- create parsing table & parse sentences
- convert between dependency tree <=> phrase structure tree
- fill chart parsing table, for given PCFG
- může tam být i jako „spočítejte MI" nebo „spočítejte H" ...

1. The Basics: Probability

Suppose your training data contains (only) the sentence (http://www.petpainter.com, slightly adapted)

```
We specialize in petbirds including but not limited to dogs , cats , and horses .
```

You are building a bigram language model $p(w_i|w_{i-1})$ using maximum likelihood estimation. You do **not** add any special symbols/words to the beginning and end of the use separate unigram distribution for the first position in any given data).

- What is the (raw) probability of `p(cats|,)`? _____
- What is the definition of the training data probability (using, let's say, the fourgram approximation)?

_____

a) 1/2

b) $P(Y) \approx \prod_i (y_i | y_{i-3}, y_{i-2}, y_{i-1})$

-> would be $P(y_0) \cdot P(y_1|y_0) \cdot P(y_2|y_0, y_1) \cdot P(y_3|y_0, y_1, y_2) \cdots$

therefore we generalize into

2. Tagging

- Describe three different approaches to tagging. Include formulas if/where appropriate.

a) HMM tagging

- represented as HMM, where states are actually hidden and can represent some semantic properties of the language. Transitions simulate relations between the language properties. Each state can emmit output (tag) as well, closing the gap between relevant (hidden) properties of the language and the desired output.
- is supported by a hypothesis that there are some hidden properties that might very well generate our output.

b) Rule-based tagging

- implemented usually as a set of Reg. Exp.
- Viterbi-like searches (without probs.) are made
- rules ~ FSA ? must be created by hand
- if still used, then only to very quickly check some basic rules of the language.
- there is problem when it returns multiple possible parses of the input, as there is no additional probability given

c) Maximum-entropy rule-based tagging

- rules are generated very simply to create big pool
- then we always try to add one new rule
  - we want to have the highest entropy when picking new feature set. ~ high recall
  - then we take such combination that gives the lowest entropy. ~ high accuracy

3. Shift-Reduce Parsing

Let's assume the following Context-Free Grammar G (here with seven numbered rules):

```
#1 S -> K W
#2 K -> K B
#3 B -> b
#4 K -> k
#5 W -> B W C
#6 C -> c
#7 W -> w
```

- Decide which of the following strings belong to the language defined by the grammar G:

kkkkbw	Yes/No: X
kbbbwcc	Yes/No: ✓
kw	Yes/No: ✓

- Create the shift-reduce parse tables for this grammar.

0	S -> . A W	A 1
	A -> . h B	
	A -> . h	h 2
1	S -> A . W	W 3
	W -> . B W C	B 4
	A -> A . B	
	W -> . w	w 5
	B -> . b	b 6
2	A -> h .	# 4
3	S -> A W .	# 1
4	W -> B . W C	W 7
	W -> . B W C	B 4
	W -> . w	w 5
	B -> . b	b 6
	A -> A B .	# 2
5	W -> w .	# 7
6	B -> b .	# 3
7	W -> B W . C	C 8
	C -> . C	C 9
8	W -> B W C .	# 5
9	C -> c .	# 6

4. Statistical Parsing

- What is a Probabilistic CFG? Use five sentences at most.

It is way to introduce language parsing, for example for tagging, with probability included. Problem with typical CFG is that even the most obscure sentences within the training data will have the same weight as other branches of parsing, even though it does not mirror reality. Therefore we introduce:

$P(T) = \prod_i p(r_i)$ → prob. of parse tree, applying rules r_i

$P(w) = \sum_j P(T_j)$ → prob. of parse of sentence w.

Now suppose the following "treebank" data, where the pair of symbols (...)XX denotes a (sub)tree rooted in a nonterminal XX.

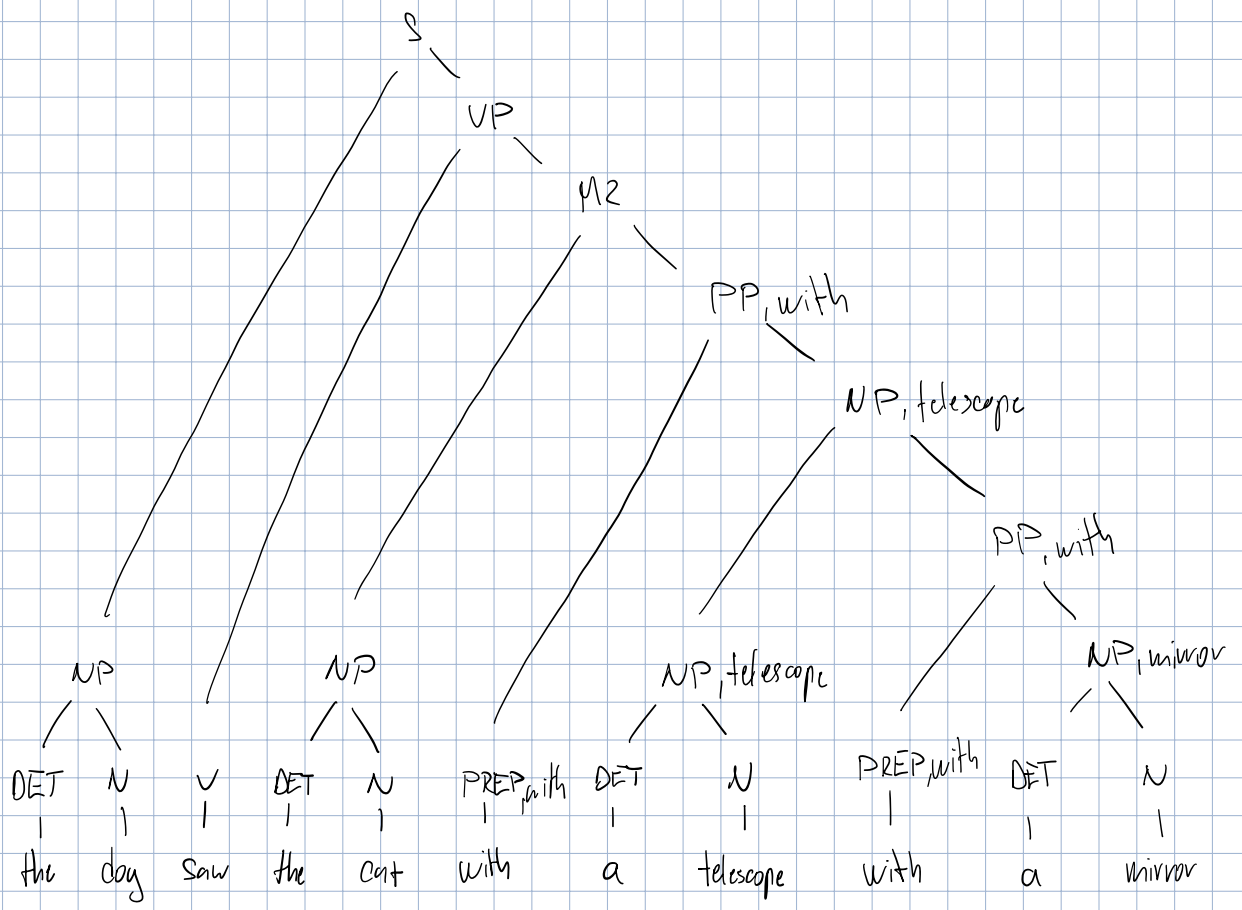
Sentence 1:

```
(((the)DET
(dog)N)NP
((saw)V
(((the)DET
(cat)N)NP
((with)PREP
((a)DET
(telescope)N)NP
((with)PREP
((a)DET
(mirror)N)NP)PP)NP)PP)M2)VP)S
```

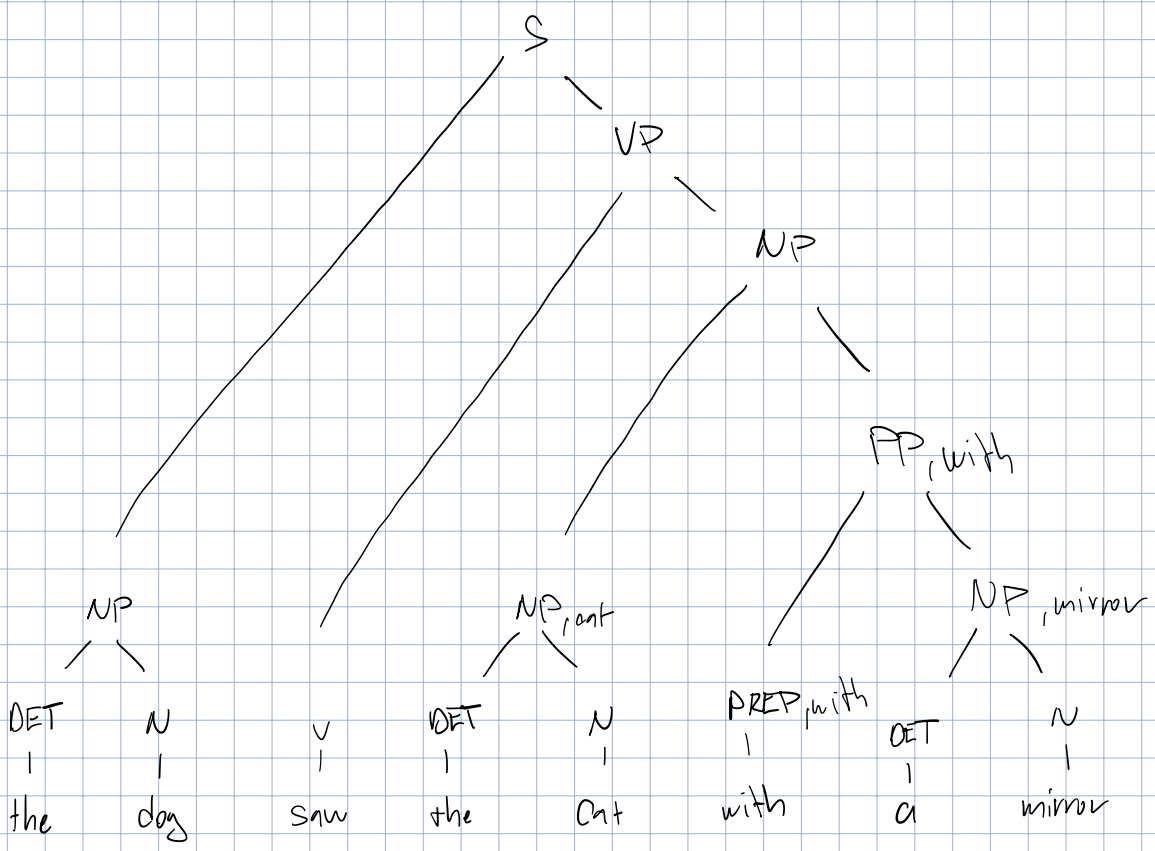
Sentence 2:

```
(((the)DET
(dog)N)NP
((saw)V
(((the)DET
(cat)N)NP
((with)PREP
((a)DET
(mirror)N)NP)PP)NP)VP)S
```

- Draw the usual parse trees, with terminal symbols (words) as leaves, and other nodes labeled by nonterminals (use the next page):



```
(((the)DET
(dog)N)NP
((saw)V
(((the)DET
(cat)N)NP
((with)PREP
((a)DET
(mirror)N)NP)PP)NP)VP)S
```



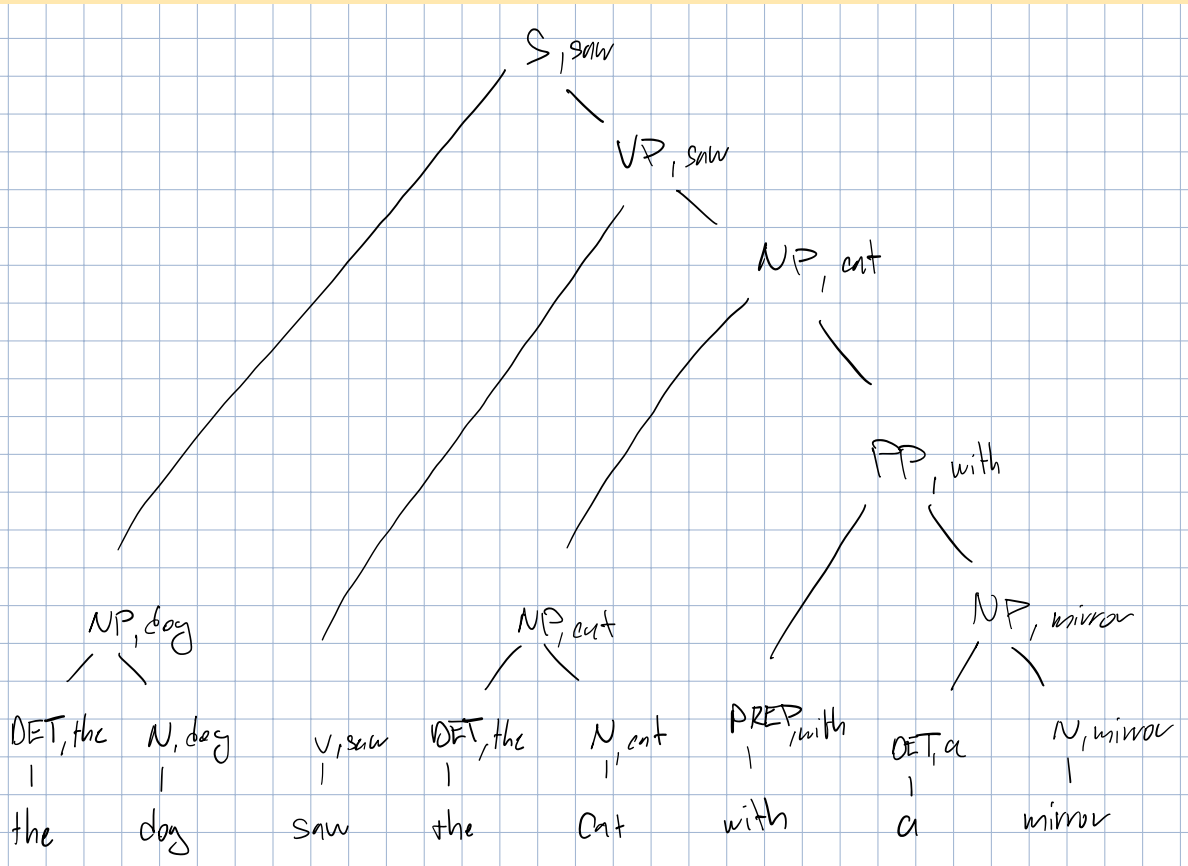
- Write down the (general, non-lexicalized) CFG grammar G as extracted from this training data.

#01	S -> NP VP	2
#02	VP -> V M2	1
#03	VP -> V NP	1
#04	NP -> DET N	7
#05	NP -> NP PP	2
#06	V -> saw	2
#07	M2 -> NP PP	1
#08	PP -> PREP NP	3
#09	PREP -> with	3
#10	DET -> the	4
#11	DET -> a	3
#12	N -> dog	2
#13	N -> cat	2
#14	N -> mirror	2
#15	N -> telescope	1

OK. Now you have everything to answer these "numerical" questions:

- Estimate p(NP -> NP PP) = 2/9
- Estimate p(NP -> PREP N) = 0
- Estimate p(VP -> V NP) = 1/2
- Estimate p(VP -> V M2) = 1/2

- Now draw the **lexicalized** parse tree for the **second** sentence, assuming that whenever there is a choice of heads in any given subtree, chose (in the following precedence order): VP, V, PREP, N, NP.

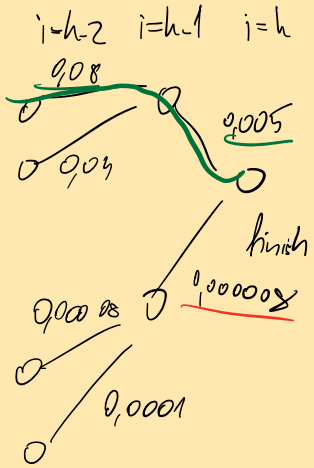


Suppose we stick to the usual three independence asumptions and define a probability of a parse tree s (of a sentence W) as a simple product of p(<rule>) over all rules used in the course of generation of the sentence W.

Describe the main idea of the algorithm to compute:

- (a) the best parse given a sentence and a model. Use five sentences at most.

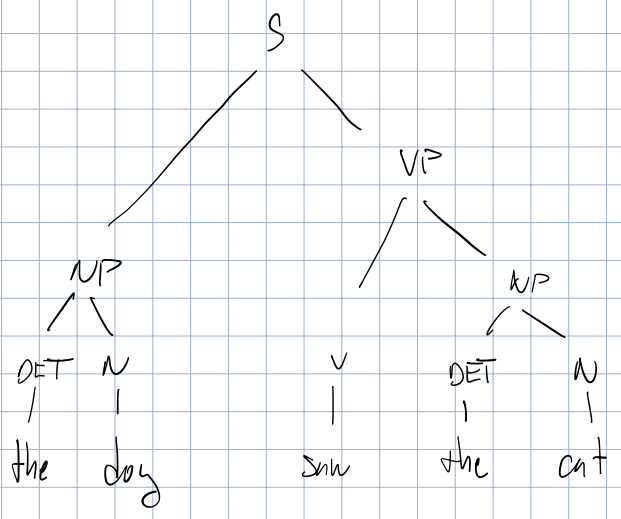
We can modify Viterbi to always remember for each state the most likely path to it. If we get to the final word, we then backtrack and go over the most likely paths.



- (b) the probability of a string given the model. Use five sentences at most.

Now instead of always taking just one argmax, if we come to a state from two paths, we sum the probs, making the state more likely to happen as more paths lead to it.
We get the probability as just the prob. at the final state.

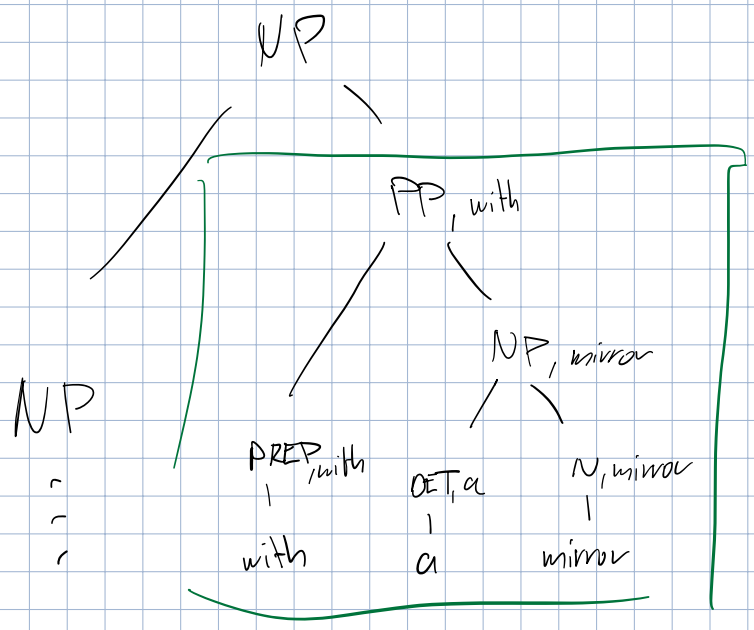
- What is then the probablity of the following parse tree, using the estimated nonlexicalized PCFG:
  p((((the)DET (dog)N)NP ((saw)V ((the)DET (cat)N)NP)VP)S) = _____ 0,008062
- What is the probability of the string "the dog saw the cat with a mirror"?
  p(sentence$_2$) = _____ 0,000091305
- Now a lexicalized grammar question. What is the (raw) probability estimate of the lexicalized rule p(PP(with) -> PREP(with) NP(mirror))? [Imagine you generate the lexicalized rules, too, under the same independence assumptions; however, to answer this question, there is no need to generate them all, of course. However, do not forget to look at **both** training sentences!]
  p(PP(with) -> PREP(with) NP(mirror)) = _____ 2/3



0) S -> NP VP 1
1) NP -> DET N 7/9
2) DET -> the 4/7
3) N -> dog 2/7
4) VP -> V NP 1/2
5) V -> saw 1
6) NP -> DET N 3/9
7) DET -> the 4/7
8) N -> cat 2/7

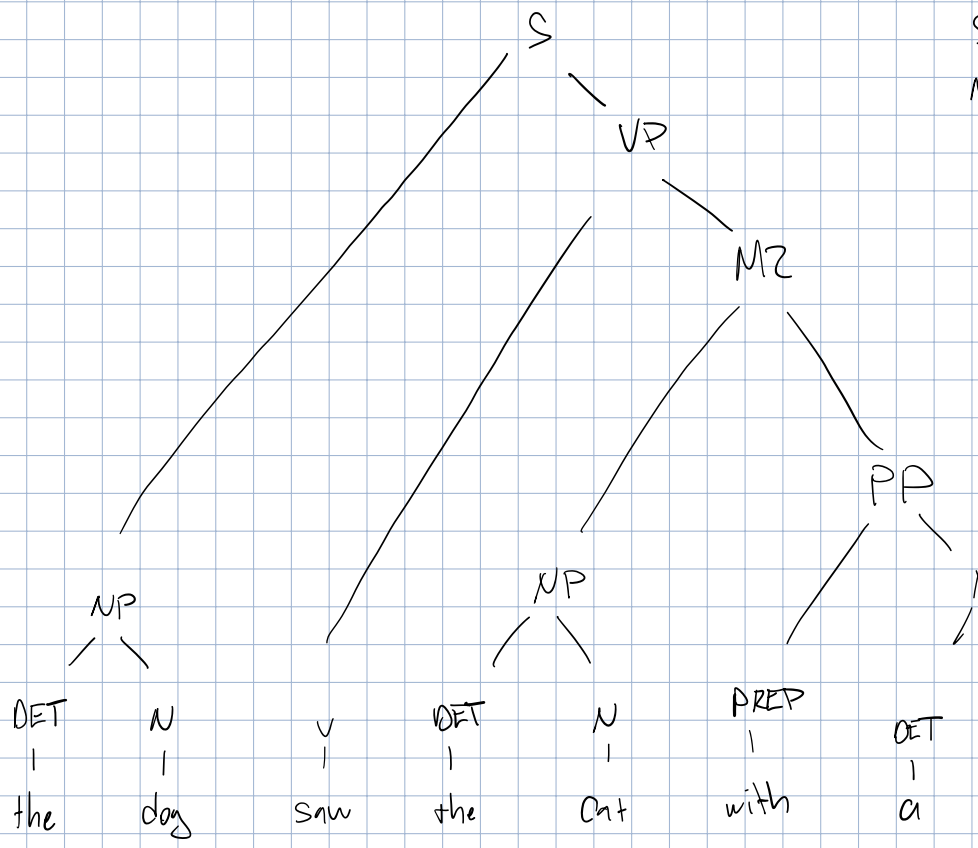
11

0,008062...



NP -> NP PP 2/9
PP -> PREP NP 1
NP -> DET N 7/9
DET -> a 3/7
N -> mirror 2/7

0,0001706 =



S -> NP VP 1
NP -> DET N 7/9
DET -> the 4/7
N -> dog 2/7
VP -> V MZ 1/2
V -> saw 1
MZ -> NP PP 1
NP -> DET N 2/9
DET -> the 4/7
N -> cat 2/7
PP -> PREP NP 1
PREP -> with 1
NP -> DET N 7/9
DET -> a 3/7
N -> mirror 2/7

0,0007679

Another shift-reduce example:

0	S -> . A R	A1
	A -> . A B	
	A -> . a	a2
1	S -> A . R	R3
	A -> A . B	B4
	R -> . B R C	
	R -> . r	r5
	B -> . b	b6
2	A -> a .	#3
3	S -> A R .	#1
4	A -> A B .	#2
	R -> B . R C	R7
	R -> . B R C	B4
	R -> . r	r5

aaaabr	X
abbbrcc	✓
ar	✓

stack	states	comments
abbbrc	0	
bbbrc	2 0	
Abbbrc	0	#3
bbbrc	1 0	
bbrc	6 1 0	
Bbbrc	1 0	#4
bbrc	4 1 0	
brc	6 4 1 0	
Bbrc	4 1 0	#4
brc	4 4 1 0	
rc	6 4 4 1 0	
Brc	4 4 1 0	#4
rc	4 4 4 1 0	

	B	-> .b	b6
5	R	-> r.	#6
6	B	-> b.	#4
7	R	-> BR.C	C8
	C	-> .c	c9
8	R	-> BRC.	#5
9	C	-> c.	#7

c	544410	
Rc	44410	#6
c	744410	
—	9744410	
C	744410	#7
—	8744410	
R	4410	#5
—	744410	
AR	10	#2
ARc	410	#2
Arc	410	#2
ABrc	10	#2
Abrc	10	#2
ABbrc	0	#2
Bbrc	10	
brc	410	
rc	6410	
Brc	410	#4
rc	4410	
c	54410	
Rc	4410	#6
c	74410	
—	974410	
C	74410	#7
—	874410	
R	410	#5
—	7410	

Entropy: $H(P) = -\sum_x P(x) \cdot \log P(x)$ NLL

Perplexity $= 2^{H(P)}$

$$H(P_{bigram}) = -\sum_x P(x,y) \cdot \log P(x|y)$$

Cross Entropy:

$$H(P,Q) = -\sum_x P(x) \cdot \log Q(x)$$

$$H(P,x) \geq H(P) \quad \forall x$$

$$H(P,P) = H(P)$$

we : 1/16

specialize : 1/16

in : 1/16

petbirds : 1/16

including : 1/15

but : 1/15

not : 1/16

limited : 1/16

to : 1/16

dogs : 1/14

, 2/16

cats : 1/16

and : 1/16

horses : 1/16