Unsupervised Learning of Linguistic Structure: Morphology

John A. Goldsmith

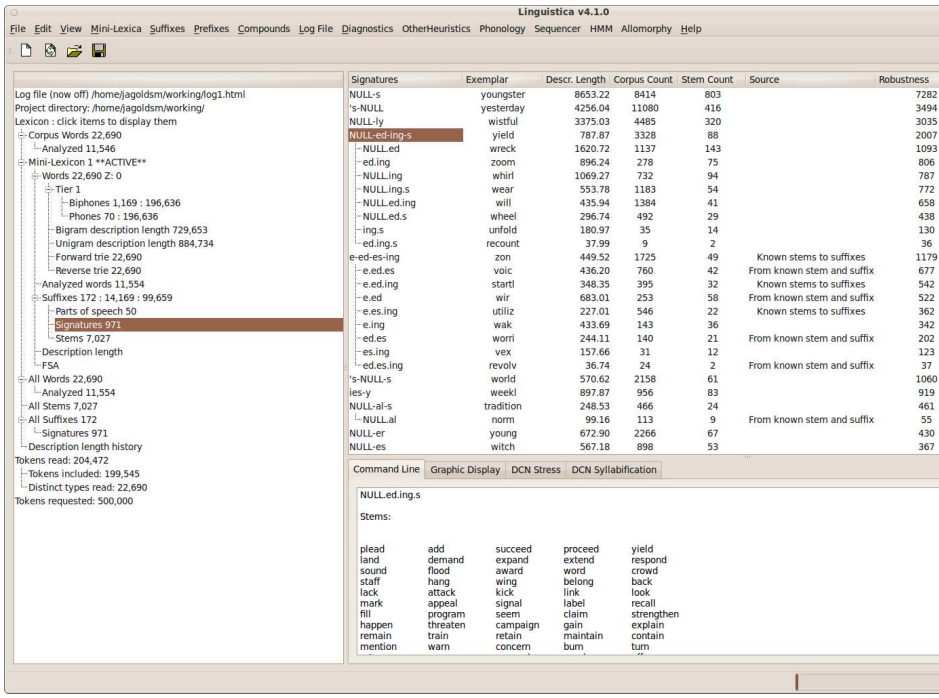
July 8, 2015

Contents

1 Class 1: Overview of information theory and machine learning for linguists

1.1 Course outline

1. Class 1: An introduction to machine learning, probability, and viewing linguistic theory through those lenses.
2. Class 2:
   a) Word discovery (or *word breaking*): a non-probabilistic approach (Sequitur) and an MDL-based (probabilistic) approach.
   b) Morph and morpheme discovery
      i. Zellig Harris and successor frequency. Good and bad points.
      ii. Definition of signatures, and signature-based morphology learning. Linguistica.



3. Class 3: Unsupervised learning of morphology. The challenges:
   a) Layering (or morphological width)
   b) Alignment of signatures
   c) Allomorphy
   d) FSA
   e) Nonconcatenative morphology: Germanic strong verbs; Semitic languages.
4. Class 4: Beyond morphology
   a) Learning the syntactic coherence of past tense and past participles in English automatically.
   b) The general problem of the induction of PoS.
   c) MDL as the basis of linguistic theory.

1.2 Class 1 overview

1. Overview of 2 week course (10 minutes)
2. Unsupervised learning: what it means for linguistics (5 minutes)
   a) 3 kinds of knowledge: native speaker, linguist, theory.

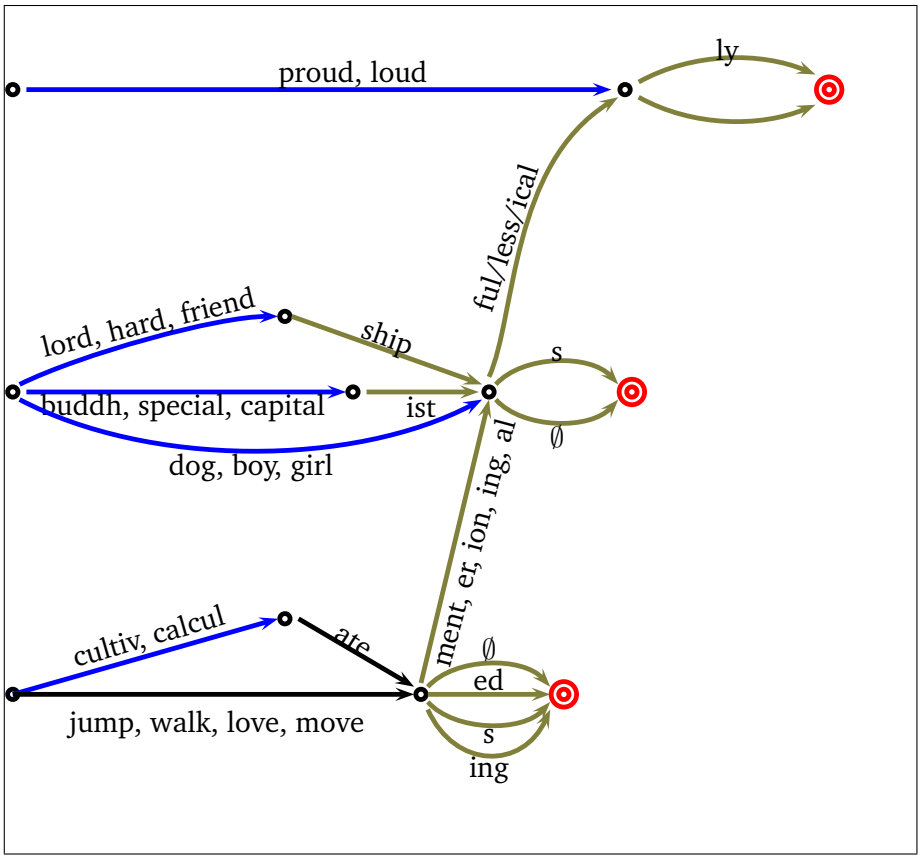


Fig. 1.1: English morphology: morphemes on edges

b) The inadequacy of the claim of innateness as an answer to the problem of language learning. We learn things by trying to build a learner: the biggest help comes from ideas about complexity, not "silver bullet" answers.

c) Four kinds of structure to discover:

i. Sequential structure

ii. Chunking structure (morphemes, words, phrases)

iii. Categorization

iv. Hierarchical structure

3. Big questions (8 minutes)

a) Chomsky's 3 notions of linguistic theory (1957): we opt for the first one.

b) Machine learning: what is it?

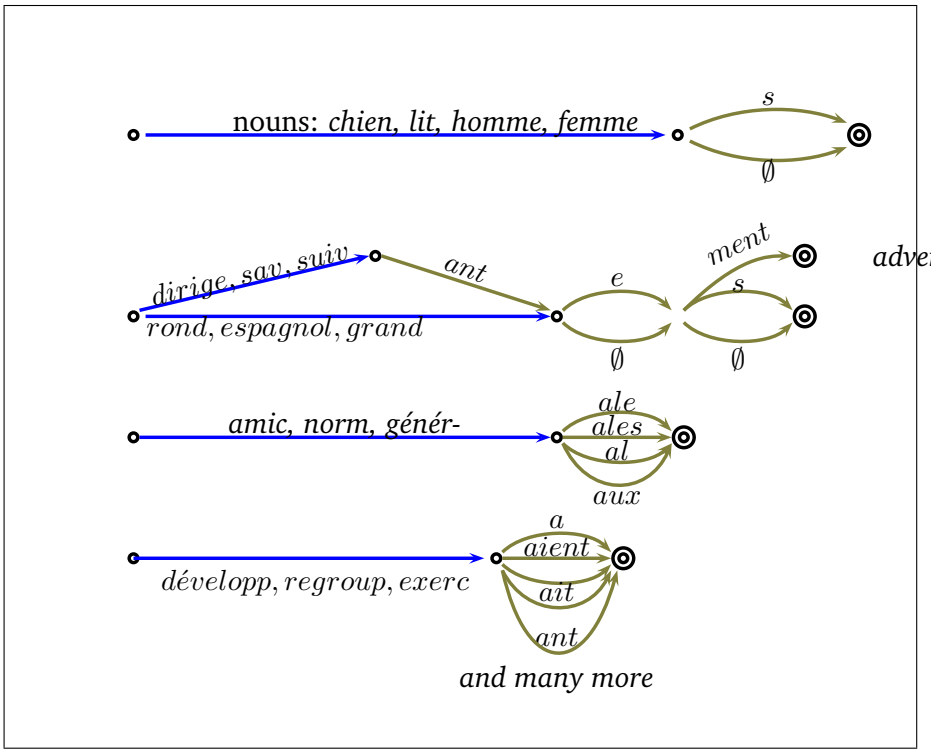


Fig. 1.2: French

c) The general structure of a hill-climbing learner
d) Generative grammar and machine learning: complementary goals
e) Question of innate knowledge (picture of big and little information innately)

4. Probability and information theory (5 minutes)
   a) probability in the abstract (5 minutes)
   b) probability of events (20 minutes)
   c) encoding of events (10 minutes)
   d) encoding of grammars (10 minutes)
5. What is next (10 minutes)

1.3 Unsupervised learning and other sorts: what it means for linguistics

1.3.1 Machine learning: what is it?

Machine learning: This term arose in the mid-1980s in the context of computer science. It is the study of how to infer generalizations from large amounts of data, and involves methods and concepts from mathematics, statistics, and computer science.

Learning: supervised learning, unsupervised learning; learning with unlabeled examples.

1.3.2 3 kinds of knowledge: native speaker, linguist, theory

The knowledge of the speaker accounts for his/her ability to speak and understanding.

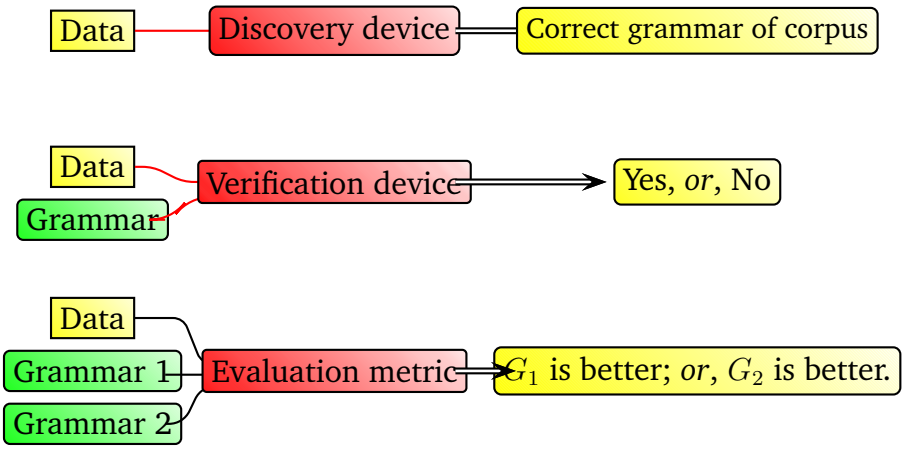
The knowledge of the linguist offers an *explanation* of that ability.

The knowledge of linguistic theory offers an *explanation* and *justification* for why the linguist says what she does (and not something else).

The educated non-linguist listens to a linguist talking about Hungarian vowel harmony and asks: why are they going on about these exceptions? I know how to speak my language: I am rarely, if ever, in doubt—something that I can't say about these linguist types.

The linguist listens to discussions of linguistic theory (of the sort that we will engage in here!) and says, I know what the morphemes are in Swahili: why are they going on and on about the methods to determine what the morphemes are? I am rarely, if ever, in doubt—something that I can't say about these unsupervised learning sorts.

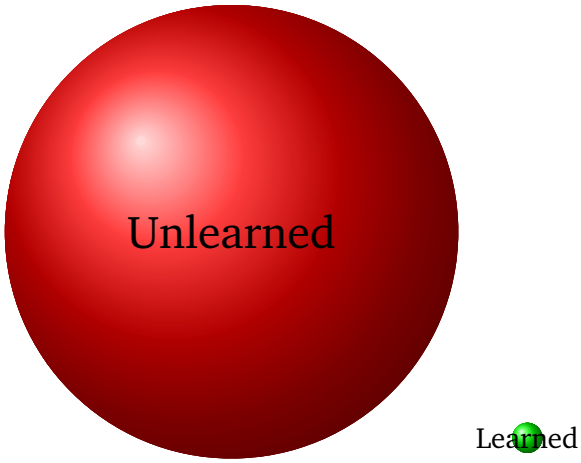
1.3.3 Chomsky's 3 notions of linguistic theory (1957)



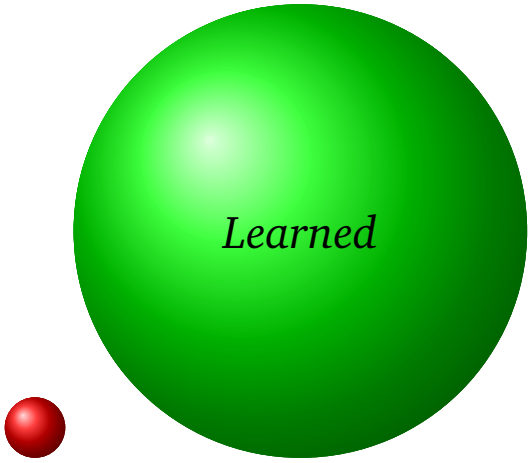
We opt for the first one; generative grammar opts for the third.

1.3.4 Innateness

The inadequacy of the claim of innateness as an answer to the problem of language learning. We learn things by trying to build a learner: the biggest help comes from ideas about complexity, not "silver bullet" answers.



What is the proper balance of the Learned and the Unlearned? That's the big question.



We can learn what is useful to know, as we pointed out above.

The deepest level at which we need to understand this work is the way it approaches the question of knowledge and learning of language. We are at a point when it is no longer news to say that it is not obvious how language can be learned. What we know now is that it is no longer possible to respond to that by saying that what is not learned must therefore be innate. What we are looking at, in this work, is knowledge that is hard to learn, and just as hard to have innately. Neither approach is a solution. Something new and different needs to be said: we need a deep learner, a smart learner, and we need to figure out its architecture. How will machine learning help us? That is the timely question. I hope to offer some tentative suggestions by looking at the acquisition of morphology.

1.3.5 Summary

The kind of linguistics I will be describing is close to Chomsky's original notion of generative grammar, but it is distinct from it in three ways.

1. It aims to provide mechanisms (algorithms) to project grammars from data.

2. It sees the evaluation task as composed of two equally important terms: grammar simplicity and fit of grammar to data. Generative grammar recognized(s) only the first.

3. It exploits the notion of computational complexity when it can. Generative grammar was based on philosophical naturalism and psychologism, grounding the deepest explanations in the discoveries of science and psychology.

1.3.6 Four kinds of structure to discover:

1. Sequential structure
2. Chunking structure (morphemes, words, phrases)

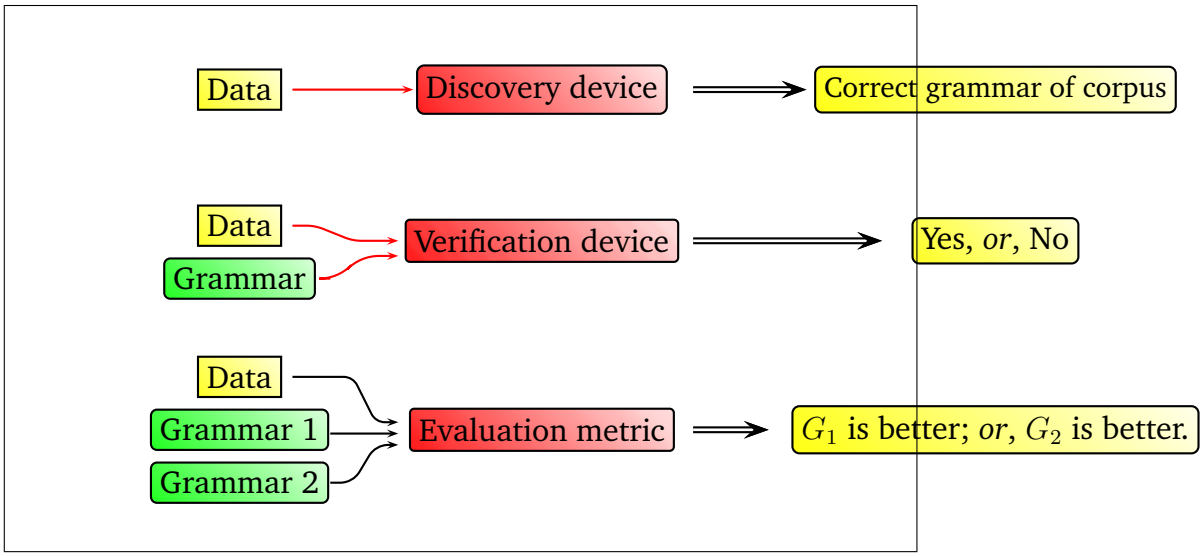


Fig. 1.3: Chomsky's three conceptions of linguistic theory

3. Categorization
4. Hierarchical structure

1.3.7 The general structure of a hill-climbing learner

1.3.8 Generative grammar and machine learning: complementary goals

The **generativist** focuses, for now, on determining the hypothesis space ('class') that includes all human languages; excluding humanly impossible languages is important too, but not as important as making sure the hypothesis space is large enough to include at least one grammar rich and complex enough to model each attested language.

The **information theoretic linguist** focuses on enriching the hypothesis space *much* more slowly, and puts primary focus on methods to *find* the particular grammar within that hypothesis space that *optimizes some reasonable measure*. What is a reasonable measure? It could be *probability* (which we would maximize), or better yet, *description*

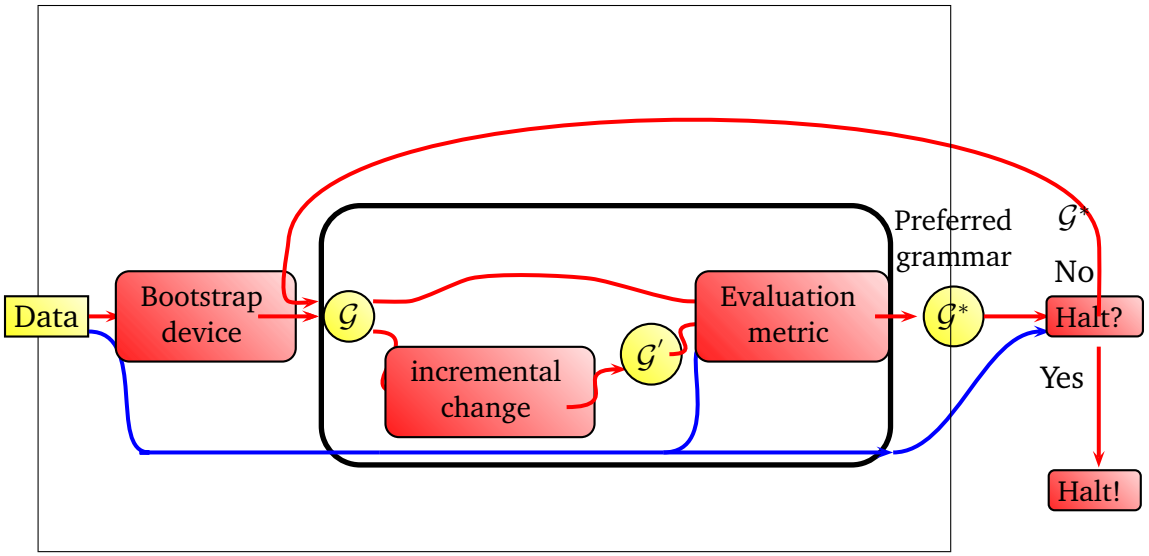


Fig. 1.4: Unsupervised learning of grammars

length (which we would minimize, because in some respects it is inverted with respect to probability).

Given data $\mathcal{D}$, find $g = \arg\max_{g \in \mathcal{G}} p_g(\mathcal{D})$.

Given data $\mathcal{D}$, find $g = \arg\max_{g \in \mathcal{G}} [p_g(\mathcal{D}) - cost(g)]$.

These are two very different goals! And a person could perfectly well want to work on both problems.

Very important: The most important reason that we develop probabilistic models is to evaluate and compare different grammars. (It is not in order to assign probabilities to data that already exists, or that does not exist yet. We are not rolling dice.)

Importance in computational sphere of quantitative measurement of success. There is little emphasis on evaluating a model based on its fit to a scientist's intuition.

Probability is the quantitative theory of evidence.

1.4 Probability and information theory

1.4.1 probability of events

(20 minutes)

slides: Probability for linguists.

Probability for linguists

John A Goldsmith

July 6, 2015

Overall strategy

1. probabilities and distributions
2. unigram probability
3. a word about *parametric* distributions
4. -1 $\times log_2$ probability (or *plog*: positive log probability)
5. *bigram* probability: conditional probability
6. *mutual information*: the log of the ratio of the observed to the "expected"
7. average plog $\rightarrow$ *entropy*
8. encoding events: compression, optimal compression, and cross-entropy
9. encoding grammars optimally

A distribution

Big point 1

A distribution is a list of numbers that are not negative and that sum to 1.

$$\sum_i p_i = 1$$

$$p_i \geq 0$$

A probabilistic grammar

- A probabilistic model, or grammar, is a universe of possibilities ("sample space") + a distribution.
- A probabilistic grammar is a distribution over all strings of the IPA alphabet.
- It is not a formalism stating which strings are *in* and which are *out*.

The purpose of a probabilistic model

Big point 2

The purpose of a probabilistic model is to test the model against the data.

- Suppose we have some well-chosen data D. Then the best grammar is the one that assigns the highest probability to D, all other things being equal.
- The goal is not to test the data!
- Therefore: all grammars must be probabilistic, so they can be tested and evaluated.

Probability

- The *quantitative theory of evidence*.
- If we have *variable* data, then probability is the best model to use.
- If we have *categorical* (not variable) data, probability is still the best model to use.

Probabilities and frequencies

Probabilities and frequencies are not the same thing.

- Frequencies are *observed*.
- Probabilities are values in a system that a human being creates and *assigns*.
- We can choose to assign probabilities as the observed frequencies—buy that is not always a good idea.
- This is a good idea only so long as we don't need to handle yet-unseen (never before seen) data.
- In many cases, this choice maximizes the probability of the data.
- They both deal with *distributions* (i.e., the observed frequencies and the probability distributions of a model).

Probabilities and frequencies

Probabilities and frequencies are not the same thing.

- *Counts* are counts: the number of things or events that fall in some category.
- *Frequency* is ambiguous: it either means count (less often) or it means *relative frequency*: a ratio between a count of something and the total number of things that fall within the larger category.
- There are 63,147 occurrences of *the* in the Brown Corpus, out of 1,017,904; 6.2% of the words in the Brown Corpus are *the*.

English, French, Spanish

Let's take a look at some languages.
And for starters, let's just look at *unigram* frequencies: the frequencies at which items appear, not conditioned by the environment.
`people.cs.uchicago.edu/jagoldsm/course/class1`

Plogs

- We will assign probabilities to every outcome we consider.
- Each of these is typically quite small.
- We therefore use a slightly different way of talking about small numbers: plogs.

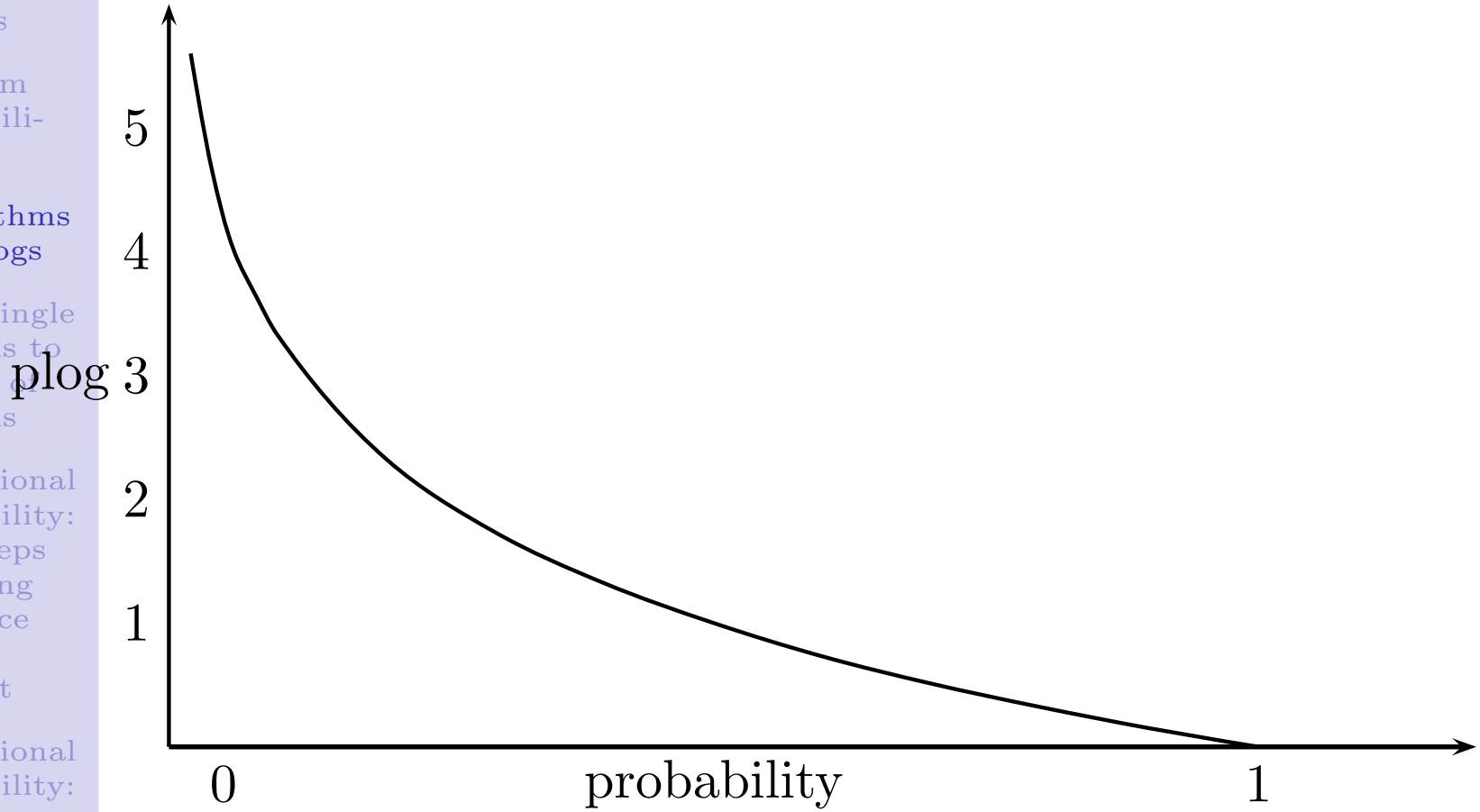
Inverse log probabilities, or *plogs*

A way to describe small numbers... upside down.

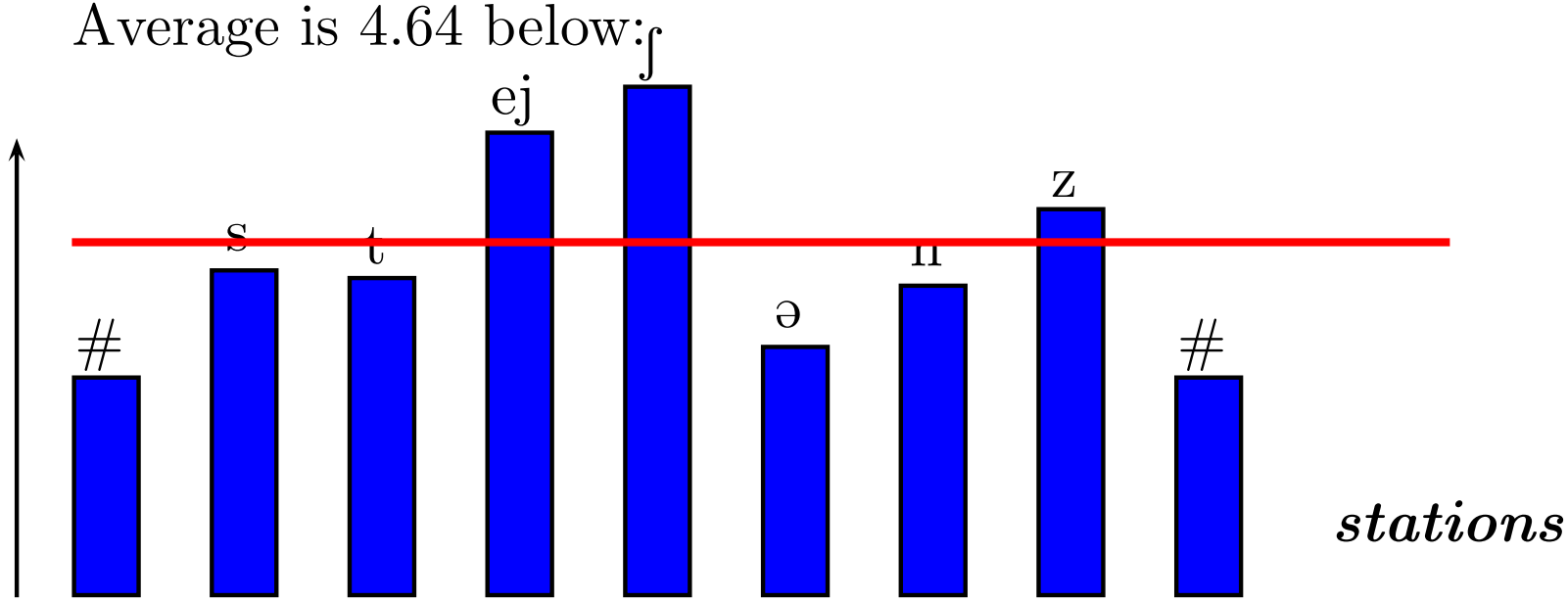
A probability	its plog
0.5	1
0.25	2
0.128	3
$\frac{1}{16}$	4
$\frac{1}{32}$	5
$\frac{1}{1024}$	10
...	...
$\frac{1}{1{,}000{,}000}$	almost 20

- The *bigger* the plog, the *smaller* the probability.
- It's a bit like a measure of markedness, if you think of more marked things as being less frequent.
- $plog(x) = -log_2(x) = log_2(\frac{1}{x})$

Plogs



Average is 4.64 below:



This diagram from a visually interactive program displaying phonological complexity at:

`http://hum.uchicago.edu/~jagoldsm/PhonologicalComplex`

Most and least frequent phonemes in English

rank	phoneme	frequency	plog
1	#	0.20	2.30
2	ə	0.066	3.92
3	n	0.058	4.10
4	t	0.056	4.17
5	s	0.041	4.61
6	r	0.040	4.76
7	d	0.037	4.85
8	l	0.035	4.94
9	k	0.026	5.27
10	ǽ	0.025	5.31
45	ɔ́y	0.000 78	10.32
46	ǎe	0.000 69	10.50
47	ž	0.000 54	10.84
48	aǎy	0.000 38	11.36
49	ă	0.000 36	11.42

average plogs

rank	orthography	phonemes	$av.plog_1$
1	a	ə	3.11
2	an	ən	3.44
3	to	tə	3.47
4	and	ənd	3.80
5	eh	ɛ́	3.88
6	the	ə	3.88
7	can	kən	3.90
8	an	ǽn	3.91
9	Ann	ǽn	3.91
10	in	ín	3.91

Worst words in English

rank	orthography	phonemes	$av.plog_1$
63,195	bourgeois	bʌ̆ržwá	7.21
63,196	Ceausescu	čʊ̆čɛ́skŭ	7.21
63,197	Peugeot	p yŭžó	7.22
63,198	Giraud	žaў̆ró	7.24
63,199	Godoy	gádoў̆	7.27
63,200	geoid	ǰǐɔў̆d	7.40
63,201	Cesare	čĕzárĕ	7.40
63,202	Thurgood	θɚ́gʌ̆d	7.47
63,203	Chenoweth	čɛ́nʊ̆wɛ̆θ	7.49
63,204	Qureshey	kəréšĕ	7.54

Word counts and frequencies

	word	count	frequency	plog
1	the	69903	0.068271	3.87
2	of	36341	0.035493	4.81
3	and	28772	0.028100	5.15
4	to	26113	0.025503	5.29
5	a	23309	0.022765	5.46
6	in	21304	0.020807	5.59
7	that	10780	0.010528	6.57
8	is	10100	0.009864	6.66
9	was	9814	0.009585	6.70
10	he	9799	0.009570	6.70
11	for	9472	0.009251	6.77
12	it	9082	0.008870	6.82
13	with	7277	0.007107	7.14
14	as	7244	0.007075	7.14
15	his	6992	0.006829	7.19

Unigram model

- The probability of a string S, of length L, is $\lambda(L)$ times the probability of each of the symbols.
- $p_U(S) = \lambda(L) \times \prod_i S[i]$
- If we sum over *all* strings of a given length l, the sum of their probabilities is $\lambda(l)$. That's just math.
- This is the model that takes no information about ordering into account.
- Because plogs are additive, it makes sense to ask what the average plog of a word is. In the unigram model, they describe an extensive property.

Conditional probabilty

- p(A, given B)
- p($A|B$)
- $\frac{p(A\,and\,B)}{p(B)}$
- p(A's name is "John") < p(A's name is "John" given that A is male and American)
- p(A=Queen of hearts)
- p(A=Queen of hearts | A is a red card)

Conditional probability in a string

- p(S[i]=h given that S[i-1]=t)
- p(S[i]=h | S[i-1]=t)
- p (S[i]=book | S[i-1] = the) > p(S[i]=book)
- p (S[i]=the | S[i+1]=book) > p (S[i]=book)
- These are not statements of *causality*.

Addition is easier to understand than multiplication

- In the unigram model, the probabilility of the string = product of the probabilities of its symbols.[1]
- If we use plogs, the log probability of the string is the sum of the plogs of its symbols.

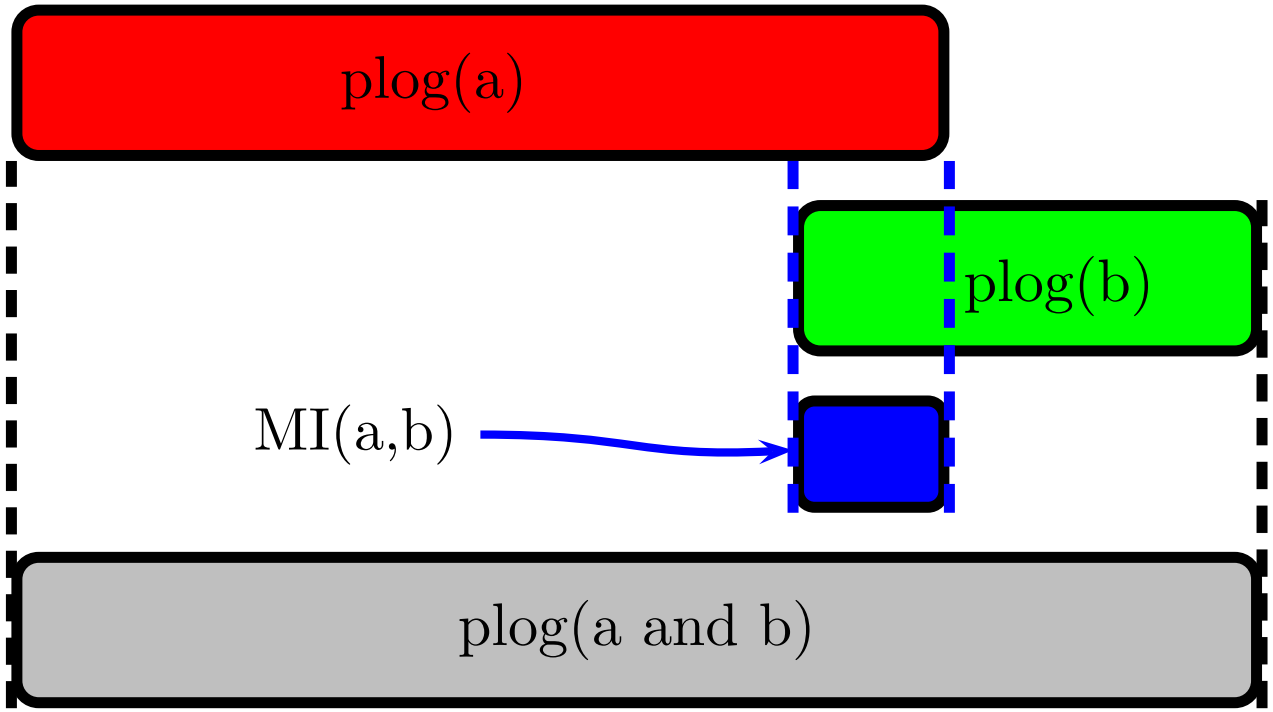
[1] ignoring length of string. . .

Using plogs with conditional probability

- The probability goes up when we use a better model (i.e., one that encodes more knowledge about the system) that takes into consideration the factors in the neighborhood that helped lead to the events we saw.
- The bigram conditional probability is usually greater than the unigram probability in real data.
- The difference between the bigram plog and the unigram plog is called the *mutual information* (MI).

$$log \frac{p(AandB)}{p(A)p(B)} = log \frac{p(AandB)}{p(A)} \frac{1}{p(B)} = log\, p(B|A) - log\, p(B)$$

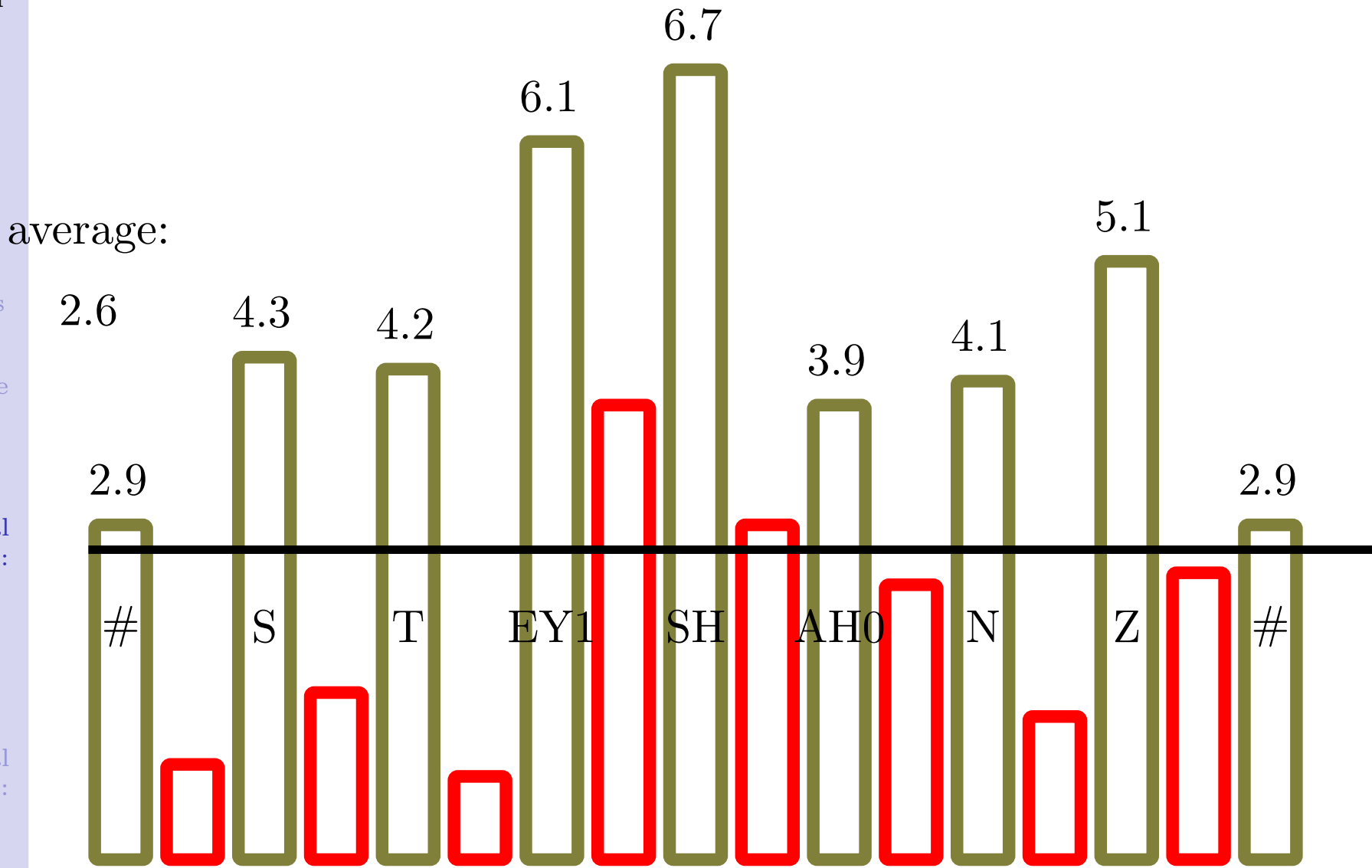
Pointwise mutual information (MI)



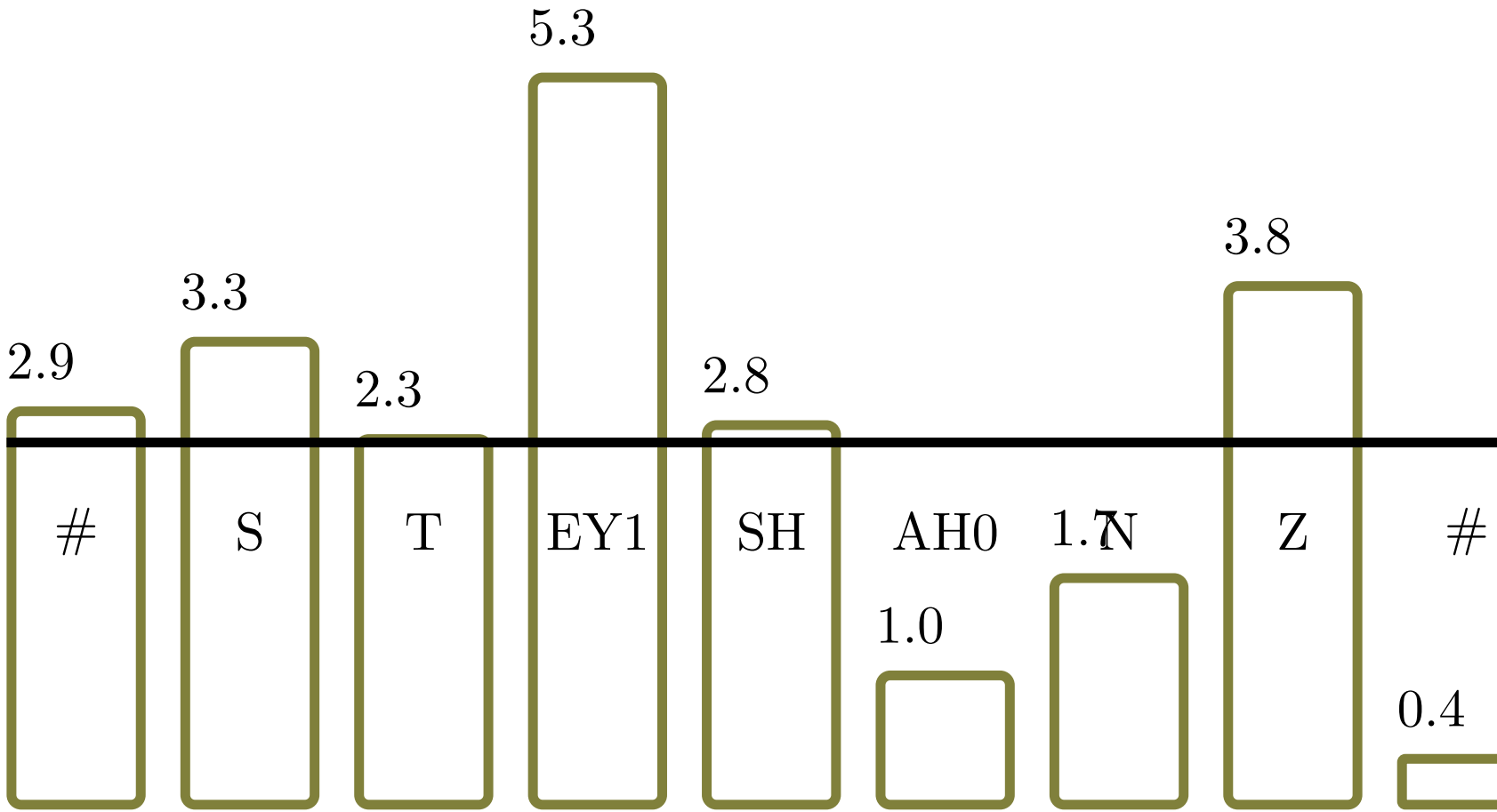
A reminder about events, and "a & b"

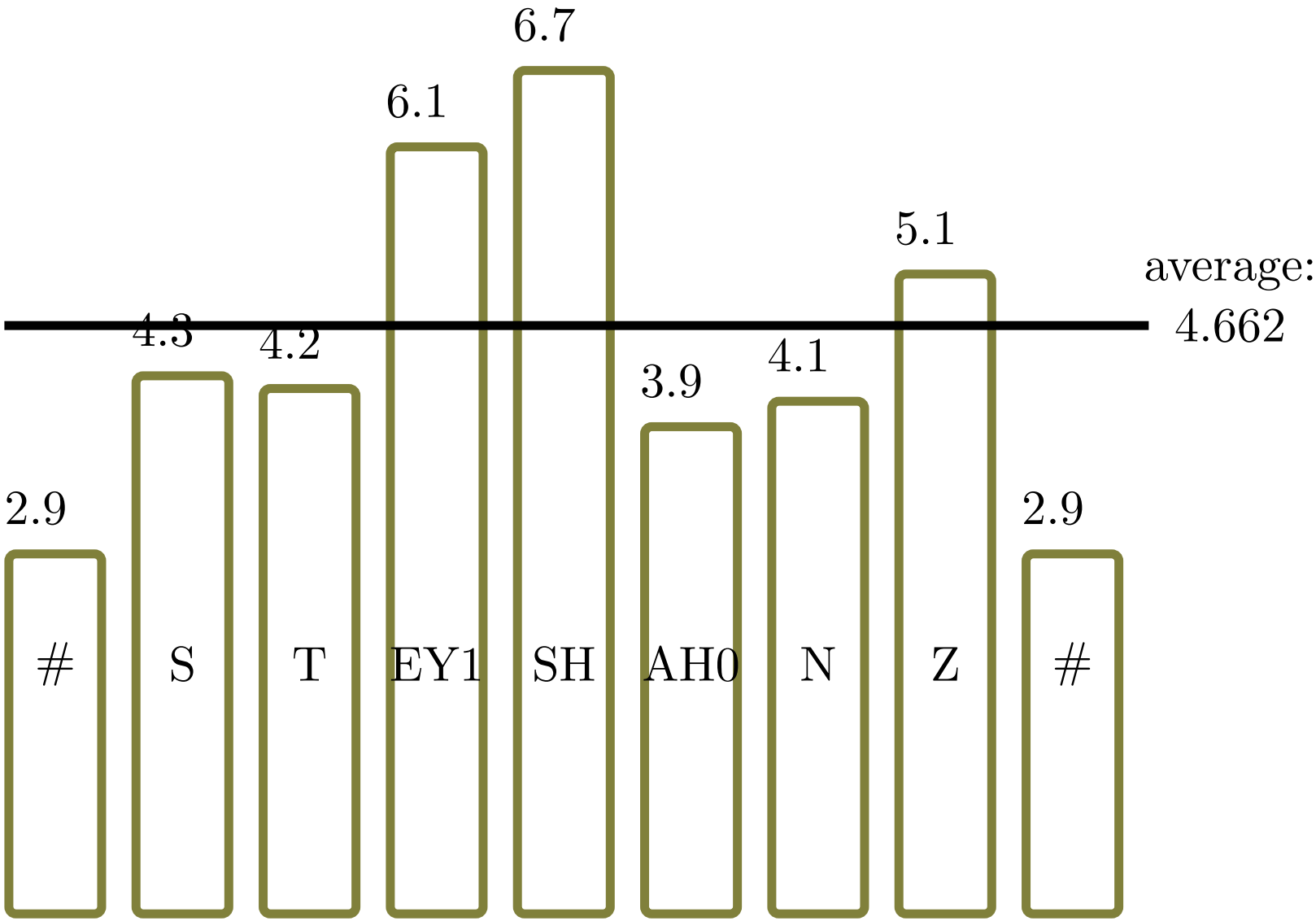
- There is no implicit statement about location of the events when we write "a & b".
- p(W[i] = "of" & W[i+1]="the")
- p(W[i] = "of" & W[i+5] = "the")
- If we look at the second, the MI will be very close to zero.

Unigram model with MI



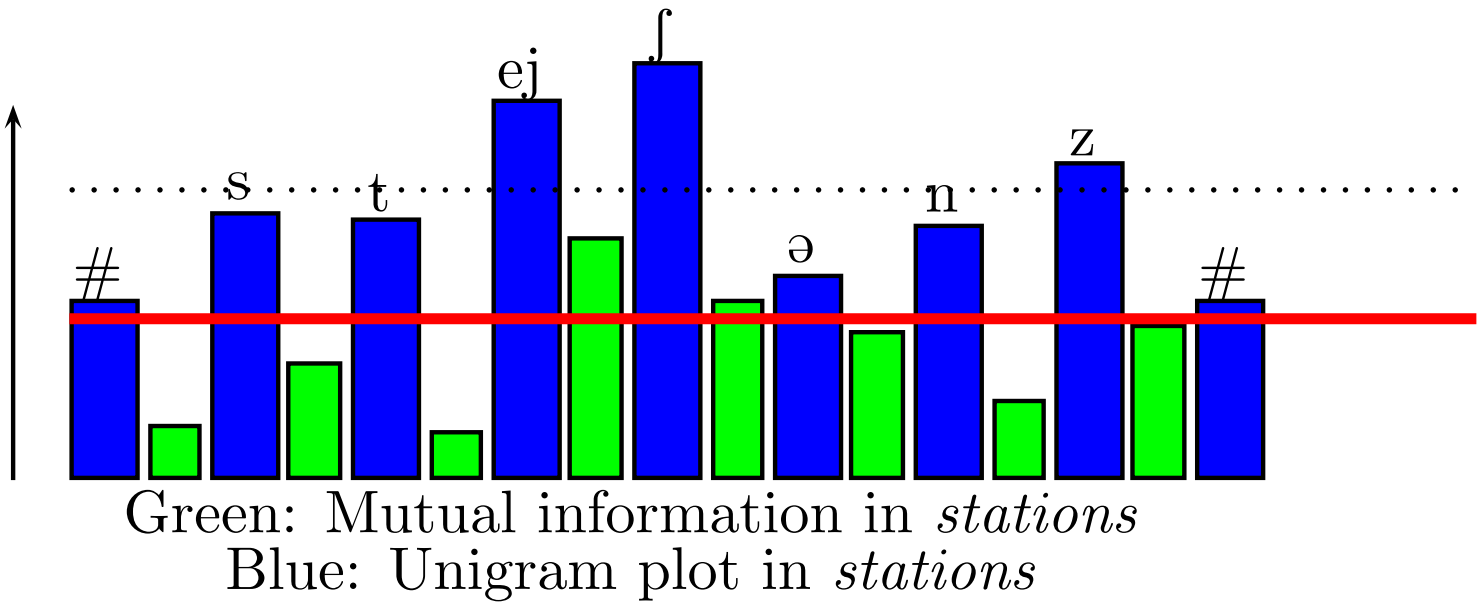
Bigram model




6.7
6.1
5.1
average:
4.662
4.3
4.2
3.9
4.1
2.9
2.9
#
S
T
EY1
SH
AH0
N
Z
#

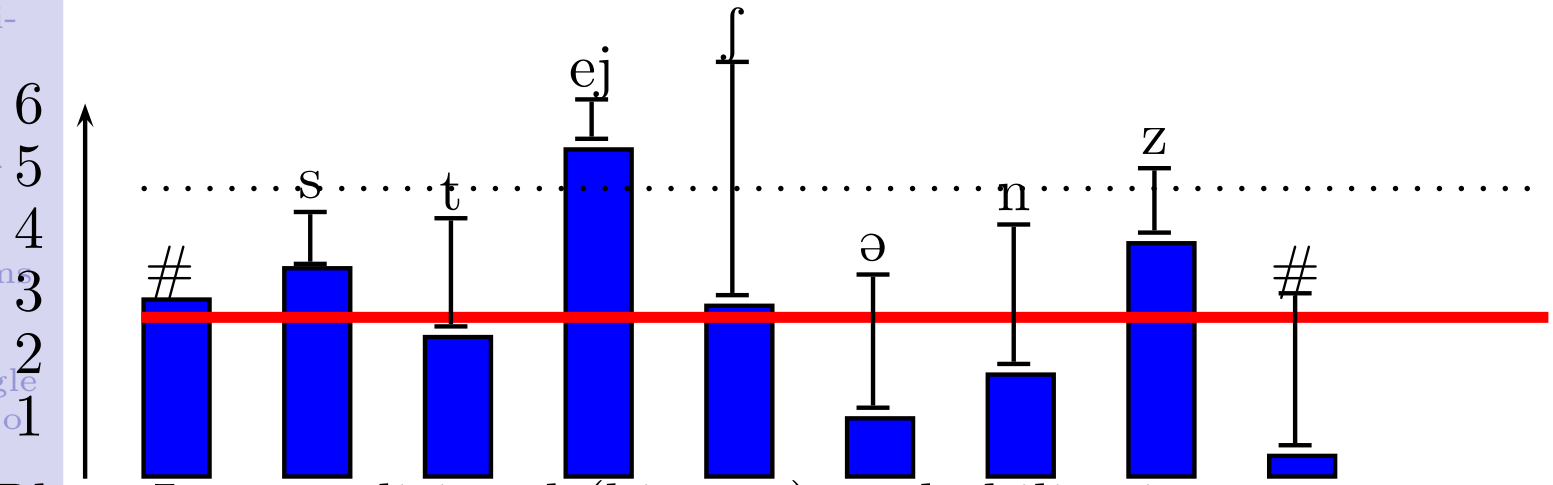
- $p_U = \prod p(S[i])$
- $= p_U(\text{thecatisonthemat})$
- $= \text{p}_U(t) \times p_U(h) \times p_U(e) \times p_U(c) \ldots \times p_U(t)$
- $= p_U(a) \times p_U(a) \times p_U(c) \times p_U(e) \times p_U(e) \ldots \times p_U(t)$
- $=(\text{p}_U(a))^2 \times p_U(c) \times (p_U(e))^2 \times (p_U(e))^2 \ldots \times p_U(t)$
- $=\prod_{\text{l in alphabet A}} p(a)^{\text{count of l in string}}$

Average below is 2.58 (down from 4.64)



Green: Mutual information in *stations*

Blue: Unigram plot in *stations*

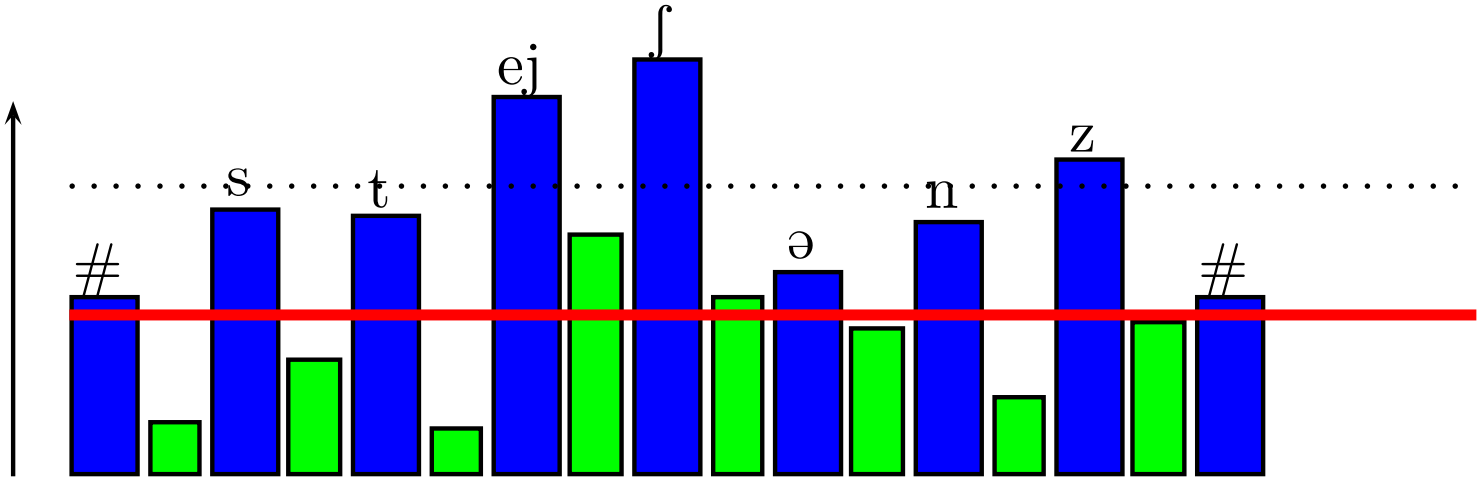


Blue: Log conditional (bigram) probability in *stations*

Decrease from unigram model is exactly the mutual information

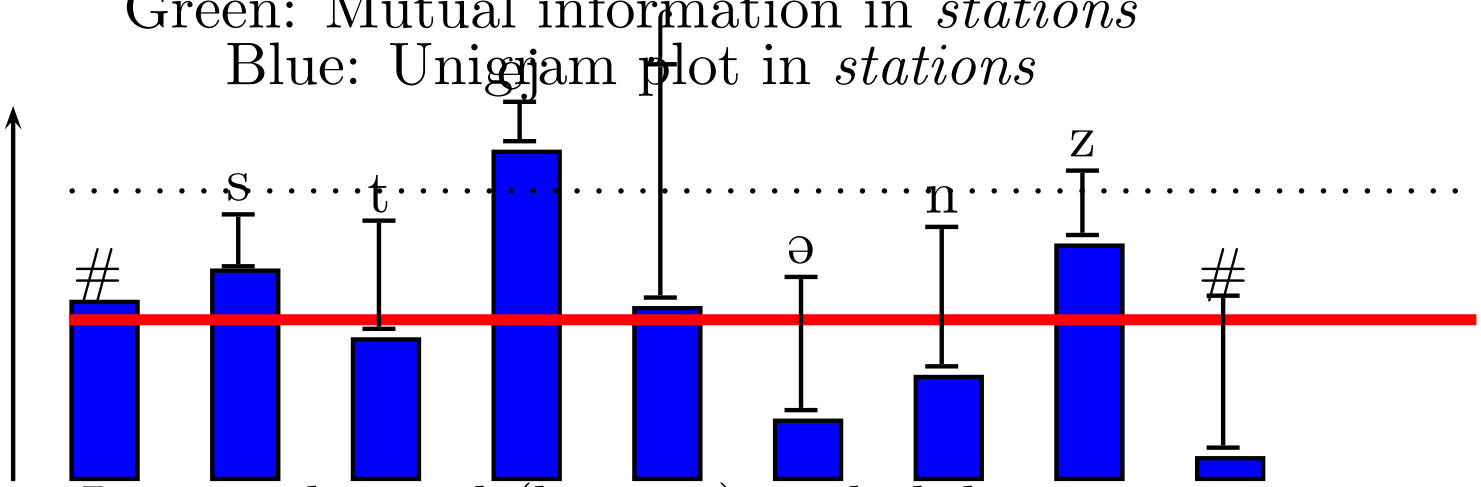
Conditional probability: first steps in taking sequence into account

Average below is 2.58 (down from 4.64)



Green: Mutual information in *stations*

Blue: Unigram plot in *stations*

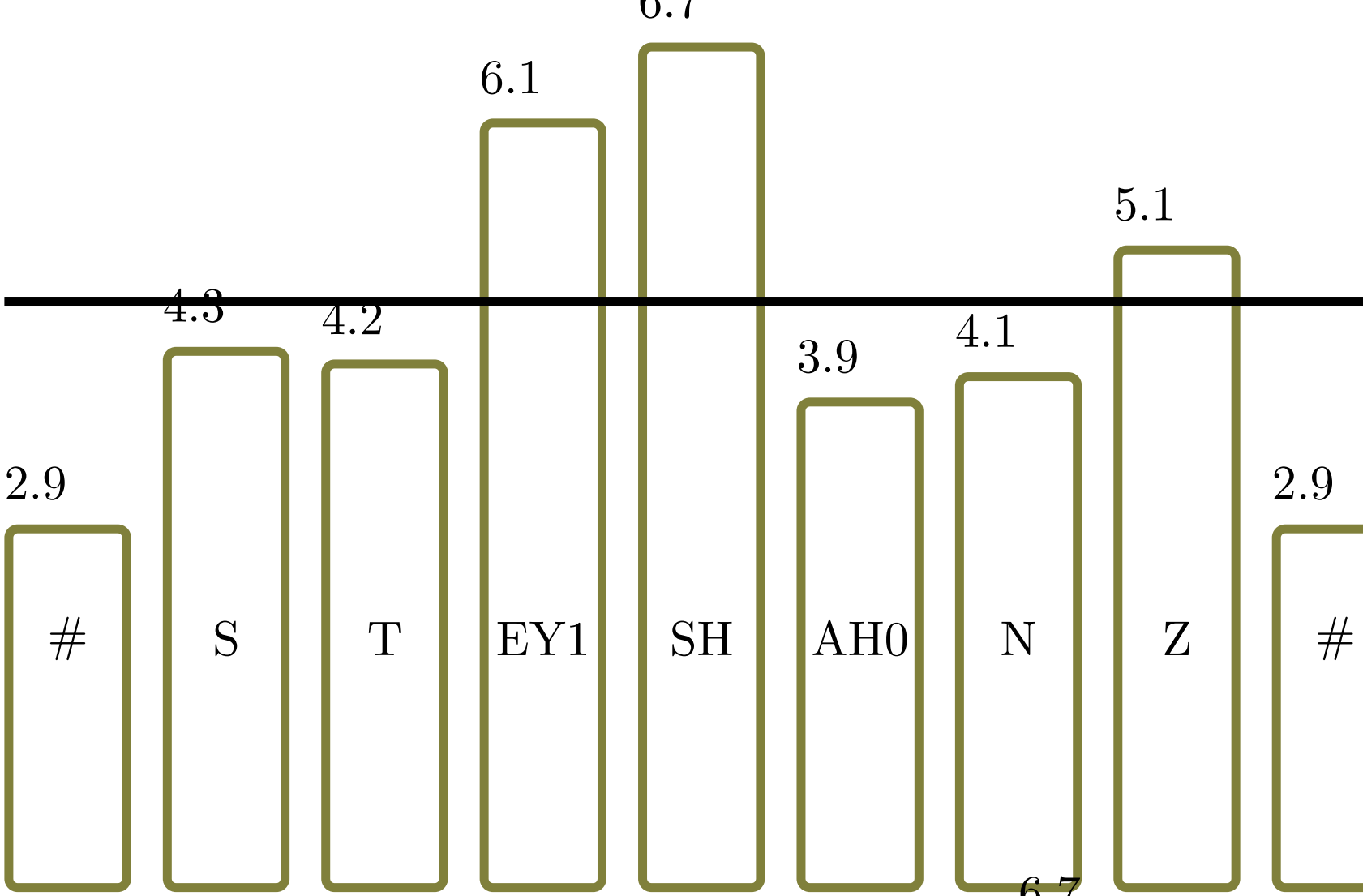


Blue: Log conditional (bigram) probability in *stations*

Decrease from unigram model is exactly the mutual information

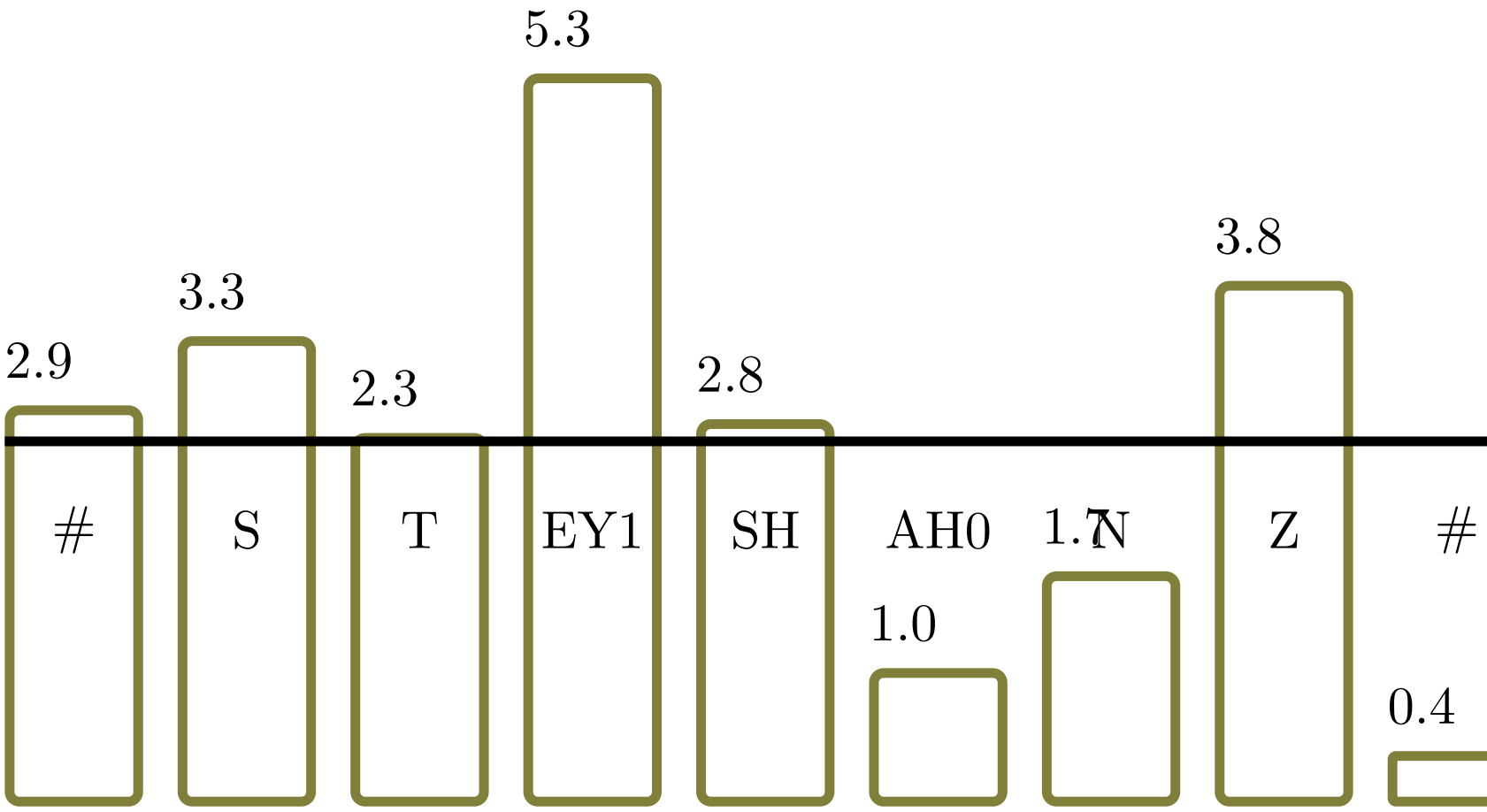
Using plogs with conditional probability

- We saw that the probability goes up when we use a better model that takes into consideration the factors in the neighborhood that helped lead to the events we saw.
- The bigram conditional probability is usually greater than the unigram probability in real data.
- The difference between the bigram plog and the unigram plog is called the *mutual information* (MI).

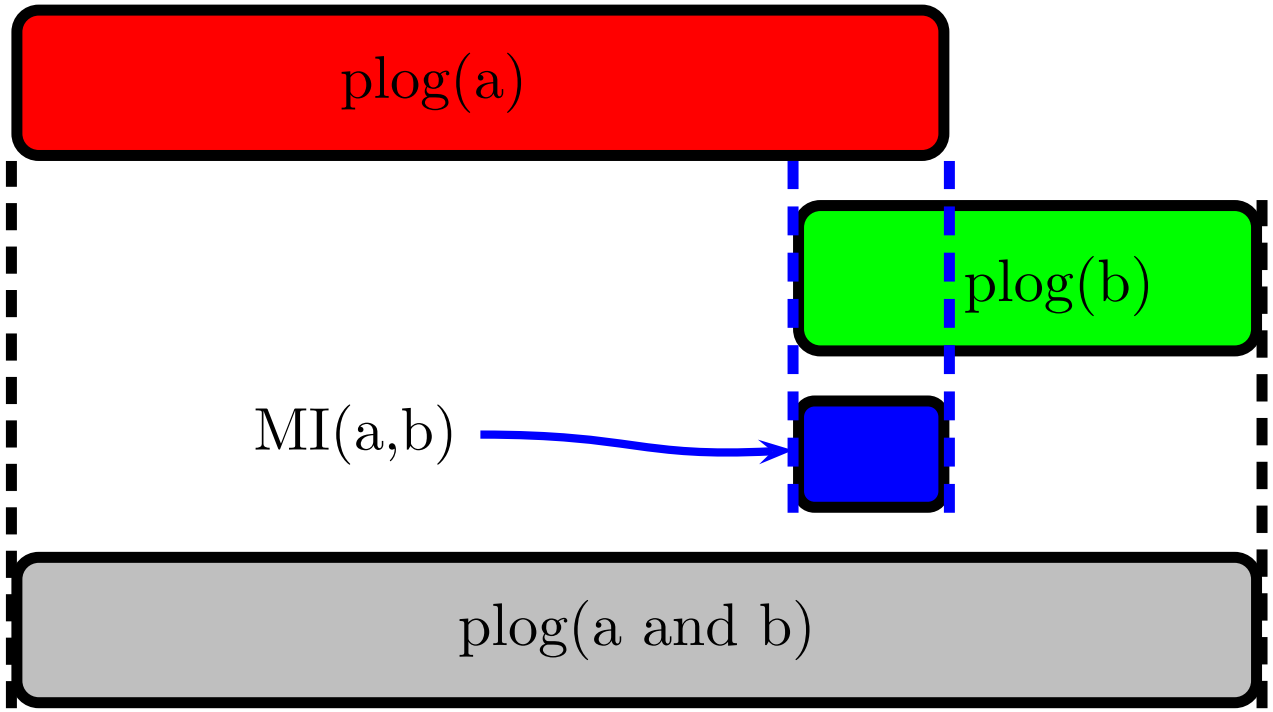


average:

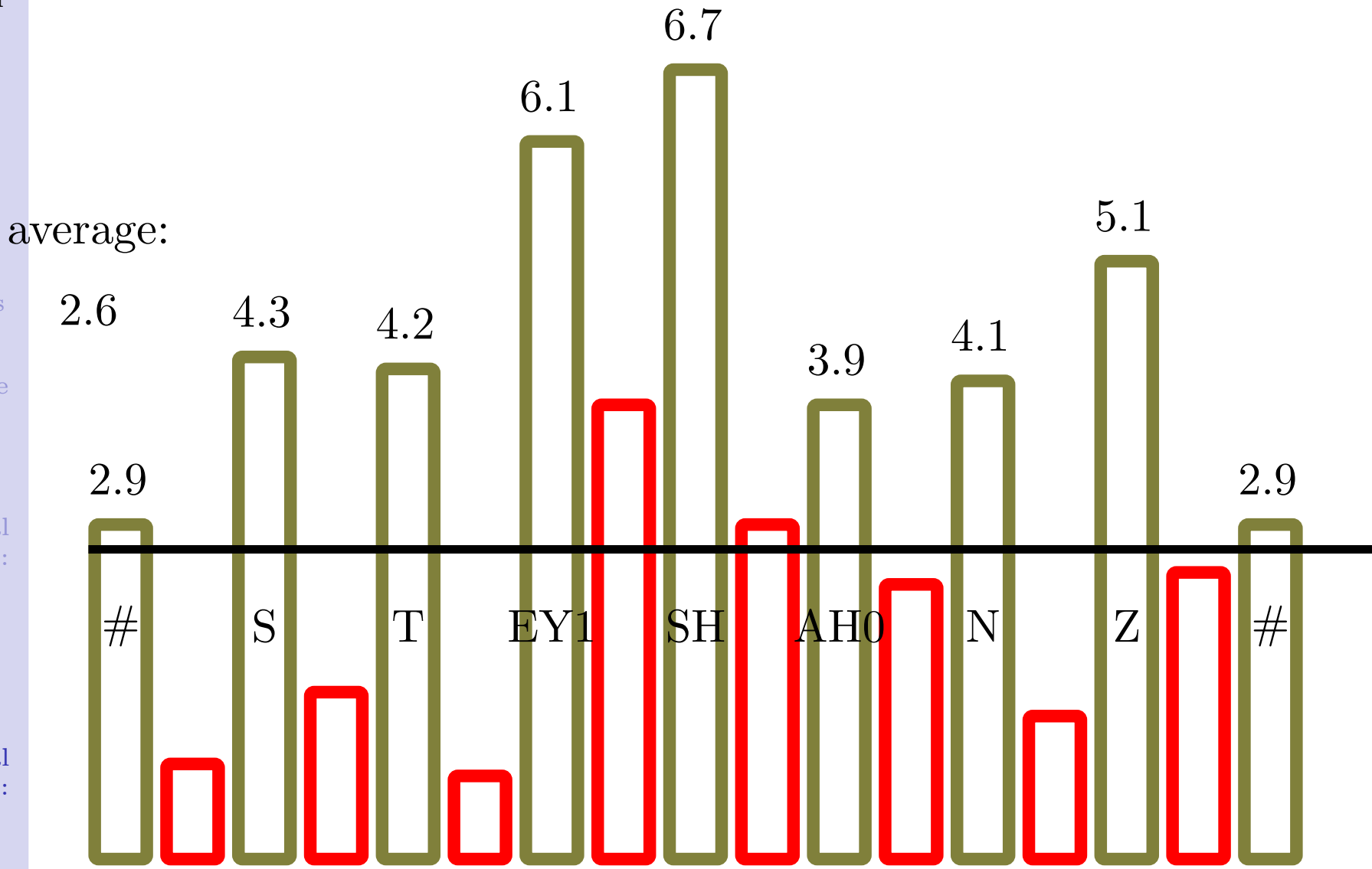
2.6 4.3 4.2 6.1 6.7 3.9 4.1 5.1 2.9


2.9
3.3
2.3
5.3
2.8
1.0
1.7
3.8
0.4
#
S
T
EY1
SH
AH0
N
Z
#

Pointwise mutual information (MI)



Unigram model with MI



Word counts and frequencies: repeated

	word	count	frequency	plog
1	the	69903	0.068 271	3.87
2	of	36341	0.035 493	4.81
3	and	28772	0.028 100	5.15
4	to	26113	0.025 503	5.29
5	a	23309	0.022 765	5.46
6	in	21304	0.020 807	5.59
7	that	10780	0.010 528	6.57
8	is	10100	0.009 864	6.66
9	was	9814	0.009 585	6.70
10	he	9799	0.009 570	6.70
11	for	9472	0.009 251	6.77
12	it	9082	0.008 870	6.82
13	with	7277	0.007 107	7.14
14	as	7244	0.007 075	7.14
15	his	6992	0.006 829	7.19

Top of the Brown Corpus for words following *the*

	word	count	count / 69,936
0	first	664	0.009 49
1	same	629	0.008 99
2	other	419	0.005 99
3	most	419	0.005 99
4	new	398	0.005 69
5	world	393	0.005 62
6	united	385	0.005 51
7	state	271	0.004 18
8	two	267	0.003 82
9	only	260	0.003 72
10	time	250	0.003 57
11	way	239	0.003 42
12	old	234	0.003 35
13	last	223	0.003 19
14	house	216	0.003 09

Top of the Brown Corpus for words following *of*.

	word	count	count / 36,388
1	the	9724	0.267
2	a	1473	0.040 5
3	his	810	0.022 3
4	this	553	0.015 20
5	their	342	0.009 40
6	course	324	0.008 90
7	these	306	0.008 41
8	them	292	0.008 02
9	an	276	0.007 58
10	all	256	0.007 04
11	her	252	0.006 93
12	our	251	0.006 90
13	its	229	0.006 29
14	it	205	0.005 63
15	that	156	0.004 29

Cross entropy: where we keep the empirical frequencies, but vary the distribution whose plog we use to compute the entropy. This is the "cross-entropy" of one distribution to the other (but not symmetrical!). Entropy, or self-entropy, is always smaller than cross-entropy.

$$\sum_x p(x) ln \frac{q(x)}{p(x)} \leq \sum_x p(x)(1 - \frac{q(x)}{p(x)}) \quad (1)$$

Why? Look at the plot of $ln(x)$, and compute its first and second derivatives, and its value at (1,0).

$$= \sum_x p(x) - \sum_x p(x) \frac{q(x)}{p(x)} = 1 - 1 = 0. \quad (2)$$

So $\sum_x p(x) ln(\frac{q(x)}{p(x)} \leq 0$, which is to say, the cross-entropy always exceeds the entropy that isn't cross, when we use natural logs as our base.

But we can maintain the inequality when we switch to base 2 logs (which is what we use with plogs), since it just amounts to multiplying both sides by a constant. First we get:

$$\sum_x p(x) ln\, q(x) \leq \sum_x p(x) ln\, p(x) \tag{3}$$

and then we multiply by -1:

$$\sum_x p(x) plog p(x) \leq \sum_x p(x) plog\, q(x) \tag{4}$$

The Kullback-Leibler divergence $D_{KL}(p, q)$ is defined as KL divergence

$$\sum_x p(x)\, ln\, \frac{p(x)}{q(x)} \tag{5}$$

You see that it's the difference between the cross-entropy and the self-entropy—pay careful attention to the *absence* of a minus before the sum.

$$\prod_{i=1}^{i=len(string)} S[i] = \prod_{l \in lexicon} l^{count_S(l)}. \quad (6)$$

$$logprob(S) = \sum_{lexicon} count_S(l) logprob(l). \quad (7)$$

$$plog(S) = \sum_{lexicon} count_S(l) plog(l). \quad (8)$$

If we divide through by the length of our string, we get the average which is Shannon's entropy:

$$entropy(S) = \sum_{lexicon} freq_S(l)\, plog(l). \quad (9)$$

This is more familiar if we write $-\sum p(x) logp(x)$.

cross-entropy of two distributions

$$-\sum_{x \in X} \mathrm{p(x)}\ \log\ \mathrm{q(x)}. \tag{10}$$

cross-entropy is less than self-entropy

- p() and q() are two different distributions.
- How do $-\sum$ p(x) log p(x) and $-\sum$p(x) log q(x) compare?
- $-\sum$ p(x) log p(x) $+\sum$p(x) log q(x) $=\sum$ p(x) log $\frac{q(x)}{p(x)}$
- Suppose we use natural logs: then we know that $ln(x) \leq (x-1)$.
- $\sum$ p(x) log $\frac{q(x)}{p(x)} \leq \sum$ p(x) [$\frac{q(x)}{p(x)}$ - 1] = $\sum p(x) - \sum q(x) = 1 - 1 = 0$
- So $-\sum$ p(x) log p(x) (the entropy) is always smaller than the cross-entropy $-\sum$p(x) log q(x)

Entropy

The entropy of a string, or signal, or language, is the average plog of the symbols.

A word about averages:

Middle school: Take the values from a set, add them altogether, and divide by the number of items in the set.

Really: Consider all possible values, and for each determine how many items take that value, and then normalize the counts of those items. Take a weighted sum of the values, where you weight by the frequency (=normalized counts). An average always takes the form $\sum_x f(x)d(x)$, where $f()$ is the function you care about, and $d()$ is a distribution over the set you care about.

1.4.2 encoding of events, and compression

(10 minutes)

1. Switch gears to information theory and encoding. We now leave the domain of anything like randomness. Everything is sharp and clean—austere, even.

2. Lossless versus lossy compression.

What the linguist cares about…

Redundancy

Structure

What the linguist cares about…

Redundancy

Structure

and wants to find, and bring to everyone's attention.

Information theory wants to identify…

Redundancy

What the linguist cares about…

Structure

and wants to find, and bring to everyone's attention.

Information theory wants to identify…

Redundancy

and to remove it, to leave the message: what makes this moment different from all others.

What the linguist cares about…

Structure

and wants to find, and bring to everyone's attention.

Information theory wants to identify…

Redundancy

and to remove it, to leave the message: what makes this moment different from all others.

What the linguist cares about…

Structure

and wants to find, and bring to everyone's attention.

Information theory wants to identify…

Redundancy

and to remove it, to leave the message: what makes this moment different from all others.

What the linguist cares about…

Structure

and wants to find, and bring to everyone's attention.

Both sides agree that a signal has two aspects to it:

- the structure of the system
- the content of this moment's message

Information theory wants to identify…

Redundancy

and to remove it, to leave the message: what makes this moment different from all others.

What the linguist cares about…

Structure

and wants to find, and bring to everyone's attention.

Redundancy means the loss of what had seemed to be choices, a loss due to the very system itself.

Information theory wants to identify…

Redundancy

and to remove it, to leave the message: what makes this moment different from all others.

What the linguist cares about…

Structure

and wants to find, and bring to everyone's attention.

Both sides can do their job when they employ probabilistic grammars.

Information theory wants to identify…

Redundancy

and to remove it, to leave the message: what makes this moment different from all others.

What the linguist cares about…

Structure

and wants to find, and bring to everyone's attention.

Both sides can do their job when they employ probabilistic grammars.

We measure redundancy by the difference in length of a

rough
dumb
raw
surfacey

description of a signal and its compressed length.

3. The shortest compression system that we can devise is a test of our understanding of the signal system, or language. And it is fully quantifiable. There is an equivalence that we must understand between *redundancy* in the system and *discoverable structure* in the data (that is, in the language). We wish to pull all of the redundancy = structure out that we possible can. And then we can measure what is left over:

4. *the measure of what is left over is the plog of the signal.* (That really is amazing). Our goal, therefore, is to come up with a grammatical description—a grammar—that minimizes the plog of the data, that is, that maximizes the probability of the data. There is a reason for this.

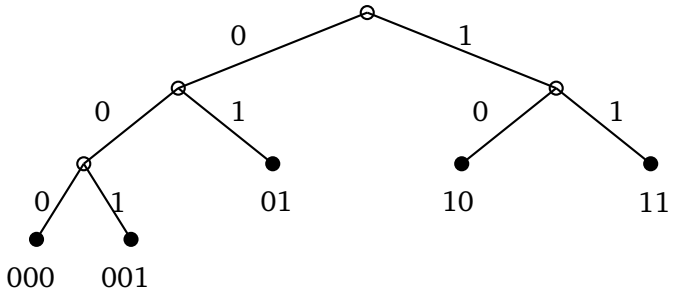
5. We assume that an encoding is a set of 0s and 1s, because their difference is the most natural unit of information.

6. A finite sequence of 0s and 1s can be thought of as a real number between 0 and 1: 0.s[1]s[2]s[3]...s[n].

7. If we set up a system of sequences of 0s and 1s for each unit in our system, then we have a natural encoding. If we impose the *no-prefix condition*, then there is a natural connection between an encoding system and a distribution. That is *not obvious*!

   Picture

8. Lying behind this picture is the notion that there is a natural connection between a binary string of length n and the number 2^{-n}. And a natural encoding of this sort will always create (or can be associated with) a well-formed probability distribution. The more probable an event is, the shorter its encoding. In particular, an event's encoding will be equal to the plog of its probability.



Fundamental fact: The best encoding system (i.e., the one that produces the shortest encodings) is always the one in which a symbol s of probabily k is encoded by a string of 0s and 1s whose length is $-1 \times log\, p(s)$.

The length of the encoding that we are able to provide for a signal is our estimate of how much information is in that signal that is not due to the structure (i.e., the redundancy) of the system (i.e., the language).

Redundancy = structure = system. What is left when we remove it is a message, as far as we can tell.

1.4.3 encoding of grammars

(10 minutes)

We now have a way of encoding signals in a compact way, and a way about thinking about the degree of compactness.

This way—which is the *information theoretic* way of looking at things—works just as well for grammars as it does for signals.

A grammar can always be formalized as a set of decisions, chosen from a universal inventory of choices. Just like with a signal, we can deal equally with conceptions of grammar in which there are a finite or in which there are an infinite number of grammars.

We can consider all sorts of different ways of encoding an idea for a grammar, and we will select the one that is the shortest as our bet on how to describe the language that we are studying.

Imagine a phrase-structure grammar with k non-terminal categories, m lexical categories, and N vocabulary items. Then a grammar can be written as a sequence of rewrite rules, and we can calculate the compressed length of the grammar.

$A \rightarrow B\,C; B \rightarrow n; C \rightarrow v; kim_n; slept_v;$

Our goal will be to show that the process of coming up with the most compact description of a grammar corresponds quite to the linguist's idea of finding the best grammar for a set of data.

1.5 Terms we discussed today

count	frequency	distribution	set
multiset	probability	corpus (corpora)	machine learning
logarithm	base 2 log(arithm)	plog	entropy
conditional probability	mutual information	machine learning	unsupervised learning
semi-supervised learning			

1.6 What is next

(10 minutes)

1.7 Unsupervised learning: Minimum Description Length analysis

1. Maximize the probability of the data:

   Find g such that $pr_g(D)$ is the greatest.

   $$\hat{g} = argmax\, pr_g(D)$$

   or...

   $$p(g|D) = \frac{p(D|g)p(g)}{p(D)}$$

   What is p(D)?

Maximize both	$p(D\|g)$ and $p(g)$	– which is to say,
maximize	$logp(D\|g) + logp(g)$,	
or minimize	$-log\, p(D\|g) - log(g)$,	

   which is the information content of the data given the model, plus the complexity of the model.

2. How do we measure or compute the probability of a model g? One possible answer: Kolmogorov complexity.

   - Select a particular programming language (Turing machine) through a fairness principle.
   - Find the shortest program m that embodies [computes] the model.
   - Measure m's length expressed in bits.
   - $p(g) = 2^{-length(m)}$

"There are few areas of science in which one would seriously consider the possibility of developing a general, practical, mechanical method for choosing among several theories, each compatible with the available data." (Noam Chomsky, *Syntactic Structures* (1957))

Unsupervised learning has two parts:

a) Hypothesis generation.

b) Hypothesis testing (evaluation).

Minimum Description Length (MDL) analysis is an excellent candidate for hypothesis testing. But it's not enough.

Interpreting this graph: The x-axis and y-axis both quantities measured in *bits*. The x-axis marks how many bits we are allowed to use to write a grammar to describe the data: the more bits we are allowed, the better our description will be, until the point where we are over-fitting the data. Thus each point along the x-axis represents a possible grammar-length; but for any given length l, we care only about the grammar g that assigns the highest probability to the data, i.e., the $best$ grammar. The red line indicates how many bits of data are left unexplained by the grammar, a quantity which is equal to -1 * log probability of the data as assigned by the grammar. The blue line shows the sum of these two qunantities (which is the conditional *description length* of the data). The black line gives the length of the grammar.

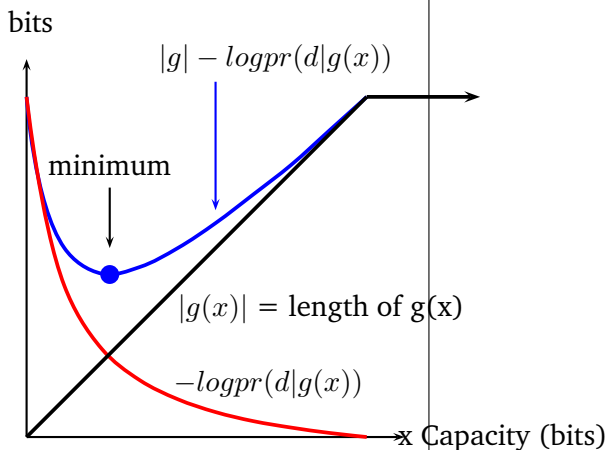


Fig. 1.5: MDL optimization

1.8 A bunch of general and important points

1.8.1 The notion of learning from data

The agony of humility of learning from data and using quantitative evaluation

1.8.2 The importance of quantitative measurements

Precision and recall

How well do the documents that your system gives you actually satisfy what you are looking for?

How sure are you that you got back all of the documents you really wanted?

Document retrieval

$$\frac{\#(\text{appropriate documents returned})}{\#(\text{documents returned})}$$

$$\text{Recall } \frac{\#(\text{appropriate documents returned})}{\#(\text{appropriate documents})}$$

What would we chose? Suppose we have been given a large set of data from a previously unanalyzed language, and four different analyses of the verbal system are being offered by four different linguists. Each has an account of the verbal morphology using rules that are (individually) of equal complexity. There are 100 verb stems. Verbs in each group use the same rules; verbs in different groups use entirely different rules.

Linguist 1 found that he had to divide the verbs into 10 groups with 10 verbs in each group.
Linguist 2 found that she had to divide the verbs into 10 groups, with 50 in the first group, 30 in the second group, 6 in the third

group, and 2 in each of 7 small groups.
Linguist 3 found that he had just one group of verbs, with a set of rules that worked for all of them.
Linguist 4 found that she had to divide the verbs into 50 groups, each with 2 stems in it.

Rank these four analyses according how good you think they are—sight unseen.

Hopefully you ranked them this way:

Best: Analysis 3
Analysis 2
Analysis 1
Worst: Analysis 4

And *why?* Because the entropy of the sets that they created goes in that order. That's not a coincidence—*entropy measures our intuition of the degree of organization of information.*

The entropy of a set is $-\sum pr(a_i)\, log\, pr(a_i)$, where we sum over the probability of each subset making up the whole—and where the log is the $base_2\ log$.

- The entropy of Linguist 1's set of verbs is$-1 \times 10 \times \frac{1}{10} \times log\frac{1}{10} = log(10) = 3.32$.
- The entropy of Linguist 2's set of verbs is $-1 \times \frac{1}{2} \times log\frac{1}{2} + 0.3 \times log(0.3) + 0.06 \times log(0.06) + 0.14 \times log(0.02)) = 0.346 + 0.361 + 0.169 + 0.548 = 1.42$.
- The entropy of Linguist 3's set of verbs is $-1 \times 1 \times log(1) = 0$.

- The entropy of Linguist 4's set of verbs is $-1 \times 50 \times \frac{1}{50} \times log(0.02) = 3.91$.

The concept of entropy can be used to quantify the notion of elegance of analysis.