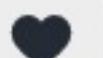


NLP1 2023/24

Language Modelling

Lecturer: Wilker Aziz
(week 1, lecture b)



Where are we at?

- What makes NLP hard
- Text classification
- Language modelling



Self-study: introduction to probabilistic graphical models (PGMs)

[Video](#) and [slides](#)



Ask me anything about PGMs

0 questions

0 upvotes

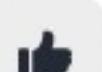
Outline

- Language model
- NGram LM
- Tabular parameterisation
- Evaluation



What is a language model (LM)?

Waiting for responses ...



Order these sentences from most to least plausible

1st | What a nice dog!

2nd | What a dog nice!

3rd | A dog what nice!

4th | What a nice cat!



CONTINUE WITH 1 WORD

Natural language processing is _____

Waiting for responses ...



Those are some of the things an LM can do



Language Model

A language model is a *probability distribution* over the set of all strings in a language.

Being a probability distribution means that 1) an LM can assign probability to text, and 2) you can generate text by drawing samples from the LM.



Applications

- Order alternative sentences (e.g., in speech recognition)
- Generate text in context (e.g., autocomplete)
- Backbone of various NLP systems: translation, summarisation, chatbots



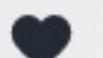
Goals for this class: to learn

- what LMs *are*
- how to *prescribe* LMs
- how to *estimate* LMs
- how to *evaluate* LMs

A good LM is

a probability distribution whose samples
resemble observed text.

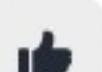
Examples: if the typical sentence has 30 words, sampling from a good LM will reproduce that pattern; if the typical sentence has SUBJ VERB OBJ (in this order), samples from a good LM will exhibit that pattern too.



WHAT IS THE PROBABILITY OF THE SENTENCE BELOW?

What a nice dog!

Waiting for responses ...



Think about it...

A text may be more or less plausible, more or less frequent, more or less expected depending on where we see it (e.g., news, fiction, classroom), who produces it, who it's meant for, etc.

There's no unique notion of THE probability of a piece of text.

Fun fact. De Finetti's seminal *Theory of Probability* starts with a provocative claim: PROBABILITY DOES NOT EXIST. Not as an inherent property of events, rather, probability exists as a construct within models.



Text does not have probability as an inherent property

As probabilities aren't observable, we cannot really learn LMs by regressing from text to target probabilities, instead we learn to assign probability to observed text by designing a statistical parametric model of how texts come about, and optimise our model's parameters using a statistical criterion.



Prescribing LMs

Sample space. To prescribe a probability distribution, we start by specifying the set of outcomes that the LM can generate and assign probability to.

Probability mass function (pmf). We then prescribe a mechanism to assign probability to each and every outcome in the sample space.

Text as a Data Type

Text is a linguistic structure which we view as a finite sequence.

We regard any one piece of text as a finite-length sequence of symbols (or tokens), each coming from the same **countably finite** set of known symbols (i.e., the vocabulary). Example: (what, a, nice, dog, !).

Digital text is a stream of characters, hence we need a tokenisation algorithm.

Sample space

The sample space of a language model is the set of all finite-length sequences made of symbols from a finite vocabulary of symbols \mathcal{W} . We call this set \mathcal{W}^* .

Example: if \mathcal{W} is a good approximation to the vocabulary of English words, then \mathcal{W}^* will include most valid pieces of English-written text (e.g., "the cat is going around"). It also includes many sequences that aren't valid English (e.g., "cat the going around is").



Formalisation

W is a random *word*. An outcome w is a symbol in a vocabulary \mathcal{W} of size V .

$X = \langle W_1, \dots, W_L \rangle$ is a random *sequence* of L words. We can also denote it $W_{1:L}$. An outcome $\langle w_1, \dots, w_l \rangle$ is a sequence of l symbols from \mathcal{W} .

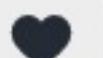
A language model is a mechanism to assign probability $P_X(\langle w_1, \dots, w_l \rangle)$ to any outcome $w_{1:l} \in \mathcal{W}^*$.



Challenge

P_X is a distribution over a countably infinite space of variable-length sequences.

There is no standard statistical distribution over such a sample space. Hence, we need to develop one.

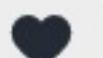


Key Idea

Re-express the probability of a sequence (an outcome from a countably **infinite** space) using the probabilities of the "steps" needed to generate it. Each "step" being an outcome from a countably **finite** space.

$$\begin{aligned} P_X(\langle \text{He, went, to, the, store, EOS} \rangle) &= P_{W|H}(\text{He}|\langle \rangle) \\ &\quad \times P_{W|H}(\text{went}|\langle \text{He} \rangle) \\ &\quad \times P_{W|H}(\text{to}|\langle \text{He, went} \rangle) \\ &\quad \times P_{W|H}(\text{the}|\langle \text{He, went, to} \rangle) \\ &\quad \times P_{W|H}(\text{store}|\langle \text{He, went, to, the} \rangle) \\ &\quad \times P_{W|H}(\text{EOS}|\langle \text{He, went, to, the, store} \rangle) \end{aligned}$$

Example. W is a random word, H is a random *history* of words, EOS is a special word marking the end of a piece of text.



Chain rule

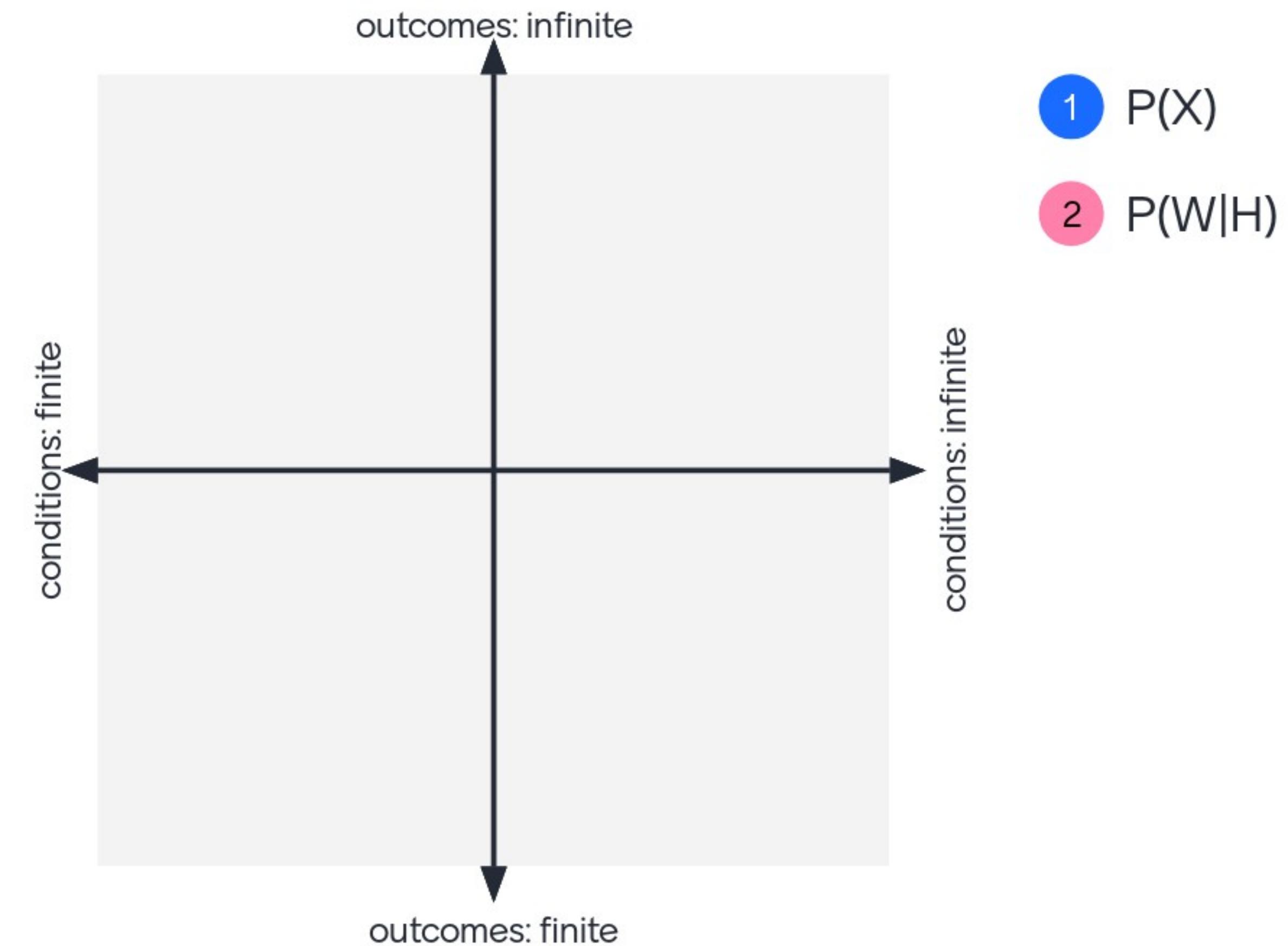
We re-express the probability our LM assigns to $w_{1:l}$ by applying chain rule:

$$P_X(w_{1:l}) = \prod_{i=1}^l P_{W|H}(w_i | w_{<i})$$

Here H is a random *history* of previous tokens. An outcome $w_{<i} = \langle w_1, \dots, w_{i-1} \rangle$ is a sequence of $i - 1$ symbols from \mathcal{W} . When $i \leq 1$, $w_{<i}$ is the empty sequence.



What chain rule achieves



History-conditioned word distributions

For any given history h (e.g., "what a nice"), we need to be able to assign probability $P_{W|H}(w|h)$ to any symbol w in the vocabulary.

That is, we need to assign probability to $P_{W|H}(\text{dog}|h)$, $P_{W|H}(\text{thing}|h)$, $P_{W|H}(\text{what}|h)$, etc., for each and every single one of the V words we know.



Categorical distribution

$$O \sim \text{Categorical}(0.1, 0.2, 0.7)$$

Pronounced: O is distributed as a 3-way
Categorical random variable with
parameter vector $(0.1, 0.2, 0.7)$, whose
coordinates prescribe the probabilities of
the outcomes.

Categorical distribution

We may store the Categorical parameter in a variable, like $\pi = (0.1, 0.2, 0.7)$.

Then we write: $O \sim \text{Categorical}(\pi)$.

Under this statistical model the probability $P_O(k)$ of $O = k$ is precisely π_k .

(For outcomes that are symbols, like "dog" we first create a one-to-one correspondence between words and numerical identifiers)



Categorical distribution

Sometimes, our parameter is specific to a context c ,
and we store it in a variable like $\pi^{(c)} = (0.2, 0.3, 0.5)$.

In a programming language like python you can see
 $\pi^{(c)}$ as something like `dict[c]=(0.2, 0.3, 0.5)`.

Then we write: $O|C = c \sim \text{Categorical}(\pi^{(c)})$.
Pronounced: **given** $C = c$, the distribution of O is
Categorical with parameter $\pi^{(c)}$.

Under this statistical model the probability $P_{O|C}(k|c)$
of $O = k$ conditioned on $C = c$ is precisely $\pi_o^{(c)}$.



Assigning Probability or Generating Outcomes?

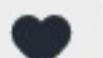
Our procedure to assign probability to an outcome also prescribes a *sampler* (or *simulator*), that is, an algorithm to *generate* outcomes from the LM distribution P_X .

This procedure is also known as a *generative story*.

Generative story

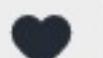
1. Start with an empty history $H = \langle \rangle$. Set $i = 1$.
2. Condition on the available history $w_{<i}$ and draw a word w_i with probability $P_{W|H}(w_i | w_{<i})$ extending the history with it.
3. If w_i is a special end-of-sequence symbol (EOS), terminate, else increment i and repeat (2).

This specifies a **factorisation** of P_X in terms of factors of the kind $P_{W|H}$.



Summary

- An LM is a distribution P_X over text
- Rather than working with P_X directly, we re-express it via chain rule
- For any given history h , an LM predicts a distribution $P_{W|H=h}$ over the vocabulary
- The vocabulary is finite, so $P_{W|H=h}$ is a tractable V -dimensional object.
- But there's no limit to the set of possible histories



Outline

- Language model
- NGram LM
- Tabular parameterisation
- Evaluation



How can we deal with or circumvent the need to deal with a growing history?

Waiting for responses ...



$$\begin{aligned} P_X(\langle \text{He, went, to, the, store, EOS} \rangle) &\stackrel{\text{ind.}}{=} P_W(\text{He}) \\ &\times P_W(\text{went}) \\ &\times P_W(\text{to}) \\ &\times P_W(\text{the}) \\ &\times P_W(\text{store}) \\ &\times P_W(\text{EOS}) \end{aligned}$$

Unigram LM

We could say words are independent of one another.

Though, is it realistic?



Which statement is true under a unigram LM?

$P("a\ nice\ dog") > P("dog\ a\ nice")$



$P("a\ nice\ dog") = P("dog\ a\ nice")$



$$\begin{aligned} P_X(\langle \text{He, went, to, the, store, EOS} \rangle) &\stackrel{\text{ind.}}{=} P_{W|H}(\text{He}|\langle \rangle) \\ &\times P_{W|H}(\text{went}|\langle \text{He} \rangle) \\ &\times P_{W|H}(\text{to}|\langle \text{went} \rangle) \\ &\times P_{W|H}(\text{the}|\langle \text{to} \rangle) \\ &\times P_{W|H}(\text{store}|\langle \text{the} \rangle) \\ &\times P_{W|H}(\text{EOS}|\langle \text{store} \rangle) \end{aligned}$$

Bigram LM

Maybe we say a word is independent of every other word except the previous one?



$$\begin{aligned} P_X(\langle \text{He, went, to, the, store, EOS} \rangle) &\stackrel{\text{ind.}}{=} P_{W|H}(\text{He}|\langle \rangle) \\ &\times P_{W|H}(\text{went}|\langle \text{He} \rangle) \\ &\times P_{W|H}(\text{to}|\langle \text{He went} \rangle) \\ &\times P_{W|H}(\text{the}|\langle \text{went to} \rangle) \\ &\times P_{W|H}(\text{store}|\langle \text{to the} \rangle) \\ &\times P_{W|H}(\text{EOS}|\langle \text{the store} \rangle) \end{aligned}$$

Trigram LM

Maybe it should depend on the previous 2 words?



Markov assumption

The key idea in the NGram LM is to make a conditional independence assumption:

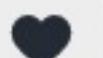
$$P_X(w_{1:l}) = \prod_{i=1}^l P_{W|H}(w_i | \langle w_{i-n+1}, \dots, w_{i-1} \rangle)$$

a word is independent on all but the recent history of n-1 words. This is the so-called **Markov assumption of order n-1.**



Outline

- Language model
- NGram LM
- **Tabular parameterisation**
- Evaluation



Tabular CPDs

Each history-conditioned word distribution
is categorical:

$$W|H = h \sim \text{Categorical}(\boldsymbol{\pi}^{(h)})$$

Where $\boldsymbol{\pi}^{(h)}$ is V-dimensional, $0 \leq \pi_k^{(h)} \leq 1$

for any k , and $\sum_{k=1}^V \pi_k^{(h)} = 1$.

We store $\boldsymbol{\pi}^{(h)}$ for each and every h we know
in a table.



Parameter Estimation

For each unique history h , we have a cpd:

$$W|H = h \sim \text{Categorical}(\boldsymbol{\pi}^{(h)})$$

Given a dataset of observed texts, the maximum likelihood estimate of the conditional probability

$\pi_w^{(h)}$ of w given h is:

$$\pi_w^{(h)} = \frac{\text{count}_{HW}(h, w)}{\sum_{o \in \mathcal{W}} \text{count}_{HW}(h, o)}$$



Unigram LM: size of tabular representation of $P(W|H)$

0	0	0
V	V^*V	V^VV
✓	✗	✗

$x^{(1)} = \text{i want to go to the shop}$										$x^{(2)} = \text{we went to the shop}$										count _W (w)					Total		
h	i	we	want	went	go	to	the	shop	EOS	i	we	want	went	go	to	the	shop	EOS	i	we	want	went	go	to	the	shop	EOS
$\langle \rangle$	1		1		1	2	1	1	1	1		1		1		1	1	1	1	1	1	1	1	3	2	2	14

Table: Table of unigram counts

Unigrams from a toy dataset



Bigram LM: size of tabular representation of $P(W|H)$

0	0	0
V	V^*V	V^VV
✗	✓	✗

	$x^{(1)} = \text{i want to go to the shop}$						$x^{(2)} = \text{we went to the shop}$						count _{HW} (h, w)						Total										
h	i	we	want	went	go	to	the	shop	EOS	i	we	want	went	go	to	the	shop	EOS	i	we	want	went	go	to	the	shop	EOS		
$\langle \text{BOS} \rangle$	1									1									1	1							2		
$\langle \text{i} \rangle$				1																1								1	
$\langle \text{we} \rangle$																					1								1
$\langle \text{want} \rangle$						1																						1	
$\langle \text{went} \rangle$																												1	
$\langle \text{go} \rangle$							1																					2	
$\langle \text{to} \rangle$					1		1																1		2		3		
$\langle \text{the} \rangle$								1																	2		2		
$\langle \text{shop} \rangle$									1																	2		2	

Table: Table of bigram counts

Bigrams from a toy dataset



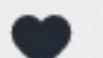
NGram LM: size of tabular representation of $P(W|H)$

0	0	0
V	V^*n	V^n
✗	✗	✓

Memorise phrases

The NGram LM makes up a sentence by gluing phrases, which it memorises in its tabular cpds along with their probabilities.

An increase in the order has an exponential cost: $V^n \rightarrow V^{n+1}$



Parameter Estimation

Given a dataset of observed texts, the maximum likelihood estimate of the conditional probability

$\pi_w^{(h)}$ of w given h is:

$$\pi_w^{(h)} = \frac{\text{count}_{HW}(h, w)}{\sum_{o \in \mathcal{W}} \text{count}_{HW}(h, o)}$$

Can you foresee a problem with this?

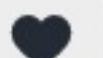


Data sparsity

The longer the history, the less likely it is that we have seen it.

Most of the possible history-word pairs will never be seen.

Tricks: smoothing, interpolation, backoff, etc.
Optional: [section 3.5 of textbook](#).



Outline

- Language model
- NGram LM
- Tabular parameterisation
- Evaluation



Intrinsic

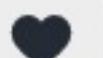
We assess the surprisal (negative log probability) that our model assigns to unseen texts, but we typically re-express it in terms of *perplexity per token* (a measure of average confusion).

Required: [section 3.8 of textbook.](#)



Extrinsic

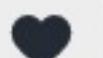
Plug the LM in a task (e.g., autocomplete)
and measure the performance of the task.



Statistical

Compare how the statistics of **generated** text distribute in relation to statistics of **observed** text.

Examples: [Meister and Cotterell, 2021](#) and
[Baan et al, 2023](#)



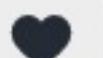
What did you learn today?

Waiting for responses ...



Summary

- LMs are distributions over sentences
- Chain rule is the key to an LM
- Markov assumption (cond. indep.) leads to a viable model
- Classic approach uses inefficient tabular cpds
- Modern approach parameterises cpds using NNs



What next?

