Natural Language Processing (CSEP 517): Introduction & Language Models

Noah Smith

University of Washington
nasmith@cs.washington.edu

March 27, 2017

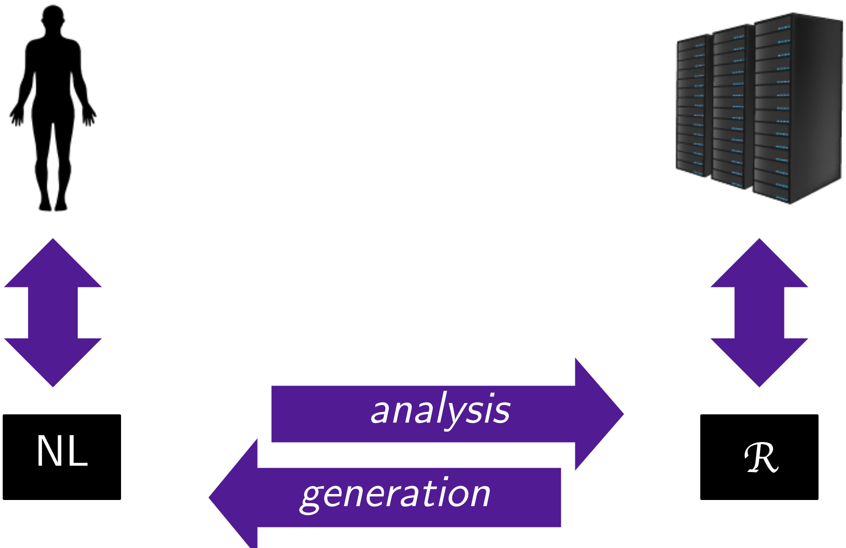
What is NLP?

NL $\in$ {Mandarin Chinese, English, Spanish, Hindi, ..., Lushootseed}

Automation of:

- analysis (NL $\rightarrow \mathcal{R}$)
- generation ($\mathcal{R} \rightarrow$ NL)
- acquisition of $\mathcal{R}$ from knowledge and data

What is $\mathcal{R}$?


NL
analysis
generation
$\mathcal{R}$


THINK
ΣΚΕΨΟΥ
PIENSE
TENK
DENKE
DINK
DENK
THINK
ДУМАЙ
考えろ
सोचिए
$24,000
$77,147
$21,600
WATSON

What does it mean to “know” a language?

Levels of Linguistic Knowledge

speech

text

phonetics

orthography

phonology

morphology

lexemes

"shallower"

syntax

semantics

"deeper"

pragmatics

discourse

Orthography

ลูกศิษย์วัดกระทิงยังยื้อปิดถนนทางขึ้นไปนมัสการพระบาทเขาคิชฌกูฏ หวิดปะทะกับเจ้าถิ่นที่ออกมาเผชิญหน้าเพราะเดือดร้อนสัญจรไม่ได้ ผวจ.เร่งทุกฝ่ายเจรจาก่อนที่ชื่อเสียงของจังหวัดจะเสียหายไปมากกว่านี้ พร้อมเสนอหยุดจัดงาน 15 วัน....

Morphology

uygarlaştıramadıklarımızdanmışsınızcasına
"(behaving) as if you are among those whom we could not civilize"

TIFGOSH ET HA-LELED BA-GAN
"you will meet the boy in the park"

unfriend, Obamacare, Manfuckinghattan

The Challenges of "Words"

- Segmenting text into words (e.g., Thai example)
- Morphological variation (e.g., Turkish and Hebrew examples)
- Words with multiple meanings: *bank*, *mean*
- Domain-specific meanings: *latex*
- Multiword expressions: *make a decision*, *take out*, *make up*, *bad hombres*

Example: Part-of-Speech Tagging

ikr smh he asked fir yo last name

so he can add u on fb lololol

Example: Part-of-Speech Tagging

I know, right	shake my head			for	your		
ikr	smh	he	asked	fir	yo	last	name

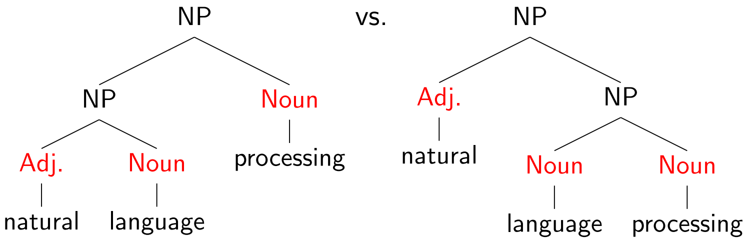
				you		Facebook	laugh out loud
so	he	can	add	u	on	fb	lololol

Example: Part-of-Speech Tagging

I know, right	shake my head			for	your		
ikr	smh	he	asked	fir	yo	last	name
!	G	O	V	P	D	A	N
interjection	acronym	pronoun	verb	prep.	det.	adj.	noun

				you		Facebook	laugh out loud
so	he	can	add	u	on	fb	lololol
P	O	V	V	O	P	∧	!
preposition						proper noun	

Syntax


NP
NP
Adj.
natural
Noun
language
Noun
processing
vs.
NP
Adj.
natural
NP
Noun
language
Noun
processing

Morphology + Syntax

A ship-shipping ship, shipping shipping-ships.

Syntax + Semantics

We saw the woman with the telescope wrapped in paper.

Syntax + Semantics

We saw the woman with the telescope wrapped in paper.

- Who has the telescope?

Syntax + Semantics

We saw the woman with the telescope wrapped in paper.

- Who has the telescope?
- Who or what is wrapped in paper?

Syntax + Semantics

We saw the woman with the telescope wrapped in paper.

- Who has the telescope?
- Who or what is wrapped in paper?
- An event of perception, or an assault?

Semantics

Every fifteen minutes a woman in this country gives birth.

– Groucho Marx

Semantics

Every fifteen minutes a woman in this country gives birth. Our job is to find this woman, and stop her!

– Groucho Marx

Can $\mathcal{R}$ be "Meaning"?

Depends on the application!

- Giving commands to a robot
- Querying a database
- Reasoning about relatively closed, grounded worlds

Harder to formalize:

- Analyzing opinions
- Talking about politics or policy
- Ideas in science

Why NLP is Hard

1. Mappings across levels are complex.
   - A string may have many possible interpretations in different contexts, and resolving **ambiguity** correctly may rely on knowing a lot about the world.
   - **Richness**: any meaning may be expressed many ways, and there are immeasurably many meanings.
   - Linguistic **diversity** across languages, dialects, genres, styles, . . .
2. Appropriateness of a representation depends on the application.
3. Any $\mathcal{R}$ is a theorized construct, not directly observable.
4. There are many sources of variation and noise in linguistic input.

Desiderata for NLP Methods

(ordered arbitrarily)

1. Sensitivity to a wide range of the phenomena and constraints in human language
2. Generality across different languages, genres, styles, and modalities
3. Computational efficiency at construction time and runtime
4. Strong formal guarantees (e.g., convergence, statistical efficiency, consistency, etc.)
5. High accuracy when judged against expert annotations and/or task-specific performance

NLP $\stackrel{?}{=}$ Machine Learning

- To be successful, a machine learner needs bias/assumptions; for NLP, that might be linguistic theory/representations.
- $\mathcal{R}$ is not directly observable.
- Early connections to information theory (1940s)
- Symbolic, probabilistic, and connectionist ML have all seen NLP as a source of inspiring applications.

NLP $\stackrel{?}{=}$ Linguistics

- NLP must contend with NL data as found in the world
- NLP $\approx$ computational linguistics
- Linguistics has begun to use tools originating in NLP!

Fields with Connections to NLP

- Machine learning
- Linguistics (including psycho-, socio-, descriptive, and theoretical)
- Cognitive science
- Information theory
- Logic
- Theory of computation
- Data science
- Political science
- Psychology
- Economics
- Education

The Engineering Side

- Application tasks are difficult to define formally; they are always evolving.
- Objective evaluations of performance are always up for debate.
- Different applications require different $\mathcal{R}$.
- People who succeed in NLP for long periods of time are foxes, not hedgehogs.

Today's Applications

- Conversational agents
- Information extraction and question answering
- Machine translation
- Opinion and sentiment analysis
- Social media analysis
- Rich visual understanding
- Essay evaluation
- Mining legal, medical, or scholarly literature

Factors Changing the NLP Landscape
(Hirschberg and Manning, 2015)

- Increases in computing power
- The rise of the web, then the social web
- Advances in machine learning
- Advances in understanding of language in social context

Administrivia

Course Website

http://courses.cs.washington.edu/courses/csep517/17sp/

Your Instructors

Noah (instructor):

- UW CSE professor since 2015, teaching NLP since 2006, studying NLP since 1998, first NLP program in 1991
- Research interests: machine learning for structured problems in NLP, NLP for social science

George (TA):

- Computer Science Ph.D. student
- Research interests: machine learning for multilingual NLP

Outline of CSE 517

1. **Probabilistic language models**, which define probability distributions over text passages. (about 2 weeks)
2. **Text classifiers**, which infer attributes of a piece of text by "reading" it. (about 1 week)
3. **Sequence models** (about 1 week)
4. **Parsers** (about 2 weeks)
5. **Semantics** (about 2 weeks)
6. **Machine translation** (about 1 week)

Readings

- Main reference text: Jurafsky and Martin, 2008, some chapters from new edition (Jurafsky and Martin, forthcoming) when available
- Course notes from the instructor and others
- Research articles

Lecture slides will include references for deeper reading on some topics.

Evaluation

- Approximately five assignments (A1–5), completed individually (50%).
- Quizzes (20%), given roughly weekly, online
- An exam (30%), to take place at the end of the quarter

Evaluation

- Approximately five assignments (A1–5), completed individually (50%).
  - Some pencil and paper, mostly programming
  - Graded mostly on your writeup (so please take written communication seriously!)
- Quizzes (20%), given roughly weekly, online
- An exam (30%), to take place at the end of the quarter

To-Do List

- Entrance survey: due Wednesday
- Online quiz: due Friday
- Print, sign, and return the academic integrity statement
- Read: Jurafsky and Martin (2008, ch. 1), Hirschberg and Manning (2015), and Smith (2017);
  optionally, Jurafsky and Martin (2016) and Collins (2011) §2
- A1, out today, due April 7

Very Quick Review of Probability

- Event space (e.g., $\mathcal{X}$, $\mathcal{Y}$)—in this class, usually discrete

Very Quick Review of Probability

- Event space (e.g., $\mathcal{X}$, $\mathcal{Y}$)—in this class, usually discrete
- Random variables (e.g., X, Y)

Very Quick Review of Probability

- Event space (e.g., $\mathcal{X}$, $\mathcal{Y}$)—in this class, usually discrete
- Random variables (e.g., X, Y)
- Typical statement: "random variable X takes value $x \in \mathcal{X}$ with probability $p(X = x)$, or, in shorthand, $p(x)$"

Very Quick Review of Probability

- Event space (e.g., $\mathcal{X}$, $\mathcal{Y}$)—in this class, usually discrete
- Random variables (e.g., X, Y)
- Typical statement: "random variable X takes value $x \in \mathcal{X}$ with probability $p(X = x)$, or, in shorthand, $p(x)$"
- Joint probability: $p(X = x, Y = y)$

Very Quick Review of Probability

- Event space (e.g., $\mathcal{X}$, $\mathcal{Y}$)—in this class, usually discrete
- Random variables (e.g., X, Y)
- Typical statement: "random variable X takes value $x \in \mathcal{X}$ with probability $p(X = x)$, or, in shorthand, $p(x)$"
- Joint probability: $p(X = x, Y = y)$
- Conditional probability: $p(X = x \mid Y = y)$

Very Quick Review of Probability

- Event space (e.g., $\mathcal{X}$, $\mathcal{Y}$)—in this class, usually discrete
- Random variables (e.g., X, Y)
- Typical statement: "random variable X takes value $x \in \mathcal{X}$ with probability $p(X = x)$, or, in shorthand, $p(x)$"
- Joint probability: $p(X = x, Y = y)$
- Conditional probability: $p(X = x \mid Y = y) = \dfrac{p(X = x, Y = y)}{p(Y = y)}$

Very Quick Review of Probability

- Event space (e.g., $\mathcal{X}$, $\mathcal{Y}$)—in this class, usually discrete
- Random variables (e.g., X, Y)
- Typical statement: "random variable X takes value $x \in \mathcal{X}$ with probability $p(X = x)$, or, in shorthand, $p(x)$"
- Joint probability: $p(X = x, Y = y)$
- Conditional probability: $p(X = x \mid Y = y) = \dfrac{p(X = x, Y = y)}{p(Y = y)}$
- Always true:
  $p(X = x, Y = y) = p(X = x \mid Y = y) \cdot p(Y = y) = p(Y = y \mid X = x) \cdot p(X = x)$

Very Quick Review of Probability

- Event space (e.g., $\mathcal{X}$, $\mathcal{Y}$)—in this class, usually discrete
- Random variables (e.g., X, Y)
- Typical statement: "random variable X takes value $x \in \mathcal{X}$ with probability $p(X = x)$, or, in shorthand, $p(x)$"
- Joint probability: $p(X = x, Y = y)$
- Conditional probability: $p(X = x \mid Y = y) = \dfrac{p(X = x, Y = y)}{p(Y = y)}$
- Always true:
  $p(X = x, Y = y) = p(X = x \mid Y = y) \cdot p(Y = y) = p(Y = y \mid X = x) \cdot p(X = x)$
- Sometimes true: $p(X = x, Y = y) = p(X = x) \cdot p(Y = y)$

Very Quick Review of Probability

- Event space (e.g., $\mathcal{X}$, $\mathcal{Y}$)—in this class, usually discrete
- Random variables (e.g., X, Y)
- Typical statement: "random variable X takes value $x \in \mathcal{X}$ with probability $p(X = x)$, or, in shorthand, $p(x)$"
- Joint probability: $p(X = x, Y = y)$
- Conditional probability: $p(X = x \mid Y = y) = \dfrac{p(X = x, Y = y)}{p(Y = y)}$
- Always true:
  $p(X = x, Y = y) = p(X = x \mid Y = y) \cdot p(Y = y) = p(Y = y \mid X = x) \cdot p(X = x)$
- Sometimes true: $p(X = x, Y = y) = p(X = x) \cdot p(Y = y)$
- The difference between *true* and *estimated* probability distributions

Language Models: Definitions

- $\mathcal{V}$ is a finite set of (discrete) symbols (☺ "words" or possibly characters); $V = |\mathcal{V}|$
- $\mathcal{V}^\dagger$ is the (infinite) set of sequences of symbols from $\mathcal{V}$ whose final symbol is 🛑
- $p : \mathcal{V}^\dagger \rightarrow \mathbb{R}$, such that:
  - For any $\boldsymbol{x} \in \mathcal{V}^\dagger$, $p(\boldsymbol{x}) \geq 0$
  - $\displaystyle\sum_{\boldsymbol{x} \in \mathcal{V}^\dagger} p(\boldsymbol{X} = \boldsymbol{x}) = 1$

  (I.e., p is a proper probability distribution.)

Language modeling: estimate p from examples, $\boldsymbol{x}_{1:n} = \langle \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n \rangle$.

Immediate Objections

1. Why would we want to do this?
2. Are the nonnegativity and sum-to-one constraints really necessary?
3. Is "finite $\mathcal{V}$" realistic?

Motivation: Noisy Channel Models

A pattern for modeling a pair of random variables, $\boldsymbol{D}$ and $\boldsymbol{O}$:

$$\boxed{\text{source}} \longrightarrow \boldsymbol{D} \longrightarrow \boxed{\text{channel}} \longrightarrow \boldsymbol{O}$$

Motivation: Noisy Channel Models

A pattern for modeling a pair of random variables, $\boldsymbol{D}$ and $\boldsymbol{O}$:

$$\boxed{\text{source}} \longrightarrow \boldsymbol{D} \longrightarrow \boxed{\text{channel}} \longrightarrow \boldsymbol{O}$$

- $\boldsymbol{D}$ is the plaintext, the true message, the missing information, the output

Motivation: Noisy Channel Models

A pattern for modeling a pair of random variables, $\boldsymbol{D}$ and $\boldsymbol{O}$:

$$\boxed{\text{source}} \longrightarrow \boldsymbol{D} \longrightarrow \boxed{\text{channel}} \longrightarrow \boldsymbol{O}$$

- $\boldsymbol{D}$ is the plaintext, the true message, the missing information, the output
- $\boldsymbol{O}$ is the ciphertext, the garbled message, the observable evidence, the input

Motivation: Noisy Channel Models

A pattern for modeling a pair of random variables, $\boldsymbol{D}$ and $\boldsymbol{O}$:

$$\boxed{\text{source}} \longrightarrow \boldsymbol{D} \longrightarrow \boxed{\text{channel}} \longrightarrow \boldsymbol{O}$$

- $\boldsymbol{D}$ is the plaintext, the true message, the missing information, the output
- $\boldsymbol{O}$ is the ciphertext, the garbled message, the observable evidence, the input
- Decoding: select $\boldsymbol{d}$ given $\boldsymbol{O} = \boldsymbol{o}$.

$$\begin{aligned}
\boldsymbol{d}^* &= \operatorname*{argmax}_{\boldsymbol{d}} p(\boldsymbol{d} \mid \boldsymbol{o}) \\
&= \operatorname*{argmax}_{\boldsymbol{d}} \frac{p(\boldsymbol{o} \mid \boldsymbol{d}) \cdot p(\boldsymbol{d})}{p(\boldsymbol{o})} \\
&= \operatorname*{argmax}_{\boldsymbol{d}} \underbrace{p(\boldsymbol{o} \mid \boldsymbol{d})}_{\text{channel model}} \cdot \underbrace{p(\boldsymbol{d})}_{\text{source model}}
\end{aligned}$$

Noisy Channel Example: Speech Recognition

$$\boxed{\text{source}} \longrightarrow \text{sequence in } \mathcal{V}^{\dagger} \longrightarrow \boxed{\text{channel}} \longrightarrow \text{acoustics}$$

- Acoustic model defines $p(\text{sounds} \mid \boldsymbol{d})$ (channel)
- Language model defines $p(\boldsymbol{d})$ (source)

Noisy Channel Example: Speech Recognition

Credit: Luke Zettlemoyer

word sequence	$\log p(\text{acoustics} \mid \text{word sequence})$
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 signs are indeed in english	-14757
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

Noisy Channel Example: Machine Translation

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

Noisy Channel Examples

- Speech recognition
- Machine translation
- Optical character recognition
- Spelling and grammar correction

Immediate Objections

1. Why would we want to do this?
2. Are the nonnegativity and sum-to-one constraints really necessary?
3. Is "finite $\mathcal{V}$" realistic?

Evaluation: Perplexity

Intuitively, language models should assign high probability to real language they have not seen before.

For out-of-sample ("held-out" or "test") data $\bar{\boldsymbol{x}}_{1:m}$:

- Probability of $\bar{\boldsymbol{x}}_{1:m}$ is $\prod_{i=1}^{m} p(\bar{\boldsymbol{x}}_i)$

Evaluation: Perplexity

Intuitively, language models should assign high probability to real language they have not seen before.
For out-of-sample ("held-out" or "test") data $\bar{\boldsymbol{x}}_{1:m}$:

- Probability of $\bar{\boldsymbol{x}}_{1:m}$ is $\displaystyle\prod_{i=1}^{m} p(\bar{\boldsymbol{x}}_i)$
- Log-probability of $\bar{\boldsymbol{x}}_{1:m}$ is $\displaystyle\sum_{i=1}^{m} \log_2 p(\bar{\boldsymbol{x}}_i)$

Evaluation: Perplexity

Intuitively, language models should assign high probability to real language they have not seen before.

For out-of-sample ("held-out" or "test") data $\bar{\boldsymbol{x}}_{1:m}$:

- Probability of $\bar{\boldsymbol{x}}_{1:m}$ is $\displaystyle\prod_{i=1}^{m} p(\bar{\boldsymbol{x}}_i)$
- Log-probability of $\bar{\boldsymbol{x}}_{1:m}$ is $\displaystyle\sum_{i=1}^{m} \log_2 p(\bar{\boldsymbol{x}}_i)$
- Average log-probability per word of $\bar{\boldsymbol{x}}_{1:m}$ is

$$l = \frac{1}{M} \sum_{i=1}^{m} \log_2 p(\bar{\boldsymbol{x}}_i)$$

if $M = \sum_{i=1}^{m} |\bar{\boldsymbol{x}}_i|$ (total number of words in the corpus)

Evaluation: Perplexity

Intuitively, language models should assign high probability to real language they have not seen before.

For out-of-sample ("held-out" or "test") data $\bar{\boldsymbol{x}}_{1:m}$:

- Probability of $\bar{\boldsymbol{x}}_{1:m}$ is $\displaystyle\prod_{i=1}^{m} p(\bar{\boldsymbol{x}}_i)$
- Log-probability of $\bar{\boldsymbol{x}}_{1:m}$ is $\displaystyle\sum_{i=1}^{m} \log_2 p(\bar{\boldsymbol{x}}_i)$
- Average log-probability per word of $\bar{\boldsymbol{x}}_{1:m}$ is

$$l = \frac{1}{M} \sum_{i=1}^{m} \log_2 p(\bar{\boldsymbol{x}}_i)$$

  if $M = \sum_{i=1}^{m} |\bar{\boldsymbol{x}}_i|$ (total number of words in the corpus)
- Perplexity (relative to $\bar{\boldsymbol{x}}_{1:m}$) is 2^{-l}

Evaluation: Perplexity

Intuitively, language models should assign high probability to real language they have not seen before.

For out-of-sample ("held-out" or "test") data $\bar{\boldsymbol{x}}_{1:m}$:

- Probability of $\bar{\boldsymbol{x}}_{1:m}$ is $\displaystyle\prod_{i=1}^{m} p(\bar{\boldsymbol{x}}_i)$
- Log-probability of $\bar{\boldsymbol{x}}_{1:m}$ is $\displaystyle\sum_{i=1}^{m} \log_2 p(\bar{\boldsymbol{x}}_i)$
- Average log-probability per word of $\bar{\boldsymbol{x}}_{1:m}$ is

$$l = \frac{1}{M} \sum_{i=1}^{m} \log_2 p(\bar{\boldsymbol{x}}_i)$$

  if $M = \sum_{i=1}^{m} |\bar{\boldsymbol{x}}_i|$ (total number of words in the corpus)
- Perplexity (relative to $\bar{\boldsymbol{x}}_{1:m}$) is 2^{-l}

Lower is better.

Understanding Perplexity

$$2^{-\frac{1}{M}\sum_{i=1}^{m}\log_2 p(\bar{\boldsymbol{x}}_i)}$$

It's a branching factor!

- Assign probability of 1 to the test data $\Rightarrow$ perplexity $= 1$
- Assign probability of $\frac{1}{|\mathcal{V}|}$ to every word $\Rightarrow$ perplexity $= |\mathcal{V}|$
- Assign probability of 0 to *anything* $\Rightarrow$ perplexity $= \infty$
  - This motivates a stricter constraint than we had before:
    - For any $\boldsymbol{x} \in \mathcal{V}^{\dagger}$, $p(\boldsymbol{x}) > 0$

Perplexity

- Perplexity on conventionally accepted test sets is often reported in papers.
- Generally, I won't discuss perplexity numbers much, because:
  - Perplexity is only an intermediate measure of performance.
  - Understanding the models is more important than remembering how well they perform on particular train/test sets.
- If you're curious, look up numbers in the literature; always take them with a grain of salt!

Immediate Objections

1. Why would we want to do this?
2. Are the nonnegativity and sum-to-one constraints really necessary?
3. Is “finite $\mathcal{V}$” realistic?

Is "finite $\mathcal{V}$" realistic?

No

Is “finite $\mathcal{V}$” realistic?

No
no
n0
-no
notta
Nº
/no
//no
(no
|no

The Language Modeling Problem

Input: $\boldsymbol{x}_{1:n}$ ("training data")
Output: $p : \mathcal{V}^{\dagger} \rightarrow \mathbb{R}^{+}$
☺ p should be a "useful" measure of plausibility (not grammaticality).

A Trivial Language Model

$$p(\boldsymbol{x}) = \frac{|\{i \mid \boldsymbol{x}_i = \boldsymbol{x}\}|}{n} \qquad = \frac{c_{\boldsymbol{x}_{1:n}}(\boldsymbol{x})}{n}$$

A Trivial Language Model

$$p(\boldsymbol{x}) = \frac{|\{i \mid \boldsymbol{x}_i = \boldsymbol{x}\}|}{n} \qquad = \frac{c_{\boldsymbol{x}_{1:n}}(\boldsymbol{x})}{n}$$

What if $\boldsymbol{x}$ is not in the training data?

Using the Chain Rule

$$
p(\boldsymbol{X} = \boldsymbol{x}) = \left(\begin{array}{l}
p(X_1 = x_1 \mid X_0 = x_0) \\
\cdot\ p(X_2 = x_2 \mid X_{0:1} = x_{0:1}) \\
\cdot\ p(X_3 = x_3 \mid X_{0:2} = x_{0:2}) \\
\vdots \\
\cdot\ p(X_\ell = \quad \mid X_{0:\ell-1} = x_{0:\ell-1})
\end{array} \right)
$$

$$
= \prod_{j=1}^{\ell} p(X_j = x_j \mid X_{0:j-1} = x_{0:j-1})
$$

Unigram Model

$$
\begin{aligned}
p(\boldsymbol{X} = \boldsymbol{x}) &= \prod_{j=1}^{\ell} p(X_j = x_j \mid X_{0:j-1} = x_{0:j-1}) \\
&\stackrel{\text{assumption}}{=} \prod_{j=1}^{\ell} p_{\boldsymbol{\theta}}(X_j = x_j) = \prod_{j=1}^{\ell} \theta_{x_j} \approx \prod_{j=1}^{\ell} \hat{\theta}_{x_j}
\end{aligned}
$$

Maximum likelihood estimate:

$$
\begin{aligned}
\forall v \in \mathcal{V}, \hat{\theta}_v &= \frac{|\{i, j \mid [\boldsymbol{x}_i]_j = v\}|}{N} \\
&= \frac{c_{\boldsymbol{x}_{1:n}}(v)}{N}
\end{aligned}
$$

where $N = \sum_{i=1}^{n} |\boldsymbol{x}_i|$.
Also known as "relative frequency estimation."

Unigram Model

$$p(\boldsymbol{X} = \boldsymbol{x}) = \prod_{j=1}^{\ell} p(X_j = x_j \mid X_{0:j-1} = x_{0:j-1})$$

$$\overset{\text{assumption}}{=} \prod_{j=1}^{\ell} p_{\boldsymbol{\theta}}(X_j = x_j) = \prod_{j=1}^{\ell} \theta_{x_j} \approx \prod_{j=1}^{\ell} \hat{\theta}_{x_j}$$

Maximum likelihood estimate:

$$\forall v \in \mathcal{V}, \hat{\theta}_v = \frac{|\{i, j \mid [\boldsymbol{x}_i]_j = v\}|}{N}$$

$$= \frac{c_{\boldsymbol{x}_{1:n}}(v)}{N}$$

where $N = \sum_{i=1}^{n} |\boldsymbol{x}_i|$.
Also known as "relative frequency estimation."

Unigram Models: Assessment

Pros:

- Easy to understand
- Cheap
- Good enough for information retrieval (maybe)

Cons:

- "Bag of words" assumption is linguistically inaccurate
  - $p(\text{the the the the}) \gg p(\text{I want ice cream})$
- Data sparseness; high variance in the estimator
- "Out of vocabulary" problem

Markov Models $\equiv$ n-gram Models

$$p(\boldsymbol{X} = \boldsymbol{x}) = \prod_{j=1}^{\ell} p(X_j = x_j \mid X_{0:j-1} = x_{0:j-1})$$

$$\stackrel{\text{assumption}}{=} \prod_{j=1}^{\ell} p_{\boldsymbol{\theta}}(X_j = x_j \mid X_{j-\mathsf{n}+1:j-1} = x_{j-\mathsf{n}+1:j-1})$$

$(\mathsf{n} - 1)$th-order Markov assumption $\equiv$ n-gram model

- Unigram model is the $\mathsf{n} = 1$ case
- For a long time, trigram models ($\mathsf{n} = 3$) were widely used
- 5-gram models ($\mathsf{n} = 5$) are not uncommon now in MT

Estimating n-Gram Models

	unigram	bigram	trigram
$p_{\boldsymbol{\theta}}(\boldsymbol{x}) =$	$\prod_{j=1}^{\ell} \theta_{x_j}$	$\prod_{j=1}^{\ell} \theta_{x_j \mid x_{j-1}}$	$\prod_{j=1}^{\ell} \theta_{x_j \mid x_{j-2} x_{j-1}}$
Parameters:	θ_v $\forall v \in \mathcal{V}$	$\theta_{v \mid v'}$ $\forall v \in \mathcal{V}, v' \in \mathcal{V} \cup \{\bigcirc\}$	$\theta_{v \mid v'' v'}$ $\forall v \in \mathcal{V}, v', v'' \in \mathcal{V} \cup \{\bigcirc\}$
MLE:	$\frac{c(v)}{N}$	$\frac{c(v'v)}{\sum_{u \in \mathcal{V}} c(v'u)}$	$\frac{c(v''v'v)}{\sum_{u \in \mathcal{V}} c(v''v'u)}$

General case:

$$\prod_{j=1}^{\ell} \theta_{x_j \mid \boldsymbol{x}_{j-\mathsf{n}+1:j-1}} \qquad \theta_{v \mid \boldsymbol{h}}, \ \forall v \in \mathcal{V}, \boldsymbol{h} \in (\mathcal{V} \cup \{\bigcirc\})^{\mathsf{n}-1} \qquad \frac{c(\boldsymbol{h}v)}{\sum_{u \in \mathcal{V}} c(\boldsymbol{h}u)}$$

The Problem with MLE

- The curse of dimensionality: the number of parameters grows exponentially in n
- Data sparseness: most n-grams will never be observed, even if they are linguistically plausible
- No one actually uses the MLE!

Smoothing

A few years ago, I'd have spent a whole lecture on this! ☹

- Simple method: add $\lambda > 0$ to every count (including zero-counts) before normalizing
- What makes it hard: ensuring that the probabilities over all sequences sum to one
  - Otherwise, perplexity calculations break
- Longstanding champion: modified Kneser-Ney smoothing (Chen and Goodman, 1998)
- Stupid backoff: reasonable, easy solution when you don't care about perplexity (Brants et al., 2007)

Interpolation

If p and q are both language models, then so is

$$\alpha p + (1 - \alpha) q$$

for any $\alpha \in [0, 1]$.

- This idea underlies many smoothing methods
- Often a new model q only beats a reigning champion p when interpolated with it
- How to pick the "hyperparameter" α?

Algorithms To Know

- Score a sentence $\boldsymbol{x}$
- Train from a corpus $\boldsymbol{x}_{1:n}$
- Sample a sentence given $\boldsymbol{\theta}$

n-gram Models: Assessment

Pros:

- Easy to understand
- Cheap (with modern hardware; Lin and Dyer, 2010)
- Good enough for machine translation, speech recognition, ...

Cons:

- Markov assumption is linguistically inaccurate
  - (But not as bad as unigram models!)
- Data sparseness; high variance in the estimator
- "Out of vocabulary" problem

Dealing with Out-of-Vocabulary Terms

- Define a special OOV or "unknown" symbol UNK. Transform some (or all) rare words in the training data to UNK.
  - ☹ You cannot fairly compare two language models that apply different UNK treatments!
- Build a language model at the *character* level.

What's wrong with n-grams?

Data sparseness: most histories and most words will be seen only rarely (if at all).

What's wrong with n-grams?

Data sparseness: most histories and most words will be seen only rarely (if at all).

Next central idea: teach histories and words how to share.

Log-Linear Models: Definitions

We define a conditional log-linear model $p(Y \mid X)$ as:

- $\mathcal{Y}$ is the set of events/outputs (☺ for language modeling, $\mathcal{V}$)
- $\mathcal{X}$ is the set of contexts/inputs (☺ for n-gram language modeling, $\mathcal{V}^{\mathsf{n}-1}$)
- $\boldsymbol{\phi} : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}^d$ is a feature vector function
- $\mathbf{w} \in \mathbb{R}^d$ are the model parameters

$$p_{\mathbf{w}}(Y = y \mid X = x) = \frac{\exp \mathbf{w} \cdot \boldsymbol{\phi}(x, y)}{\sum_{y' \in \mathcal{Y}} \exp \mathbf{w} \cdot \boldsymbol{\phi}(x, y')}$$

References I

Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. Large language models in machine translation. In *Proc. of EMNLP-CoNLL*, 2007.

Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Center for Research in Computing Technology, Harvard University, 1998.

Michael Collins. Log-linear models, MEMMs, and CRFs, 2011. URL `http://www.cs.columbia.edu/~mcollins/crf.pdf`.

Julia Hirschberg and Christopher D. Manning. Advances in natural language processing. *Science*, 349(6245): 261–266, 2015. URL `https://www.sciencemag.org/content/349/6245/261.full`.

Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, second edition, 2008.

Daniel Jurafsky and James H. Martin. N-grams (draft chapter), 2016. URL `https://web.stanford.edu/~jurafsky/slp3/4.pdf`.

Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, third edition, forthcoming. URL `https://web.stanford.edu/~jurafsky/slp3/`.

Jimmy Lin and Chris Dyer. *Data-Intensive Text Processing with MapReduce*. Morgan and Claypool, 2010.

Noah A. Smith. Probabilistic language models 1.0, 2017. URL `http://homes.cs.washington.edu/~nasmith/papers/plm.17.pdf`.