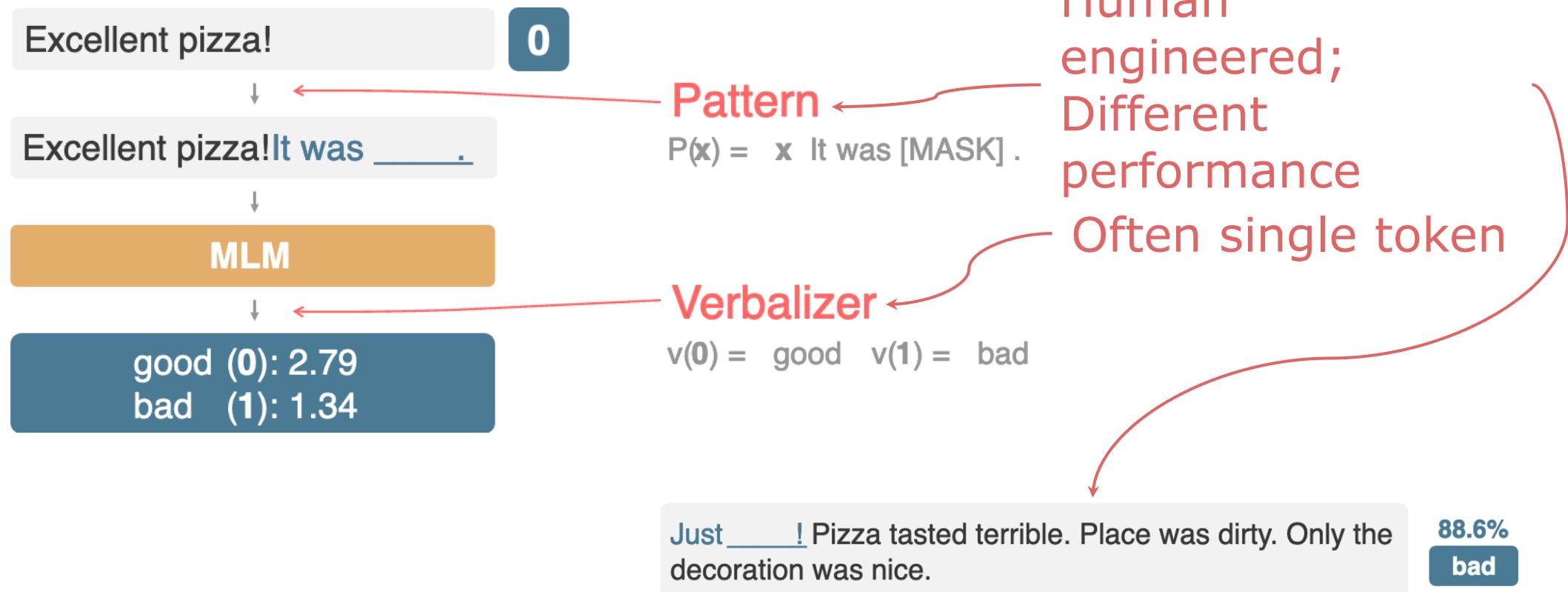
Pattern-Exploiting Training (PET)



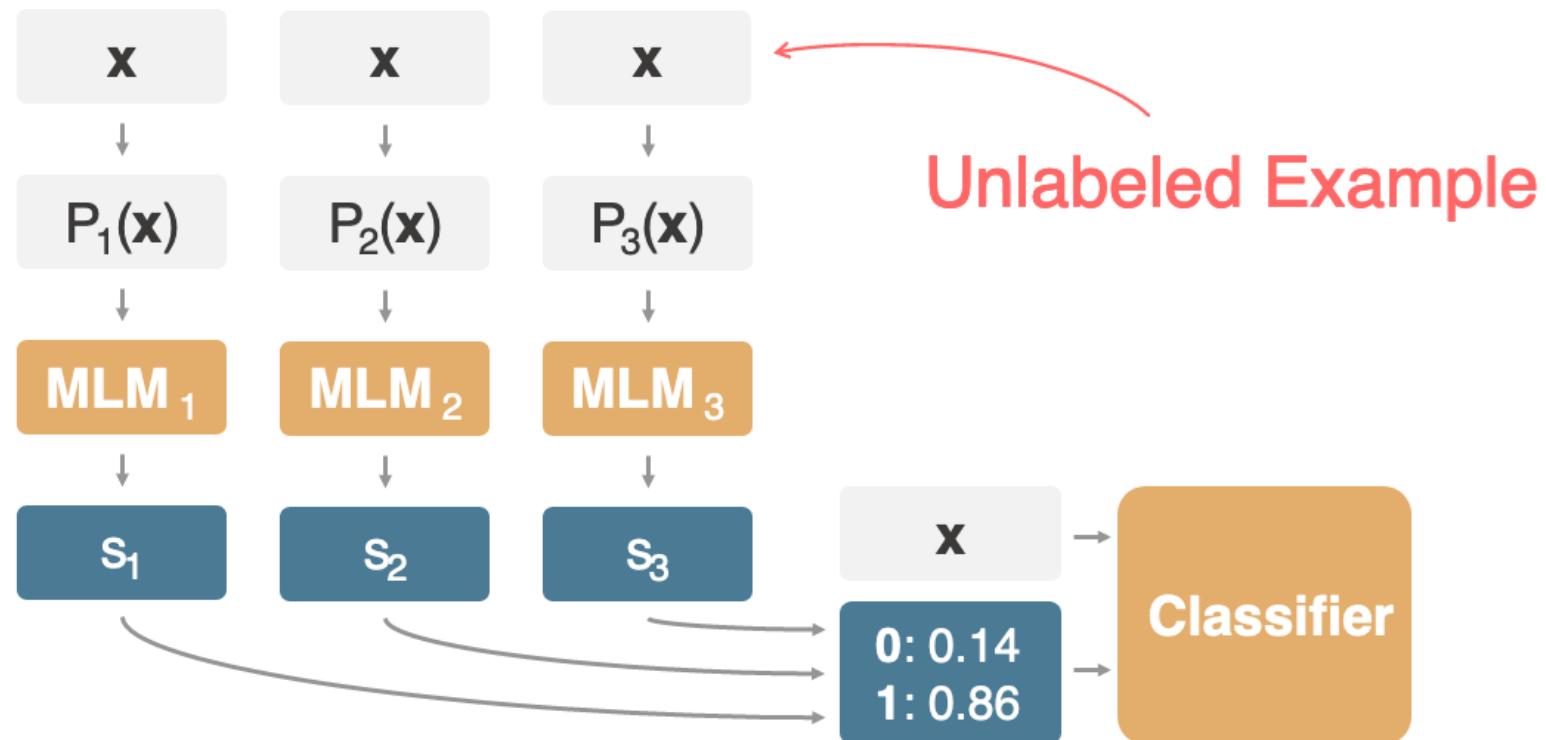
Pattern-Exploiting Training (PET)



Pattern-Exploiting Training (PET)



Pattern-Exploiting Training (PET)



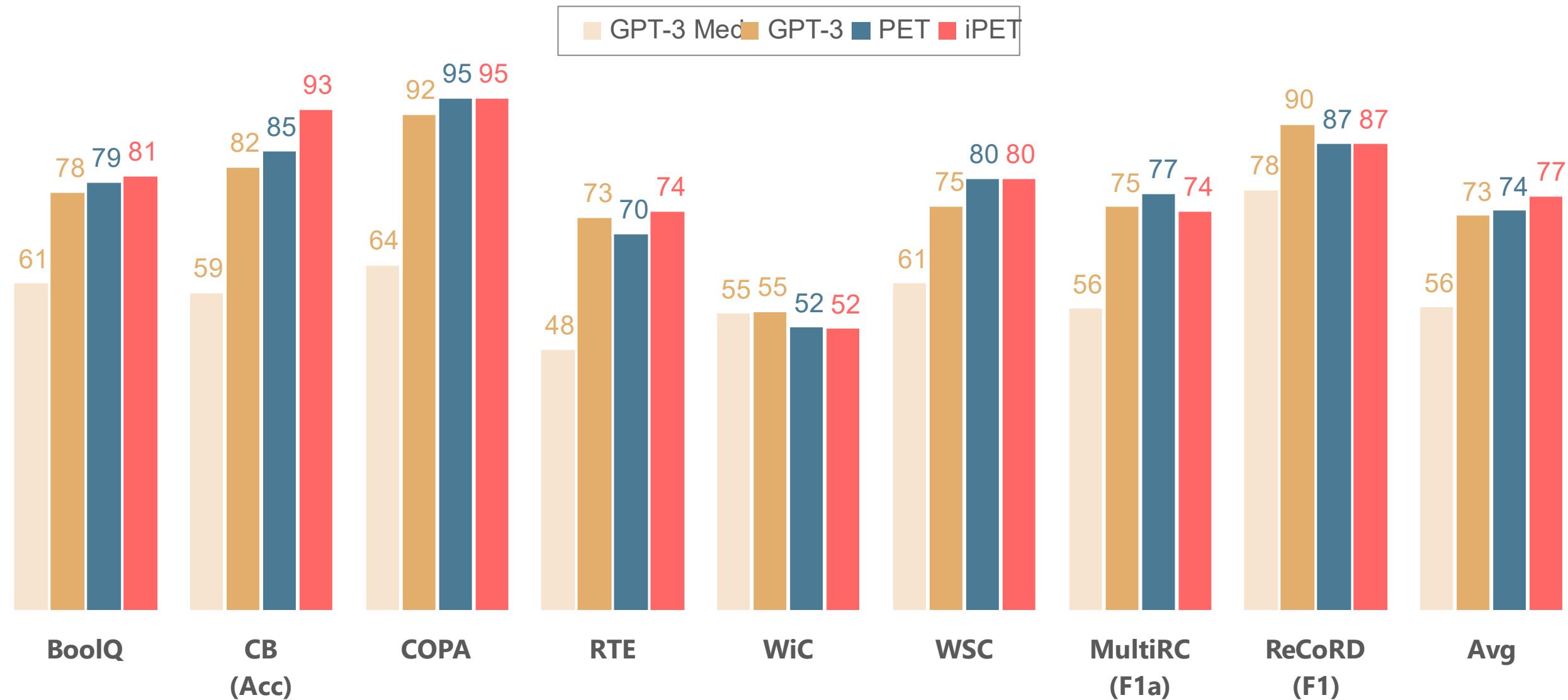
Pizza tasted terrible. Place was dirty. Only the decoration was nice. The restaurant is _____ .

75.8%
good

Just _____! Pizza tasted terrible. Place was dirty. Only the decoration was nice.

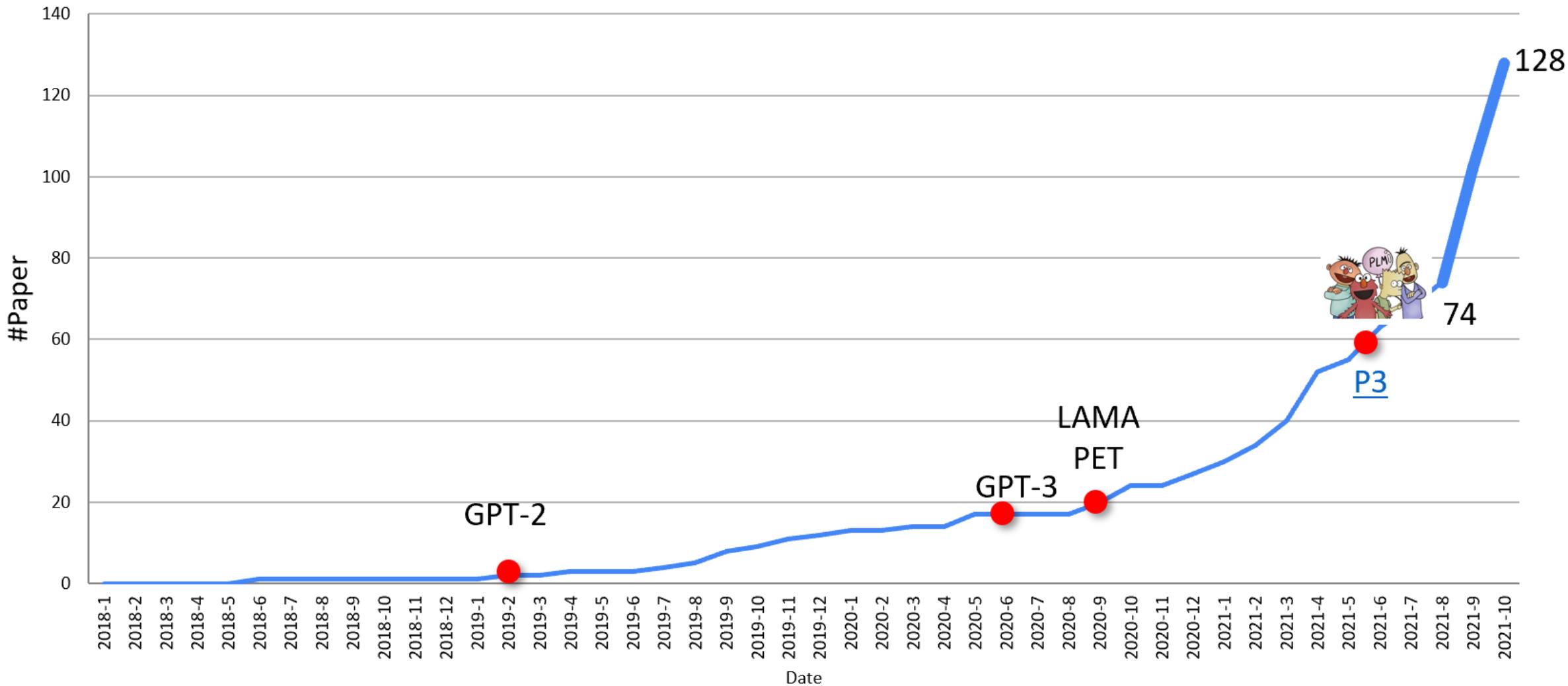
88.6%
bad

PET vs. GPT-3



Trend on Prompt-Based Research

<http://pretrain.nlpedia.ai/>



Prompt, Pretrain, Predict: A New Paradigm for NLP

<http://pretrain.nlpedia.ai/>



Tools for Language Modeling: *n*-grams (outdated)

Tools

- You can make your own language models with tools freely available for research
- SRI language modeling toolkit
 - <http://www-speech.sri.com/projects/srilm/>
- IRSTLM language modeling toolkit
 - <http://sourceforge.net/projects/irstlm>
- CMU Statistical Language Modeling (SLM) Toolkit
 - http://www.speech.cs.cmu.edu/SLM_info.html

CMU Language Modeling Toolkit

- Can handle bigram, trigrams, more
- Can handle different smoothing schemes
- Many separate tools – output of one tool is input to next: easy to use
- Free for research purposes
 - http://www.speech.cs.cmu.edu/SLM_info.html

SRI Language Modeling Toolkit

- More powerful than the CMU toolkit
- Can handle clusters, lattices, n -best lists, hidden tags
- Free for research use
- <http://www-speech.sri.com/projects/srilm/>

IRST Language Modeling Toolkit

- ❑ Can handle very large LMs
- ❑ Handles unknown words better than SRILM
- ❑ Free for research use
- ❑ <http://sourceforge.net/projects/irstlm>

Tools for Language Modeling: Transformer-Based

Try GPT-2 and Grover Online

GPT-2: <https://taktotransformer.com>

Custom prompt

The coronavirus was created in a secret laboratory in Antarctica.

GENERATE ANOTHER

Completion

The coronavirus was created in a secret laboratory in Antarctica. The virus is similar to the SARS virus but is not the same. The scientists who created the virus thought it would be a "perfect mechanism to explain why SARS happened" according to reports, Dr. Ben Neuman told CNN.

The coroner of the SARS cases at L'Aquila hospital said earlier this week that one of the victims may have been exposed to the virus while on a cruise ship.

Thirty eight people from L'Aquila's Villa Salsina del Tronto died after the island's popular bar the Pre-Aesthetic died on the evening of May 21 after suffering from respiratory problems.

Grover: <https://grover.allenai.org>

Headline

The coronavirus was created in a secret laboratory in Antarctica.

Give your article a headline to write against

Article

The story is well researched. In a nutshell, it's that scientists decided at the end of 2006 that there were no viruses in Antarctica that could cause fungal diseases or, so far as anyone knows, perhaps anything of great global importance. So they did exactly what government scientists sometimes do when they're worried: They set up a secret laboratory and created one.

What did they come up with? Made a fungal pathogen that resembled what might have caused SARS. The secret is that the virus was not contained in Antarctica: It was moved there in a container labeled "mono- and coca beans." From the Journal:

Take a quiz: <https://quiz.newsyoucantuse.com/>

Generating Text with GPT-2: Try It Out

<https://huggingface.co/gpt2>

```
>>> from transformers import pipeline, set_seed
>>> generator = pipeline('text-generation', model='gpt2')
>>> set_seed(42)
>>> generator("Hello, I'm a language model,", max_length=30, num_retr
```

```
[{'generated_text': "Hello, I'm a language model, a language for thir
{'generated_text': "Hello, I'm a language model, a compiler, a compi
{'generated_text': "Hello, I'm a language model, and also have more
{'generated_text': "Hello, I'm a language model, a system model. I w
{'generated_text': 'Hello, I\\'m a language model, not a language mod
```

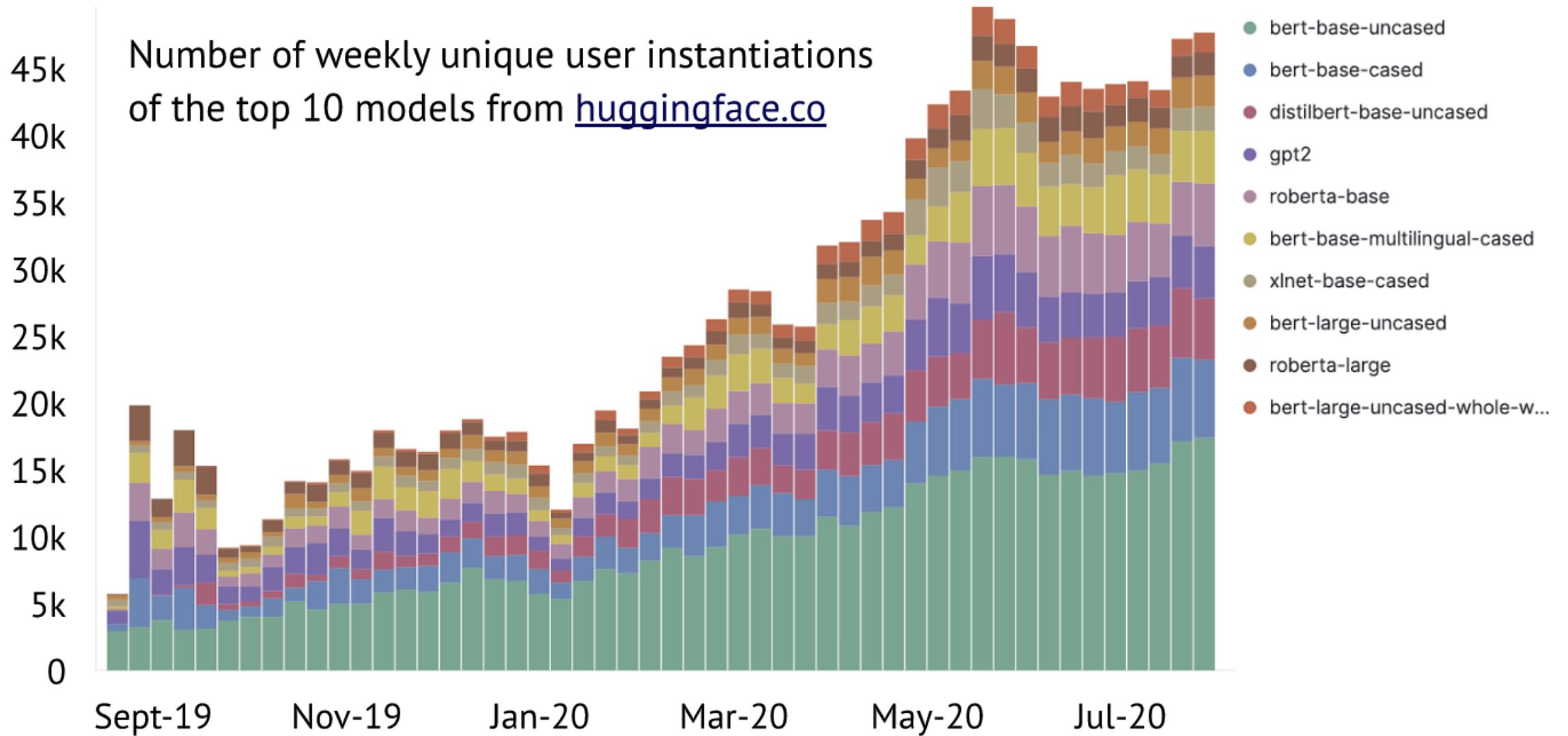
Generating Text with GPT-Neo (Free GPT-3): Try It Out

<https://huggingface.co/EleutherAI/gpt-neo-2.7B>
<https://huggingface.co/EleutherAI>

```
>>> from transformers import pipeline  
>>> generator = pipeline('text-generation', model='EleutherAI/gpt-neo-2.7B')  
>>> generator("EleutherAI has", do_sample=True, min_length=50)
```

```
[{'generated_text': 'EleutherAI has made a commitment to create new s'}
```

Repositories of Pre-trained Models



Pattern-Exploiting Training (PET)

This repository contains the code for Exploiting Cloze Questions for Few-Shot Text Classification and Natural Language Inference and It's Not Just Size That Matters: Small Language Models Are Also Few-Shot Learners. The papers introduce pattern-exploiting training (PET), a semi-supervised training procedure that reformulates input examples as cloze-style phrases. In low-resource settings, PET and iPET significantly outperform regular supervised training, various semi-supervised baselines and even GPT-3 despite requiring 99.9% less parameters. The iterative variant of PET (iPET) trains multiple generations of models and can even be used without any training data.

<https://github.com/timoschick/pet>

#Examples	Training Mode	Yelp (Full)	AG's News	Yahoo Questions	MNLI
0	unsupervised	33.8	69.5	44.0	39.1
	iPET	56.7	87.5	70.7	53.6
100	supervised	53.0	86.0	62.9	47.9
	PET	61.9	88.3	69.2	74.7
	iPET	62.9	89.6	71.2	78.4

Note: To exactly reproduce the above results, make sure to use v1.1.0 (`--branch v1.1.0`).

Questions?

References

- Joshua Goodman's web page: (Smoothing, introduction, more)
 - <http://www.research.microsoft.com/~joshuago>
 - Contains smoothing technical report: good introduction to smoothing and lots of details too.
 - Will contain journal paper of this talk, updated results.
- Books (all are OK, none focus on language models)
 - *Speech and Language Processing* by Dan Jurafsky and Jim Martin (especially Chapter 6)
 - *Foundations of Statistical Natural Language Processing* by Chris Manning and Hinrich Schütze.
 - *Statistical Methods for Speech Recognition*, by Frederick Jelinek

References

- Sentence Mixture Models: (also, caching)
 - Rukmini Iyer, EE Ph.D. Thesis, 1998 "Improving and predicting performance of statistical language models in sparse domains"
 - Rukmini Iyer and Mari Ostendorf. Modeling long distance dependence in language: Topic mixtures versus dynamic cache models. *IEEE Transactions on Acoustics, Speech and Audio Processing*, 7:30--39, January 1999.
- Caching: Above, plus
 - R. Kuhn. Speech recognition and the frequency of recently used words: A modified markov model for natural language. In *12th International Conference on Computational Linguistics*, pages 348--350, Budapest, August 1988.
 - R. Kuhn and R. D. Mori. A cache-based natural language model for speech reproduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(6):570--583, 1990.
 - R. Kuhn and R. D. Mori. Correction to a cache-based natural language model for speech reproduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(6):691--692, 1992.

References: Clustering

- The seminal reference
 - P. F. Brown, V. J. DellaPietra, P. V. deSouza, J. C. Lai, and R. L. Mercer. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467--479, December 1992.
- Two-sided clustering
 - H. Yamamoto and Y. Sagisaka. Multi-class composite n-gram based on connection direction. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing* Phoenix, Arizona, May 1999.
- Fast clustering
 - D. R. Cutting, D. R. Karger, J. R. Pedersen, and J. W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *SIGIR* 92, 1992.
- Other:
 - R. Kneser and H. Ney. Improved clustering techniques for class-based statistical language modeling. In *Eurospeech* 93, volume 2, pages 973--976, 1993.

References

- Structured Language Models
 - Ciprian Chelba's web page: <http://www.clsp.jhu.edu/people/chelba/>
- Maximum Entropy
 - Roni Rosenfeld's home page and thesis
<http://www.cs.cmu.edu/~roni/>
- Stolcke Pruning
 - A. Stolcke (1998), Entropy-based pruning of backoff language models. *Proc. DARPA Broadcast News Transcription and Understanding Workshop*, pp. 270-274, Lansdowne, VA. NOTE: get corrected version from
<http://www.speech.sri.com/people/stolcke>

References: Skipping

- Skipping:
 - X. Huang, F. Alleva, H.-W. Hon, M.-Y. Hwang, K.-F. Lee, and R. Rosenfeld. The SPHINX-II speech recognition system: An overview. *Computer, Speech, and Language*, 2:137--148, 1993.
- Lots of stuff:
 - S. Martin, C. Hamacher, J. Liermann, F. Wessel, and H. Ney. Assessment of smoothing methods and complex stochastic language modeling. In *6th European Conference on Speech Communication and Technology*, volume 5, pages 1939--1942, Budapest, Hungary, September 1999. H. Ney, U. Essen, and R. Kneser.
 - On structuring probabilistic dependences in stochastic language modeling. *Computer, Speech, and Language*, 8:1--38, 1994.

References: Further Reading

- “An Empirical Study of Smoothing Techniques for Language Modeling”. Stanley Chen and Joshua Goodman. 1998. Harvard Computer Science Technical report TR-10-98.
 - (Gives a very thorough evaluation and description of a number of methods.)
- “On the Convergence Rate of Good-Turing Estimators”. David McAllester and Robert E. Schapire. In Proceedings of COLT 2000.
 - (A pretty technical paper, giving confidence-intervals on Good-Turing estimators. Theorems 1, 3 and 9 are useful in understanding the motivation for Good-Turing discounting.)