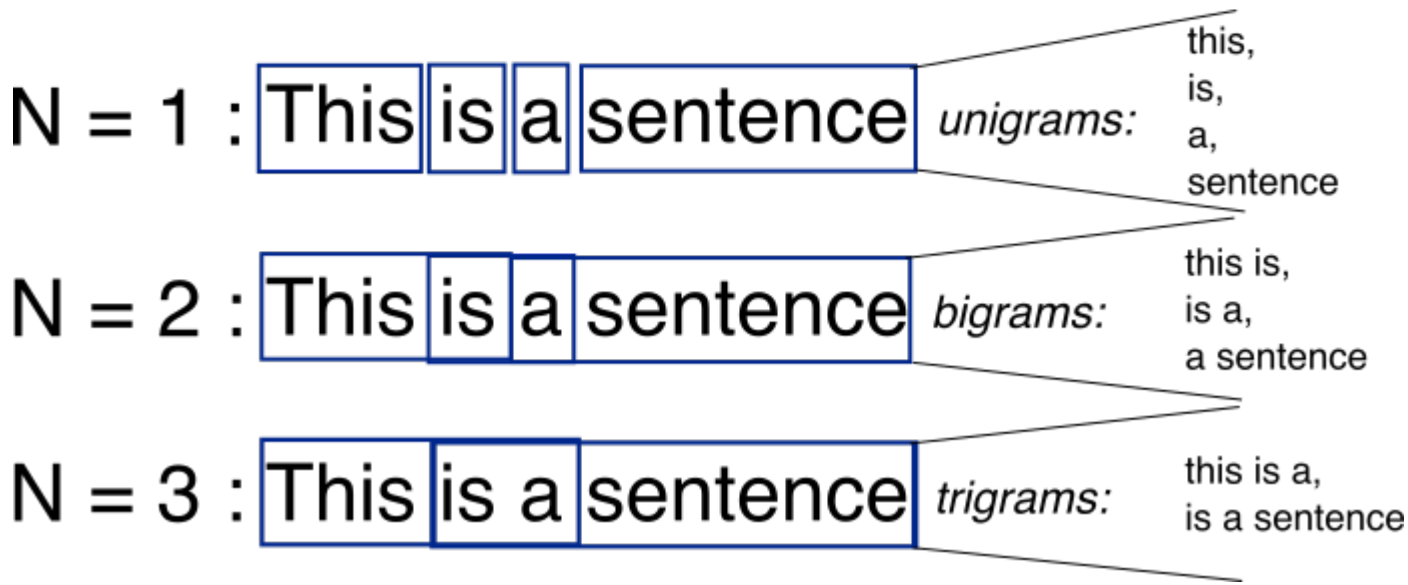
Lecture 6: Representing Words

Kai-Wei Chang
CS @ UCLA
kw@kwchang.net

Couse webpage: https://uclanlp.github.io/CS269-17/

Bag-of-Words with N-grams

- N-grams: a contiguous sequence of n tokens from a given piece of text



http://recognize-speech.com/language-model/n-gram-model/comparison

Language model

- Probability distributions over sentences (i.e., word sequences)

  P(W) = P($w_1 w_2 w_3 w_4 \dots w_k$)

- Can use them to generate strings

  P($w_k \mid w_2 w_3 w_4 \dots w_{k-1}$)

- Rank possible sentences
  - P("Today is Tuesday") > P("Tuesday Today is")
  - P("Today is Tuesday") > P("Today is Los Angeles")

N-Gram Models

- Unigram model: $P(w_1)P(w_2)P(w_3)\ldots P(w_n)$
- Bigram model:
  $P(w_1)P(w_2|w_1)P(w_3|w_2)\ldots P(w_n|w_{n-1})$
- Trigram model:
  $P(w_1)P(w_2|w_1)P(w_3|w_2,w_1)\ldots P(w_n|w_{n-1}w_{n-2})$
- N-gram model:
  $P(w_1)P(w_2|w_1)\ldots P(w_n|w_{n-1}w_{n-2}\ldots w_{n-N})$

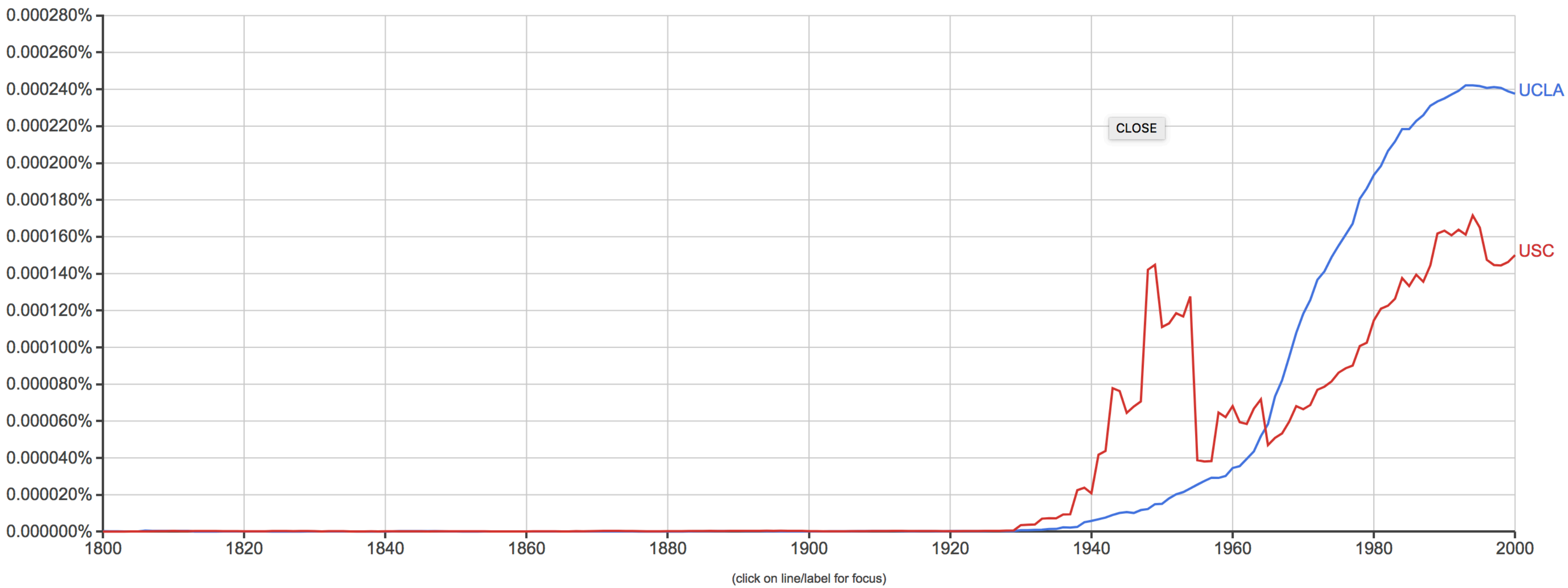
Random language via n-gram

- http://www.cs.jhu.edu/~jason/465/PowerPoint/lect01,3tr-ngram-gen.pdf

Collection of n-gram

- https://research.googleblog.com/2006/08/all-our-n-gram-are-belong-to-you.html

N-Gram Viewer



https://books.google.com/ngrams

How to represent words?

- N-gram -- cannot capture word similarity
- Word clusters
  - Brown Clustering
  - Part-of-speech tagging
- Continuous space representation
  - Word embedding

Brown Clustering

- Similar to language model
  But, basic unit is "word clusters"
- Intuition: similar words appear in similar context
- Recap: Bigram Language Models
  - $P(w_0, w_1, w_2, \dots, w_n)$
    $= P(w_1 \mid w_0) P(w_2|w_1) \dots P(w_n \mid w_{n-1})$
    $= \Pi_{i=1}^{n} P(w_i \mid w_{i-1})$

w_0 is a dummy word representing "begin of a sentence"

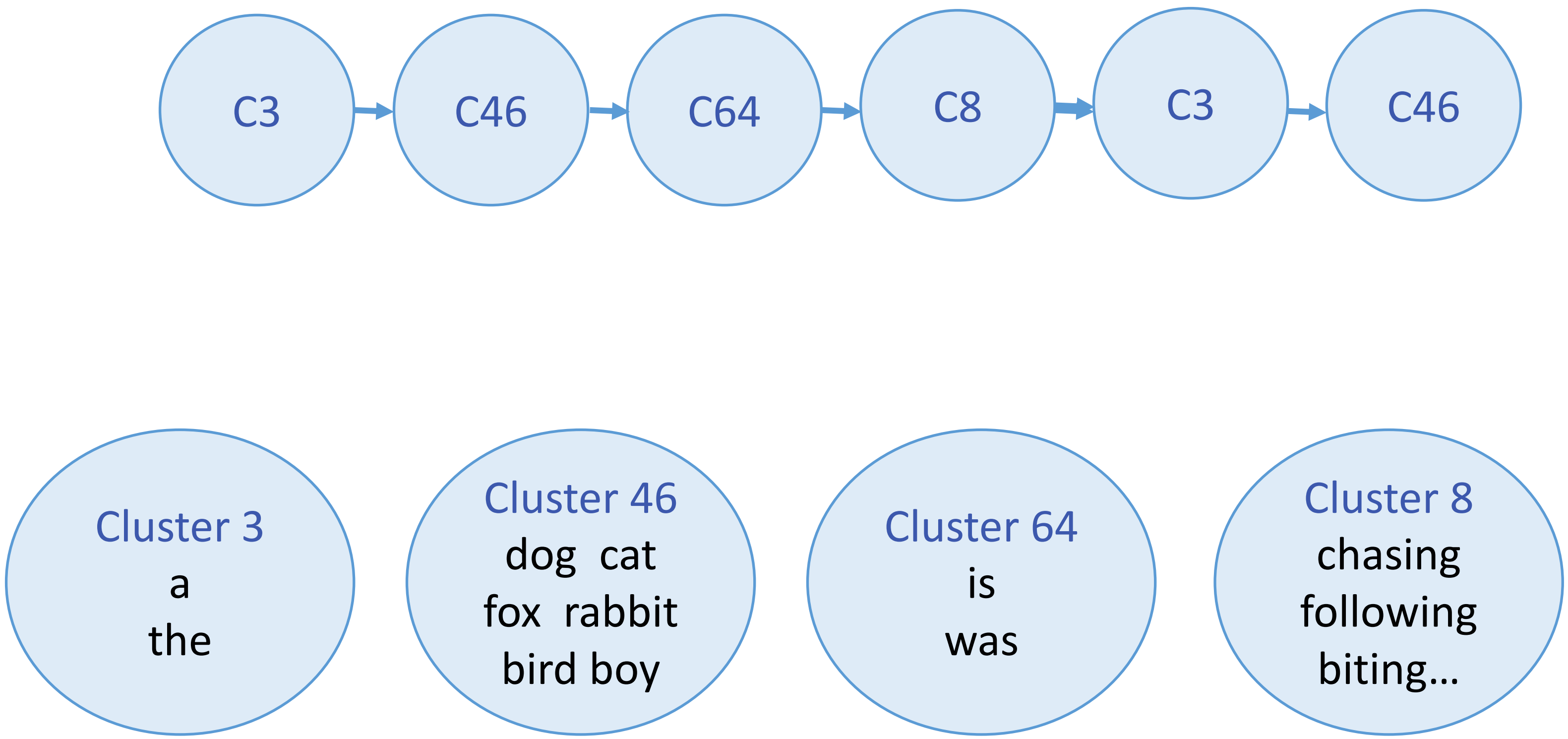
Motivation example

- "a dog is chasing a cat"
  - $P(w_0, \text{"}a\text{"}, \text{"}dog\text{"}, \ldots, \text{"}cat\text{"})$
    $= P(\text{"}a\text{"} \mid w_0) P(\text{"}dog\text{"} \mid \text{"}a\text{"}) \ldots P(\text{"}cat\text{"} \mid \text{"}a\text{"})$
- Assume Every word belongs to a cluster

Cluster 3	Cluster 46	Cluster 64	Cluster 64
a the	dog cat fox rabbit bird boy	is was	chasing following biting…

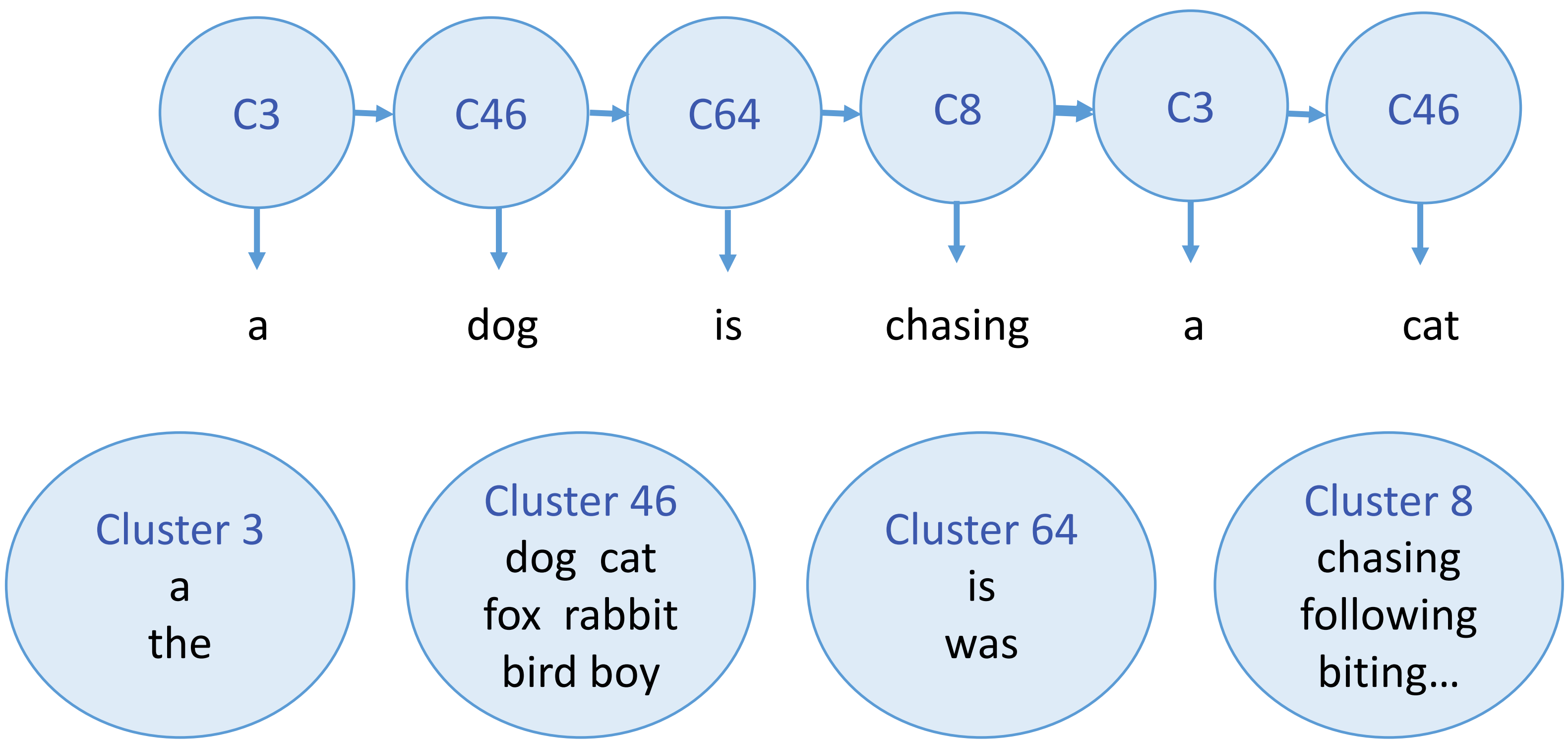
Motivation example

- Assume every word belongs to a cluster
  - "a dog is chasing a cat"



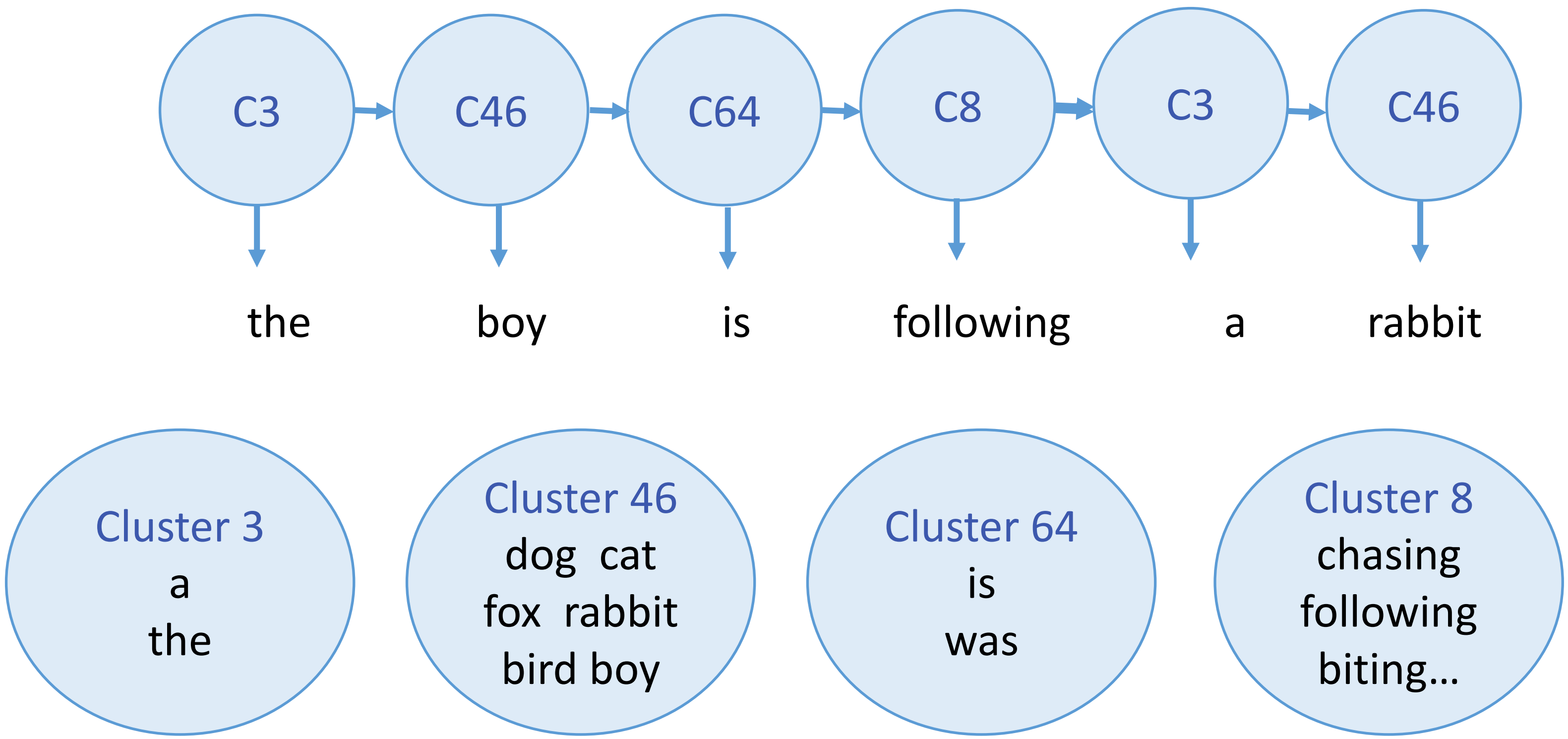
Motivation example

- Assume every word belongs to a cluster
  - “a dog is chasing a cat”



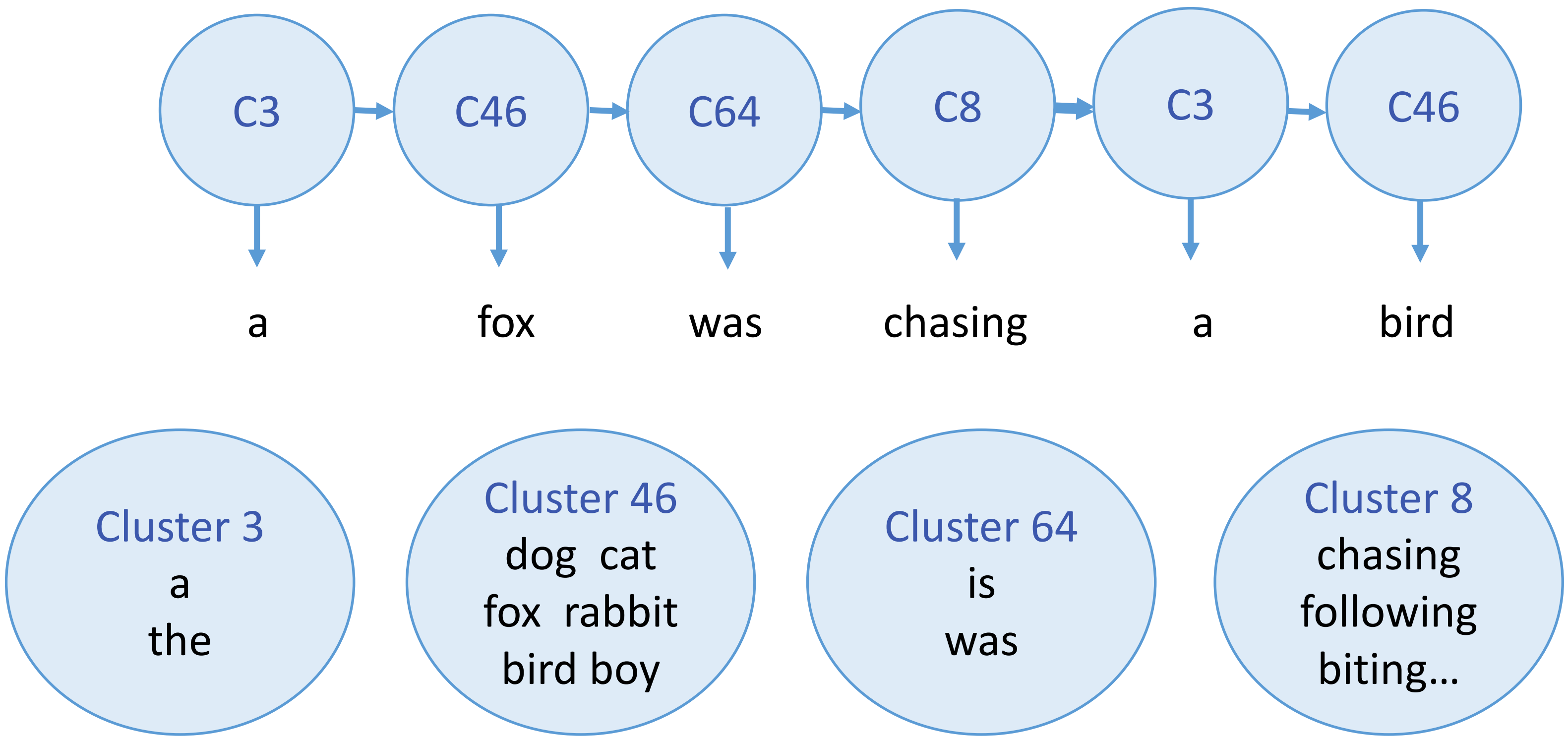
Motivation example

- Assume every word belongs to a cluster
  - “the boy is following a rabbit”



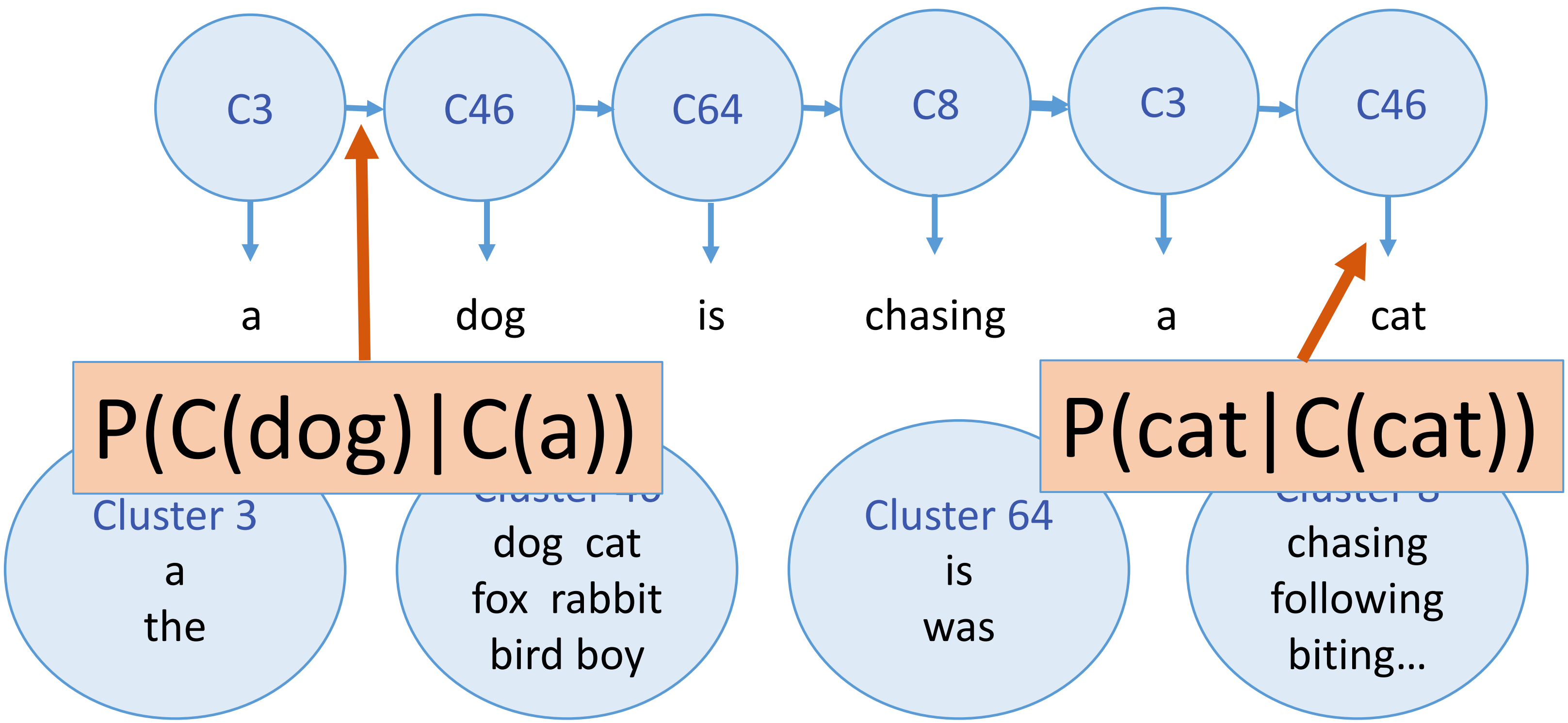
Motivation example

- Assume every word belongs to a cluster
  - "a fox was chasing a bird"



Brown Clustering

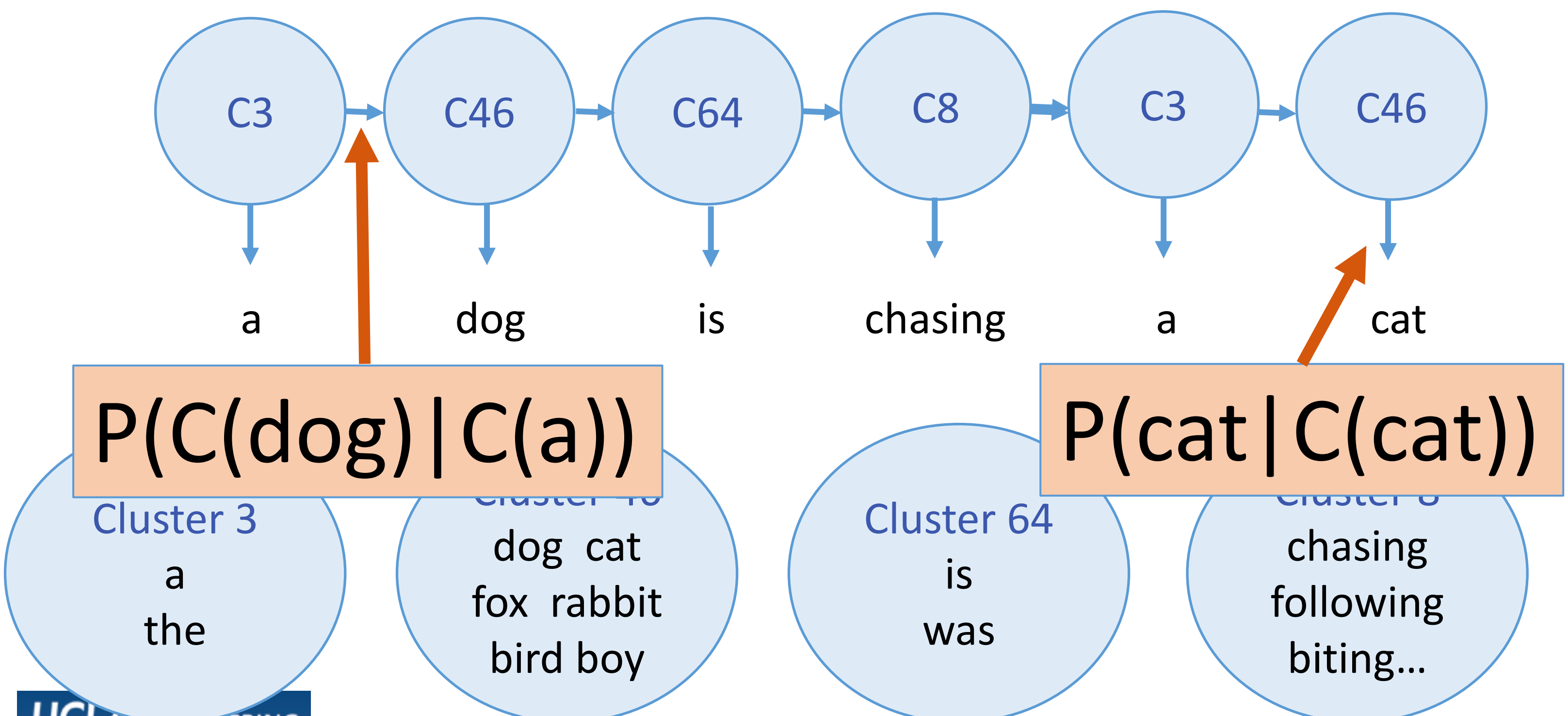
- Let $C(w)$ denote the cluster that w belongs to
  - “a dog is chasing a cat”



Brown clustering model

- P("a dog is chasing a cat")

  = P(C("a")|C_0) P(C("dog")|C("a")) P(C("dog")|C("a"))…
  P("a"|C("a"))P("dog"|C("dog"))…



Brown clustering model

- P("a dog is chasing a cat")

  = P(C("a")|C_0) P(C("dog")|C("a")) P(C("dog")|C("a"))…
  P("a"|C("a"))P("dog"|C("dog"))...

- In general

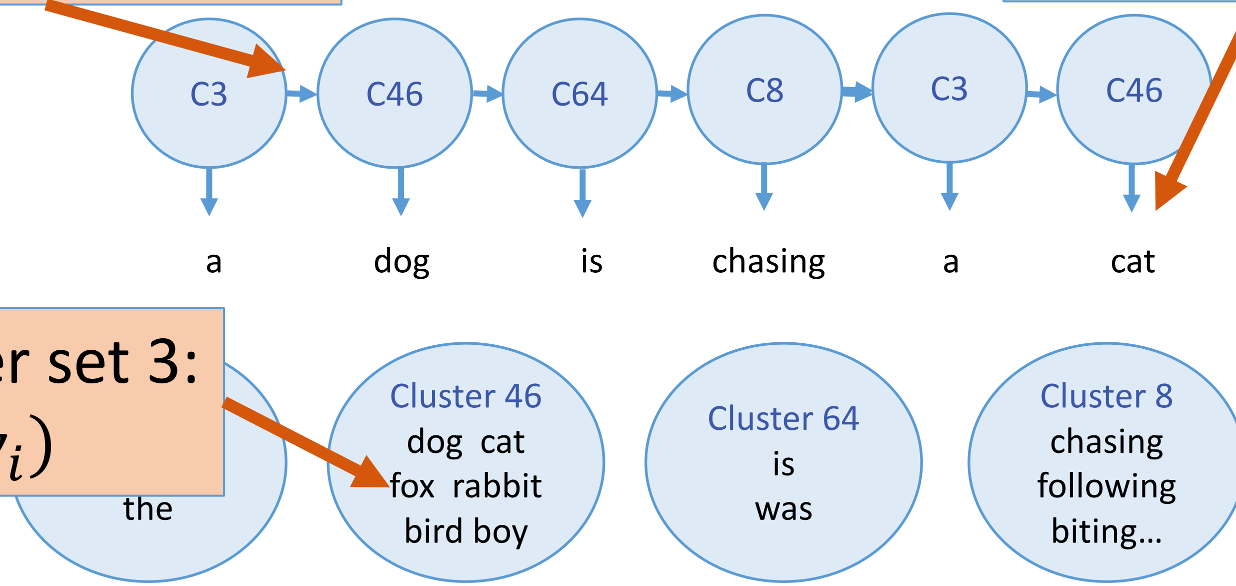
$$
\begin{aligned}
&P(w_0, w_1, w_2, \ldots, w_n) \\
&= P(C(w_1) \mid C(w_0))P(C(w_2)|C(w_1)) \ldots P(C(w_n) \mid C(w_{n-1})) \\
&\quad P(w_1|C(w_1)P(w_2|C(w_2)) \ldots P(w_n|C(w_n)) \\
&= \Pi_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{P}(C(\mathrm{w}_i) \mid C(w_{i-1}))P(w_i \mid C(w_i))
\end{aligned}
$$

Model parameters

$$P(w_0, w_1, w_2, \ldots, w_n) = \Pi_{i=1}^{n} P(C(w_i) \mid C(w_{i-1}))P(w_i \mid C(w_i))$$

Parameter set 1: $P(C(w_i)|C(w_{i-1}))$

Parameter set 2: $P(w_i|C(w_i))$



Parameter set 3: $C(w_i)$

Model parameters

$P(w_0, w_1, w_2, \dots, w_n)$
$= \Pi_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{P}(\, C(\mathrm{w}_i) \mid C(w_{i-1})\,) P(w_i \mid C(w_i))$

- A vocabulary set W
- A function $C: W \rightarrow \{1, 2, 3, \dots k\}$
  - A partition of vocabulary into k classes
- Conditional probability $P(c' \mid c)$ for $c, c' \in \{1, \dots, k\}$
- Conditional probability $P(w \mid c)$ for $c, c' \in \{1, \dots, k\}, w \in c$

θ represents the set of conditional probability parameters
C represents the clustering

Log likelihood

$$\text{LL}(\theta, C) = \log P(w_0, w_1, w_2, \dots, w_n \mid \theta, C)$$
$$= \log \Pi_{i=1}^{n} \text{P}(C(\text{w}_i) \mid C(w_{i-1}))P(w_i \mid C(w_i))$$
$$= \sum_{i=1}^{n} [\log \text{P}(C(\text{w}_i) \mid C(w_{i-1})) + \log P(w_i \mid C(w_i))]$$

- Maximizing $\text{LL}(\theta, C)$ can be done by alternatively update θ and C
  1. $\max_{\theta \in \Theta} LL(\theta, C)$
  2. $\max_{C} LL(\theta, C)$

$$\max_{\theta \in \Theta} LL(\theta, C)$$

$$\mathrm{LL}(\theta, C) = \log P(w_0, w_1, w_2, \ldots, w_n \mid \theta, C)$$
$$= \log \Pi_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{P}(C(\mathrm{w}_i) \mid C(w_{i-1}))P(w_i \mid C(w_i))$$
$$= \sum_{\mathrm{i=1}}^{\mathrm{n}} [\log \mathrm{P}(C(\mathrm{w}_i) \mid C(w_{i-1})) + \log P(w_i \mid C(w_i))]$$

- $P(c' \mid c) = \frac{\#(c',c)}{\#c}$

- $P(w \mid c) = \frac{\#(w,c)}{\#c}$

$$\max_{C} LL(\theta, C)$$

$$\max_{C} \sum_{\mathrm{i}=1}^{\mathrm{n}} [\log \mathrm{P}(\, C(\mathrm{w}_i) \mid C(w_{i-1})\,) + \log P(w_i \mid C(w_i))\,]$$

$$= \mathrm{n} \sum_{c=1}^{k} \sum_{c'=1}^{k} p(c, c') \log \frac{p(c,c')}{p(c)p(c')} + G$$

where G is a constant

- Here,

$$p(c, c') = \frac{\#(c,c')}{\sum_{c,c'} \#(c,c')} \quad , \quad p(c) = \frac{\#(c)}{\sum_{c} \#(c)}$$

- $\frac{p(c,c')}{p(c)p(c')} = \frac{p(\,c|c'\,)}{p(c)}$ (mutual information)

$$\max_{C} LL(\theta, C)$$

$$\max_{C} \sum_{\mathrm{i}=1}^{\mathrm{n}} [\log \mathrm{P}(C(\mathrm{w}_i) \mid C(w_{i-1})) + \log P(w_i \mid C(w_i))]$$

$$= \mathrm{n} \sum_{c=1}^{k} \sum_{c'=1}^{k} p(c, c') \log \frac{p(c,c')}{p(c)p(c')} + G$$

Algorithm 1

- Start with |V| clusters
  each word is in its own cluster
- The goal is to get k clusters
- We run |V|-k merge steps:
  - Pick 2 clusters and merge them
  - Each step pick the merge maximizing $LL(\theta, C)$
- Cost? (can be improved to $O(|V|^3)$)
  O(|V|-k) $O(|V|^2)$ $O(|V|^2)$ = $O(|V|^5)$
  #Iters #pairs compute LL

Algorithm 2

- m : a hyper-parameter, sort words by frequency
- Take the top m most frequent words, put each of them in its own cluster $c_1, c_2, c_3, \dots c_m$
- For $i = (m+1) \dots |V|$
  - Create a new cluster c_{m+1} (we have m+1 clusters)
  - Choose two cluster from m+1 clusters based on $LL(\theta, C)$ and merge ⇒ back to m clusters
- Carry out (m-1) final merges ⇒ full hierarchy
- Running time $O(|V|m^2 + n)$, n=#words in corpus

Example clusters (Brown+1992)

Friday Monday Thursday Wednesday Tuesday Saturday Sunday weekends Sundays Saturdays
June March July April January December October November September August
people guys folks fellows CEOs chaps doubters commies unfortunates blokes
down backwards ashore sideways southward northward overboard aloft downwards adrift
water gas coal liquid acid sand carbon steam shale iron
great big vast sudden mere sheer gigantic lifelong scant colossal
man woman boy girl lawyer doctor guy farmer teacher citizen
American Indian European Japanese German African Catholic Israeli Italian Arab
pressure temperature permeability density porosity stress velocity viscosity gravity tension
mother wife father son husband brother daughter sister boss uncle
machine device controller processor CPU printer spindle subsystem compiler plotter
John George James Bob Robert Paul William Jim David Mike
anyone someone anybody somebody
feet miles pounds degrees inches barrels tons acres meters bytes
director chief professor commissioner commander treasurer founder superintendent dean custodian
liberal conservative parliamentary royal progressive Tory provisional separatist federalist PQ
had hadn't hath would've could've should've must've might've
asking telling wondering instructing informing kidding reminding bothering thanking deposing
that tha theat
head body hands eyes voice arm seat eye hair mouth

Example Hierarchy(Miller+2004)

lawyer	1000001101000
newspaperman	100000110100100
stewardess	100000110100101
toxicologist	10000011010011
slang	1000001101010
babysitter	100000110101100
conspirator	1000001101011010
womanizer	1000001101011011
mailman	10000011010111
salesman	100000110110000
bookkeeper	1000001101100010
troubleshooter	10000011011000110
bouncer	10000011011000111
technician	1000001101100100
janitor	1000001101100101
saleswoman	1000001101100110
...	
Nike	1011011100100101011100
Maytag	10110111001001010111010
Generali	10110111001001010111011
Gap	1011011100100101011110
Harley-Davidson	10110111001001010111110
Enfield	101101110010010101111110
genus	101101110010010101111111
Microsoft	10110111001001011000
Ventritex	101101110010010110010
Tractebel	1011011100100101100110
Synopsys	1011011100100101100111
WordPerfect	1011011100100101101000
....	
John	101110010000000000
Consuelo	101110010000000001
Jeffrey	101110010000000010
Kenneth	10111001000000001100
Phillip	101110010000000011010

UCLA ENGINEER
Computer Science