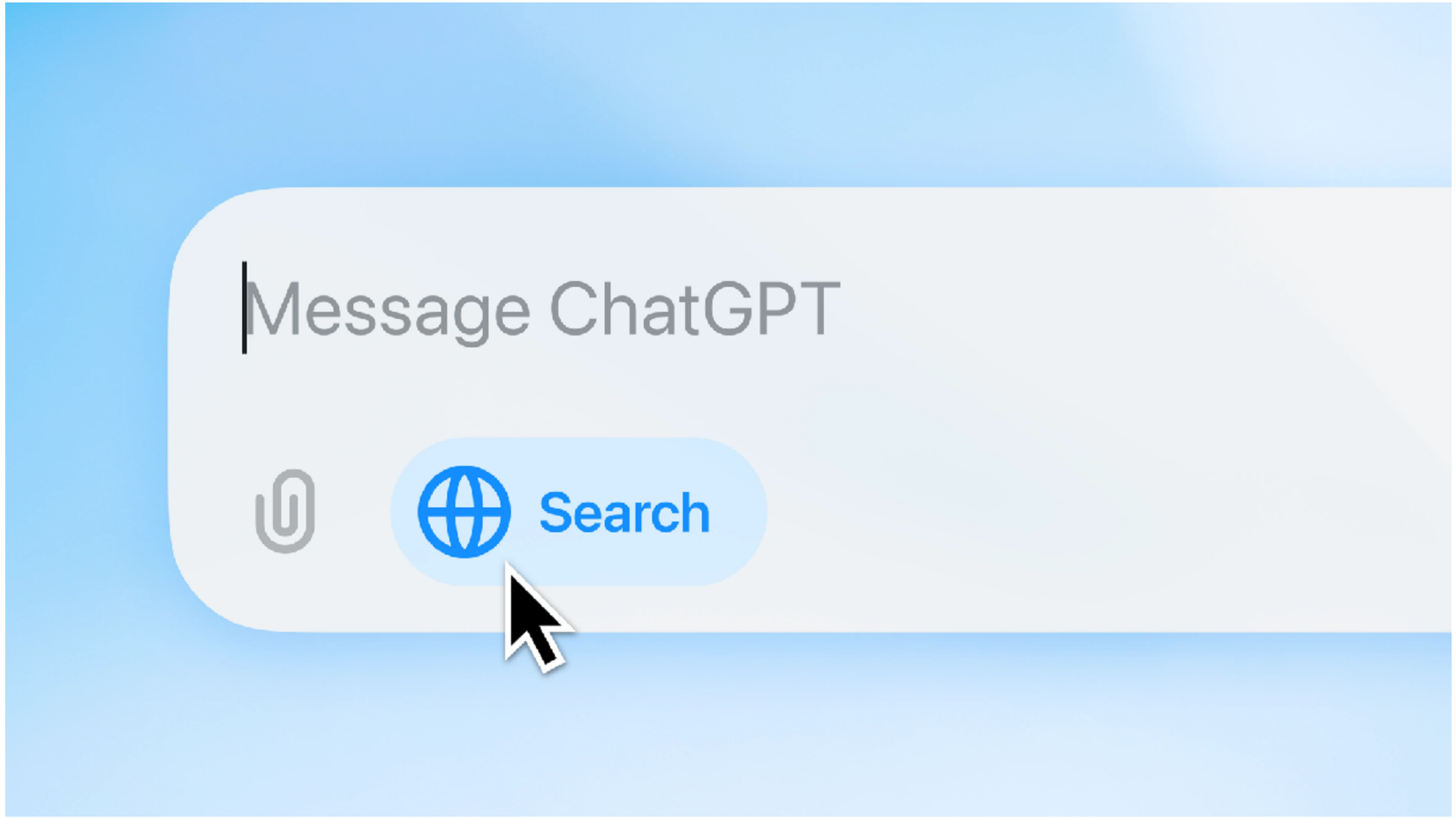
COMP0087 25/26
Lecture 3: N-gram Language Models

20/01/2026

comp0087.github.io

Language models are everywhere



Language models are everywhere

I am studying NLP at _____

my *the* *work*

Language models are everywhere

I am studying NLP at University _____

of *Park* *now*

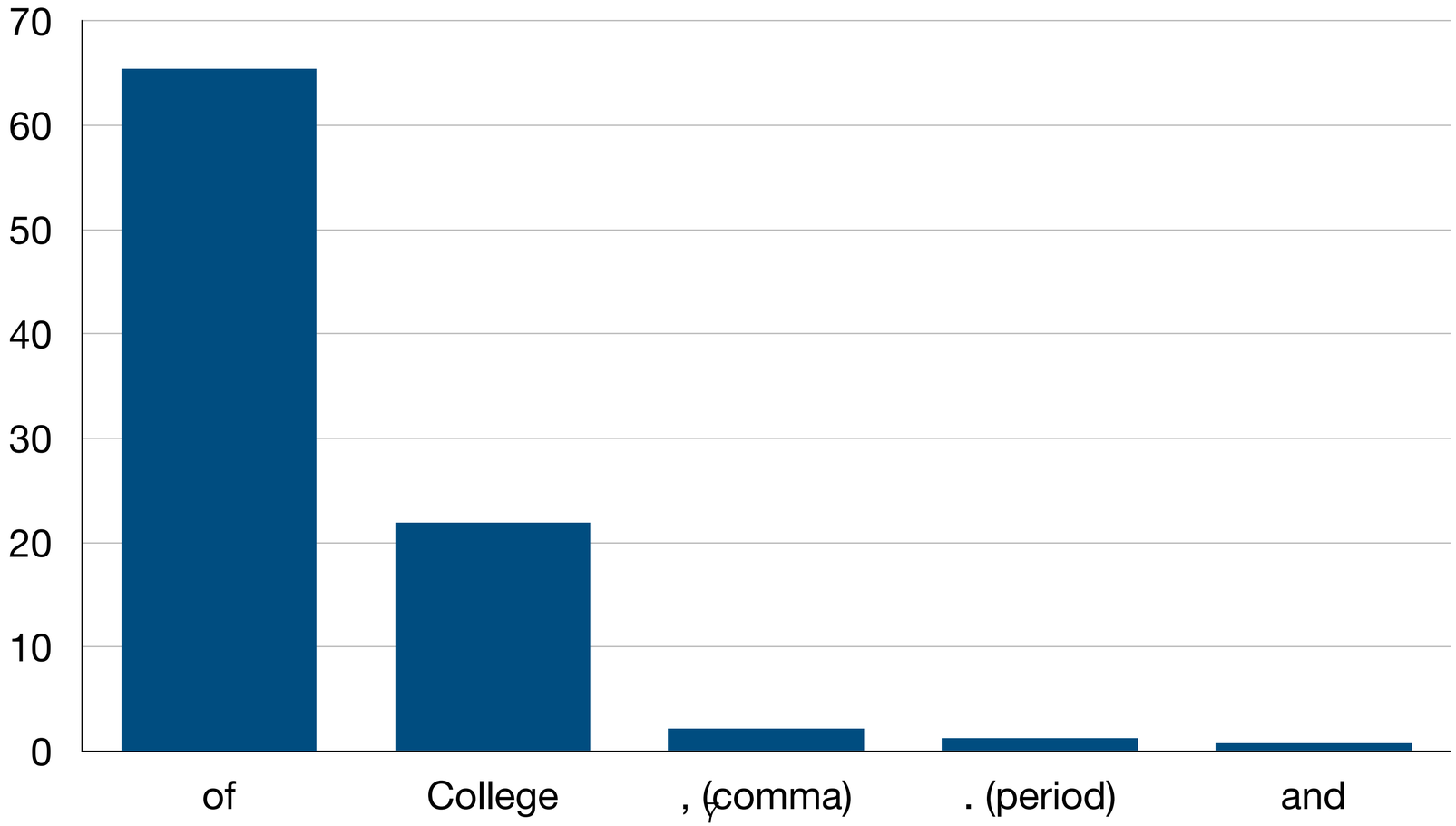
GPT-2 Prediction

I am studying NLP at University _____

```
from transformers import GPT2LMHeadModel, GPT2Tokenizer
import torch

model = GPT2LMHeadModel.from_pretrained("gpt2")
tokenizer = GPT2Tokenizer.from_pretrained("gpt2")

inputs = tokenizer("I am studying NLP at University", return_tensors="pt")

with torch.no_grad():
    logits = model(**inputs).logits[0, -1, :]
    probs = torch.softmax(logits, dim=0)

top_probs, top_ids = torch.topk(probs, 5)
for prob, id in zip(top_probs, top_ids):
    print(f"'{tokenizer.decode(id)}': {prob:.3f}")
```

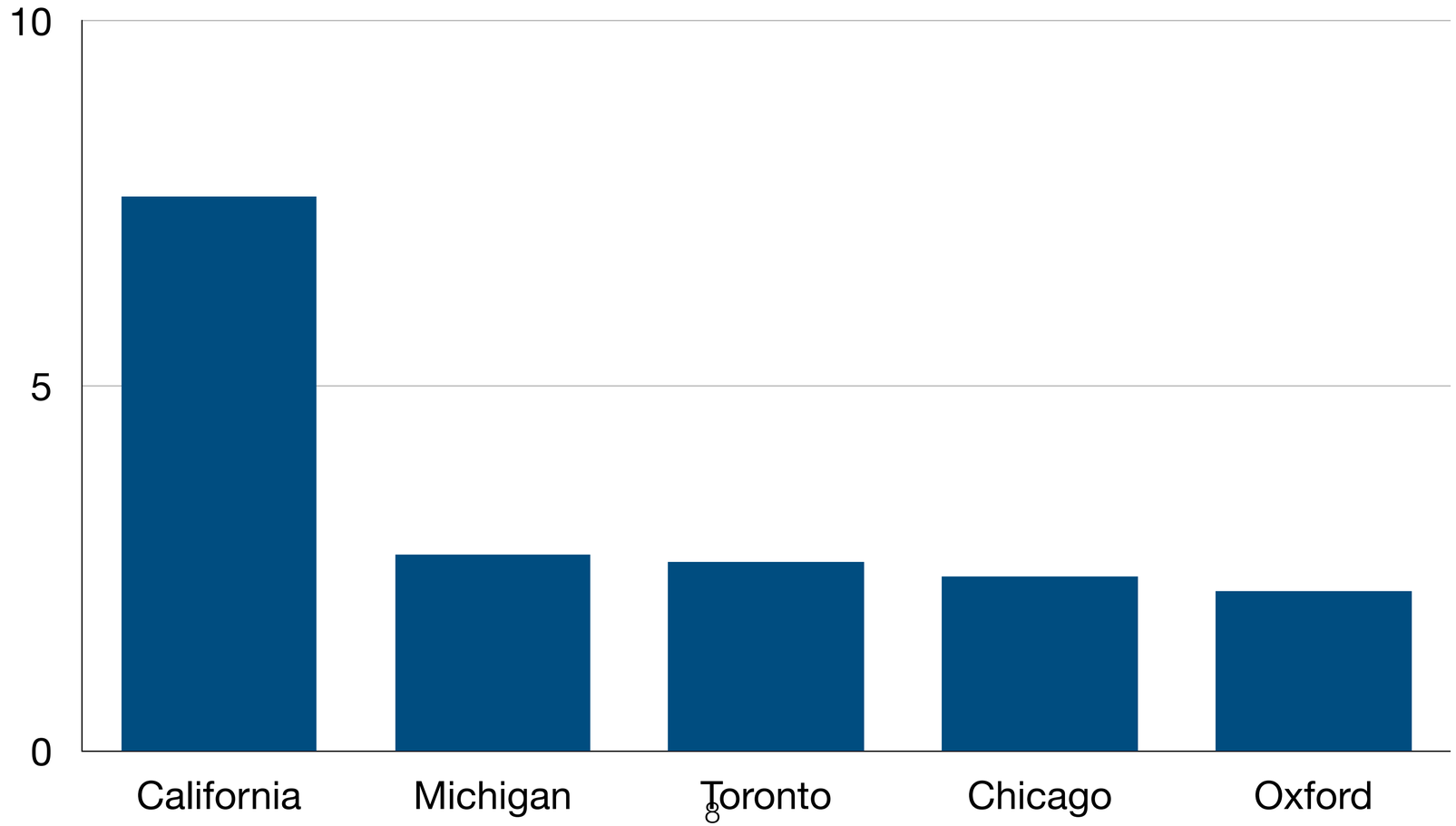
GPT-2 Prediction

I am studying NLP at University _____



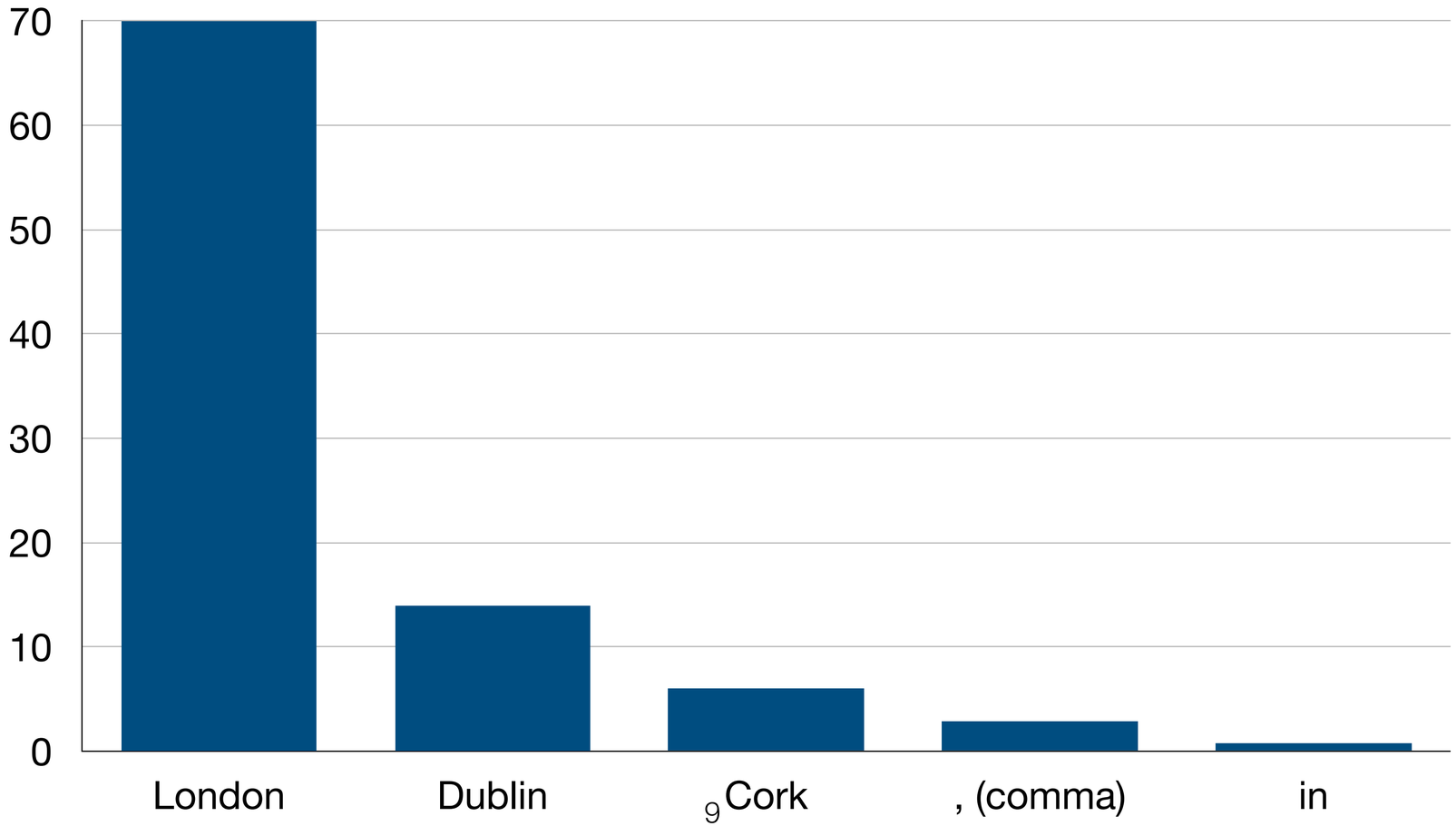
GPT-2 Prediction

I am studying NLP at University of _____



GPT-2 Prediction

I am studying NLP at University College _____



Why?

I am studying NLP at University _____

of

How to change it?

I am studying NLP at University _____

College

How to change it?

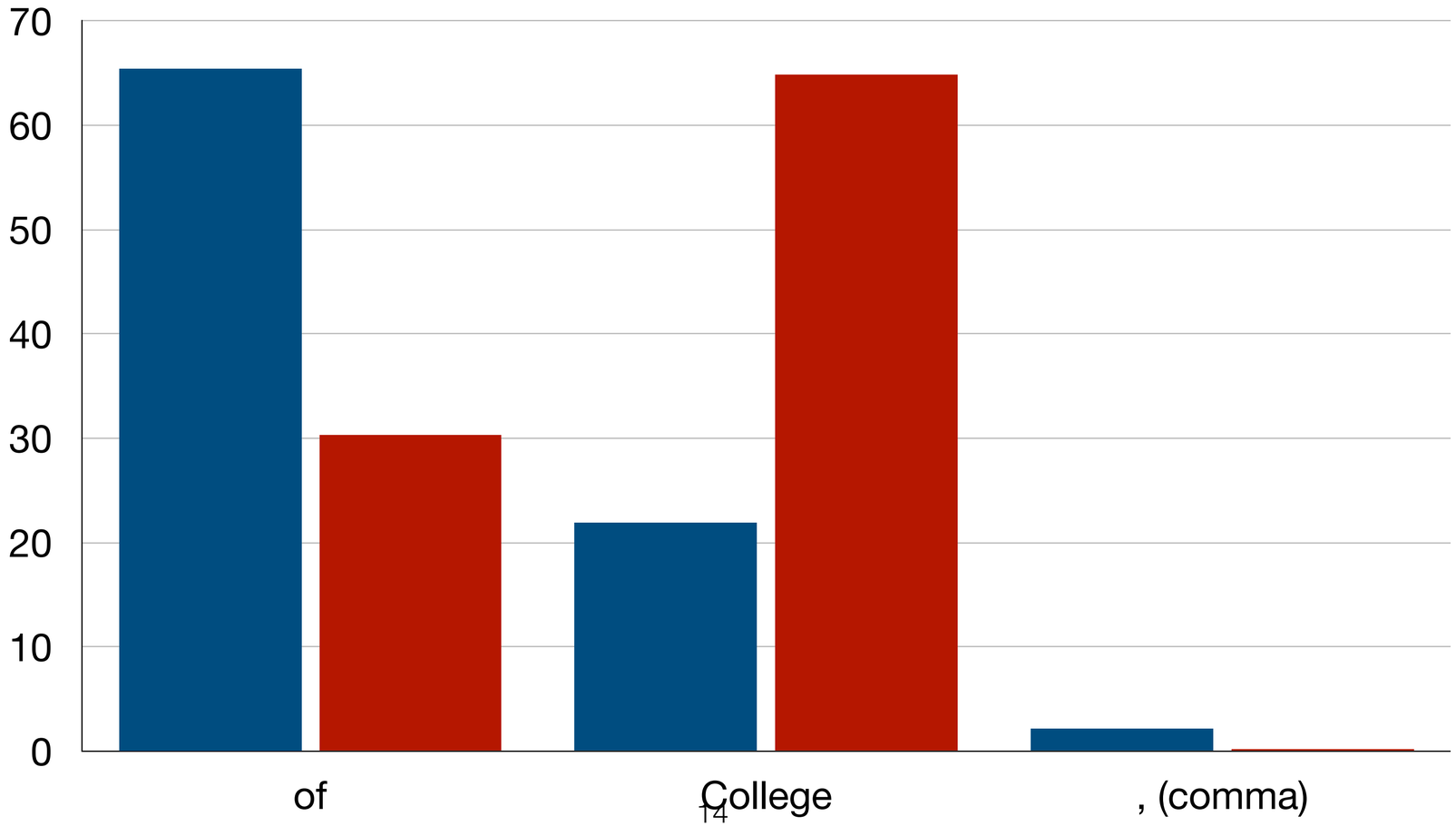
- System prompt: add UCL information
- Fine-tune the model: add more data about UCL NLP

Finetune the model

- We only have a single sentence.
- Single-step gradient update?

Finetune model

I am studying NLP at University _____



You need to

- Have access to LM weight
- Familiar with PyTorch or other DL frameworks
- Basic understanding of optimisation tricks

Research direction

- How to update the parametric knowledge of LLMs
- How to override LLM knowledge with context
- Side effects of tuning on certain facts

A simple method

Repeat the fact 100 times.

No neural networks today.
Counting is all you need.

Data, always data

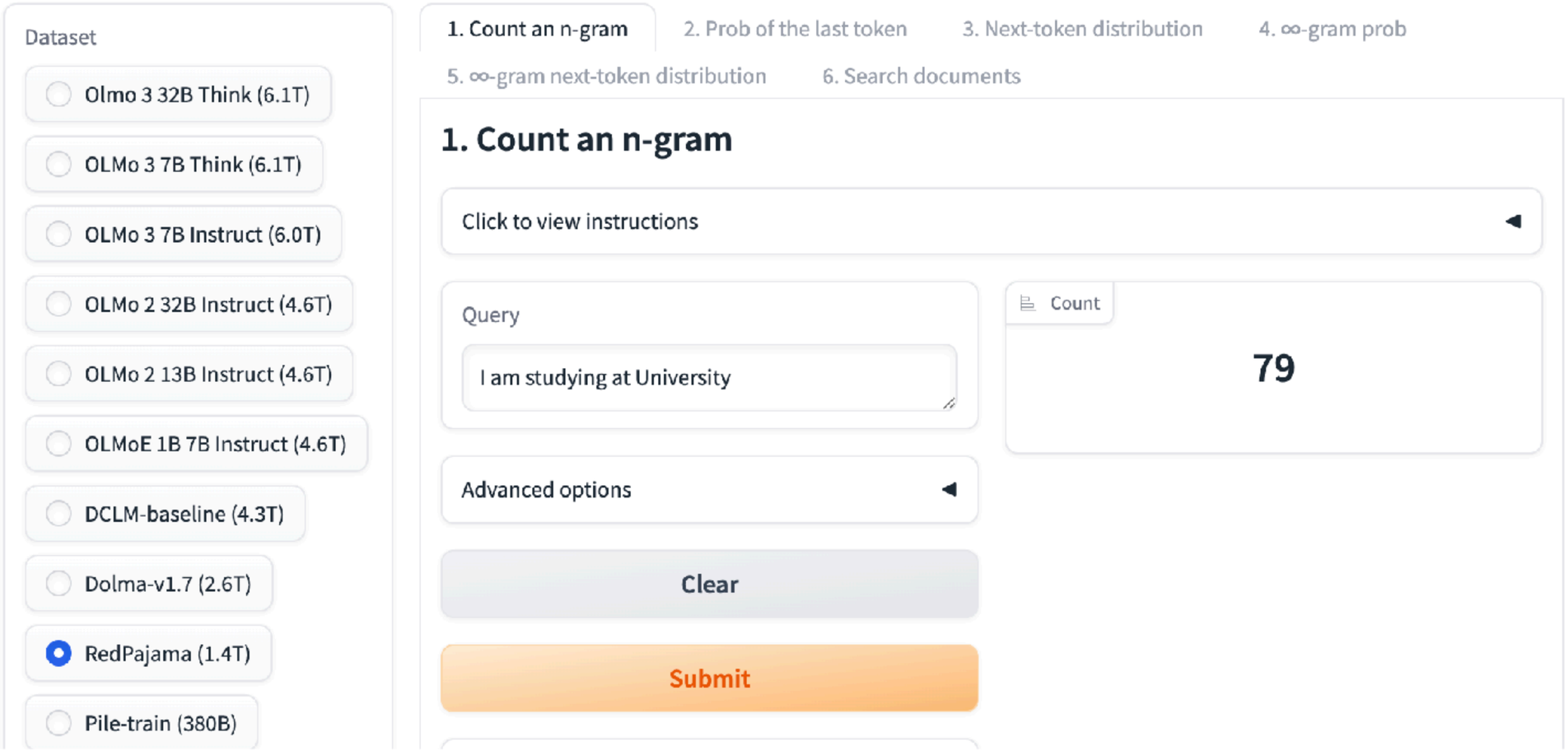
Edit the data distribution directly.

Estimate the probability

I am studying NLP at University _____

$$P(\text{College} \mid \text{I am studying NLP at University}) = \frac{\text{Count}(\text{I am studying NLP at University College})}{\text{Count}(\text{I am studying NLP at University})}$$

Let’s do some counting



https://huggingface.co/spaces/liujch1998/infini-gram

Let’s do some counting

I am studying at University _____

The estimation is based on the RedPajama dataset.

$$\text{Count}(\text{I am studying at University College}) = 3$$
$$\text{Count}(\text{I am studying at University}) = 79$$

Let’s do some counting

I am studying at University _____

The estimation is based on the RedPajama dataset.

$$\text{P(College | I am studying at University)} = \frac{\text{Count(I am studying at University College)}}{\text{Count(I am studying at University)}} = \frac{3}{79} = 3.8\,\%$$

Let’s do some counting

I am studying at University _____

The estimation is based on the Dolma-v1.7 dataset.

$$\text{Count}(\text{I am studying at University College}) = 17$$
$$\text{Count}(\text{I am studying at University}) = 301$$

Let’s do some counting

I am studying at University _____

The estimation is based on the Dolma-v1.7 dataset.

$$\text{P(College | I am studying at University)} = \frac{\text{Count(I am studying at University College)}}{\text{Count(I am studying at University)}} = \frac{17}{301} = 5.6\,\%$$

Not always possible

I am studying NLP at University _____

The estimation is based on the Dolma-v1.7 dataset.

$$P(\text{College} \mid \text{I am studying NLP at University}) =$$

$$\frac{\text{Count(I am studying NLP at University College)}}{\text{Count(I am studying NLP at University)}} = \frac{0}{0}$$

Calculate sentence probability

$$P(\text{I am studying at UCL}) = ?$$

Calculate sentence probability

$$P(\text{I am studying at UCL}) =$$

$$\text{P(I)} \times \text{P(am} \mid \text{I)} \times \text{P(studying} \mid \text{I am)} \times$$
$$\text{P(at} \mid \text{I am studying)} \times \text{P(UCL} \mid \text{I am studying at)}$$

Not always possible: Sparsity Issue

$$P(\text{I am studying NLP at UCL}) =$$

$$\text{P(I)} \times \text{P(am} \mid \text{I)} \times \text{P(studying} \mid \text{I am)} \times$$

$$\text{P(NLP} \mid \text{I am studying)} \times \text{P(at} \mid \text{I am studying NLP)} \times$$

$$\text{P(UCL} \mid \text{I am studying NLP at)}$$

A simplified version

Hypothesis: A word can represent the meaning of its surrounding words.

A simplified version

- The problem: Estimating $P(w_n \mid w_1, w_2, \ldots, w_{n-1})$ requires too much data
- Long histories are rare or unseen in the corpus
- Storage grows exponentially with context length

Solution: only recent context matters

$$P(w_n \mid w_1, w_2, \ldots, w_{n-1}) = P(w_n \mid w_{n-k+1}, \ldots, w_{n-1})$$

Markov assumption

A simplified version

$$P(w_n \mid w_1, w_2, \ldots, w_{n-1}) = P(w_n \mid w_{n-k+1}, \ldots, w_{n-1})$$

$P(\text{UCL} \mid \text{I am studying NLP at}) \approx P(\text{UCL} \mid \text{at})$

$\approx P(\text{UCL})$ *k=1, uni-gram*

$\approx P(\text{UCL} \mid \text{at})$ *k=2, bi-gram*

$\approx P(\text{UCL} \mid \text{NLP at})$ *k=3, tri-gram*

Now this is possible: bigram

I am studying NLP at University _____

The estimation is based on the Dolma-v1.7 dataset.

$$P(\text{College} \mid \text{I am studying NLP at University}) =$$

$$P(\text{College} \mid \text{University}) =$$

$$\frac{\text{Count}(\text{University College})}{\text{Count}(\text{University})} = \frac{2{,}837{,}124}{353{,}875{,}619} = 0.8\,\%$$

But

I am studying NLP at University _____

The estimation is based on the Dolma-v1.7 dataset.

P(College | I am studying NLP at University) =

P(College | University)

NLP is missing

Now this is possible: 4-gram

I am studying NLP at University _____

The estimation is based on the Dolma-v1.7 dataset.

P(College | I am studying NLP at University) =

P(College | NLP at University) =

$$\frac{\text{Count(NLP at University College)}}{\text{Count(NLP at University)}} = \frac{1}{12} = 8.3\,\%$$

N-gram and sparsity issue

$$P(\text{UCL} \mid \text{I am studying NLP at}) \approx P(\text{UCL} \mid \text{at})$$

$$\approx P(\text{UCL}) \qquad \textit{k=1, uni-gram}$$

$$\approx P(\text{UCL} \mid \text{at}) \qquad \textit{k=2, bi-gram}$$

$$\approx P(\text{UCL} \mid \text{NLP at}) \qquad \textit{k=3, tri-gram}$$

Higher order estimation lead to curse of sparsity.

Scaling of N

Large Language Models in Machine Translation

Thorsten Brants **Ashok C. Popat** **Peng Xu** **Franz J. Och** **Jeffrey Dean**

Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94303, USA
{brants,popat,xp,och,jeff}@google.com

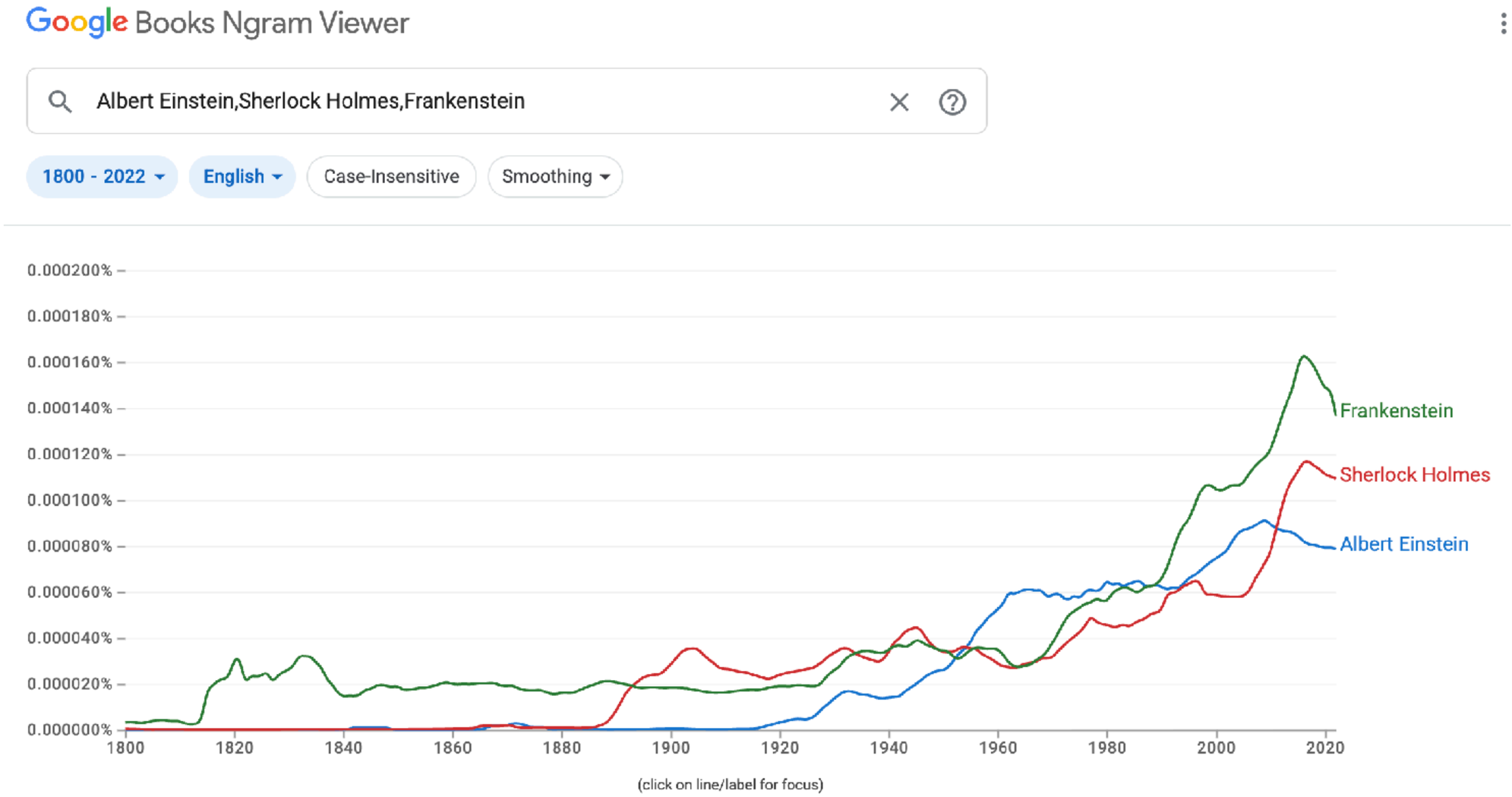
Scaling of N

We trained 5-gram language models on amounts of text varying from 13 million to 2 trillion tokens.

	target	*webnews*	*web*
# tokens	237M	31G	1.8T
vocab size	200k	5M	16M
# n-grams	257M	21G	300G
LM size (SB)	2G	89G	1.8T
time (SB)	20 min	8 hours	1 day
time (KN)	2.5 hours	2 days	–
# machines	100	400	1500

Table 2: Sizes and approximate training times for 3 language models with Stupid Backoff (SB) and Kneser-Ney Smoothing (KN).

Google N-gram dataset



Dealing with zeros

I am studying NLP at University _____

$$P(\text{College} \mid \text{I am studying NLP at University}) =$$

$$\frac{\text{Count(I am studying NLP at University College)}}{\text{Count(I am studying NLP at University)}} = \frac{0}{0}$$

Add-one smoothing

I am studying NLP at University _____

P(College | I am studying NLP at University) =

$$\frac{\text{Count(I am studying NLP at University College)} + 1}{\text{Count(I am studying NLP at University)} + V} = \frac{0 + 1}{0 + V}$$

Add-one smoothing

$$\text{P(College} \mid \text{I am studying NLP at University)} =$$

$$\frac{\text{Count(I am studying NLP at University College)} + 1}{\text{Count(I am studying NLP at University)} + V} = \frac{0 + 1}{0 + V}$$

Pros:

- It's easy to understand and implement.
- No n-gram has a probability of zero.

Add-one smoothing

$$\text{P(College} \mid \text{I am studying NLP at University)} =$$

$$\frac{\text{Count(I am studying NLP at University College)} + 1}{\text{Count(I am studying NLP at University)} + V} = \frac{0 + 1}{0 + V}$$

Cons:

- V could be very large for real-world use cases.
- Treats all unseen n-grams equally.

Add-k smoothing

P(College | I am studying NLP at University) =

$$\frac{\text{Count(I am studying NLP at University College)} + k}{\text{Count(I am studying NLP at University)} + kV} = \frac{0 + k}{0 + kV}, \; 0 < k < 1$$

k can be estimated from the dev set.

How to handle unseen ngrams?

Add-one and add-k smoothing treat them equally.

She visited her grandmother's cottage.

The archipelago has beautiful sternutation.

Lower-gram statistics can help

She visited her grandmother's cottage.

The archipelago has beautiful sternutation.

Interpolation of ngrams

P(College | I am studying NLP at University) =

λ_1 * P(College | I am studying NLP at University) +

λ_2 * P(College | am studying NLP at University) +

λ_3 * P(College | studying NLP at University) +

λ_4 * P(College | NLP at University) +

λ_5 * P(College | at University) +

λ_6 P(College | University) + λ_7 P(College)

Stupid Backoff

$$S(w_i \mid w_{i-n+1}^{i-1}) = \begin{cases} \dfrac{\text{count}(w_{i-n+1}^{i})}{\text{count}(w_{i-n+1}^{i-1})} & \text{if count}(w_{i-n+1}^{i}) > 0 \\ \lambda \cdot S(w_i \mid w_{i-n+2}^{i-1}) & \text{otherwise} \end{cases}$$

Stupid Backoff

P(College | I am studying NLP at University)

Not found.

λ*P(College | am studying NLP at University)

Stupid Backoff

λ*P(College | am studying NLP at University)

Not found.

λ*λ*P(College | studying NLP at University)

Stupid Backoff

λ*λ*P(College | studying NLP at University)

Not found.

λ*λ*λ*P(College | NLP at University)

Stupid Backoff

λ*λ*λ*P(College | NLP at University)

Not found.

λ*λ*λ*λ*P(College | at University)

Stupid Backoff

λ*λ*λ*λ*P(College | at University)

Not found.

λ*λ*λ*λ*λ*P(College | University)

Likely > 0

Stupid Backoff

- A very strong baseline for language model research.
- It is useful for large-scale datasets.

Language model evaluation

How do we know a model is good at a task?

- Question answering
- Machine translation

Task-based Evaluation

Pros

- Evaluation reflects real-world needs
- Task-based metrics are intuitive

Cons

- Evaluation can be slow and expensive
- Requires labeled data for important tasks
- Results may depend heavily on task choice

Computation cost analysis

Evaluation set A: 100 QA pairs

Evaluation set B: 1,000 QA pairs

Intrinsic evaluation for LM

Can we evaluate a language model by the probability it assigns to a test set?

Given a sequence $w_1, \ldots, w_T$

$$P(w_1, \ldots, w_T) = \prod_{t=1}^{T} P(w_t \mid w_{<t})$$

Intrinsic evaluation for LM

$$P(w_1, \ldots, w_T) = \prod_{t=1}^{T} P(w_t \mid w_{<t})$$

- This probability is extremely small
- Longer sequences → smaller values
- Not comparable across sentences of different lengths

Length impact

$$P(w_1, \ldots, w_T) = \prod_{t=1}^{T} P(w_t \mid w_{<t})$$

- Sentence A (10 words): $P = 10^{-8}$
- Sentence B (100 words): $P = 10^{-80}$
- Smaller value does not mean worse modelling

Solution: Perplexity

- Normalize by sequence length
- Use per-word average (log-likelihood)

Solution: Perplexity

$$\text{PPL}(w_1, \ldots, w_T) = P(w_1, \ldots, w_T)^{-\frac{1}{T}} = \sqrt[T]{\frac{1}{P(w_1, \ldots, w_T)}}$$

Lower is better

Solution: Perplexity

$$\text{PPL}(w_1, \ldots, w_T) = \sqrt[T]{\prod_{t=1}^{T} \frac{1}{P(w_t \mid w_{<t})}}$$

Lower is better

Unigram Perplexity

$$\text{PPL}(w_1, \ldots, w_T) = \sqrt[T]{\prod_{t=1}^{T} \frac{1}{P(w_t)}}$$

I am studying NLP at UCL

P(I) × P(am) × P(studying) × P(NLP) × P(at) × P(UCL)

geometric mean of inv. prob

Bi-gram Perplexity

$$\text{PPL}(w_1, \ldots, w_T) = \sqrt[T]{\prod_{t=1}^{T} \frac{1}{P(w_t \mid w_{t-1})}}$$

<s> I am studying NLP at UCL </s>

P(I|<s>) × P(am | I) × P(studying | am) × P(NLP| studying) × P(at| NLP) × P(UCL| at) × P(</s>| UCL)

Tri-gram Perplexity

$$\text{PPL}(w_1, \ldots, w_T) = \sqrt[T]{\prod_{t=1}^{T} \frac{1}{P(w_t \mid w_{t-2} w_{t-1})}}$$

<s> I am studying NLP at UCL </s>

P(I|<s>) × P(am|<s>,I) × P(studying|I,am) × P(NLP|am,studying) × P(at|studying,NLP) × P(UCL|NLP,at) × P(</s>|at,UCL)

Perplexity Evaluation

Doc1: <s> I am studying NLP at UCL </s>

…

Doc N: <s> The Polar Environment Atmospheric Research Laboratory (PEARL) is an atmospheric research facility in the Canadian High Arctic. </s>

<s> I am studying NLP at UCL </s> …<s> The Polar Environment Atmospheric Research Laboratory (PEARL) is an atmospheric research facility in the Canadian High Arctic. </s> (~1 millions words)

Perplexity Evaluation

$$P(w_t \mid w_{t-1})^{-1/N}$$

$$-\frac{1}{N} \log P(w_t \mid w_{t-1})$$

$$\exp^{-\frac{1}{N} \log P(w_t|w_{t-1})}$$

Perplexity Evaluation

	Unigram	Bigram	Trigram
Perplexity	962	170	109

Connecting ngram LMs with modern LLMs

$$P(y \mid x) = \lambda P_{\text{neural}}(y \mid x) + (1 - \lambda) P_{\text{counting}}(y \mid x)$$

Neural LM	Size	Reference Data	Validation			Test		
			Neural	+ ∞-gram		Neural	+ ∞-gram	
GPT-2	117M	Pile-train	22.82	**13.71**	(42%)	22.86	**13.58**	(42%)
GPT-2	345M	Pile-train	16.45	**11.22**	(34%)	16.69	**11.18**	(35%)
GPT-2	774M	Pile-train	15.35	**10.39**	(35%)	15.40	**10.33**	(35%)
GPT-2	1.6B	Pile-train	14.42	**9.93**	(33%)	14.61	**9.93**	(34%)
GPT-Neo	125M	Pile-train	13.50	**10.76**	(22%)	14.08	**10.79**	(25%)
GPT-Neo	1.3B	Pile-train	8.29	**7.31**	(13%)	8.61	**7.36**	(16%)
GPT-Neo	2.7B	Pile-train	7.46	**6.69**	(12%)	7.77	**6.76**	(15%)
GPT-J	6.7B	Pile-train	6.25	**5.75**	(10%)	6.51	**5.85**	(12%)
Llama-2	7B	Pile-train	5.69	**5.05**	(14%)	5.83	**5.06**	(16%)
Llama-2	13B	Pile-train	5.30	**4.75**	(13%)	5.43	**4.76**	(15%)
Llama-2	70B	Pile-train	4.59	**4.21**	(11%)	4.65	**4.20**	(12%)
Llama-2	7B	Pile-train + RedPajama	5.69	**4.66**	(22%)	5.83	**4.66**	(24%)
Llama-2	13B	Pile-train + RedPajama	5.30	**4.41**	(21%)	5.43	**4.42**	(23%)
Llama-2	70B	Pile-train + RedPajama	4.59	**3.96**	(18%)	4.65	**3.95**	(19%)

Table 1: Perplexity (lower is better) on Pile's validation and test sets. Numbers in parentheses are the relative perplexity improvement. The first eight rows share the same tokenizer, and the last six rows share the same tokenizer.

Connecting ngram LMs with modern LLMs

$$P(y \mid x) = \lambda P_{\text{neural}}(y \mid x) + (1 - \lambda) P_{\text{counting}}(y \mid x)$$

Large language models cannot memorise everything.

Thank you!