



**SI 630**

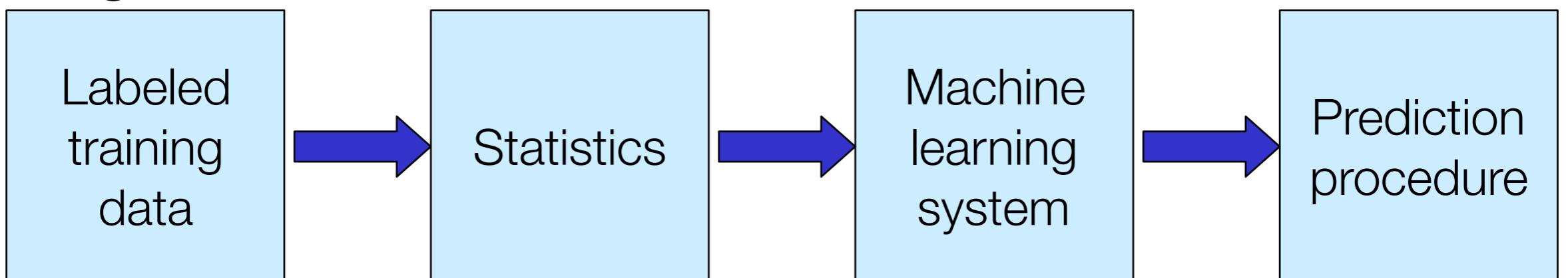
# Natural Language Processing: Algorithms and People

Lecture 7: Unsupervised Learning  
Feb. 19, 2020



# Supervised Learning

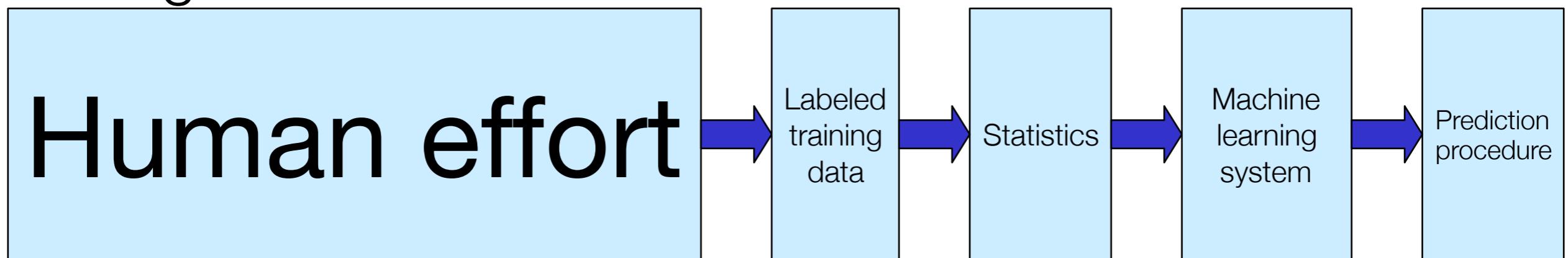
Training:



- Standard statistical systems use a **supervised** paradigm.

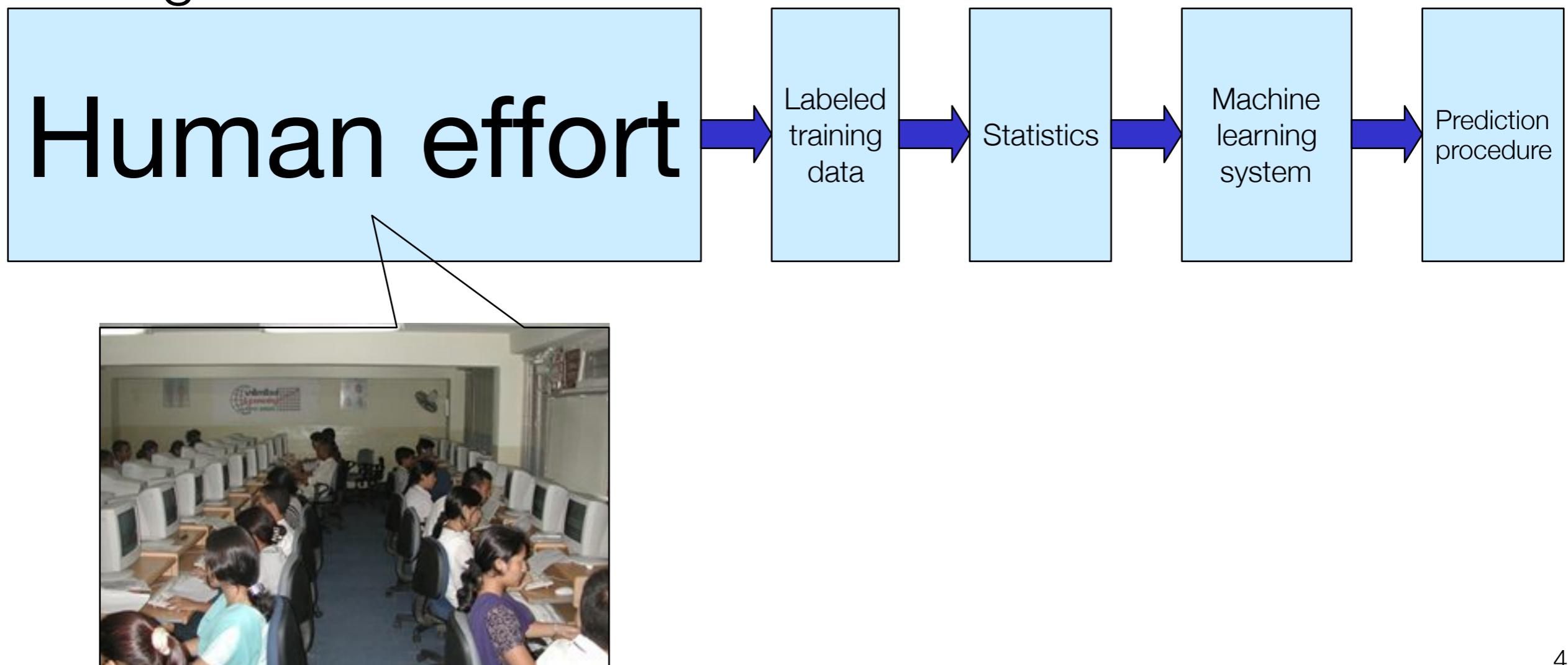
# The real story: Annotating labeled data is labor-intensive!!!

Training:



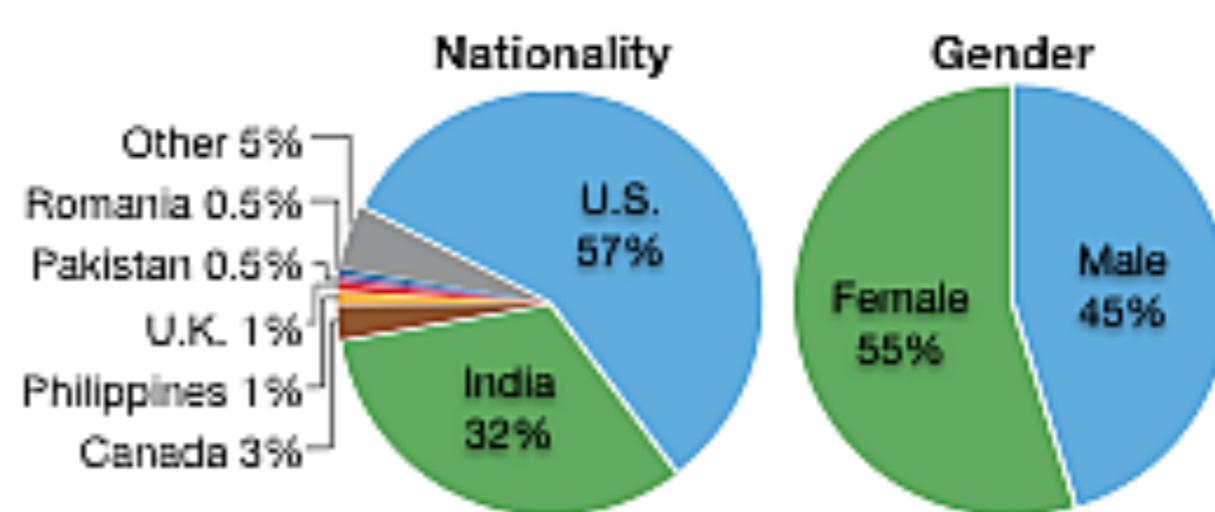
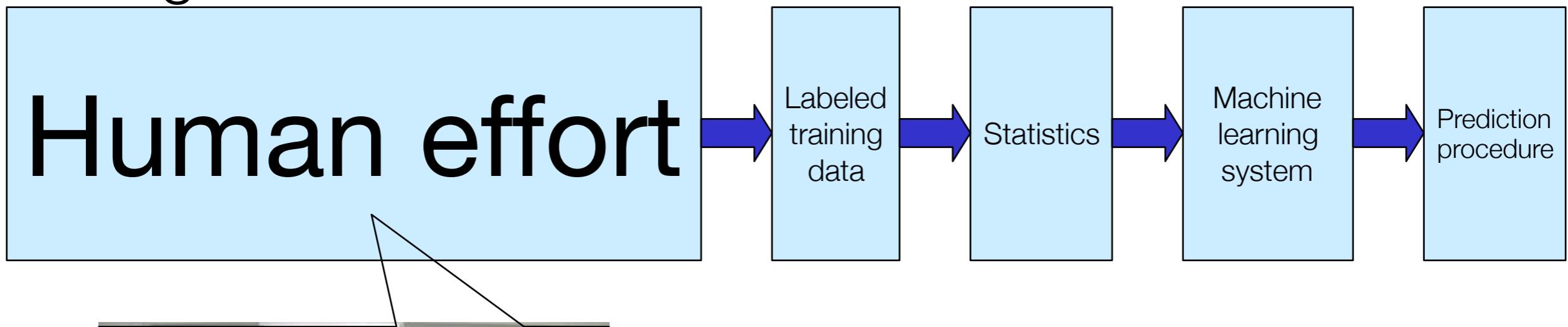
# The real story: Annotating labeled data is labor-intensive!!!

Training:



# The real story: Annotating labeled data is labor-intensive!!!

Training:



Ross et al. (2011)

# The real story

- Reliance on training data also means that moving to a new language, domain, or even genre can be difficult.

# The real story

- Reliance on training data also means that moving to a new language, domain, or even genre can be difficult.
- But unlabeled data is cheap!

# The real story

- Reliance on training data also means that moving to a new language, domain, or even genre can be difficult.
- But unlabeled data is cheap!
- It would be nice to use the unlabeled data directly to learn the labelings you want in your model.

# The real story

- Reliance on training data also means that moving to a new language, domain, or even genre can be difficult.
- But unlabeled data is cheap!
- It would be nice to use the unlabeled data directly to learn the labelings you want in your model.
- Today we'll look at methods for doing this, which is called unsupervised learning

# Motivating Example

# Motivating Example



Donald J. Trump  @realDonaldTrump

The long anticipated release of the [#JFKFiles](#) will take place tomorrow. So interesting!

7:56 PM - Oct 25, 2017

19,252 replies 34,465 retweets 121,513 likes

@REALDONALDTRUMP

# Motivating Example

Donald J. Trump   
@realDonaldTrump

The long anticipated release of the [#JFKFiles](#) will take place tomorrow. So interesting!

7:56 PM - Oct 25, 2017

19,252 replies 34,465 retweets 121,513 likes

@REALDONALDTRUMP

- JFK Assassination Records - 2017 Additional Documents Release. July 24, 2017: 3,810 documents
  - October 26, 2017: 2,891 documents
  - November 3, 2017: 676 documents
  - November 9, 2017: 13,213 documents
  - November 17, 2017: 10,744 documents
  - December 15, 2017: 3,539 documents

# Motivating Example

 Donald J. Trump   
@realDonaldTrump

The long anticipated release of the #JFKFiles will take place tomorrow. So interesting!

7:56 PM - Oct 25, 2017

19,252 34,465 121,513

@REALDONALDTRUMP

- JFK Assassination Records - 2017 Additional Documents Release. July 24, 2017: 3,810 documents
  - October 26, 2017: 2,891 documents
  - November 3, 2017: 676 documents
  - November 9, 2017: 13,213 documents
  - November 17, 2017: 10,744 documents
  - December 15, 2017: 3,539 documents



# Motivating Example

 Donald J. Trump   
@realDonaldTrump

The long anticipated release of the #JFKFiles will take place tomorrow. So interesting!

7:56 PM - Oct 25, 2017

19,252 34,465 121,513

@REALDONALDTRUMP

- JFK Assassination Records - 2017 Additional Documents Release. July 24, 2017: 3,810 documents
  - October 26, 2017: 2,891 documents
  - November 3, 2017: 676 documents
  - November 9, 2017: 13,213 documents
  - November 17, 2017: 10,744 documents
  - December 15, 2017: 3,539 documents
- 337,626 pages of documents!  
How can we make sense of this quickly as a journalist?



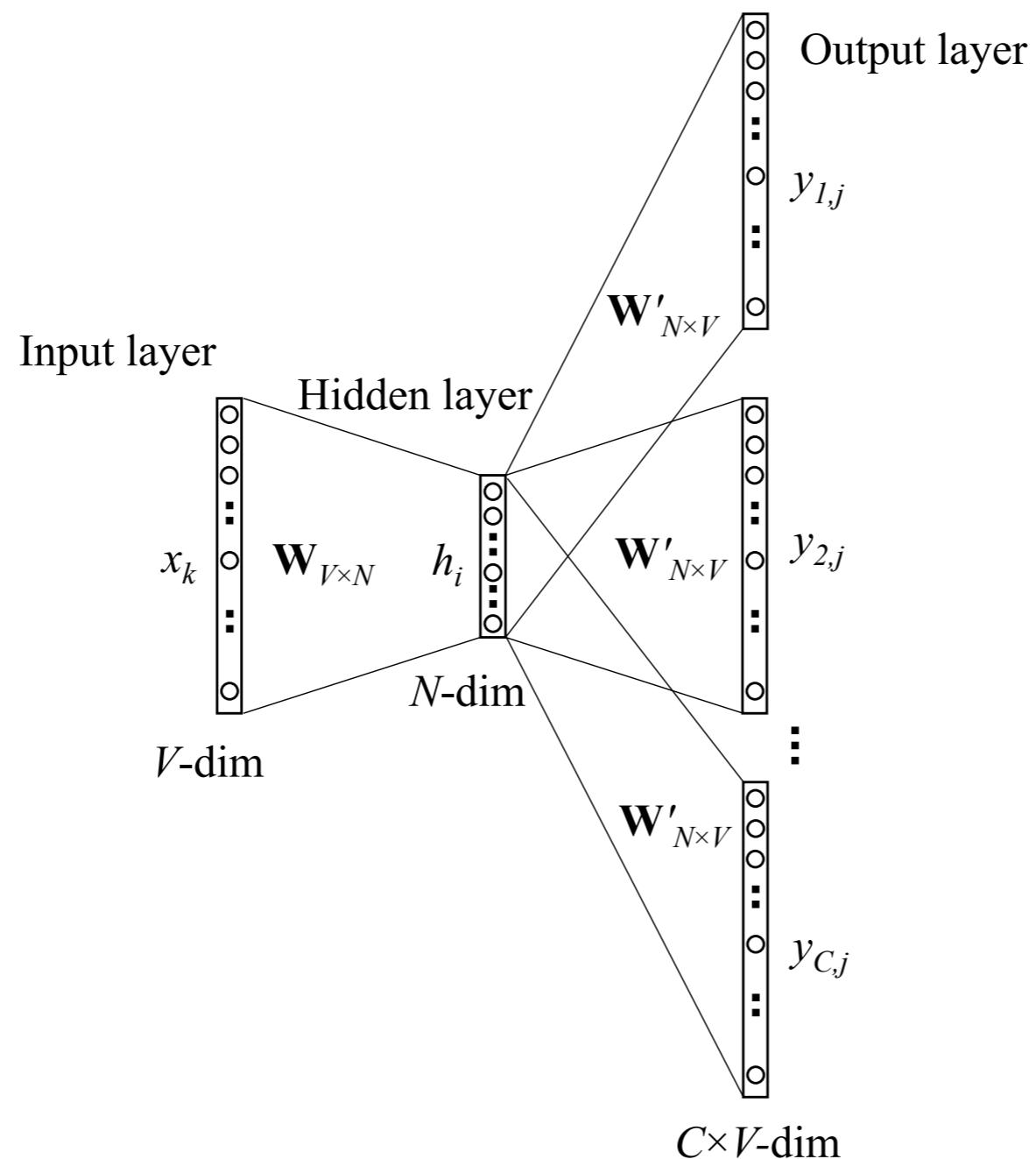
# Today's plan

- Look at three kinds of unsupervised learning for NLP
- Start with words and move to documents



# Brown Clustering

# We know how to learn word meaning



# What if we wanted to learn categories of words?

“You shall know a word by the company it keeps”

[Firth 1957]

# What if we wanted to learn categories of words?

“You shall know a word by the company it keeps”

[Firth 1957]

everyone likes \_\_\_\_\_

# What if we wanted to learn categories of words?

“You shall know a word by the company it keeps”

[Firth 1957]

everyone likes \_\_\_\_\_

a bottle of \_\_\_\_\_

is on the table

# What if we wanted to learn categories of words?

“You shall know a word by the company it keeps”

[Firth 1957]

everyone likes \_\_\_\_\_

a bottle of \_\_\_\_\_ is on the table

\_\_\_\_\_ makes you drunk

# What if we wanted to learn categories of words?

“You shall know a word by the company it keeps”

[Firth 1957]

everyone likes	_____	
a bottle of	_____	is on the table
	_____	makes you drunk
a cocktail with	_____	and seltzer

# Brown clustering combines the Distributional Hypothesis with Language Modeling!

- An **agglomerative clustering algorithm** that clusters words based on which words precede or follow them
- These word clusters can be turned into a kind of vector
- We'll give a very brief sketch here.

# Brown Clustering

- Similar to a language model but the **basic unit is a word cluster**
- Intuition is that similar words appear in similar contexts
- Recap: Bigram Language Model:

$$\prod_i^n P(w_i \mid w_{i-1}) \times P(STOP \mid w_n)$$

Maximum likelihood estimate  $\frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$

# Motivating example

# Motivating example

- “a dog is chasing a cat”

# Motivating example

- “a dog is chasing a cat”
  - $P(\text{“a”, “dog”, …, “cat”}) = P(\text{“a”}|\text{START}) \dots P(\text{“cat”}|\text{“a”})$

# Motivating example

- “a dog is chasing a cat”
  - $P(\text{“a”, “dog”, …, “cat”}) = P(\text{“a”}|\text{START}) \dots P(\text{“cat”}|\text{“a”})$
- Assume every word belongs to a cluster

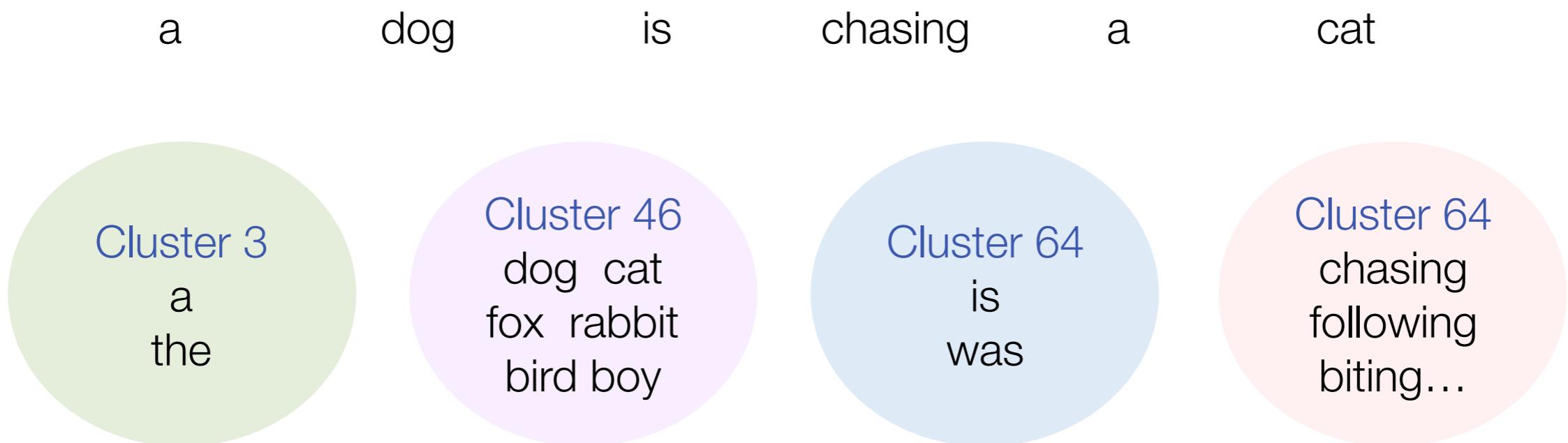
# Motivating example

- “a dog is chasing a cat”
  - $P(\text{"a"}, \text{"dog"}, \dots, \text{"cat"}) = P(\text{"a"}|\text{START}) \dots P(\text{"cat"}|\text{"a"})$
- Assume every word belongs to a cluster



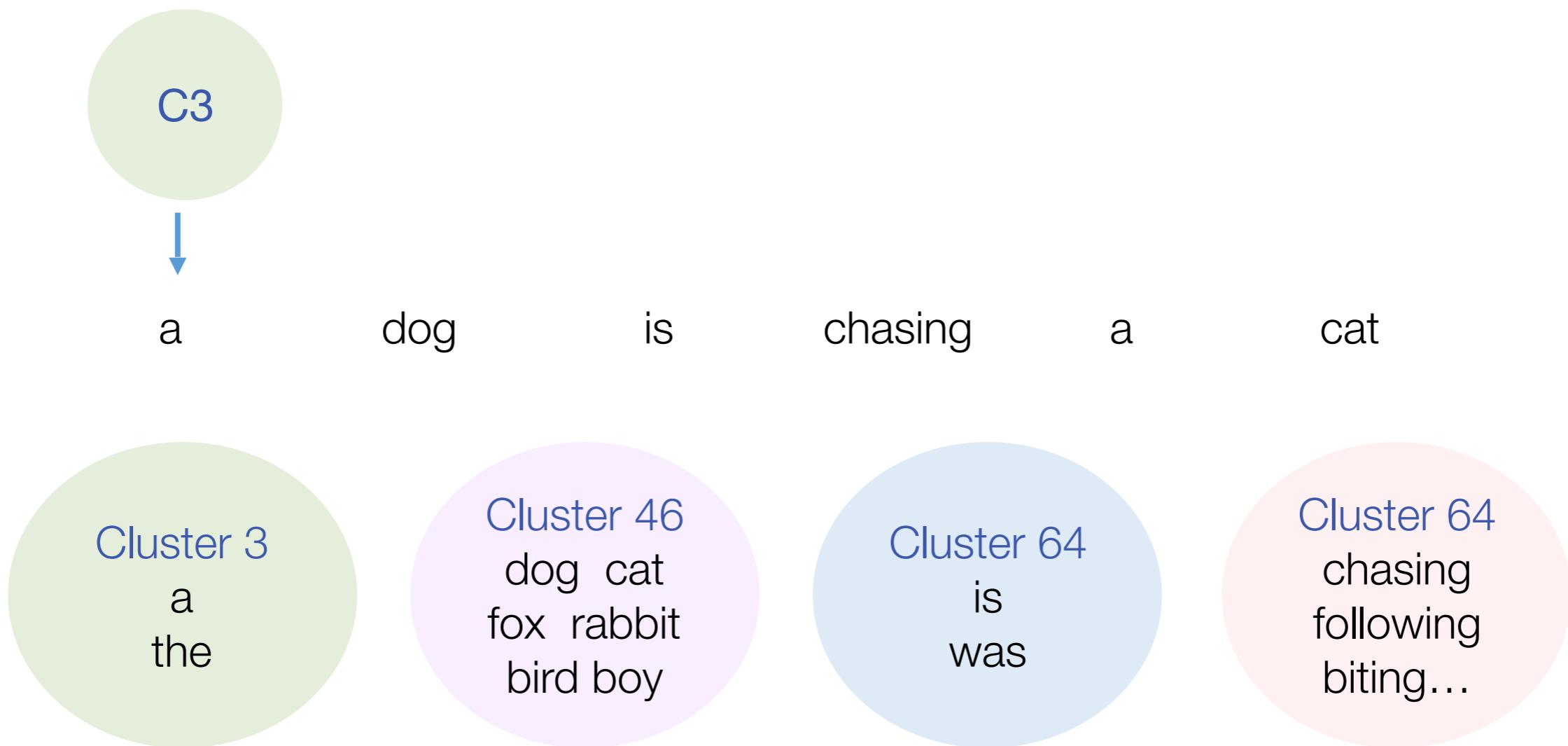
# Motivating example

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



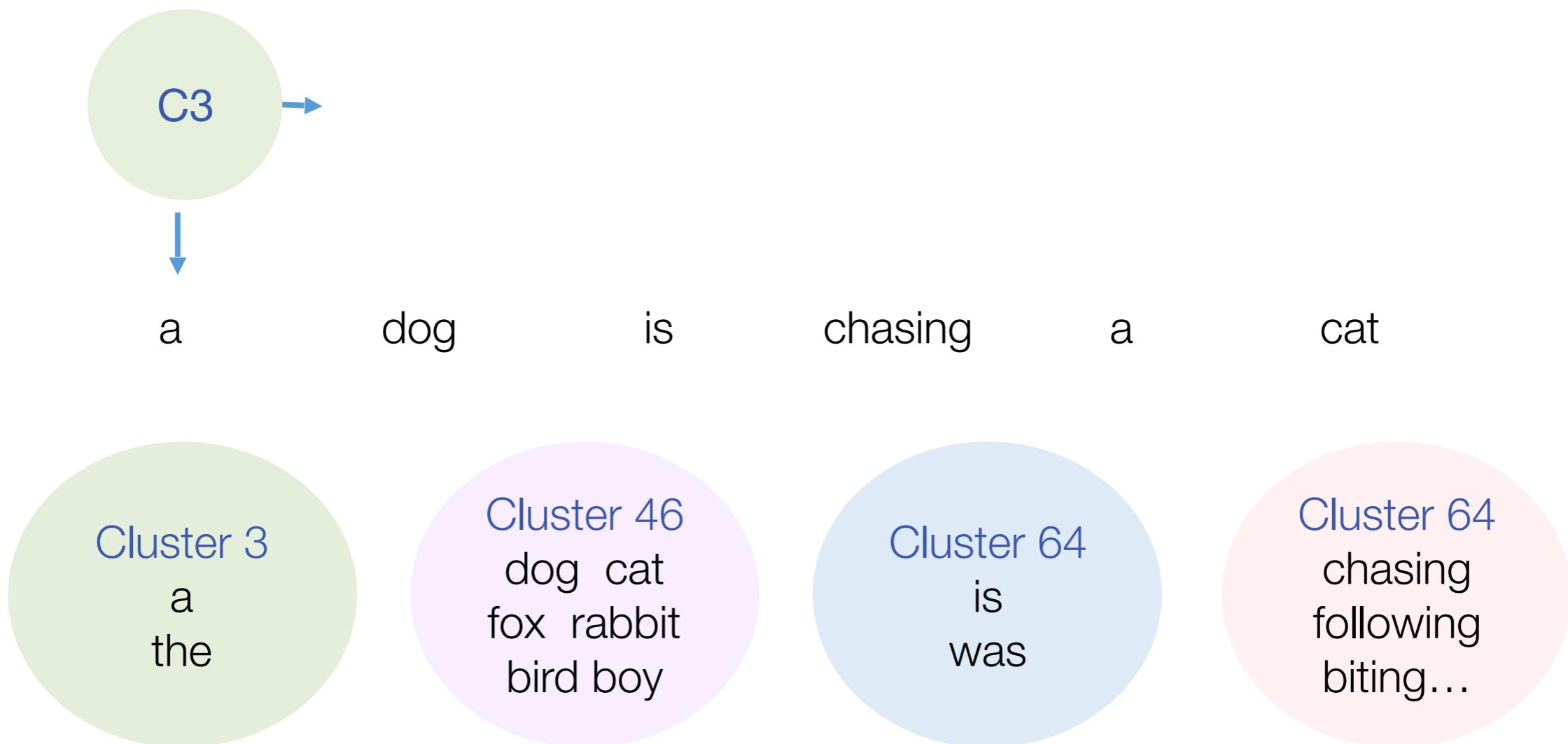
# Motivating example

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



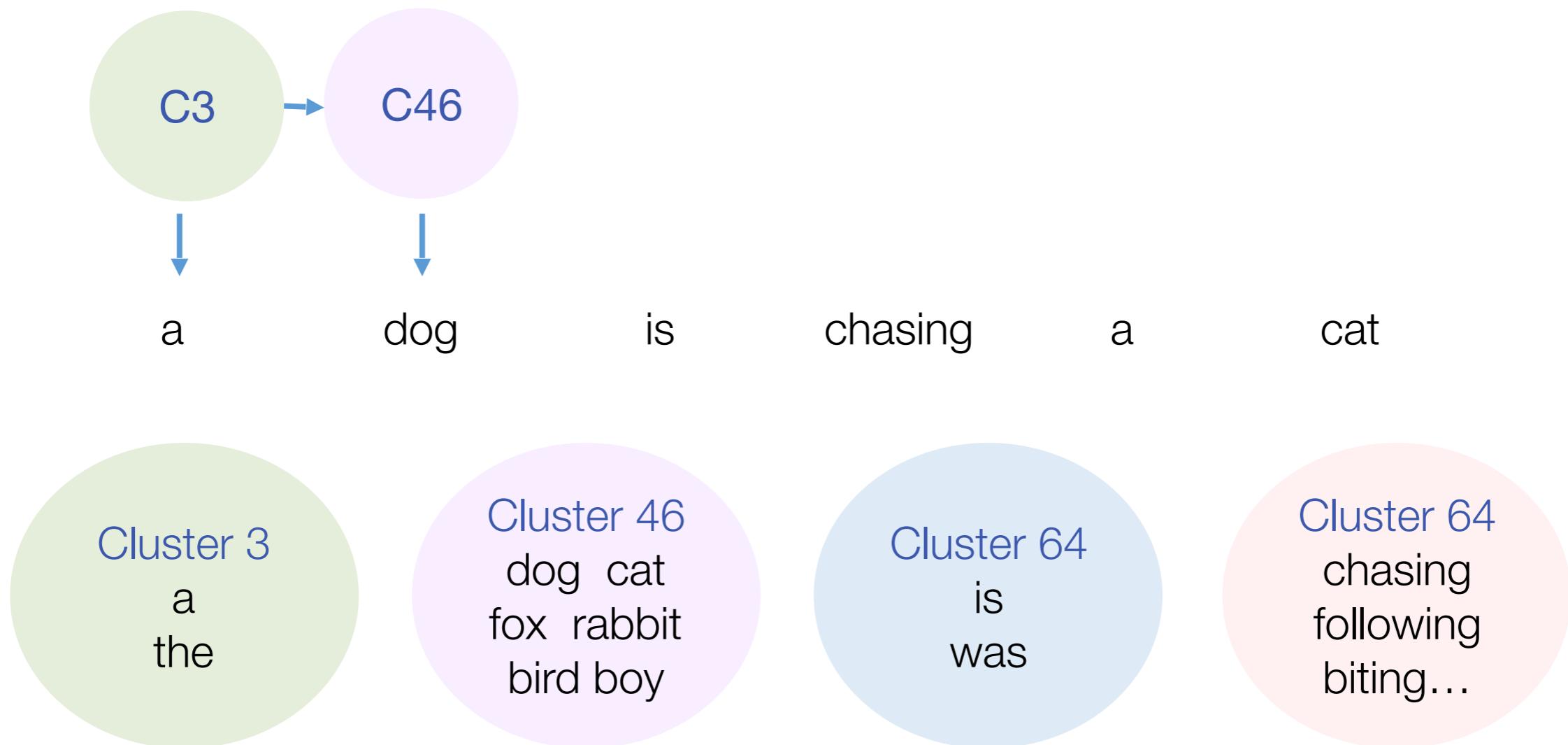
# Motivating example

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



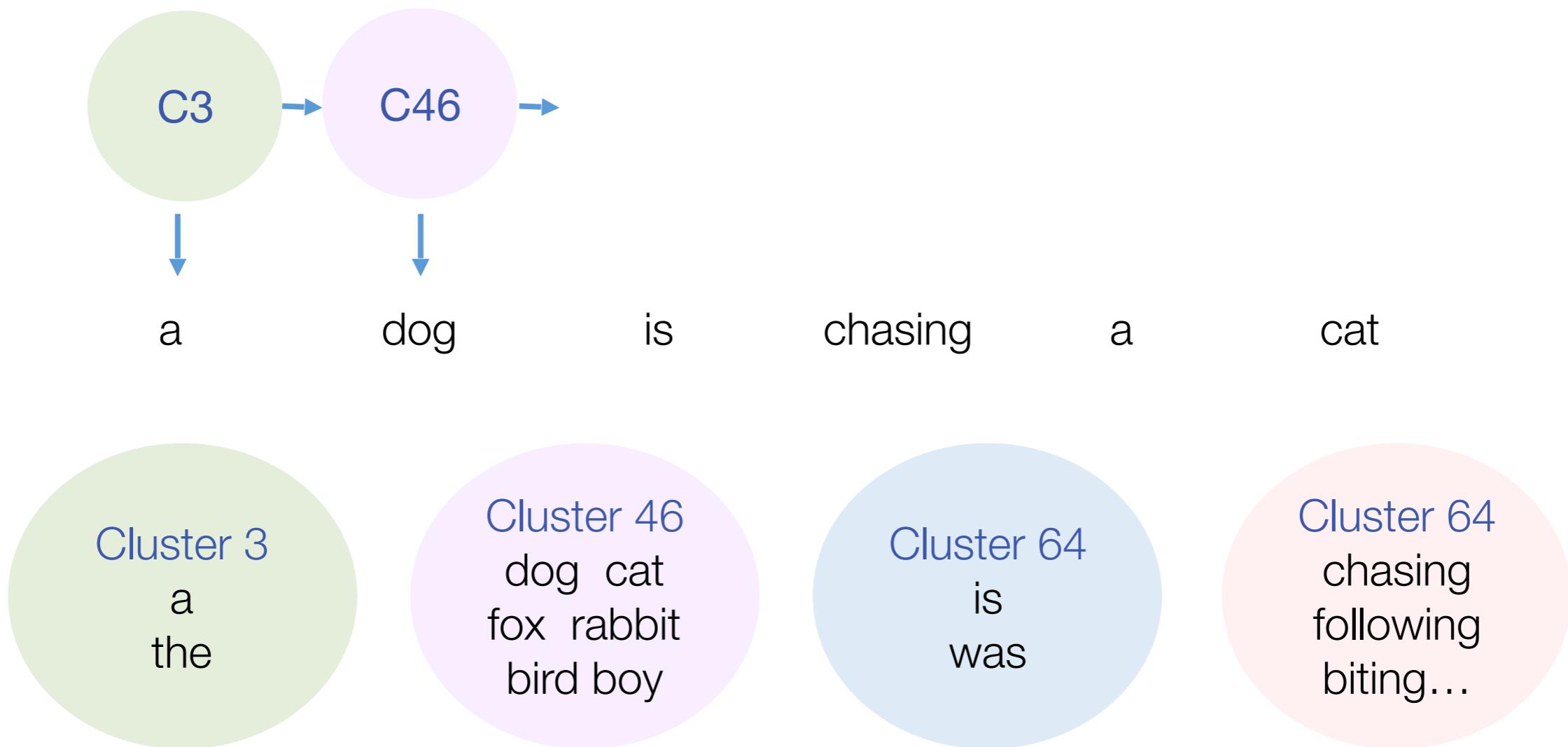
# Motivating example

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



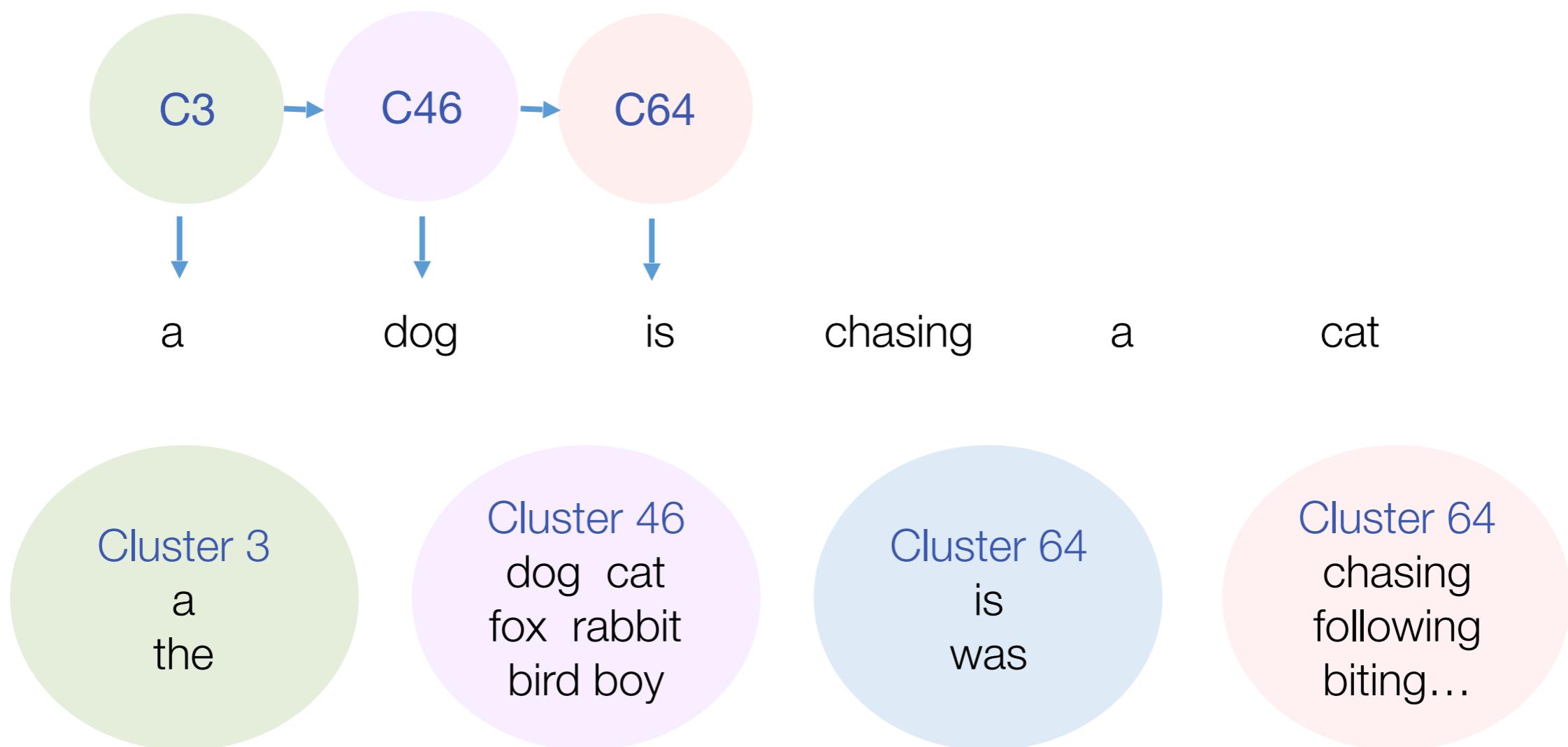
# Motivating example

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



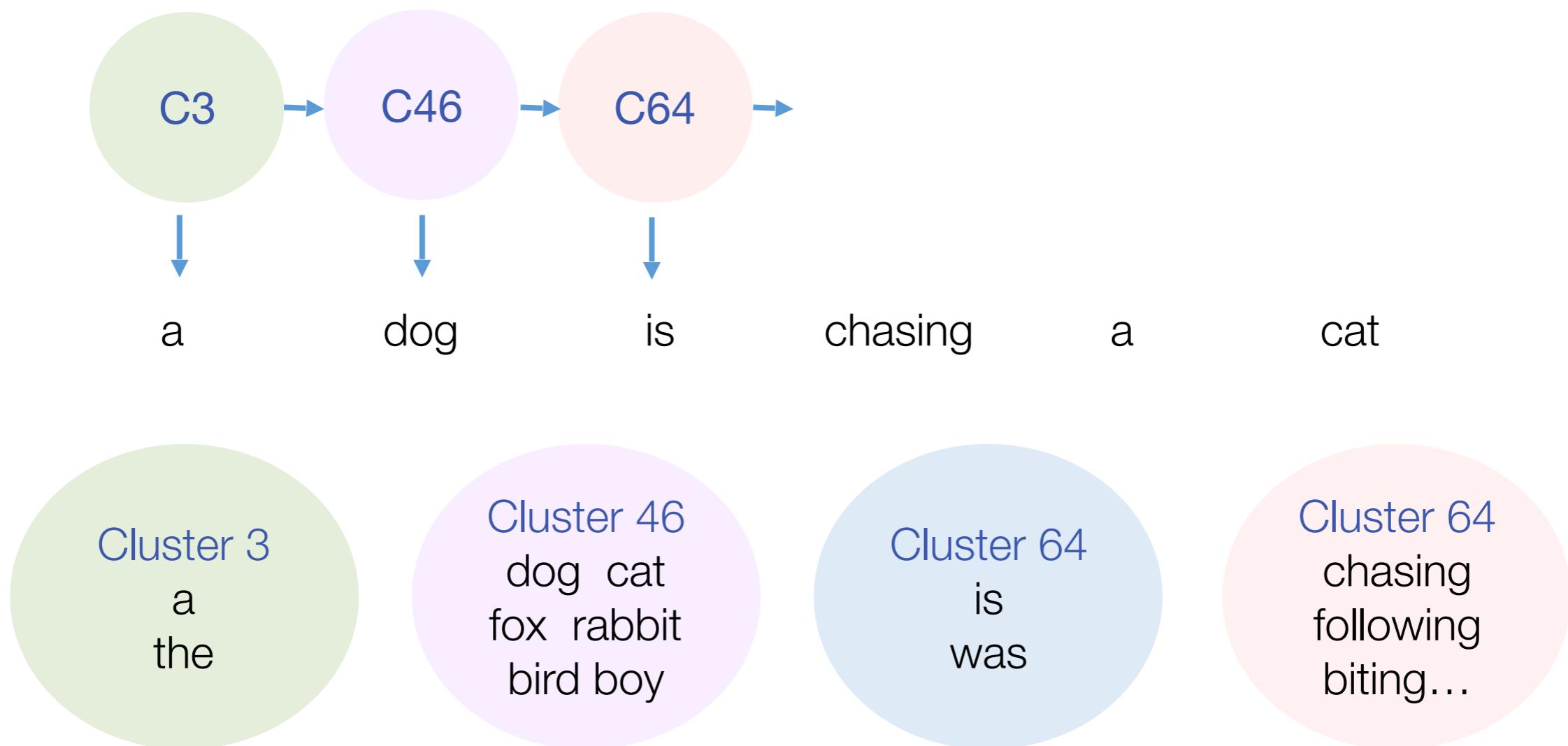
# Motivating example

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



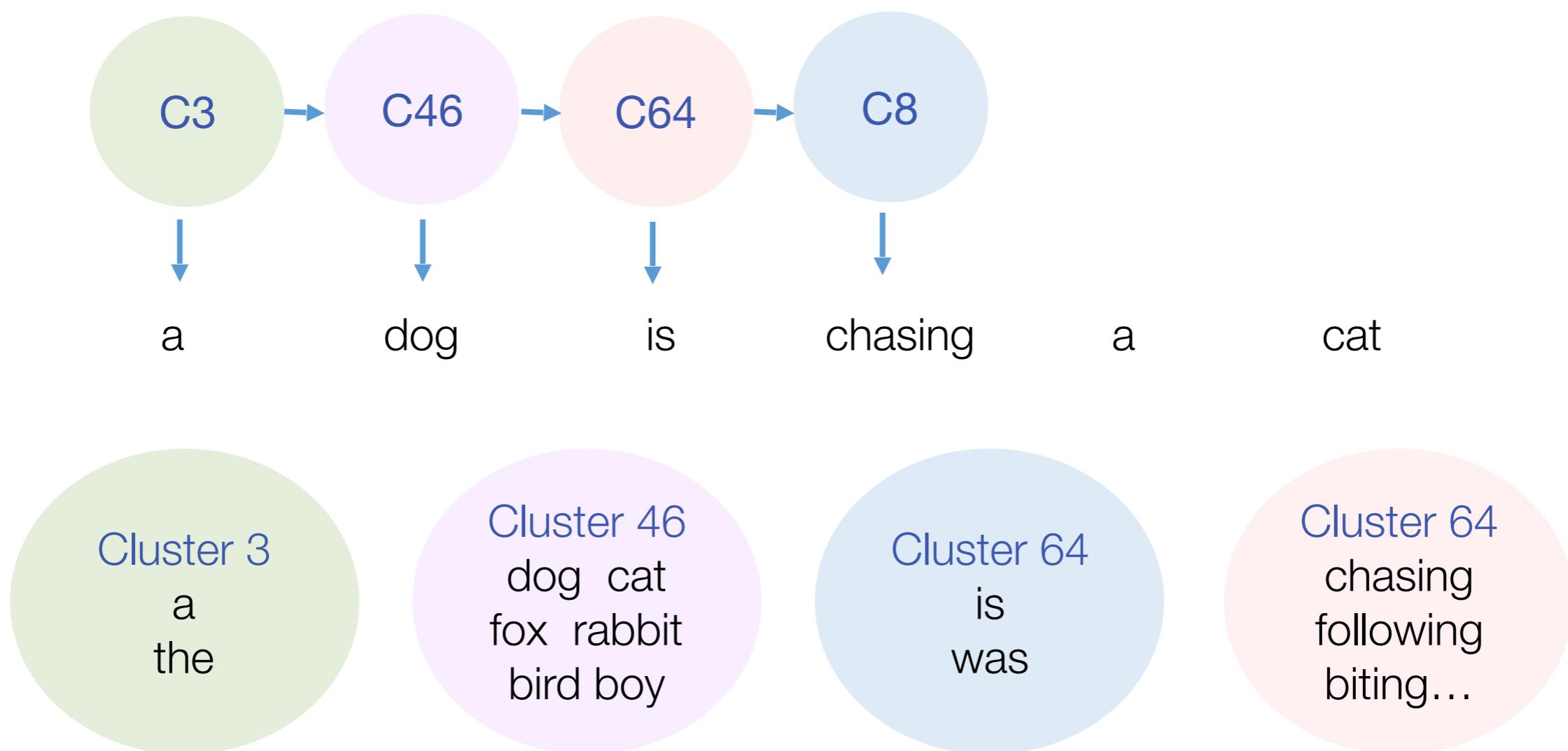
# Motivating example

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



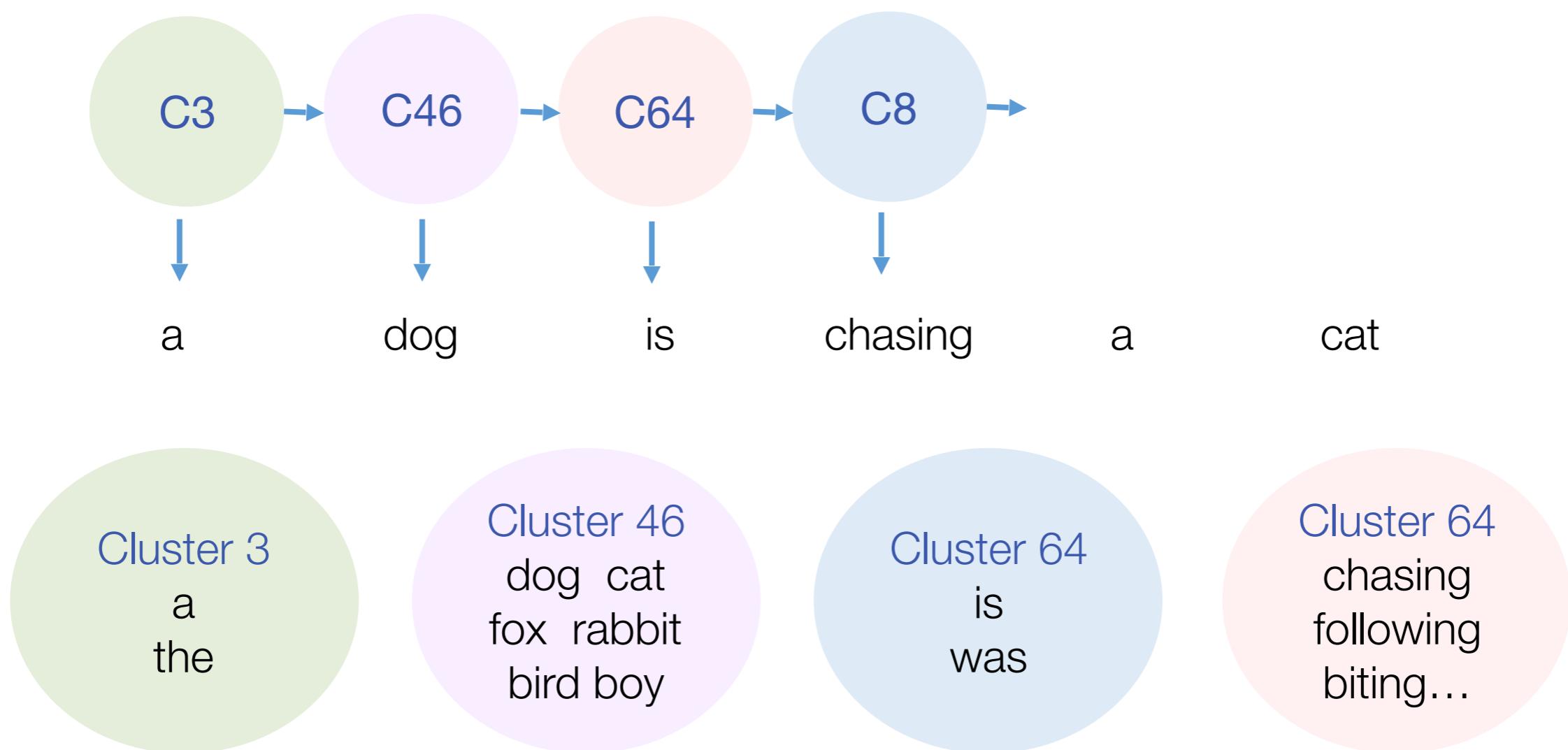
# Motivating example

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



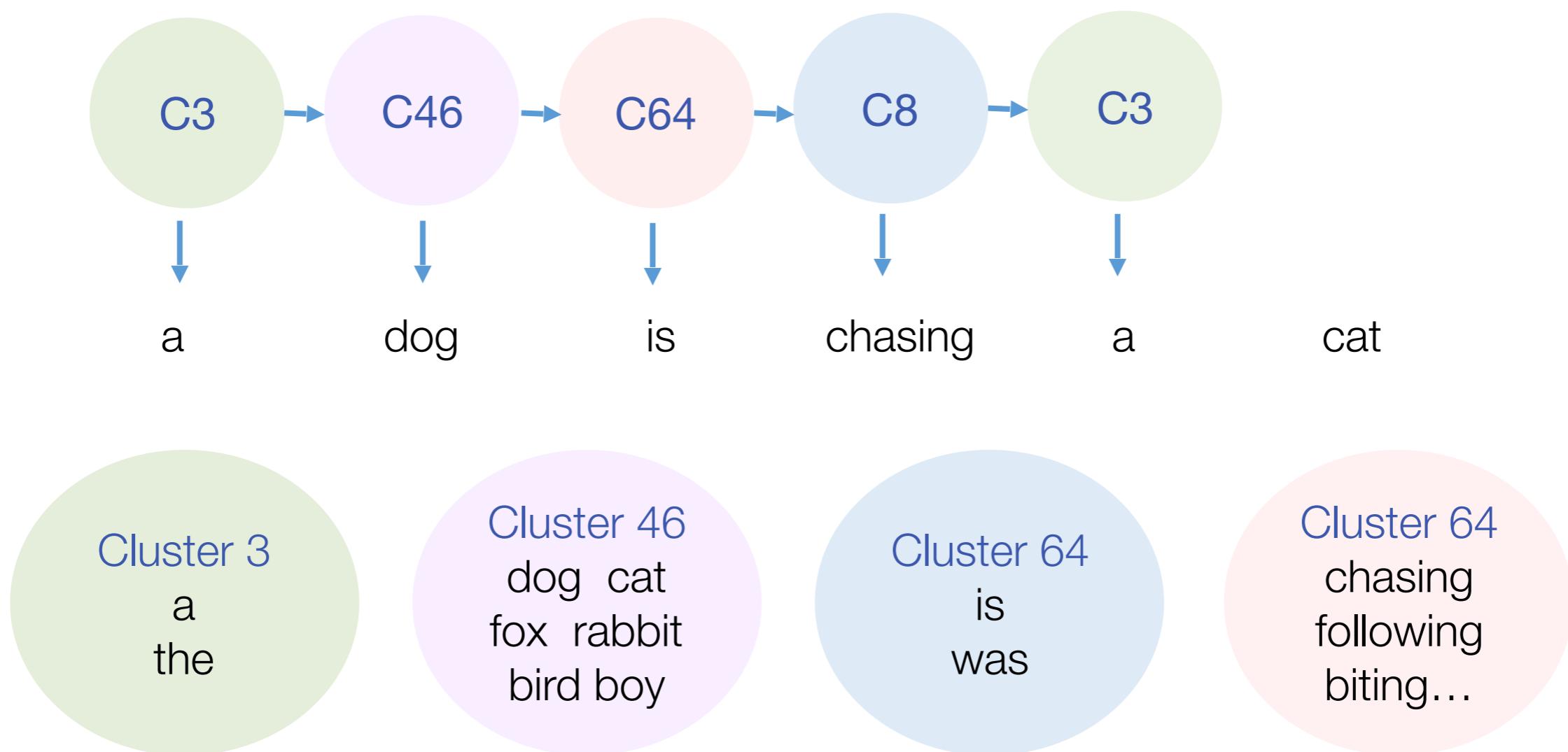
# Motivating example

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



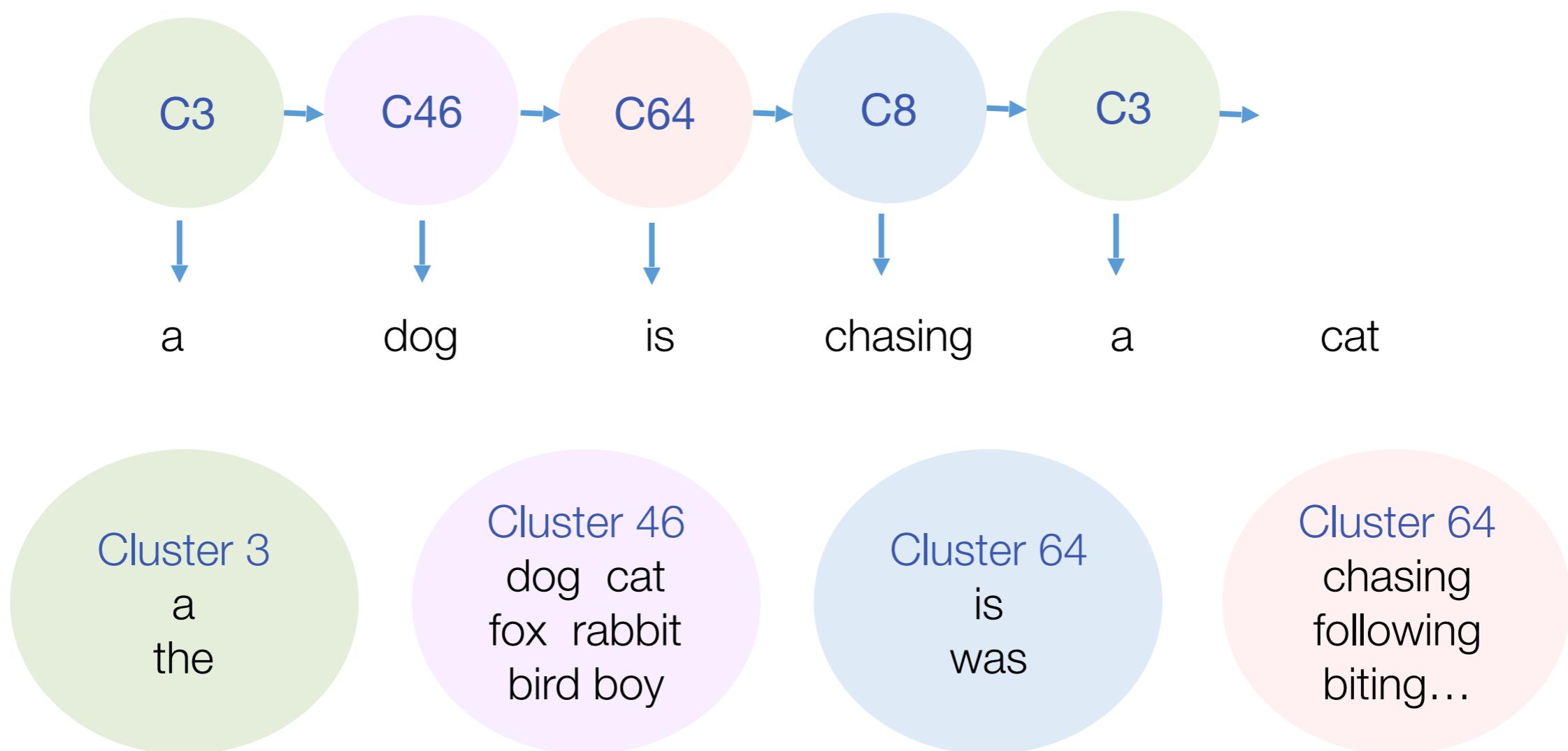
# Motivating example

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



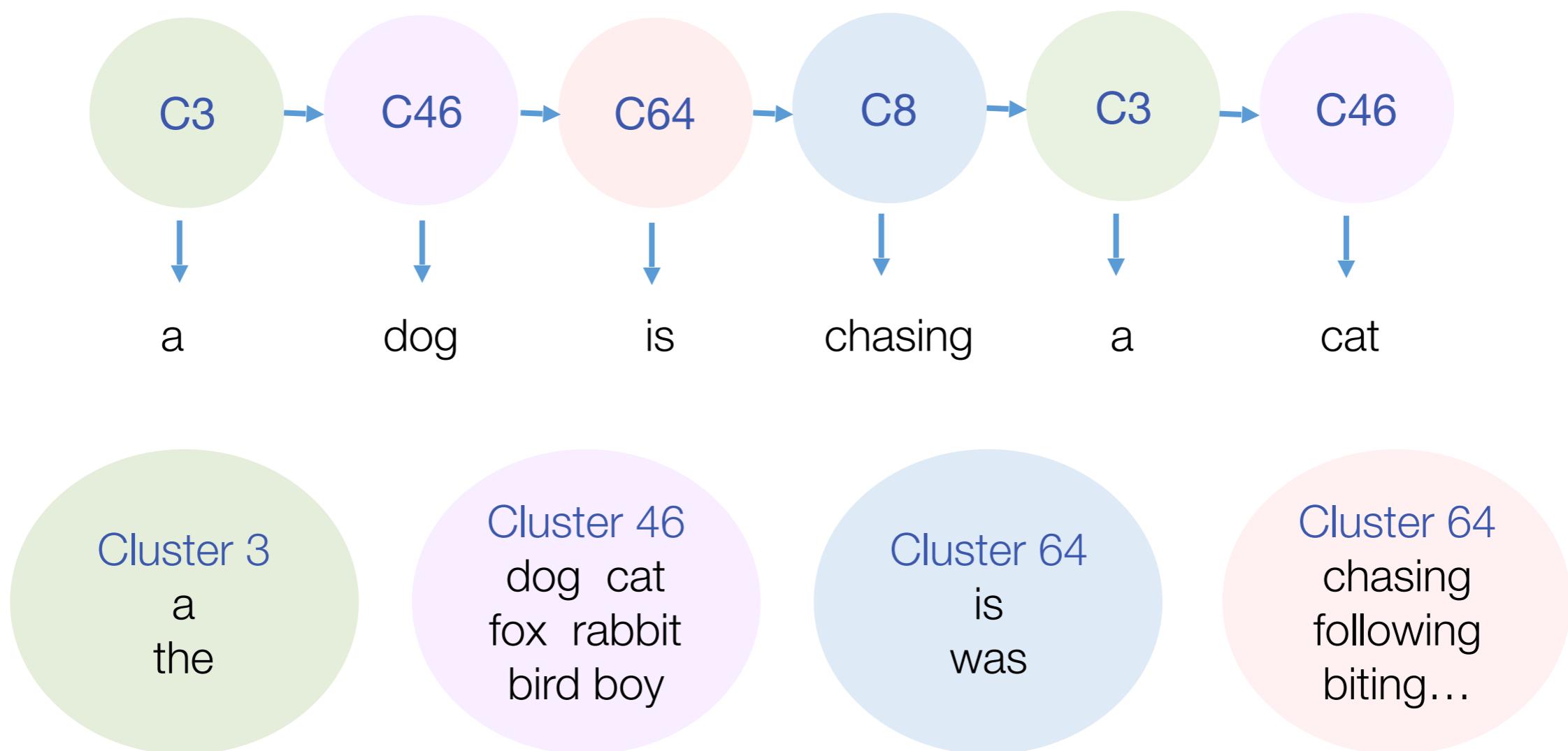
# Motivating example

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



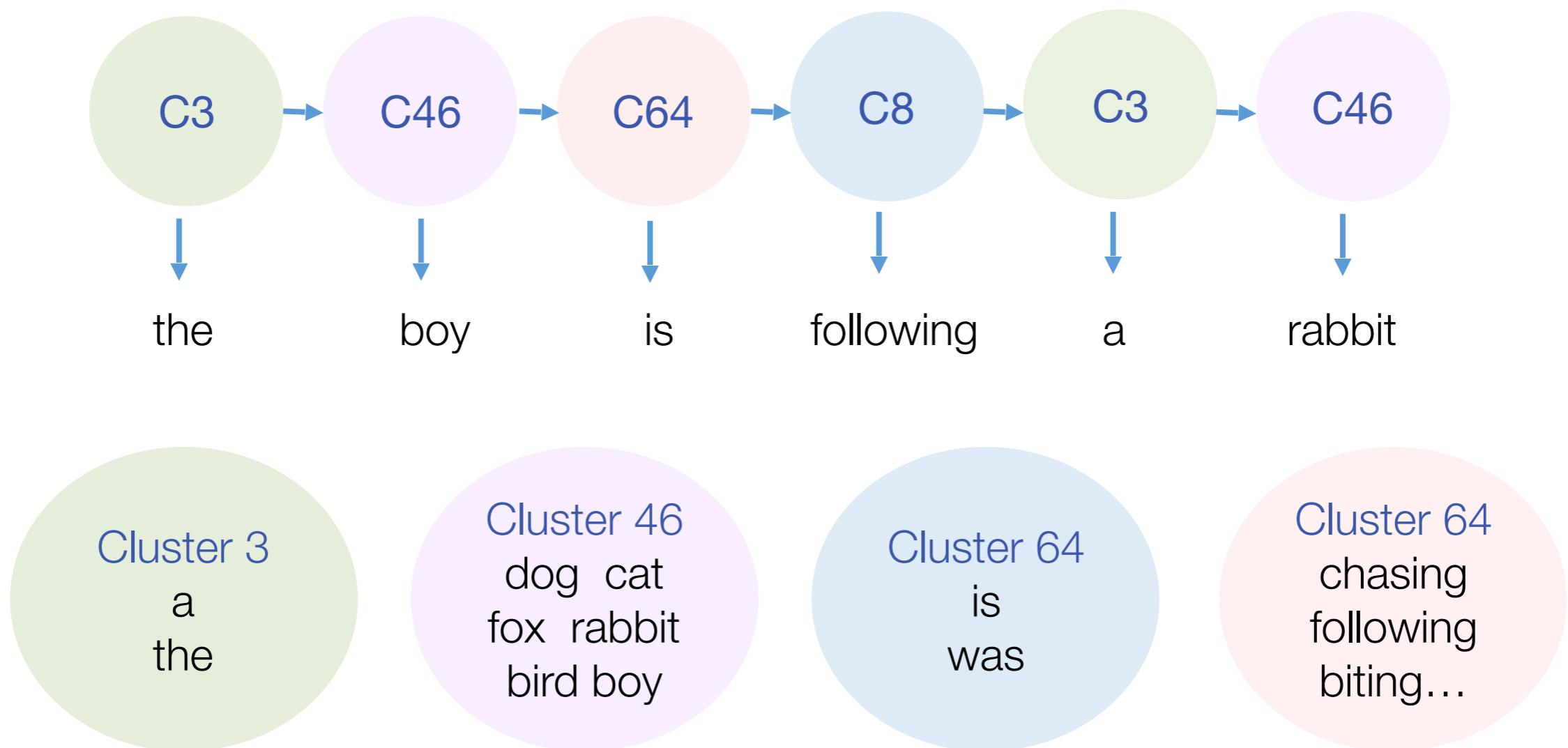
# Motivating example

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



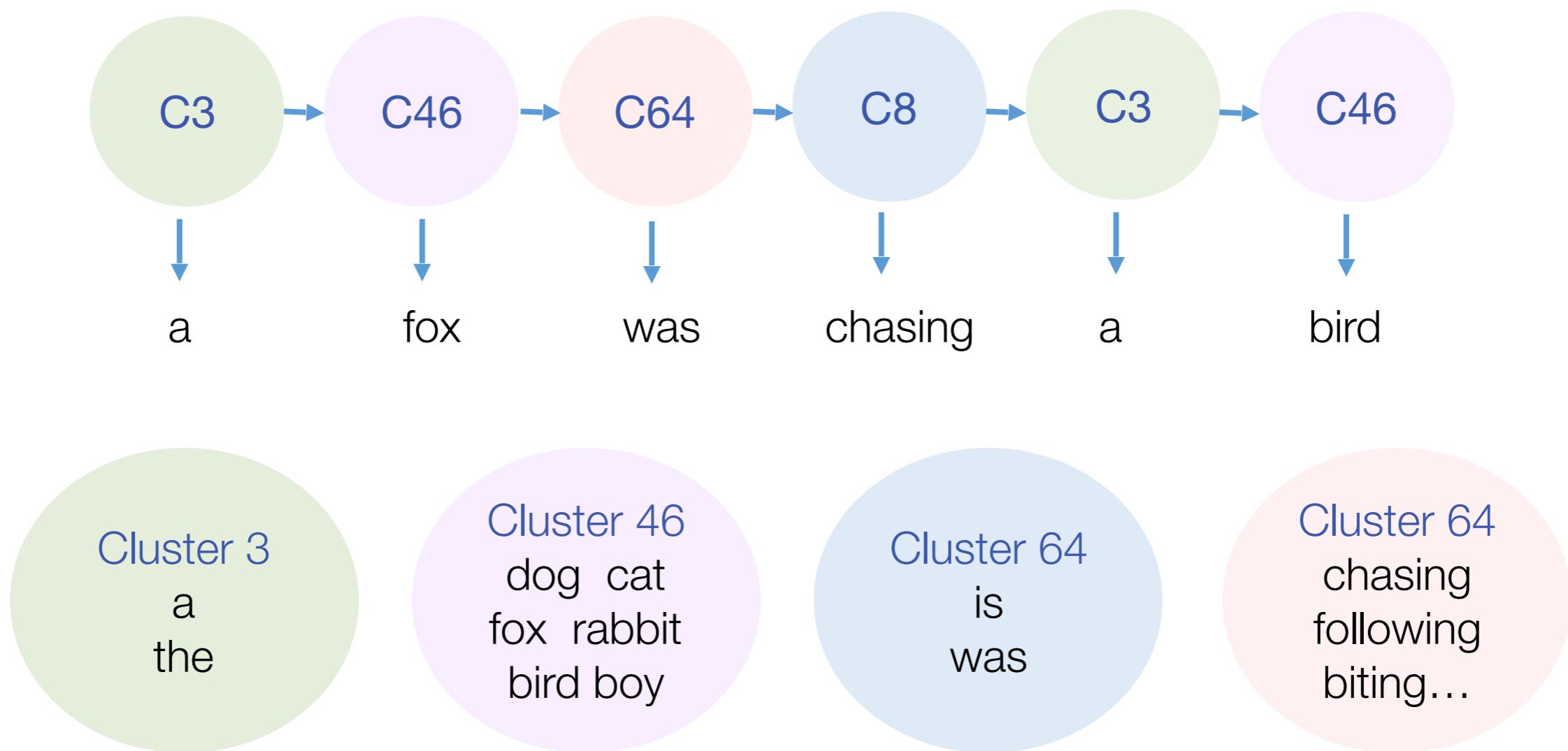
# Motivating example

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



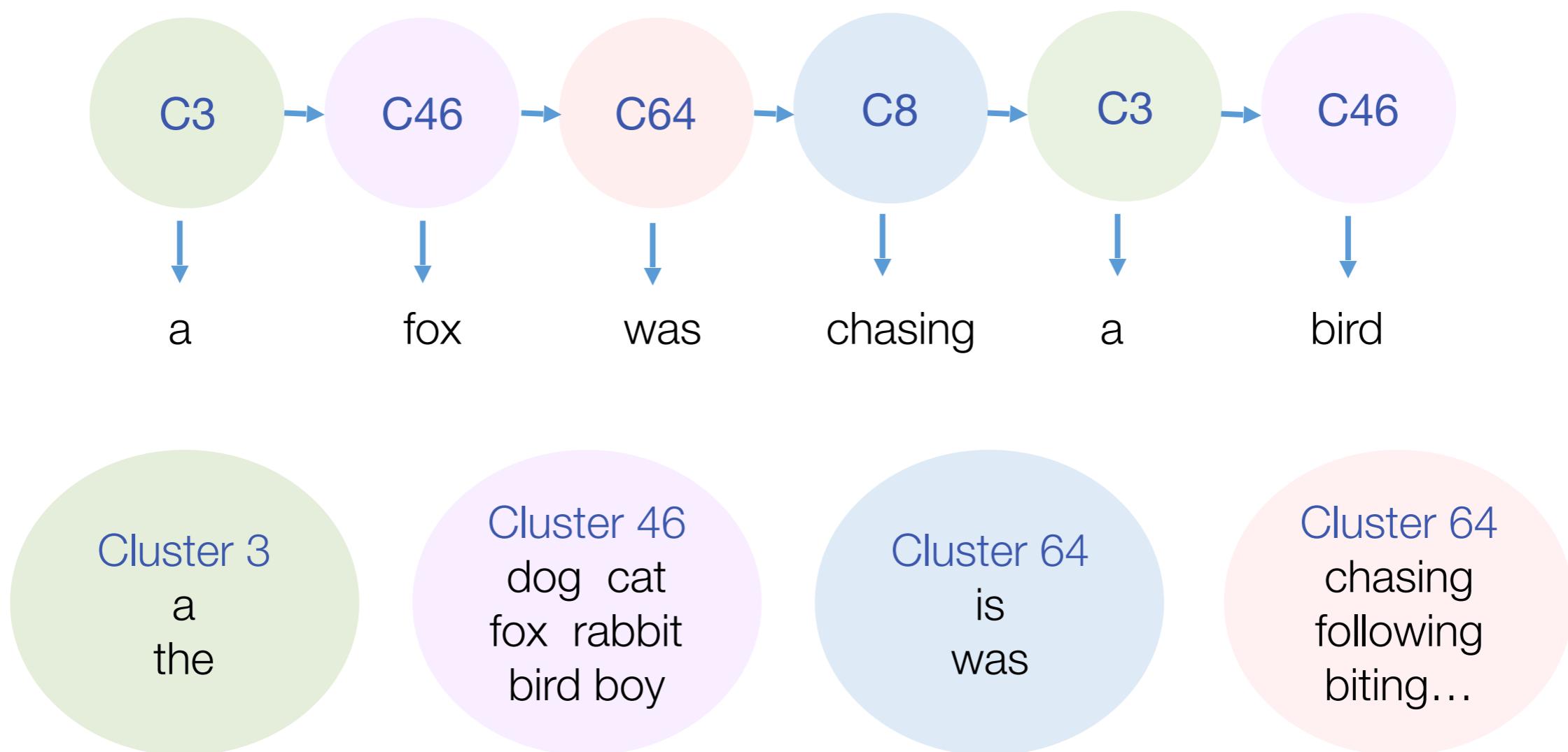
# Motivating example

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



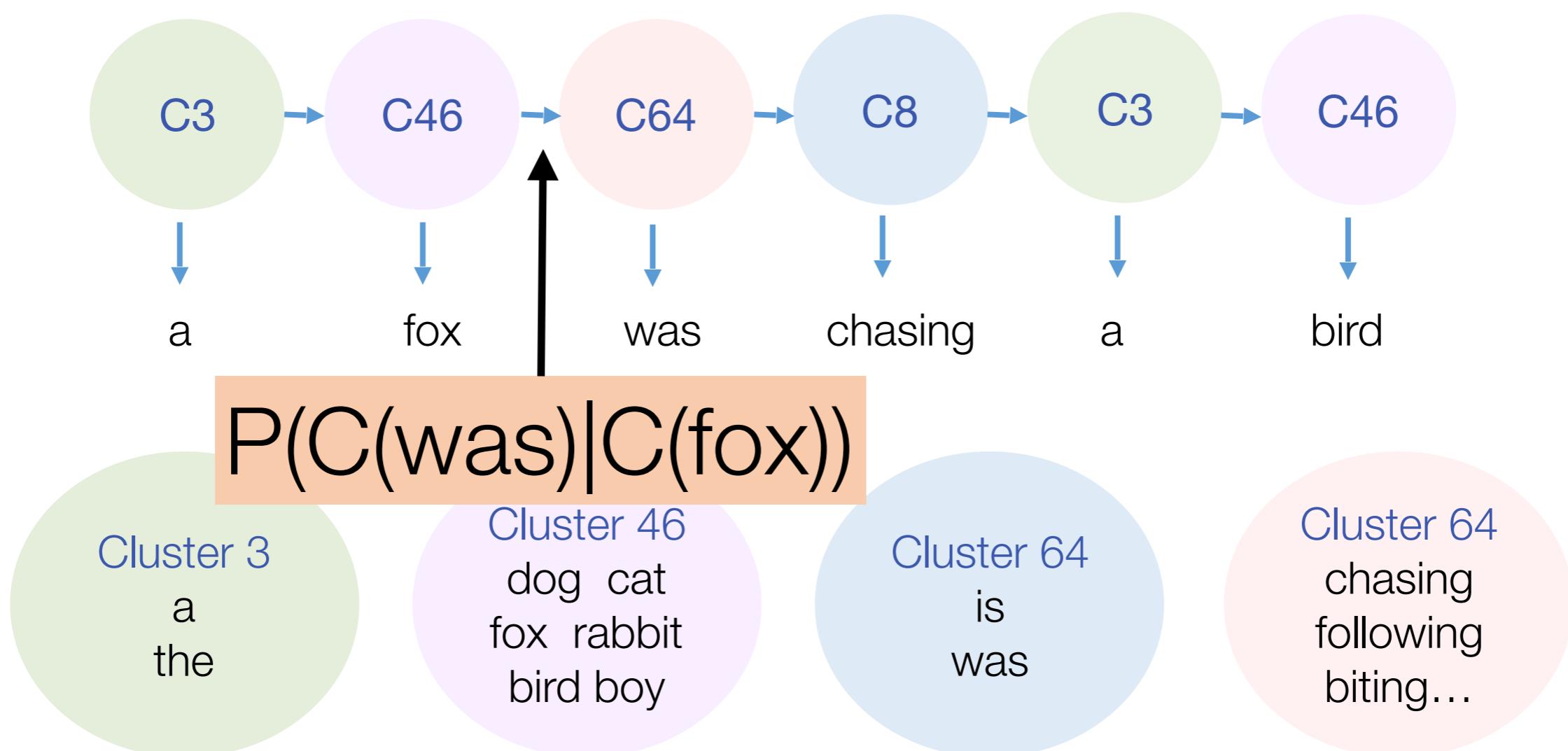
# How do we measure the probability of this sequence?

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



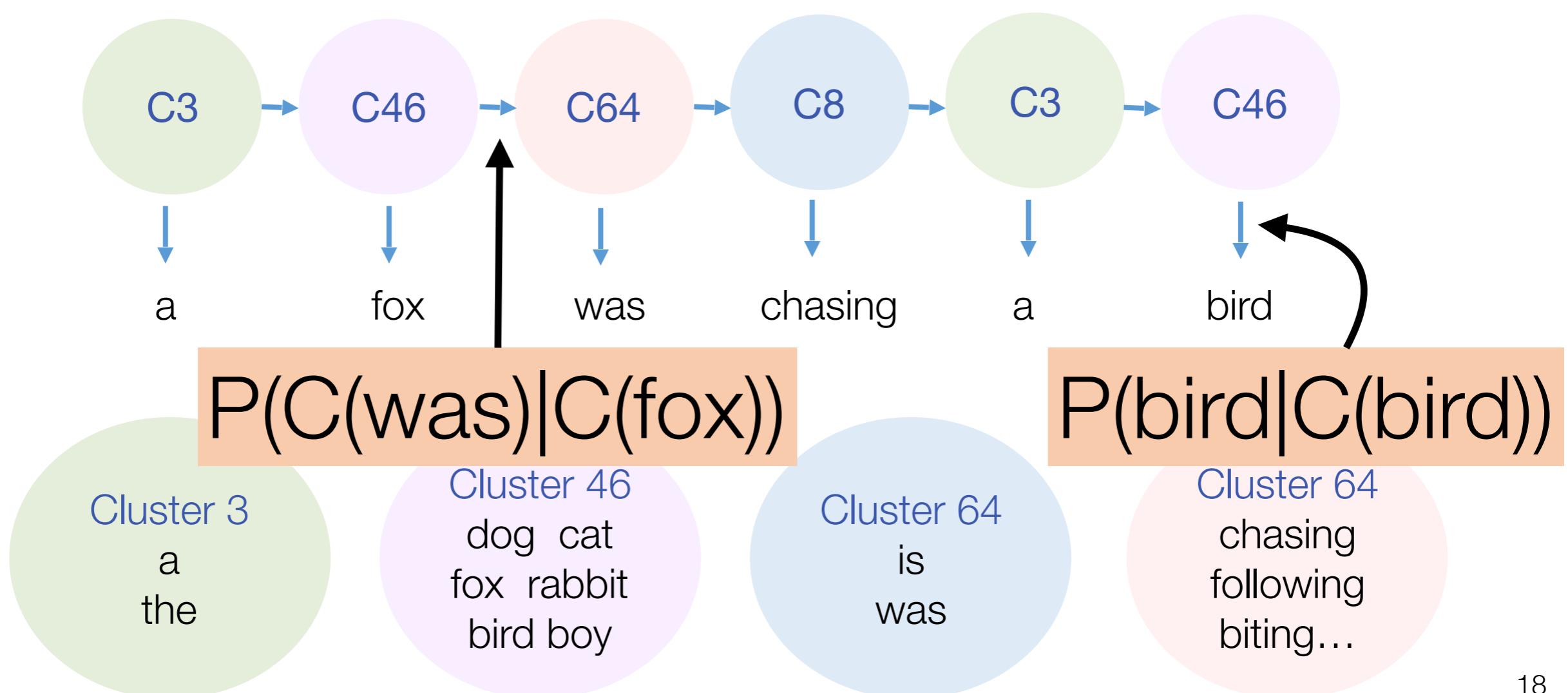
# How do we measure the probability of this sequence?

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



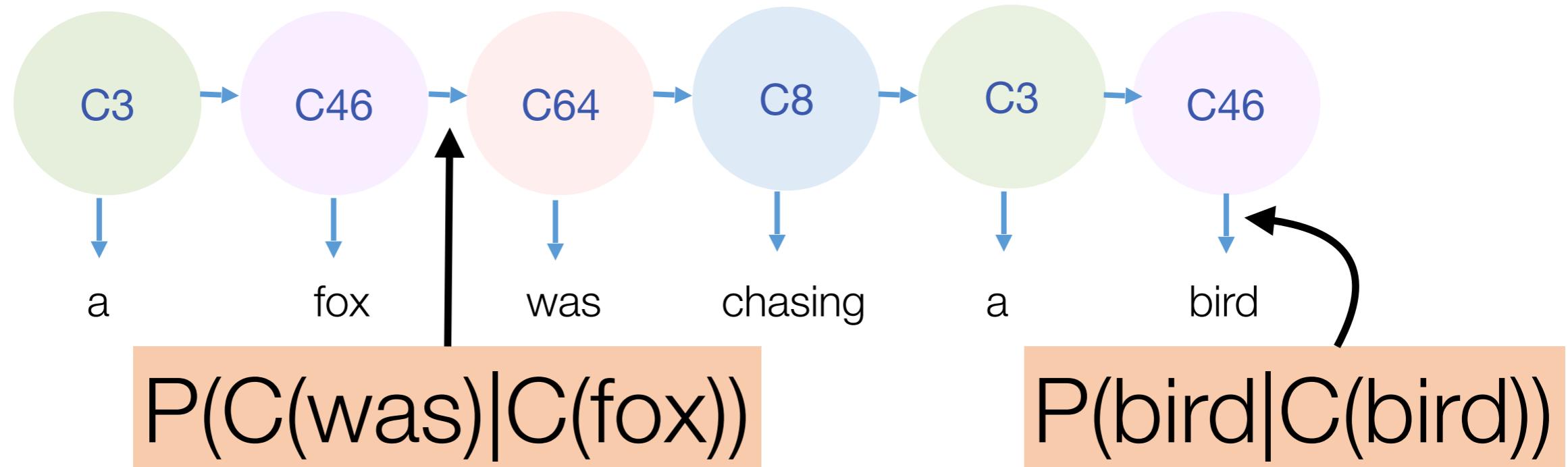
# How do we measure the probability of this sequence?

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



# How do we measure the probability of this sequence?

- “a fox was chasing a bird”
- $P(\text{“a”, “dog”, …, “cat”}) =$ 
  - Before:  $P(\text{“a”}|\text{START}) \dots P(\text{“cat”}|\text{“a”})$
  - Now=  $P(C_3|\text{START}) P(\text{“a”}|C_3) P(C_{46}|C_3) \dots P(\text{“bird”}|C_{46})$



# Class-based language model

- Suppose each word was in some class  $c_i$ :

$$P(w_i | w_{i-1}) = P(c_i | c_{i-1})P(w_i | c_i)$$

$$P(\text{corpus} | C) = \prod_{i=1}^n P(c_i | c_{i-1})P(w_i | c_i)$$

Where do the classes come from?

# Brown clustering algorithm

- Each word is initially assigned to its own cluster.

# Brown clustering algorithm

- Each word is initially assigned to its own cluster.
- We now consider merging each pair of clusters.  
The highest quality merge is chosen.

# Brown clustering algorithm

- Each word is initially assigned to its own cluster.
- We now consider merging each pair of clusters.  
The highest quality merge is chosen.
  - Quality  $\approx$  merge the two words that have the most similar probabilities of preceding and following words

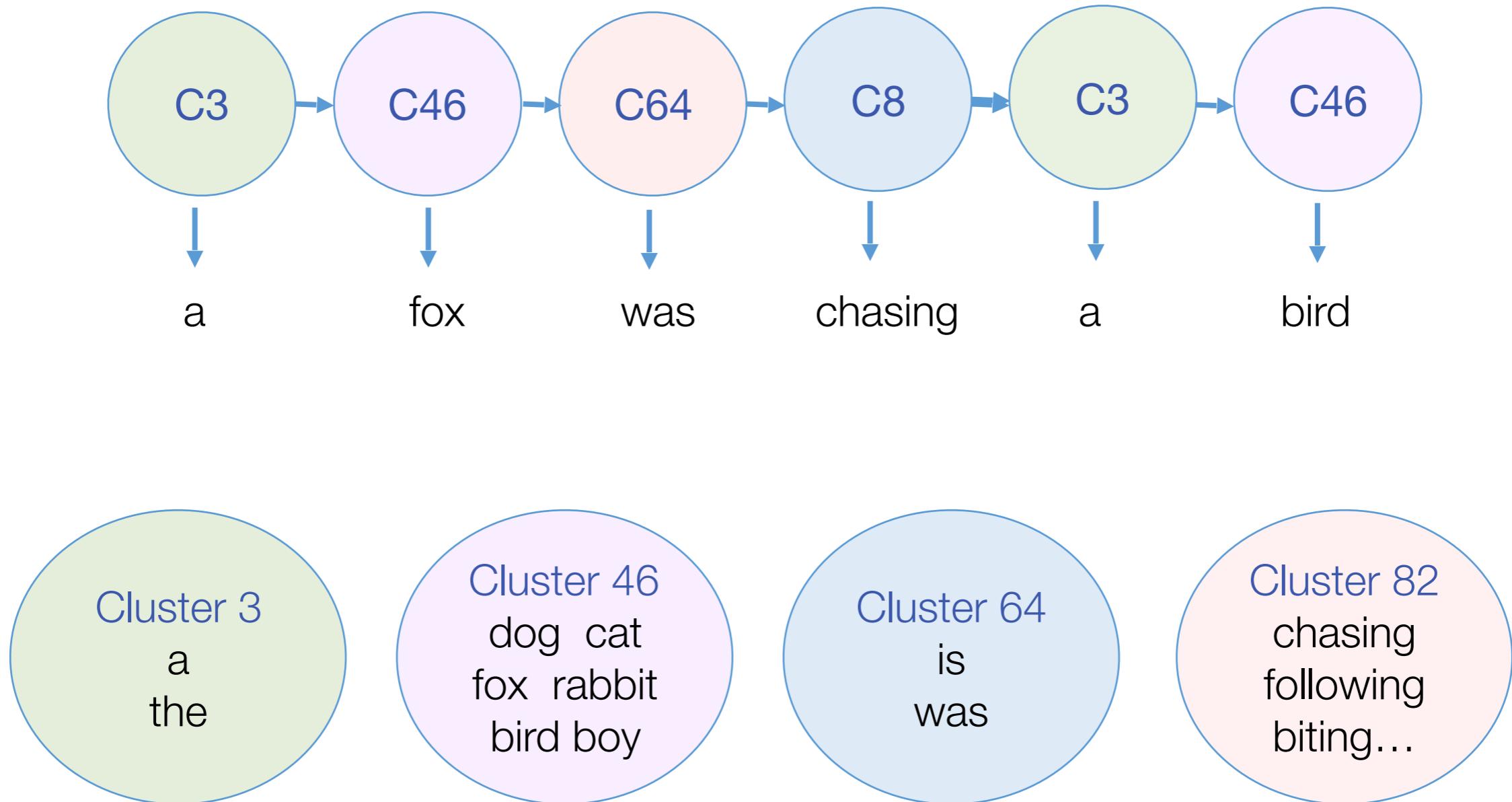
# Brown clustering algorithm

- Each word is initially assigned to its own cluster.
- We now consider merging each pair of clusters.  
The highest quality merge is chosen.
  - Quality  $\approx$  merge the two words that have the most similar probabilities of preceding and following words
  - More technically, quality = smallest decrease in the likelihood of the corpus according to a class-based language model

# Brown clustering algorithm

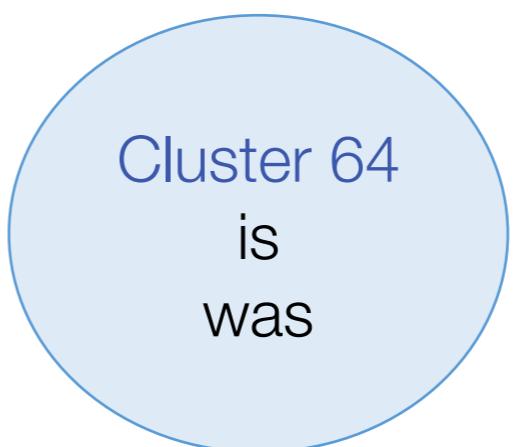
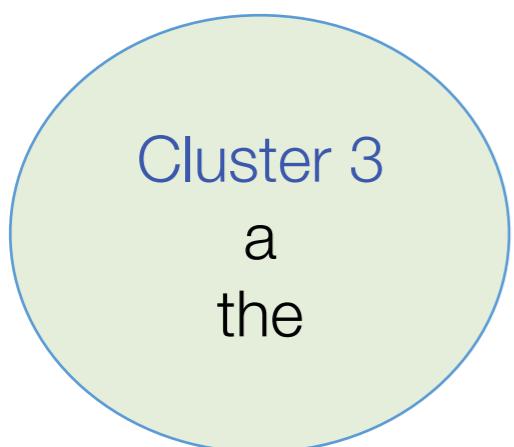
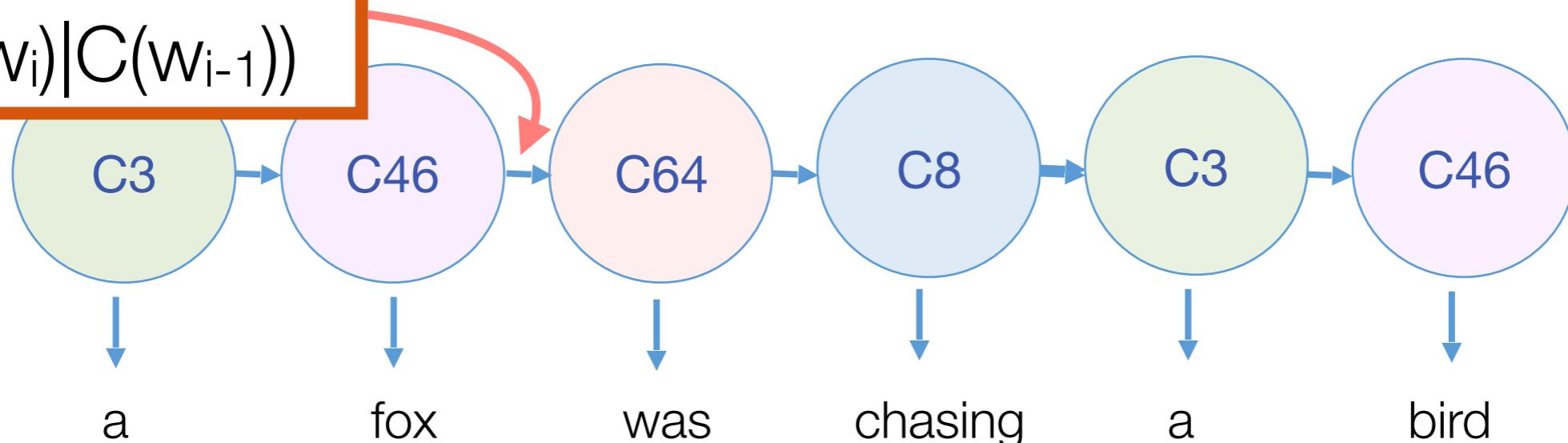
- Each word is initially assigned to its own cluster.
- We now consider merging each pair of clusters.  
The highest quality merge is chosen.
  - Quality  $\approx$  merge the two words that have the most similar probabilities of preceding and following words
  - More technically, quality = smallest decrease in the likelihood of the corpus according to a class-based language model
- Clustering proceeds until all words are in one big cluster.

# Model Parameters



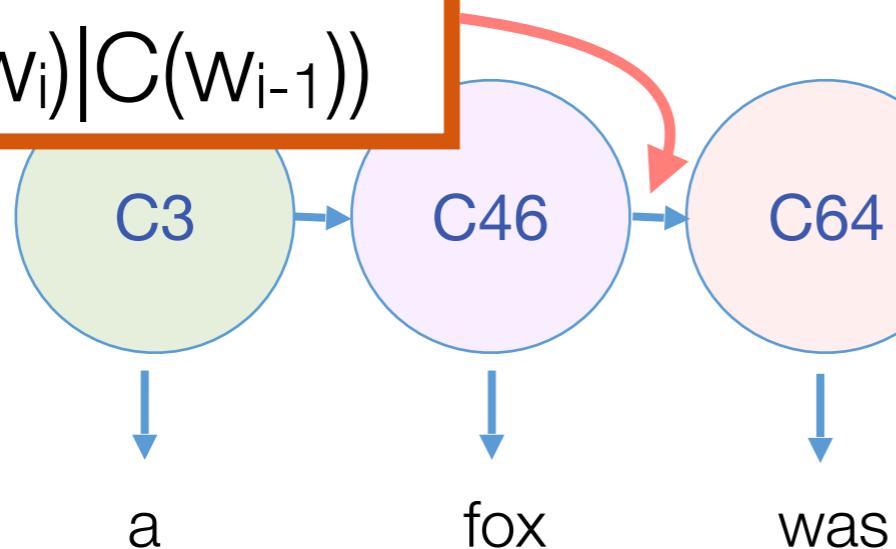
# Model Parameters

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

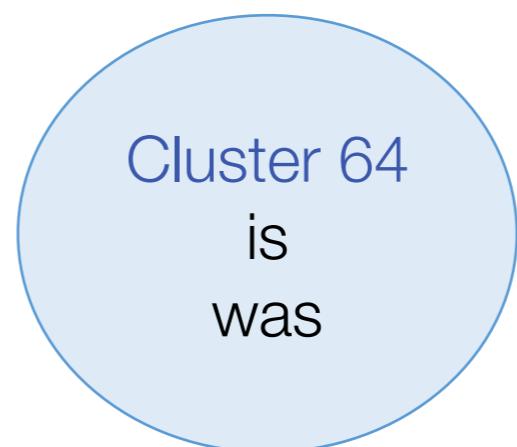
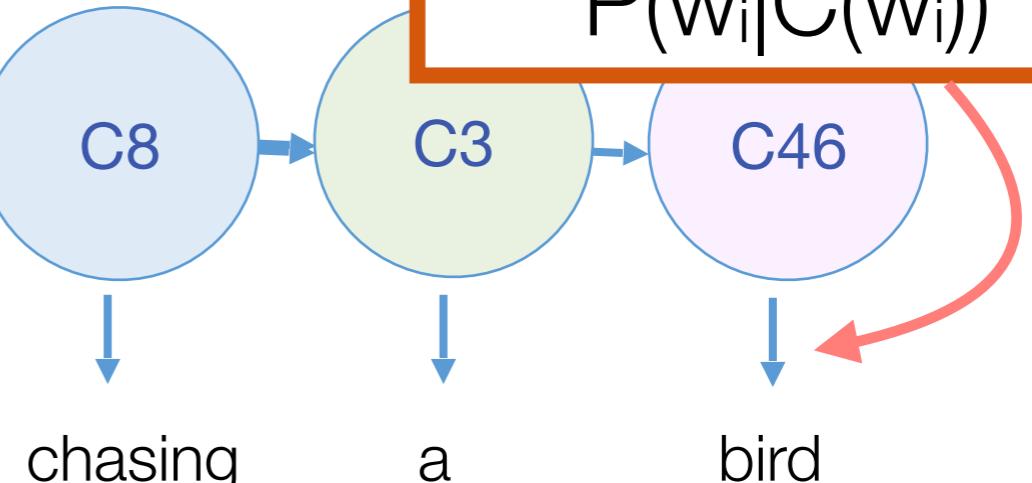


# Model Parameters

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

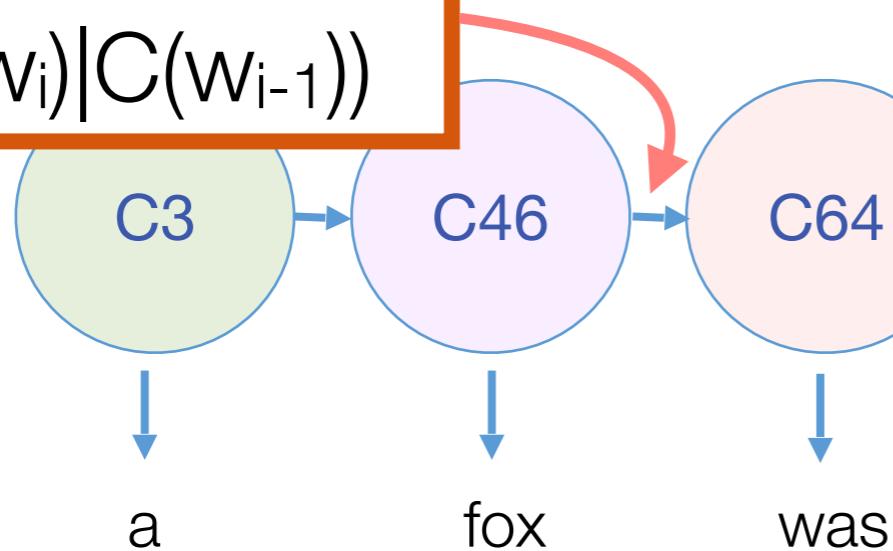


Parameter Set 2:  
 $P(w_i|C(w_i))$

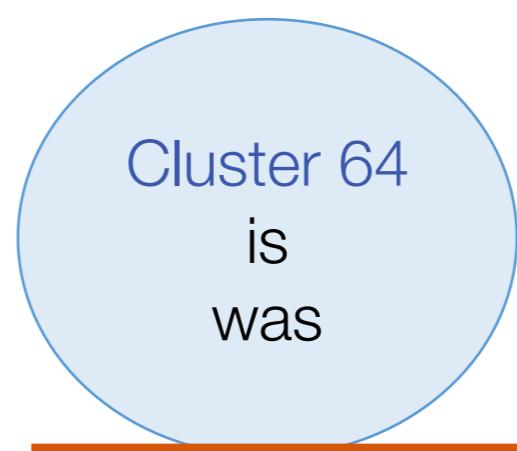
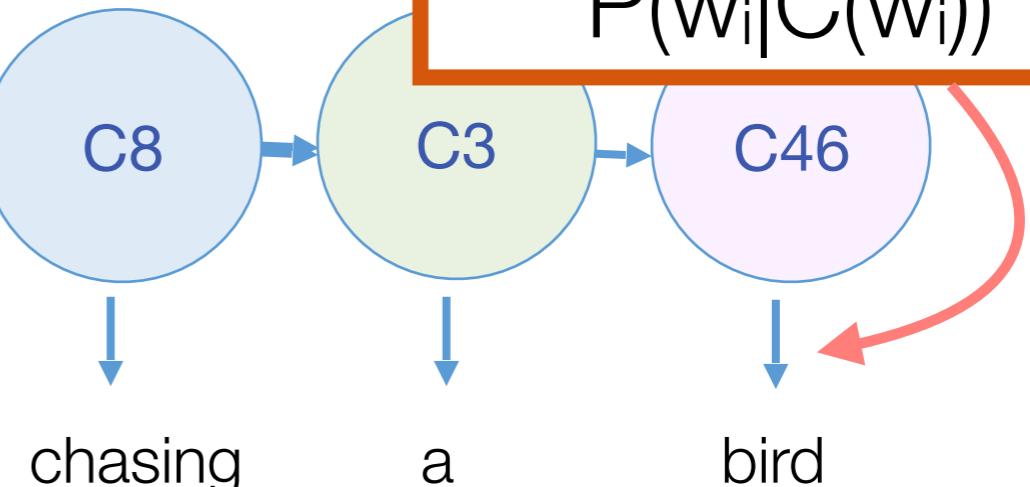


# Model Parameters

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



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



Parameter Set 3:  
 $C(w_i)$

# Learning the Clustering

# Learning the Clustering

- $\theta$  represents the set of conditional probability parameters

# Learning the Clustering

- $\theta$  represents the set of conditional probability parameters
- $C$  represents the clustering

$$\text{LL}(\theta, C) = \sum_{i=1}^n [\log P(C(w_i) | C(w_{i-1})) + \log P(w_i | C(w_i))]$$

# Learning the Clustering

- $\theta$  represents the set of conditional probability parameters
- $C$  represents the clustering

$$\text{LL}(\theta, C) = \sum_{i=1}^n [\log P(C(w_i) | C(w_{i-1})) + \log P(w_i | C(w_i))]$$

- Maximizing  $\text{LL}(\theta, C)$  can be done by alternatively updating  $\theta$  and  $C$

# Learning the Clustering

- $\theta$  represents the set of conditional probability parameters
- $C$  represents the clustering

$$\text{LL}(\theta, C) = \sum_{i=1}^n [\log P(C(w_i) | C(w_{i-1})) + \log P(w_i | C(w_i))]$$

- Maximizing  $\text{LL}(\theta, C)$  can be done by alternatively updating  $\theta$  and  $C$
- Maximizing  $\text{LL}(\theta, C)$  is “how much do the current classes and transition probabilities explain the data I see”

# Maximizing $\theta$ for $LL(\theta, C)$

- Count things!
  - $P(C_i | C_{i-1}) = (\#(C_{i-1}, C_i)) / (\#C_i)$
  - $P(w | C) = (\#(w, C)) / (\#C)$

# Algorithm 1 for Brown Clustering

# Algorithm 1 for Brown Clustering

- Start with  $|V|$  clusters, with each word in its own cluster

# Algorithm 1 for Brown Clustering

- Start with  $|V|$  clusters, with each word in its own cluster
- Goal: get  $k$  clusters of words

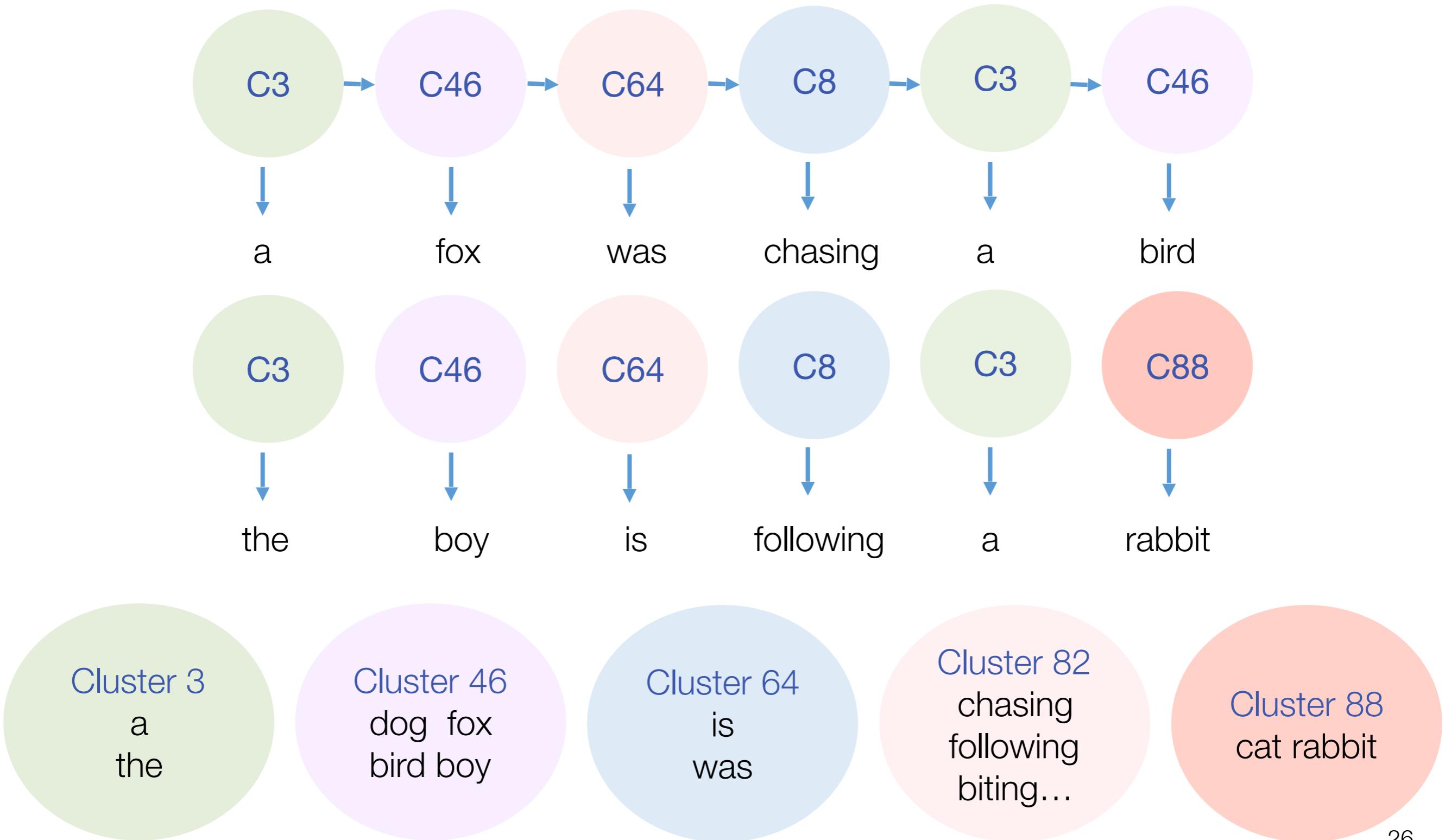
# Algorithm 1 for Brown Clustering

- Start with  $|V|$  clusters, with each word in its own cluster
- Goal: get  $k$  clusters of words
- Run  $|V|-k$  merge steps

# Algorithm 1 for Brown Clustering

- Start with  $|V|$  clusters, with each word in its own cluster
- Goal: get  $k$  clusters of words
- Run  $|V|-k$  merge steps
  - Each step, chose to merge the two clusters that maximize  $LL(\theta, C)$

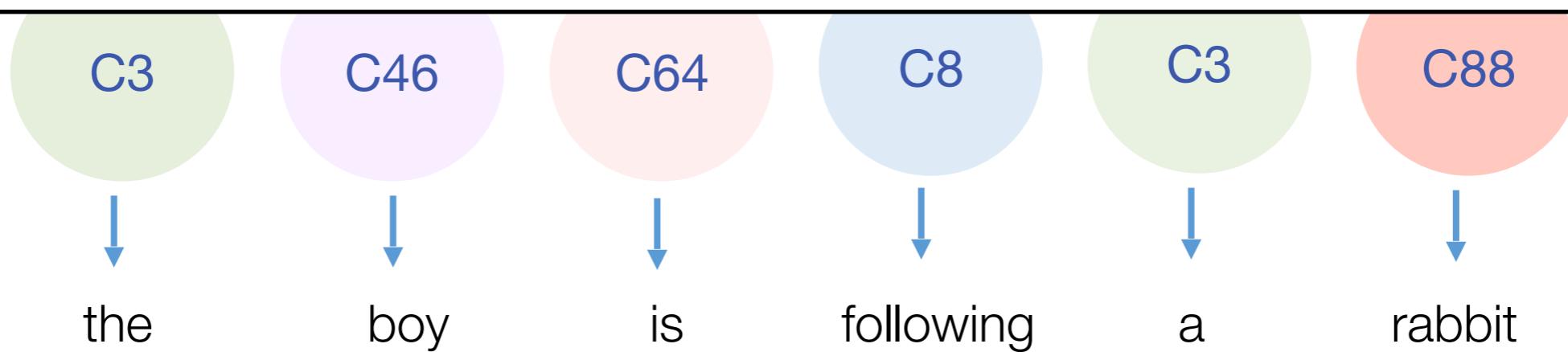
Each step, chose to merge the two clusters that maximize  $LL(\theta, C)$



# Each step, chose to merge the two clusters that maximize $LL(\theta, C)$



Does merging these two clusters give us better log likelihood?  
(are the new clusters are better able to explain the data we see?)



Cluster 3  
a  
the

Cluster 46  
dog fox  
bird boy

Cluster 64  
is  
was

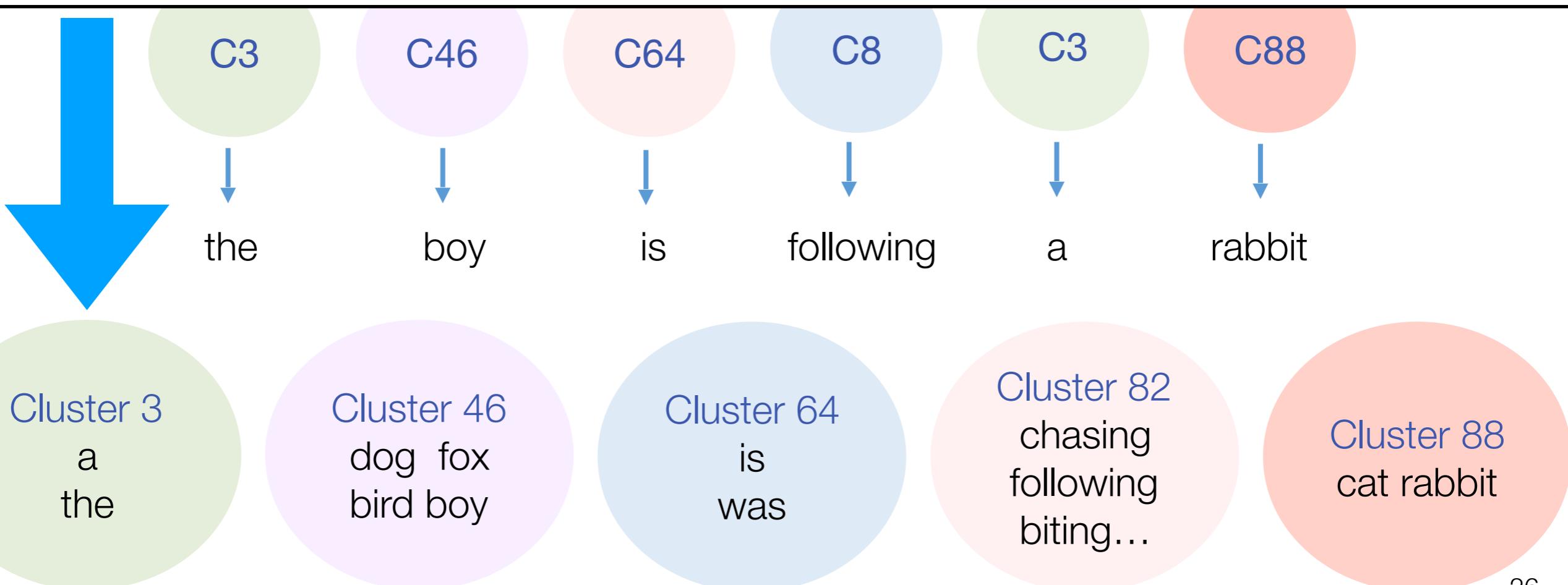
Cluster 82  
chasing  
following  
biting...

Cluster 88  
cat rabbit

# Each step, chose to merge the two clusters that maximize $LL(\theta, C)$



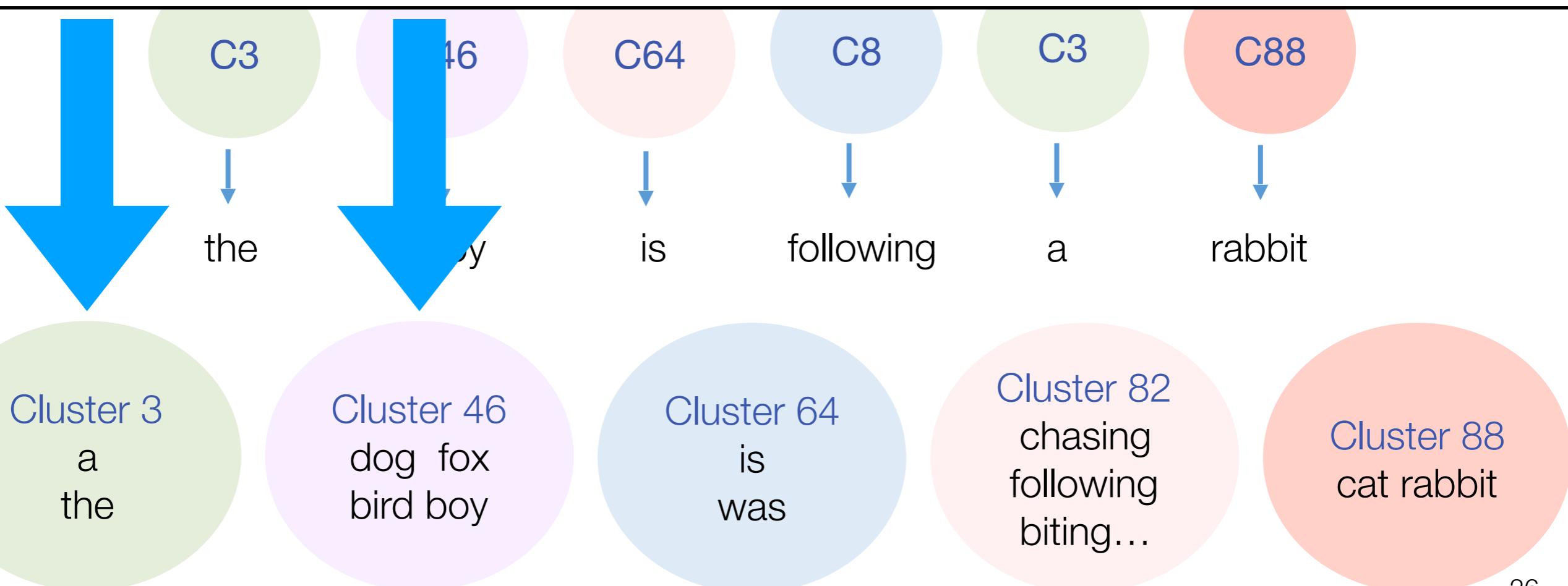
Does merging these two clusters give us better log likelihood?  
(are the new clusters are better able to explain the data we see?)



Each step, chose to merge the two clusters that maximize  $LL(\theta, C)$



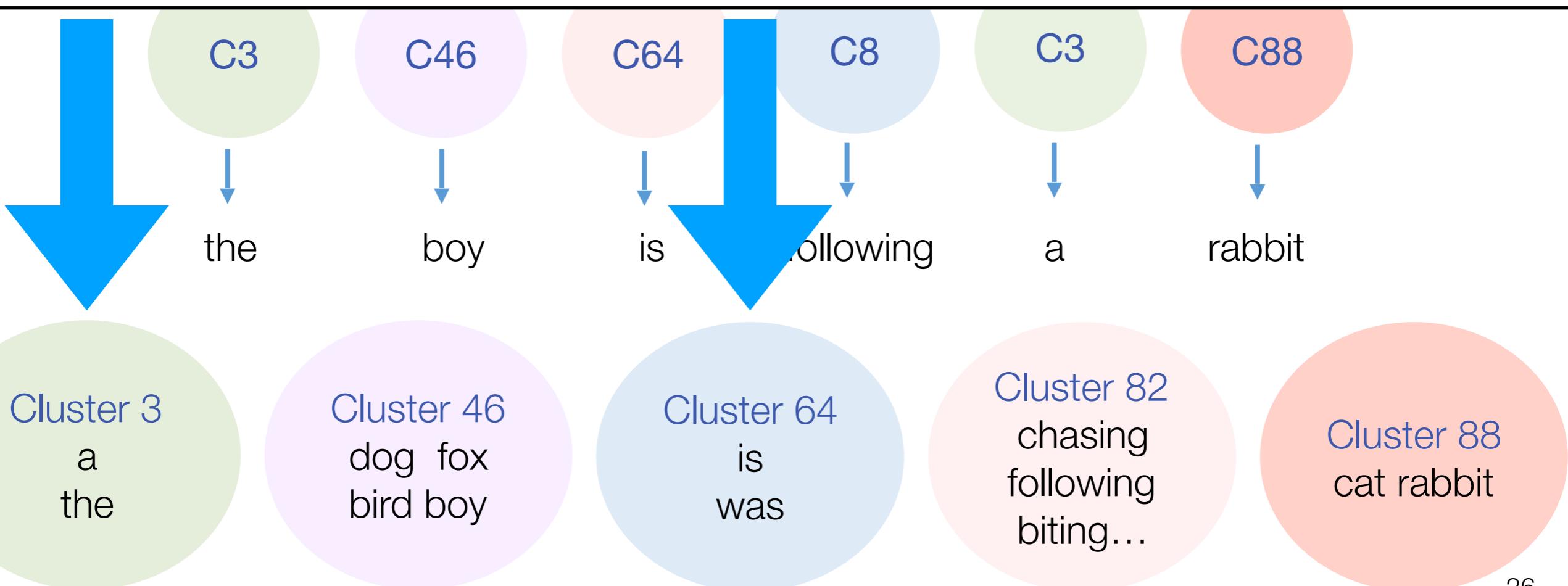
Does merging these two clusters give us better log likelihood?  
(are the new clusters are better able to explain the data we see?)



# Each step, chose to merge the two clusters that maximize $LL(\theta, C)$



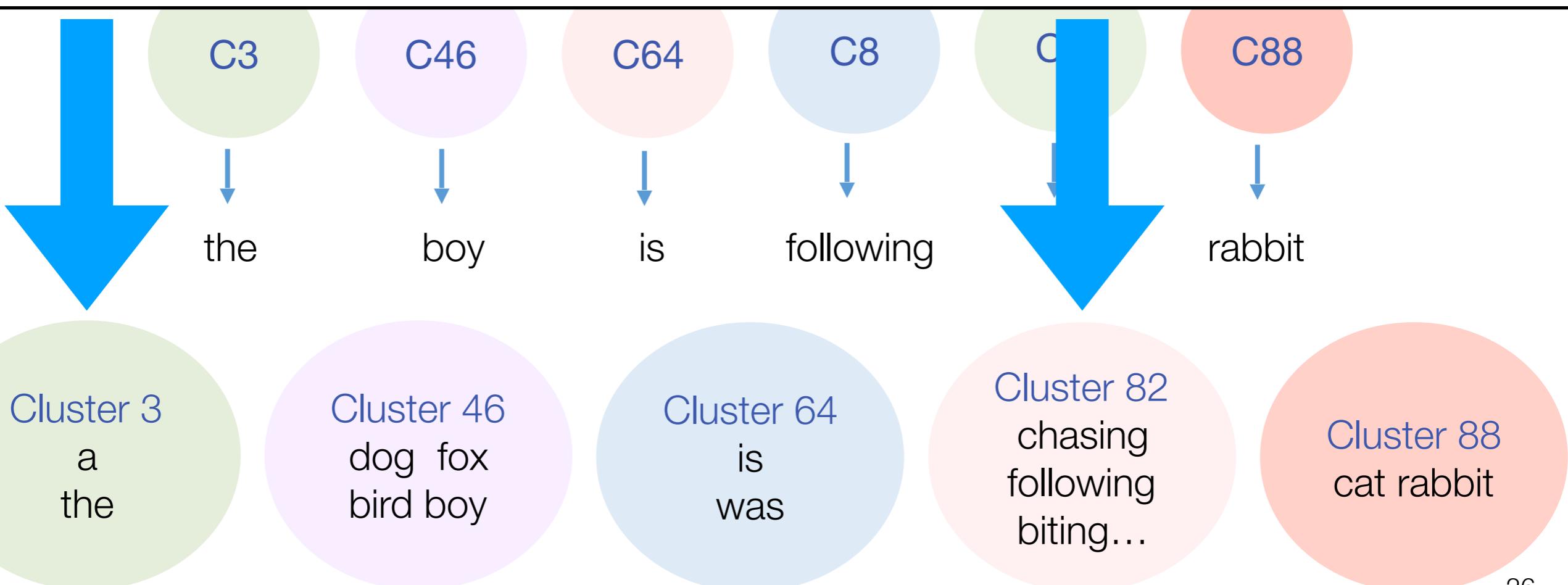
Does merging these two clusters give us better log likelihood?  
(are the new clusters are better able to explain the data we see?)



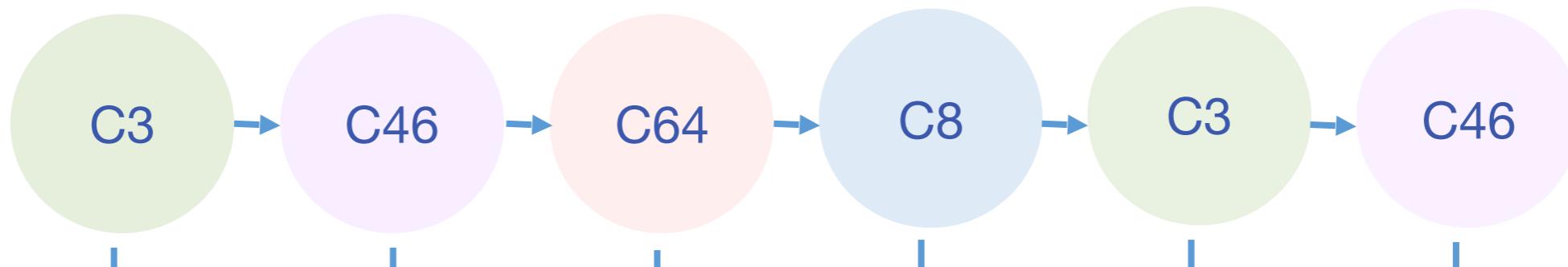
Each step, chose to merge the two clusters that maximize  $LL(\theta, C)$



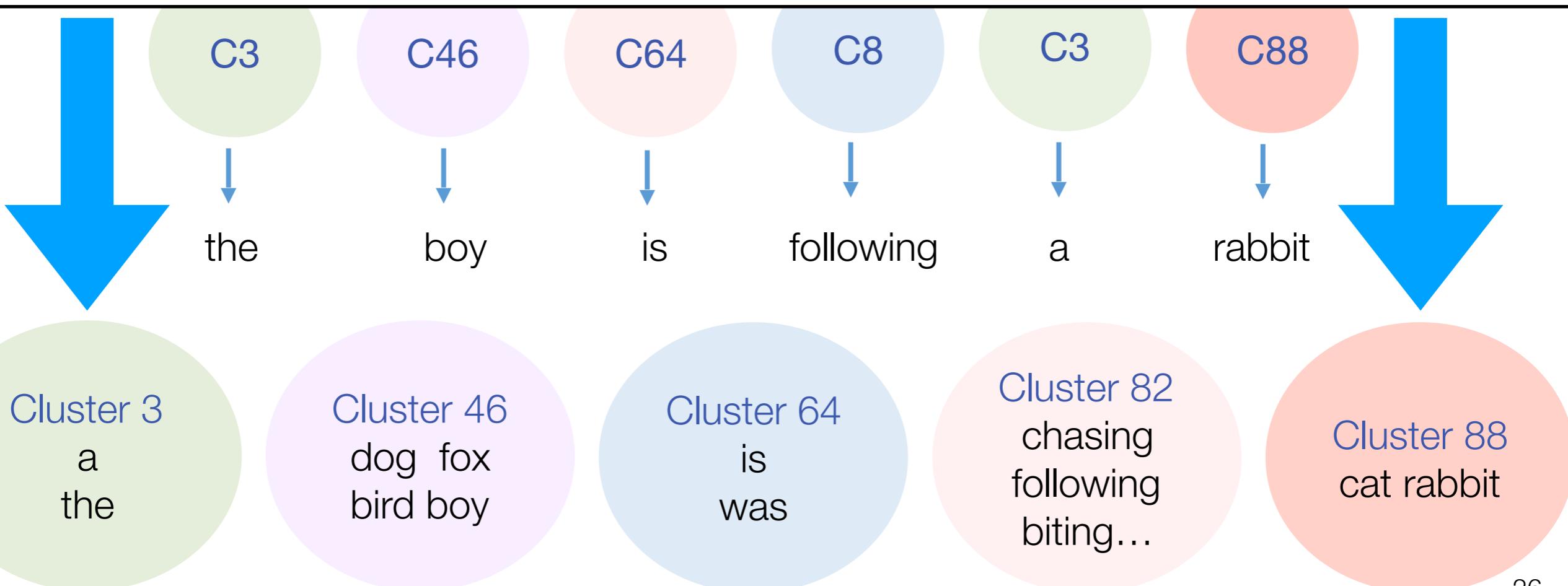
Does merging these two clusters give us better log likelihood?  
(are the new clusters are better able to explain the data we see?)



Each step, chose to merge the two clusters that maximize  $LL(\theta, C)$



Does merging these two clusters give us better log likelihood?  
(are the new clusters are better able to explain the data we see?)



# Algorithm 1 for Brown Clustering

- Start with  $|V|$  clusters, with each word in its own cluster
- Goal: get  $k$  clusters of words
- Run  $|V|-k$  merge steps
  - Each step, chose to merge the two clusters that maximize  $LL(\theta, C)$

# Algorithm 1 for Brown Clustering

- Start with  $|V|$  clusters, with each word in its own cluster
- Goal: get  $k$  clusters of words
- Run  $|V|-k$  merge steps
  - Each step, chose to merge the two clusters that maximize  $LL(\theta, C)$
- Cost?

# Algorithm 1 for Brown Clustering

- Start with  $|V|$  clusters, with each word in its own cluster
- Goal: get  $k$  clusters of words
- Run  $|V|-k$  merge steps
  - Each step, chose to merge the two clusters that maximize  $LL(\theta, C)$
- Cost?
  - # Iters      # pairs      LL computation

# Algorithm 1 for Brown Clustering

- Start with  $|V|$  clusters, with each word in its own cluster
- Goal: get  $k$  clusters of words
- Run  $|V|-k$  merge steps
  - Each step, chose to merge the two clusters that maximize  $LL(\theta, C)$
- Cost?
  - # Iters              # pairs              LL computation  
 $O(|V|-k)$                $O(|V|^2)$                $O(|V|^2)$               =  $O(|V|^5)$

# Algorithm 1 for Brown Clustering

- Start with  $|V|$  clusters, with each word in its own cluster
- Goal: get  $k$  clusters of words
- Run  $|V|-k$  merge steps
  - Each step, chose to merge the two clusters that maximize  $LL(\theta, C)$
- Cost?
  - # Iters                  # pairs                  LL computation  
 $O(|V|-k)$                    $O(|V|^2)$                    $O(|V|^2)$                   =  $O(|V|^5)$
  - (Can be improved to  $O(|V|^3)$ )

# Algorithm 2 for Brown Clustering

# Algorithm 2 for Brown Clustering

- $m$  : a hyper-parameter ; start by sorting words by frequency

# Algorithm 2 for Brown Clustering

- $m$  : a hyper-parameter ; start by sorting words by frequency
- Put each of the top  $m$  most-frequent words in its own cluster

$C_1, \dots, C_m$

# Algorithm 2 for Brown Clustering

- $m$  : a hyper-parameter ; start by sorting words by frequency
- Put each of the top  $m$  most-frequent words in its own cluster  
 $c_1, \dots, c_m$
- For  $i = (m+1) \dots |V|$ 
  - Create a new cluster  $c_{m+1}$  (so we have  $m+1$  clusters)
  - Each step, chose to merge the two clusters that maximize  $LL(\theta, C)$ , merge them so we now have  $m$  clusters

# Algorithm 2 for Brown Clustering

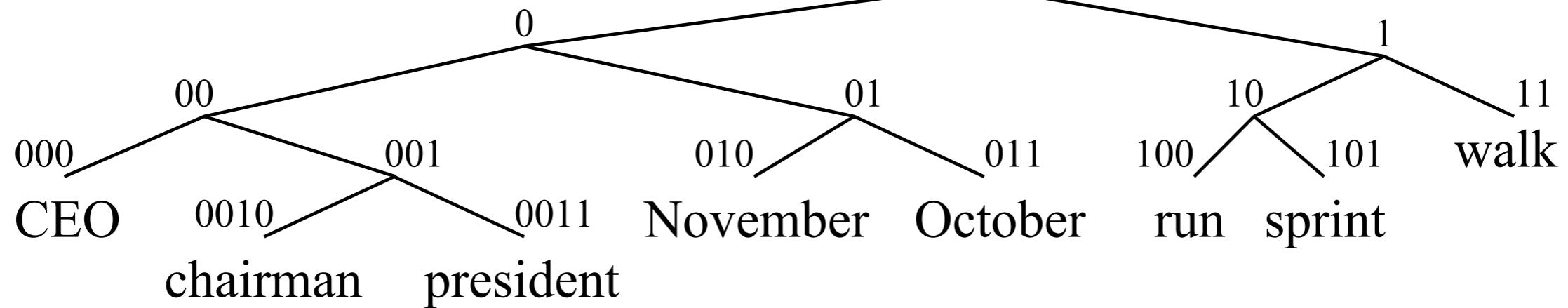
- $m$  : a hyper-parameter ; start by sorting words by frequency
- Put each of the top  $m$  most-frequent words in its own cluster  
 $c_1, \dots, c_m$
- For  $i = (m+1) \dots |V|$ 
  - Create a new cluster  $c_{m+1}$  (so we have  $m+1$  clusters)
  - Each step, chose to merge the two clusters that maximize  $LL(\theta, C)$ , merge them so we now have  $m$  clusters
- Carry out  $m-1$  final merges to get a full hierarchy

# Algorithm 2 for Brown Clustering

- $m$  : a hyper-parameter ; start by sorting words by frequency
- Put each of the top  $m$  most-frequent words in its own cluster  
 $c_1, \dots, c_m$
- For  $i = (m+1) \dots |V|$ 
  - Create a new cluster  $c_{m+1}$  (so we have  $m+1$  clusters)
  - Each step, chose to merge the two clusters that maximize  $LL(\theta, C)$ , merge them so we now have  $m$  clusters
- Carry out  $m-1$  final merges to get a full hierarchy
- Cost?  $O(|V|m^2 + n)$  where  $n$  is the # of words in corpus

# Brown Clusters as vectors

- By tracing the order in which clusters are merged, the model builds a binary tree from bottom to top.
- Each word represented by binary string = path from root to leaf
- Each intermediate node is a cluster
- Chairman is 0010, “months” = 01, and verbs = 1



# Brown cluster examples

Friday Monday Thursday Wednesday Tuesday Saturday Sunday weekends Sundays Saturdays  
June March July April January December October November September August  
pressure temperature permeability density porosity stress velocity viscosity gravity tension  
anyone someone anybody somebody  
had hadn't hath would've could've should've must've might've  
asking telling wondering instructing informing kidding reminding bothering thanking depositing  
mother wife father son husband brother daughter sister boss uncle  
great big vast sudden mere sheer gigantic lifelong scant colossal  
down backwards ashore sideways southward northward overboard aloft downwards adrift

# Example clusters

---

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 cus-  
todian  
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 depositing  
that tha theat  
head body hands eyes voice arm seat eye hair mouth

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	10110111001001010111110
Harley-Davidson	10110111001001010111110
Enfield	101101110010010101111110
genus	101101110010010101111111
Microsoft	101101110010010111000
Ventrifex	1011011100100101110010
Tractebel	10110111001001011100110
Synopsys	10110111001001011100111
WordPerfect	10110111001001011101000
....	
John	1011100100000000000
Consuelo	101110010000000001
Jeffrey	1011100100000000010
Kenneth	101110010000000001100
Phillip	1011100100000000011010
WILLIAM	1011100100000000011011
Timothy	101110010000000001110
Terrence	1011100100000000011110
Jerald	1011100100000000011111
Harold	1011100100000000100
Frederic	1011100100000000101
Wendell	101110010000000011

# Example Hierarchy

from Miller (2004)

Works really well in social people,  
especially with lots of spelling variation

<b>Cluster ID</b>	<b>Constituent word types</b>
00111001	can cn cann caan cannn ckan shalll ccan caaan cannnn caaaaan
00101111001	ii id ion iv ll iii ud wd uma ul idnt provoking hed 1+1 ididnt hast ine 2+2 idw #thingsblackpeopledo iiii #onlywhitepeople dost doan uon apt-get
0111101011110	hoping wishing considering wishin contemplating dreading regretting hopin hopeing considerin suspecting regreting wishn comtemplating hopen



# K-Means Clustering

# What is Clustering?

# What is Clustering?

- Cluster: a collection of data objects
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters

# What is Clustering?

- Cluster: a collection of data objects
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters

# What is Clustering?

- Cluster: a collection of data objects
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters
- Clustering is **unsupervised classification**:
  - no predefined classes

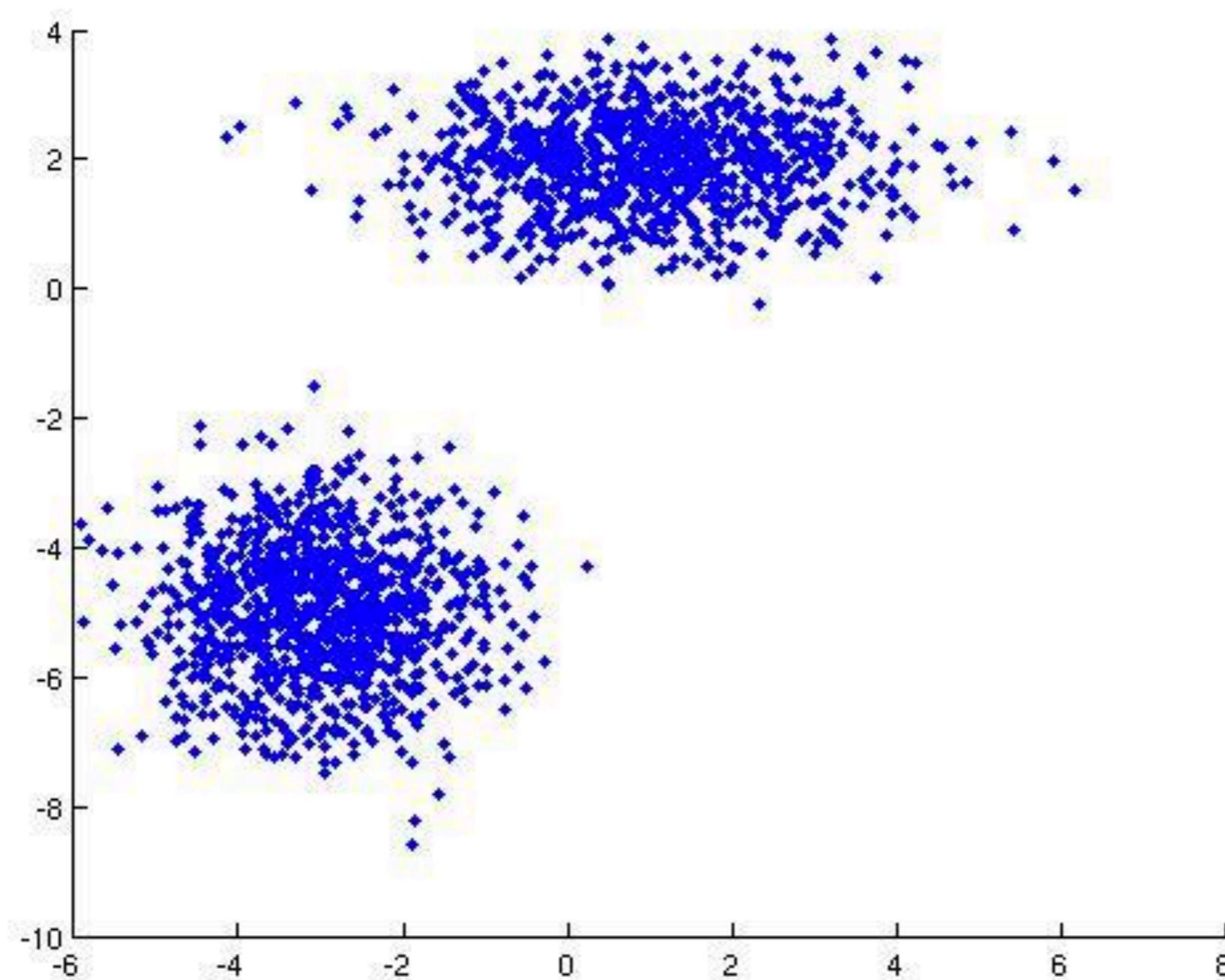
# What is Clustering?

- Cluster: a collection of data objects
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters
- Clustering is **unsupervised classification**:
  - no predefined classes
- Typical applications
  - As a **stand-alone tool** to get insight into data distribution
  - As a **preprocessing step** for other NLP algorithms

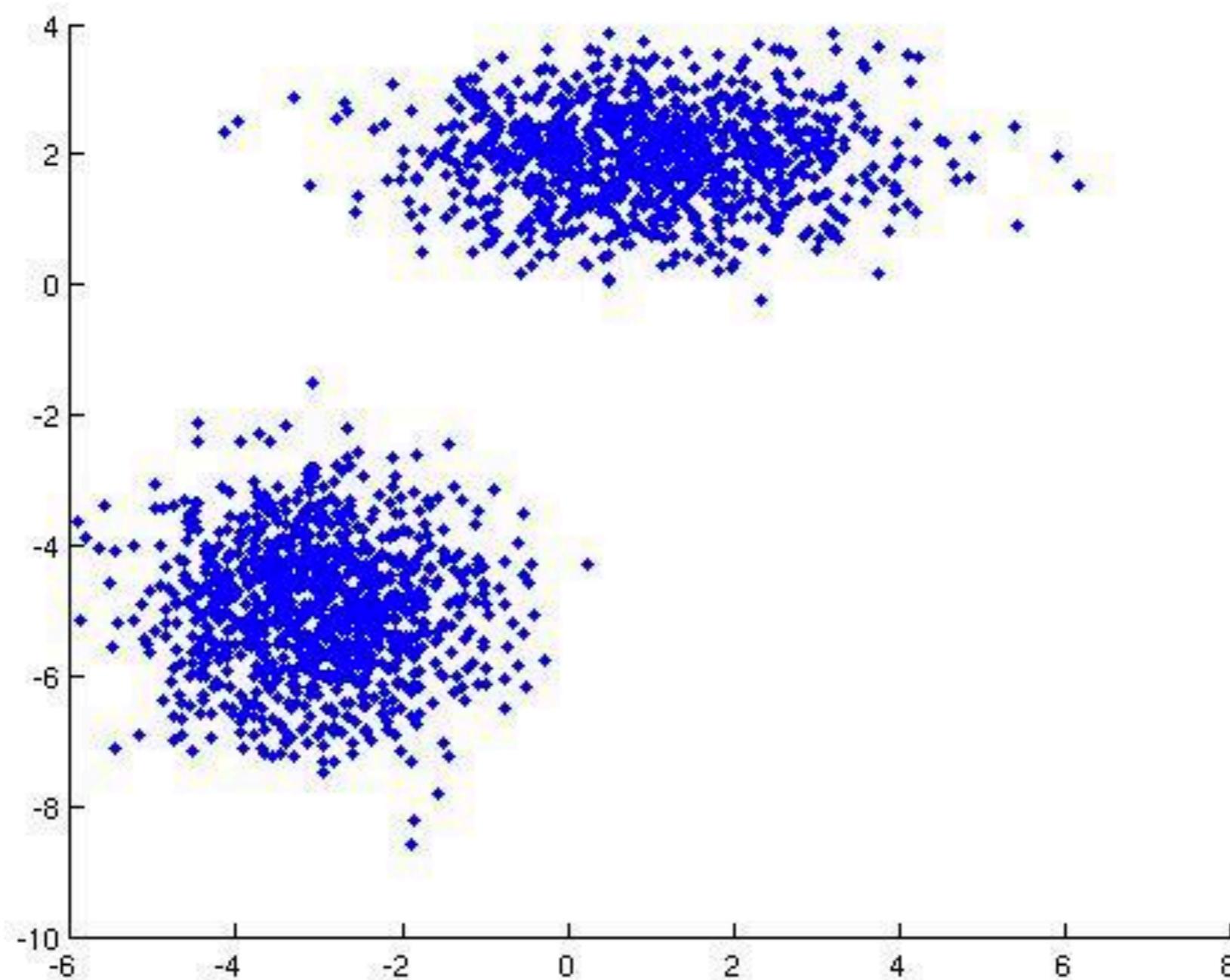
# Clustering in NLP

- Most linguistic items can be represented as vectors, e.g., your word2vec vectors
- We compare linguistic items using similarity metrics — often cosine similarity
- This lets us apply standard clustering methods to text
  - but we often visualize algorithms in 2D as points

# How many clusters are there here?

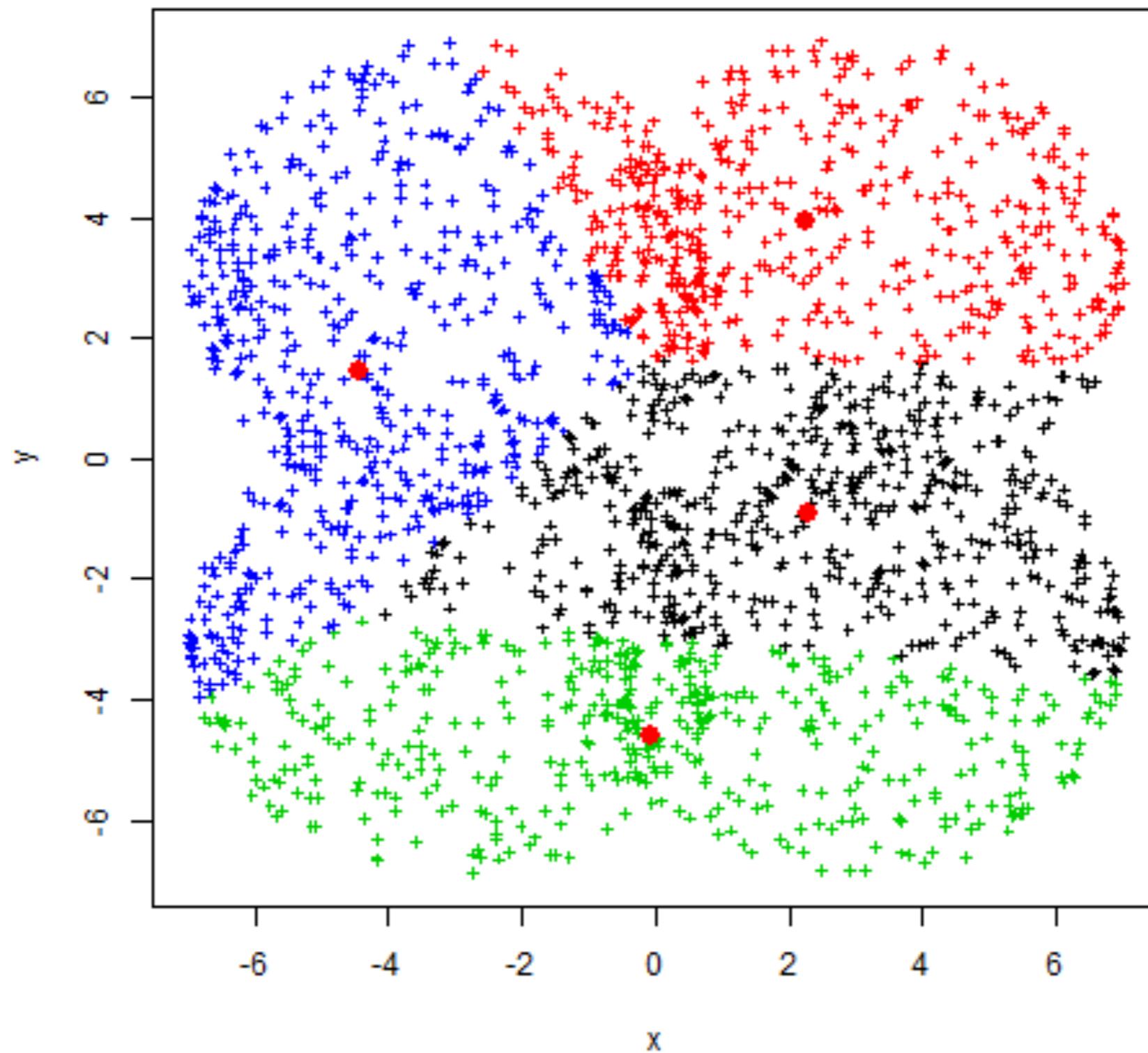


If I told you there were two clusters,  
how would we find the center?



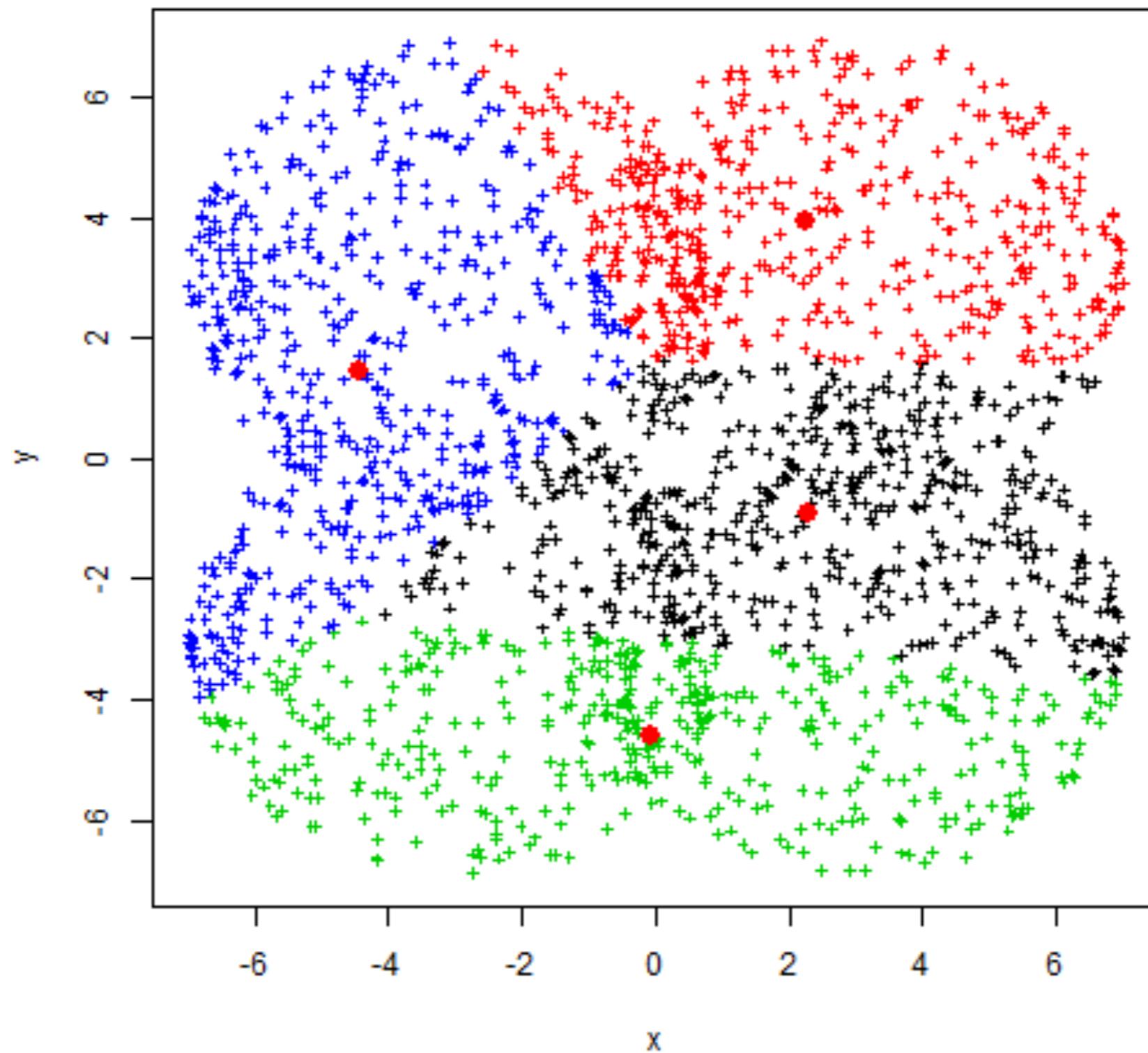
# K-Means

K Means Clustering



# K-Means

K Means Clustering



# K-Means

- Groups data into K clusters and attempts to group data points to minimize the distance to their central mean.
- Algorithm works by iterating between two stages until the data points converge.

# Problem Formulation

- Given a data set of  $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$  which consists of  $N$  random instances of a random  $D$ -dimensional variable  $\mathbf{x}$ .
- Introduce a set of  $K$  prototype vectors,  $\mu_k$  where  $k=1, \dots, K$  and  $\mu_k$  corresponds to the mean of the  $k^{\text{th}}$  cluster (also known as a **centroid**).

# Problem Formulation

- Goal is to find a grouping of data points and prototype vectors that minimizes the *distance* of each data point.
  - In spatial cases, we define distance as the sum of squares
  - In text, we often define distance as cosine distance

# Problem Formulation (cont.)

- This can be formalized by introducing an indicator variable for each data point:
  - $r_{nk}$  is  $\{0,1\}$ , and  $k=1,\dots,K$
  - The indicator is 1 if the data point is in the cluster
- Our objective function becomes:

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|x_n - \mu_k\|^2$$

# How K-Means works

- Algorithm initializes the  $K$  prototype vectors to  $K$  distinct random data points.
- Cycles between two stages until convergence is reached.
- Convergence: since there are only a finite set of possible assignments.

# How K-Means works

$$r_{nk} = \begin{cases} 1 & \text{if } k = \operatorname{argmin}_j \|x_n - \mu_k\|^2 \\ 0 & \text{otherwise} \end{cases}$$

$$\mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$$

# How K-Means works

$$r_{nk} = \begin{cases} 1 & \text{if } k = \operatorname{argmin}_j \|x_n - \mu_k\|^2 \\ 0 & \text{otherwise} \end{cases}$$

Is this cluster's centroid the closest?

$$\mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$$

# How K-Means works

- 1. For each data point, determine  $r_{nk}$  where:

$$r_{nk} = \begin{cases} 1 & \text{if } k = \operatorname{argmin}_j \|x_n - \mu_k\|^2 \\ 0 & \text{otherwise} \end{cases}$$

Is this cluster's centroid the closest?

$$\mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$$

# How K-Means works

- 1. For each data point, determine  $r_{nk}$  where:

$$r_{nk} = \begin{cases} 1 & \text{if } k = \operatorname{argmin}_j \|x_n - \mu_k\|^2 \\ 0 & \text{otherwise} \end{cases}$$

Is this cluster's centroid the closest?

- 2. Update  $\mu_k$ :

$$\mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$$

# How K-Means works

- 1. For each data point, determine  $r_{nk}$  where:

$$r_{nk} = \begin{cases} 1 & \text{if } k = \operatorname{argmin}_j \|x_n - \mu_k\|^2 \\ 0 & \text{otherwise} \end{cases}$$

Is this cluster's centroid the closest?

- 2. Update  $\mu_k$ :

$$\mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$$

# of items in the cluster

# How K-Means works

- 1. For each data point, determine  $r_{nk}$  where:

$$r_{nk} = \begin{cases} 1 & \text{if } k = \operatorname{argmin}_j \|x_n - \mu_k\|^2 \\ 0 & \text{otherwise} \end{cases}$$

Is this cluster's centroid the closest?

- 2. Update  $\mu_k$ : The sum of all vectors in the cluster

$$\mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$$

# of items in the cluster

# K-Means

Initialization example

- Pick  $K$  cluster centers (note the unfortunate choice)



# K-Means

Initialization example

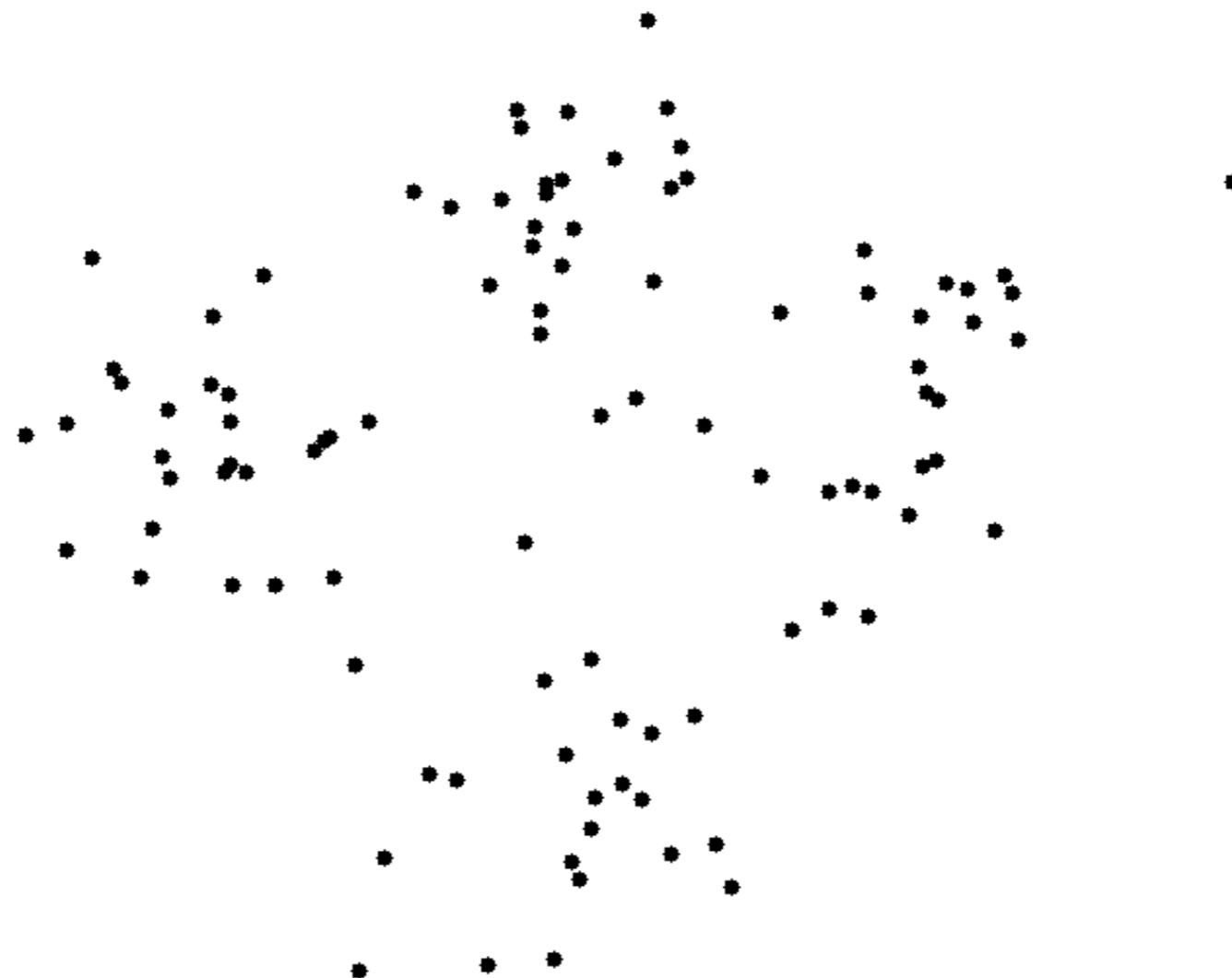
- Pick  $K$  cluster centers (note the unfortunate choice)



# K-Means

Initialization example

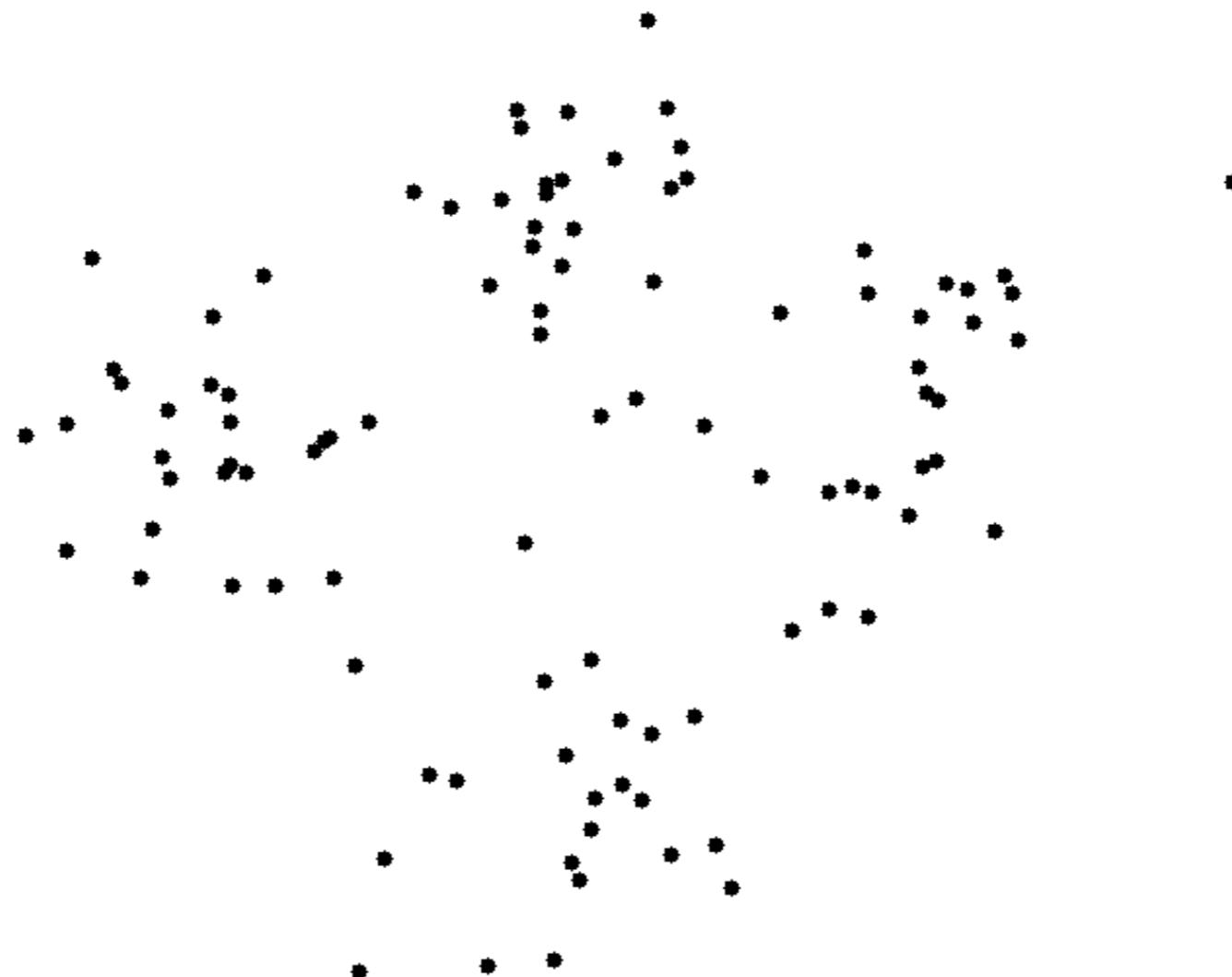
- Pick  $K$  cluster centers (random choice)



# K-Means

Initialization example

- Pick  $K$  cluster centers (random choice)



# K-Means++

K-Means with smart initial seeding

# K-Means++

K-Means with smart initial seeding

- Choose one center uniformly at random from among the data points.

# K-Means++

K-Means with smart initial seeding

- Choose one center uniformly at random from among the data points.
- For each data point  $x$ , compute  $D(x)$ , the distance between  $x$  and the nearest center that has already been chosen.

# K-Means++

K-Means with smart initial seeding

- Choose one center uniformly at random from among the data points.
- For each data point  $x$ , compute  $D(x)$ , the distance between  $x$  and the nearest center that has already been chosen.
- Choose one new data point at random as a new center, using a weighted probability distribution where a point  $x$  is chosen with probability proportional to  $D(x)^2$ .

# K-Means++

K-Means with smart initial seeding

- Choose one center uniformly at random from among the data points.
- For each data point  $x$ , compute  $D(x)$ , the distance between  $x$  and the nearest center that has already been chosen.
- Choose one new data point at random as a new center, using a weighted probability distribution where a point  $x$  is chosen with probability proportional to  $D(x)^2$ .
- Repeat Steps 2 and 3 until  $k$  centers have been chosen.

# K-Means++

K-Means with smart initial seeding

- Choose one center uniformly at random from among the data points.
- For each data point  $x$ , compute  $D(x)$ , the distance between  $x$  and the nearest center that has already been chosen.
- Choose one new data point at random as a new center, using a weighted probability distribution where a point  $x$  is chosen with probability proportional to  $D(x)^2$ .
- Repeat Steps 2 and 3 until  $k$  centers have been chosen.
- Now that the initial centers have been chosen, proceed using standard  $k$ -means.

# K-Means++

- This seeding method yields considerable improvement in the final error of  $k$ -means
- Takes more time to initialize
- Once initialized, K-Means converges quickly
- Usually faster than K-Means
- 1000 times less prone to error than K-Means

# Pros and Cons of K-Means

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|x_n - \mu_k\|^2$$

- Convergence:  $J$  may converge to a local minima and not the global minimum. May have to repeat algorithm multiple times.
- With a large data set, the distance calculations can be slow.
- $K$  is an input parameter. If  $K$  is inappropriately chosen it may yield poor results.

# Local Minima

- K-Means might not find the best possible assignments and centers.
- Consider points 0, 20, 32.
  - K-means can converge to centers at 10, 32.
  - Or to centers at 0, 26.
- Heuristic solutions
  - Start with many random starting points and pick the best solution.





# **EM**: Expectation Maximization

# Soft Clustering

- Clustering typically assumes that each instance is given a “hard” assignment to exactly one cluster.

# Soft Clustering

- Clustering typically assumes that each instance is given a “hard” assignment to exactly one cluster.
- Does not allow uncertainty in class membership or for an instance to belong to more than one cluster.

# Soft Clustering

- Clustering typically assumes that each instance is given a “hard” assignment to exactly one cluster.
- Does not allow uncertainty in class membership or for an instance to belong to more than one cluster.
- **Soft clustering** gives probabilities that an instance belongs to each of a set of clusters.

# Soft Clustering

- Clustering typically assumes that each instance is given a “hard” assignment to exactly one cluster.
- Does not allow uncertainty in class membership or for an instance to belong to more than one cluster.
- **Soft clustering** gives probabilities that an instance belongs to each of a set of clusters.
- Each instance is assigned a probability distribution across a set of discovered categories (probabilities of all categories must sum to 1).

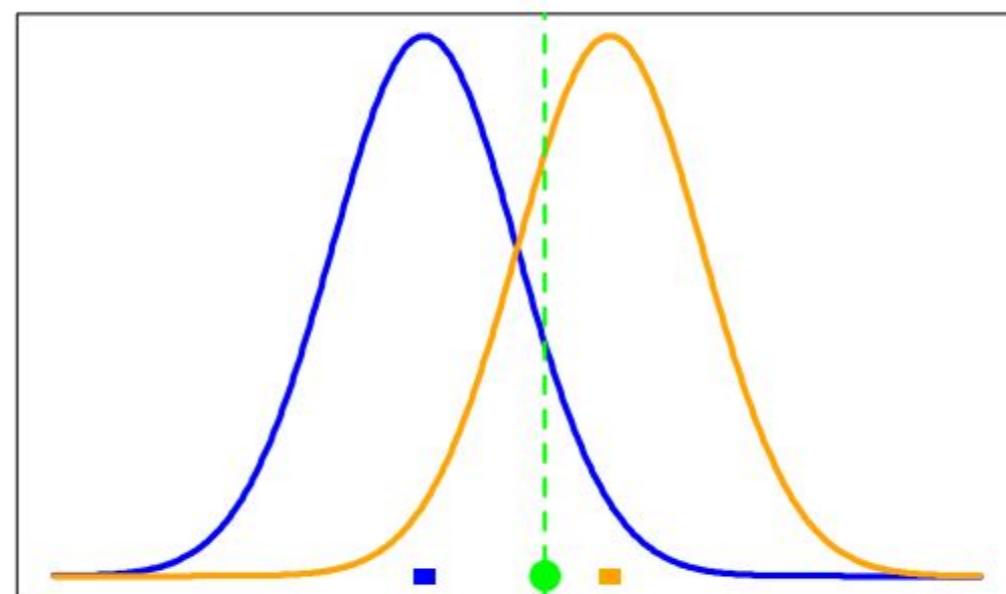
# EM

- Soft Assignments
  - A point is partially assigned to all clusters.
- Uses a probabilistic formulation
- Tends to work better than K-Means

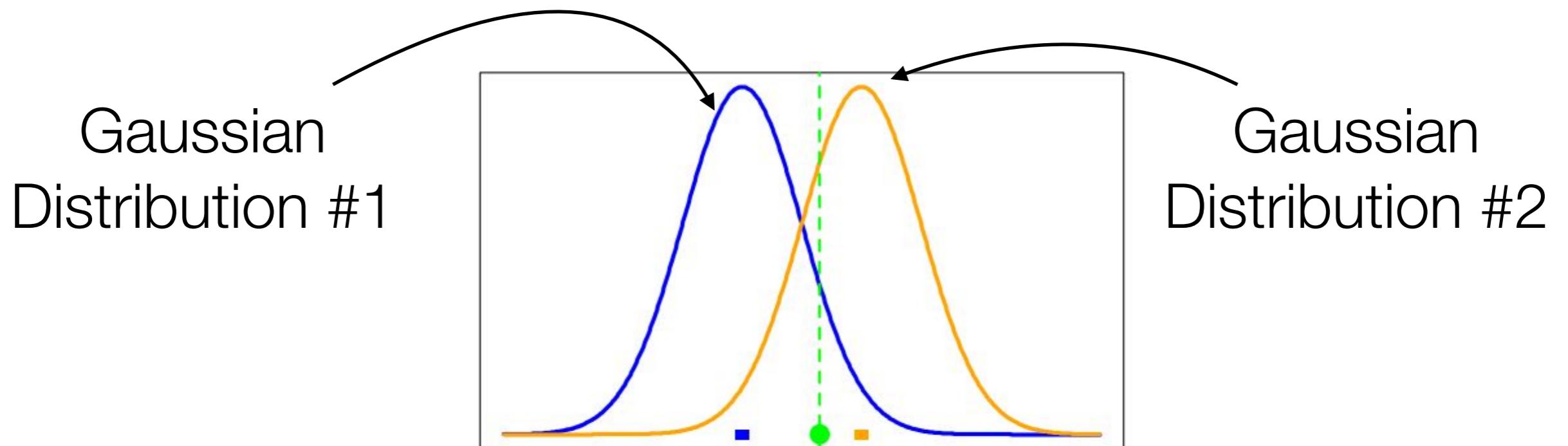
# Mixture of Gaussians

- A gaussian specifies a distribution over points according to two parameters: mean  $m$  and variance  $\sigma$
- $g(x; m, \sigma)$  is the probability of a point  $x$  based on a Gaussian Distribution with mean  $m$  and variance  $\sigma$

# Intuition



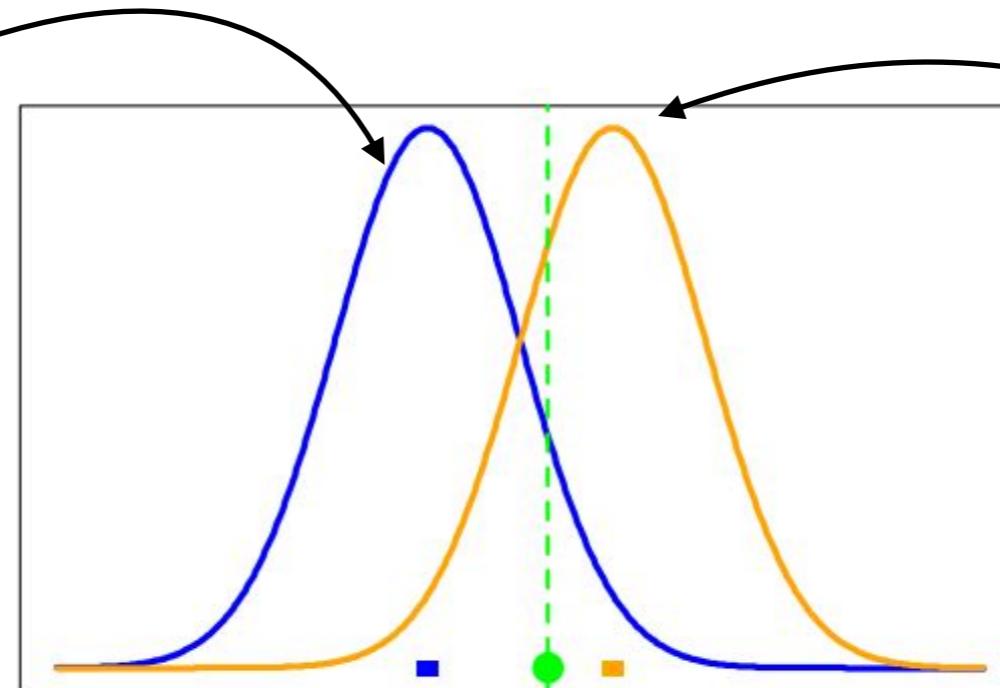
# Intuition



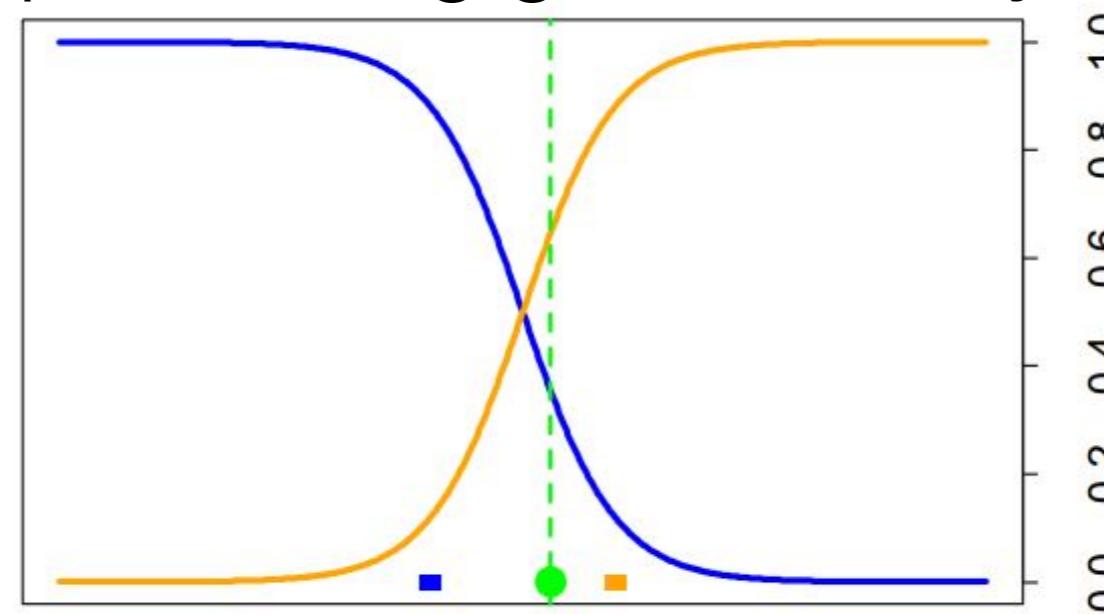
# Intuition

Gaussian  
Distribution #1

Gaussian  
Distribution #2



Probability of a point being generated by each distribution



# EM Intuition Building

# EM Intuition Building

- Suppose we have two coins A and B

# EM Intuition Building

- Suppose we have two coins A and B
- Assume bias of A is  $\theta_1$ , e.g.,  $p(\text{heads}|A)$

# EM Intuition Building

- Suppose we have two coins A and B
- Assume bias of A is  $\theta_1$ , e.g.,  $p(\text{heads}|A)$
- Assume bias of B is  $\theta_2$ , e.g.,  $p(\text{heads}|B)$

# EM Intuition Building

- Suppose we have two coins A and B
- Assume bias of A is  $\theta_1$ , e.g.,  $p(\text{heads}|A)$
- Assume bias of B is  $\theta_2$ , e.g.,  $p(\text{heads}|B)$
- We want to find  $\theta_1$  and  $\theta_2$  by performing a bunch of trials (coin flips)

# EM Intuition Building: Experiment #1

- We do 5 trials by picking a coin at random
- Then we flip the chosen coin 10 times

# EM Intuition Building: Experiment #1

- We do 5 trials by picking a coin at random
- Then we flip the chosen coin 10 times

Trial	Results
A	H T T T H H T H T H

# EM Intuition Building: Experiment #1

- We do 5 trials by picking a coin at random
- Then we flip the chosen coin 10 times

Trial	Results
A	H T T T H H T H T H
B	H H H H T H H H H H

# EM Intuition Building: Experiment #1

- We do 5 trials by picking a coin at random
- Then we flip the chosen coin 10 times

Trial	Results
A	H T T T H H T H T H
B	H H H H T H H H H H
B	H T H H H H H T H H

# EM Intuition Building: Experiment #1

- We do 5 trials by picking a coin at random
- Then we flip the chosen coin 10 times

Trial	Results
A	H T T T H H T H T H
B	H H H H T H H H H H
B	H T H H H H H T H H
A	H T H T T T H H T T

# EM Intuition Building: Experiment #1

- We do 5 trials by picking a coin at random
- Then we flip the chosen coin 10 times

Trial	Results
A	H T T T H H T H T H
B	H H H H T H H H H H
B	H T H H H H H T H H
A	H T H T T T H H T T
B	T H H H T H H H T H

# EM Intuition Building: Experiment #1

- We do 5 trials by picking a coin at random
- Then we flip the chosen coin 10 times

Trial	Results	
A	H T T T H H T H T H	$\theta_1 =$
B	H H H H T H H H H H	
B	H T H H H H H T H H	
A	H T H T T T H H T T	$\theta_2 =$
B	T H H H T H H H T H	

# EM Intuition Building: Experiment #1

- We do 5 trials by picking a coin at random
- Then we flip the chosen coin 10 times

Trial	Results
A	H T T T H H T H T H
B	H H H H T H H H H H
B	H T H H H H H T H H
A	H T H T T T H H T T
B	T H H H T H H H T H

$$\theta_1 = \frac{\text{\# A heads}}{\text{total \# of A flips}}$$

$$\theta_2 = \frac{\text{\# B heads}}{\text{total \# of B flips}}$$

# EM Intuition Building: Experiment #1

Trial	Results
A	H T T T H H T H T H
B	H H H H T H H H H H
B	H T H H H H H T H H
A	H T H T T T H H T T
B	T H H H T H H H T H

# EM Intuition Building: Experiment #1

Trial	Results									
A	H	T	T	T	H	H	T	H	T	H
B	H	H	H	H	T	H	H	H	H	H
B	H	T	H	H	H	H	H	T	H	H
A	H	T	H	T	T	T	H	H	T	T
B	T	H	H	H	T	H	H	H	T	H

A	B
5 H, 5 T	
	9 H, 1 T
	8 H, 2 T
4 H, 6 T	
	7 H, 3 T

# EM Intuition Building: Experiment #1

Trial

A
B
B
A
B

Results

H T T T H H T H T H

H H H H T H H H H H

H T H H H H H T H H

H T H T T H H T T T

T H H H T H H H T H

A	B
5 H, 5 T	
	9 H, 1 T
	8 H, 2 T
4 H, 6 T	
	7 H, 3 T
24 H, 6 T	9 H, 11 T

# EM Intuition Building:

## Experiment #1

Trial

A
B
B
A
B

Results

H T T T H H T H T H

H H H H T H H H H H

H T H H H H H T H H

H T H T T H H T T T

T H H H T H H H H T H

A	B
5 H, 5 T	
	9 H, 1 T
	8 H, 2 T
4 H, 6 T	
	7 H, 3 T

24 H, 6 T    9 H, 11 T

$$\theta_1 = \frac{24}{24 + 6} = 0.8$$

$$\theta_2 = \frac{9}{9 + 11} = 0.45$$

# EM Intuition Building: Experiment #2

Trial	Results
?	H T T T H H T H T H
?	H H H H T H H H H H
?	H T H H H H H T H H
?	H T H T T T H H T T
?	T H H H T H H H T H

**Challenge:** What if we don't know which code was flipped during each trial? (The coin name becomes a **hidden variable**)

# EM Intuition Building: Experiment #2

Trial	Results									
?	H	T	T	T	H	H	T	H	T	H
?	H	H	H	H	T	H	H	H	H	H
?	H	T	H	H	H	H	H	T	H	H
?	H	T	H	T	T	T	H	H	T	T
?	T	H	H	H	T	H	H	H	T	H

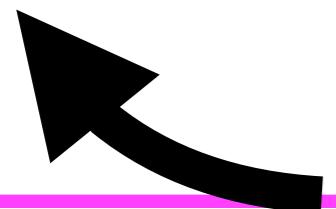
# EM Intuition Building: Experiment #2

Trial	Results									
?	H	T	T	T	H	H	T	H	T	H
?	H	H	H	H	T	H	H	H	H	H
?	H	T	H	H	H	H	H	T	H	H
?	H	T	H	T	T	T	H	H	T	T
?	T	H	H	H	T	H	H	H	T	H

**Step 1:**  $\hat{\theta}_A^{(0)} = 0.6$   
initialize  
parameters  $\hat{\theta}_B^{(0)} = 0.5$

# EM Intuition Building: Experiment #2

Trial	Results									
?	H	T	T	T	H	H	T	H	T	H
?	H	H	H	H	T	H	H	H	H	H
?	H	T	H	H	H	H	H	T	H	H
?	H	T	H	T	T	T	H	H	T	T
?	T	H	H	H	T	H	H	H	T	H



**Step 1:**  $\hat{\theta}_A^{(0)} = 0.6$   
initialize  
parameters  $\hat{\theta}_B^{(0)} = 0.5$

# EM Intuition Building: Experiment #2

2: E-Step

The diagram illustrates the EM algorithm's iterative process. A curved arrow labeled "2: E-Step" points from the "Results" section to the "Trial" section. Below this, another arrow points from the "Trial" section back up to the "Results" section.

Trial	Results	$p(x A)$	$p(x B)$
?	H T T T H H T H T H		
?	H H H H T H H H H H		
?	H T H H H H H T H H		
?	H T H T T T H H T T		
?	T H H H T H H H T H		

**Step 1:**  $\hat{\theta}_A^{(0)} = 0.6$   
initialize  
parameters  $\hat{\theta}_B^{(0)} = 0.5$

# EM Intuition Building: Experiment #2

2: E-Step

The diagram shows a curved arrow pointing from a pink box containing 'Step 1' parameters to the 'Results' section. A straight arrow points from the 'Results' section to the rightmost columns.

Trial	Results										$p(x A)$	$p(x B)$
?	H	T	T	T	H	H	T	H	T	H	0.45 × A, 0.55 × B	
?	H	H	H	H	T	H	H	H	H	H		
?	H	T	H	H	H	H	H	T	H	H		
?	H	T	H	T	T	T	H	H	T	T		
?	T	H	H	H	T	H	H	H	T	H		

**Step 1:**  $\hat{\theta}_A^{(0)} = 0.6$   
initialize  
parameters  $\hat{\theta}_B^{(0)} = 0.5$

# EM Intuition Building: Experiment #2

2: E-Step

Trial	Results	$p(x A)$	$p(x B)$	A	B
?	H T T T H H T H T H	$0.45 \times A, 0.55 \times B$		$\approx 2.2H, 2.2T$	$\approx 2.8H, 2.8T$
?	H H H H T H H H H H				
?	H T H H H H H T H H				
?	H T H T T T H H T T				
?	T H H H T H H H T H				

**Step 1:**  $\hat{\theta}_A^{(0)} = 0.6$   
initialize  
parameters  $\hat{\theta}_B^{(0)} = 0.5$

# EM Intuition Building: Experiment #2

2: E-Step

Trial	Results	$p(x A)$	$p(x B)$	A	B
?	H T T T H H T H T H	$0.45 \times A, 0.55 \times B$		$\approx 2.2H, 2.2T$	$\approx 2.8H, 2.8T$
?	H H H H T H H H H H	$0.80 \times A, 0.20 \times B$			
?	H T H H H H H T H H				
?	H T H T T T H H T T				
?	T H H H T H H H T H				

**Step 1:**  $\hat{\theta}_A^{(0)} = 0.6$   
initialize  
parameters  $\hat{\theta}_B^{(0)} = 0.5$

# EM Intuition Building: Experiment #2

2: E-Step

Trial	Results	$p(x A)$	$p(x B)$	A	B
?	H T T T H H T H T H	$0.45 \times A, 0.55 \times B$		$\approx 2.2H, 2.2T$	$\approx 2.8H, 2.8T$
?	H H H H T H H H H H	$0.80 \times A, 0.20 \times B$		$\approx 7.2H, 0.8T$	$\approx 1.8H, 0.2T$
?	H T H H H H H T H H				
?	H T H T T T H H T T				
?	T H H H T H H H T H				

**Step 1:**  $\hat{\theta}_A^{(0)} = 0.6$   
initialize  
parameters  $\hat{\theta}_B^{(0)} = 0.5$

# EM Intuition Building: Experiment #2

2: E-Step

Trial	Results	$p(x A)$	$p(x B)$	A	B
?	H T T T H H T H T H	$0.45 \times A, 0.55 \times B$		$\approx 2.2H, 2.2T$	$\approx 2.8H, 2.8T$
?	H H H H T H H H H H	$0.80 \times A, 0.20 \times B$		$\approx 7.2H, 0.8T$	$\approx 1.8H, 0.2T$
?	H T H H H H H T H H	$0.73 \times A, 0.27 \times B$		$\approx 5.9H, 1.5T$	$\approx 2.1H, 0.5T$
?	H T H T T H H T T T				
?	T H H H T H H H T H				

**Step 1:**  $\hat{\theta}_A^{(0)} = 0.6$   
initialize  
parameters  $\hat{\theta}_B^{(0)} = 0.5$

# EM Intuition Building: Experiment #2

2: E-Step

Trial	Results	$p(x A)$	$p(x B)$	A	B
?	H T T T H H T H T H	$0.45 \times A, 0.55 \times B$		$\approx 2.2H, 2.2T$	$\approx 2.8H, 2.8T$
?	H H H H T H H H H H	$0.80 \times A, 0.20 \times B$		$\approx 7.2H, 0.8T$	$\approx 1.8H, 0.2T$
?	H T H H H H H T H H	$0.73 \times A, 0.27 \times B$		$\approx 5.9H, 1.5T$	$\approx 2.1H, 0.5T$
?	H T H T T H H T T T	$0.35 \times A, 0.65 \times B$		$\approx 1.4H, 2.1T$	$\approx 2.6H, 3.9T$
?	T H H H T H H H T H				

**Step 1:**  $\hat{\theta}_A^{(0)} = 0.6$   
 initialize  
 parameters  $\hat{\theta}_B^{(0)} = 0.5$

# EM Intuition Building: Experiment #2

2: E-Step

Trial	Results	$p(x A)$	$p(x B)$	A	B
?	H T T T H H T H T H	$0.45 \times A, 0.55 \times B$		$\approx 2.2H, 2.2T$	$\approx 2.8H, 2.8T$
?	H H H H T H H H H H	$0.80 \times A, 0.20 \times B$		$\approx 7.2H, 0.8T$	$\approx 1.8H, 0.2T$
?	H T H H H H H T H H	$0.73 \times A, 0.27 \times B$		$\approx 5.9H, 1.5T$	$\approx 2.1H, 0.5T$
?	H T H T T H H T T T	$0.35 \times A, 0.65 \times B$		$\approx 1.4H, 2.1T$	$\approx 2.6H, 3.9T$
?	T H H H T H H H T H	$0.65 \times A, 0.25 \times B$		$\approx 4.5H, 1.9T$	$\approx 2.5H, 1.1T$

此时A出现trial1情况 概率为 $0.6^5 * 0.4^5$   
B为 $0.5^{10}$ , 得到 $0.45 : 0.55$

Step 1:  $\hat{\theta}_A^{(0)} = 0.6$   
initialize  
parameters  $\hat{\theta}_B^{(0)} = 0.5$

# EM Intuition Building: Experiment #2

2: E-Step

Trial	Results	$p(x A)$	$p(x B)$	A	B
?	H T T T H H T H T H	$0.45 \times A, 0.55 \times B$		$\approx 2.2H, 2.2T$	$\approx 2.8H, 2.8T$
?	H H H H T H H H H H	$0.80 \times A, 0.20 \times B$		$\approx 7.2H, 0.8T$	$\approx 1.8H, 0.2T$
?	H T H H H H H T H H	$0.73 \times A, 0.27 \times B$		$\approx 5.9H, 1.5T$	$\approx 2.1H, 0.5T$
?	H T H T T H H T T T	$0.35 \times A, 0.65 \times B$		$\approx 1.4H, 2.1T$	$\approx 2.6H, 3.9T$
?	T H H H T H H H T H	$0.65 \times A, 0.25 \times B$		$\approx 4.5H, 1.9T$	$\approx 2.5H, 1.1T$
				$\approx 21.3H, 8.6T$	$\approx 11.7H, 8.4T$

此时A出现trial1情况概率为 $0.6^5 * 0.4^5$   
B为 $0.5^{10}$ , 得到 $0.45 : 0.55$

**Step 1:**  $\hat{\theta}_A^{(0)} = 0.6$   
 initialize  
 parameters  $\hat{\theta}_B^{(0)} = 0.5$

# EM Intuition Building: Experiment #2

2: E-Step

Trial	Results	$p(x A)$	$p(x B)$	A	B
?	H T T T H H T H T H	$0.45 \times A, 0.55 \times B$		$\approx 2.2H, 2.2T$	$\approx 2.8H, 2.8T$
?	H H H H T H H H H H	$0.80 \times A, 0.20 \times B$		$\approx 7.2H, 0.8T$	$\approx 1.8H, 0.2T$
?	H T H H H H H T H H	$0.73 \times A, 0.27 \times B$		$\approx 5.9H, 1.5T$	$\approx 2.1H, 0.5T$
?	H T H T T H H T T T	$0.35 \times A, 0.65 \times B$		$\approx 1.4H, 2.1T$	$\approx 2.6H, 3.9T$
?	T H H H T H H H T H	$0.65 \times A, 0.25 \times B$		$\approx 4.5H, 1.9T$	$\approx 2.5H, 1.1T$
				$\approx 21.3H, 8.6T$	$\approx 11.7H, 8.4T$

**Step 1:**  $\hat{\theta}_A^{(0)} = 0.6$   
 initialize  
 parameters  $\hat{\theta}_B^{(0)} = 0.5$

3: M-Step

# EM Intuition Building: Experiment #2

2: E-Step

Trial	Results	$p(x A)$	$p(x B)$	A	B
?	H T T T H H T H T H	$0.45 \times A, 0.55 \times B$		$\approx 2.2H, 2.2T$	$\approx 2.8H, 2.8T$
?	H H H H T H H H H H	$0.80 \times A, 0.20 \times B$		$\approx 7.2H, 0.8T$	$\approx 1.8H, 0.2T$
?	H T H H H H H T H H	$0.73 \times A, 0.27 \times B$		$\approx 5.9H, 1.5T$	$\approx 2.1H, 0.5T$
?	H T H T T H H T T T	$0.35 \times A, 0.65 \times B$		$\approx 1.4H, 2.1T$	$\approx 2.6H, 3.9T$
?	T H H H T H H H T H	$0.65 \times A, 0.25 \times B$		$\approx 4.5H, 1.9T$	$\approx 2.5H, 1.1T$
				$\approx 21.3H, 8.6T$	$\approx 11.7H, 8.4T$

**Step 1:**  $\hat{\theta}_A^{(0)} = 0.6$   
 initialize  
 parameters  $\hat{\theta}_B^{(0)} = 0.5$

$$\hat{\theta}_A^{(1)} \approx \frac{21.3}{21.3 + 8.6} \approx$$

$$\hat{\theta}_B^{(1)} \approx \frac{11.7}{11.7 + 8.4} \approx$$

3: M-Step

# EM Intuition Building: Experiment #2

2: E-Step

Trial	Results	$p(x A)$	$p(x B)$	A	B
?	H T T T H H T H T H	$0.45 \times A, 0.55 \times B$		$\approx 2.2H, 2.2T$	$\approx 2.8H, 2.8T$
?	H H H H T H H H H H	$0.80 \times A, 0.20 \times B$		$\approx 7.2H, 0.8T$	$\approx 1.8H, 0.2T$
?	H T H H H H H T H H	$0.73 \times A, 0.27 \times B$		$\approx 5.9H, 1.5T$	$\approx 2.1H, 0.5T$
?	H T H T T H H T T T	$0.35 \times A, 0.65 \times B$		$\approx 1.4H, 2.1T$	$\approx 2.6H, 3.9T$
?	T H H H T H H H T H	$0.65 \times A, 0.25 \times B$		$\approx 4.5H, 1.9T$	$\approx 2.5H, 1.1T$
				$\approx 21.3H, 8.6T$	$\approx 11.7H, 8.4T$

**Step 1:**  $\hat{\theta}_A^{(0)} = 0.6$   
 initialize  
 parameters  $\hat{\theta}_B^{(0)} = 0.5$

$$\hat{\theta}_A^{(1)} \approx \frac{21.3}{21.3 + 8.6} \approx 0.71$$

$$\hat{\theta}_B^{(1)} \approx \frac{11.7}{11.7 + 8.4} \approx 0.58$$

3: M-Step

# EM Intuition Building: Experiment #2

2: E-Step

Trial	Results	$p(x A)$	$p(x B)$	A	B
?	H T T T H H T H T H	$0.45 \times A, 0.55 \times B$		$\approx 2.2H, 2.2T$	$\approx 2.8H, 2.8T$
?	H H H H T H H H H H	$0.80 \times A, 0.20 \times B$		$\approx 7.2H, 0.8T$	$\approx 1.8H, 0.2T$
?	H T H H H H H T H H	$0.73 \times A, 0.27 \times B$		$\approx 5.9H, 1.5T$	$\approx 2.1H, 0.5T$
?	H T H T T H H T T T	$0.35 \times A, 0.65 \times B$		$\approx 1.4H, 2.1T$	$\approx 2.6H, 3.9T$
?	T H H H T H H H T H	$0.65 \times A, 0.25 \times B$		$\approx 4.5H, 1.9T$	$\approx 2.5H, 1.1T$
				$\approx 21.3H, 8.6T$	$\approx 11.7H, 8.4T$

**Step 1:** initialize parameters  $\hat{\theta}_A^{(0)} = 0.6$   
 $\hat{\theta}_B^{(0)} = 0.5$

$$\hat{\theta}_A^{(1)} \approx \frac{21.3}{21.3 + 8.6} \approx 0.71$$

$$\hat{\theta}_B^{(1)} \approx \frac{11.7}{11.7 + 8.4} \approx 0.58$$

3: M-Step

4: Finish

$\hat{\theta}_A^{(10)} \approx 0.80$   
 $\hat{\theta}_B^{(10)} \approx 0.52$

# What happens if we have more than two coins? A mixture of K Gaussians

- A distribution generated by randomly selecting one of K Gaussians, then randomly draw a point from that distribution.
- Gaussian  $k$  with a probability of  $p_k$

$$p(x; m, \sigma) = \sum_{k=1}^K p_k g(x; m_k, \sigma_k)$$

- Goal: find  $p_k$ ,  $\sigma_k$ ,  $m_k$  that maximize the probability of all our data points  $x$ .

# What happens if we have more than two coins? A mixture of K Gaussians

- A distribution generated by randomly selecting one of K Gaussians, then randomly draw a point from that distribution.
- Gaussian k with a probability of  $p_k$

probability of a  $x$  being generated from a gaussian with mean  $m$  and variance  $\sigma$

$$p(x; m, \sigma) = \sum_{k=1}^K p_k g(x; m_k, \sigma_k)$$

- Goal: find  $p_k$ ,  $\sigma_k$ ,  $m_k$  that maximize the probability of all our data points  $x$ .

# Back to EM

- Iterative Algorithm
- Goal: group some primitives together
- Chicken and Egg problem:
  - Items in group -> Description of the group
  - Description of the group -> Items in group

# Brace Yourselves..



# EM

- Iterative Algorithm: E Step and M Step
- E Step:
- Compute the probability that point n is generated by distribution k

$$p^{(i)}(k|n) = \frac{p_k^{(i)} g(x_n; m_k^{(i)}, \sigma_k^{(i)})}{\sum_{m=1}^K p_k^{(i)} g(x_n; m_k^{(i)}, \sigma_k^{(i)})}$$

# EM

- M Step:

$$m_k^{(i+1)} = \frac{\sum_{n=1}^N p^{(i)}(k|n)x_n}{\sum_{n=1}^N p^{(i)}(k|n)}$$

$$\sigma_k^{(i+1)} = \sqrt{\frac{1}{D} \frac{\sum_{n=1}^N p^{(i)}(k|n) ||x_n - m_k^{(i+1)}||^2}{\sum_{n=1}^N p^{(i)}(k|n)}}$$

$$p_k^{(i+1)} = \frac{1}{N} \sum_{n=1}^N p^{(i)}(k|n)$$

# EM

- Converges to a locally optimal solution
- Each step increases the probability of the points given the distributions.
- Can get stuck in local optima  
(less than K-Means)

# EM and K-means

# EM and K-means

- Notice the similarity between EM for Normal mixtures and K-means.
  - The expectation step is the assignment.
  - The maximization step is the update of centers

# EM and K-means

- Notice the similarity between EM for Normal mixtures and K-means.
  - The expectation step is the assignment.
  - The maximization step is the update of centers
- K-means is a simplified EM.

# EM and K-means

- Notice the similarity between EM for Normal mixtures and K-means.
  - The expectation step is the assignment.
  - The maximization step is the update of centers
- K-means is a simplified EM.
- K-means makes a hard decision while EM makes a soft decision when updating the parameters of the model.

# Ethics, Society, and Computing (ESC)

## Undergraduate Student Mixer

Free food, games, and conversation.  
No previous ESC experience necessary.

Monday, February 24, 2020  
4:30–6:30 p.m., drop-ins welcome  
South Lounge, Michigan Union

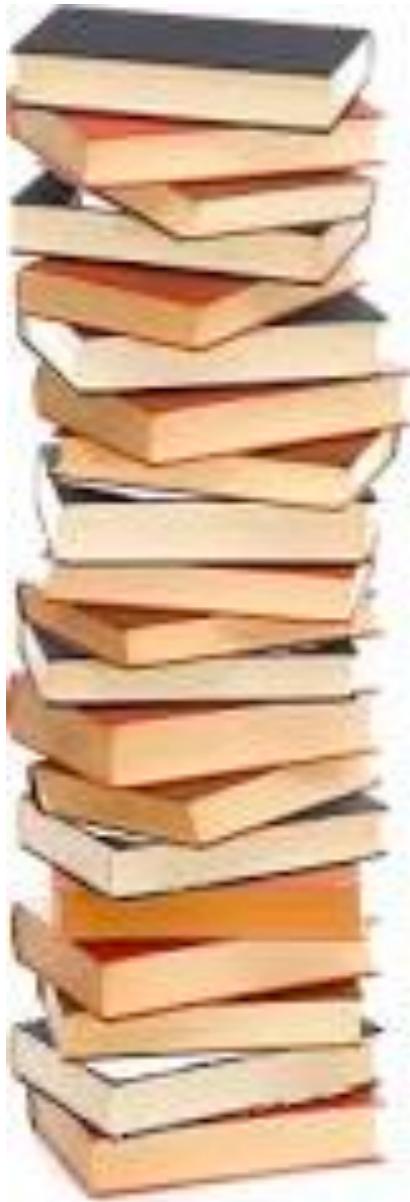


[esc.umich.edu](http://esc.umich.edu)



# Topic Modeling: Latent Dirichlet Allocation

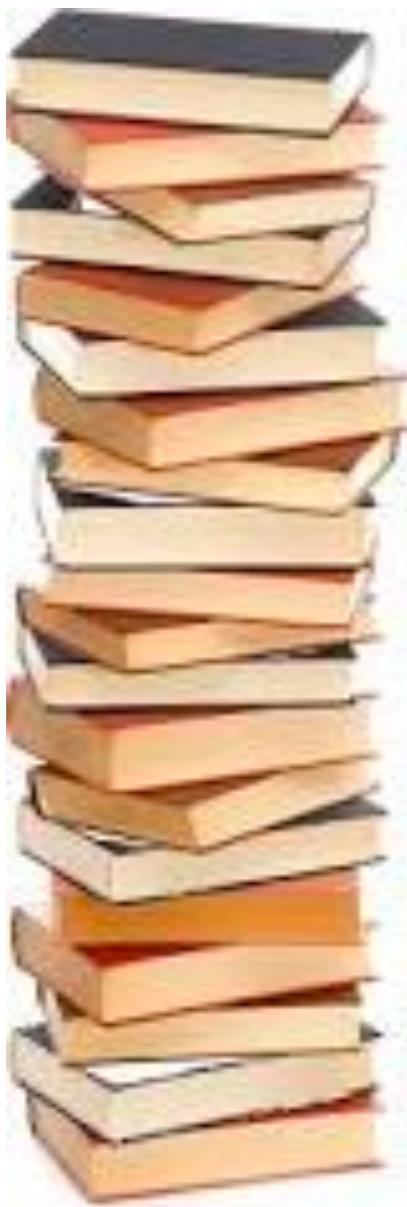
# Why Use Topic Models?



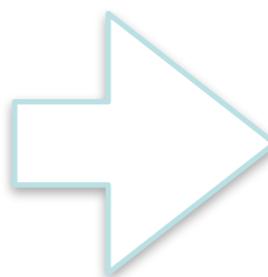
- Often you have a large collection of text
  - Decades of books
  - Newly released gov't docs
  - Wikipedia
- We want to know what happens in it
- But it's infeasible to read it all!
- Topic models provide an automatic way to get high-level themes from a corpus

# Topic Modeling at a high level

An input corpus



How many topics? k



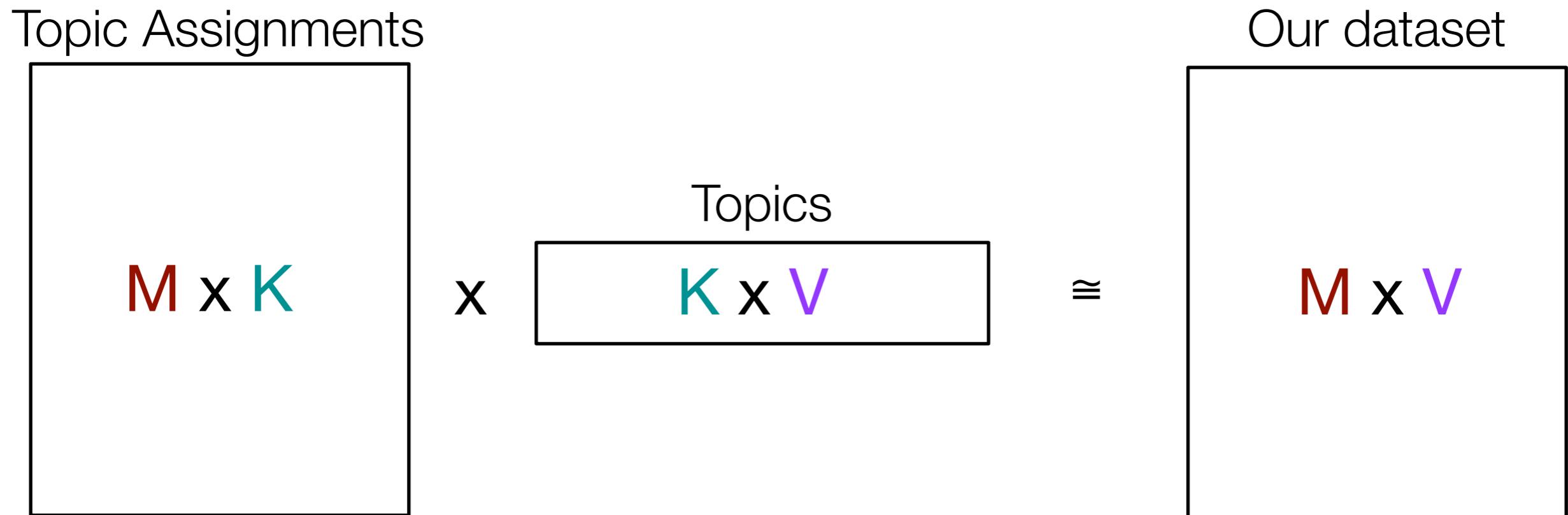
- Each doc's topics
- Each topic's words

# Example: Topics

human genome	evolution	disease	computer models
dna	species	bacteria	information
genetic	organisms	diseases	data
genes	life	resistance	computers
sequence	origin	bacterial	system
gene	biology	new	network
molecular	groups	strains	systems
sequencing	phylogenetic	control	model
map	living	infectious	parallel
information	diversity	malaria	methods
genetics	group	parasite	networks
mapping	new	parasites	software
project	two	united	new
sequences	common	tuberculosis	simulations

- Example from David Blei's slides

# Topic Modeling is a kind of Matrix Factorization



$M$  documents

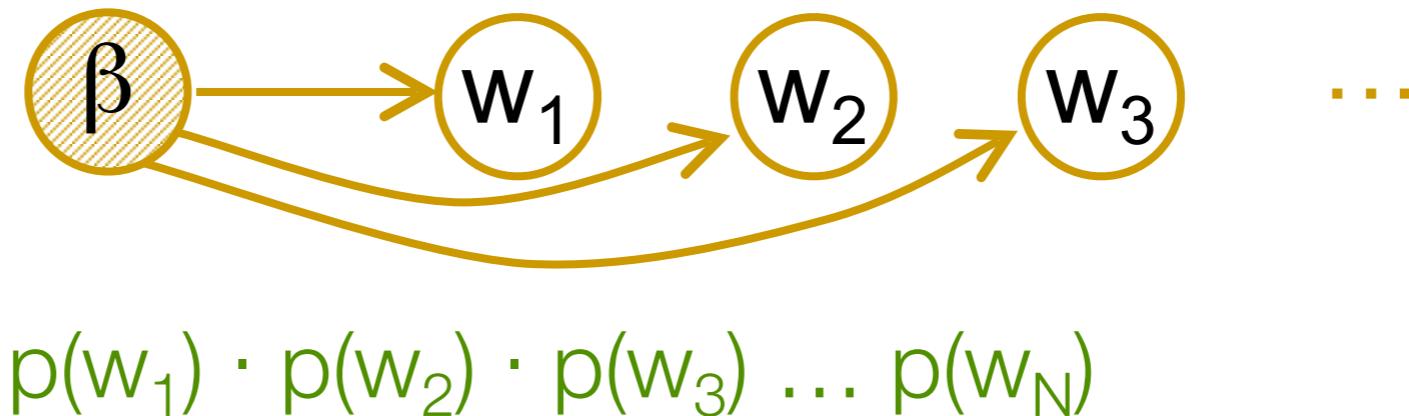
$V$  unique words (vocabulary)

$K$  topics

# Demo Time!

<https://mimno.infosci.cornell.edu/jsLDA/j lda.html>

# Consider a unigram model for generating text



# We can explicitly show model's parameters $\beta$

$\beta$  is a **vector** that says which unigrams are likely

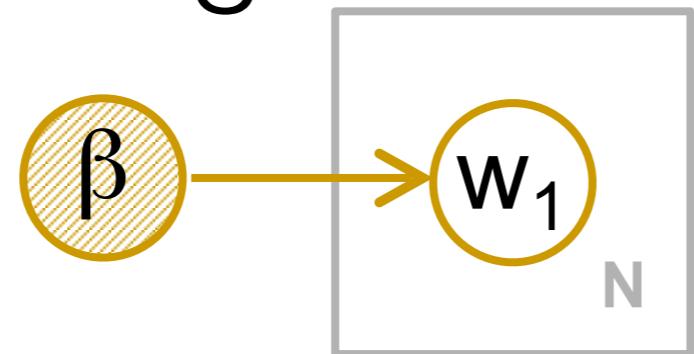


$$p(\beta) \cdot p(w_1 | \beta) \cdot p(w_2 | \beta) \cdot p(w_3 | \beta) \dots p(w_N | \beta)$$

The arrows show that each word is generated from the **same** parameters,  $\beta$

# Plate notation simplifies diagram

$\beta$  is a **vector** that says which unigrams are likely

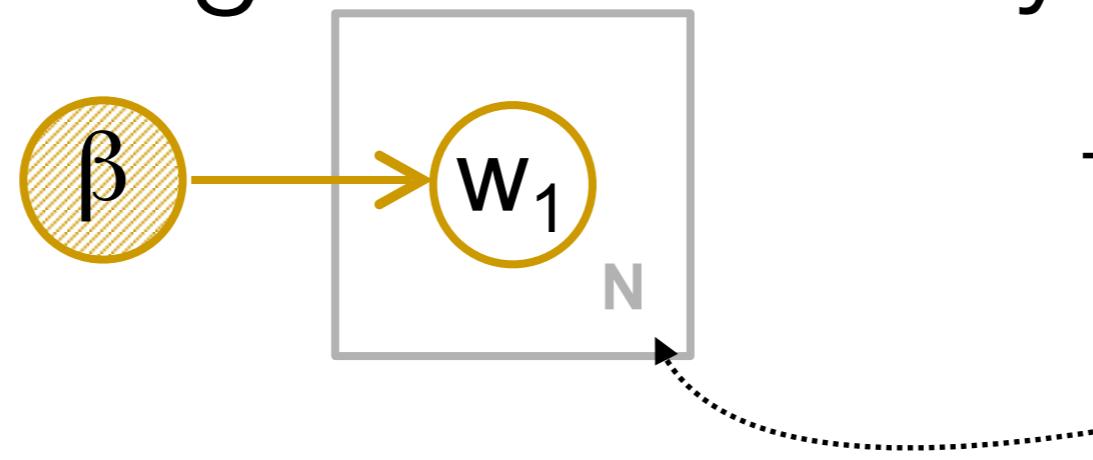


$$p(\beta) \cdot p(w_1 | \beta) \cdot p(w_2 | \beta) \cdot p(w_3 | \beta) \dots$$

The arrows show that each word is generated from the **same** parameters,  $\beta$

# Plate notation simplifies diagram

$\beta$  is a **vector** that says which unigrams are likely



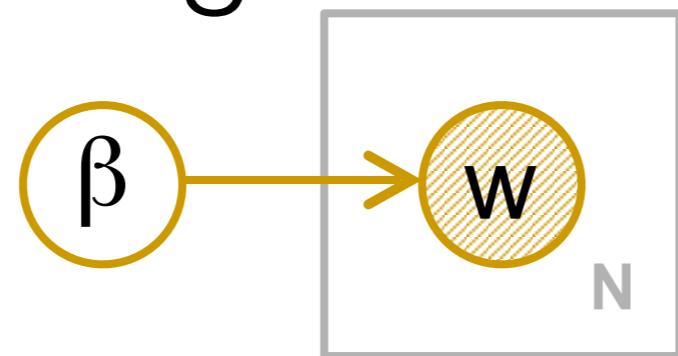
The  $N$  says how many things are generated

$$p(\beta) \cdot p(w_1 | \beta) \cdot p(w_2 | \beta) \cdot p(w_3 | \beta) \dots$$

The arrows show that each word is generated from the **same** parameters,  $\beta$

# Learn $\beta$ from observed words

$\beta$  is a **vector** that says which unigrams are likely



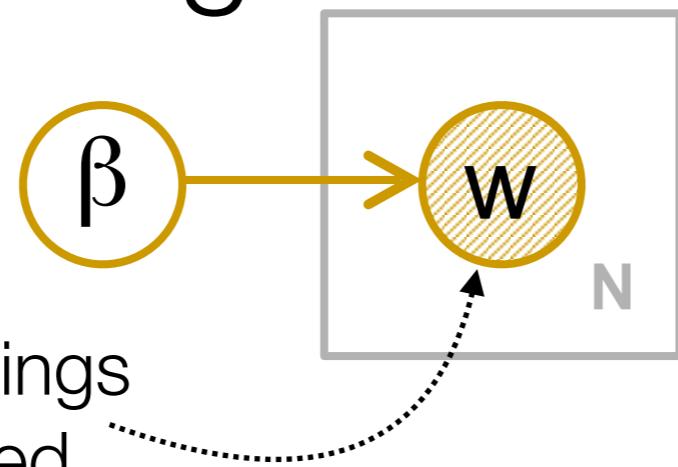
The  $N$  says how many things are generated

$$p(\beta) \cdot p(w_1 | \beta) \cdot p(w_2 | \beta) \cdot p(w_3 | \beta) \dots$$

The arrows show that each word is generated from the **same** parameters,  $\beta$

# Learn $\beta$ from observed words

$\beta$  is a **vector** that says which unigrams are likely



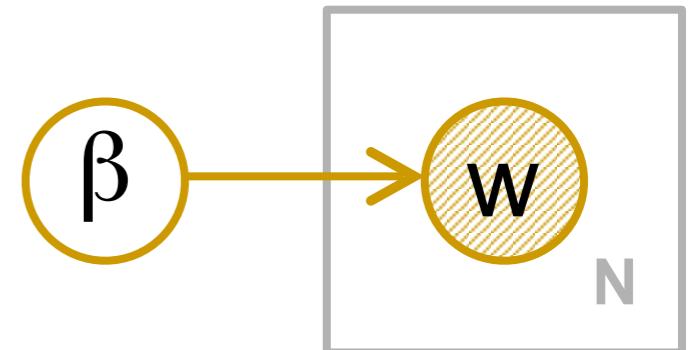
In Plate notation, the things we observe are shaded

The  $N$  says how many things are generated

$$p(\beta) \cdot p(w_1 | \beta) \cdot p(w_2 | \beta) \cdot p(w_3 | \beta) \dots$$

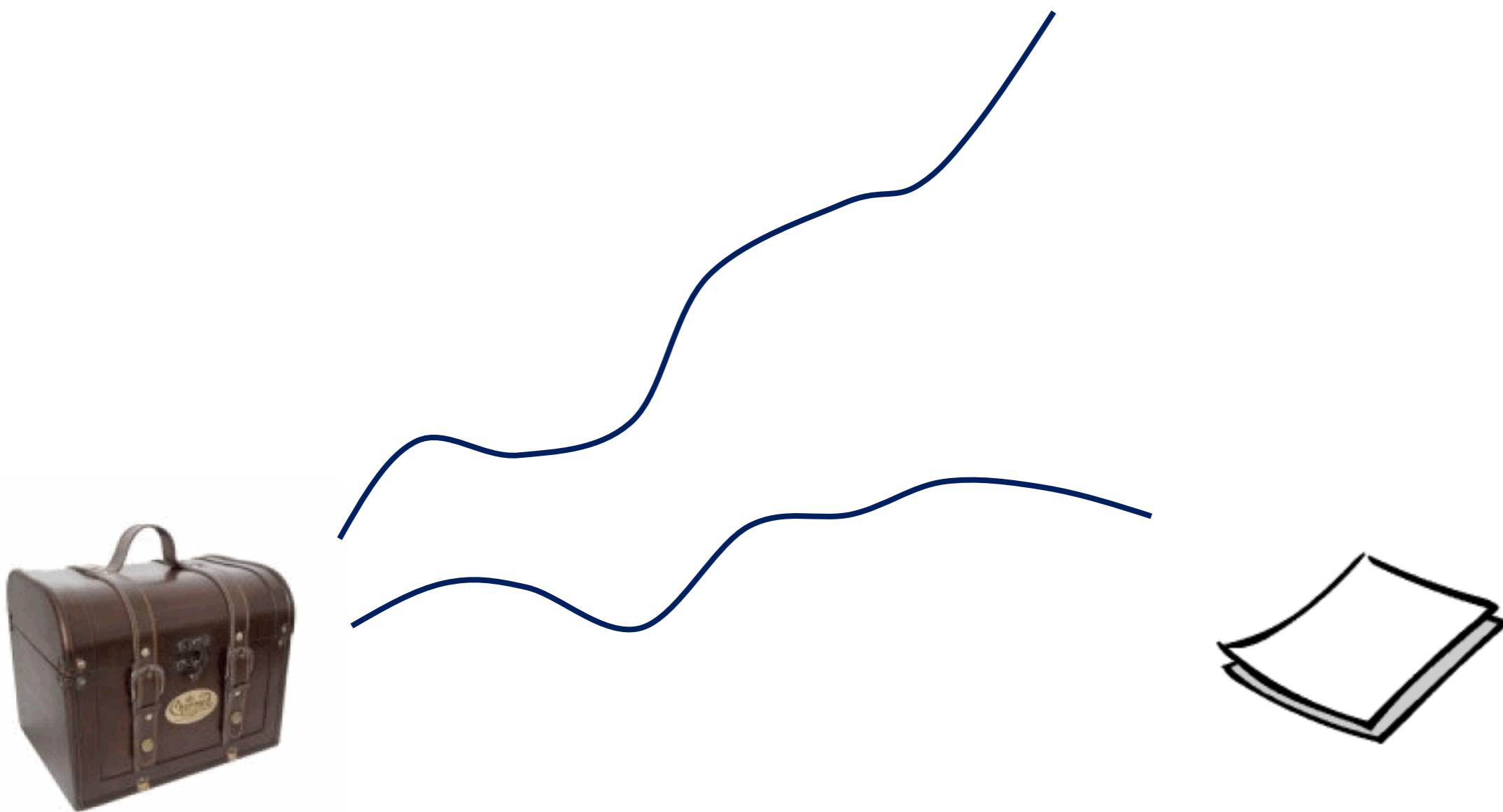
The arrows show that each word is generated from the **same** parameters,  $\beta$

# Extending this model to topics

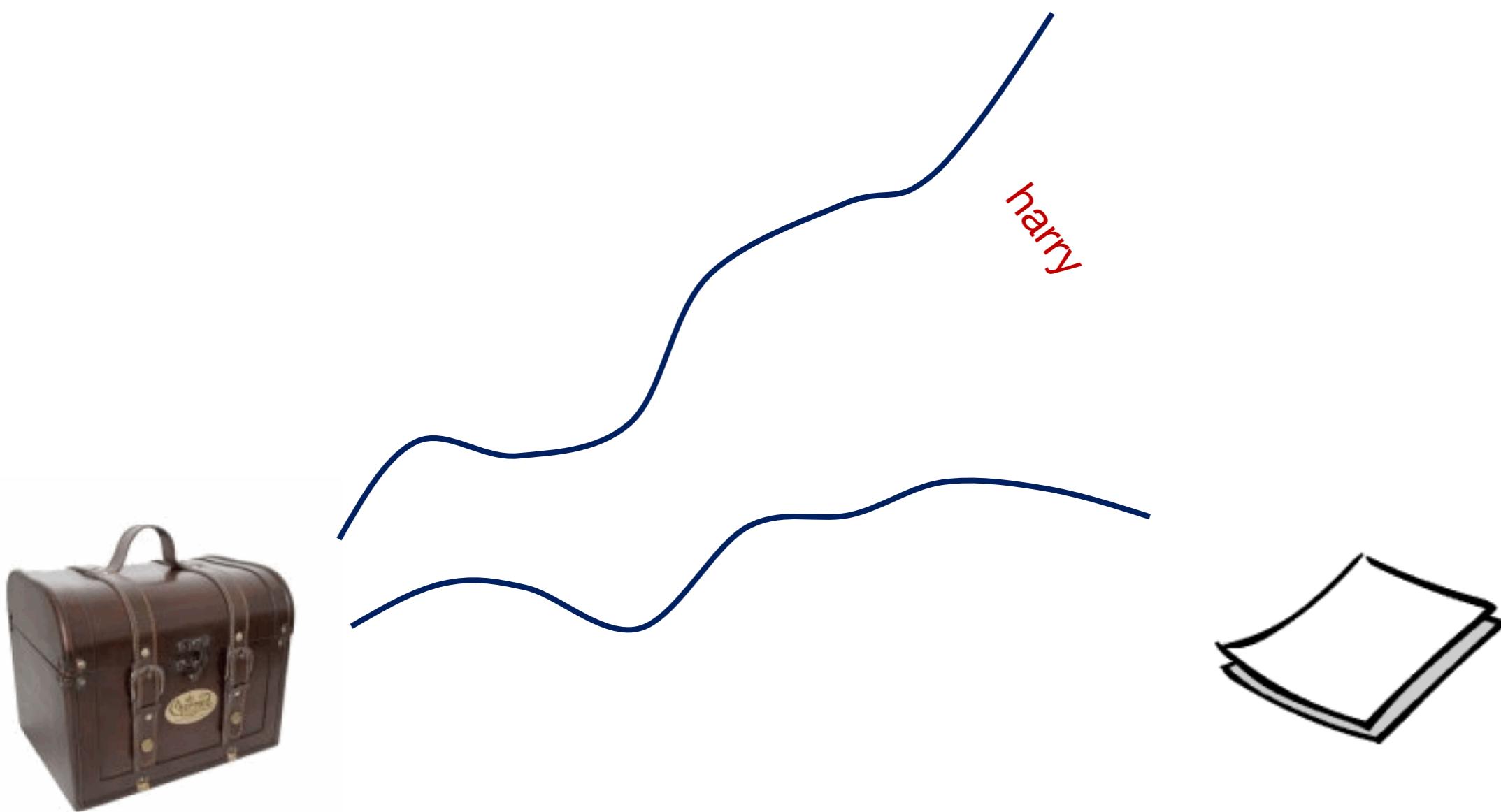


- $\beta$  represents a probability distribution for generating words in all documents
- But many documents are on different topics
- What if we wanted wanted to have different  $\beta$  for different topics?

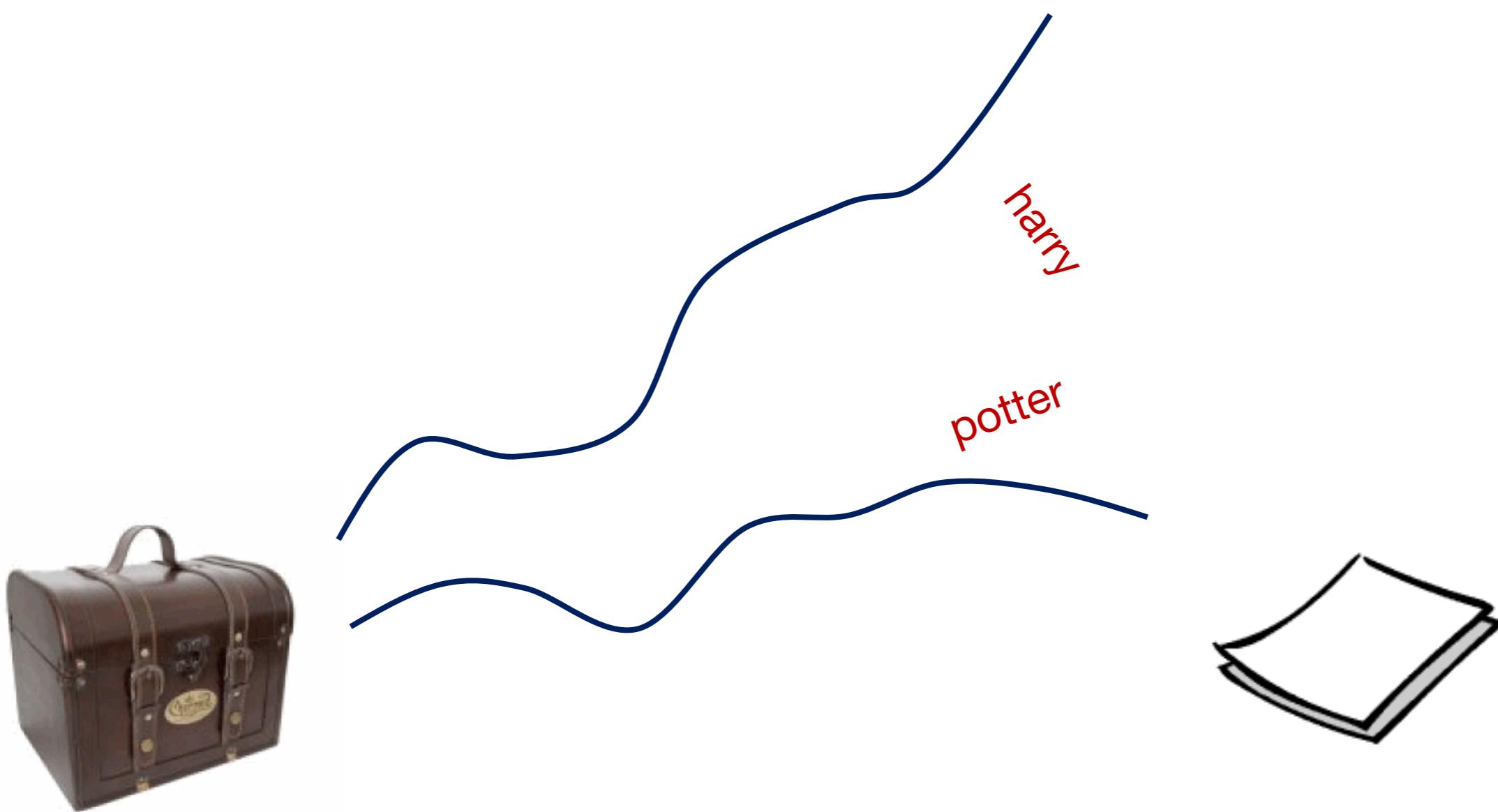
# Generative Model of Text



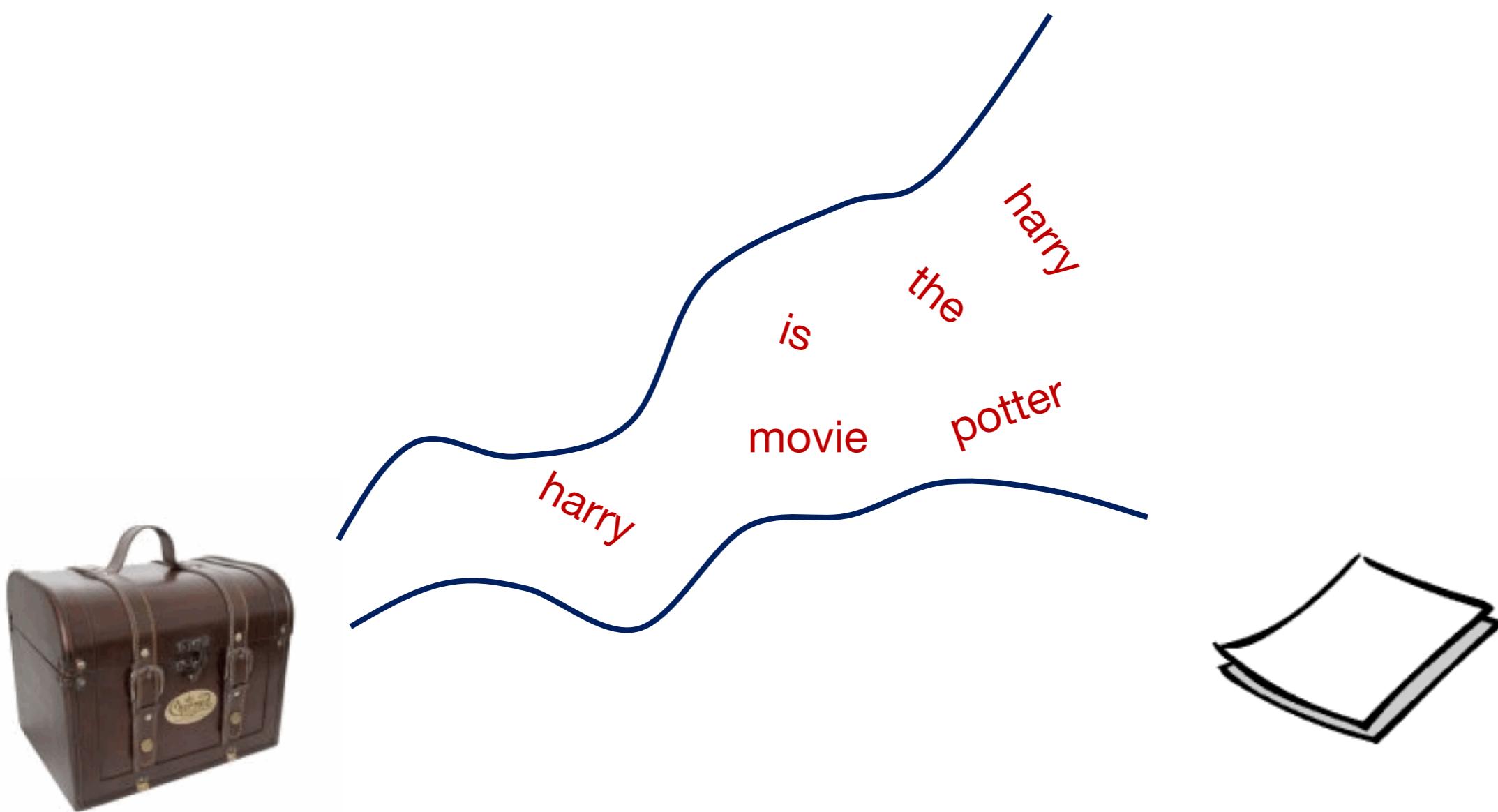
# Generative Model of Text



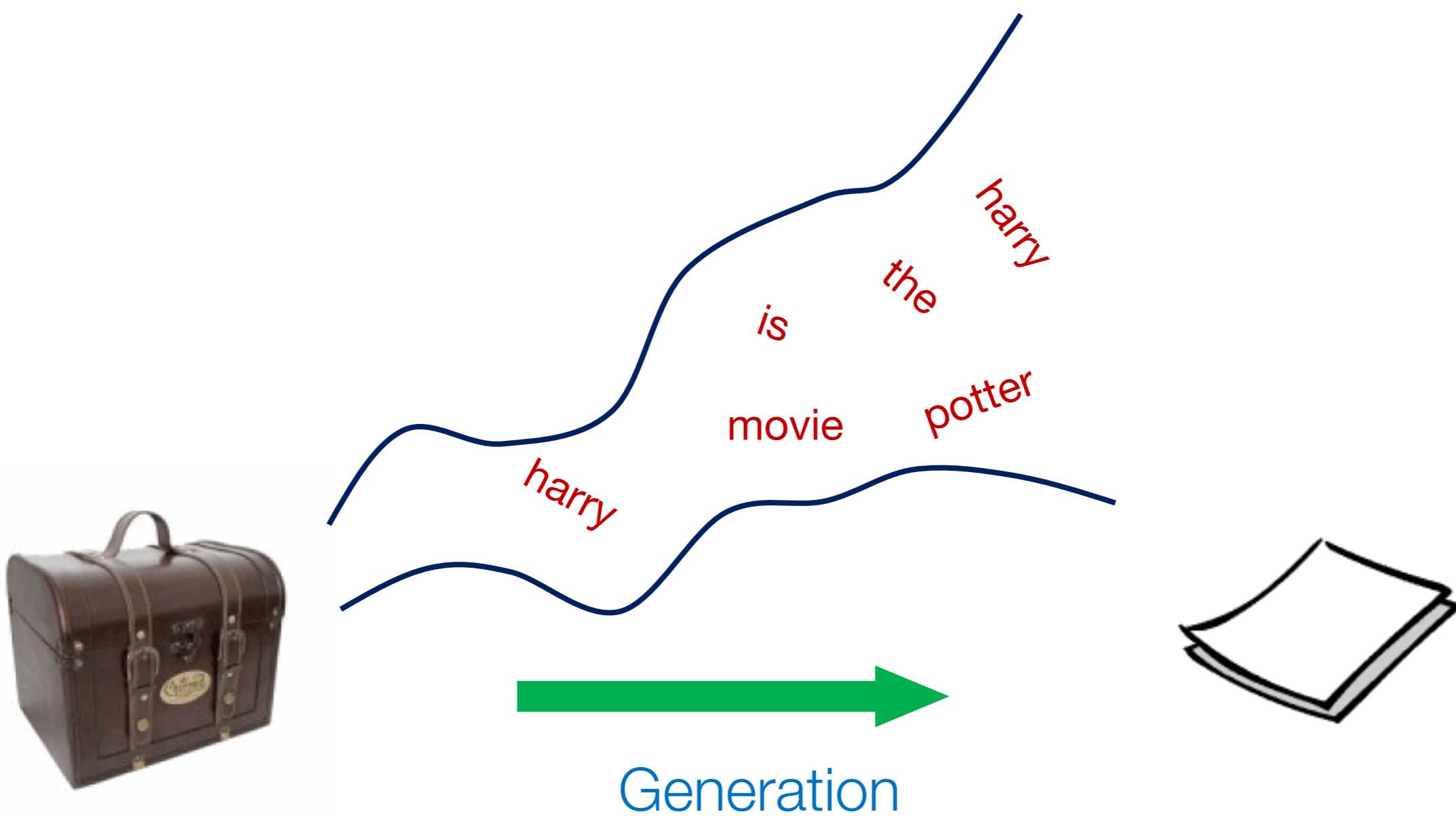
# Generative Model of Text



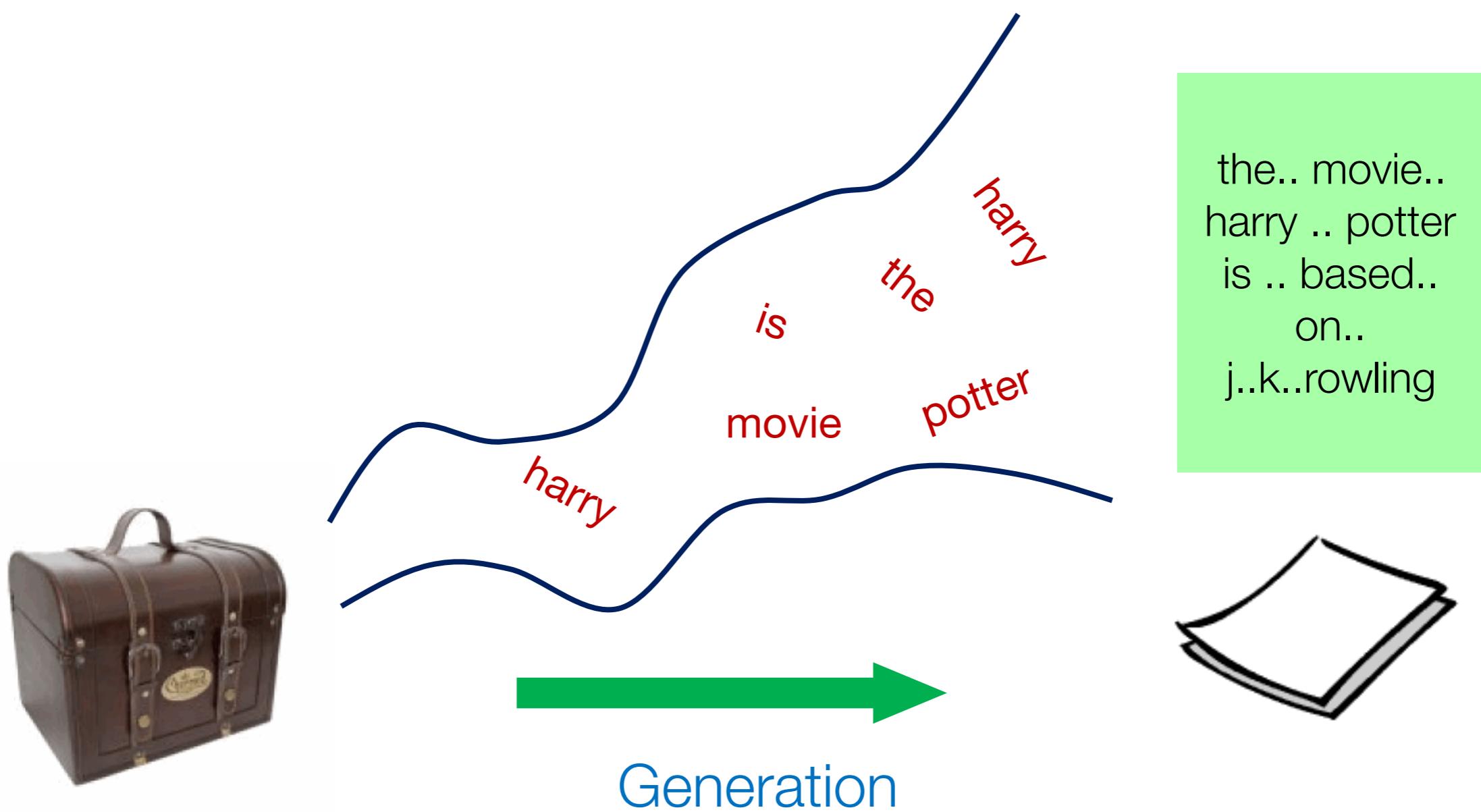
# Generative Model of Text



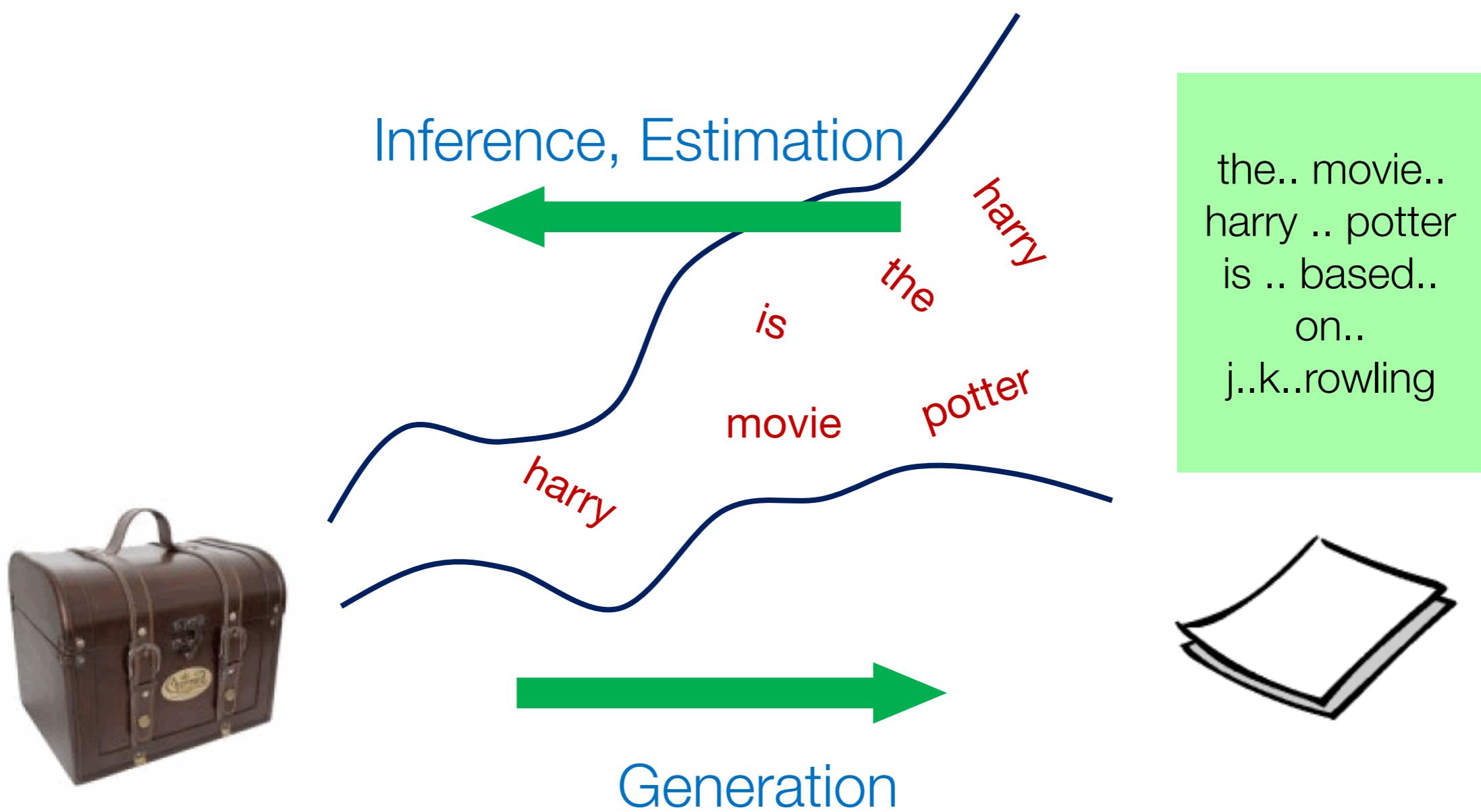
# Generative Model of Text



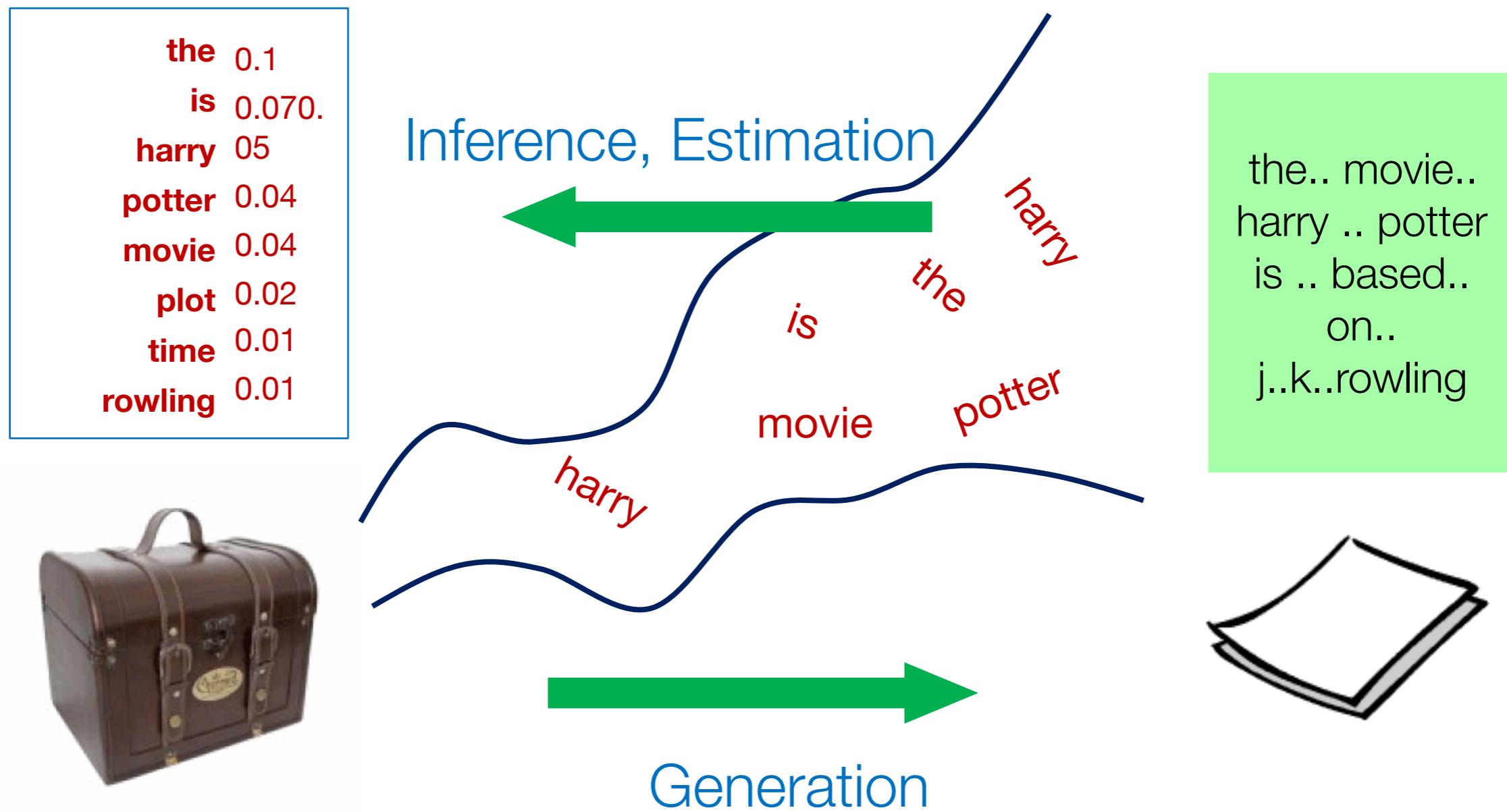
# Generative Model of Text



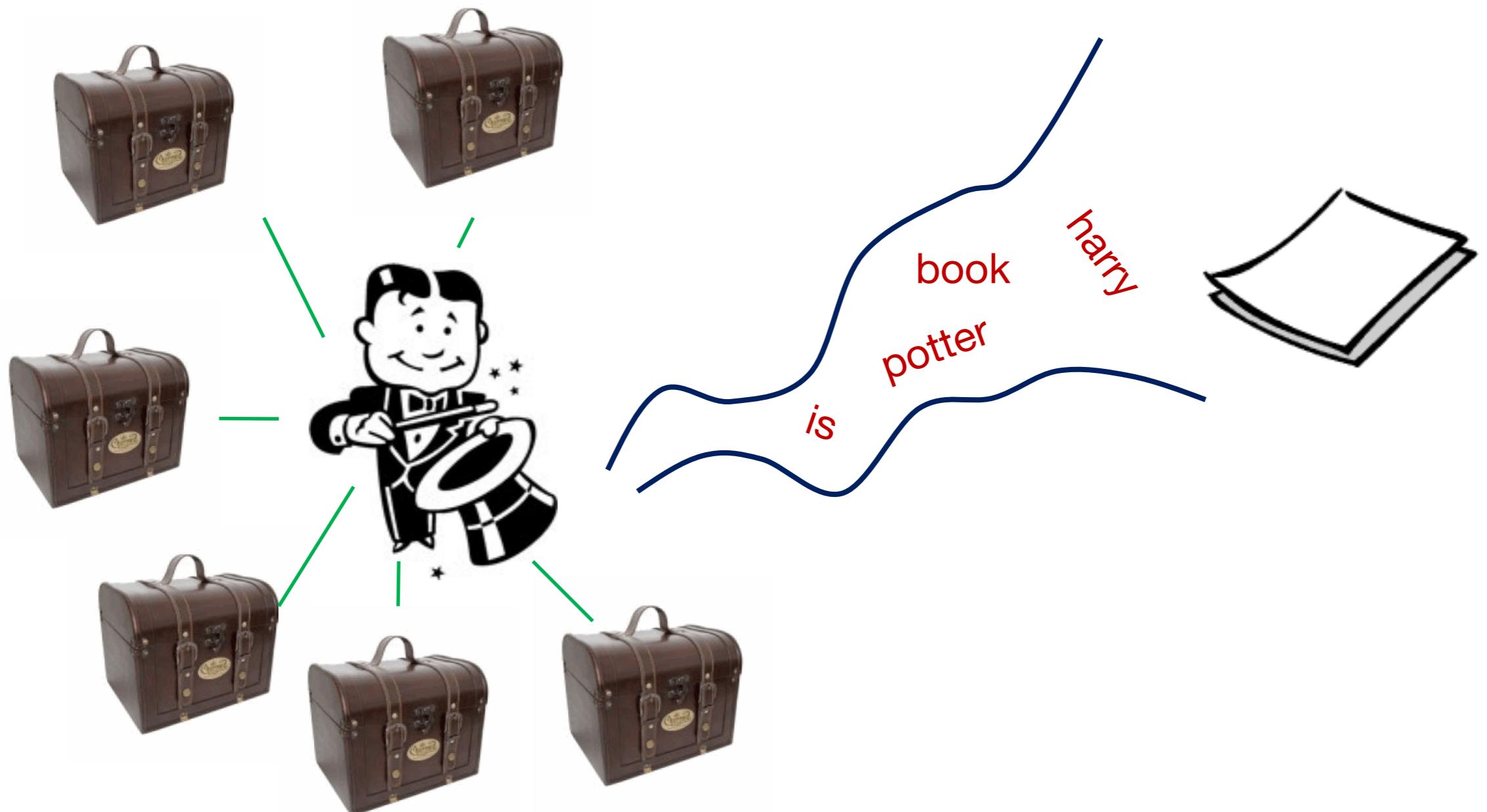
# Generative Model of Text



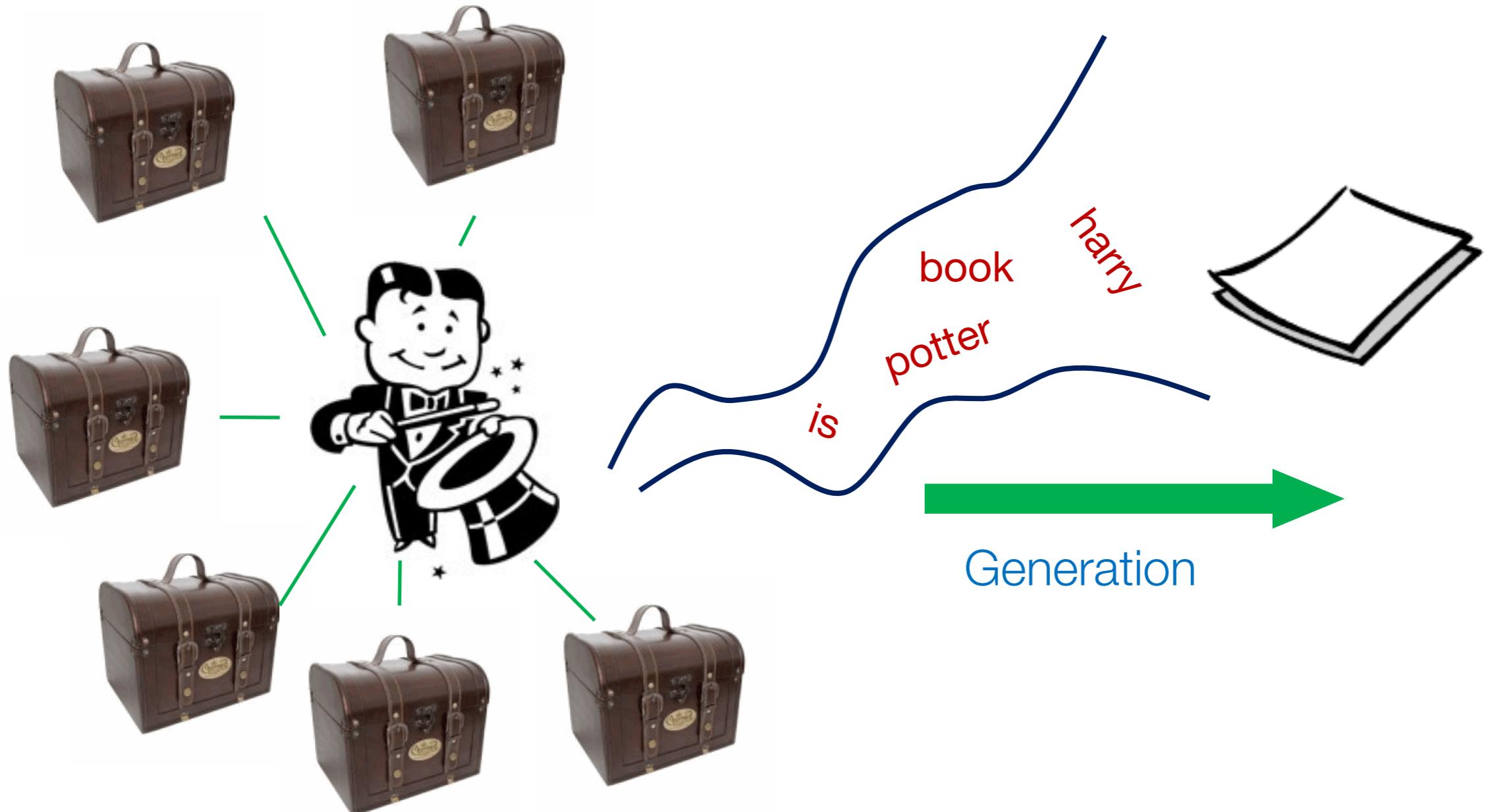
# Generative Model of Text



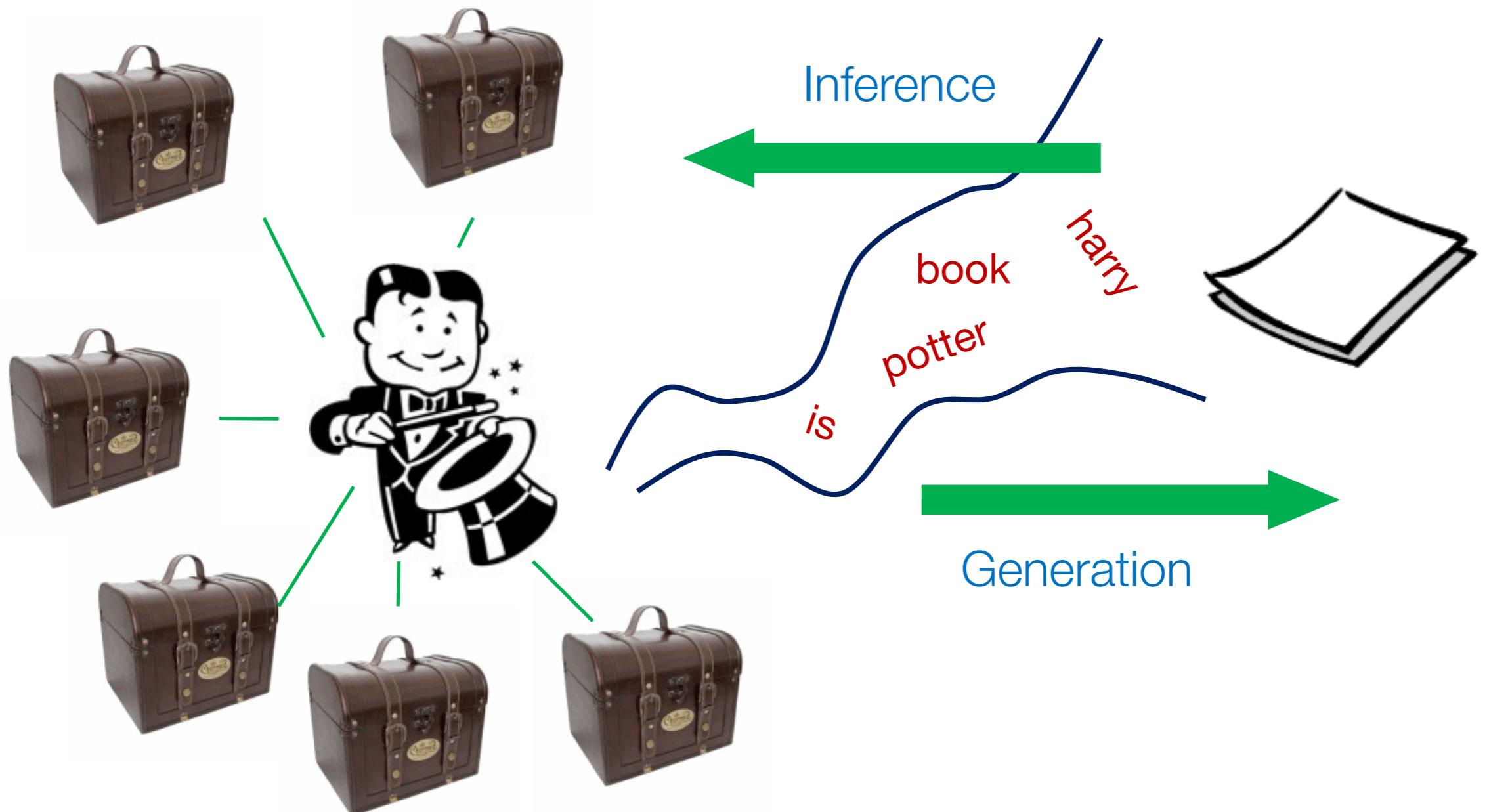
# A Generative Model can be Much More Complicated (e.g., a Mixture Model)



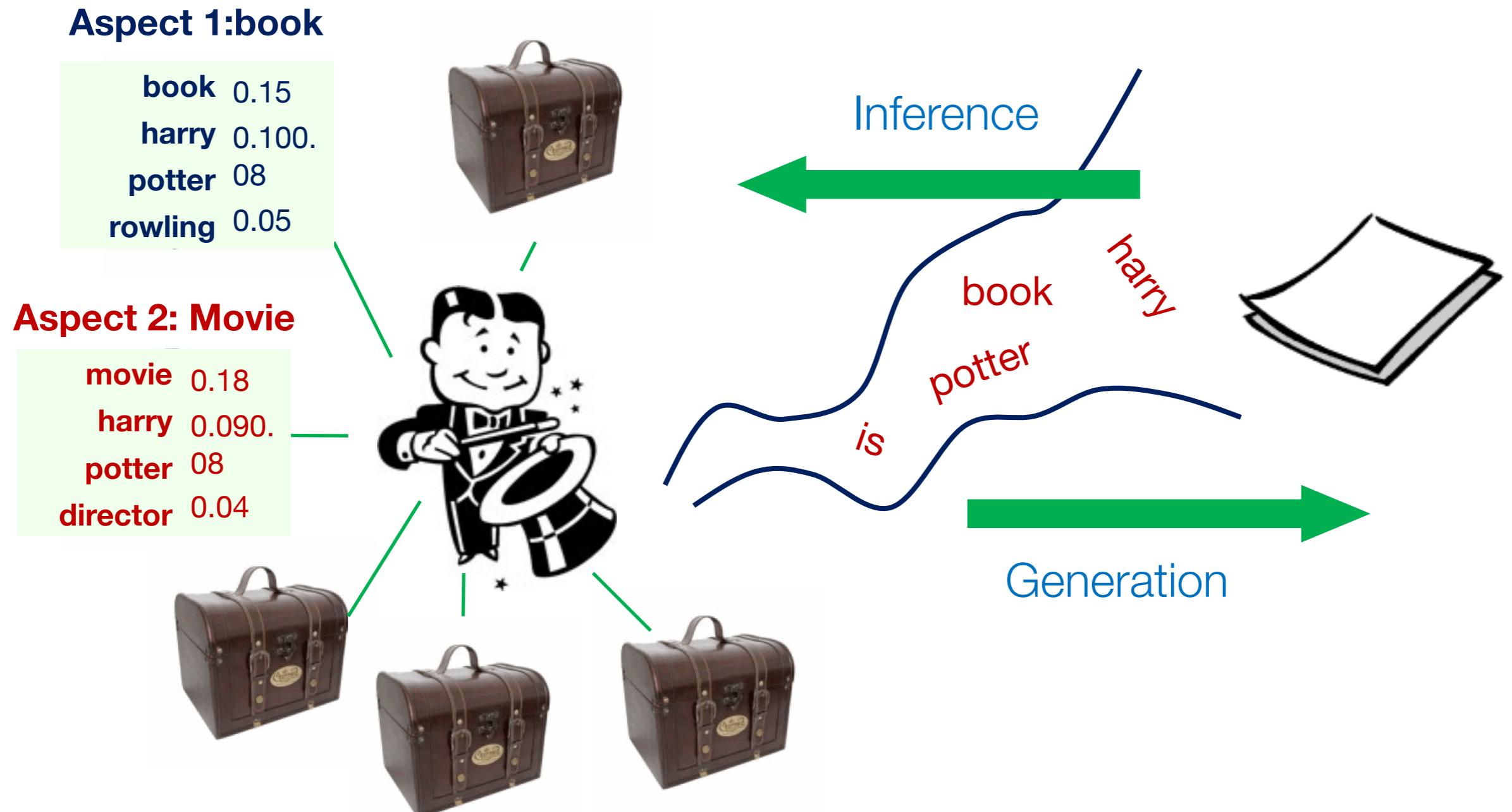
# A Generative Model can be Much More Complicated (e.g., a Mixture Model)



# A Generative Model can be Much More Complicated (e.g., a Mixture Model)



# A Generative Model can be Much More Complicated (e.g., a Mixture Model)



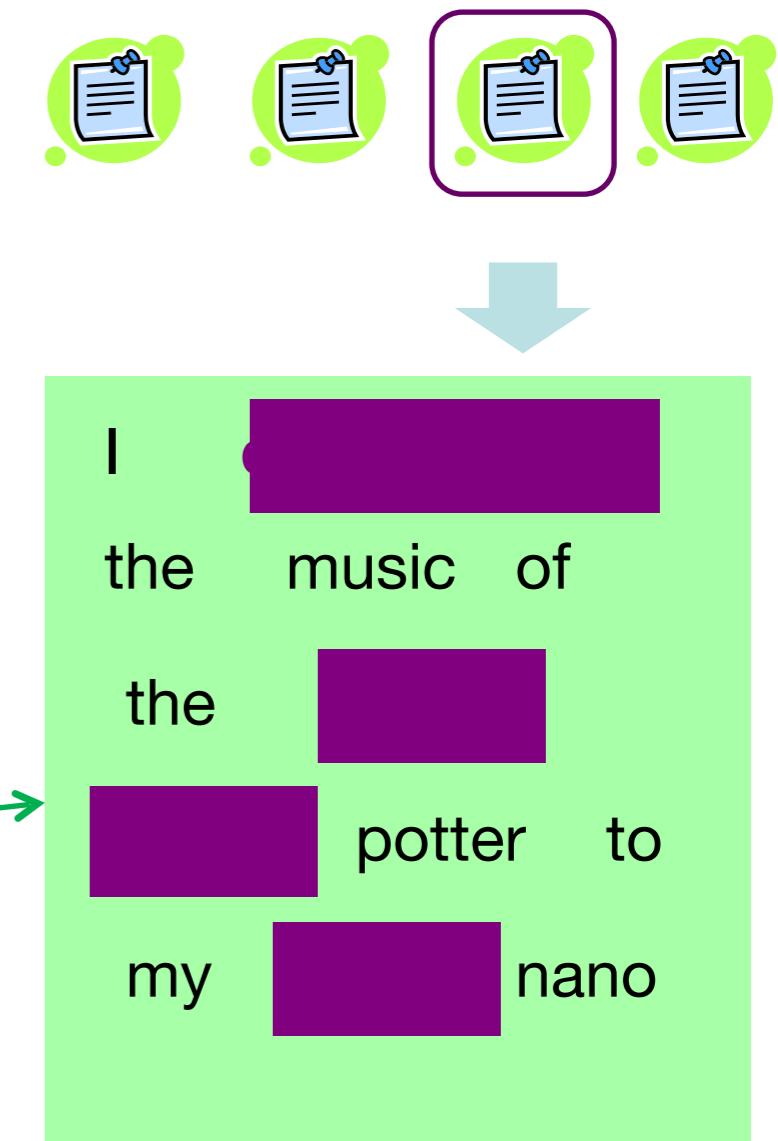
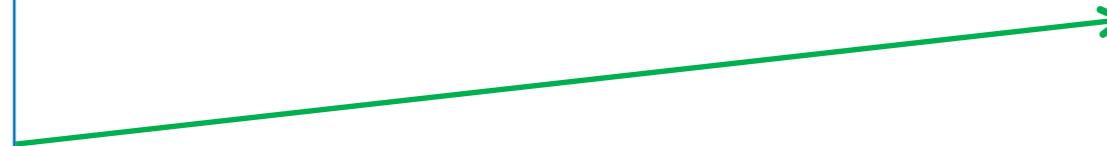
# The Simplest Generative Model: Unigram Model

- Only one topic in the entire corpus, or one topic in each document
- Generative process:
- For each document  $d$ :
  - For each word token  $w_n$  in  $d$ :
  - Choose a word  $w$  according to the multinomial distribution  $P(w)$ .

# Simple Unigram – Generative Process

$$P(d) = \prod_{w \in d} P(w)$$

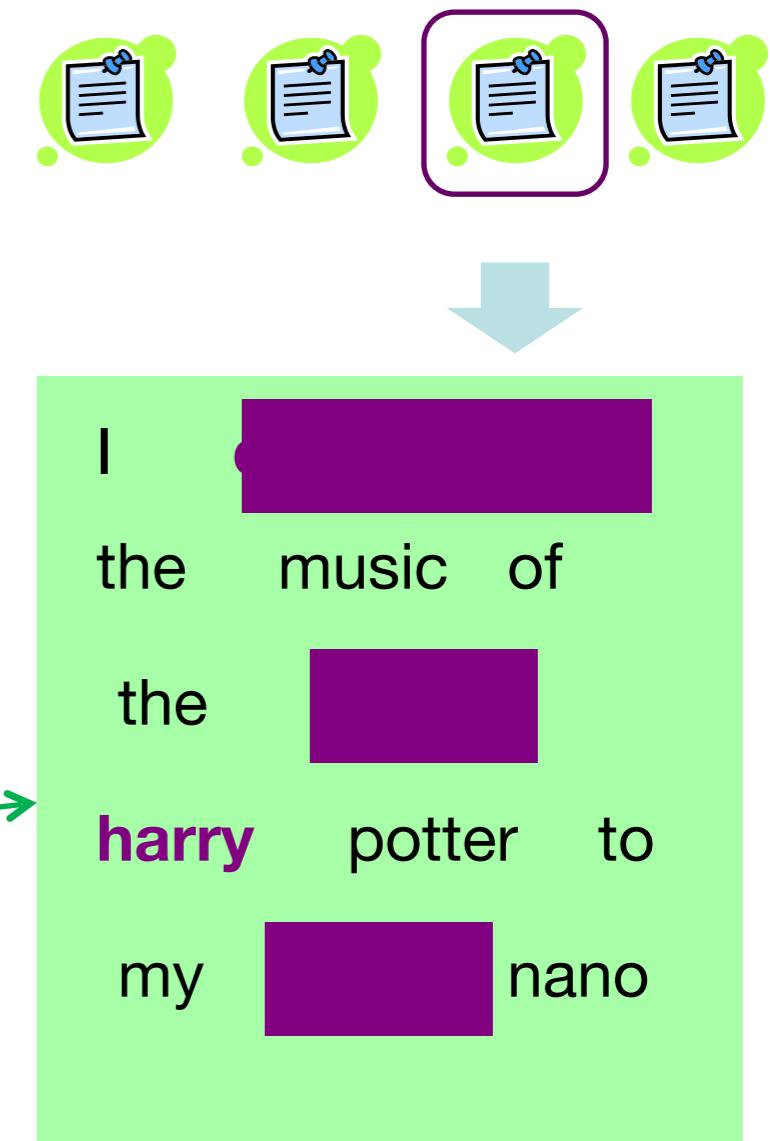
<i>movie</i>	0.100
<i>harry</i>	.090.
<i>potter</i>	05
<i>ipod</i>	0.01
<i>music</i>	0.02



# Simple Unigram – Generative Process

$$P(d) = \prod_{w \in d} P(w)$$

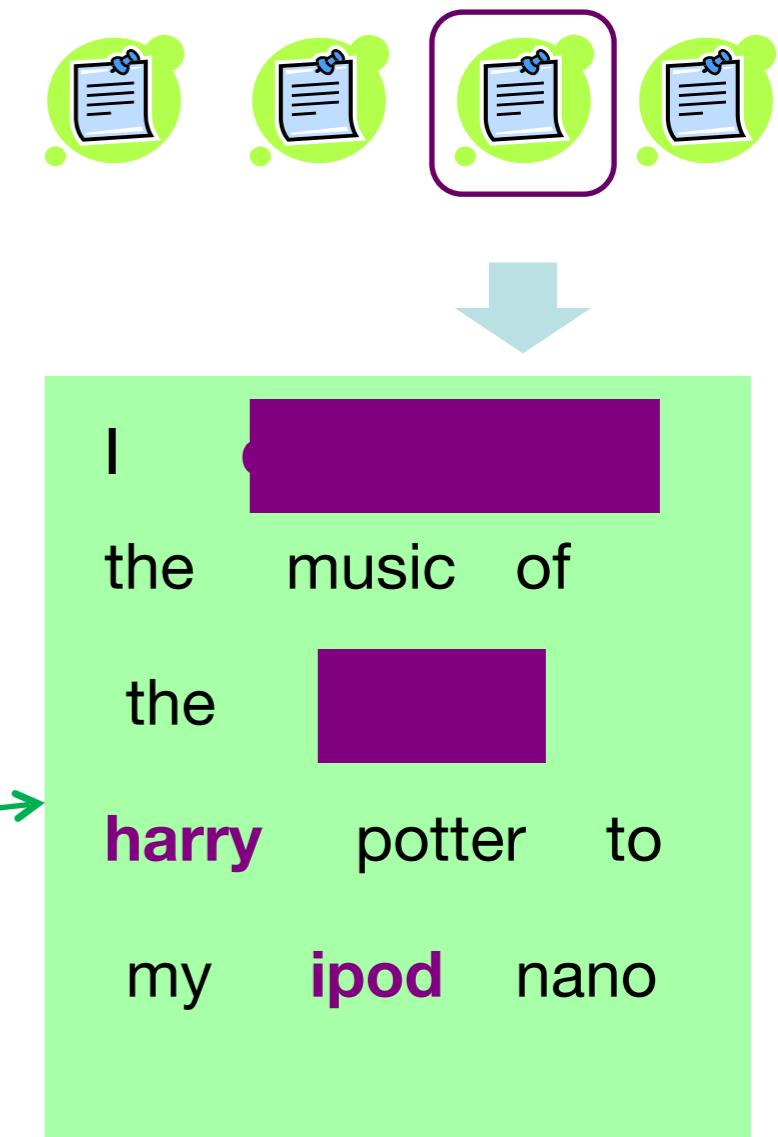
<i>movie</i>	0.100
<i>harry</i>	.090.
<i>potter</i>	05
<i>ipod</i>	0.01
<i>music</i>	0.02



# Simple Unigram – Generative Process

$$P(d) = \prod_{w \in d} P(w)$$

<i>movie</i>	0.100
<i>harry</i>	.090.
<i>potter</i>	05
<i>ipod</i>	0.01
<i>music</i>	0.02



# Simple Unigram – Generative Process

$$P(d) = \prod_{w \in d} P(w)$$

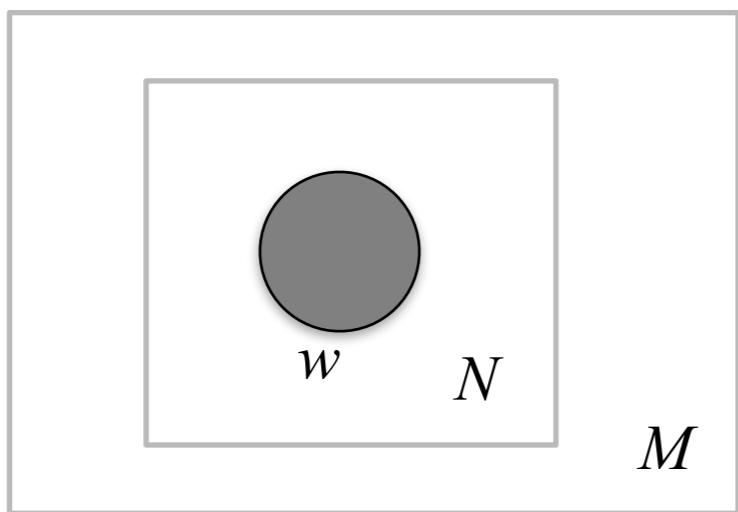
<i>movie</i>	0.100
<i>harry</i>	.090.
<i>potter</i>	05
<i>ipod</i>	0.01
<i>music</i>	0.02



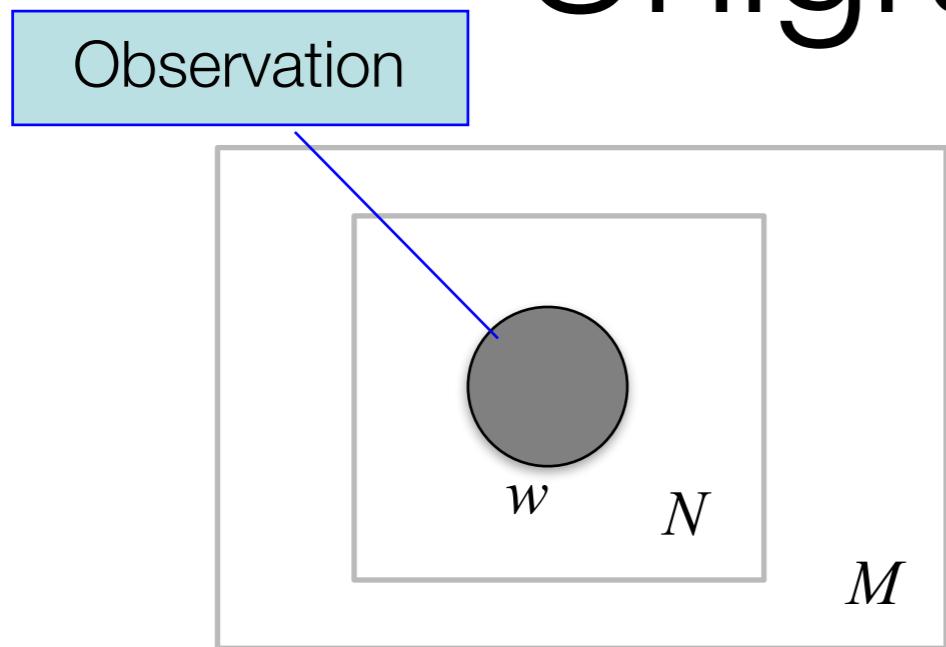
I **downloaded**  
the music of  
the **movie**  
**harry** potter to  
my **ipod** nano

# Graphical Structure of Unigram Models

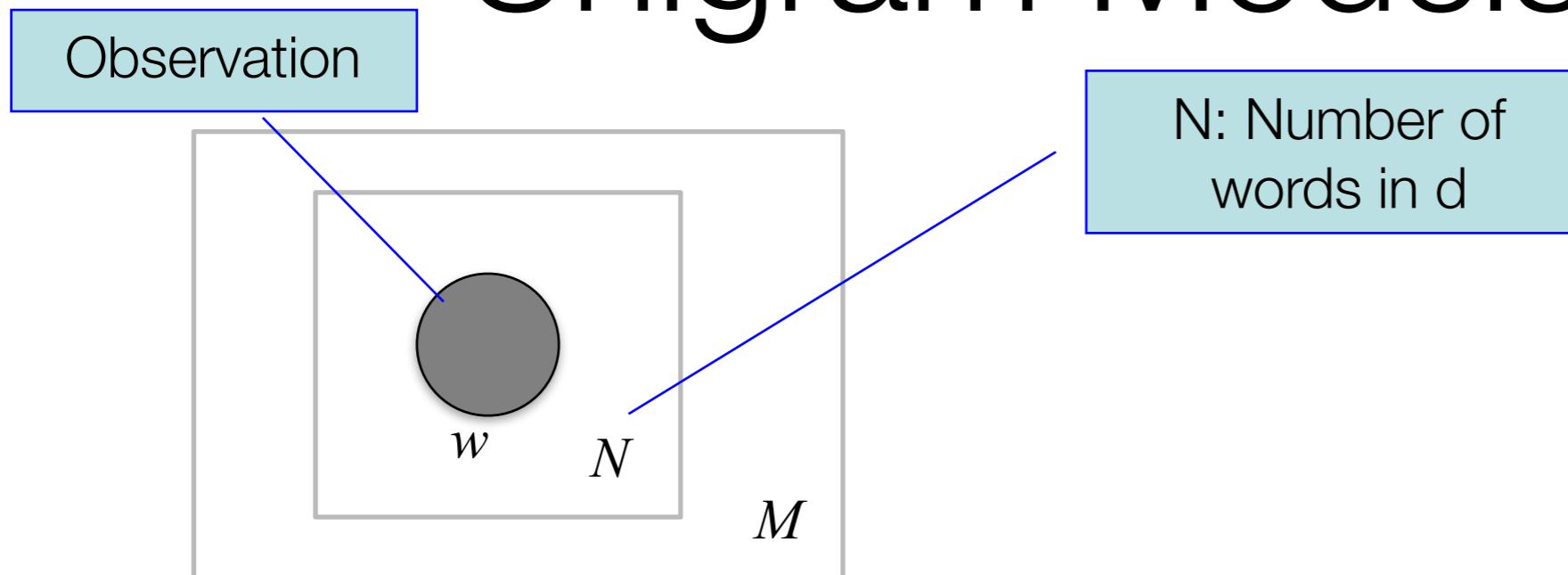
# Graphical Structure of Unigram Models



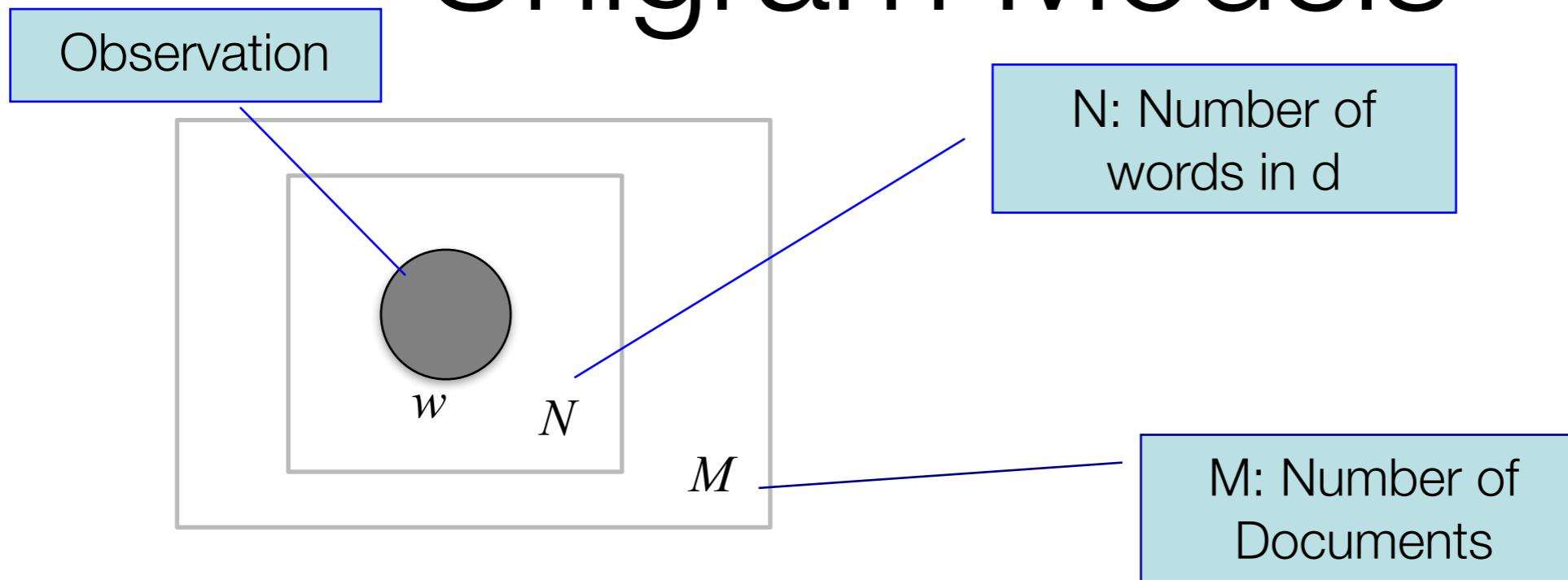
# Graphical Structure of Unigram Models



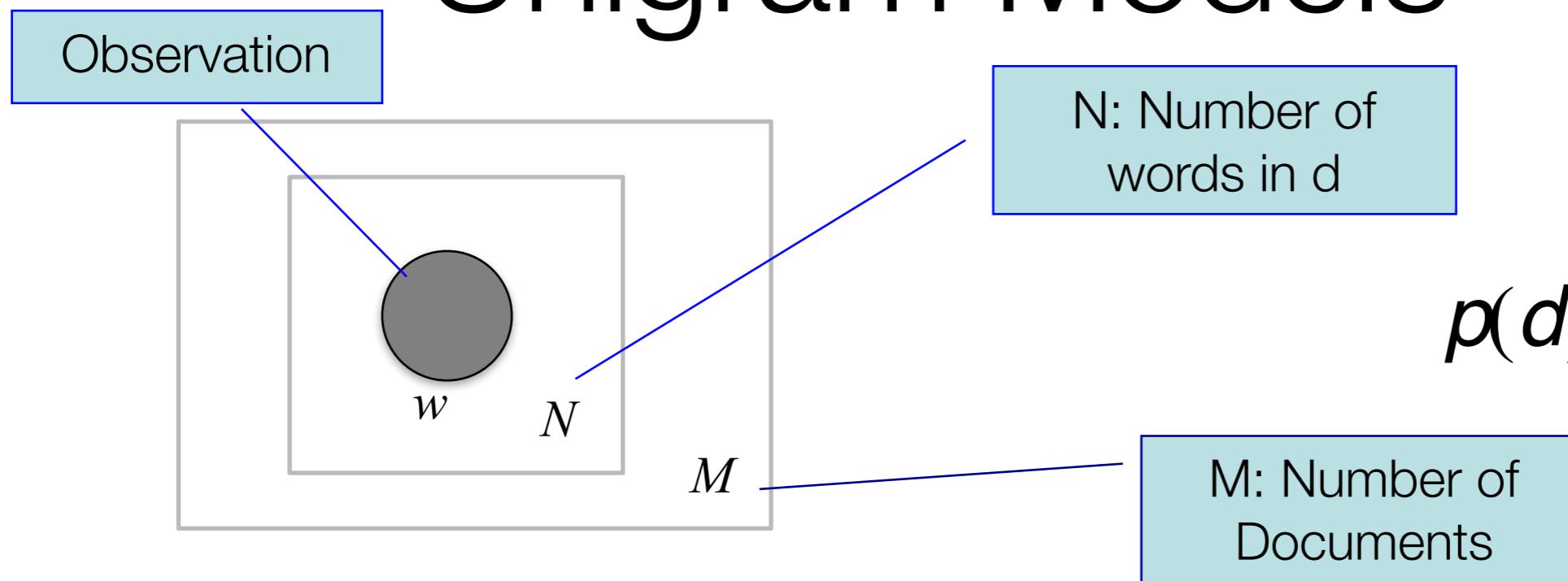
# Graphical Structure of Unigram Models



# Graphical Structure of Unigram Models

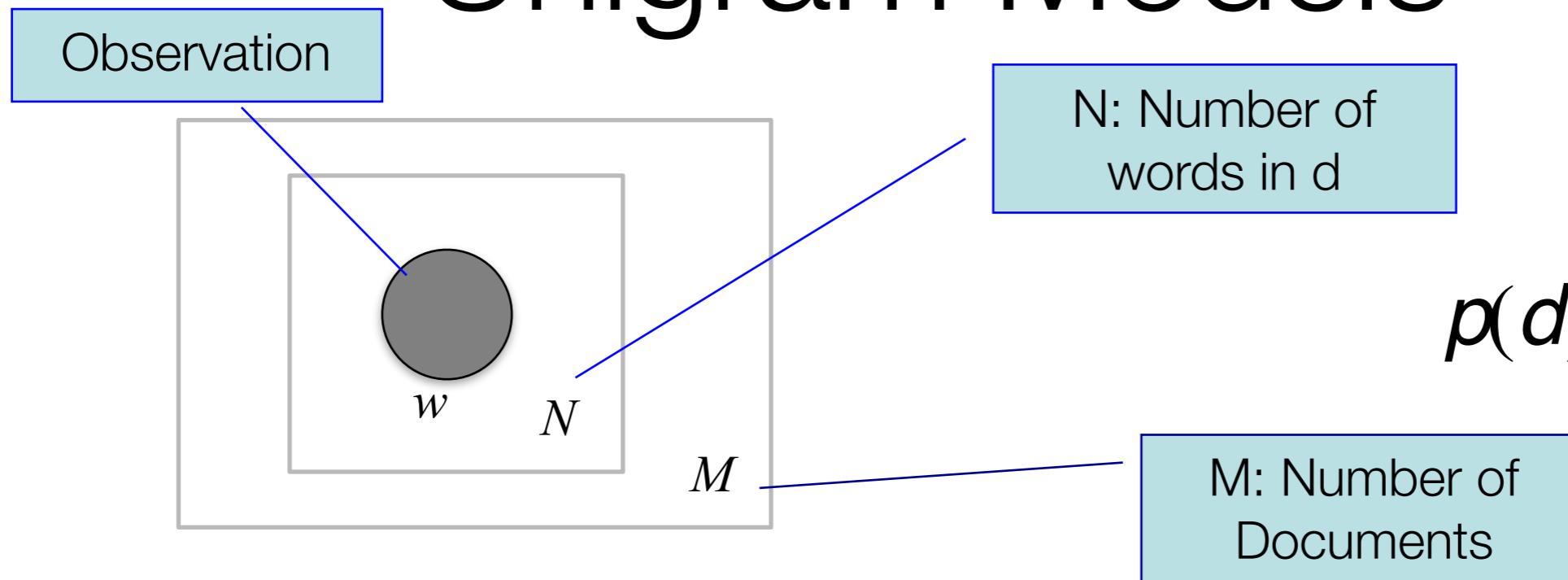


# Graphical Structure of Unigram Models



$$p(d) = \prod_{n=1}^N p(w_n)$$

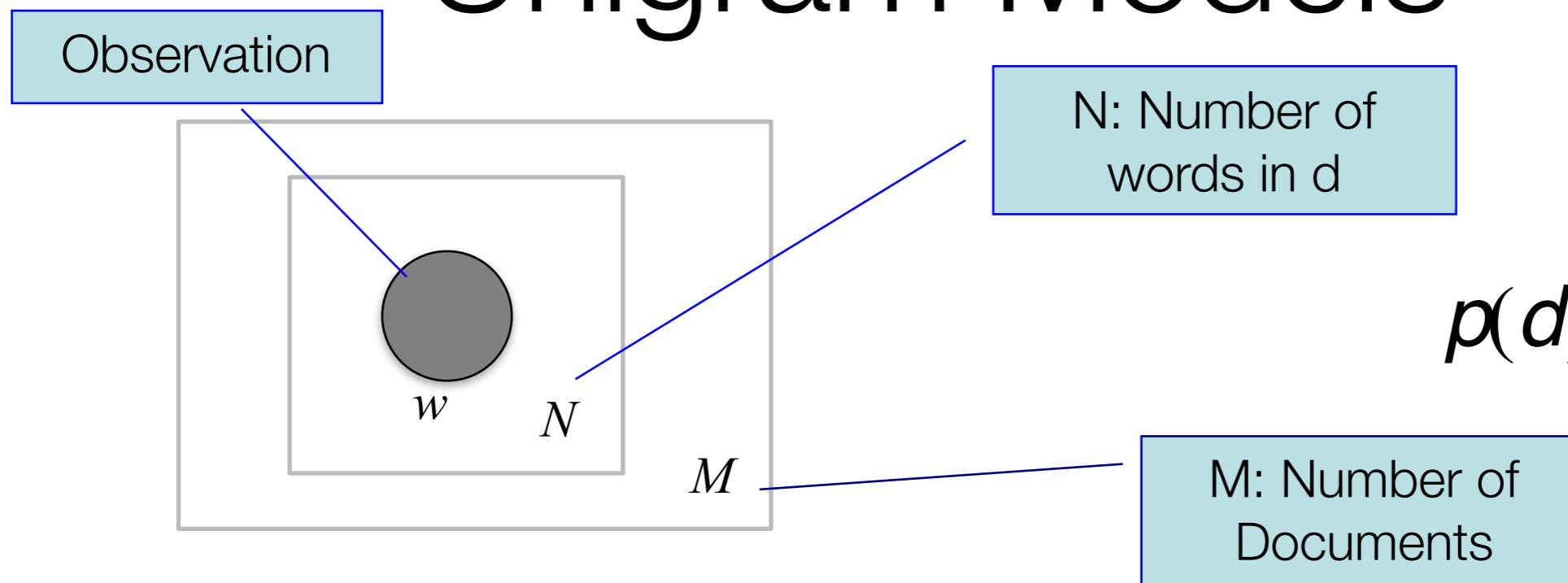
# Graphical Structure of Unigram Models



$$p(d) = \prod_{n=1}^N p(w_n)$$

- No dependency among variables

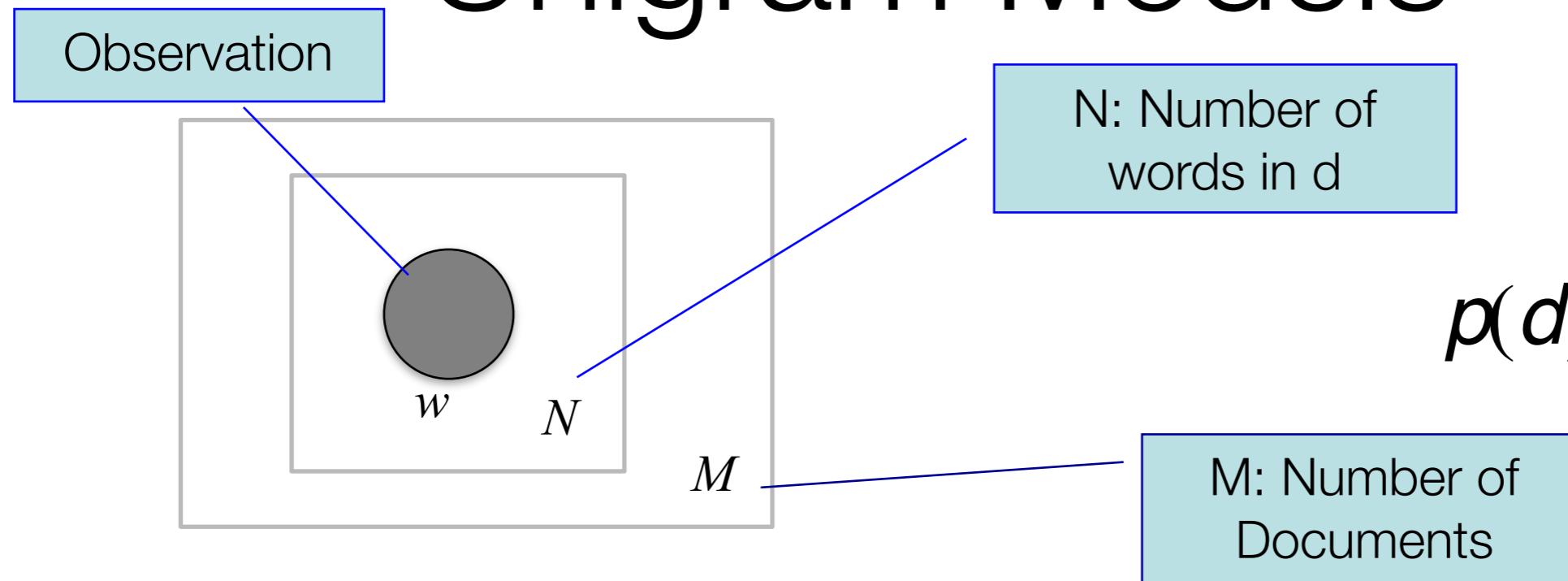
# Graphical Structure of Unigram Models



$$p(d) = \prod_{n=1}^N p(w_n)$$

- No dependency among variables
- Parameter:  $p(w)$

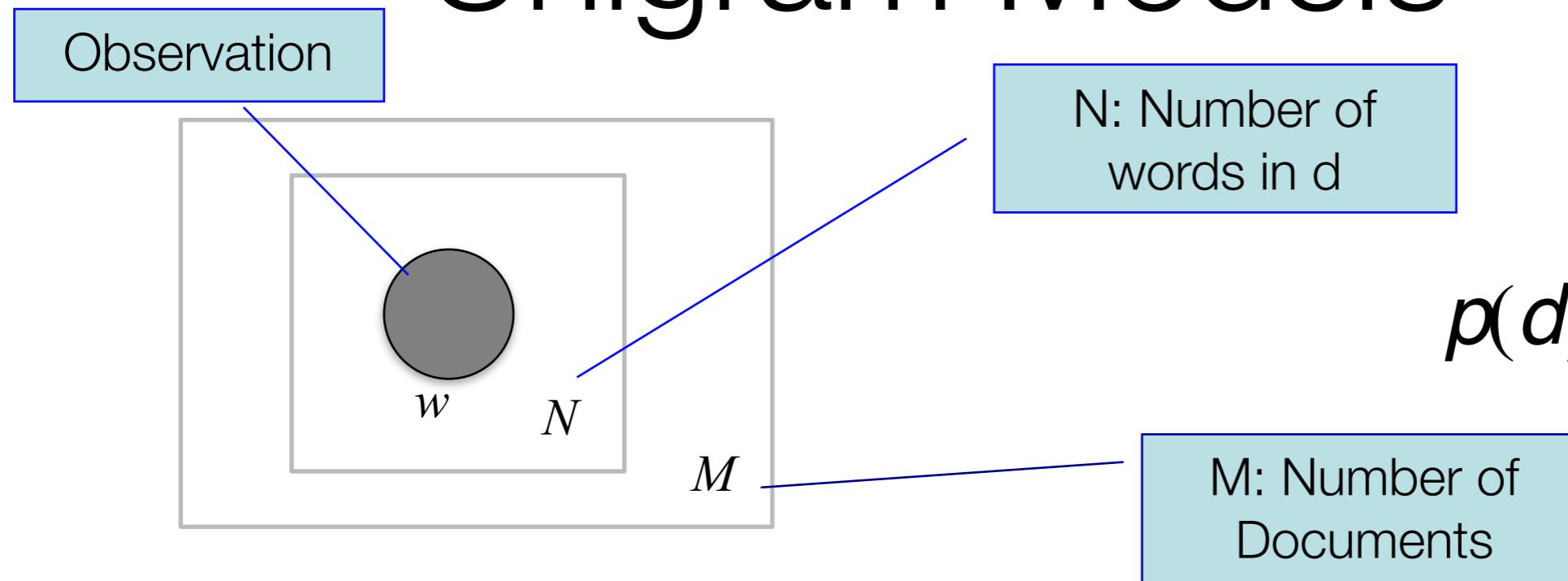
# Graphical Structure of Unigram Models



$$p(d) = \prod_{n=1}^N p(w_n)$$

- No dependency among variables
- Parameter:  $p(w)$
- Parameter estimation:
  - Maximum likelihood estimation (MLE)

# Graphical Structure of Unigram Models



$$p(d) = \prod_{n=1}^N p(w_n)$$

- No dependency among variables
- Parameter:  $p(w)$
- Parameter estimation:
  - Maximum likelihood estimation (MLE)
- Applications:
  - language modeling based information retrieval

# Mixture of Unigrams

# Mixture of Unigrams

- Allow multiple topics in a corpus.

# Mixture of Unigrams

- Allow multiple topics in a corpus.
- The generative model of text becomes a mixture model

# Mixture of Unigrams

- Allow multiple topics in a corpus.
- The generative model of text becomes a mixture model
- Generative process:
  - For every document  $d$ 
    - Choose a topic  $z$  according to a distribution  $P(z)$
    - For every word token in  $d$ 
      - Choose a word  $w_n$  according to word distribution  $P(w|z)$

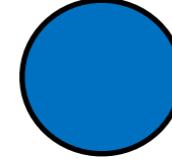
# Mixture of Unigrams – Generative Process

$$P(d) = \sum_{i=1..K} P(z=i) \prod_{w \in d} P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01

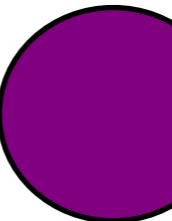
movie	0.100
harry	.090.
potter	05
iPod	0.01
music	0.02

Topic 1



Apple iPod

Topic 2



Harry Potter



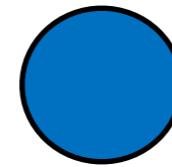
I [REDACTED]  
the music of  
the [REDACTED]  
[REDACTED] potter to  
my [REDACTED] nano

# Mixture of Unigrams – Generative Process

$$P(d) = \sum_{i=1..K} P(z=i) \prod_{w \in d} P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01

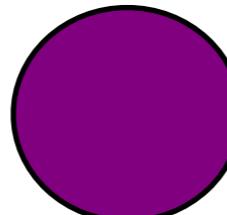
Topic 1



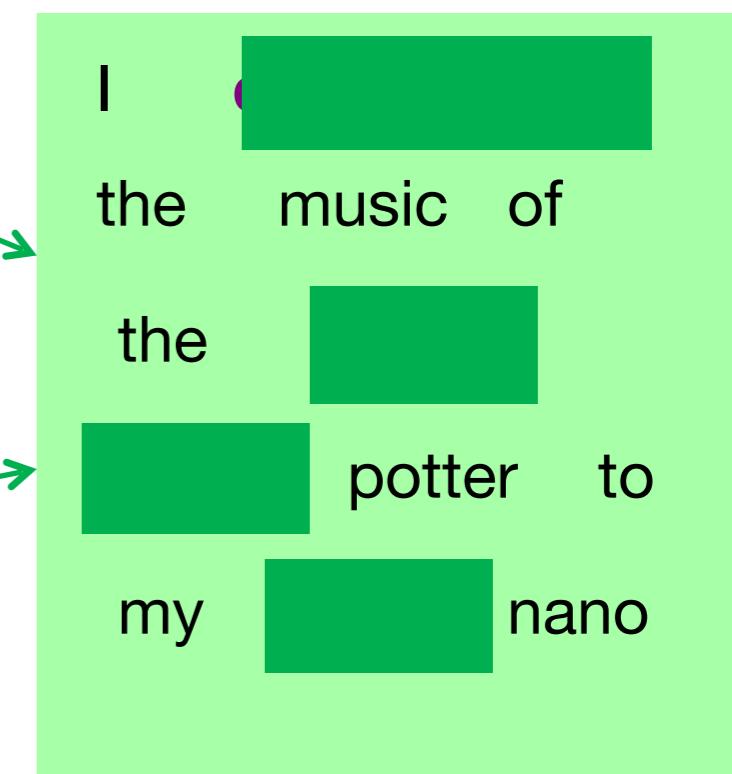
Apple iPod

movie	0.100
harry	.090.
potter	05
iPod	0.01
music	0.02

Topic 2



Harry Potter

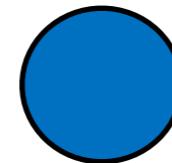


# Mixture of Unigrams – Generative Process

$$P(d) = \sum_{i=1..K} P(z=i) \prod_{w \in d} P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01

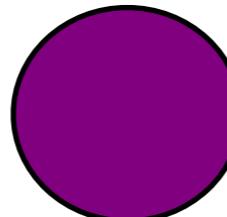
Topic 1



Apple iPod

movie	0.100
harry	.090.
potter	05
ipod	0.01
music	0.02

Topic 2



Harry Potter



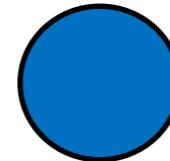
I { music of  
the music of  
the potter to  
my nano

# Mixture of Unigrams – Generative Process

$$P(d) = \sum_{i=1..K} P(z=i) \prod_{w \in d} P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01

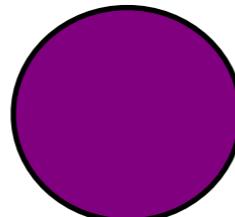
Topic 1



Apple iPod

movie	0.100
harry	.090.
potter	05
ipod	0.01
music	0.02

Topic 2



Harry Potter



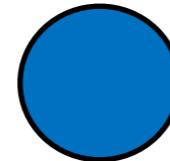
I { music of  
the music of  
the potter to  
my nano

# Mixture of Unigrams – Generative Process

$$P(d) = \sum_{i=1..K} P(z=i) \prod_{w \in d} P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01

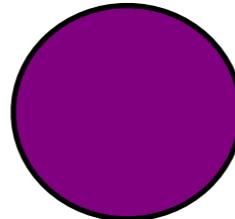
Topic 1



Apple iPod

movie	0.100
harry	.090.
potter	05
ipod	0.01
music	0.02

Topic 2



Harry Potter



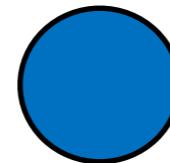
I { music of  
the harry potter to  
my nano

# Mixture of Unigrams – Generative Process

$$P(d) = \sum_{i=1..K} P(z=i) \prod_{w \in d} P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01

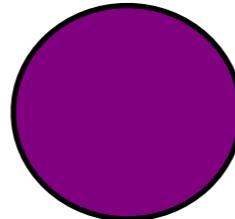
Topic 1



Apple iPod

movie	0.100
harry	.090.
potter	05
ipod	0.01
music	0.02

Topic 2



Harry Potter



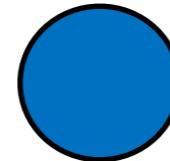
I { music of  
the harry potter to  
my ipod nano

# Mixture of Unigrams – Generative Process

$$P(d) = \sum_{i=1..K} P(z=i) \prod_{w \in d} P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01

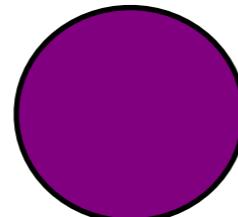
Topic 1



Apple iPod

movie	0.100
harry	.090.
potter	05
iPod	0.01
music	0.02

Topic 2



Harry Potter



I **downloaded**  
the music of  
the **movie**  
**harry** potter to  
my **iPod** nano

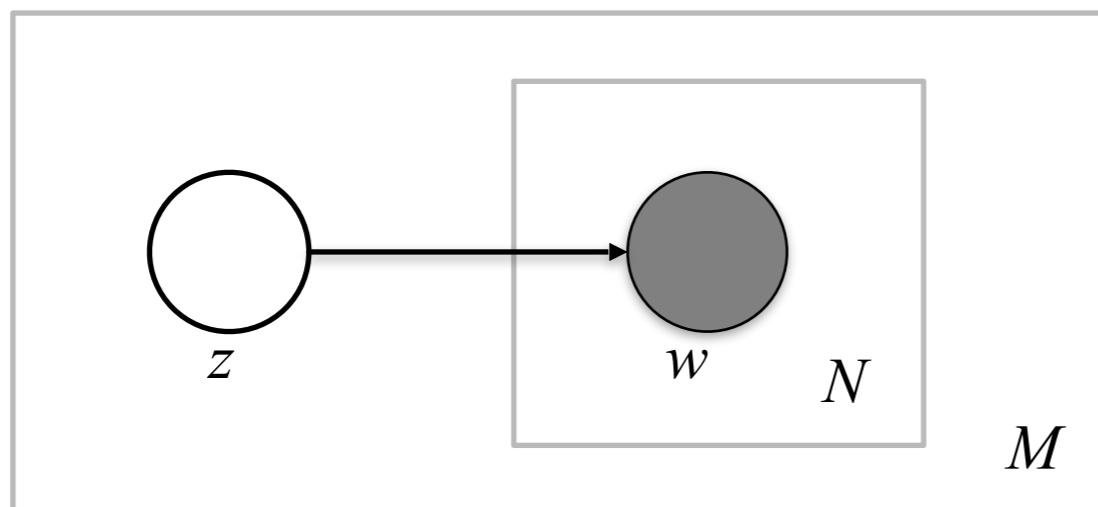
# Mixture of Unigrams – Graphical Model

# Mixture of Unigrams – Graphical Model

- There are  $k$  topics in the collection, but each document only cover one topic

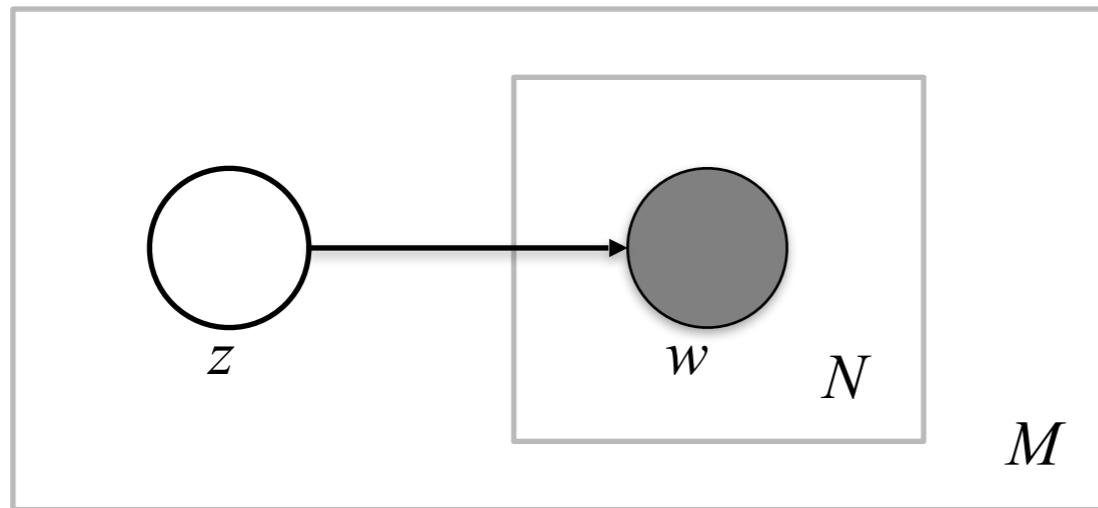
# Mixture of Unigrams – Graphical Model

- There are  $k$  topics in the collection, but each document only cover one topic



# Mixture of Unigrams – Graphical Model

- There are  $k$  topics in the collection, but each document only cover one topic



$$p(d) = \sum_z p(z) \prod_{n=1}^N p(w_n | z)$$

- Estimation: MLE and EM Algorithm
- Application: simple document clustering

# Mixture of Unigram: Pros and Cons

# Mixture of Unigram: Pros and Cons

- Allow each document to select among multiple identities (topics)

# Mixture of Unigram: Pros and Cons

- Allow each document to select among multiple identities (topics)
- But once decided, all words in the document share the identity of the document.

# Mixture of Unigram: Pros and Cons

- Allow each document to select among multiple identities (topics)
- But once decided, all words in the document share the identity of the document.
- Essentially a probabilistic version of K-means.

# Mixture of Unigram: Pros and Cons

- Allow each document to select among multiple identities (topics)
- But once decided, all words in the document share the identity of the document.
- Essentially a probabilistic version of K-means.
- Simple and easy to estimate

# Mixture of Unigram: Pros and Cons

- Allow each document to select among multiple identities (topics)
- But once decided, all words in the document share the identity of the document.
- Essentially a probabilistic version of K-means.
- Simple and easy to estimate
- Performs well on short documents, e.g., tweets

# Mixture of Unigram: Pros and Cons

- Allow each document to select among multiple identities (topics)
- But once decided, all words in the document share the identity of the document.
- Essentially a probabilistic version of K-means.
- Simple and easy to estimate
- Performs well on short documents, e.g., tweets
- Cons:
  - Usually different parts of a document present different topics
  - Desirable: words in a document to take different identities

# Probabilistic Latent Semantic Analysis

(Hofmann, 1999)

- Also known as probabilistic latent semantic indexing (PLSI)
- Generative process:

# Probabilistic Latent Semantic Analysis

(Hofmann, 1999)

- Also known as probabilistic latent semantic indexing (PLSI)
- Generative process:
  - For every document  $d$

# Probabilistic Latent Semantic Analysis

(Hofmann, 1999)

- Also known as probabilistic latent semantic indexing (PLSI)
- Generative process:
  - For every document  $d$
  - For every word token in  $d$

# Probabilistic Latent Semantic Analysis

(Hofmann, 1999)

- Also known as probabilistic latent semantic indexing (PLSI)
- Generative process:
  - For every document  $d$
  - For every word token in  $d$
  - Choose a topic  $z_n$  according to  $P(z|d)$

# Probabilistic Latent Semantic Analysis

(Hofmann, 1999)

- Also known as probabilistic latent semantic indexing (PLSI)
- Generative process:
  - For every document  $d$ 
    - For every word token in  $d$ 
      - Choose a topic  $z_n$  according to  $P(z|d)$
      - Choose a word  $w_n$  according to  $P(w|z_n)$

# Probabilistic Latent Semantic Analysis

(Hofmann, 1999)

- Also known as probabilistic latent semantic indexing (PLSI)
- Generative process:
  - For every document  $d$ 
    - For every word token in  $d$ 
      - Choose a topic  $z_n$  according to  $P(z|d)$
      - Choose a word  $w_n$  according to  $P(w|z_n)$
    - Both  $P(z|d)$  and  $P(w|z)$  are fixed but unknown parameters – to be estimated

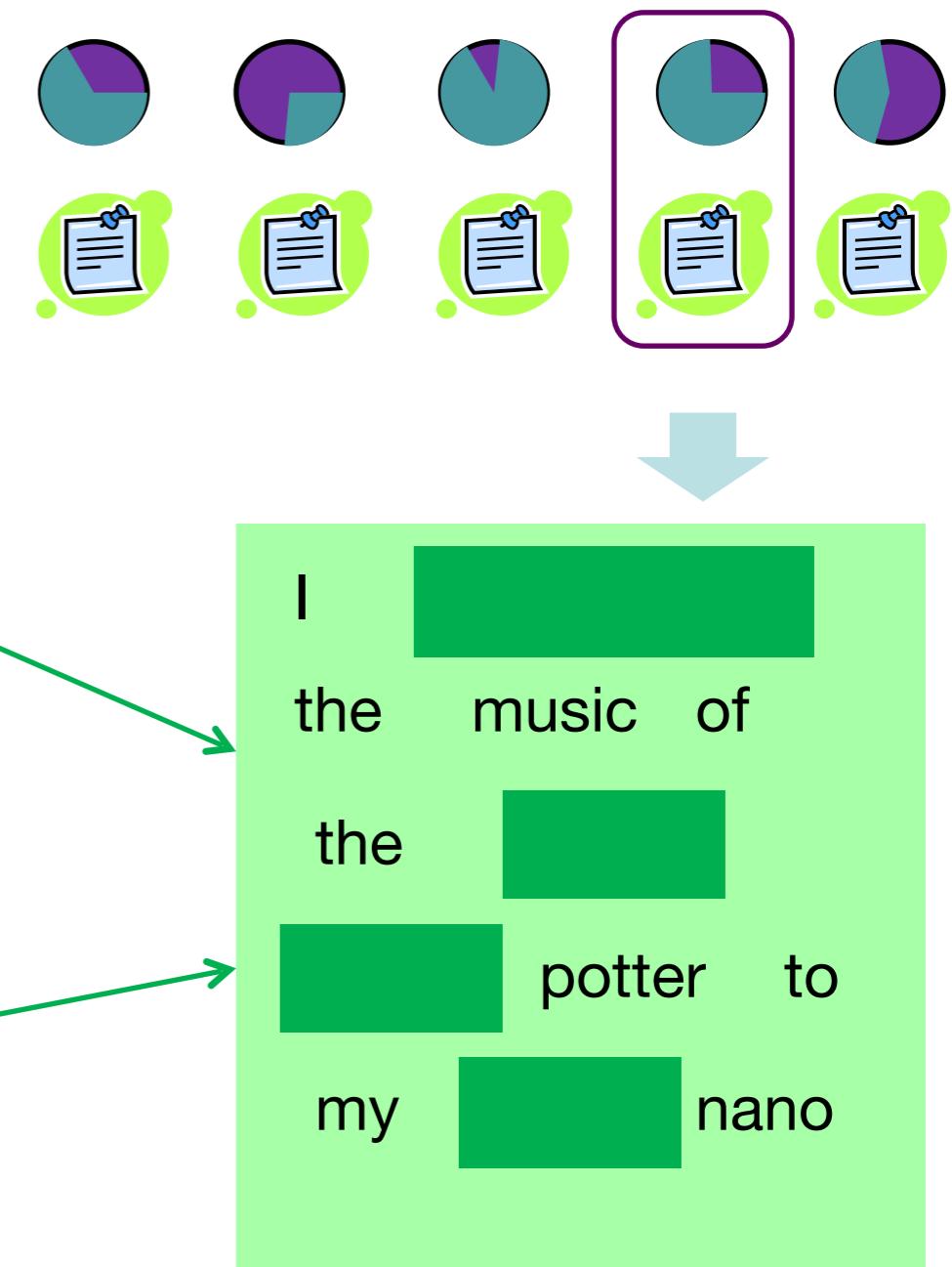
# PLSA – Generative Process

$$P(d) = \prod_{w \in d} \sum_{i=1..K} P(z=i | d) P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01



<i>movie</i>	0.100
<i>harry</i>	.090.
<i>potter</i>	05
<i>iPod</i>	0.01
<i>music</i>	0.02

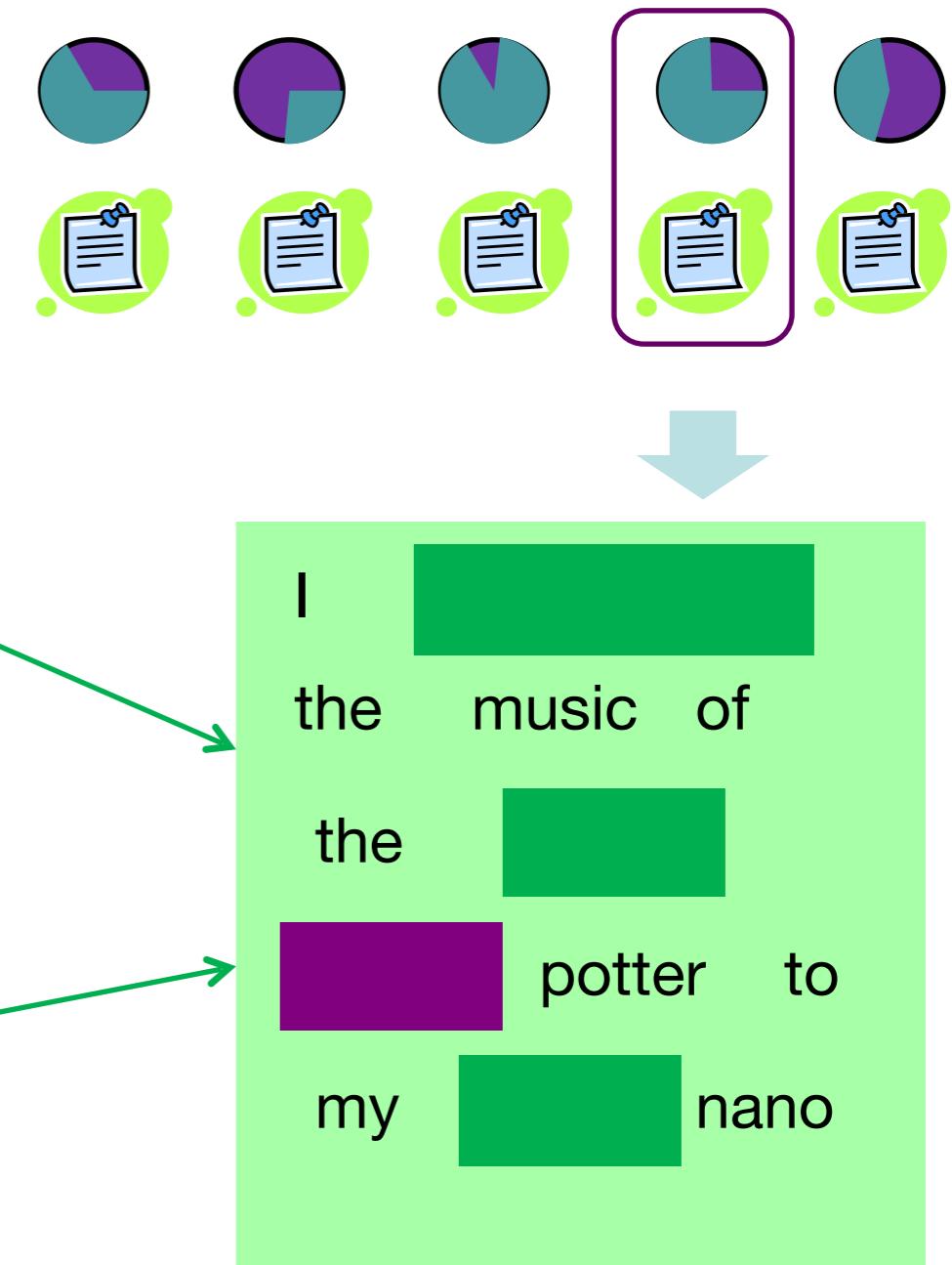


# PLSA – Generative Process

$$P(d) = \prod_{w \in d} \sum_{i=1..K} P(z=i | d) P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01

movie	0.100
harry	.090.
potter	05
iPod	0.01
music	0.02



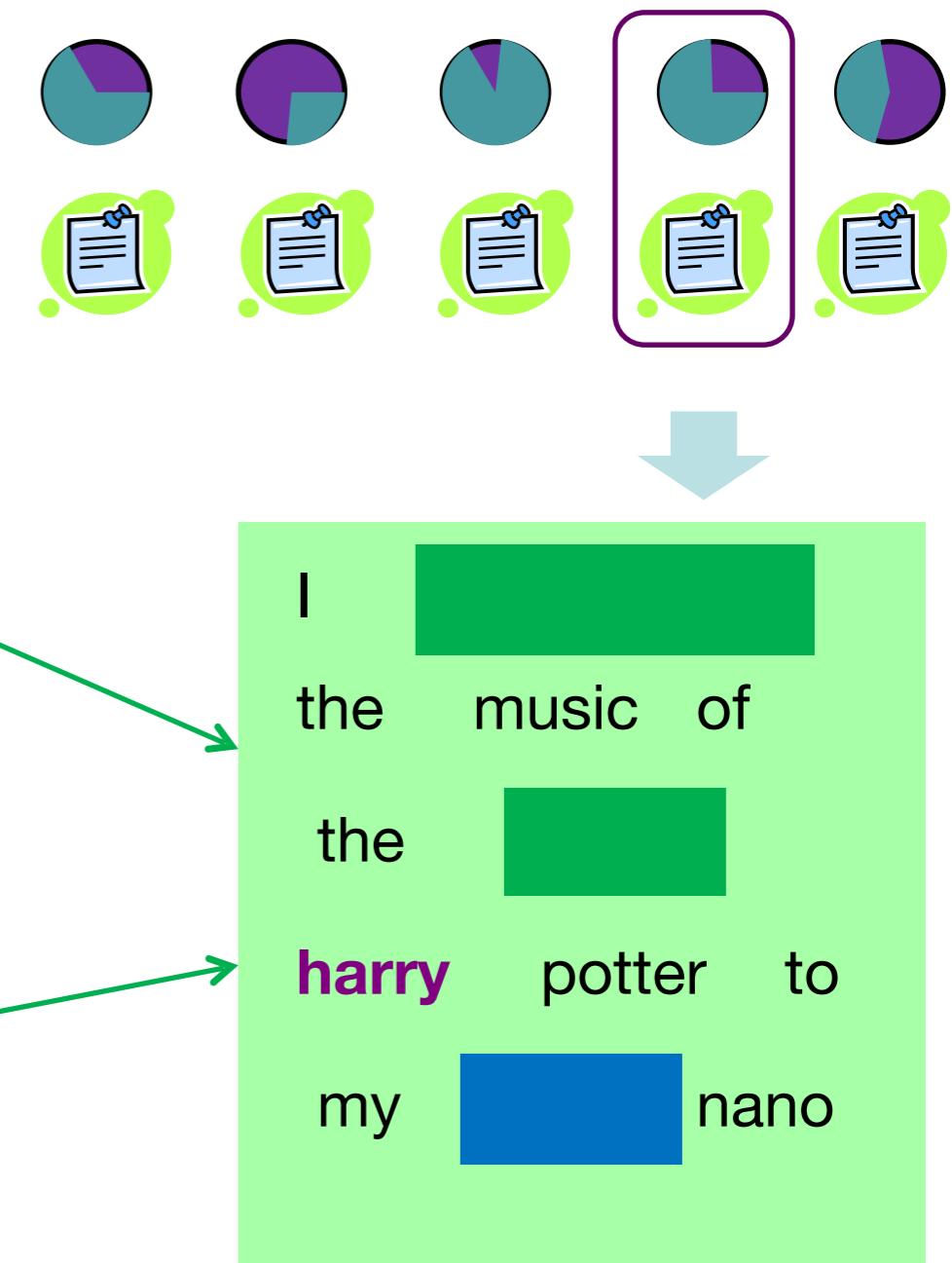
# PLSA – Generative Process

$$P(d) = \prod_{w \in d} \sum_{i=1..K} P(z=i | d) P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01



movie	0.100
harry	.090.
potter	05
ipod	0.01
music	0.02



# PLSA – Generative Process

$$P(d) = \prod_{w \in d} \sum_{i=1..K} P(z=i | d) P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01



movie	0.100
harry	.090.
potter	05
ipod	0.01
music	0.02



I [REDACTED]  
the music of  
the [REDACTED]  
harry potter to  
my ipod nano

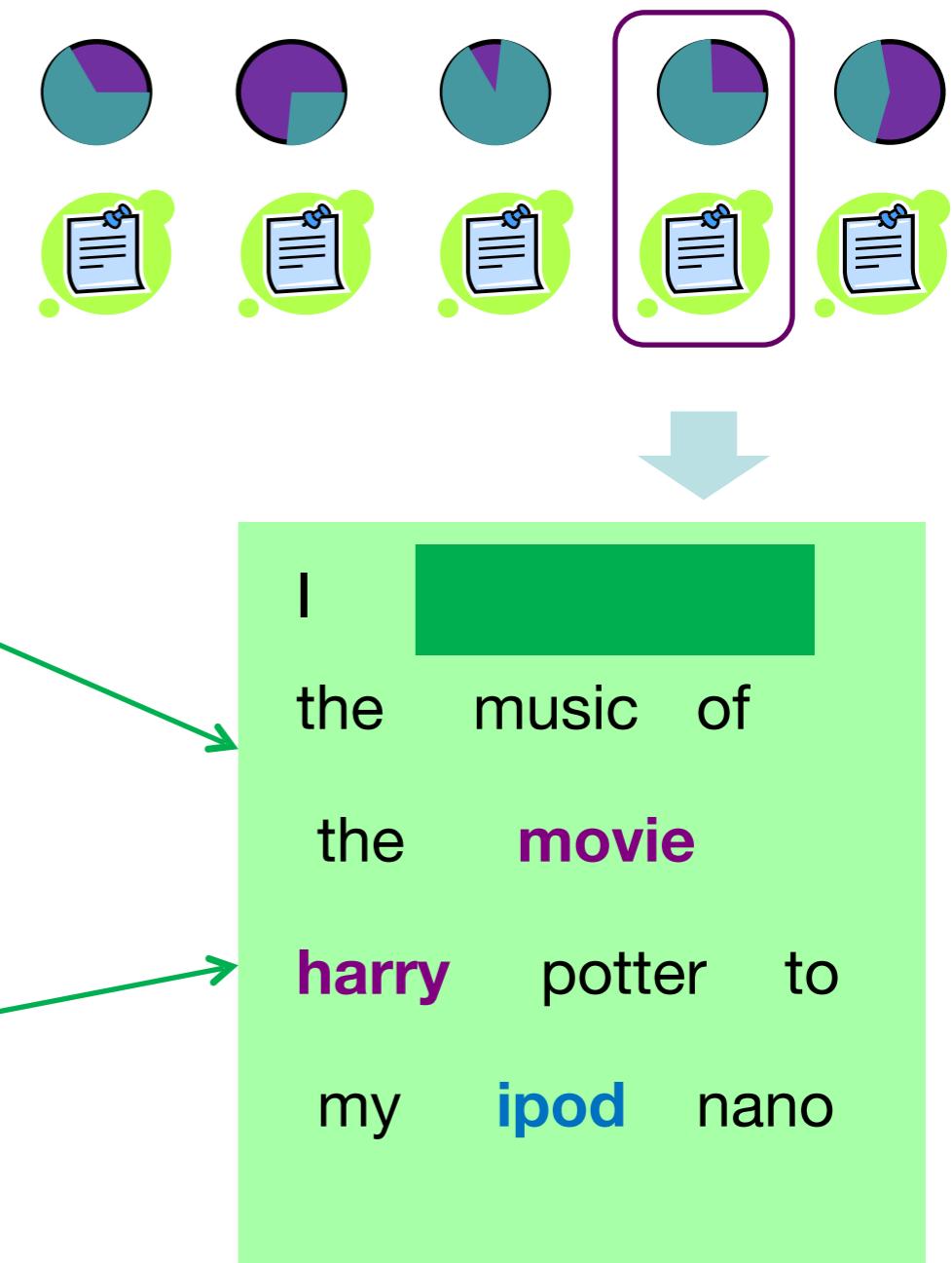
# PLSA – Generative Process

$$P(d) = \prod_{w \in d} \sum_{i=1..K} P(z=i | d) P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01



movie	0.100
harry	.090.
potter	05
ipod	0.01
music	0.02



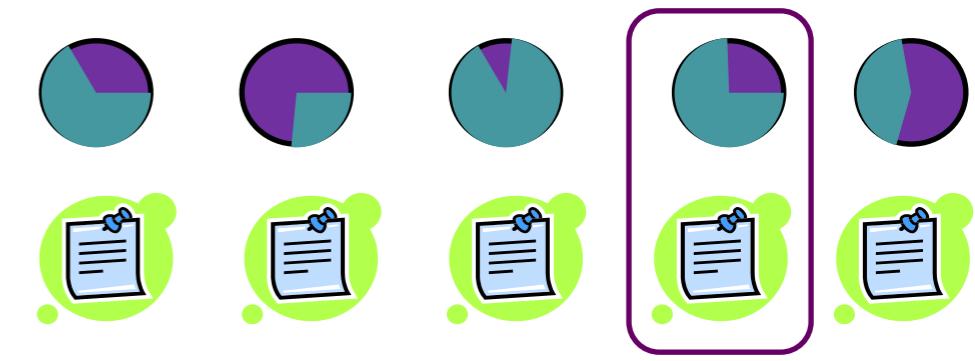
# PLSA – Generative Process

$$P(d) = \prod_{w \in d} \sum_{i=1..K} P(z=i | d) P(w | Topic_i)$$

ipod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01



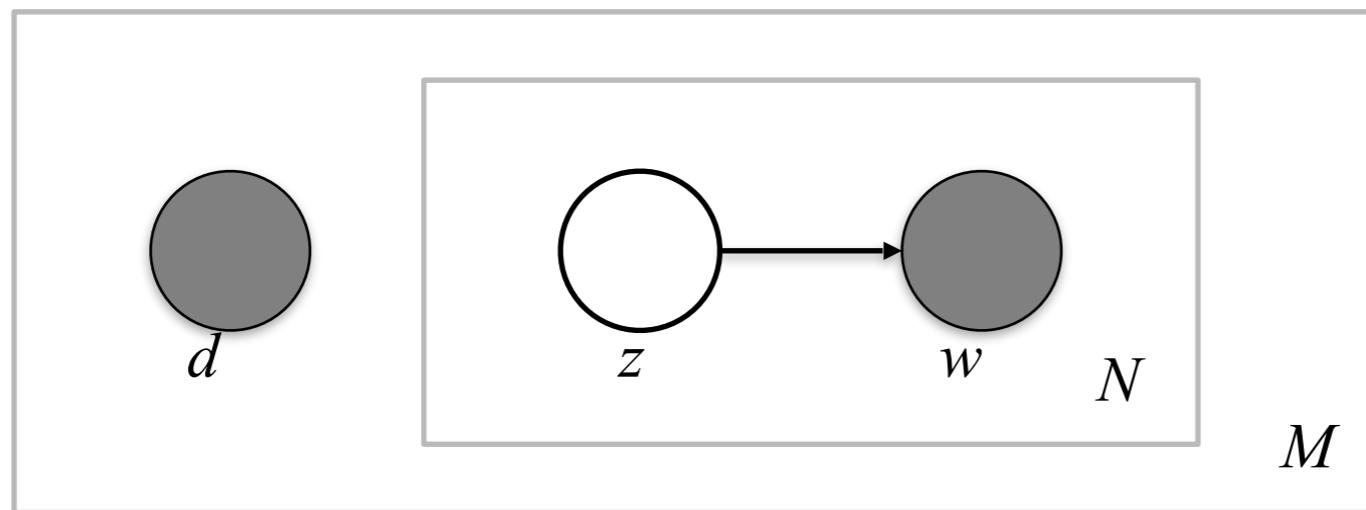
movie	0.100
harry	.090.
potter	05
ipod	0.01
music	0.02



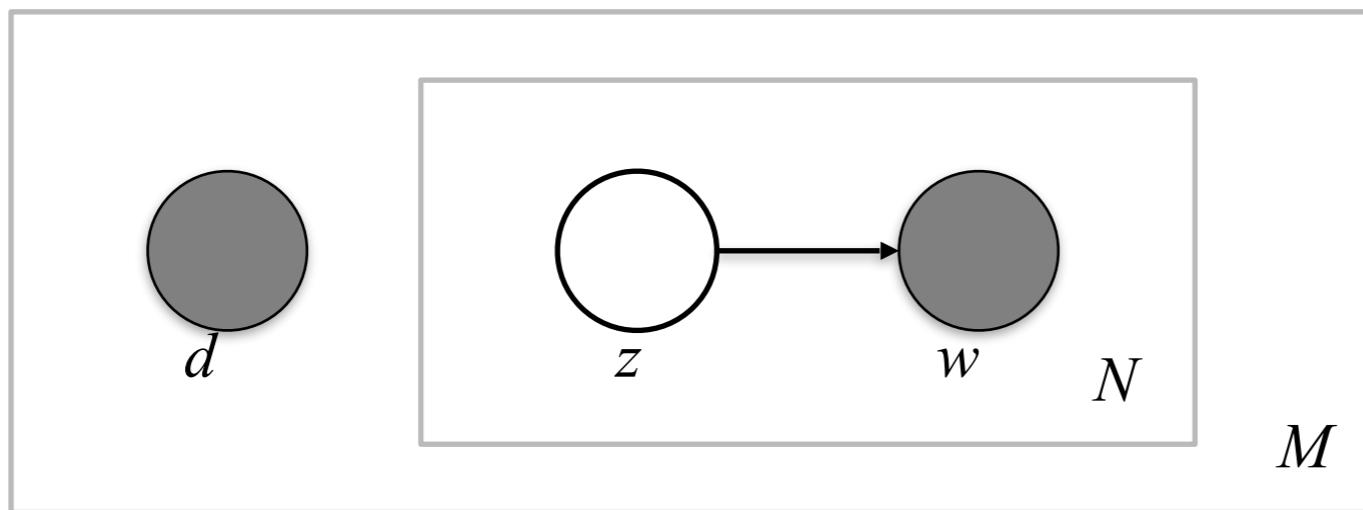
I **downloaded**  
the music of  
the **movie**  
**harry** potter to  
my **ipod** nano

# Graphical Model of PLSA

# Graphical Model of PLSA

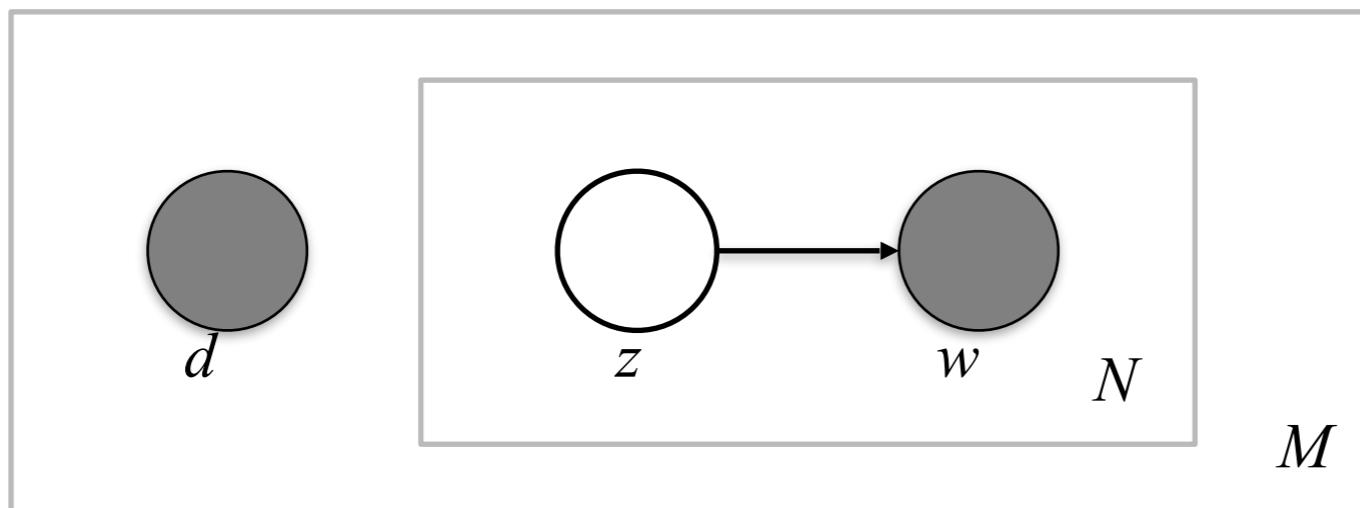


# Graphical Model of PLSA



$$p(w_n, d) = p(d) \sum_z p(w_n | z) p(z | d)$$

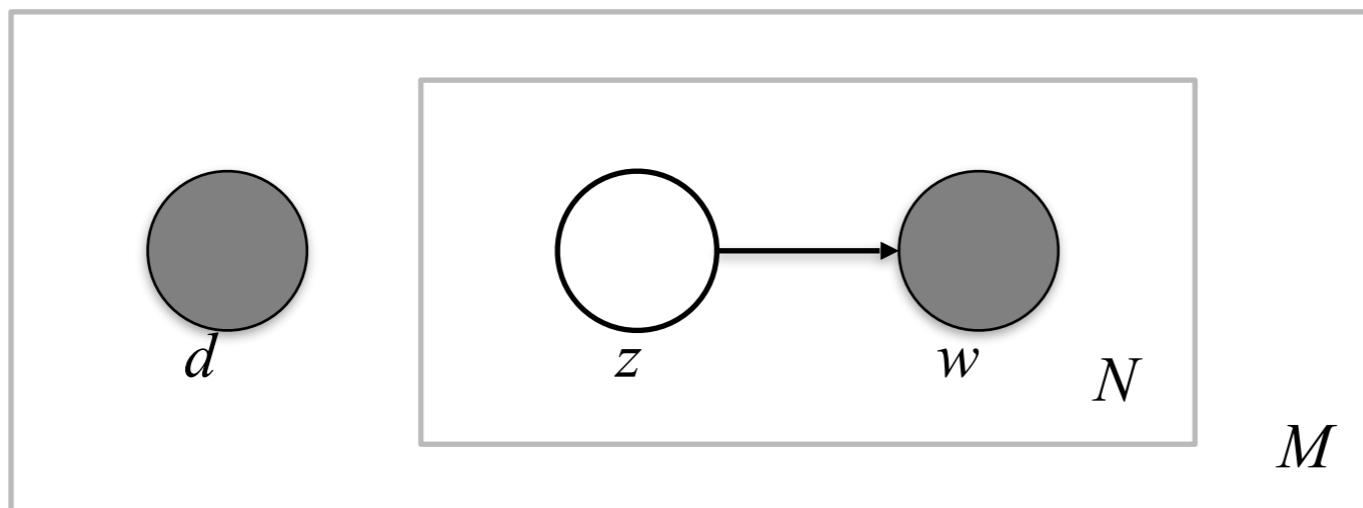
# Graphical Model of PLSA



$$p(w_n, d) = p(d) \sum_z p(w_n | z) p(z | d)$$

- Assume uniform distribution of  $p(d)$

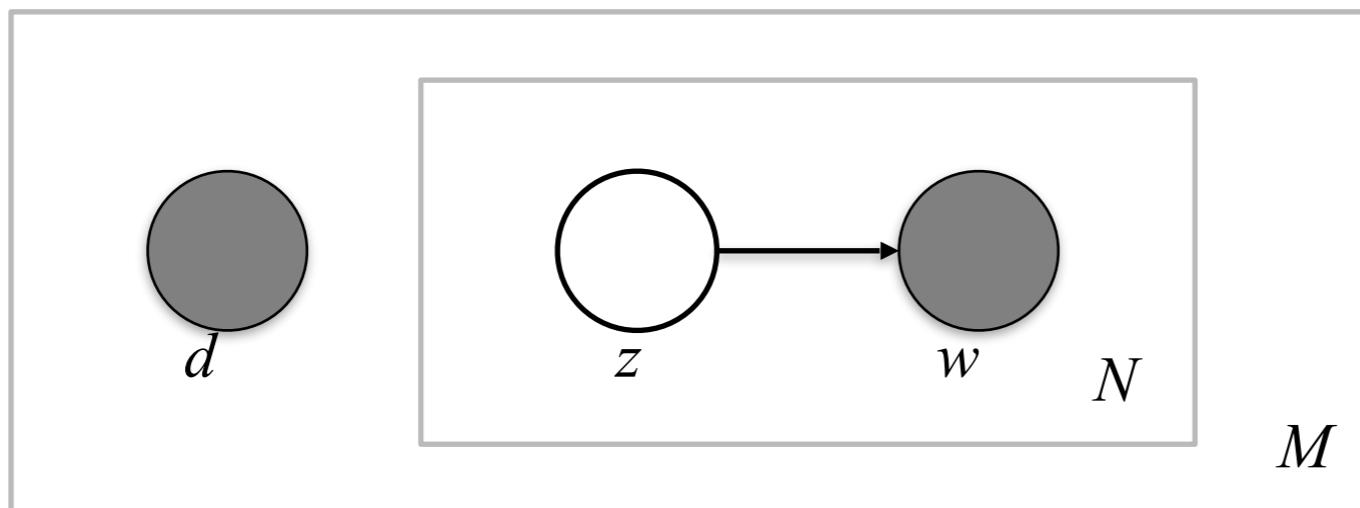
# Graphical Model of PLSA



$$p(w_n, d) = p(d) \sum_z p(w_n | z) p(z | d)$$

- Assume uniform distribution of  $p(d)$
- Parameters:  $P(w|z)$ ,  $P(z|d)$

# Graphical Model of PLSA



$$p(w_n, d) = p(d) \sum_z p(w_n | z) p(z | d)$$

- Assume uniform distribution of  $p(d)$
- Parameters:  $P(w|z)$ ,  $P(z|d)$
- Parameter estimation: MLE, Expectation-Maximization

# PLSA: Pros and Cons

- One of the most effective topic models

# PLSA: Pros and Cons

- One of the most effective topic models
- Allow a document to present multiple identities (topics)

# PLSA: Pros and Cons

- One of the most effective topic models
- Allow a document to present multiple identities (topics)
- Allow words in the same document to take different identities. Word identities are affected, but not forced to be identical to the document's identity.

# PLSA: Pros and Cons

- One of the most effective topic models
- Allow a document to present multiple identities (topics)
- Allow words in the same document to take different identities. Word identities are affected, but not forced to be identical to the document's identity.
- A “soft” version of K-means

# PLSA: Pros and Cons

- One of the most effective topic models
- Allow a document to present multiple identities (topics)
- Allow words in the same document to take different identities. Word identities are affected, but not forced to be identical to the document's identity.
- A “soft” version of K-means
- Cons:
  - Not a complete Bayesian model: hard to interpret unseen documents
  - Overfits the data (because number of parameters grows linearly with the number of documents).

# Latent Dirichlet Allocation

(Blei&Ng&Jordan, 2003)

# Latent Dirichlet Allocation

(Blei&Ng&Jordan, 2003)

- The Bayesian version of PLSA

# Latent Dirichlet Allocation

(Blei&Ng&Jordan, 2003)

- The Bayesian version of PLSA
- Treats the document-topic mixture weights as a k-parameter hidden random variable (a multinomial).

# Latent Dirichlet Allocation

(Blei&Ng&Jordan, 2003)

- The Bayesian version of PLSA
- Treats the document-topic mixture weights as a k-parameter hidden random variable (a multinomial).
- Places a **Dirichlet prior** on all the multinomial mixing weights. Sample the topic mixture weights once per document.

# Latent Dirichlet Allocation

(Blei&Ng&Jordan, 2003)

- The Bayesian version of PLSA
- Treats the document-topic mixture weights as a k-parameter hidden random variable (a multinomial).
- Places a **Dirichlet prior** on all the multinomial mixing weights. Sample the topic mixture weights once per document.
- The weights for word multinomial distributions are still considered as fixed parameters to be estimated.

# Latent Dirichlet Allocation

(Blei&Ng&Jordan, 2003)

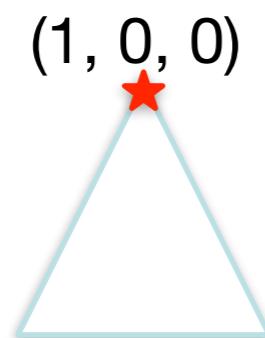
- The Bayesian version of PLSA
- Treats the document-topic mixture weights as a k-parameter hidden random variable (a multinomial).
- Places a **Dirichlet prior** on all the multinomial mixing weights. Sample the topic mixture weights once per document.
- The weights for word multinomial distributions are still considered as fixed parameters to be estimated.
- For a fuller Bayesian approach, can place a Dirichlet prior to these word multinomial distributions to smooth the probabilities.

# Multinomial Distributions

- Distribution over discrete outcomes
- Represented by a non-negative vector that sums to one
- Picture repression of multinomial distributions over three discrete outcomes

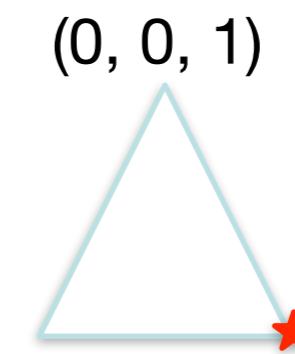
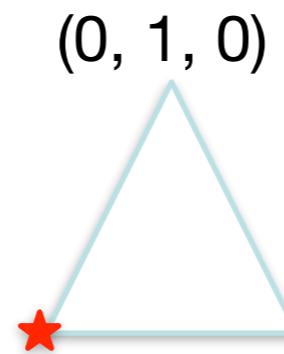
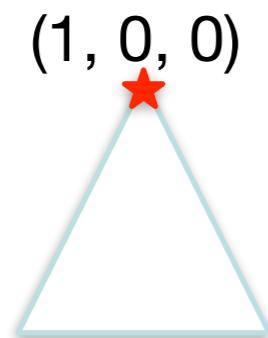
# Multinomial Distributions

- Distribution over discrete outcomes
- Represented by a non-negative vector that sums to one
- Picture repression of multinomial distributions over three discrete outcomes



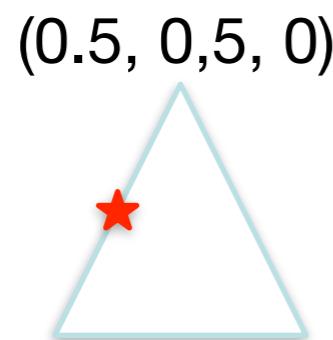
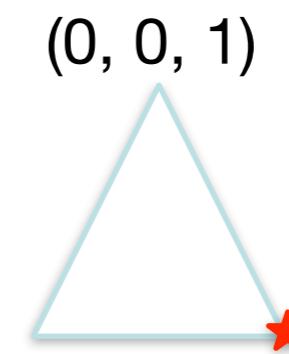
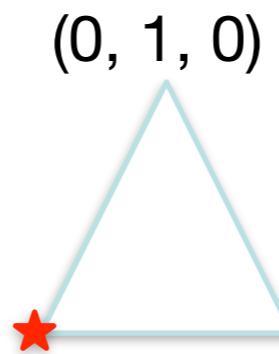
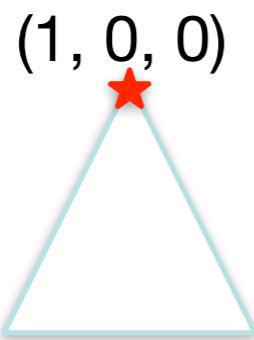
# Multinomial Distributions

- Distribution over discrete outcomes
- Represented by a non-negative vector that sums to one
- Picture repression of multinomial distributions over three discrete outcomes



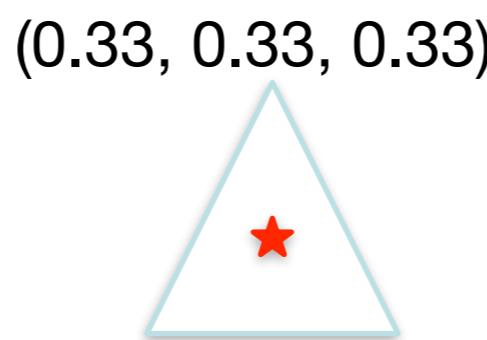
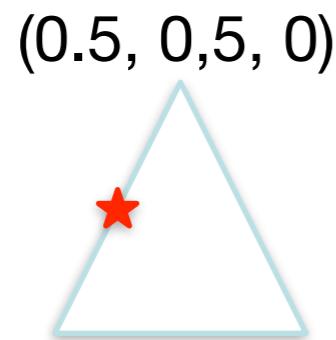
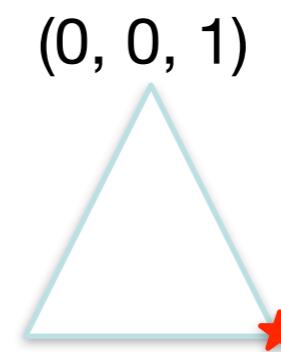
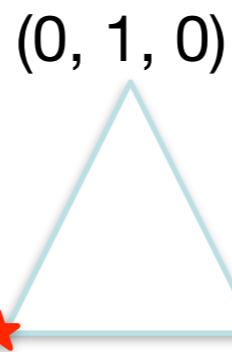
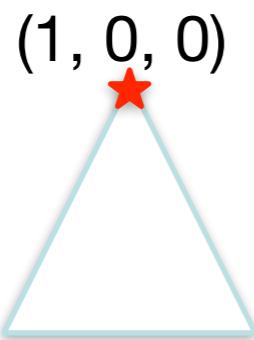
# Multinomial Distributions

- Distribution over discrete outcomes
- Represented by a non-negative vector that sums to one
- Picture repression of multinomial distributions over three discrete outcomes



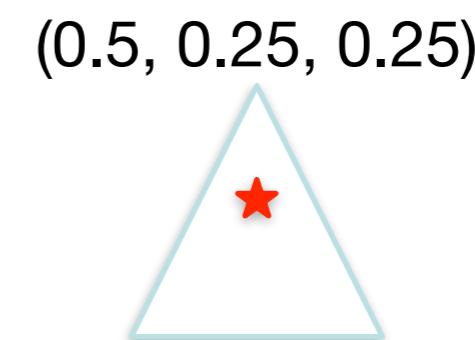
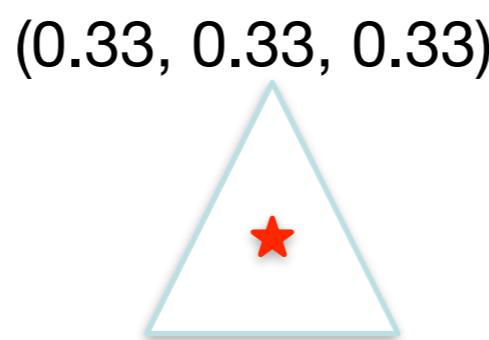
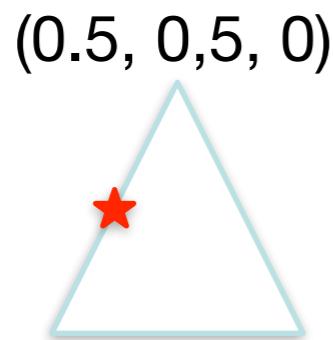
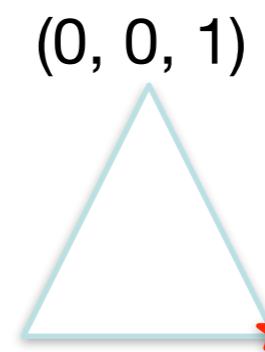
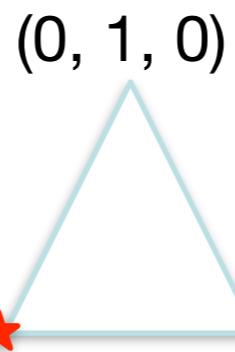
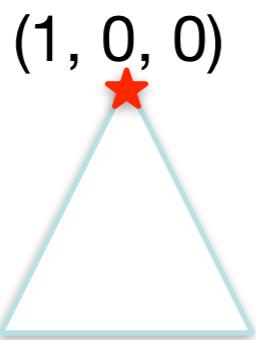
# Multinomial Distributions

- Distribution over discrete outcomes
- Represented by a non-negative vector that sums to one
- Picture repression of multinomial distributions over three discrete outcomes



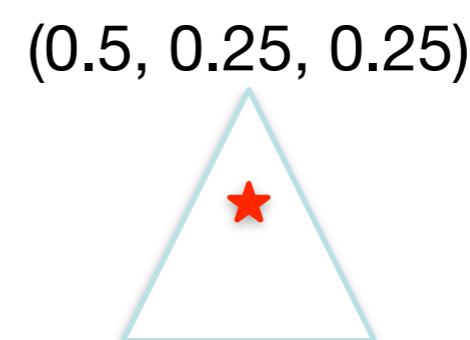
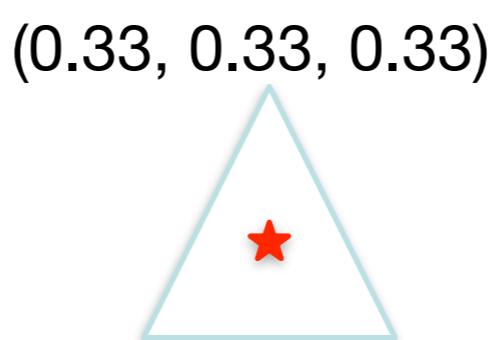
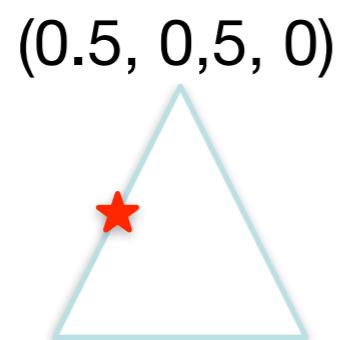
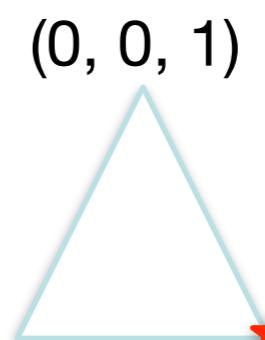
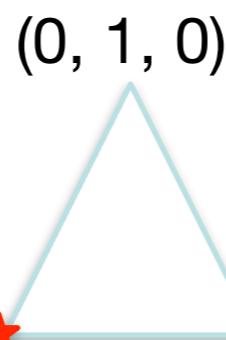
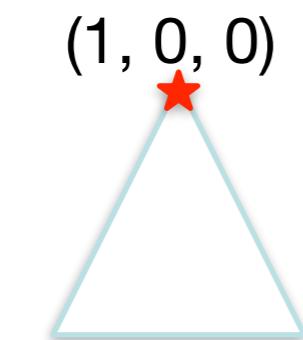
# Multinomial Distributions

- Distribution over discrete outcomes
- Represented by a non-negative vector that sums to one
- Picture repression of multinomial distributions over three discrete outcomes



# Multinomial Distributions

- Distribution over discrete outcomes
- Represented by a non-negative vector that sums to one
- Picture repression of multinomial distributions over three discrete outcomes



The Dirichlet distribution will give us a distribution over where a multinomial can be placed

# Dirichlet Distribution as Priors over Multinomials

Think of  $\alpha$  as a scale and  $m$  as a center

$$P(p|\alpha\mathbf{m}) = \frac{\Gamma(\sum_k \alpha m_k)}{\prod_k \gamma(\alpha m_k)} \prod \rho_k^{\alpha m_k - 1}$$

# Dirichlet Distribution as Priors over Multinomials

Think of  $\alpha$  as a scale and  $m$  as a center

$$P(p|\alpha m) = \frac{\Gamma(\sum_k \alpha m_k)}{\prod_k \gamma(\alpha m_k)} \prod \rho_k^{\alpha m_k - 1}$$

# Dirichlet Distribution as Priors over Multinomials

Think of  $\alpha$  as a scale and  $m$  as a center

$$P(p|\alpha\mathbf{m}) = \frac{\Gamma(\sum_k \alpha m_k)}{\prod_k \gamma(\alpha m_k)} \prod \rho_k^{\alpha m_k - 1}$$

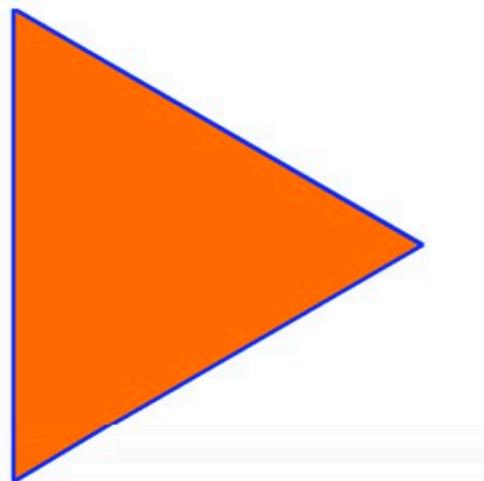
$$\alpha=3, \mathbf{m}=(0.3, 0.3, 0.3)$$

# Dirichlet Distribution as Priors over Multinomials

Think of  $\alpha$  as a scale and  $m$  as a center

$$P(p|\alpha\mathbf{m}) = \frac{\Gamma(\sum_k \alpha m_k)}{\prod_k \gamma(\alpha m_k)} \prod \rho_k^{\alpha m_k - 1}$$

$\alpha=3, \mathbf{m}=(0.3,0.3,0.3)$

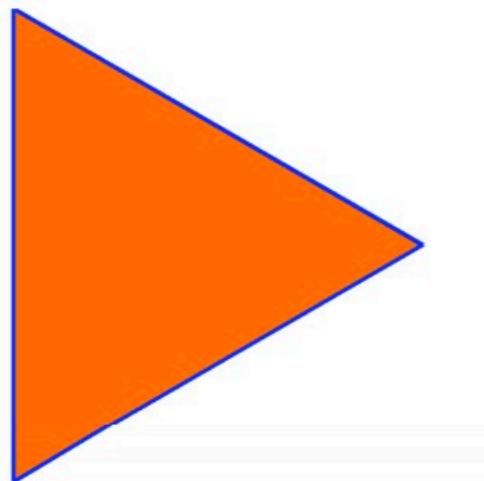


# Dirichlet Distribution as Priors over Multinomials

Think of  $\alpha$  as a scale and  $m$  as a center

$$P(p|\alpha\mathbf{m}) = \frac{\Gamma(\sum_k \alpha m_k)}{\prod_k \gamma(\alpha m_k)} \prod \rho_k^{\alpha m_k - 1}$$

$\alpha=3, \mathbf{m}=(0.3,0.3,0.3)$

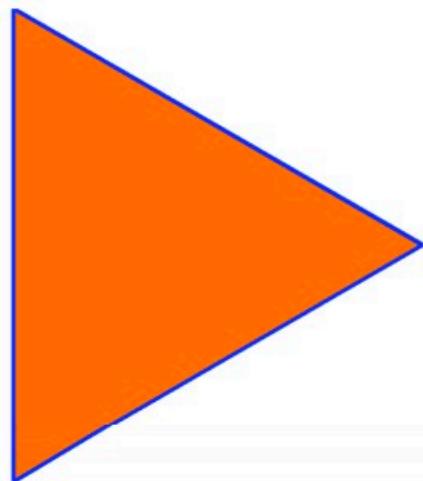


# Dirichlet Distribution as Priors over Multinomials

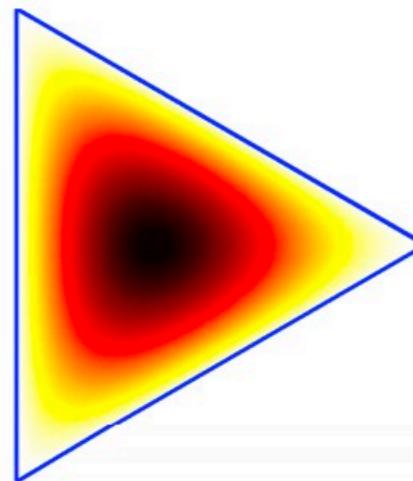
Think of  $\alpha$  as a scale and  $m$  as a center

$$P(p|\alpha\mathbf{m}) = \frac{\Gamma(\sum_k \alpha m_k)}{\prod_k \gamma(\alpha m_k)} \prod \rho_k^{\alpha m_k - 1}$$

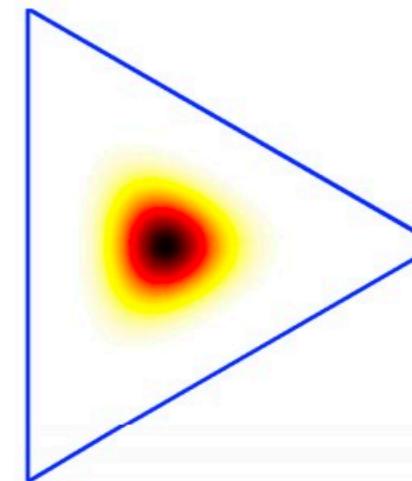
$\alpha=3, \mathbf{m}=(0.3,0.3,0.3)$



$\alpha=6, \mathbf{m}=(0.3,0.3,0.3)$



$\alpha=30, \mathbf{m}=(0.3,0.3,0.3)$

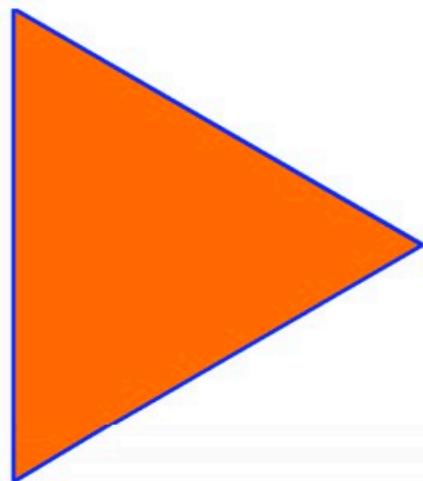


# Dirichlet Distribution as Priors over Multinomials

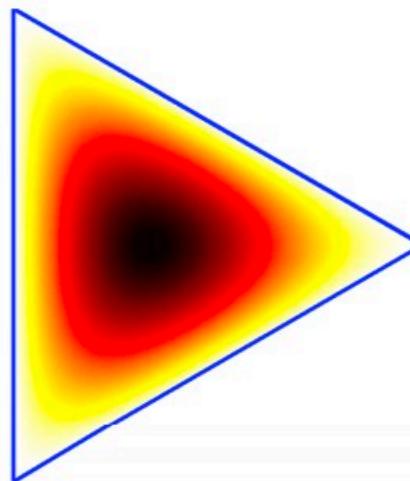
Think of  $\alpha$  as a scale and  $m$  as a center

$$P(p|\alpha\mathbf{m}) = \frac{\Gamma(\sum_k \alpha m_k)}{\prod_k \gamma(\alpha m_k)} \prod \rho_k^{\alpha m_k - 1}$$

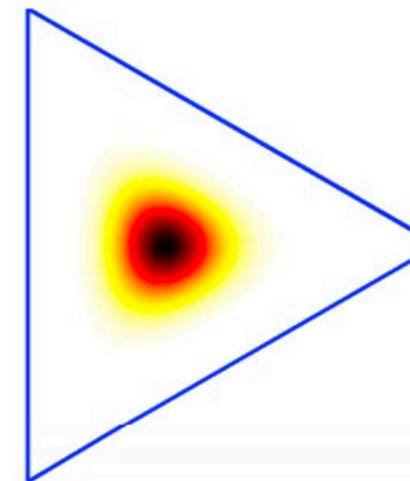
$\alpha=3, \mathbf{m}=(0.3,0.3,0.3)$



$\alpha=6, \mathbf{m}=(0.3,0.3,0.3)$



$\alpha=30, \mathbf{m}=(0.3,0.3,0.3)$



$\alpha=14, \mathbf{m}=(1/7, 5/7, 1/7)$

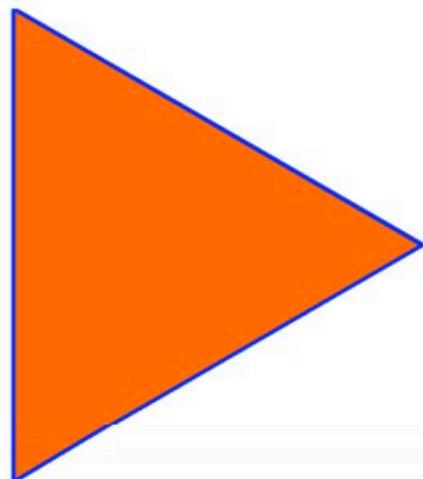
$\alpha=14, \mathbf{m}=(1/7, 1/7, 5/7)$

# Dirichlet Distribution as Priors over Multinomials

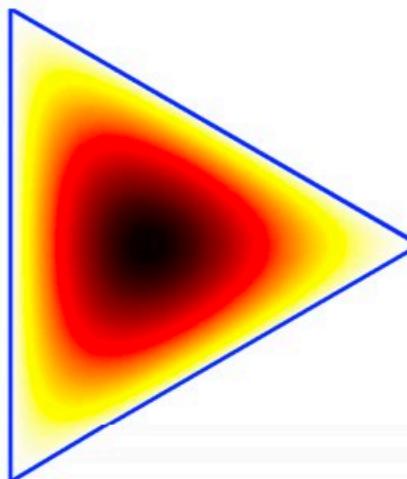
Think of  $\alpha$  as a scale and  $m$  as a center

$$P(p|\alpha m) = \frac{\Gamma(\sum_k \alpha m_k)}{\prod_k \gamma(\alpha m_k)} \prod \rho_k^{\alpha m_k - 1}$$

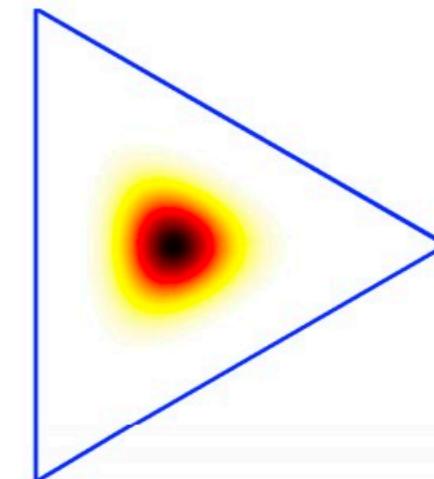
$\alpha=3, \mathbf{m}=(0.3,0.3,0.3)$



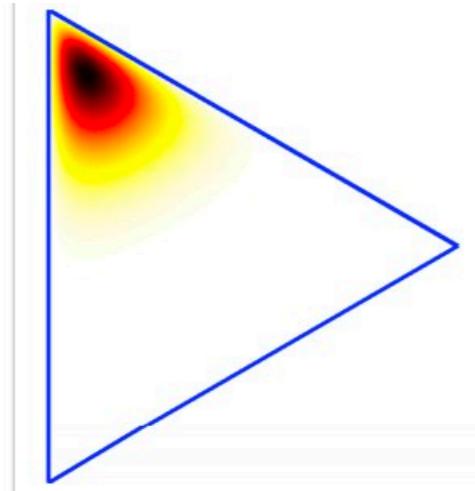
$\alpha=6, \mathbf{m}=(0.3,0.3,0.3)$



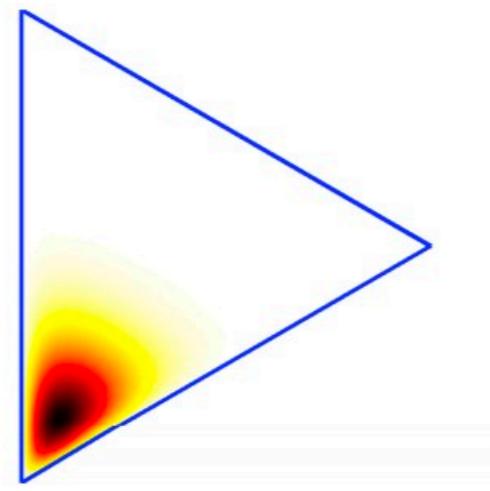
$\alpha=30, \mathbf{m}=(0.3,0.3,0.3)$



$\alpha=14, \mathbf{m}=(1/7, 5/7, 1/7)$



$\alpha=14, \mathbf{m}=(1/7, 1/7, 5/7)$

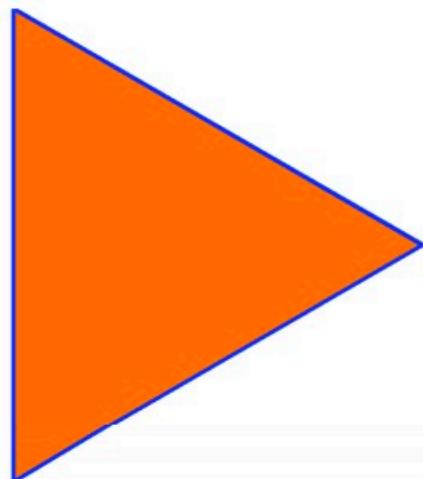


# Dirichlet Distribution as Priors over Multinomials

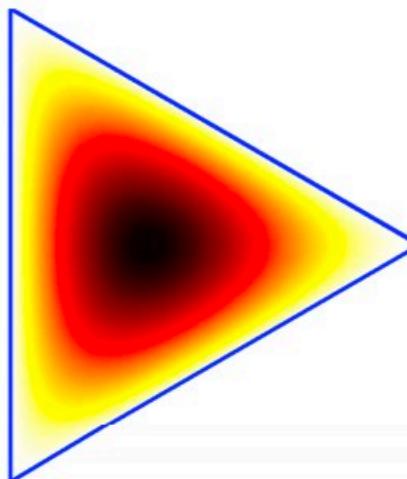
Think of  $\alpha$  as a scale and  $m$  as a center

$$P(p|\alpha m) = \frac{\Gamma(\sum_k \alpha m_k)}{\prod_k \gamma(\alpha m_k)} \prod \rho_k^{\alpha m_k - 1}$$

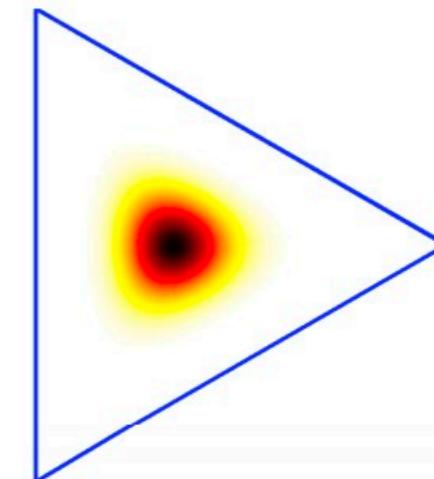
$\alpha=3, \mathbf{m}=(0.3,0.3,0.3)$



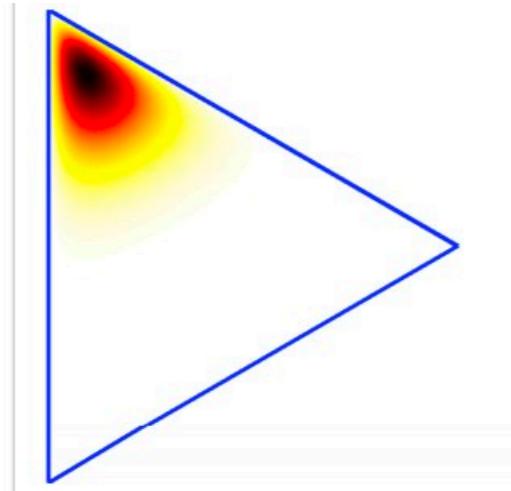
$\alpha=6, \mathbf{m}=(0.3,0.3,0.3)$



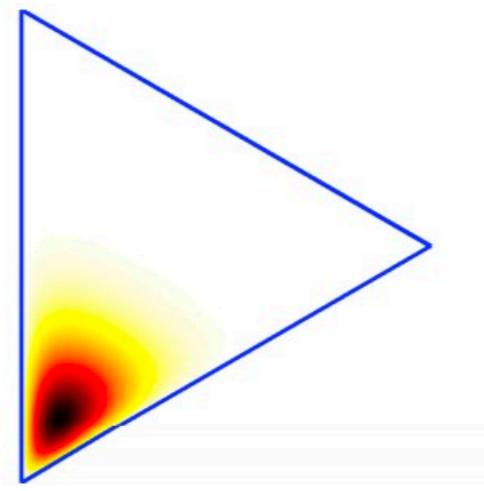
$\alpha=30, \mathbf{m}=(0.3,0.3,0.3)$



$\alpha=14, \mathbf{m}=(1/7, 5/7, 1/7)$



$\alpha=14, \mathbf{m}=(1/7, 1/7, 5/7)$

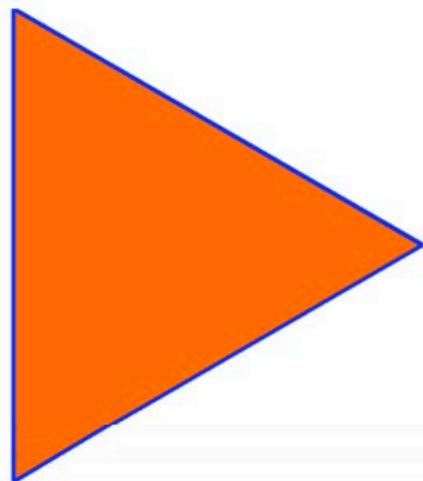


# Dirichlet Distribution as Priors over Multinomials

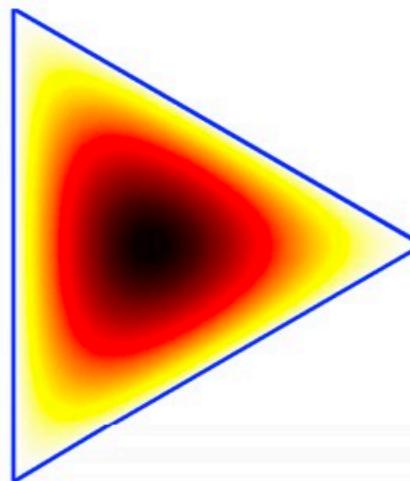
Think of  $\alpha$  as a scale and  $m$  as a center

$$P(p|\alpha m) = \frac{\Gamma(\sum_k \alpha m_k)}{\prod_k \gamma(\alpha m_k)} \prod \rho_k^{\alpha m_k - 1}$$

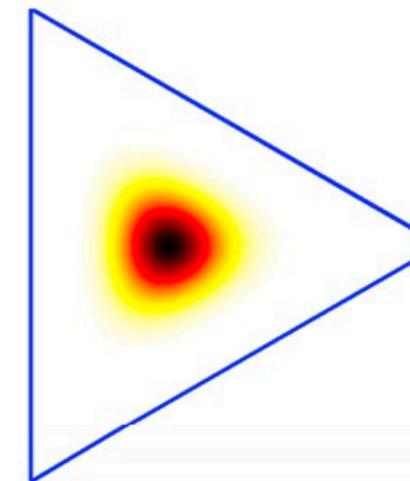
$\alpha=3, \mathbf{m}=(0.3,0.3,0.3)$



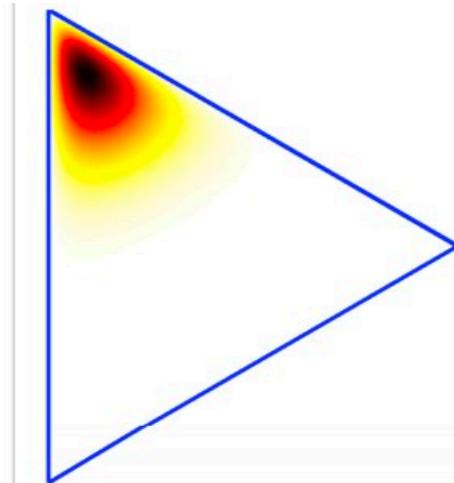
$\alpha=6, \mathbf{m}=(0.3,0.3,0.3)$



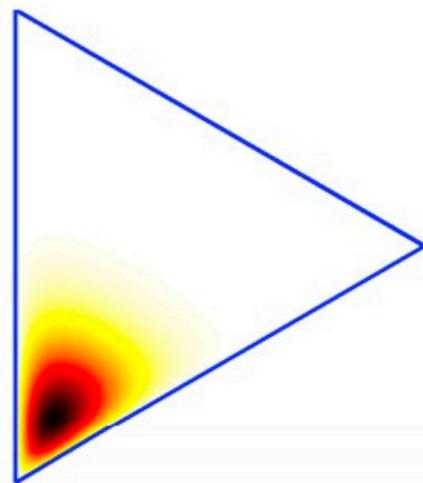
$\alpha=30, \mathbf{m}=(0.3,0.3,0.3)$



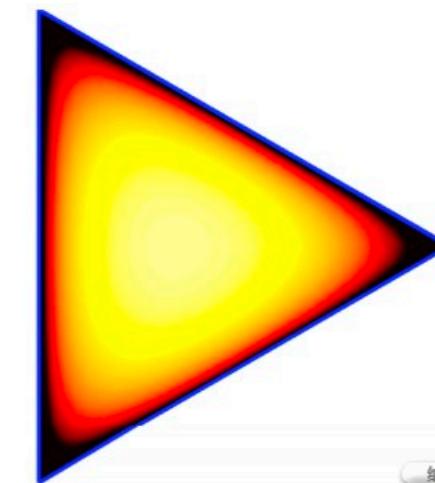
$\alpha=14, \mathbf{m}=(1/7,5/7,1/7)$



$\alpha=14, \mathbf{m}=(1/7, 1/7, 5/7)$

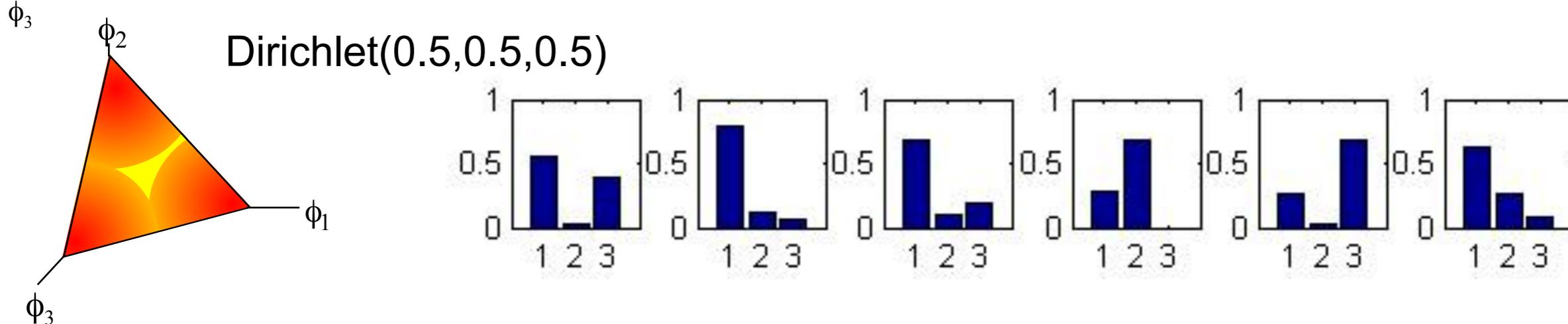
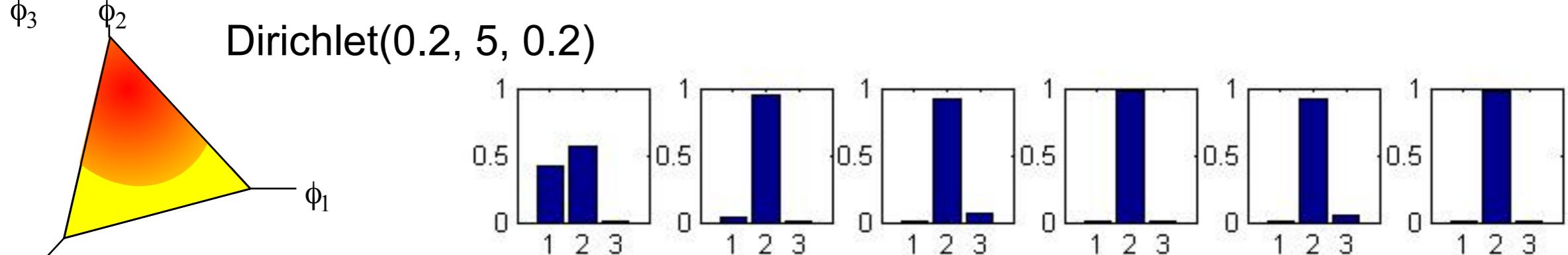
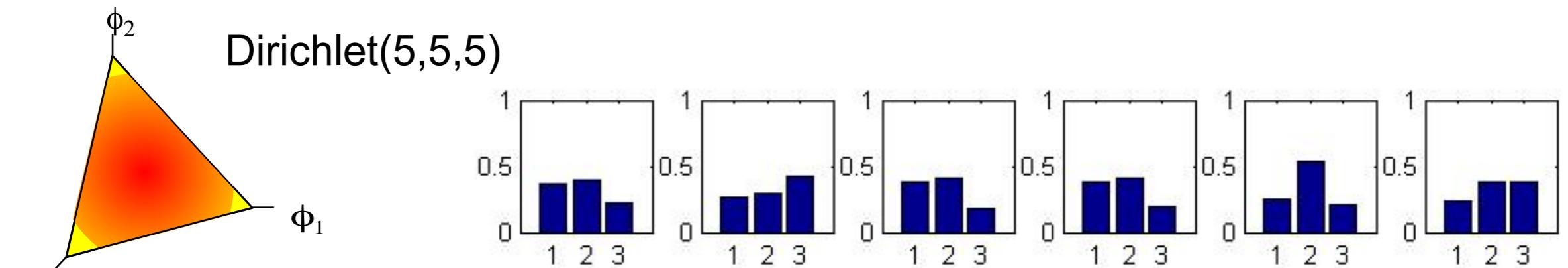


$\alpha=.2, \mathbf{m}=(0.3,0.3,0.3)$



# Dirichlet Distribution

- Example draws from a Dirichlet Distribution over the 3-simplex:



# LDA – Generative Process

- For each document  $d$ :

# LDA – Generative Process

- For each document  $d$ :
  - Choose document length

# LDA – Generative Process

- For each document  $d$ :
  - Choose document length  $N \sim Poisson(\xi)$

# LDA – Generative Process

- For each document  $d$ :
  - Choose document length  $N \sim Poisson(\xi)$
  - Choose a topic mixture distribution

# LDA – Generative Process

- For each document  $d$ :
  - Choose document length  $N \sim Poisson(\xi)$
  - Choose a topic mixture distribution  $\theta \sim Dir(\alpha)$

# LDA – Generative Process

- For each document  $d$ :
  - Choose document length  $N \sim Poisson(\xi)$
  - Choose a topic mixture distribution  $\theta \sim Dir(\alpha)$
  - For each word token in  $d$ :
    - Choose a topic

# LDA – Generative Process

- For each document  $d$ :
  - Choose document length  $N \sim Poisson(\xi)$
  - Choose a topic mixture distribution  $\theta \sim Dir(\alpha)$
  - For each word token in  $d$ :
    - Choose a topic  $Z_n \sim multinomial(\theta)$

# LDA – Generative Process

- For each document  $d$ :
  - Choose document length  $N \sim Poisson(\xi)$
  - Choose a topic mixture distribution  $\theta \sim Dir(\alpha)$
  - For each word token in  $d$ :
    - Choose a topic  $z_n \sim multinomial(\theta)$
    - Choose a word  $w_n$  from a multinomial distribution conditioned on the topic  $z_n$ .

# LDA – Generative Process

- For each document  $d$ :
    - Choose document length  $N \sim Poisson(\xi)$
    - Choose a topic mixture distribution  $\theta \sim Dir(\alpha)$
    - For each word token in  $d$ :
      - Choose a topic  $z_n \sim multinomial(\theta)$
      - Choose a word  $w_n$  from a multinomial distribution conditioned on the topic  $z_n$ .
- $\beta$  is a  $k$  by  $V$  matrix parameterized with the word probabilities.

# LDA – Generative Process

- For each document  $d$ :
    - Choose document length  $N \sim Poisson(\xi)$
    - Choose a topic mixture distribution  $\theta \sim Dir(\alpha)$
    - For each word token in  $d$ :
      - Choose a topic  $z_n \sim multinomial(\theta)$
      - Choose a word  $w_n$  from a multinomial distribution conditioned on the topic  $z_n$ .
- $\beta$  is a  $k$  by  $V$  matrix parameterized with the word probabilities.

$$[\beta]_{k \times V} \quad \beta_{ij} = p(w_j | z=i)$$

# LDA – Generative Process

$$P(d) = \prod_{w \in d} \sum_{i=1..K} P(z=i | d) P(w | Topic_i)$$

iPod 0.15  
nano 0.080  
music .05  
download 0.02  
apple 0.01

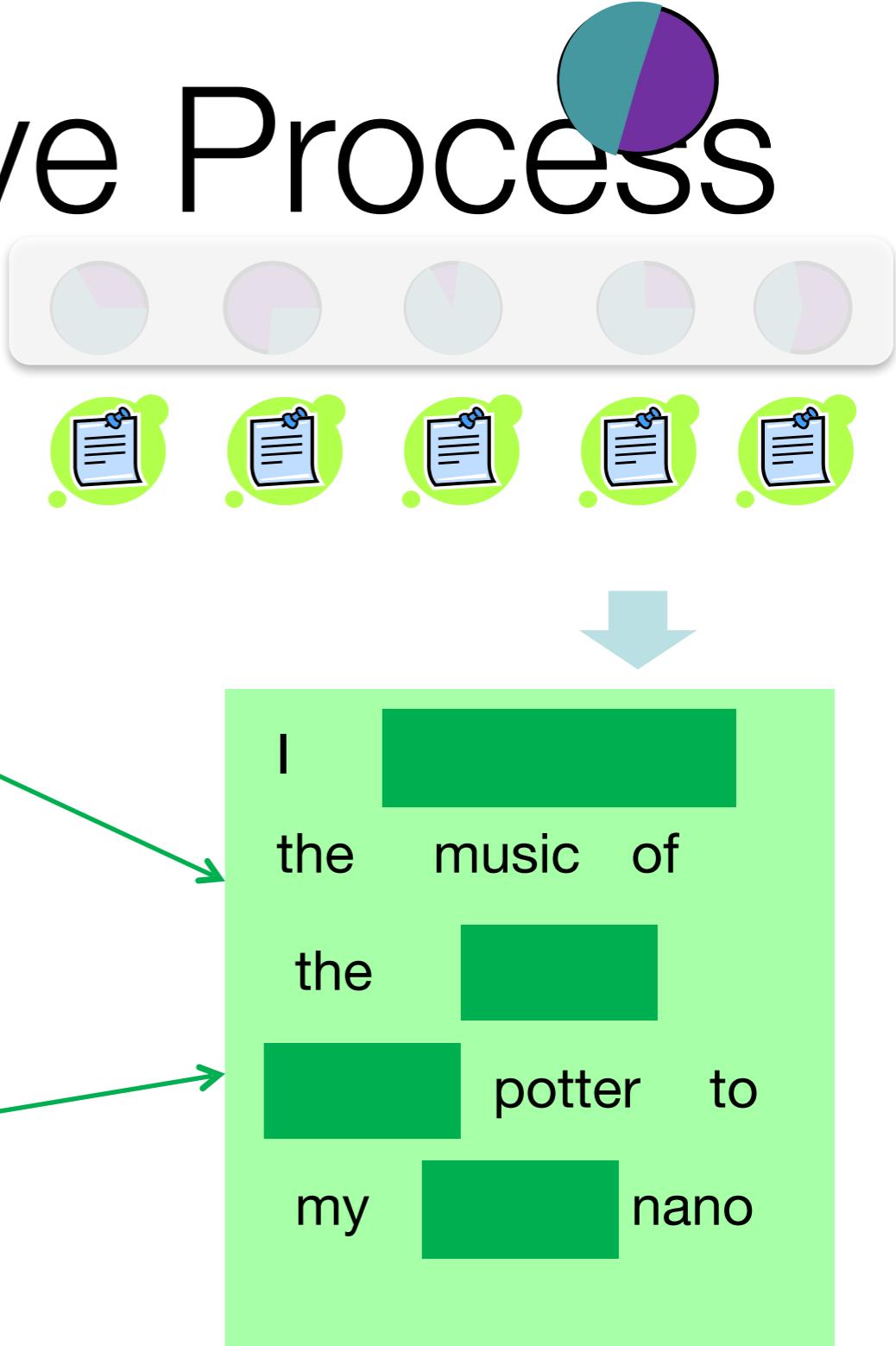
movie 0.100  
harry .090.  
potter 05  
ipod 0.01  
music 0.02

Topic 1

Apple iPod

Topic 2

Harry Potter



# LDA – Generative Process

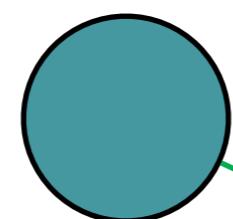
$$P(d) = \prod_{w \in d} \sum_{i=1..K} P(z=i | d) P(w | Topic_i)$$

ipod 0.15  
 nano 0.080  
 music .05  
 download 0.02  
 apple 0.01

*movie* 0.100  
*harry* .090.  
*potter* 05  
*ipod* 0.01  
*music* 0.02

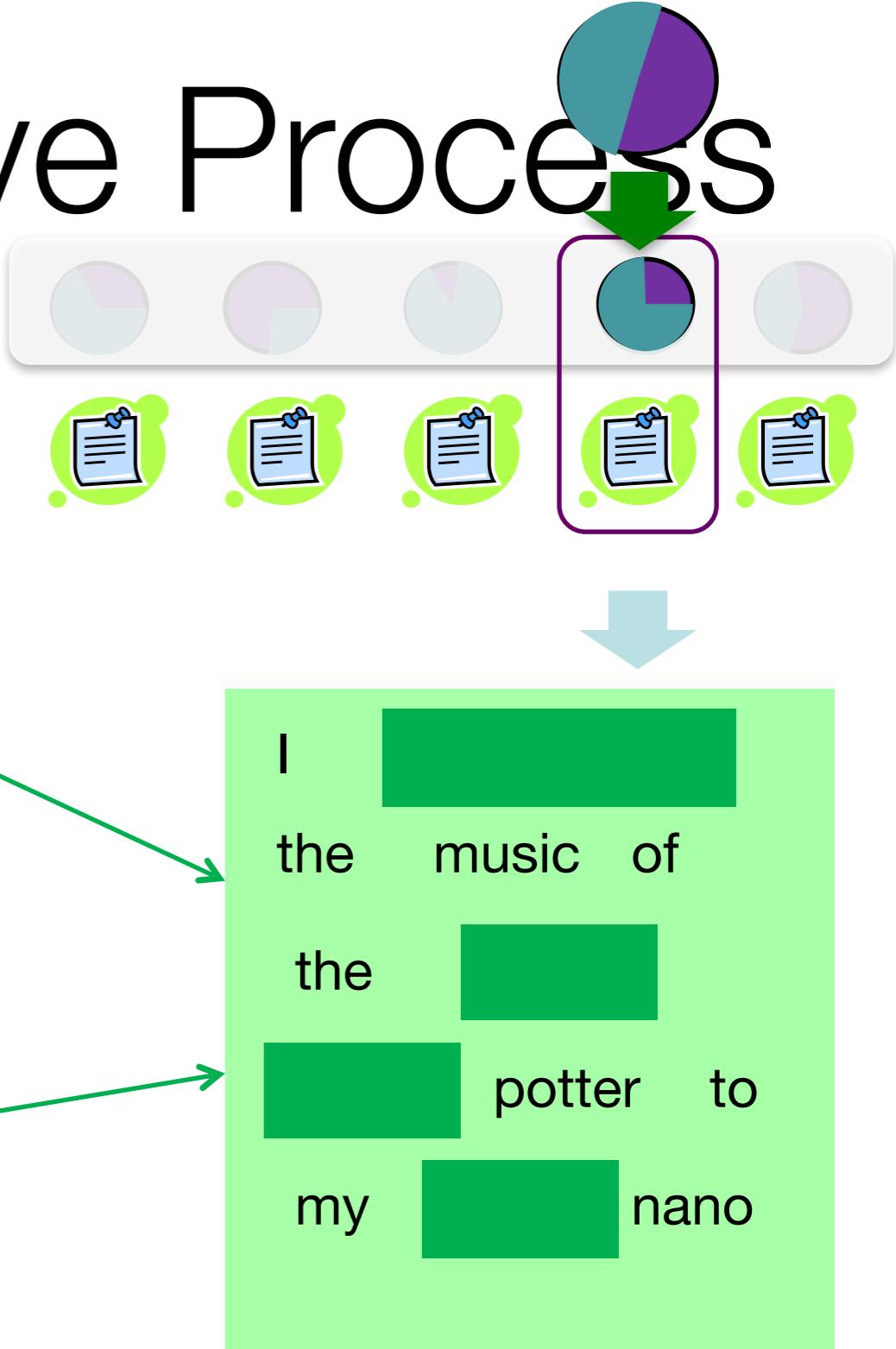
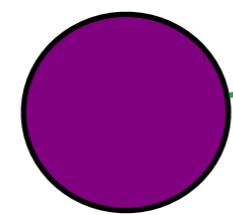
Topic 1

Apple iPod



Topic 2

Harry Potter

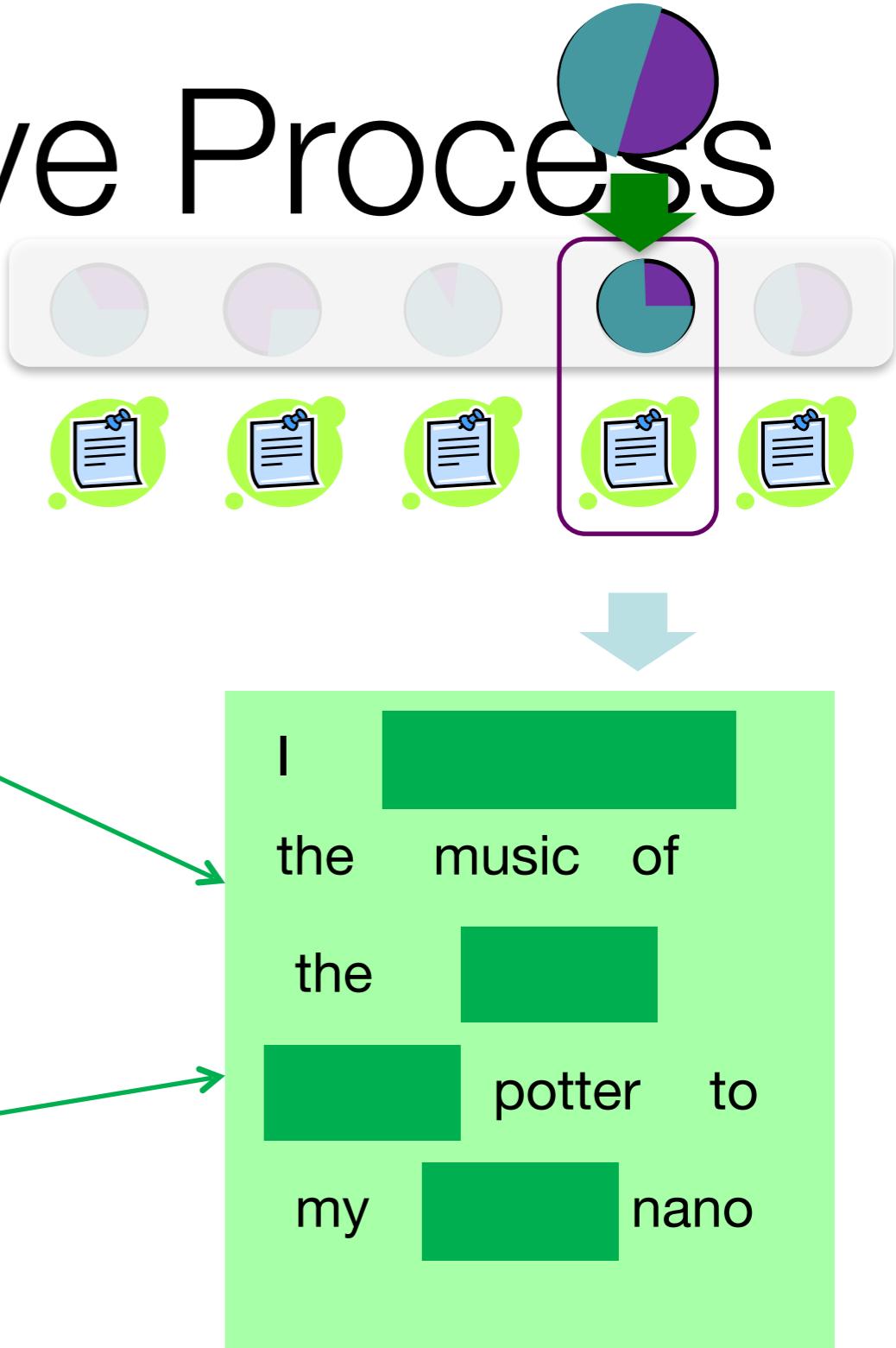


# LDA – Generative Process

$$P(d) = \prod_{w \in d} \sum_{i=1..K} P(z=i | d) P(w | Topic_i)$$

ipod 0.15  
 nano 0.080  
 music .05  
 download 0.02  
 apple 0.01

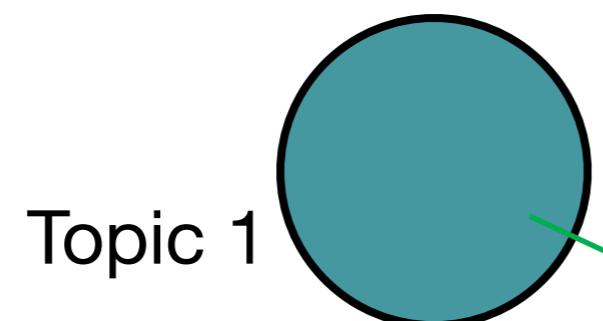
*movie* 0.100  
*harry* .090.  
*potter* 05  
*ipod* 0.01  
*music* 0.02



# LDA – Generative Process

$$P(d) = \prod_{w \in d} \sum_{i=1..K} P(z=i | d) P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01



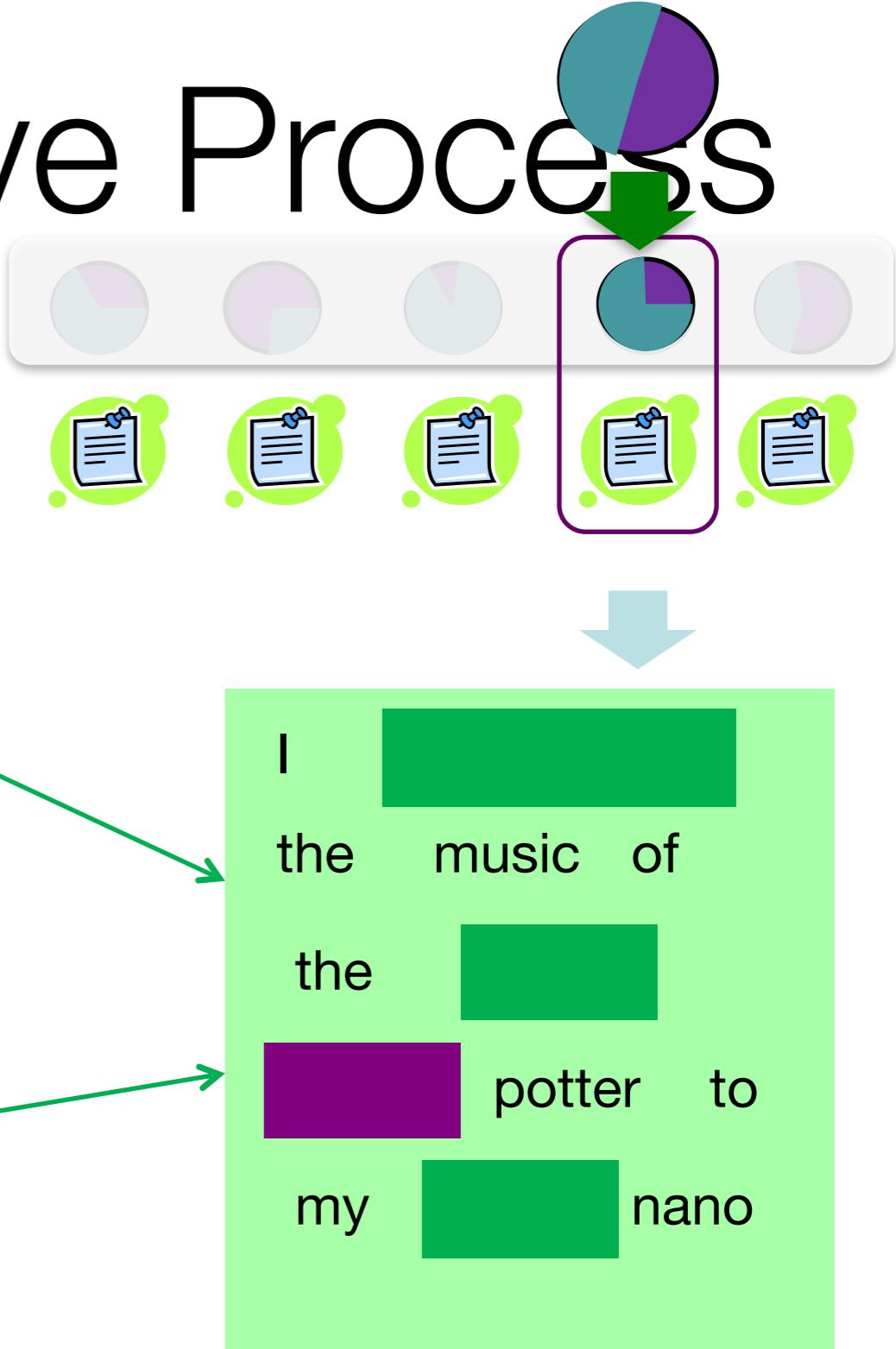
Apple iPod

movie	0.100
harry	.090.
potter	05
ipod	0.01
music	0.02

Topic 2



Harry Potter



# LDA – Generative Process

$$P(d) = \prod_{w \in d} \sum_{i=1..K} P(z=i | d) P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01

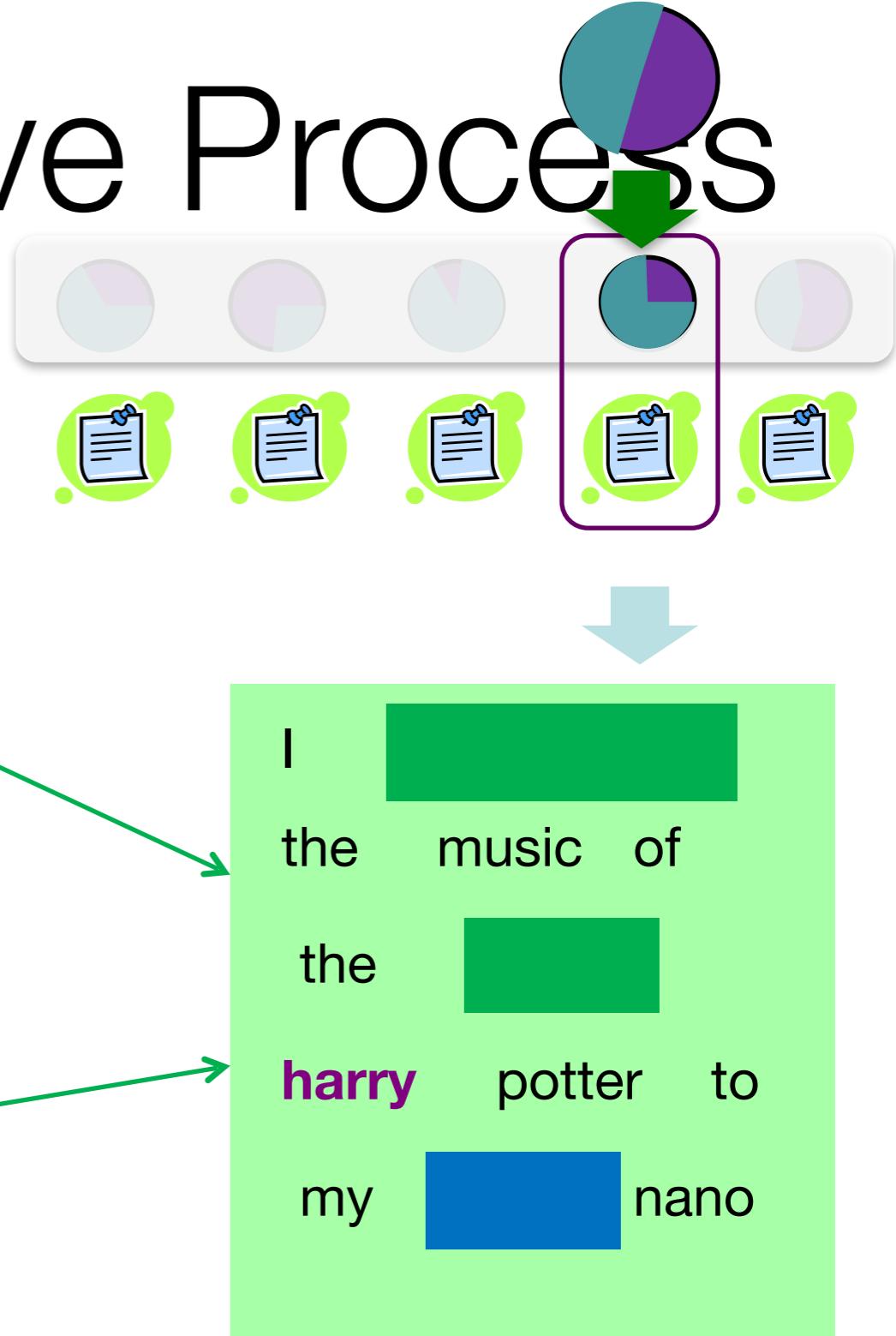
Topic 1

Apple iPod

movie	0.100
harry	.090.
potter	05
ipod	0.01
music	0.02

Topic 2

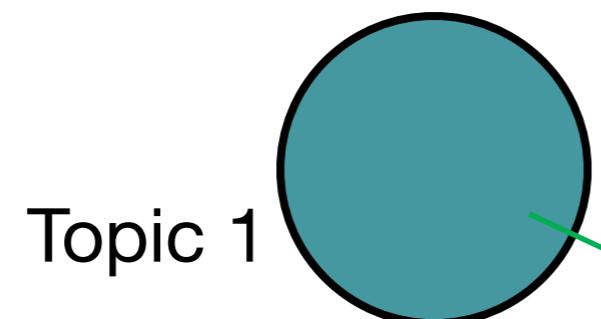
Harry Potter



# LDA – Generative Process

$$P(d) = \prod_{w \in d} \sum_{i=1..K} P(z=i | d) P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01



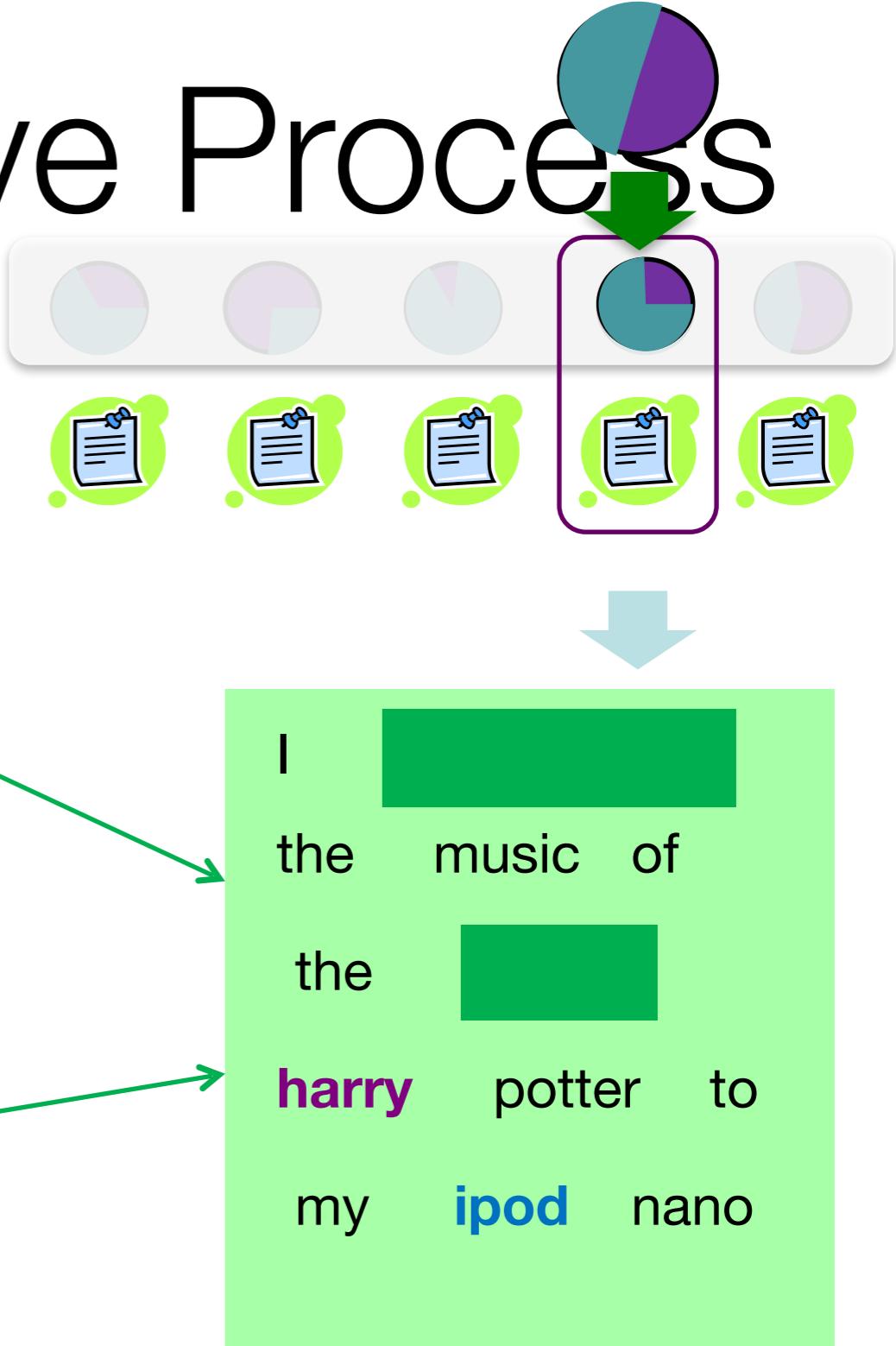
Apple iPod

movie	0.100
harry	.090.
potter	05
ipod	0.01
music	0.02

Topic 2



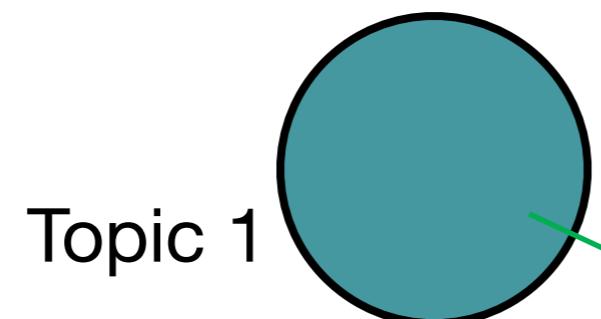
Harry Potter



# LDA – Generative Process

$$P(d) = \prod_{w \in d} \sum_{i=1..K} P(z=i | d) P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01



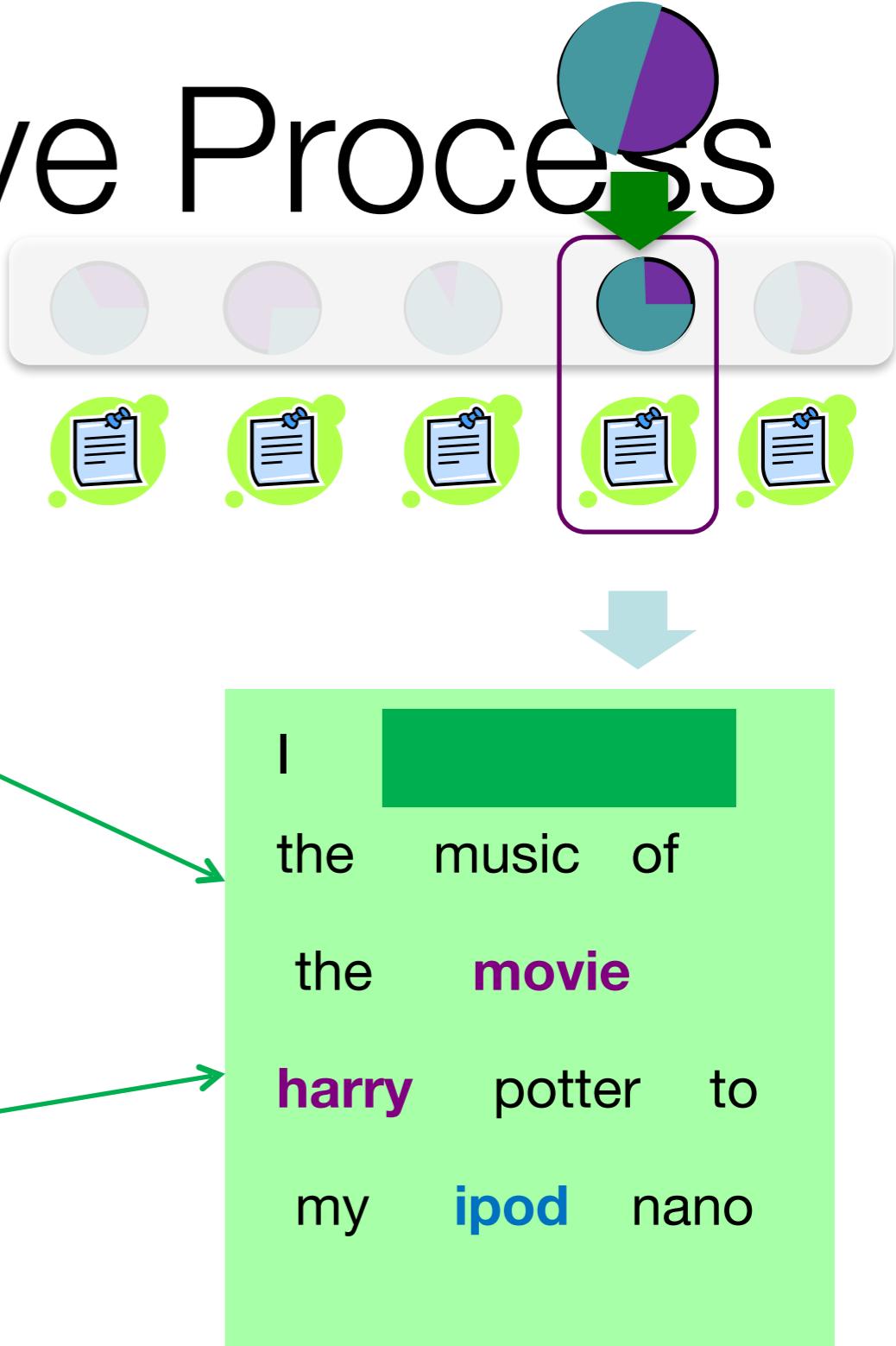
Apple iPod

movie	0.100
harry	.090.
potter	05
ipod	0.01
music	0.02

Topic 2



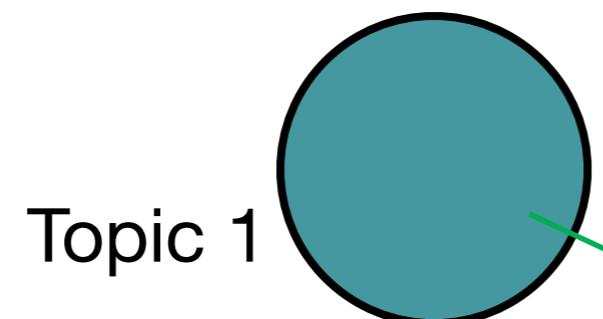
Harry Potter



# LDA – Generative Process

$$P(d) = \prod_{w \in d} \sum_{i=1..K} P(z=i | d) P(w | Topic_i)$$

iPod	0.15
nano	0.080
music	.05
download	0.02
apple	0.01



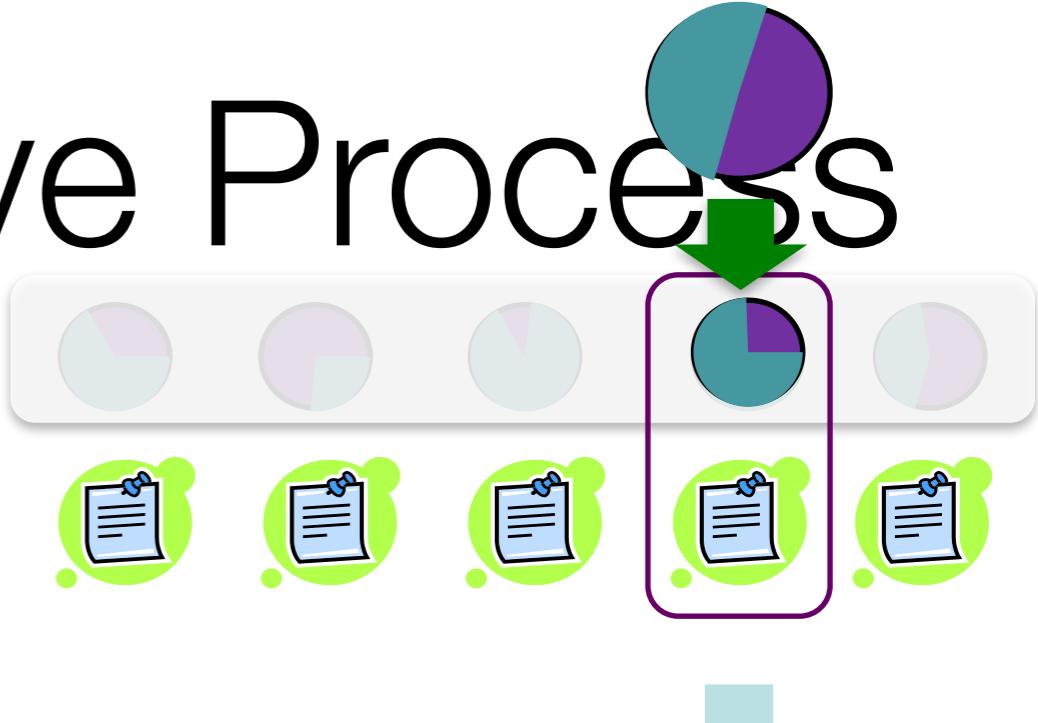
Apple iPod

movie	0.100
harry	.090.
potter	05
ipod	0.01
music	0.02

Topic 2



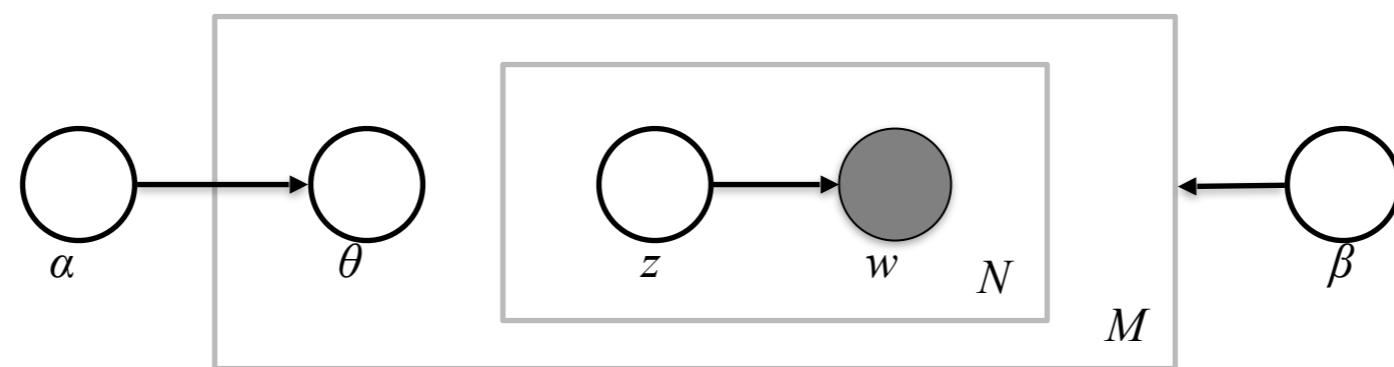
Harry Potter



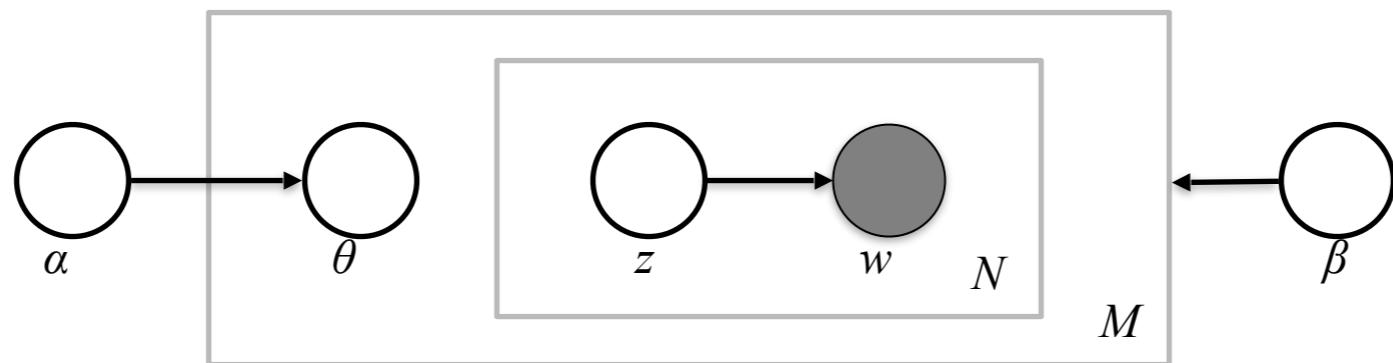
I **downloaded**  
the music of  
the **movie**  
**harry** potter to  
my **ipod** nano

# Graphical Model of LDA

# Graphical Model of LDA

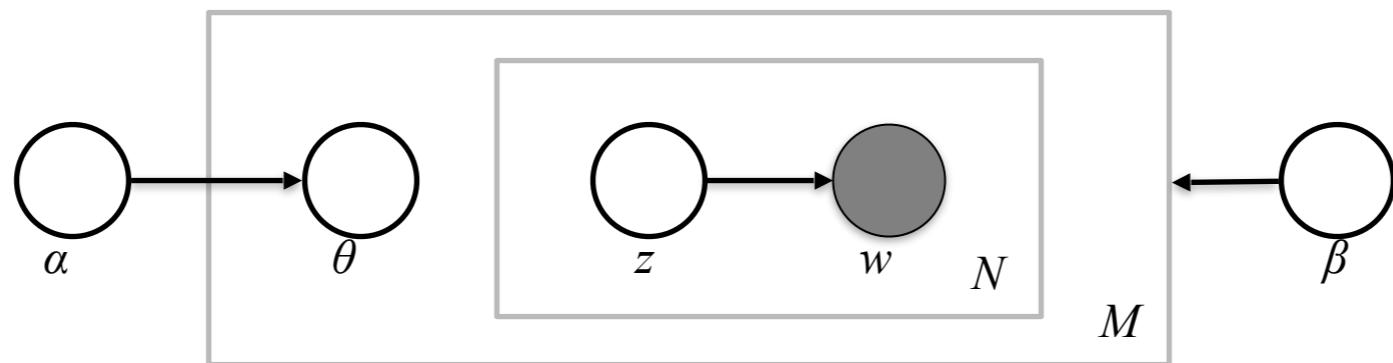


# Graphical Model of LDA



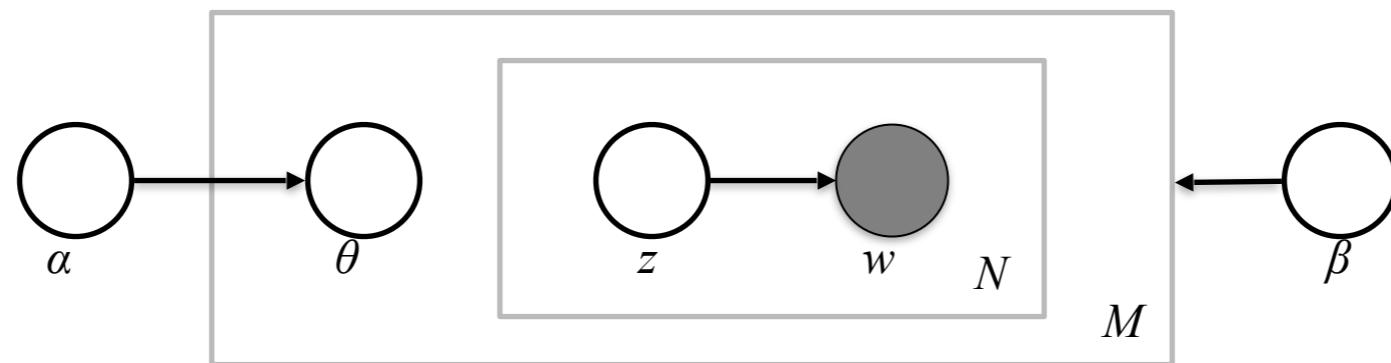
Complete likelihood:  
(if we observe all latent variables)

# Graphical Model of LDA



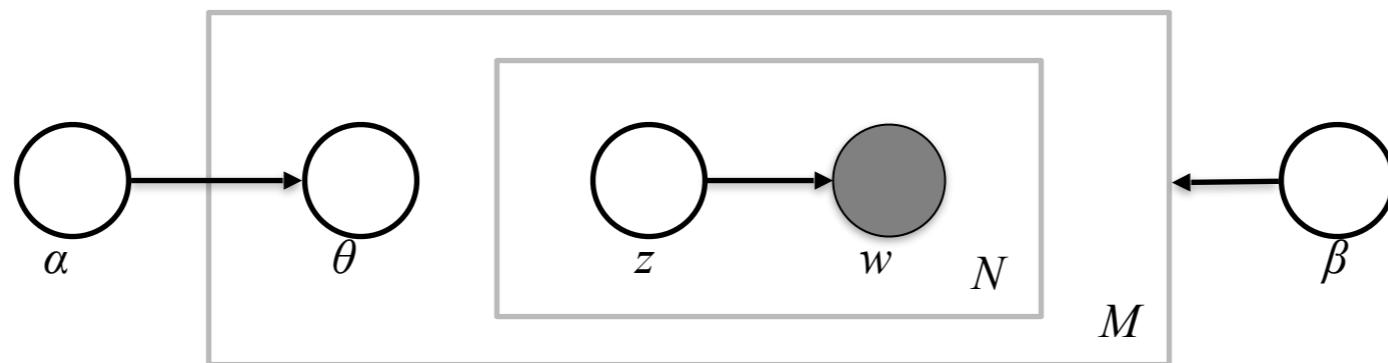
Complete likelihood:  $p(\theta, \mathbf{z}, \mathbf{w} | \alpha, \beta) =$   
(if we observe all latent variables)

# Graphical Model of LDA



Complete likelihood:  $p(\theta, \mathbf{z}, \mathbf{w} | \alpha, \beta) = p(\theta | \alpha) \prod_{n=1}^N p(z_n | \theta) p(w_n | z_n, \beta)$   
(if we observe all latent variables)

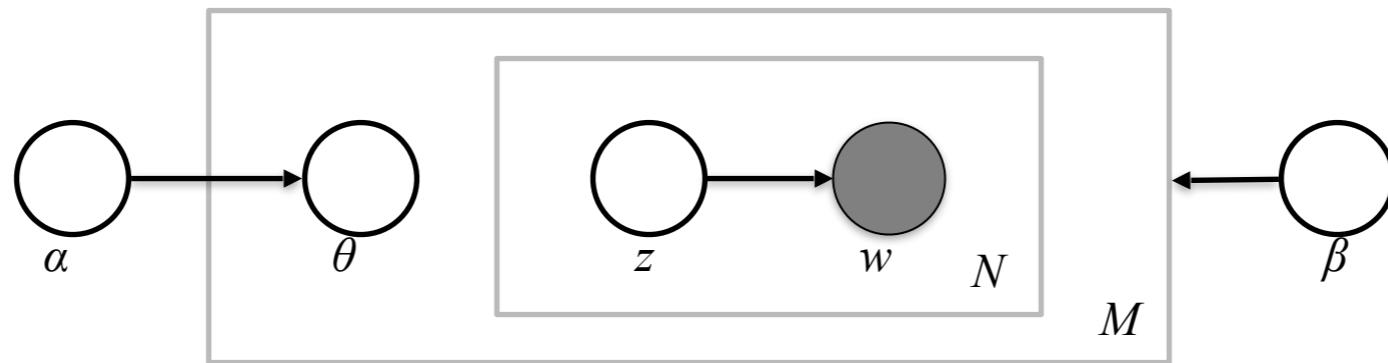
# Graphical Model of LDA



Complete likelihood:  $p(\theta, \mathbf{z}, \mathbf{w} | \alpha, \beta) = p(\theta | \alpha) \prod_{n=1}^N p(z_n | \theta) p(w_n | z_n, \beta)$   
(if we observe all latent variables)

Data likelihood:

# Graphical Model of LDA

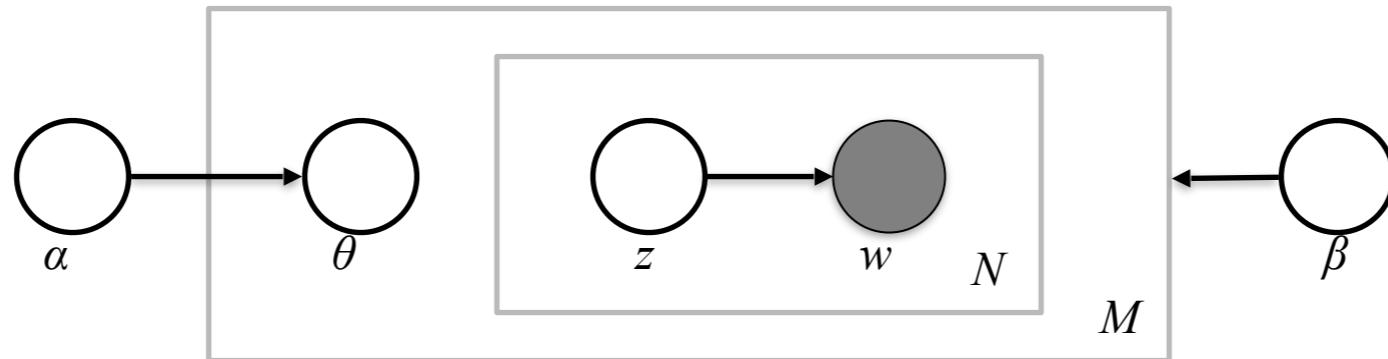


Complete likelihood:  $p(\theta, \mathbf{z}, \mathbf{w} | \alpha, \beta) = p(\theta | \alpha) \prod_{n=1}^N p(z_n | \theta) p(w_n | z_n, \beta)$   
(if we observe all latent variables)

$$p(\mathbf{w} | \alpha, \beta) = \int p(\theta | \alpha) \left( \prod_{n=1}^N \sum_{z_n} p(z_n | \theta) p(w_n | z_n, \beta) \right) d\theta$$

Data likelihood:

# Graphical Model of LDA



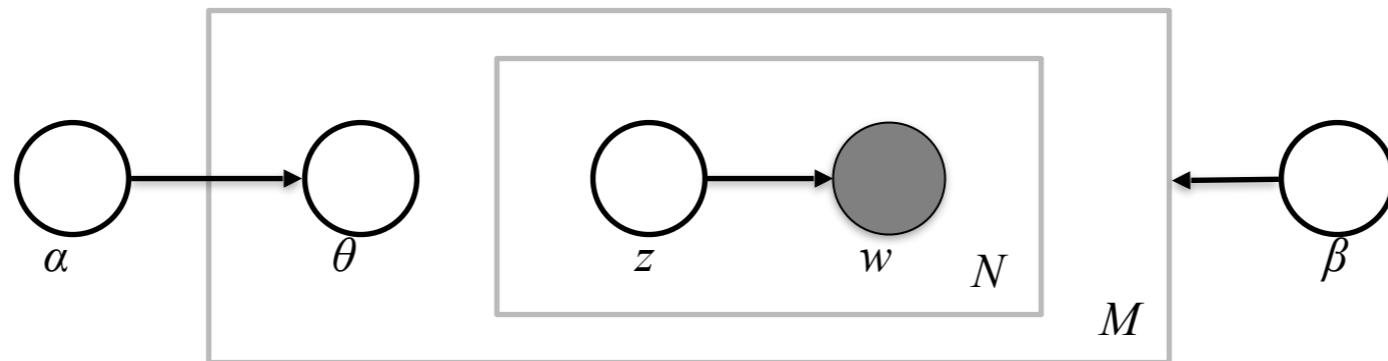
Complete likelihood:  $p(\theta, \mathbf{z}, \mathbf{w} | \alpha, \beta) = p(\theta | \alpha) \prod_{n=1}^N p(z_n | \theta) p(w_n | z_n, \beta)$   
(if we observe all latent variables)

$$p(\mathbf{w} | \alpha, \beta) = \int p(\theta | \alpha) \left( \prod_{n=1}^N \sum_{z_n} p(z_n | \theta) p(w_n | z_n, \beta) \right) d\theta$$

Data likelihood:

- $P(z|d)$  are no longer fixed parameters to be estimated...

# Graphical Model of LDA



Complete likelihood:  $p(\theta, \mathbf{z}, \mathbf{w} | \alpha, \beta) = p(\theta | \alpha) \prod_{n=1}^N p(z_n | \theta) p(w_n | z_n, \beta)$   
(if we observe all latent variables)

$$p(\mathbf{w} | \alpha, \beta) = \int p(\theta | \alpha) \left( \prod_{n=1}^N \sum_{z_n} p(z_n | \theta) p(w_n | z_n, \beta) \right) d\theta$$

Data likelihood:

- $P(z|d)$  are no longer fixed parameters to be estimated...
- But inference becomes much more complicated

# Putting all the pieces together

# Putting all the pieces together

- Latent Dirichlet Allocation (LDA) is a fully Bayesian version of ...

# Putting all the pieces together

- Latent Dirichlet Allocation (LDA) is a fully Bayesian version of ...
- Probabilistic Latent Semantic Indexing (pLSI) is a probabilistic version of ...

# Putting all the pieces together

- Latent Dirichlet Allocation (LDA) is a fully Bayesian version of ...
- Probabilistic Latent Semantic Indexing (pLSI) is a probabilistic version of ...
- Latent Semantic Indexing (LSI) is a dimensional reduction using the Singular Value Decomposition (SVD)

# Putting all the pieces together

- Latent Dirichlet Allocation (LDA) is a fully Bayesian version of ...
- Probabilistic Latent Semantic Indexing (pLSI) is a probabilistic version of ...
- Latent Semantic Indexing (LSI) is a dimensional reduction using the Singular Value Decomposition (SVD)
- PLSA is also proved to be equivalent to non-negative matrix factorization (with a specific loss function)

# Application to corpus data

- TASA corpus: text from first grade to college
- 26414 word types, over 37000 documents, used approximately 6 million word tokens
- Run Gibbs for models with  $T = 300, 500, \dots, 1700$  topics

# A selection from 500 topics $[P(w|z = j)]$

THEORY	SPACE	ART	STUDENTS	BRAIN	CURRENT
SCIENTISTS	EARTH	PAINT	TEACHER	NERVE	ELECTRICITY
EXPERIMENT	MOON	ARTIST	STUDENT	SENSE	ELECTRIC
OBSERVATIONS	PLANET	PAINTING	TEACHERS	SENSES	CIRCUIT
SCIENTIFIC	ROCKET	PAINTED	TEACHING	ARE	IS
EXPERIMENTS	MARS	ARTISTS	CLASS	NERVOUS	ELECTRICAL
HYPOTHESIS	ORBIT	MUSEUM	CLASSROOM	NERVES	VOLTAGE
EXPLAIN	ASTRONAUTS	WORK	SCHOOL	BODY	FLOW
SCIENTIST	FIRST	PAINTINGS	LEARNING	SMELL	BATTERY
OBSERVED	SPACECRAFT	STYLE	PUPILS	TASTE	WIRE
EXPLANATION	JUPITER	PICTURES	CONTENT	TOUCH	WIRES
BASED	SATELLITE	WORKS	INSTRUCTION	MESSAGES	SWITCH
OBSERVATION	SATELLITES	OWN	TAUGHT	IMPULSES	CONNECTED
IDEA	ATMOSPHERE	SCULPTURE	GROUP	CORD	ELECTRONS
EVIDENCE	SPACESHIP	PAINTER	GRADE	ORGANS	RESISTANCE
THEORIES	SURFACE	ARTS	SHOULD	SPINAL	POWER
BELIEVED	SCIENTISTS	BEAUTIFUL	GRADES	FIBERS	CONDUCTORS
DISCOVERED	ASTRONAUT	DESIGNS	CLASSES	SENSORY	CIRCUITS
OBSERVE	SATURN	PORTRAIT	PUPIL	PAIN	TUBE
FACTS	MILES	PAINTERS	GIVEN	IS	NEGATIVE

# A selection from 500 topics $[P(w|z = j)]$

MIND	STORY	FIELD	SCIENCE	BALL	JOB
WORLD	STORIES	MAGNETIC	STUDY	GAME	WORK
DREAM	TELL	MAGNET	SCIENTISTS	TEAM	JOBS
DREAMS	CHARACTER	WIRE	SCIENTIFIC	FOOTBALL	CAREER
THOUGHT	CHARACTERS	NEEDLE	KNOWLEDGE	BASEBALL	EXPERIENCE
IMAGINATION	AUTHOR	CURRENT	WORK	PLAYERS	EMPLOYMENT
MOMENT	READ	COIL	RESEARCH	PLAY	OPPORTUNITIES
THOUGHTS	TOLD	POLES	CHEMISTRY	FIELD	WORKING
OWN	SETTING	IRON	TECHNOLOGY	PLAYER	TRAINING
REAL	TALES	COMPASS	MANY	BASKETBALL	SKILLS
LIFE	PLOT	LINES	MATHEMATICS	COACH	CAREERS
IMAGINE	TELLING	CORE	BIOLOGY	PLAYED	POSITIONS
SENSE	SHORT	ELECTRIC	FIELD	PLAYING	FIND
CONSCIOUSNESS	FICTION	DIRECTION	PHYSICS	HIT	POSITION
STRANGE	ACTION	FORCE	LABORATORY	TENNIS	FIELD
FEELING	TRUE	MAGNETS	STUDIES	TEAMS	OCCUPATIONS
WHOLE	EVENTS	BE	WORLD	GAMES	REQUIRE
BEING	TELLS	MAGNETISM	SCIENTIST	SPORTS	OPPORTUNITY
MIGHT	TALE	POLE	STUDYING	BAT	EARN
HOPE	NOVEL	INDUCED	SCIENCES	TERRY	ABLE

# A selection from 500 topics $[P(w|z = j)]$

MIND	STORY	<b>FIELD</b>	SCIENCE	BALL	JOB
WORLD	STORIES	MAGNETIC	STUDY	GAME	WORK
DREAM	TELL	MAGNET	SCIENTISTS	TEAM	JOBS
DREAMS	CHARACTER	WIRE	SCIENTIFIC	FOOTBALL	CAREER
THOUGHT	CHARACTERS	NEEDLE	KNOWLEDGE	BASEBALL	EXPERIENCE
IMAGINATION	AUTHOR	CURRENT	WORK	PLAYERS	EMPLOYMENT
MOMENT	READ	COIL	RESEARCH	PLAY	OPPORTUNITIES
THOUGHTS	TOLD	POLES	CHEMISTRY	<b>FIELD</b>	WORKING
OWN	SETTING	IRON	TECHNOLOGY	PLAYER	TRAINING
REAL	TALES	COMPASS	MANY	BASKETBALL	SKILLS
LIFE	PLOT	LINES	MATHEMATICS	COACH	CAREERS
IMAGINE	TELLING	CORE	BIOLOGY	PLAYED	POSITIONS
SENSE	SHORT	ELECTRIC	<b>FIELD</b>	PLAYING	FIND
CONSCIOUSNESS	FICTION	DIRECTION	PHYSICS	HIT	POSITION
STRANGE	ACTION	FORCE	LABORATORY	TENNIS	<b>FIELD</b>
FEELING	TRUE	MAGNETS	STUDIES	TEAMS	OCCUPATIONS
WHOLE	EVENTS	BE	WORLD	GAMES	REQUIRE
BEING	TELLS	MAGNETISM	SCIENTIST	SPORTS	OPPORTUNITY
MIGHT	TALE	POLE	STUDYING	BAT	EARN
HOPE	NOVEL	INDUCED	SCIENCES	TERRY	ABLE

# Variants of Latent Dirichlet Allocation

# Variants of Latent Dirichlet Allocation

- **Syntactic topic model:** A word or its topic is influenced by its syntactic position.

# Variants of Latent Dirichlet Allocation

- Syntactic topic model: A word or its topic is influenced by its syntactic position.
- Correlated topic model, hierarchical topic model, ....: Some topics resemble other topics.

# Variants of Latent Dirichlet Allocation

- [Syntactic topic model](#): A word or its topic is influenced by its syntactic position.
- [Correlated topic model, hierarchical topic model, ...](#): Some topics resemble other topics.
- [Polylingual topic model](#): All versions of the same document use the same topic mixture, even if they're in different languages.  
(Why is this useful?)

# Variants of Latent Dirichlet Allocation

- [Syntactic topic model](#): A word or its topic is influenced by its syntactic position.
- [Correlated topic model, hierarchical topic model, ...](#): Some topics resemble other topics.
- [Polylingual topic model](#): All versions of the same document use the same topic mixture, even if they're in different languages.  
(Why is this useful?)
- [Relational topic model](#): Documents on the same topic are generated separately but tend to link to one another. (Why is this useful?)

# Variants of Latent Dirichlet Allocation

- [Syntactic topic model](#): A word or its topic is influenced by its syntactic position.
- [Correlated topic model, hierarchical topic model, ...](#): Some topics resemble other topics.
- [Polylingual topic model](#): All versions of the same document use the same topic mixture, even if they're in different languages.  
(Why is this useful?)
- [Relational topic model](#): Documents on the same topic are generated separately but tend to link to one another.  
(Why is this useful?)
- [Dynamic topic model](#): We also observe a year for each document. The  $k$  topics  $\beta$  used in 2011 have evolved slightly from their counterparts in 2010.



©DreamWorks

# How to train your LDA

# How to train your LDA model

# How to train your LDA model

- We're interested in the posterior distribution:

$$p(Z|X, \Theta)$$

# How to train your LDA model

- We're interested in the posterior distribution:

$$p(Z|X, \Theta)$$

topics

# How to train your LDA model

- We're interested in the posterior distribution:

$$p(Z|X, \Theta)$$

topics                    data

# How to train your LDA model

- We're interested in the posterior distribution:

$$p(Z|X, \Theta)$$

topics                    data                    parameters

# How to train your LDA model

- We're interested in the posterior distribution:

$$p(Z|X, \Theta)$$

topics                    data                    parameters

- Here, the latent variables are the topic assignments  $z$  and the topics are Theta.  $X$  is the words (divided into documents) and  $\Theta$  are hyperparameters to the Dirichlet distributions:  $\alpha$  for topic proportions and  $\lambda$  for topics

$$p(z, \beta, \theta | \mathbf{w}, \alpha, \lambda)$$

# How to train your LDA model

- We're interested in the posterior distribution:

$$p(Z|X, \Theta)$$

topics

data

parameters

- Here, the latent variables are the topic assignments  $z$  and the topics are Theta.  $X$  is the words (divided into documents) and  $\Theta$  are hyperparameters to the Dirichlet distributions:  $\alpha$  for topic proportions and  $\lambda$  for topics

topic  
assignments

$$p(z, \beta, \theta | \mathbf{w}, \alpha, \lambda)$$

# How to train your LDA model

- We're interested in the posterior distribution:

$$p(Z|X, \Theta)$$

topics

data

parameters

- Here, the latent variables are the topic assignments  $z$  and the topics are Theta.  $X$  is the words (divided into documents) and  $\Theta$  are hyperparameters to the Dirichlet distributions:  $\alpha$  for topic proportions and  $\lambda$  for topics

topic  
assignments

$$p(z, \beta, \theta | w, \alpha, \lambda)$$

topic distributions

# How to train your LDA model

- We're interested in the posterior distribution:

$$p(Z|X, \Theta)$$

topics

data

parameters

- Here, the latent variables are the topic assignments  $z$  and the topics are Theta.  $X$  is the words (divided into documents) and  $\Theta$  are hyperparameters to the Dirichlet distributions:  $\alpha$  for topic proportions and  $\lambda$  for topics

topic  
assignments

$$p(z, \beta, \theta | w, \alpha, \lambda)$$

topic distributions

Each document's  
distribution over topics

# How to train your LDA model

- We're interested in the posterior distribution:

$$p(Z|X, \Theta)$$

topics            data            parameters

- Here, the latent variables are the topic assignments  $z$  and the topics are Theta.  $X$  is the words (divided into documents) and  $\Theta$  are hyperparameters to the Dirichlet distributions:  $\alpha$  for topic proportions and  $\lambda$  for topics

topic             $p(z, \beta, \theta | \mathbf{w}, \alpha, \lambda)$             Each document's  
assignments            topic distributions            distribution over topics

$$p(\mathbf{w}, \mathbf{z}, \theta, \beta | \alpha, \lambda) = \prod_k p(\beta_k | \lambda) \prod_d p(\theta_d | \alpha) \prod_n p(z_{d,n} | \theta_d) p(w_{d,n} | \beta_{z_{d,n}})$$

# How to train your LDA model

- We're interested in the posterior distribution:

$$p(Z|X, \Theta)$$

topics            data            parameters

- Here, the latent variables are the topic assignments  $z$  and the topics are Theta.  $X$  is the words (divided into documents) and  $\Theta$  are hyperparameters to the Dirichlet distributions:  $\alpha$  for topic proportions and  $\lambda$  for topics

topic assignments       $p(z, \beta, \theta | \mathbf{w}, \alpha, \lambda)$       Each document's  
topic distributions      distribution over topics



$$p(\mathbf{w}, \mathbf{z}, \theta, \beta | \alpha, \lambda) = \prod_k p(\beta_k | \lambda) \prod_d p(\theta_d | \alpha) \prod_n p(z_{d,n} | \theta_d) p(w_{d,n} | \beta_{z_{d,n}})$$

# Quick Detour: Monte Carlo Methods

# Quick Detour: Monte Carlo Methods

- Monte Carlo (MC) methods help you obtain a desired value by performing simulations involving probabilistic choices

# Quick Detour: Monte Carlo Methods

- Monte Carlo (MC) methods help you obtain a desired value by performing simulations involving probabilistic choices
- Example: How to calculate pi using a MC method

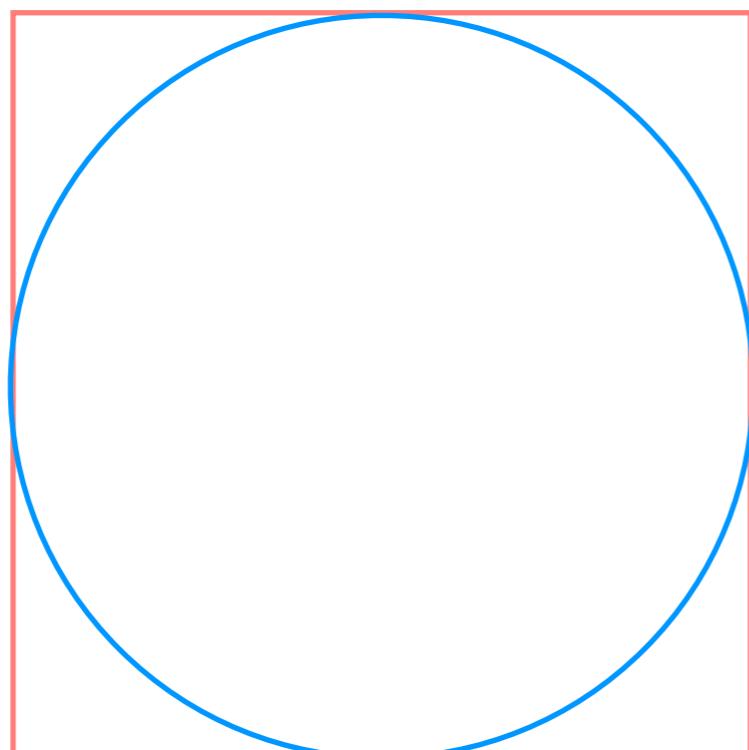
# Quick Detour: Monte Carlo Methods

- Monte Carlo (MC) methods help you obtain a desired value by performing simulations involving probabilistic choices
- Example: How to calculate pi using a MC method



# Quick Detour: Monte Carlo Methods

- Monte Carlo (MC) methods help you obtain a desired value by performing simulations involving probabilistic choices
- Example: How to calculate  $\pi$  using a MC method



# Quick Detour: Monte Carlo Methods

- Monte Carlo (MC) methods help you obtain a desired value by performing simulations involving probabilistic choices
- Example: How to calculate pi using a MC method



- Scatter rice (uniformly)

# Quick Detour: Monte Carlo Methods

- Monte Carlo (MC) methods help you obtain a desired value by performing simulations involving probabilistic choices
- Example: How to calculate pi using a MC method



- Scatter rice (uniformly)
- C is rice in circle

# Quick Detour: Monte Carlo Methods

- Monte Carlo (MC) methods help you obtain a desired value by performing simulations involving probabilistic choices
- Example: How to calculate pi using a MC method



- Scatter rice (uniformly)
- C is rice in circle
- S is rice in square parts not in circle

# Quick Detour: Monte Carlo Methods

- Monte Carlo (MC) methods help you obtain a desired value by performing simulations involving probabilistic choices
- Example: How to calculate pi using a MC method



- Scatter rice (uniformly)
- C is rice in circle
- S is rice in square parts not in circle
- $\pi \approx C / S$

# Gibbs Sampling

# Gibbs Sampling

- A form of Markov Chain Monte Carlo

# Gibbs Sampling

- A form of Markov Chain Monte Carlo
- Chain is a sequence of random variable states

# Gibbs Sampling

- A form of Markov Chain Monte Carlo
- Chain is a sequence of random variable states
- Given a state  $\{z_1, \dots, z_N\}$  and given certain technical conditions, sampling  $z_k \sim p(z_1, \dots, z_{k-1}, z_{k+1}, z_N | X, \Theta)$  for all  $k$  (repeatedly) results in a Markov Chain whose stationary distribution *is* the posterior

# Gibbs Sampling

- A form of Markov Chain Monte Carlo
- Chain is a sequence of random variable states
- Given a state  $\{z_1, \dots, z_N\}$  and given certain technical conditions, sampling  $z_k \sim p(z_1, \dots, z_{k-1}, z_{k+1}, z_N | X, \Theta)$  for all  $k$  (repeatedly) results in a Markov Chain whose stationary distribution is the **posterior**

The **posterior probability** is the probability of the parameters given the evidence

# Gibbs Sampling

- A form of Markov Chain Monte Carlo
- Chain is a sequence of random variable states
- Given a state  $\{z_1, \dots, z_N\}$  and given certain technical conditions, sampling  $z_k \sim p(z_1, \dots, z_{k-1}, z_{k+1}, z_N | X, \Theta)$  for all  $k$  (repeatedly) results in a Markov Chain whose stationary distribution is the **posterior**

The **posterior probability** is the probability of the parameters given the evidence
- For notational convenience, we'll call the whole topic assignment **z** and denote this whole thing *except* a particular topic assignment  $z_{d,n}$  as **z-d,n**

# Gibbs Sampling

- A form of Markov Chain Monte Carlo
- Chain is a sequence of random variable states
- Given a state  $\{z_1, \dots, z_N\}$  and given certain technical conditions, sampling  $z_k \sim p(z_1, \dots, z_{k-1}, z_{k+1}, z_N | X, \Theta)$  for all  $k$  (repeatedly) results in a Markov Chain whose stationary distribution is the **posterior**

The **posterior probability** is the probability of the parameters given the evidence
- For notational convenience, we'll call the whole topic assignment **z** and denote this whole thing except a particular topic assignment  $z_{d,n}$  as  $\mathbf{z}_{-d,n}$ 

Pretend we never saw the topic assignment for the  $n^{\text{th}}$  word in the  $d^{\text{th}}$  document

# What Gibbs Sampling is doing at an abstract level

Topic 1

wizard  
magic  
wand  
spell

Topic 2

movie  
book  
novel  
media

Topic 3

Harry  
Hermoine  
Ron  
Dumbledore

Topic 4

school  
teacher  
class  
prefect

# What Gibbs Sampling is doing at an abstract level

Topic 1

wizard  
magic  
wand  
spell

Topic 2

movie  
book  
novel  
media

Topic 3

Harry  
Hermoine  
Ron  
Dumbledore

Topic 4

school  
teacher  
class  
prefect

Harry Potter is a series of fantasy novels written by British author J. K. Rowling. The novels chronicle the life of a young wizard, Harry Potter, and his friends Hermione Granger and Ron Weasley, all of whom are students at Hogwarts School of Witchcraft and Wizardry. The main story arc concerns Harry's struggle against Lord Voldemort, a dark wizard who intends to become immortal, overthrow the wizard governing body known as the Ministry of Magic, and subjugate all wizards and muggles, a reference term that means non-magical people.

# What Gibbs Sampling is doing at an abstract level

Topic 1

wizard  
magic  
wand  
spell

Topic 2

movie  
book  
novel  
media

Topic 3

Harry  
Hermoine  
Ron  
Dumbledore

Topic 4

school  
teacher  
class  
prefect

Harry Potter is a series of fantasy novels written by British author J. K. Rowling. The novels chronicle the life of a young wizard, Harry Potter, and his friends Hermione Granger and Ron Weasley, all of whom are students at Hogwarts School of Witchcraft and Wizardry. The main story arc concerns Harry's struggle against Lord Voldemort, a dark wizard who intends to become immortal, overthrow the wizard governing body known as the Ministry of Magic, and subjugate all wizards and muggles, a reference term that means non-magical people.

# What Gibbs Sampling is doing at an abstract level

Topic 1

wizard  
magic  
wand  
spell

Topic 2

movie  
book  
novel  
media

Topic 3

Harry  
Hermoine  
Ron  
Dumbledore

Topic 4

school  
teacher  
class  
prefect

Harry Potter is a series of fantasy novels written by British author J. K. Rowling. The novels chronicle the life of a young wizard, Harry Potter, and his friends Hermione Granger and Ron Weasley, all of whom are students at Hogwarts School of Witchcraft and Wizardry. The main story arc concerns Harry's struggle against Lord Voldemort, a dark wizard who intends to become immortal, overthrow the wizard governing body known as the Ministry of Magic, and subjugate all wizards and muggles, a reference term that means non-magical people.

(In the real version, all words are assigned a topic!)

# What Gibbs Sampling is doing at an abstract level

Topic 1

wizard  
magic  
wand  
spell

Topic 2

movie  
book  
novel  
media

Topic 3

Harry  
Hermoine  
Ron  
Dumbledore

Topic 4

school  
teacher  
class  
prefect

Harry Potter is a series of fantasy novels written by British author J. K. Rowling. The novels chronicle the life of a young wizard, Harry Potter, and his friends Hermione Granger and Ron Weasley, all of whom are students at Hogwarts School of Witchcraft and Wizardry. The main story arc concerns Harry's struggle against Lord Voldemort, a dark wizard who intends to become immortal, overthrow the wizard governing body known as the Ministry of Magic, and subjugate all wizards and muggles, a reference term that means non-magical people.

# What Gibbs Sampling is doing at an abstract level

Topic 1

wizard  
magic  
wand  
spell

Topic 2

movie  
book  
novel  
media

Topic 3

Harry  
Hermoine  
Ron  
Dumbledore

Topic 4

school  
teacher  
class  
prefect

Harry Potter is a series of fantasy novels written by British author J. K. Rowling. The novels chronicle the life of a young wizard, Harry Potter, and his friends Hermione Granger and Ron Weasley, all of whom are students at Hogwarts School of Witchcraft and Wizardry. The main story arc concerns Harry's struggle against Lord Voldemort, a dark wizard who intends to become immortal, overthrow the wizard governing body known as the Ministry of Magic, and subjugate all wizards and muggles, a reference term that means non-magical people.

# What Gibbs Sampling is doing at an abstract level

Topic 1

wizard  
magic  
wand  
spell

Topic 2

movie  
book  
novel  
media

Topic 3

Harry  
Hermoine  
Ron  
Dumbledore

Topic 4

school  
teacher  
class  
prefect

Harry Potter is a series of fantasy novels written by British author J. K. Rowling. The novels chronicle the life of a young wizard, Harry Potter, and his friends Hermione Granger and Ron Weasley, all of whom are students at Hogwarts School of Witchcraft and Wizardry. The main story arc concerns Harry's struggle against Lord Voldemort, a dark wizard who intends to become immortal, overthrow the wizard governing body known as the Ministry of Magic, and subjugate all wizards and muggles, a reference term that means non-magical people.

(In the real version, all words are used to infer the word's new topic!)

# Gibbs Sampling Equation

$$\frac{n_{d,k} + \alpha_k}{\sum_i^K n_{d,i} + \alpha_i} \frac{v_{k,w_{d,n}} + \lambda_{w_{d,n}}}{\sum_i v_{k,i} + \lambda_i}$$

# Gibbs Sampling Equation

$$\frac{n_{d,k} + \alpha_k}{\sum_i^K n_{d,i} + \alpha_i} \frac{v_{k,w_{d,n}} + \lambda_{w_{d,n}}}{\sum_i v_{k,i} + \lambda_i}$$



How much this document  
likes topic k

# Gibbs Sampling Equation

$$\frac{n_{d,k} + \alpha_k}{\sum_i^K n_{d,i} + \alpha_i} \frac{v_{k,w_{d,n}} + \lambda_{w_{d,n}}}{\sum_i v_{k,i} + \lambda_i}$$


How much this document  
likes topic k

How much this topic likes word  
 $w_{d,n}$

# Gibbs Sampling Equation

Number of times document d  
uses topic k

$$\frac{n_{d,k} + \alpha_k}{\sum_i^K n_{d,i} + \alpha_i} \frac{v_{k,w_{d,n}} + \lambda_{w_{d,n}}}{\sum_i v_{k,i} + \lambda_i}$$

How much this document  
likes topic k

How much this topic likes word  
 $w_{d,n}$

# Gibbs Sampling Equation

Number of times document d uses topic k

$$\frac{n_{d,k} + \alpha_k}{\sum_i^K n_{d,i} + \alpha_i} \frac{v_{k,w_{d,n}} + \lambda_{w_{d,n}}}{\sum_i v_{k,i} + \lambda_i}$$

How much this document likes topic k

Number of times topic k uses word time  $w_{d,n}$

How much this topic likes word  $w_{d,n}$

# Gibbs Sampling Equation

Dirichlet parameter for the document to topic distribution

Number of times document d uses topic k

$$\frac{n_{d,k} + \alpha_k}{\sum_i^K n_{d,i} + \alpha_i} \frac{v_{k,w_{d,n}} + \lambda_{w_{d,n}}}{\sum_i v_{k,i} + \lambda_i}$$

How much this document likes topic k

Number of times topic k uses word time  $w_{d,n}$

How much this topic likes word  $w_{d,n}$

# Gibbs Sampling Equation

Dirichlet parameter for the document to topic distribution

Number of times document d uses topic k

$$\frac{n_{d,k} + \alpha_k}{\sum_i^K n_{d,i} + \alpha_i}$$

How much this document likes topic k

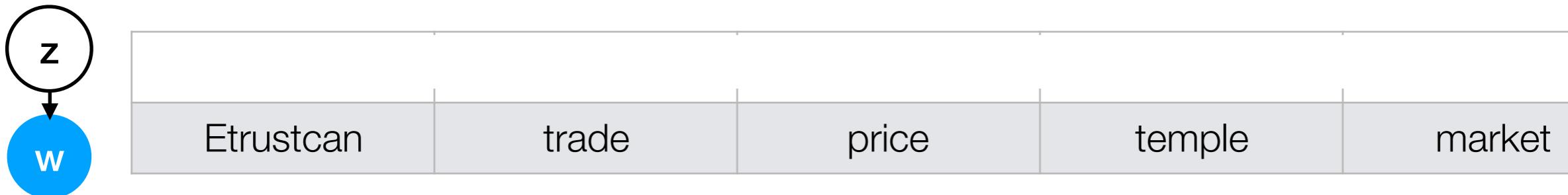
Dirichlet parameter for the topic to word distribution

Number of times topic k uses word time  $w_{d,n}$

$$\frac{v_{k,w_{d,n}} + \lambda_{w_{d,n}}}{\sum_i v_{k,i} + \lambda_i}$$

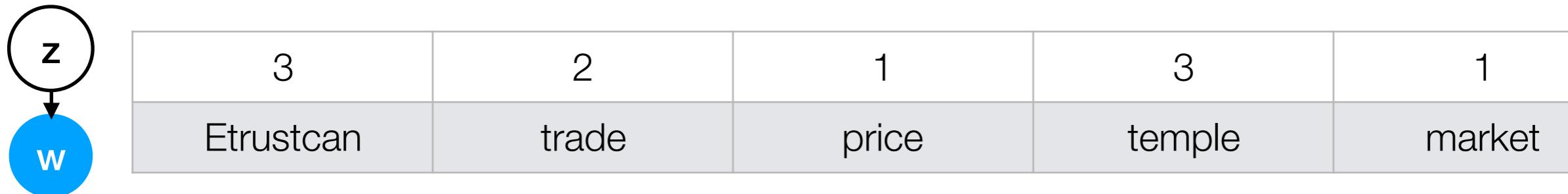
How much this topic likes word  $w_{d,n}$

# Let's do an example!



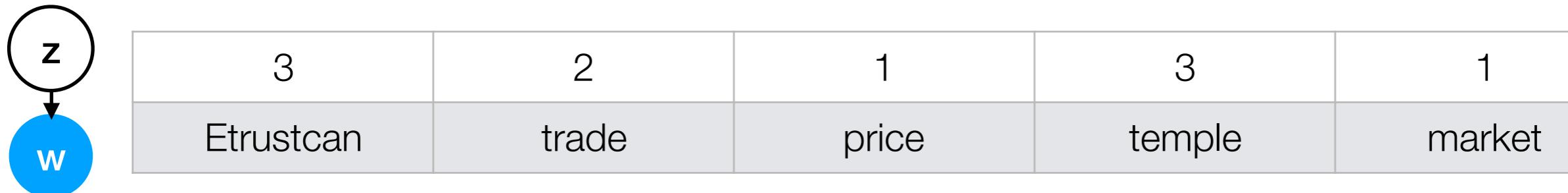
Start by assigning all words topics at random

# Let's do an example!

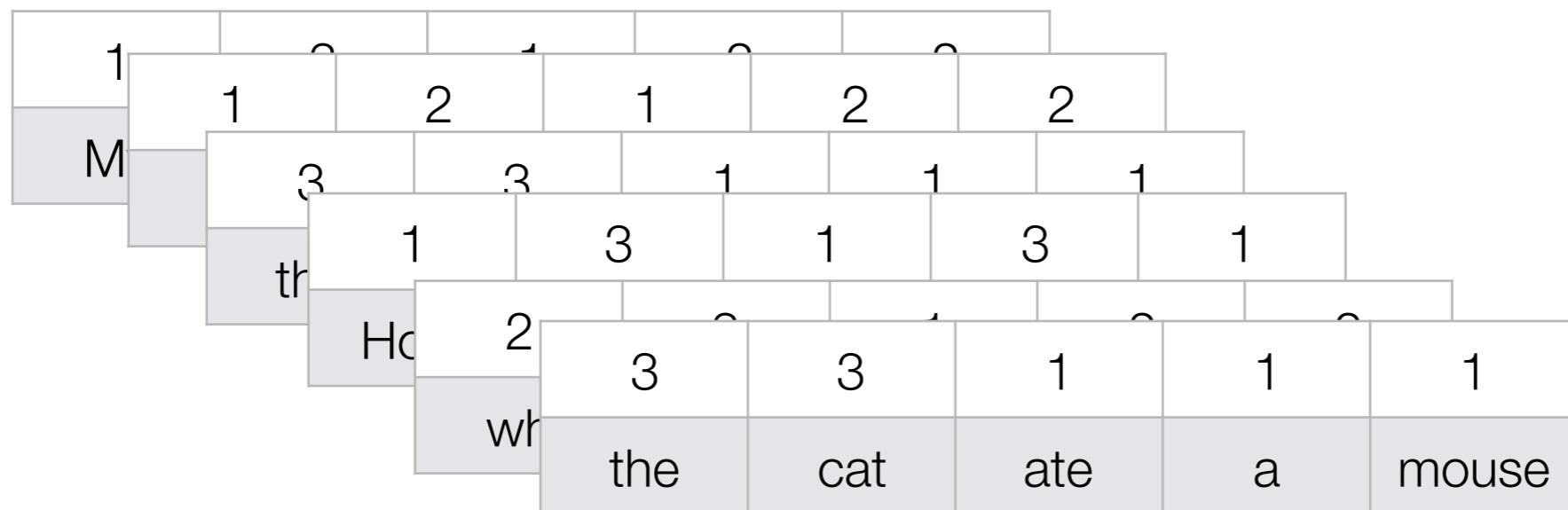


Start by assigning all words topics at random

Let's do an example!



Start by assigning all words topics at random



# Total Topic Counts

3	2	1	3	1
Etrustcan	trade	price	temple	market

Total counts from  
all documents

	<b>1</b>	<b>2</b>	<b>3</b>
Etrustcan	1	0	35
market	50	0	1
price	42	1	0
temple	0	0	20
trade	10	8	1
...			

# Total Topic Counts

3	2	1	3	1
Etrustcan	trade	price	temple	market

Total counts from  
all documents

	<b>1</b>	<b>2</b>	<b>3</b>
Etrustcan	1	0	35
market	50	0	1
price	42	1	0
temple	0	0	20
trade	10	8	1
...			

$$\frac{n_{d,k} + \alpha_k}{\sum_i^K n_{d,i} + \alpha_i} \frac{v_{k,w_{d,n}} + \lambda_{w_{d,n}}}{\sum_i v_{k,i} + \lambda_i}$$

# We want to sample this word

3	2	1	3	1
Etrustcan	trade	price	temple	market

	<b>1</b>	<b>2</b>	<b>3</b>
Etrustcan	1	0	35
market	50	0	1
price	42	1	0
temple	0	0	20
trade	10	8	1
...			

# Step 1: Unassign it and decrement its count

3	2	1	3	1
Etrustcan	trade	price	temple	market

	<b>1</b>	<b>2</b>	<b>3</b>
Etrustcan	1	0	35
market	50	0	1
price	42	1	0
temple	0	0	20
trade	10	8	1
...			

# Step 1: Unassign it and decrement its count

3	?	1	3	1
Etrustcan	trade	price	temple	market

	1	2	3
Etrustcan	1	0	35
market	50	0	1
price	42	1	0
temple	0	0	20
trade	10	8	1
...			

# Step 1: Unassign it and decrement its count

3	?	1	3	1
Etrustcan	trade	price	temple	market

	1	2	3
Etrustcan	1	0	35
market	50	0	1
price	42	1	0
temple	0	0	20
trade	10	8	1
...			

# Step 1: Unassign it and decrement its count

3	?	1	3	1
Etrustcan	trade	price	temple	market

	1	2	3
Etrustcan	1	0	35
market	50	0	1
price	42	1	0
temple	0	0	20
trade	10	7	1
...			

Step 2: Recompute the conditional distribution —  
How much does this document like each topic?

3	?	1	3	1
Etrustcan	trade	price	temple	market

Topic 1



Topic 2



Topic 3



Step 2: Recompute the conditional distribution —  
How much does this document like each topic?

3	?	1	3	1
Etrustcan	trade	price	temple	market

Topic 1



Topic 2



Topic 3



$$\frac{n_{d,k} + \alpha_k}{\sum_i^K n_{d,i} + \alpha_i} \frac{v_{k,w_{d,n}} + \lambda_{w_{d,n}}}{\sum_i v_{k,i} + \lambda_i}$$

# Step 3: How much does each topic like this word?

3	?	1	3	1
Etrustcan	trade	price	temple	market

Topic 1



Topic 2



Topic 3



	1	2	3
trade	10	7	1

# Step 3: How much does each topic like this word?

3	?	1	3	1
Etrustcan	trade	price	temple	market

Topic 1



Topic 2



Topic 3



	1	2	3
trade	10	7	1

$$\frac{n_{d,k} + \alpha_k}{\sum_i^K n_{d,i} + \alpha_i} \frac{v_{k,w_{d,n}} + \lambda_{w_{d,n}}}{\sum_i v_{k,i} + \lambda_i}$$

# Step 3: How much does each topic like this word?

3	?	1	3	1
Etrustcan	trade	price	temple	market

Topic 1



Topic 2



Topic 3

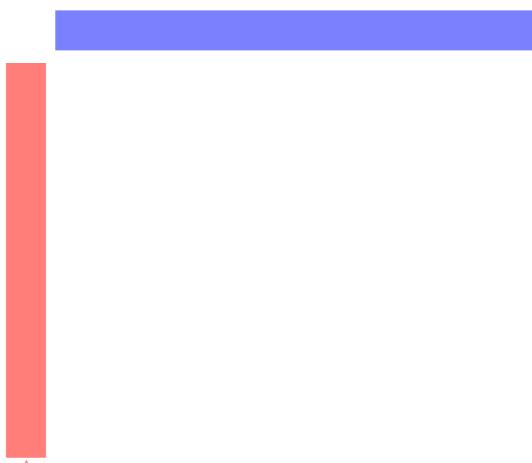


	1	2	3
trade	10	7	1

# Step 3: How much does each topic like this word?

3	?	1	3	1
Etrustcan	trade	price	temple	market

Topic 1



Topic 2



Topic 3

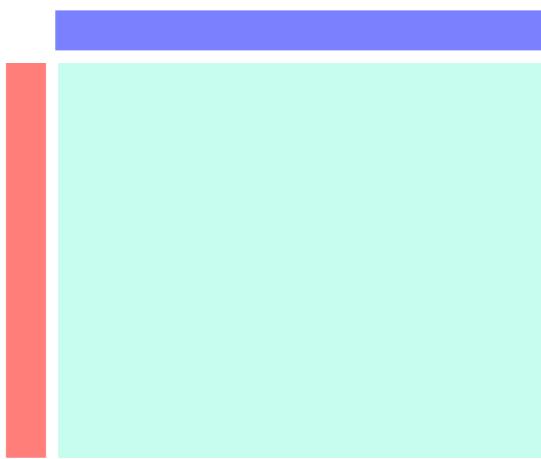


	1	2	3
trade	10	7	1

# Step 3: How much does each topic like this word?

3	?	1	3	1
Etrustcan	trade	price	temple	market

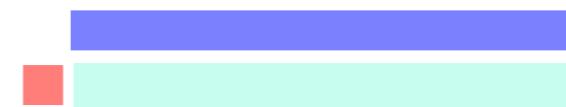
Topic 1



Topic 2



Topic 3

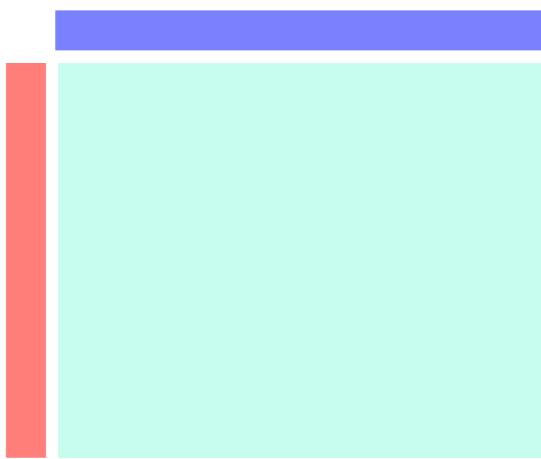


	1	2	3
trade	10	7	1

# Step 4: Sample a new topic for that word, proportional to the area

3	?	1	3	1
Etrustcan	trade	price	temple	market

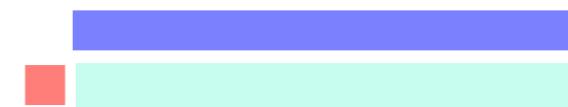
Topic 1



Topic 2



Topic 3

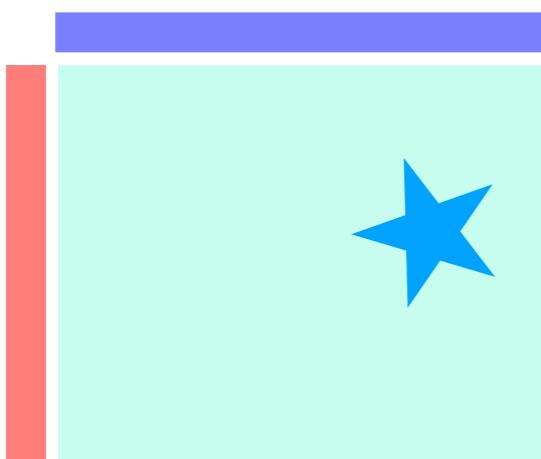


	1	2	3
trade	10	7	1

# Step 4: Sample a new topic for that word, proportional to the area

3	?	1	3	1
Etrustcan	trade	price	temple	market

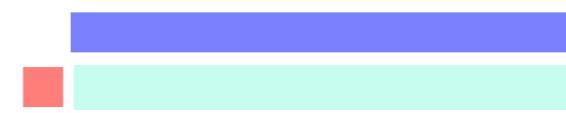
Topic 1



Topic 2



Topic 3



trade	10	7	1
	1	2	3

# Step 5: Update the counts

3	?	1	3	1
Etrustcan	trade	price	temple	market

	1	2	3
Etrustcan	1	0	35
market	50	0	1
price	42	1	0
temple	0	0	20
trade	10	7	1
...			

# Step 5: Update the counts

3	?	1	3	1
Etrustcan	trade	price	temple	market

	1	2	3
Etrustcan	1	0	35
market	50	0	1
price	42	1	0
temple	0	0	20
trade	10	7	1
...			

# Step 5: Update the counts

3	?	1	3	1
Etrustcan	trade	price	temple	market

	1	2	3
Etrustcan	1	0	35
market	50	0	1
price	42	1	0
temple	0	0	20
trade	11	7	1
...			

# Step 5: Update the counts

3	?	1	3	1
Etrustcan	trade	price	temple	market

	1	2	3
Etrustcan	1	0	35
market	50	0	1
price	42	1	0
temple	0	0	20
trade	11	7	1
...			

# Step 5: Update the counts

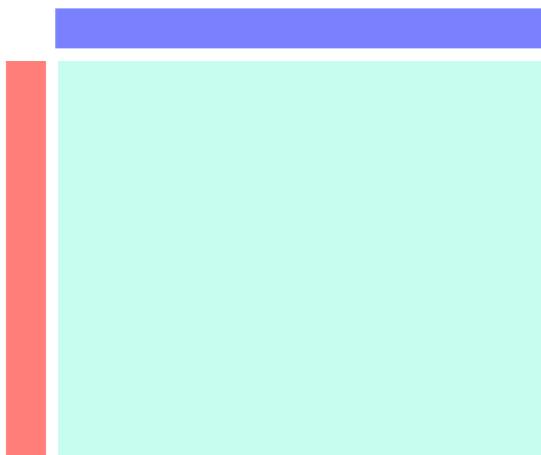
3	<b>1</b>	1	3	1
Etrustcan	trade	price	temple	market

	<b>1</b>	<b>2</b>	<b>3</b>
Etrustcan	1	0	35
market	50	0	1
price	42	1	0
temple	0	0	20
trade	<b>11</b>	7	1
...			

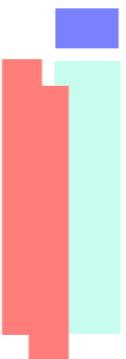
# Step 4: Sample a new topic for that word, proportional to the area

3	<b>1</b>	1	3	1
Etrustcan	trade	price	temple	market

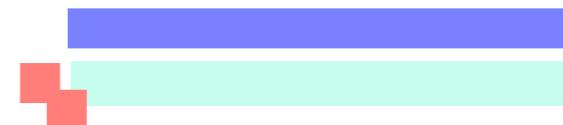
Topic 1



Topic 2



Topic 3

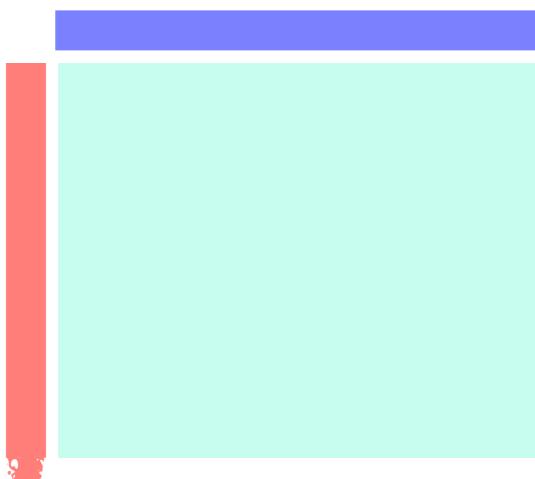


	<b>1</b>	<b>2</b>	<b>3</b>
trade	11	7	1

# Step 4: Sample a new topic for that word, proportional to the area

3	<b>1</b>	1	3	1
Etrustcan	trade	price	temple	market

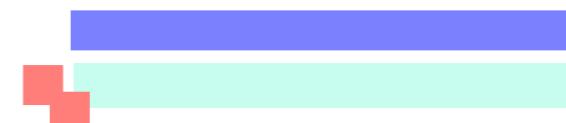
Topic 1



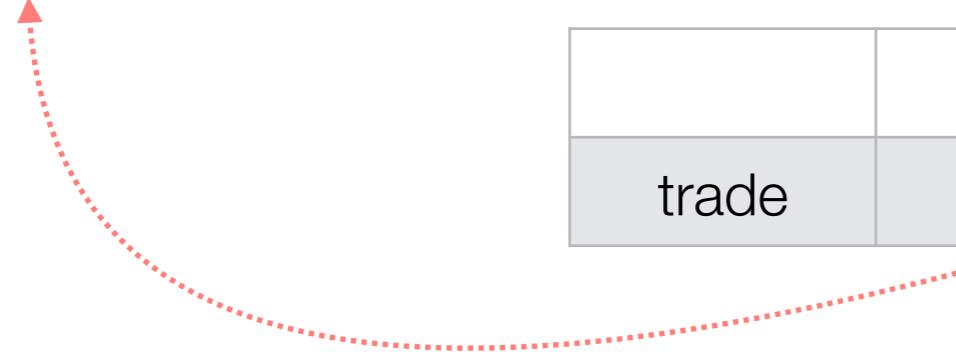
Topic 2



Topic 3



	<b>1</b>	<b>2</b>	<b>3</b>
trade	11	7	1



# Gibbs Sampling Algorithm

# Gibbs Sampling Algorithm

- For each iteration i:

# Gibbs Sampling Algorithm

- For each iteration  $i$ :
  - For each document  $d$  and word  $n$  currently assigned to  $z_{old}$

# Gibbs Sampling Algorithm

- For each iteration  $i$ :
  - For each document  $d$  and word  $n$  currently assigned to  $z_{old}$
  - Decrement  $n_{d,z_{old}}$  and  $v_{z_{old},w_{d,n}}$

# Gibbs Sampling Algorithm

- For each iteration  $i$ :
  - For each document  $d$  and word  $n$  currently assigned to  $z_{old}$ 
    - Decrement  $n_{d,z_{old}}$  and  $v_{z_{old},w_{d,n}}$
    - Sample  $z_{new} = k$  with probability proportional to

# Gibbs Sampling Algorithm

- For each iteration  $i$ :
  - For each document  $d$  and word  $n$  currently assigned to  $z_{old}$
  - Decrement  $n_{d,z_{old}}$  and  $v_{z_{old},w_{d,n}}$
  - Sample  $z_{new} = k$  with probability proportional to

$$\frac{n_{d,k} + \alpha_k}{\sum_i^K n_{d,i} + \alpha_i} \frac{v_{k,w_{d,n}} + \lambda_{w_{d,n}}}{\sum_i v_{k,i} + \lambda_i}$$

# Gibbs Sampling Algorithm

- For each iteration  $i$ :
  - For each document  $d$  and word  $n$  currently assigned to  $z_{old}$ 
    - Decrement  $n_{d,z_{old}}$  and  $v_{z_{old},w_{d,n}}$
    - Sample  $z_{new} = k$  with probability proportional to
$$\frac{n_{d,k} + \alpha_k}{\sum_i^K n_{d,i} + \alpha_i} \frac{v_{k,w_{d,n}} + \lambda_{w_{d,n}}}{\sum_i v_{k,i} + \lambda_i}$$
    - Increment  $n_{d,z_{new}}$  and  $v_{z_{new},w_{d,n}}$

# Homework 4

- You'll be implementing the Gibbs Sampling updates part of LDA!
- We've given you a bunch of skeleton code; your part is ~10 lines, very similar to what we covered today
- 90% thinking; 10% coding

# Gibbs Sampling for the Uninitiated

Philip Resnik

Department of Linguistics and  
Institute for Advanced Computer Studies  
University of Maryland  
College Park, MD 20742 USA  
`resnik AT umd.edu`

Eric Hardisty

Department of Computer Science and  
Institute for Advanced Computer Studies  
University of Maryland  
College Park, MD 20742 USA  
`hardisty AT cs.umd.edu`

## Abstract

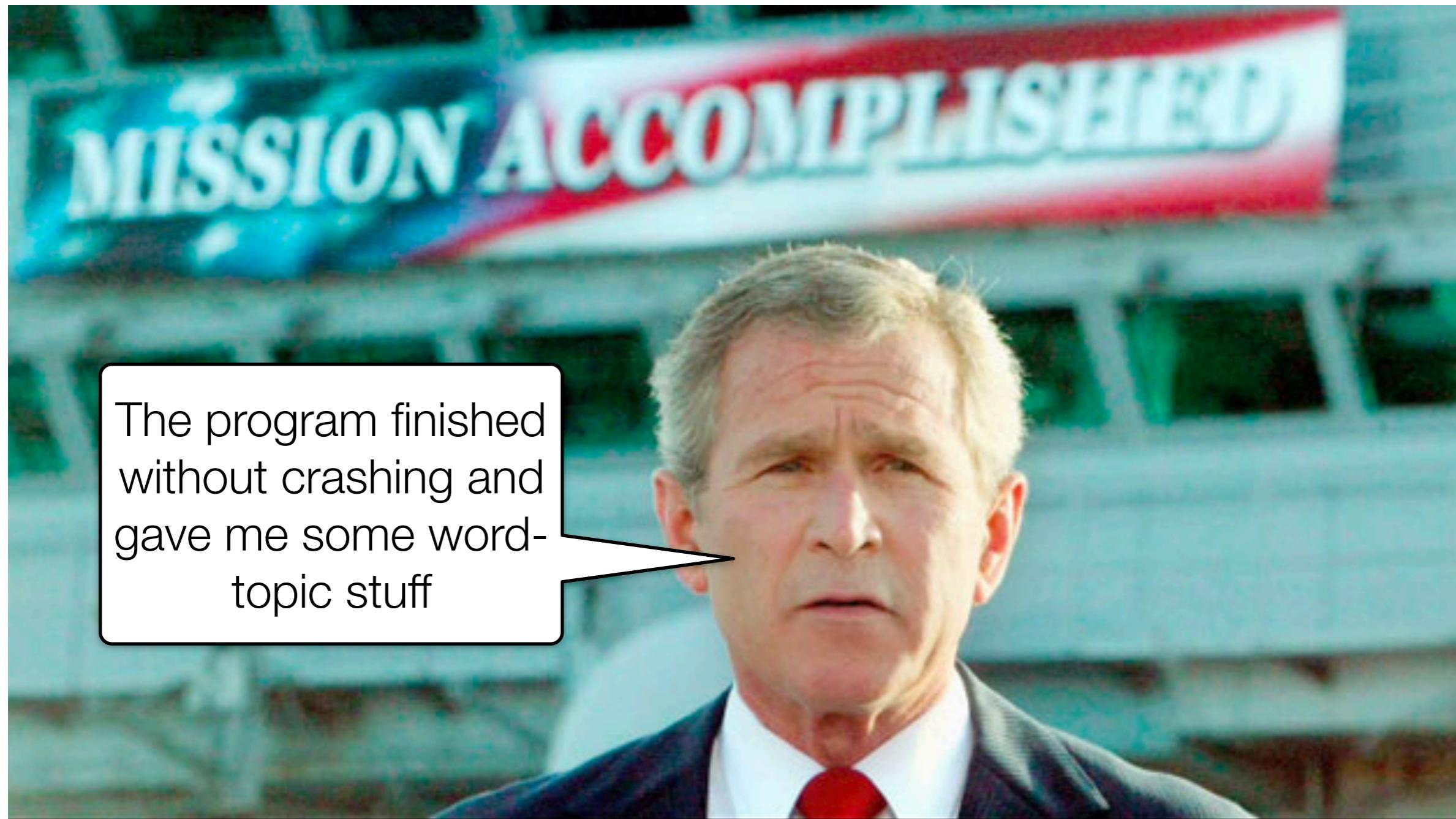
This document is intended for computer scientists who would like to try out a Markov Chain Monte Carlo (MCMC) technique, particularly in order to do inference with Bayesian models on problems related to text processing. We try to keep theory to the absolute minimum needed, though we work through the details much more explicitly than you usually see even in “introductory” explanations. That means we’ve attempted to be ridiculously explicit in our exposition and notation.

After providing the reasons and reasoning behind Gibbs sampling (and at least nodding our heads in the direction of theory), we work through an example application in detail—the derivation of a Gibbs sampler for a Naïve Bayes model. Along with the example, we discuss some practical implementation issues, including the integrating out of continuous parameters when possible. We conclude with some pointers to literature that we’ve found to be somewhat more friendly to uninitiated readers.

## 1 Introduction

Markov Chain Monte Carlo (MCMC) techniques like Gibbs sampling provide a principled way to approximate the value of an integral.

# How do we know if the topics we learned are good?



The program finished without crashing and gave me some word-topic stuff

# Option 1: Test on held out data

# Option 1: Test on held out data

- Use the generative model to measure how the probability of generating unseen documents
  - $\text{Log}(\text{prob. new docs}) \rightarrow \text{Log likelihood}$

# Option 1: Test on held out data

- Use the generative model to measure how the probability of generating unseen documents
  - $\text{Log}(\text{prob. new docs}) \rightarrow \text{Log likelihood}$
- Good:
  - Wallach (2009) show how to efficiently approximate this

# Option 1: Test on held out data

- Use the generative model to measure how the probability of generating unseen documents
  - $\text{Log}(\text{prob. new docs}) \rightarrow \text{Log likelihood}$
- Good:
  - Wallach (2009) show how to efficiently approximate this
- Bad:
  - Chang (2009) show that likelihood is not correlated with human judgements of goodness

# Option 2: Intruder Test

- Intrusion Detection (Blei et al., 2009)

# Option 2: Intruder Test

- Intrusion Detection (Blei et al., 2009)
- Show a topic's words + one intruder word
  - Intruder = high prob for other topic, low prob current topic
  - Ask user to identify intruder

cat  
dog  
mouse  
computer  
gorilla

# Option 2: Intruder Test

- Intrusion Detection (Blei et al., 2009)
- Show a topic's words + one intruder word
  - Intruder = high prob for other topic, low prob current topic
  - Ask user to identify intruder

cat  
dog  
mouse  
computer  
gorilla

phone  
android  
apple  
tv  
ipod

# Option 2: Intruder Test

- Intrusion Detection (Blei et al., 2009)
- Show a topic's words + one intruder word
  - Intruder = high prob for other topic, low prob current topic
  - Ask user to identify intruder
- Good: Correlates with human judgements!

cat  
dog  
mouse  
computer  
gorilla

phone  
android  
apple  
tv  
ipod

# Option 2: Intruder Test

- Intrusion Detection (Blei et al., 2009)
- Show a topic's words + one intruder word
  - Intruder = high prob for other topic, low prob current topic
  - Ask user to identify intruder

cat  
dog  
mouse  
computer  
gorilla

phone  
android  
apple  
tv  
ipod

- Good: Correlates with human judgements!
- Bad: Can't be done automatically

# Many other metrics

- Qualitative
  - Expert judgements
  - Word Clouds
  - Visualizing word-topic assignments
- Quantitative
  - Topic Coherence: Better models have words that are highly related according to some external source (Newman, 2010; Mimno et al., 2011)

# Fancier metrics are also possible

## 8. CONCLUSION

We proposed a unifying framework that span the space of all known coherence measures and allows to combine all main ideas in the context of coherence quantification. We evaluated 237 912 coherence measures on six different benchmarks for topic coherence—to the best of our knowledge, this is the largest number of benchmarks used for topic coherences so far. We introduced coherence measures from

Röder et al. (2015)

# Summarizing what we learned about topic models



# Topic Modeling in Practice

# Topic Modeling in Practice

- Number of topics have to be predetermined
  - Finding the right number of topics can be hard
  - Bayesian model selection – enumerate all  $K$
  - Non-parametric topic models – more complicated

# Topic Modeling in Practice

- Number of topics have to be predetermined
  - Finding the right number of topics can be hard
  - Bayesian model selection – enumerate all  $K$
  - Non-parametric topic models – more complicated
- Topics need to be interpreted
  - Reading the word distribution can be hard
  - Methods to generate labels for topics (Mei et al., 2007)

# Topic Modeling in Practice

- Number of topics have to be predetermined
  - Finding the right number of topics can be hard
  - Bayesian model selection – enumerate all  $K$
  - Non-parametric topic models – more complicated
- Topics need to be interpreted
  - Reading the word distribution can be hard
  - Methods to generate labels for topics (Mei et al., 2007)
- Evaluation is subjective
  - Likelihood of held-out data, quality of topics, quality of document clusters, human judgments, ...

# Topic Modeling in Practice (Cont.)

# Topic Modeling in Practice (Cont.)

- In general, it is a good tool for exploratory analysis
  - Understanding qualitatively what's in the corpus.

# Topic Modeling in Practice (Cont.)

- In general, it is a good tool for exploratory analysis
  - Understanding qualitatively what's in the corpus.
- Not the first choice if you have a particular task in mind
  - Classification: try SVM first
  - Labeling: try CRF
  - Clustering: try k-means, spectral clustering, etc.

# Topic Modeling in Practice (Cont.)

- In general, it is a good tool for exploratory analysis
  - Understanding qualitatively what's in the corpus.
- Not the first choice if you have a particular task in mind
  - Classification: try SVM first
  - Labeling: try CRF
  - Clustering: try k-means, spectral clustering, etc.
- But can provide “deep” features for those tasks

# Topic Modeling in Practice (Cont.)

- In general, it is a good tool for exploratory analysis
  - Understanding qualitatively what's in the corpus.
- Not the first choice if you have a particular task in mind
  - Classification: try SVM first
  - Labeling: try CRF
  - Clustering: try k-means, spectral clustering, etc.
- But can provide “deep” features for those tasks
- Works well when few training examples are available

# Topic Modeling in Practice (Cont.)

- In general, it is a good tool for exploratory analysis
  - Understanding qualitatively what's in the corpus.
- Not the first choice if you have a particular task in mind
  - Classification: try SVM first
  - Labeling: try CRF
  - Clustering: try k-means, spectral clustering, etc.
- But can provide “deep” features for those tasks
- Works well when few training examples are available
- Extremely powerful when contextual variables are observed and concerned with
  - Very easy to extend, by adding variables into the Bayesian network (graphical structure)

# Things You Don't Know About LDA

# Things You Don't Know About LDA

- There are latent requirements of your data...

# Things You Don't Know About LDA

- There are latent requirements of your data...
- Do not apply LDA directly if
  - You have too few documents
  - Your documents are too short
  - Your goal is to find smallish, tight clusters
  - The co-occurrence assumption doesn't hold in your data

# Things You Don't Know About LDA

- There are latent requirements of your data...
- Do not apply LDA directly if
  - You have too few documents
  - Your documents are too short
  - Your goal is to find smallish, tight clusters
  - The co-occurrence assumption doesn't hold in your data
- Try to leverage meta-data or context information to improve the performance of topic modeling
  - Authorship
  - Time, geographic location
  - Networks of documents and words
  - User-guidance, ...

# Resources of Topic Modeling

- Topic Modeling Bibliography
  - Collected by David Mimno: <https://mimno.infosci.cornell.edu/topics.html>
- Toolkits:
  - LDA-C: command-line version from David Blei
  - LDA++: Easy-to-use C++ implementation
  - MALLET: Java package, with good support of text mining in general
  - Gensim: Python package for on-line topic modeling (*not so good*) and wraps MALLET in a python interface (good!)
  - Mahout: Machine learning toolkit adaptable on Hadoop
- Topic modeling in digital humanities
  - <http://mith.umd.edu/topic-modeling-in-the-humanities-an-overview/>
  - <http://journalofdigitalhumanities.org/2-1/words-alone-by-benjamin-m-schmidt/>



# What you should know

# What you should know

- Unsupervised learning can help you make sense of a large pile of text

# What you should know

- Unsupervised learning can help you make sense of a large pile of text
  - But supervised learning is almost always better

# What you should know

- Unsupervised learning can help you make sense of a large pile of text
  - But supervised learning is almost always better
- Brown clustering can help you learn categories of words
  - and the brown vectors are useful representations

# What you should know

- Unsupervised learning can help you make sense of a large pile of text
  - But supervised learning is almost always better
- Brown clustering can help you learn categories of words — and the brown vectors are useful representations
- Topic modeling is ~~magic~~ a powerful technique for learning broad themes in text

# What you should know

- Unsupervised learning can help you make sense of a large pile of text
  - But supervised learning is almost always better
- Brown clustering can help you learn categories of words — and the brown vectors are useful representations
- Topic modeling is ~~magic~~ a powerful technique for learning broad themes in text
  - But is difficult to evaluate—don't trust the output without looking at it and using it for something!