StatML (Chapter 7): Language Models

Chunqi Shi

Brandeis University

shicq@brandeis.edu

October 2, 2014

Overview

Language Models

Why?

Language models answer the question:

How likely is a string of English words good English?

What?

A statistical language model assigns a probability to a sequence of m words $P(w_1, \ldots, w_m)$ by means of a probability distribution.

How?

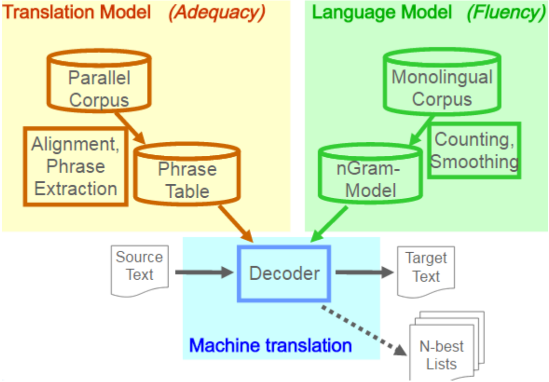
- Reordering:

  $P_{LM}(\text{the house is small}) > P_{LM}(\text{small the is house})$
- Word Choice:

  $P_{LM}(\text{I am going home}) > P_{LM}(\text{I am going house})$

Language Models & SMT Architecture

How language models work in a basic SMT architecture[1]?

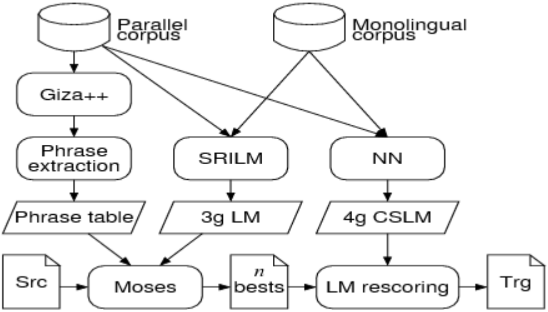


[1] http://slideplayer.us/slide/203403/

Open Source Language Models Example

Architecture of the LIMSI SMT system[2] and open language models:

- SRILM[3] (N-Gram) & NN[Neural Networks] (Continuous Space LM).
- Giza++: Translation Model.
- Moses: Decoder



[2] http://www.limsi.fr/tlp/mt/

[3] http://sourceforge.net/projects/irstlm/

Other Language Models Applications

Speech Recognition

P_{LM}(I saw a van) $>$ P_{LM}(eyes awe of an)

Spell Correction

The office is about fifteen minuets from my house.

P_{LM}(about fifteen minutes from) $>$ P_{LM}(about fifteen minuets from)

Information Retrieval

No results found for "University of Brandeis" (Query likelihood model).

P_{LM}(University of Brandeis) $>$ P_{LM}(Brandeis University)

More !!

Part-of-speech Tagging, Parsing, Summarization, Question-Answering, etc.

Probabilistic Language Modeling

How to Compute $P(W)$

$$P(W) = P(w_1, \ldots, w_m)$$

Probability of an upcoming word

$$P(w_k | w_1, w_2, \ldots, w_{k-1})$$

Decomposing using Chain Rule

$$P(w_1, \ldots, w_m) =$$
$$P(w_1)P(w_2|w_1)P(w_2|w_1, w_2) \ldots P(w_m|w_1, w_2, \ldots, w_{m-1})$$

Example

$P(\text{its water is so transparent}) =$
$P(\text{its}) \times P(\text{water}|\text{its}) \times P(\text{is}|\text{its water}) \times P(\text{so}|\text{its water is}) \times$
$P(\text{transparent}|\text{its water is so})$

Chain Rule Estimation

Joint Probability

$$P(w_1 w_2 \dots w_m) = \prod P(w_i | w_1 w_2 \dots w_i - 1)$$

How to estimate?

Maximum likelihood estimation:

$P(\text{transparent}|\text{its water is so}) =$

$$\frac{Count(\text{its water is so transparent})}{Count(\text{its water is so})}$$

Problems?

- Sparse data: NO enough data for estimating.
- Large space: HUGE possible sentences.

Markov Chain

Markov Assumption

Only previous history matters:

$P(\text{transparent}|\text{its water is so}) = (\text{transparent}|\text{so})$ or maybe

$P(\text{transparent}|\text{its water is so}) = (\text{transparent}|\text{so})$

k_{th} Order Markov Model

$$P(w_1 w_2 \ldots w_m) = \prod P(w_i | w_{i-k} w_2 \ldots w_i - 1)$$

Simple Cases

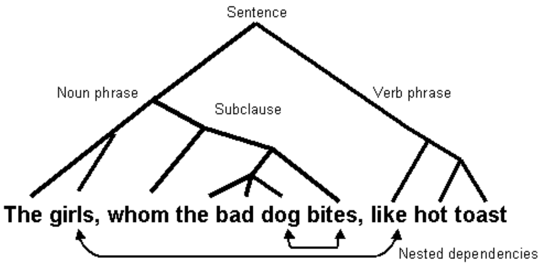
Unigram model: $P(w_1 w_2 \ldots w_m) = \prod P(w_i)$

Bigram model: $P(w_1 w_2 \ldots w_m) = \prod P(w_i | w_{i-1})$

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:



Or:

"The computer which I had just put into the machine room on the fifth floor crashed."

Bigram Example

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

<s> I am Sam </s>
<s> Sam I am </s>
<s> I do not like green eggs and ham </s>

$$P(\texttt{I} \mid \texttt{<s>}) = \tfrac{2}{3} = .67 \qquad P(\texttt{Sam} \mid \texttt{<s>}) = \tfrac{1}{3} = .33 \qquad P(\texttt{am} \mid \texttt{I}) = \tfrac{2}{3} = .67$$

$$P(\texttt{</s>} \mid \texttt{Sam}) = \tfrac{1}{2} = 0.5 \qquad P(\texttt{Sam} \mid \texttt{am}) = \tfrac{1}{2} = .5 \qquad P(\texttt{do} \mid \texttt{I}) = \tfrac{1}{3} = .33$$

Trigram Example

- Counts for trigrams and estimated word probabilities

the green (total: 1748)		
word	c.	prob.
paper	801	0.458
group	640	0.367
light	110	0.063
party	27	0.015
ecu	21	0.012

the red (total: 225)		
word	c.	prob.
cross	123	0.547
tape	31	0.138
army	9	0.040
card	7	0.031
,	5	0.022

the blue (total: 54)		
word	c.	prob.
box	16	0.296
.	6	0.111
flag	6	0.111
,	3	0.056
angel	3	0.056

  - 225 trigrams in the Europarl corpus start with the red
  - 123 of them end with cross
  - → maximum likelihood probability is $\frac{123}{225} = 0.547$.

"The red cross" and "The green party" are frequent trigrams in the Europarl corpus.

Evaluation of N-gram Models

How good is our model?

Extrinsic Evaluation: training A & B, testing, comparing accuracy of A & B by evaluation metric.
But it is time-consuming.

Intrinsic Evaluation

Perplexity: How well can we predict the next word?
Intrinsic evaluation is Bad approximation! Unless the test data looks just like the training data.
But is helpful to think about.

Intuition of Perplexity

How hard is the task of recognizing digits "0, 1, 2, 3, 4, 5, 6, 7, 8, 9"?
Perplexity 10.

Perplexity

Cross Entropy

$$H(W) = -\frac{1}{n}\log P(w_1 w_2 \ldots w_n)$$
$$= -\frac{1}{n}\sum_{i}^{n}\log P(w_i|w_1 \ldots, w_{i-1})$$

Perplexity:

$$PP(W) = 2^{H(W)} = P(W)^{-\frac{1}{n}}$$

Perplexity as branching factor

$$PP(W) = P(1,\ 2,\ \ldots,\ 10)^{-\frac{1}{10}}$$
$$= (\frac{1}{10})^{10\times-\frac{1}{10}} = (\frac{1}{10})^{-1} = 10$$

Comparison N-gram Models

Minimizing perplexity is the same as maximizing probability, thus better model.

word	unigram	bigram	trigram	4-gram
i	6.684	3.197	3.197	3.197
would	8.342	2.884	2.791	2.791
like	9.129	2.026	1.031	1.290
to	5.081	0.402	0.144	0.113
commend	15.487	12.335	8.794	8.633
the	3.885	1.402	1.084	0.880
rapporteur	10.840	7.319	2.763	2.350
on	6.765	4.140	4.150	1.862
his	10.678	7.316	2.367	1.978
work	9.993	4.816	3.498	2.394
.	4.896	3.020	1.785	1.510
</s>	4.828	0.005	0.000	0.000
average	8.051	4.072	2.634	2.251
perplexity	265.136	16.817	6.206	4.758

Generalization and Zeros

Unseen N-grams

Things that NOT ever occur in the training set. But occur in the test set.

Training Set:

1. ... denied the allegations
2. ... denied the reports
3. ... denied the claims
4. ... denied the request

Test Set:

1. ... denied the offer
2. ... denied the loan

$$P(\text{“offer”} \,|\, \text{”denied the”}) = 0$$

Smoothing

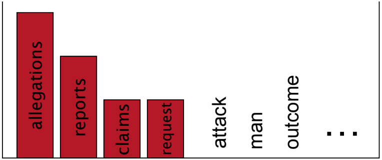
Sparse statistics, smoothing to generalize better.

Smoothing

How to smooth all words non-zeros

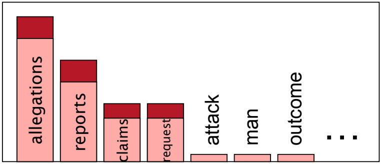
- When we have sparse statistics:

  P(w | denied the)
  3 allegations
  2 reports
  1 claims
  1 request
  7 total



- Steal probability mass to generalize better

  P(w | denied the)
  2.5 allegations
  1.5 reports
  0.5 claims
  0.5 request
  2 other
  7 total



Add-One Smoothing

Laplace smoothing

Pretend we saw each word one more time than we did.

- For all possible n-grams, add the count of one.

$$p = \frac{c+1}{n+v}$$

  - c = count of n-gram in corpus
  - n = count of history
  - v = vocabulary size

- But there are many more unseen n-grams than seen n-grams

- Example: Europarl 2-bigrams:
  - $86,700$ distinct words
  - $86,700^2 = 7,516,890,000$ possible bigrams
  - but only about $30,000,000$ words (and bigrams) in corpus

Bigram Add-One Smoothing

$$P^*(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)+1}{C(w_{n-1})+V}$$

	i	want	to	eat	chinese	food	lunch	spend
i	0.0015	0.21	0.00025	0.0025	0.00025	0.00025	0.00025	0.00075
want	0.0013	0.00042	0.26	0.00084	0.0029	0.0029	0.0025	0.00084
to	0.00078	0.00026	0.0013	0.18	0.00078	0.00026	0.0018	0.055
eat	0.00046	0.00046	0.0014	0.00046	0.0078	0.0014	0.02	0.00046
chinese	0.0012	0.00062	0.00062	0.00062	0.00062	0.052	0.0012	0.00062
food	0.0063	0.00039	0.0063	0.00039	0.00079	0.002	0.00039	0.00039
lunch	0.0017	0.00056	0.00056	0.00056	0.00056	0.0011	0.00056	0.00056
spend	0.0012	0.00058	0.0012	0.00058	0.00058	0.00058	0.00058	0.00058

Add-α Smoothing

Will α adjusted count a lot?

- Add $\alpha < 1$ to each count

$$p = \frac{c + \alpha}{n + \alpha v}$$

- What is a good value for α?
- Could be optimized on held-out set

Comparison of Add-α Smoothing

Bigram in Europarl corpus

Count	Adjusted count		Test count
c	$(c+1)\frac{n}{n+v^2}$	$(c+\alpha)\frac{n}{n+\alpha v^2}$	t_c
0	0.00378	0.00016	0.00016
1	0.00755	0.95725	0.46235
2	0.01133	1.91433	1.39946
3	0.01511	2.87141	2.34307
4	0.01888	3.82850	3.35202
5	0.02266	4.78558	4.35234
6	0.02644	5.74266	5.33762
8	0.03399	7.65683	7.15074
10	0.04155	9.57100	9.11927
20	0.07931	19.14183	18.95948

- Add-α smoothing with $\alpha = 0.00017$
- t_c are average counts of n-grams in test set that occurred c times in corpus

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:

- Add $\alpha < 1$ to each count

$$p = \frac{c + \alpha}{n + \alpha v}$$

- What is a good value for α?
- Could be optimized on held-out set

Comparison of Add-α Smoothing

Bigram in Europarl corpus

Count	Adjusted count		Test count
c	$(c+1)\frac{n}{n+v^2}$	$(c+\alpha)\frac{n}{n+\alpha v^2}$	t_c
0	0.00378	0.00016	0.00016
1	0.00755	0.95725	0.46235
2	0.01133	1.91433	1.39946
3	0.01511	2.87141	2.34307
4	0.01888	3.82850	3.35202
5	0.02266	4.78558	4.35234
6	0.02644	5.74266	5.33762
8	0.03399	7.65683	7.15074
10	0.04155	9.57100	9.11927
20	0.07931	19.14183	18.95948

- Add-α smoothing with $\alpha = 0.00017$
- t_c are average counts of n-grams in test set that occurred c times in corpus

Held-out Estimation

Deleted Estimation

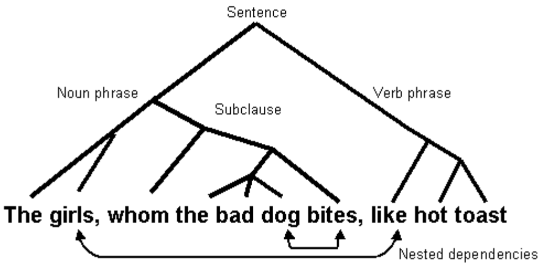
Count r	Count of counts N_r	Count in held-out T_r	Exp. count $E[r] = T_r/N_r$
0	7,515,623,434	938,504	0.00012
1	753,777	353,383	0.46900
2	170,913	239,736	1.40322
3	78,614	189,686	2.41381
4	46,769	157,485	3.36860
5	31,413	134,653	4.28820
6	22,520	122,079	5.42301
8	13,586	99,668	7.33892
10	9,106	85,666	9.41129
20	2,797	53,262	19.04992

	Adjusted count		
Count c	$(c+1)\frac{n}{n+v^2}$	$(c+\alpha)\frac{n}{n+\alpha v^2}$	Test count t_c
0	0.00378	0.00016	0.00016
1	0.00755	0.95725	0.46235
2	0.01133	1.91433	1.39946
3	0.01511	2.87141	2.34307
4	0.01888	3.82850	3.35202
5	0.02266	4.78558	4.35234
6	0.02644	5.74266	5.33762
8	0.03399	7.65683	7.15074
10	0.04155	9.57100	9.11927
20	0.07931	19.14183	18.95948

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:

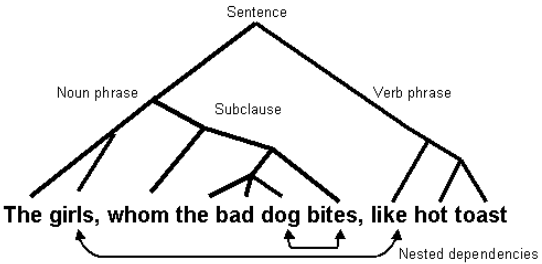


Or:
"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:

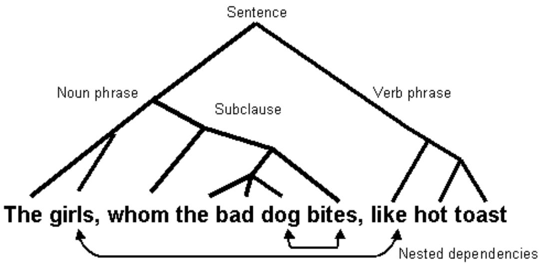


Or:
"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:



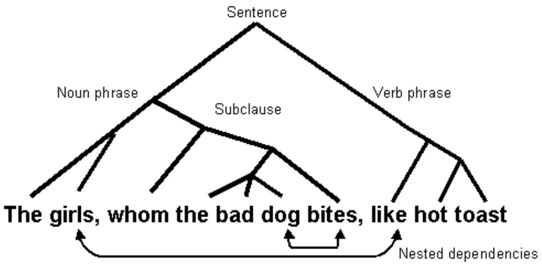
Or:

"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:

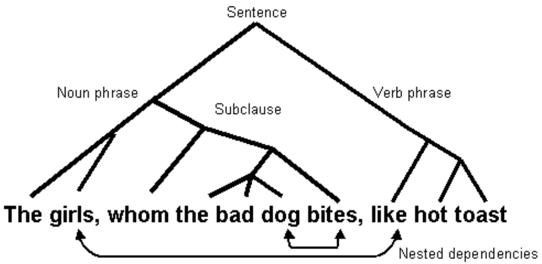


Or:
"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:



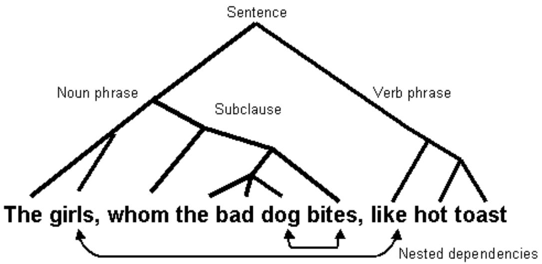
Or:

"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:



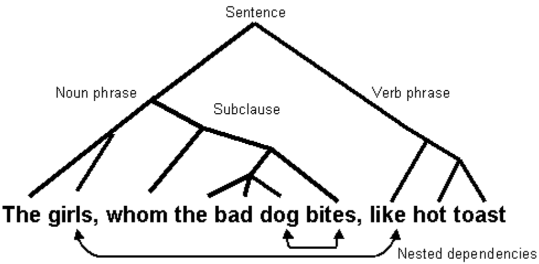
Or:

"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:



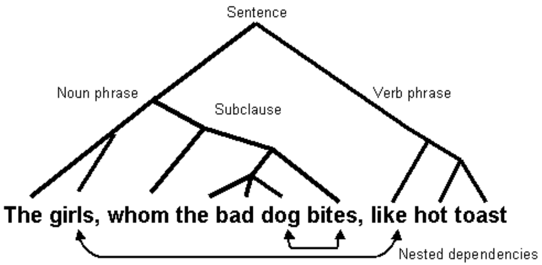
Or:

"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:

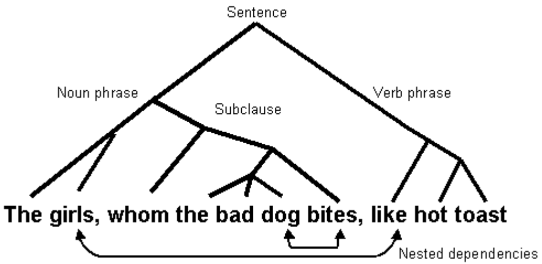


Or:
"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:

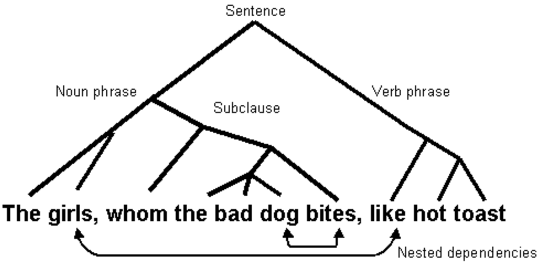


Or:
"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:



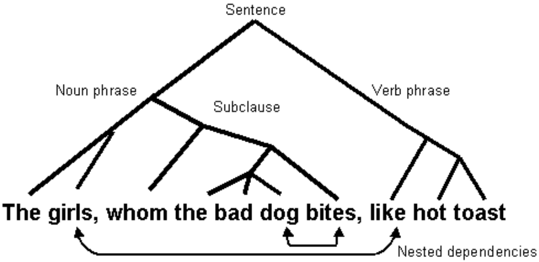
Or:

"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:

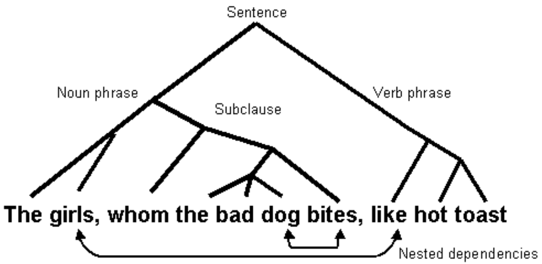


Or:
"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:



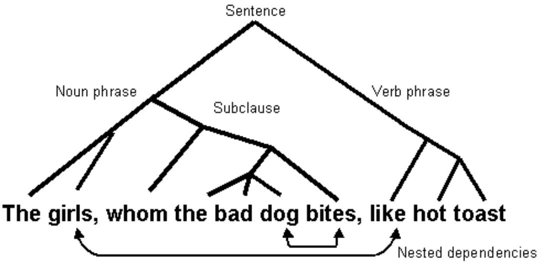
Or:

"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:

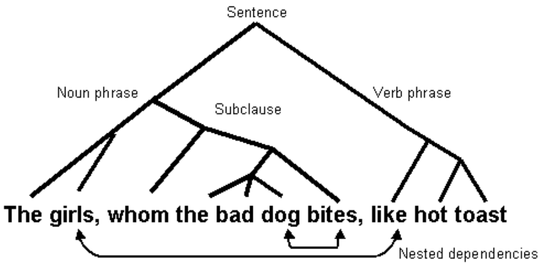


Or:
"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:

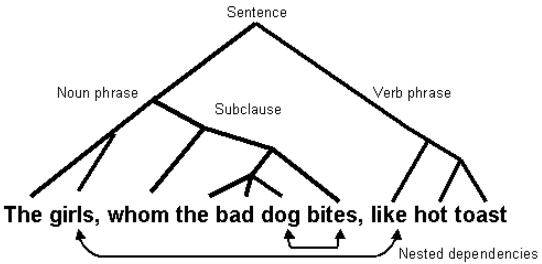


Or:
"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:



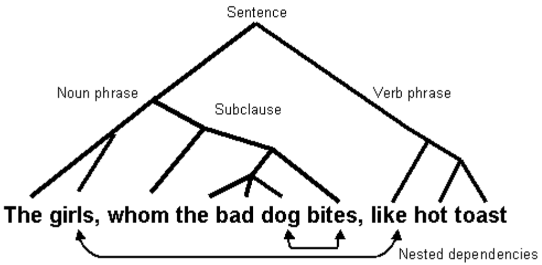
Or:

"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:



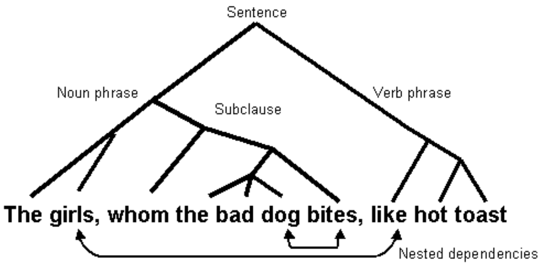
Or:

"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:

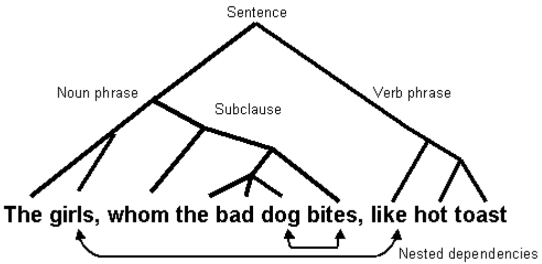


Or:
"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:

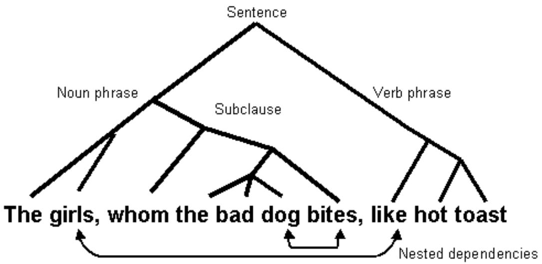


Or:
"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:



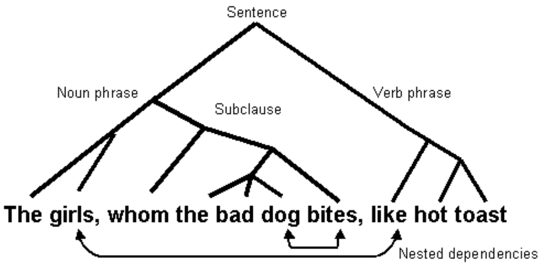
Or:

"The computer which I had just put into the machine room on the fifth floor crashed."

N-gram Models

Is Markov assumption sufficient? NO!

Language has long-distance dependencies:



Or:

"The computer which I had just put into the machine room on the fifth floor crashed."

References

Many slides are from:

- StatML book's Web site &
- Dan Jurafsky's "Language Modeling: Introduction to N-grams".

The End