Natural Language Processing

Language modeling

Yulia Tsvetkov

yuliats@cs.washington.edu

PAUL G. ALLEN SCHOOL
OF COMPUTER SCIENCE & ENGINEERING

Regularization

A solution for overfitting

Add a **regularization** term R(θ) to the loss function (for now written as maximizing logprob rather than minimizing loss)

$$\hat{\theta} = \operatorname*{argmax}_{\theta} \sum_{i=1}^{m} \log P(y^{(i)} | x^{(i)}) - \alpha R(\theta)$$

Idea: choose an R(θ) that penalizes large weights

- fitting the data well with lots of big weights not as good as fitting the data a little less well, with small weights

L2 regularization (ridge regression)

The sum of the squares of the weights

$$R(\theta) = ||\theta||_2^2 = \sum_{j=1}^{n} \theta_j^2$$

L2 regularized objective function:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \left[\sum_{i=1}^{m} \log P(y^{(i)}|x^{(i)}) \right] - \alpha \sum_{j=1}^{n} \theta_j^2$$

L1 regularization (=lasso regression)

The sum of the (absolute value of the) weights

$$R(\theta) \;=\; ||\theta||_1 = \sum_{i=1}^{n} |\theta_i|$$

L1 regularized objective function:

$$\hat{\theta} \;=\; \underset{\theta}{\operatorname{argmax}} \left[\sum_{1=i}^{m} \log P(y^{(i)}|x^{(i)}) \right] - \alpha \sum_{j=1}^{n} |\theta_j|$$

Multinomial Logistic Regression

Often we need more than 2 classes

- Positive/negative/neutral
- Parts of speech (noun, verb, adjective, adverb, preposition, etc.)
- Classify emergency SMSs into different actionable classes

If >2 classes we use **multinomial logistic regression**

= Softmax regression

= Multinomial logit

= (defunct names : Maximum entropy modeling or MaxEnt)

So "logistic regression" will just mean binary (2 output classes)

Multinomial Logistic Regression

The probability of everything must still sum to 1

P(positive|doc) + P(negative|doc) + P(neutral|doc) = 1

Need a generalization of the sigmoid called the **softmax**

- Takes a vector $z = [z_1, z_2, ..., z_k]$ of k arbitrary values
- Outputs a probability distribution
- each value in the range $[0,1]$
- all the values summing to 1

We'll discuss it more when we talk about neural networks

One-hot representation

Gold labels – one-hot representations

[0, 0, ..., 1, 0, 0]

Predicted values – vector of class probabilities

[0.1, 0.05, ..., .08, 0, 0.07]

Sigmoid → softmax

$$\frac{1}{1+e^{-z}} \longrightarrow \frac{e^{z_j}}{\sum_{c=1}^{k} e^{z_c}}$$

The **softmax** function

- Turns a vector $z = [z_1, z_2, ..., z_k]$ of k arbitrary values (logits) into probabilities

$$\text{softmax}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{k} \exp(z_j)} \quad 1 \leq i \leq k$$

- The denominator $\sum_{i=1}^{k} e^{z_i}$ is used to normalize all the values into probabilities

$$\text{softmax}(z) = \left[\frac{\exp(z_1)}{\sum_{i=1}^{k} \exp(z_i)}, \frac{\exp(z_2)}{\sum_{i=1}^{k} \exp(z_i)}, \ldots, \frac{\exp(z_k)}{\sum_{i=1}^{k} \exp(z_i)} \right]$$

softmax: a generalization of sigmoid

- For a vector z of dimensionality k, the softmax is:

$$\text{softmax}(z) = \left[\frac{\exp(z_1)}{\sum_{i=1}^{k}\exp(z_i)}, \frac{\exp(z_2)}{\sum_{i=1}^{k}\exp(z_i)}, \ldots, \frac{\exp(z_k)}{\sum_{i=1}^{k}\exp(z_i)}\right]$$

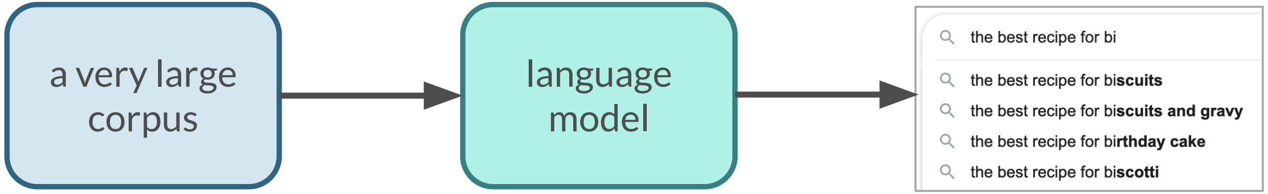
$$\text{softmax}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{k}\exp(z_j)} \quad 1 \leq i \leq k$$

Example:

$$z = [0.6, 1.1, -1.5, 1.2, 3.2, -1.1]$$

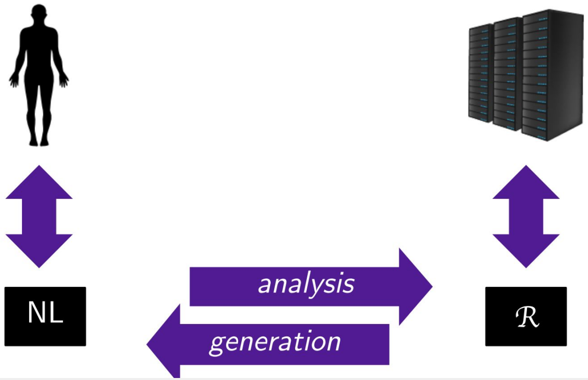
$$\text{softmax}(z) = [0.055, 0.090, 0.006, 0.099, 0.74, 0.010]$$

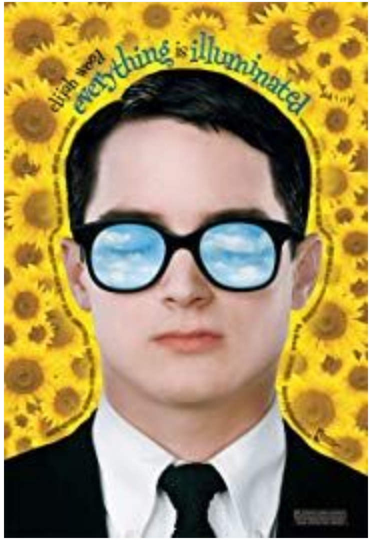
Language modeling

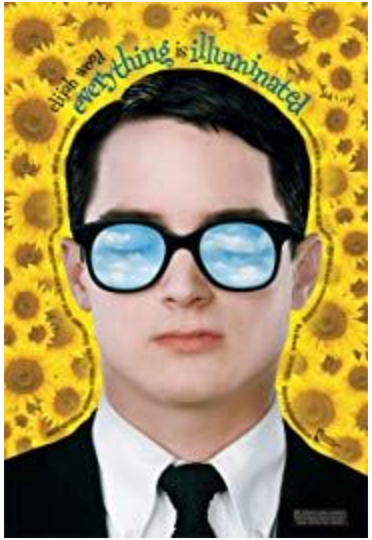


What is Natural Language Processing (NLP)?

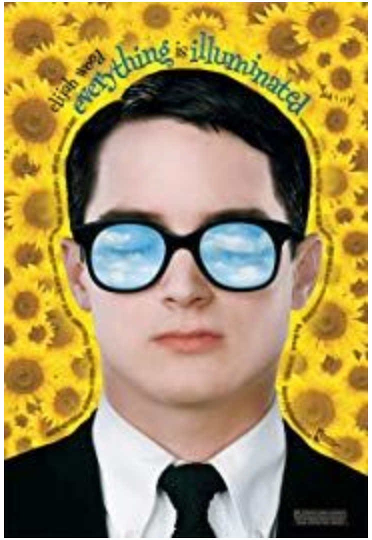
- NL $\in$ {Mandarin Chinese, Hindi, Spanish, Arabic, English, … Inuktitut, Njerep}
- Automation of NLs:
  - analysis of ("understanding") what a text means, to some extent (NL $\rightarrow \mathcal{R}$)
  - generation of fluent, meaningful, context-appropriate text ($\mathcal{R} \rightarrow$ NL)
  - acquisition of $\mathcal{R}$ from knowledge and data



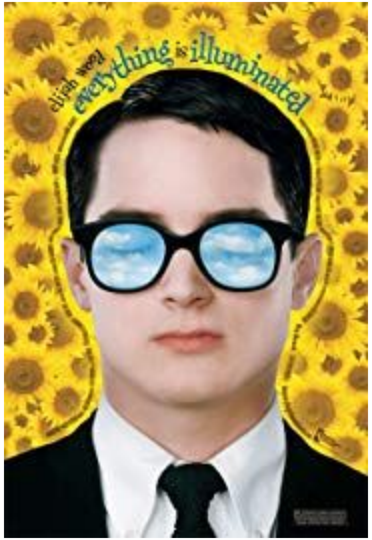

elijah wood
everything is illuminated



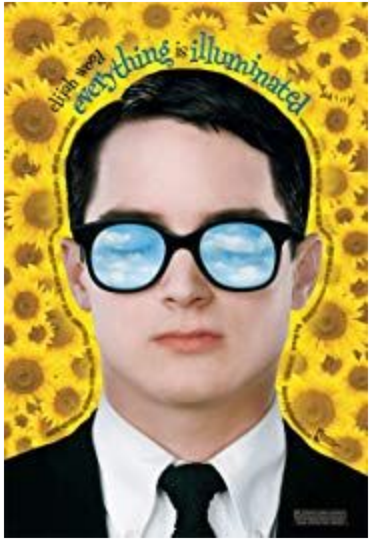
My legal name is Alexander Perchov.



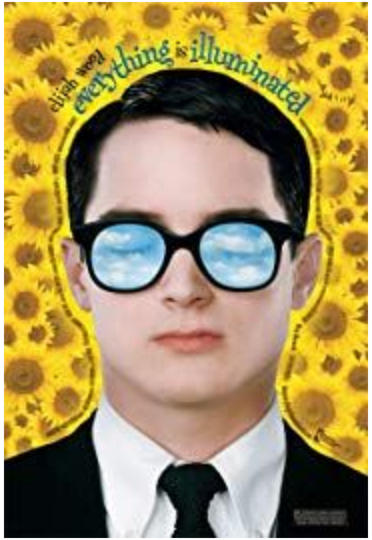
My legal name is Alexander Perchov. But all of my many friends dub me Alex, because that is a more flaccid-to-utter version of my legal name.



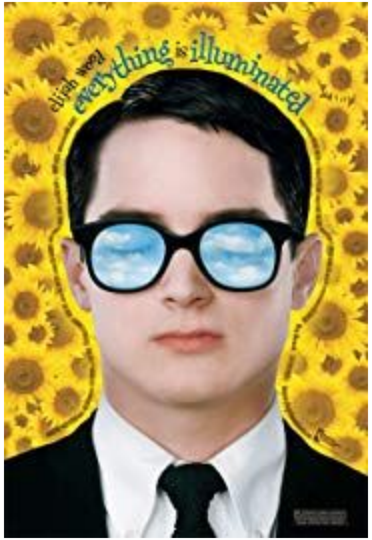
*My legal name is Alexander Perchov. But all of my many friends dub me Alex, because that is a more flaccid-to-utter version of my legal name. Mother dubs me Alexi-stop-spleening-me!, because I am always
spleening her.*



My legal name is Alexander Perchov. But all of my many friends dub me Alex, because that is a more flaccid-to-utter version of my legal name. Mother dubs me Alexi-stop-spleening-me!, because I am always
spleening her. If you want to know why I am always spleening her, it is because I am always elsewhere with friends, and disseminating so much currency, and performing so many things that can spleen a mother.



My legal name is Alexander Perchov. But all of my many friends dub me Alex, because that is a more flaccid-to-utter version of my legal name. Mother dubs me Alexi-stop-spleening-me!, because I am always
spleening her. If you want to know why I am always spleening her, it is because I am always elsewhere with friends, and disseminating so much currency, and performing so many things that can spleen a mother. Father used to dub me Shapka, for the fur hat I would don even in the summer month.



My legal name is Alexander Perchov. But all of my many friends dub me Alex, because that is a more flaccid-to-utter version of my legal name. Mother dubs me Alexi-stop-spleening-me!, because I am always spleening her. If you want to know why I am always spleening her, it is because I am always elsewhere with friends, and disseminating so much currency, and performing so many things that can spleen a mother. Father used to dub me Shapka, for the fur hat I would don even in the summer month. He ceased dubbing me that because I ordered him to cease dubbing me that. It sounded boyish to me, and I have always thought of myself as very potent and generative.

Language models play the role of …

- a judge of grammaticality
- a judge of semantic plausibility
- an enforcer of stylistic consistency
- a repository of knowledge (?)

The Language Modeling problem

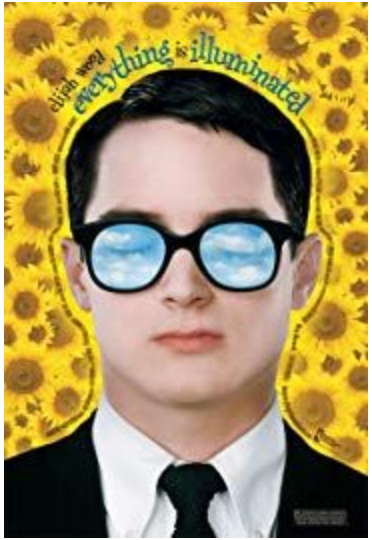
- Assign a probability to every sentence (or any string of words)
  - finite vocabulary (e.g. words or characters) *{the, a, telescope, …}*
  - infinite set of sequences
    - *a telescope STOP*
    - *a STOP*
    - *the the the STOP*
    - *I saw a woman with a telescope STOP*
    - *STOP*
    - *...*

The Language Modeling problem

- Assign a probability to every sentence (or any string of words)
  - finite vocabulary (e.g. words or characters)
  - infinite set of sequences

$$\sum_{\mathbf{e} \in \Sigma^*} p_{\mathrm{LM}}(\mathbf{e}) = 1$$

$$p_{\mathrm{LM}}(\mathbf{e}) \geq 0 \quad \forall \mathbf{e} \in \Sigma^*$$

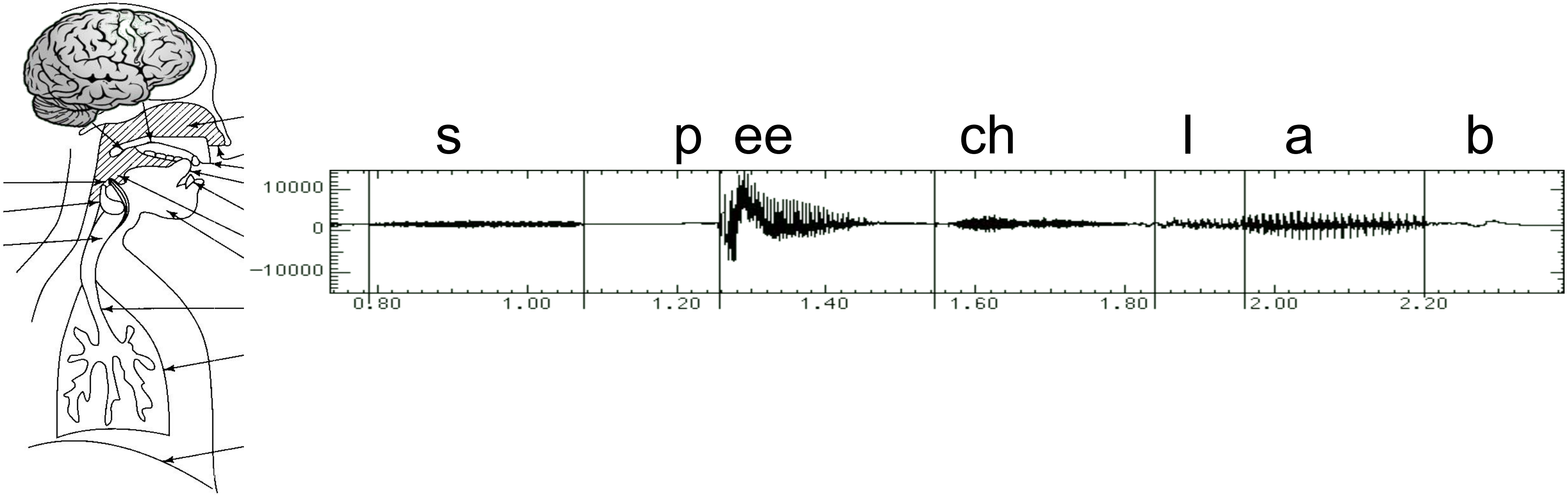


p(*disseminating so much currency STOP*) = 10^{-15}

p(*spending a lot of money STOP*) = 10^{-9}

Motivation

- Speech recognition: we want to predict a sentence given acoustics



Motivation

- Speech recognition: we want to predict a sentence given acoustics

the station signs are indeed in english	-14725
the station signs are in deep in english	-14732
the stations signs are in deep in english	-14735
the station signs are in deep into english	-14739
the station 's signs are in deep in english	-14740
the station signs are in deep in the english	-14741
the station 's signs are indeed in english	-14760
the station signs are indians in english	-14790
the station signs are indian in english	-14799
the stations signs are indians in english	-14807
the stations signs are indians and english	-14815

Motivation

- Machine translation
  - p(*strong winds*) > p(*large winds*)
- Spelling correction
  - The office is about fifteen minuets from my house
  - p(*about fifteen minutes from*) > p(*about fifteen minuets from*)
- Speech recognition
  - p(*I saw a van*) >> p(*eyes awe of an*)
- Summarization, question-answering, handwriting recognition, OCR, etc.

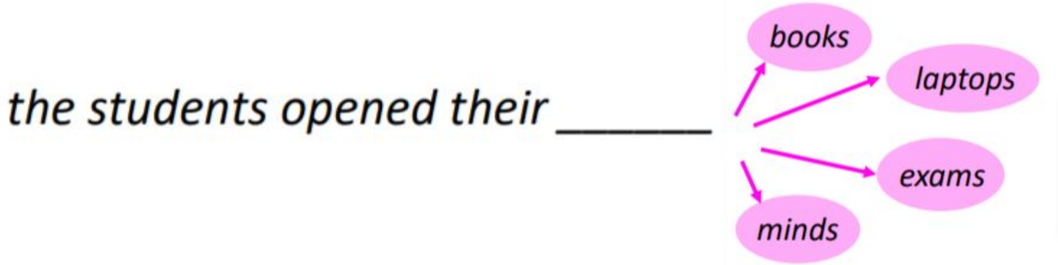
Equivalent definition

- **Language Modeling** is the task of predicting what word comes next

the students opened their ______

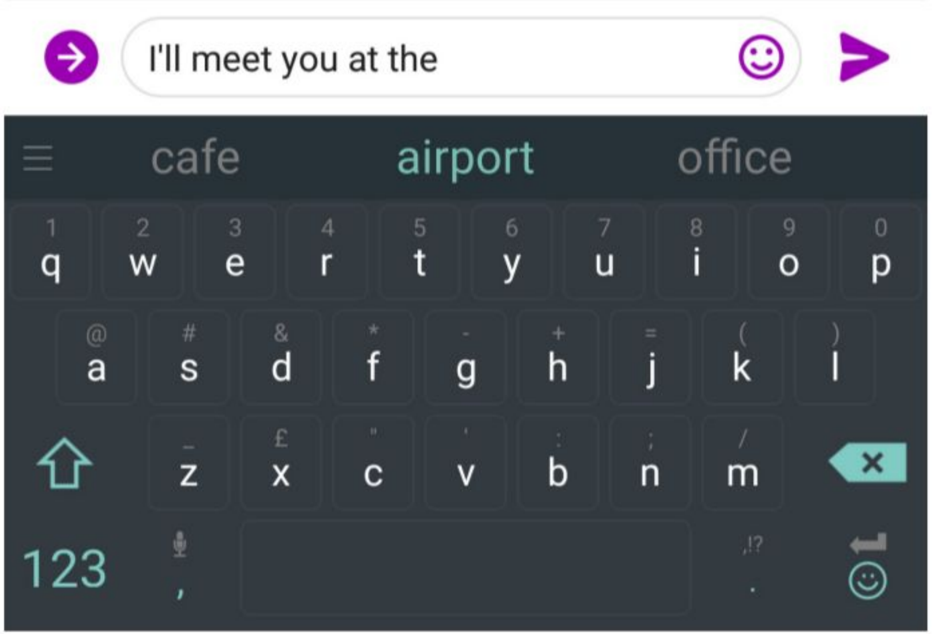
Equivalent definition

- **Language Modeling** is the task of predicting what word comes next

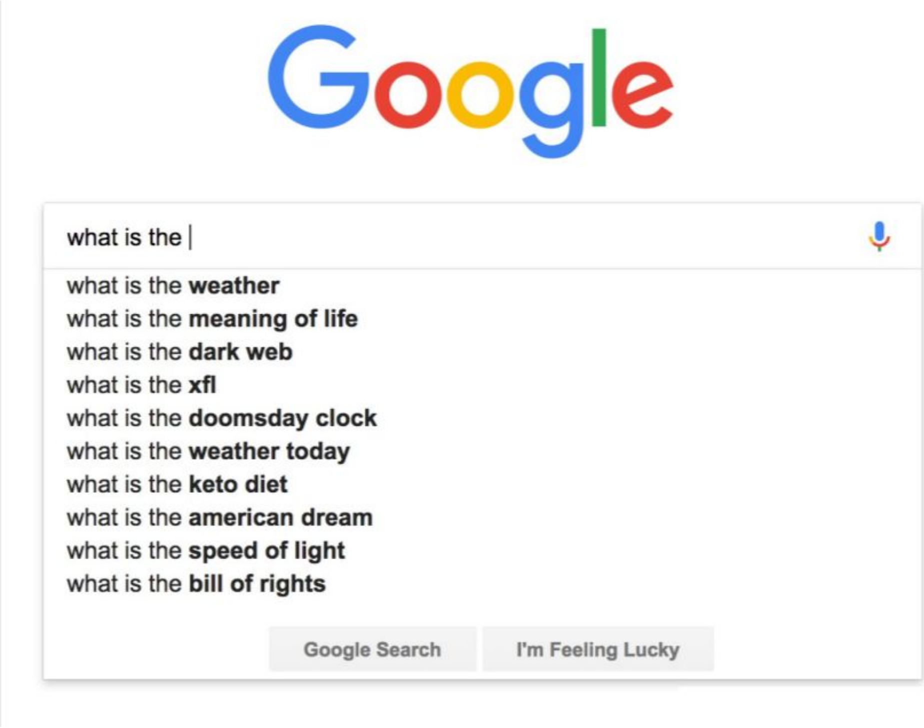


- More formally: given a sequence of words $x^{(1)}, x^{(2)}, \dots x^{(t)}$
  compute the probability distribution of the next word $x^{(t+1)}$
  Where $x^{(t+1)}$ can be any word in the vocabulary $V=\{ w_1, w_2, \dots w_{|V|}\}$

We use Language Models every day



We use Language Models every day



LLMs are language models

ChatGPT 4o

what are the weaknesses of NLP today?

NLP has made great strides but still faces key challenges:

1. **Lack of Deep Understanding**: Models often lack true **contextual comprehension**, struggling with nuanced reasoning and **long-term dependencies** in texts.
2. **Bias and Fairness**: NLP systems inherit **biases** from training data, which can result in unfair outputs. **Debiasing** remains difficult without affecting performance.
3. **Data Quality**: Models often rely on **noisy or low-quality data** from the web, and obtaining high-quality, domain-specific labeled data is expensive and time-consuming.
4. **Domain-Specific Limitations**: General NLP models don't perform well in specialized domains like **medicine** or **law** without significant **adaptation**.
5. **Explainability and Reasoning**: Most models lack **explainability** and struggle with **logical reasoning** or **multi-step inferences**, which is critical for high-stakes applications.
6. **Efficiency and Scalability**: Large models are **resource-intensive**, requiring significant computational power for both training and inference, which limits accessibility.
7. **Multilingual Limitations**: NLP models perform poorly in **low-resource languages** and **code-switching** scenarios, leaving many languages underrepresented.
8. **Safety and Misinformation**: Models can be vulnerable to **adversarial attacks** and can generate **misinformation** easily, posing ethical concerns.

These weaknesses highlight the need for improvements in understanding, fairness, domain adaptation, and safety in NLP systems.

Language Modeling

- If we have some text, then the probability of this text (according to the Language Model) is:

$$
\begin{aligned}
P(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(T)}) &= P(\boldsymbol{x}^{(1)}) \times P(\boldsymbol{x}^{(2)} | \, \boldsymbol{x}^{(1)}) \times \cdots \times P(\boldsymbol{x}^{(T)} | \, \boldsymbol{x}^{(T-1)}, \ldots, \boldsymbol{x}^{(1)}) \\
&= \prod_{t=1}^{T} \underbrace{P(\boldsymbol{x}^{(t)} | \, \boldsymbol{x}^{(t-1)}, \ldots, \boldsymbol{x}^{(1)})}_{\text{This is what our LM provides}}
\end{aligned}
$$

n-gram Language Models

"I have a dog whose name is Lucy. I have two cats, they like playing with Lucy."

- Question: How to learn a Language Model?

A trivial model

- Assume we have n training sentences
- Let $x_1, x_2, \ldots, x_n$ be a sentence, and $c(x_1, x_2, \ldots, x_n)$ be the number of times it appeared in the training data.
- Define a language model:

$$p(x_1, \ldots, x_n) = \frac{c(x_1, \ldots, x_n)}{N}$$

A trivial model

- Assume we have n training sentences
- Let $x_1, x_2, \ldots, x_n$ be a sentence, and $c(x_1, x_2, \ldots, x_n)$ be the number of times it appeared in the training data.
- Define a language model:

$$p(x_1, \ldots, x_n) = \frac{c(x_1, \ldots, x_n)}{N}$$

- No generalization!

n-gram Language Models

"I have a dog whose name is Lucy. I have two cats, they like playing with Lucy."

- Question: How to learn a Language Model?
- Answer (pre- Deep Learning): learn an *n-gram* Language Model!

n-gram Language Models

"I have a dog whose name is Lucy. I have two cats, they like playing with Lucy."

- Definition: An n-gram is a chunk of n consecutive words.

n-gram Language Models

"I have a dog whose name is Lucy. I have two cats, they like playing with Lucy."

- Definition: An n-gram is a chunk of n consecutive words.
  - unigrams: {I, have, a, dog, whose, name, is, Lucy, two, cats, they, like, playing, with}

n-gram Language Models

"I have a dog whose name is Lucy. I have two cats, they like playing with Lucy."

- Definition: An n-gram is a chunk of n consecutive words.
  - unigrams: {I, have, a, dog, whose, name, is, Lucy, two, cats, they, like, playing, with}
  - bigrams: {I have, have a, a dog, dog whose, … , with Lucy}

n-gram Language Models

"I have a dog whose name is Lucy. I have two cats, they like playing with Lucy."

- Definition: An n-gram is a chunk of n consecutive words.
  - unigrams: {I, have, a, dog, whose, name, is, Lucy, two, cats, they, like, playing, with}
  - bigrams: {I have, have a, a dog, dog whose, … , with Lucy}   have cats

n-gram Language Models

"I have a dog whose name is Lucy. I have two cats, they like playing with Lucy."

- Definition: An n-gram is a chunk of n consecutive words.
  - unigrams: {I, have, a, dog, whose, name, is, Lucy, two, cats, they, like, playing, with}
  - bigrams: {I have, have a, a dog, dog whose, … , with Lucy}   have cats

n-gram Language Models

"I have a dog whose name is Lucy. I have two cats, they like playing with Lucy."

- Definition: An n-gram is a chunk of n consecutive words.
  - unigrams: {I, have, a, dog, whose, name, is, Lucy, two, cats, they, like, playing, with}
  - bigrams: {I have, have a, a dog, dog whose, … , with Lucy}
  - trigrams: {I have a, have a dog, a dog whose, … , playing with Lucy}

n-gram Language Models

"I have a dog whose name is Lucy. I have two cats, they like playing with Lucy."

- Definition: An n-gram is a chunk of n consecutive words.
  - unigrams: {I, have, a, dog, whose, name, is, Lucy, two, cats, they, like, playing, with}
  - bigrams: {I have, have a, a dog, dog whose, … , with Lucy}
  - trigrams: {I have a, have a dog, a dog whose, … , playing with Lucy}
  - four-grams: {I have a dog, … , like playing with Lucy}
  - …

n-gram Language Models

"I have a dog whose name is Lucy. I have two cats, they like playing with Lucy."

- w_1 – a unigram
- $w_1\ w_2$ – a bigram
- $w_1\ w_2\ w_3$ – a trigram
- $w_1 w_2 \ldots w_n$ – an n-gram

n-gram Language Models

"I have a dog whose name is Lucy. I have two cats, they like playing with Lucy."

- Question: How to learn a Language Model?
- Answer (pre- Deep Learning): learn an *n-gram* Language Model!

- Idea: Collect statistics about how frequent different n-grams are and use these to predict next word

unigram probability

"I have a dog whose name is Lucy. I have two cats, they like playing with Lucy."

- corpus size m = 17
- P(Lucy) = 2/17; P(cats) = 1/17

- Unigram probability: $$P(w) = \frac{\text{count}(w)}{m} = \frac{C(w)}{m}$$

bigram probability

"I have a dog whose name is Lucy. I have two cats, they like playing with Lucy."

$$P(A \mid B) = \frac{P(A,B)}{P(B)}$$

$$P(\text{have} \mid \text{I}) = \frac{P(\text{I have})}{P(\text{I})} = \frac{2}{2} = 1$$

$$P(\text{two} \mid \text{have}) = \frac{P(\text{have two})}{P(\text{have})} = \frac{1}{2} = 0.5$$

$$P(\text{eating} \mid \text{have}) = \frac{P(\text{have eating})}{P(\text{have})} = \frac{0}{2} = 0$$

$$P(w_2 \mid w_1) = \frac{C(w_1,w_2)}{\sum_w C(w_1,w)} = \frac{C(w_1,w_2)}{C(w_1)}$$

trigram probability

"I have a dog whose name is Lucy. I have two cats, they like playing with Lucy."

$$P(A \mid B) = \frac{P(A,B)}{P(B)}$$

$$P(a \mid I\ have) = \frac{C(I\ have\ a)}{C(I\ have)} = \frac{1}{2} = 0.5$$

$$P(several \mid I\ have) = \frac{C(I\ have\ several)}{C(I\ have)} = \frac{0}{2} = 0$$

$$P(w_3 \mid w_1\, w_2) = \frac{C(w_1,w_2,w_3)}{\sum_w C(w_1,w_2,w)} = \frac{C(w_1,w_2,w_3)}{C(w_1,w_2)}$$

n-gram probability

"I have a dog whose name is Lucy. I have two cats, they like playing with Lucy."

$$P(A \mid B) = \frac{P(A,B)}{P(B)}$$

$$P(w_i \mid w_1, w_2, \ldots, w_{i-1}) = \frac{C(w_1, w_2, \ldots, w_{i-1}, w_i)}{C(w_1, w_2, \ldots, w_{i-1})}$$

Sentence/paragraph/book probability

$$P(\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(T)}) = P(\boldsymbol{x}^{(1)}) \times P(\boldsymbol{x}^{(2)}|\,\boldsymbol{x}^{(1)}) \times \cdots \times P(\boldsymbol{x}^{(T)}|\,\boldsymbol{x}^{(T-1)},\ldots,\boldsymbol{x}^{(1)})$$
$$= \prod_{t=1}^{T} P(\boldsymbol{x}^{(t)}|\,\boldsymbol{x}^{(t-1)},\ldots,\boldsymbol{x}^{(1)})$$

P(its water is so transparent that the) =

P(its) ×
P(water | its) ×
P(is | its water) ×
P(so | its water is) ×
P(transparent | its water is so) ×
… ×

P(the | its water is so transparent that) → How to estimate?

Markov assumption

- We make the Markov assumption: $x^{(t+1)}$ depends only on the preceding n-1 words
  - Markov chain is a "*…stochastic model describing a sequence of possible events in which the probability of each event depends only on the state attained in the previous event.*"

Andrei Markov

$$P(\boldsymbol{x}^{(t+1)}|\boldsymbol{x}^{(t)},\ldots,\boldsymbol{x}^{(1)}) = P(\boldsymbol{x}^{(t+1)}|\underbrace{\boldsymbol{x}^{(t)},\ldots,\boldsymbol{x}^{(t-n+2)}}_{\text{n-1 words}})$$

assumption

Markov assumption

Andrei Markov

$$P(\text{the} \mid \text{its water is so transparent that}) \approx P(\text{the} \mid \text{transparent that})$$

or maybe even

$$P(\text{the} \mid \text{its water is so transparent that}) \approx P(\text{the} \mid \text{that})$$

First-order Markov process

Chain rule

$$p(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) =$$

$$p(X_1 = x_1) \prod_{i=2}^{n} p(X_i = x_i \mid X_1 = x_1, \ldots, X_{i-1} = x_{i-1})$$

First-order Markov process

Chain rule

$$p(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) =$$

$$p(X_1 = x_1) \prod_{i=2}^{n} p(X_i = x_i \mid X_1 = x_1, \ldots, X_{i-1} = x_{i-1})$$

Markov assumption

$$= \quad P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_{i-1} = x_{i-1})$$

Second-order Markov process:

- Relax independence assumption:

$$p(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) =$$
$$p(X_1 = x_1) \times p(X_2 = x_2 \mid X_1 = x_1)$$
$$\times \prod_{i=3}^{n} p(X_i = x_i \mid X_{i-2} = x_{i-2}, X_{i-1} = x_{i-1})$$

Second-order Markov process:

- Relax independence assumption:

$$p(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) =$$
$$p(X_1 = x_1) \times p(X_2 = x_2 \mid X_1 = x_1)$$
$$\times \prod_{i=3}^{n} p(X_i = x_i \mid X_{i-2} = x_{i-2}, X_{i-1} = x_{i-1})$$

- Simplify notation:

$$x_0 = *, x_{-1} = *$$

3-gram LMs

- A trigram language model contains
  - a vocabulary V
  - a non negative parameters q(*w*|*u*,*v*) for every trigram, such that

$$w \in \mathcal{V} \cup \{\text{STOP}\}, \;\; u, v \in \mathcal{V} \cup \{*\}$$

  - the probability of a sentence $x_1, \ldots, x_n$, where x_n=STOP is

$$p(x_1, \ldots, x_n) = \prod_{i=1}^{n} q(x_i \mid x_{i-1}, x_{i-2})$$

Example

$$p(\text{the dog barks STOP}) =$$

Example

$$p(\text{the dog barks STOP}) = q(the \mid *, *) \times$$

Example

$$
\begin{aligned}
p(\text{the dog barks STOP}) = &\, q(the \mid *, *) \times \\
&\, q(dog \mid *, the) \times \\
&\, q(barks \mid the, dog) \times \\
&\, q(STOP \mid dog, barks) \times
\end{aligned}
$$

Berkeley restaurant project sentences

- can you tell me about any good cantonese restaurants close by
- mid priced that food is what i’m looking for
- tell me about chez pansies
- can you give me a listing of the kinds of food that are available
- i’m looking for a good place to eat breakfast
- when is caffe venezia open during the day

Raw bigram counts (~1000 sentences)

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

Bigram probabilities

$$P(w_i \mid w_{i-1}) = \frac{C(w_{i-1}, w_i)}{C(w_{i-1})}$$

$$P(w_1, w_2, \ldots, w_n) \approx \prod_i P(w_i \mid w_{i-1})$$

i	want	to	eat	chinese	food	lunch	spend
2533	927	2417	746	158	1093	341	278

	i	want	to	eat	chinese	food	lunch	spend
i	0.002	0.33	0	0.0036	0	0	0	0.00079
want	0.0022	0	0.66	0.0011	0.0065	0.0065	0.0054	0.0011
to	0.00083	0	0.0017	0.28	0.00083	0	0.0025	0.087
eat	0	0	0.0027	0	0.021	0.0027	0.056	0
chinese	0.0063	0	0	0	0	0.52	0.0063	0
food	0.014	0	0.014	0	0.00092	0.0037	0	0
lunch	0.0059	0	0	0	0	0.0029	0	0
spend	0.0036	0	0.0036	0	0	0	0	0

Bigram estimates of sentence probability

$$P(w_i \mid w_{i-1}) = \frac{C(w_{i-1}, w_i)}{C(w_{i-1})}$$

$$P(w_1, w_2, \ldots, w_n) \approx \prod_i P(w_i \mid w_{i-1})$$

P(<s> i want chinese food </s>) =

P(i|<s>)

× P(want|i)

× P(chinese|want)

× P(food|chinese)

× P(</s>|food)

= …

	i	want	to	eat	chinese	food	lunch	spend
i	0.002	0.33	0	0.0036	0	0	0	0.00079
want	0.0022	0	0.66	0.0011	0.0065	0.0065	0.0054	0.0011
to	0.00083	0	0.0017	0.28	0.00083	0	0.0025	0.087
eat	0	0	0.0027	0	0.021	0.0027	0.056	0
chinese	0.0063	0	0	0	0	0.52	0.0063	0
food	0.014	0	0.014	0	0.00092	0.0037	0	0
lunch	0.0059	0	0	0	0	0.0029	0	0
spend	0.0036	0	0.0036	0	0	0	0	0

What can we learn from bigram estimates?

```
P(to|want)      = 0.66
P(chinese|want)    = 0.0065
P(eat|to)       = 0.28
P (i|<s>)       = 0.25
P(food|to)      = 0.0
P(want|spend)  = 0.0
```

	i	want	to	eat	chinese	food	lunch	spend
i	0.002	0.33	0	0.0036	0	0	0	0.00079
want	0.0022	0	0.66	0.0011	0.0065	0.0065	0.0054	0.0011
to	0.00083	0	0.0017	0.28	0.00083	0	0.0025	0.087
eat	0	0	0.0027	0	0.021	0.0027	0.056	0
chinese	0.0063	0	0	0	0	0.52	0.0063	0
food	0.014	0	0.014	0	0.00092	0.0037	0	0
lunch	0.0059	0	0	0	0	0.0029	0	0
spend	0.0036	0	0.0036	0	0	0	0	0

Sampling from a language model

1 gram	Months the my and issue of year foreign new exchange's september were recession exchange new endorsed a acquire to six executives

Sampling from a language model

1 gram	Months the my and issue of year foreign new exchange's september were recession exchange new endorsed a acquire to six executives
2 gram	Last December through the way to preserve the Hudson corporation N. B. E. C. Taylor would seem to complete the major central planners one point five percent of U. S. E. has already old M. X. corporation of living on information such as more frequently fishing to keep her

Sampling from a language model

1 gram	Months the my and issue of year foreign new exchange's september were recession exchange new endorsed a acquire to six executives
2 gram	Last December through the way to preserve the Hudson corporation N. B. E. C. Taylor would seem to complete the major central planners one point five percent of U. S. E. has already old M. X. corporation of living on information such as more frequently fishing to keep her
3 gram	They also point to ninety nine point six billion dollars from two hundred four oh six three percent of the rates of interest stores as Mexico and Brazil on market conditions

Sampling from a language model

1 gram	–To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have –Hill he late speaks; or! a more to leg less first you enter
2 gram	–Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow. –What means, sir. I confess she? then all sorts, he is trim, captain.
3 gram	–Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, ’tis done. –This shall forbid it should be branded, if renown made it empty.
4 gram	–King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv’d in; –It cannot be but so.