TEXT CLASSIFICATION

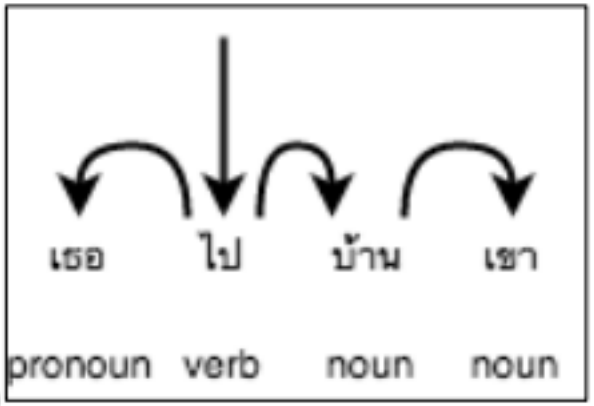
Intent, topic, sentiment, etc.

Midterm

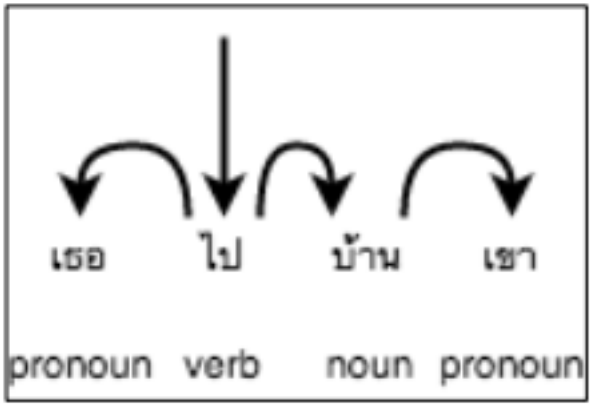
- Ambiguity

15. เธอ ไป บ้าน เขา

a)



b)

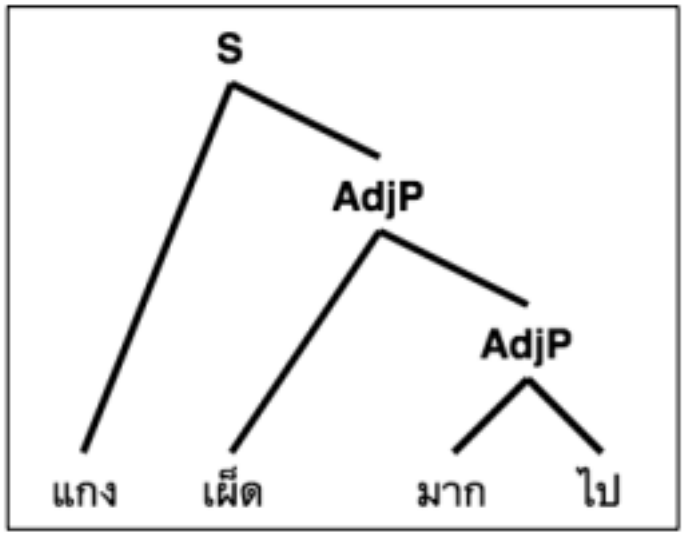


Midterm

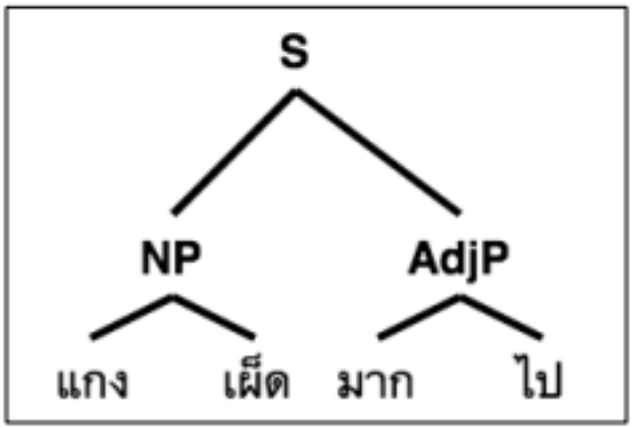
- Ambiguity

16. แกง เผ็ด มาก ไป

a)

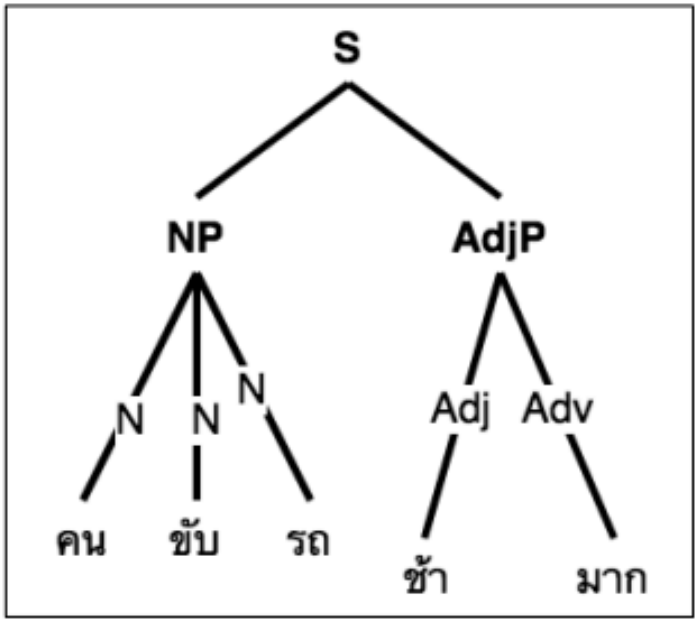


b)

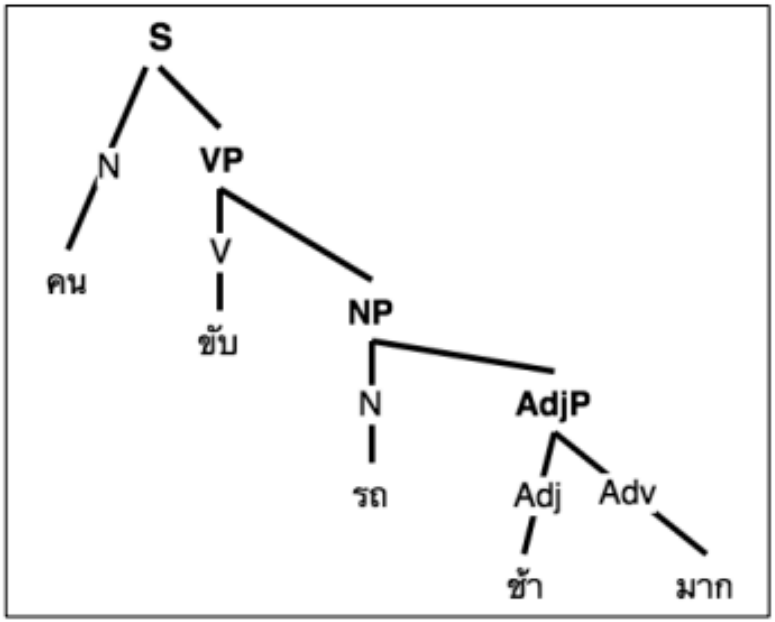


17. **คน ขับ รถ ช้า มาก**

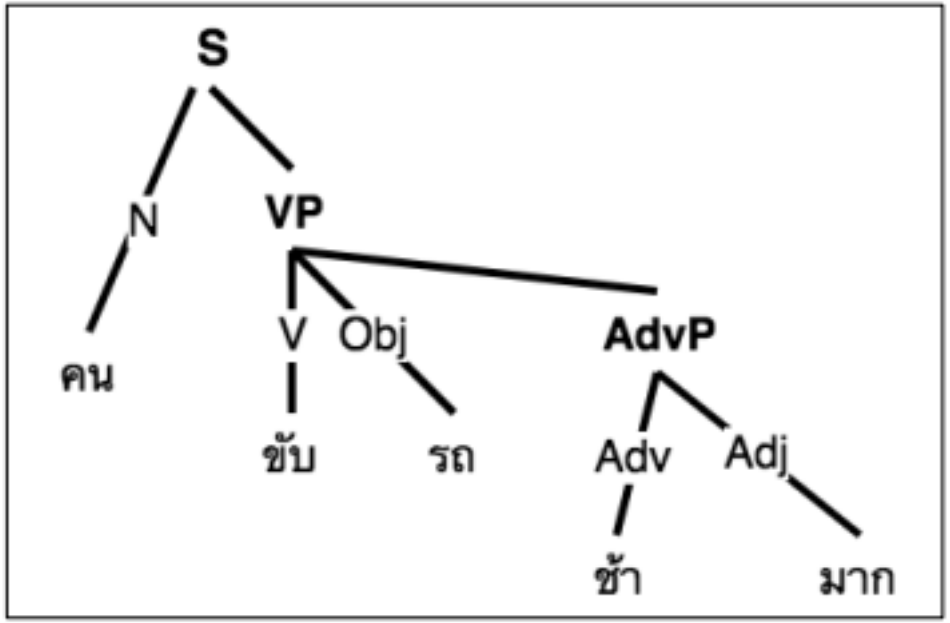
a)



b)



c)

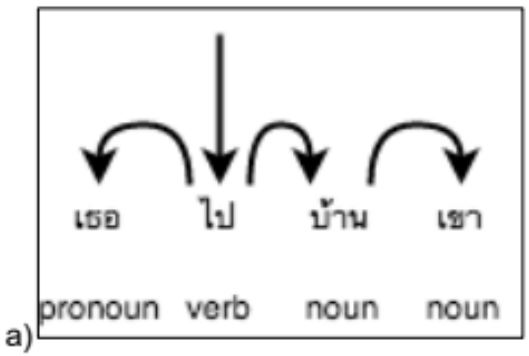


Error propagation and NLP systems

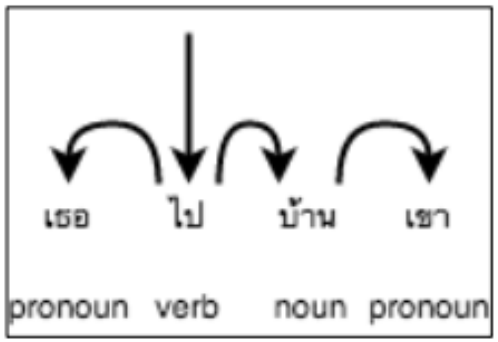
- Typical flow of NLP systems

Pipeline analysis

15. เธอ ไป บ้าน เขา



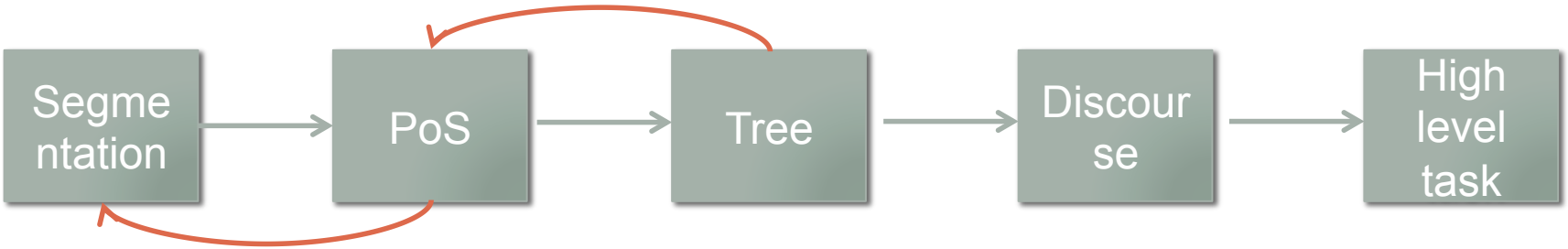
a)



b)

Error propagation and NLP systems

- Use higher level info to help lower level
  - Iterative process



Joint analysis

Error propagation and NLP systems

- Do the task together

Joint analysis

Error propagation and NLP systems

- One single task
  - Input closest to the input form – characters, words

High level task with low level structures as input

End-to-end analysis

Dropped words recovery

- Some sentence are ambiguous because of possibility of dropped words

เธอ ไป บ้าน [ของ] เขา

แกง [ที่] เผ็ด มาก ไป

คน ขับ รถ [มา] ช้า มาก

- One solution (Pipeline processing)
  - Separate into two parts:
    - Dropped word detection – determine the location
    - Dropped word prediction – determine what is missing

Dropped word detection

- เธอ ไป บ้าน เขา
- 0 0 1 0

- Similar to word segmentation. A detection task.
  - Some recurrent network should be good for this
  - Things to consider
    - Are the words already segmented in the corpus?
      - Words representation?
        - Outside corpora?
    - Use PoS tags?
      - Does the training sentence need PoS tags?
      - PoS embeddings?

- Special mention: use tree structure to calculate the probability. If there are two trees that have similar probability there should be dropped words

Dropped word prediction

- เธอ ไป บ้าน [ของ] เขา

- Context predict the middle word -> CBOW structure
- Or previous words predict next word -> n-gram
  - CBOW should be more powerful (bigger context), but both answers are accepted

End-to-end analysis

- เธอ ไป บ้าน [-X-] เขา

- X can be a proper word or blank
  - Vocabulary = N+1
  - Skip the detection task

Assumptions

- This only handle one missing word
  - How to handle multiple missing words in a row?
    - Predict phrases?
- Discourse analysis
  - Previous solution does not allow discourse
    - Can expand to discourse by expanding the context to be bigger than a sentence
- Types of dropped words considered

Corpora

- What kind of text?
  - Pantip, twitter, wikipedia, textbook, …
- Extra: How to actually construct the training corpora?
  - Painfully look at the sentences and determine dropped words
    - or
  - Look at standard sentences and determine which words can be dropped

Wongnai Challenge

- Predict star rating from review text

output

1 check-in

ร้องเรียน

ก๊วยเตี๋ยวอร่อย ราคาถูก นั่งทานในห้องแอร์

เมนูเด็ด: บะหมี่ต้มยำหมูแดง

input

ส่วนตัวชอบก๋วยเตี๋ยวต้มยำมากค่ะ รสชาติอร่อยไม่ต้องปรุงเลย แต่ซุปเปอร์ตีนไก่ใส่ถั่วงอกมาด้วย เหมือนเป็นเกาเหลาตีนไก่มากกว่าซุปเปอร์ตีนไก่ รสชาติก็ยังไม่กลมกล่อมเท่าก๋วยเตี๋ยวต้มยำ ตีนไก่เปื่อยดี ทานง่าย

#	△priv	Team Name	Kernel	Team Members	Score	Entries	Last
1	—	**blead**			0.59139	30	9h
2	—	**Danupat Kns**			0.58333	21	6h
3	▼1	**Phizaz**			0.58172	4	9d
4	▲1	**Attawit Chaiyaroj**			0.57848	14	3h
5	▼4	**Chal**			0.57633	42	1h
6	▼5	**James R.**			0.57419	32	13m
7	▼6	**Khajornal Anantanit**			0.57258	60	7h
8	▲1	**burin**			0.57258	44	14m
9	▼1	**Nattasit**			0.56774	4	5h
10	▲5	**Wannita**			0.56666	18	6h

Public

1	—	**blead**			0.58138	30	9h
2	—	**Danupat Kns**			0.56919	21	6h
3	▲1	**Attawit Chaiyaroj**			0.56297	14	3h
4	▼1	**Phizaz**			0.56182	4	9d
5	▲5	**Wannita**			0.55789	18	6h
6	▲6	**ChanatipSaetia**			0.55306	5	1h
7	▲1	**burin**			0.55284	44	14m
8	▲6	**AtomicBoyZ**			0.55192	11	17h
9	▼4	**Chal**			0.55192	42	1h
10	▼1	**Nattasit**			0.55123	4	5h

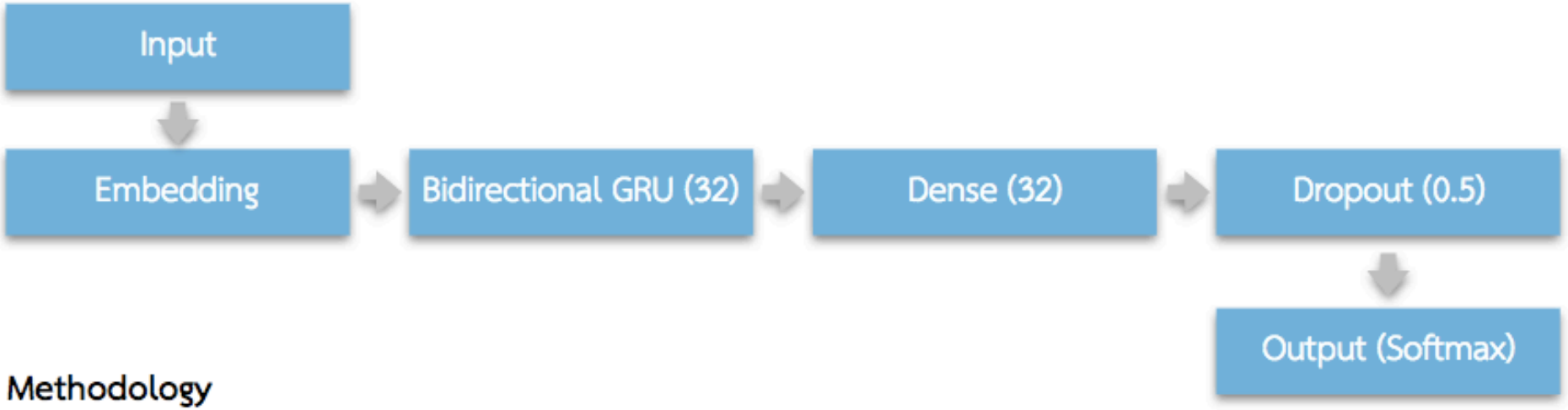
Private

Cute ideas

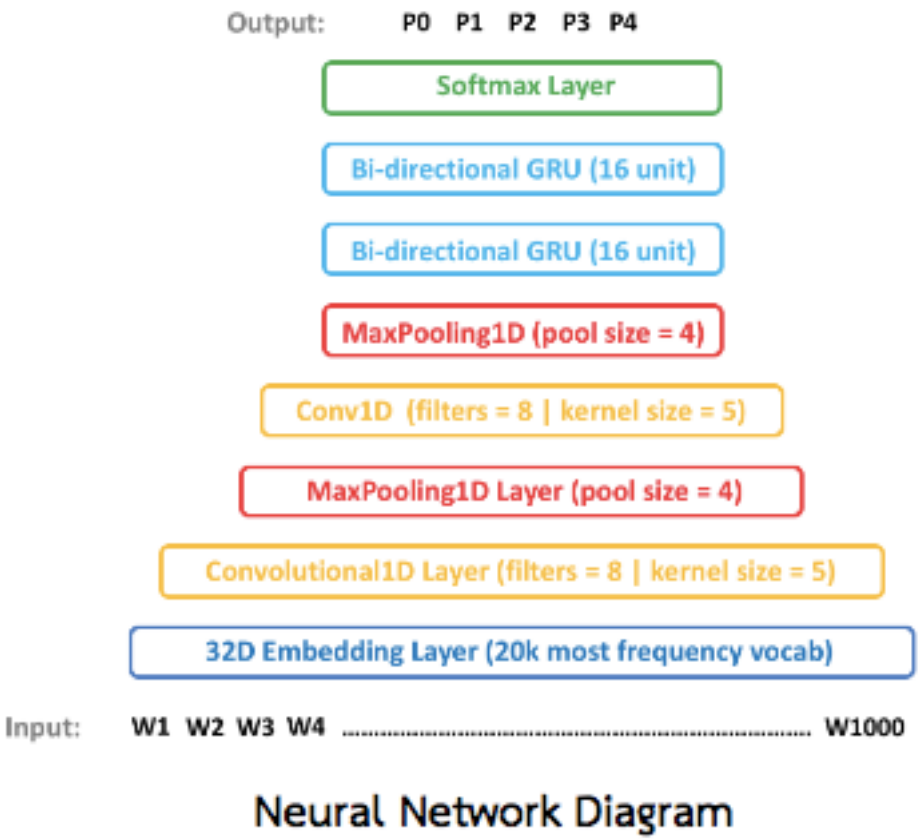
- Post-processing
  - ให้3ดาว
- Fasttext just got better
  - Trained on webcrawl instead of wiki
  - Tokenized using ICU
  - Character n-gram model to handle OOV
- Thai2vec
  - trained on thai wiki https://github.com/cstorm125/thai2vec
  - thaiNLP things https://github.com/PyThaiNLP/
- One person tried both with no significant difference (same amount of OOV)

3rd best model

- 8 words threshold for unk
- Pre-train weights from homework but adapted to task



2nd best model



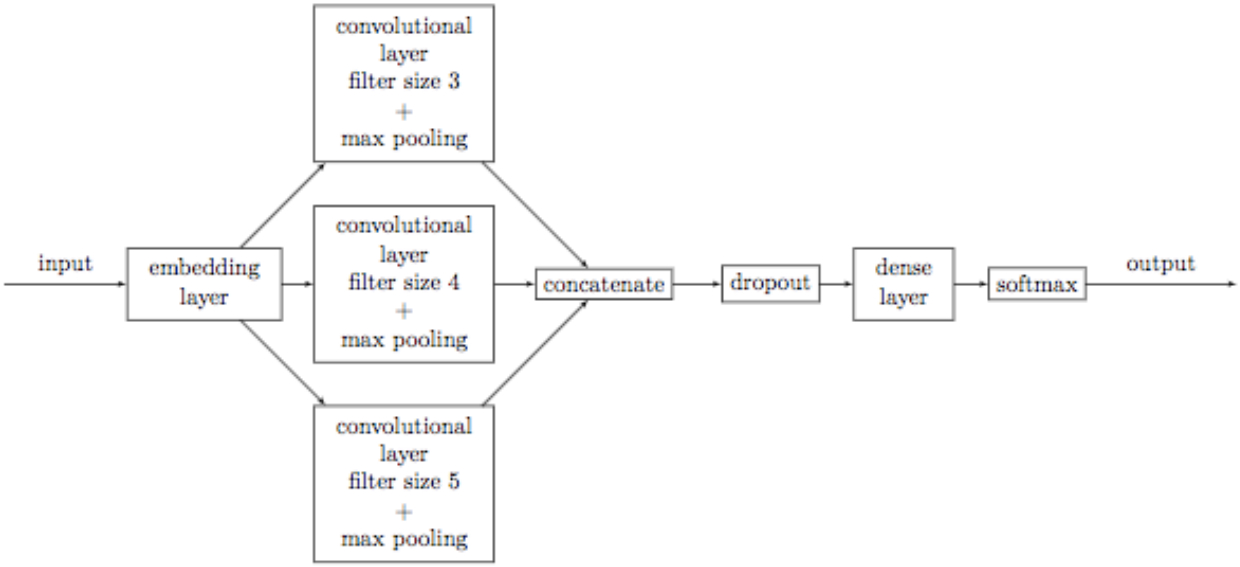
Neural Network Diagram

Filter unknowns with freq = 5
20% dropout every layer

No pre-trained vectors (?)

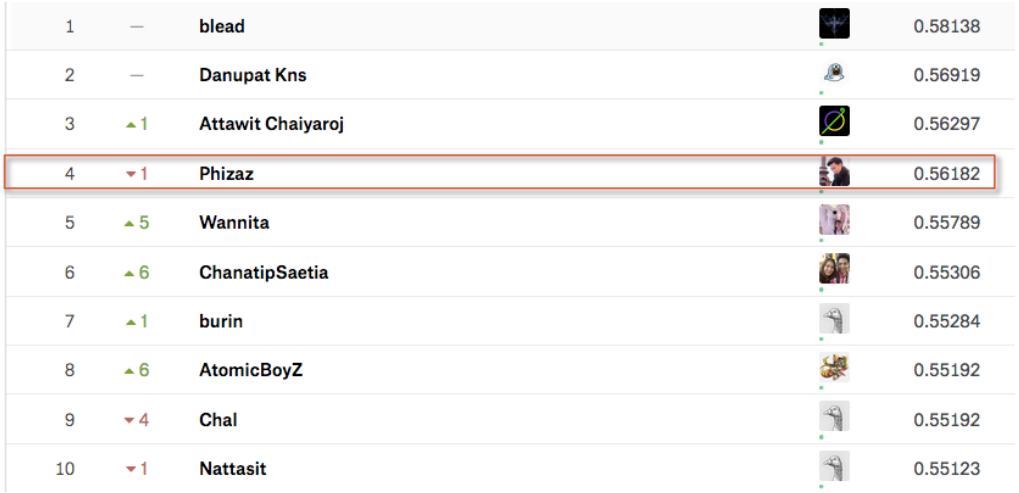
Best model

- shallow-and-wide CNN (https://arxiv.org/abs/1510.03820)
  - People should read papers!
- Thai2vec
  - Further adapted to the task



Character model (end-to-end)

- Char embedding
- 3 GRU layers
- 1000 character segment
  - Vote if longer than 1000



1	—	blead	0.58138
2	—	Danupat Kns	0.56919
3	▲1	Attawit Chaiyaroj	0.56297
4	▼1	Phizaz	0.56182
5	▲5	Wannita	0.55789
6	▲6	ChanatipSaetia	0.55306
7	▲1	burin	0.55284
8	▲6	AtomicBoyZ	0.55192
9	▼4	Chal	0.55192
10	▼1	Nattasit	0.55123

- Seems like a lucky random seed…

Yelp reviews

Duyu Tang Document Modeling with Gated Recurrent Neural Network for Sentiment Classification, 2015 http://aclweb.org/anthology/D15-1167

Wongnai challenge top model Accuracy 0.5844

Corpus	#docs	#s/d	#w/d	$\|V\|$	#class	Class Distribution
Yelp 2013	335,018	8.90	151.6	211,245	5	.09/.09/.14/.33/.36
Yelp 2014	1,125,457	9.22	156.9	476,191	5	.10/.09/.15/.30/.36
Yelp 2015	1,569,264	8.97	151.9	612,636	5	.10/.09/.14/.30/.37
IMDB	348,415	14.02	325.6	115,831	10	.07/.04/.05/.05/.08/.11/.15/.17/.12/.18

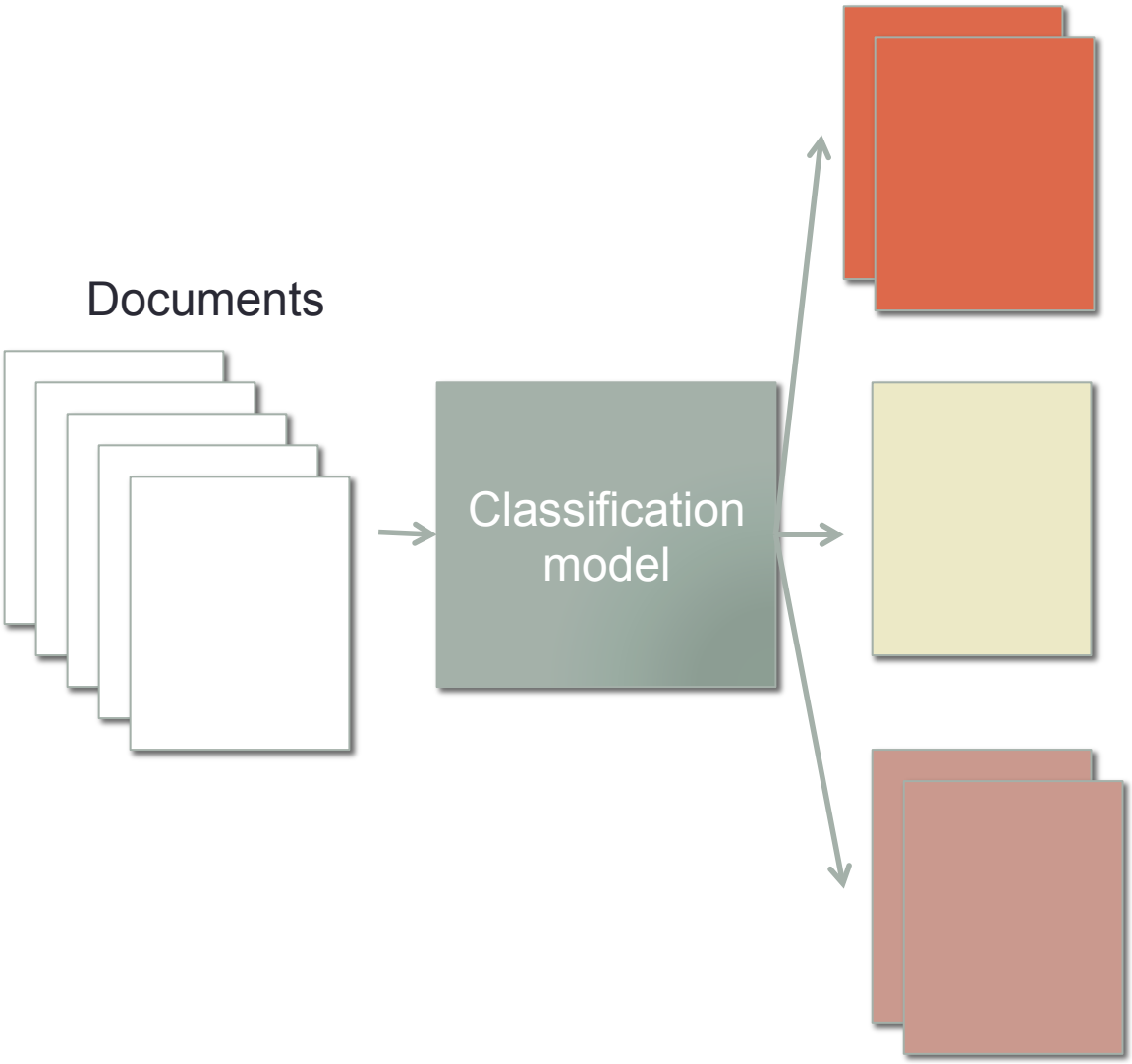
	Yelp 2013		Yelp 2014		Yelp 2015		IMDB	
	Accuracy	MSE	Accuracy	MSE	Accuracy	MSE	Accuracy	MSE
Majority	0.356	3.06	0.361	3.28	0.369	3.30	0.179	17.46
SVM + Unigrams	0.589	0.79	0.600	0.78	0.611	0.75	0.399	4.23
SVM + Bigrams	0.576	0.75	0.616	0.65	0.624	0.63	0.409	3.74
SVM + TextFeatures	0.598	0.68	0.618	0.63	0.624	0.60	0.405	3.56
SVM + AverageSG	0.543	1.11	0.557	1.08	0.568	1.04	0.319	5.57
SVM + SSWE	0.535	1.12	0.543	1.13	0.554	1.11	0.262	9.16
JMARS	N/A	–	N/A	–	N/A	–	N/A	4.97
Paragraph Vector	0.577	0.86	0.592	0.70	0.605	0.61	0.341	4.69
Convolutional NN	0.597	0.76	0.610	0.68	0.615	0.68	0.376	3.30
Conv-GRNN	0.637	0.56	0.655	0.51	0.660	0.50	0.425	**2.71**
LSTM-GRNN	**0.651**	**0.50**	**0.671**	**0.48**	**0.676**	**0.49**	**0.453**	3.00
Attention (both levels)			**68.6**					

Next lecture

TEXT CLASSIFICATION

Intent, topic, sentiment

Text/document classification



Document classification

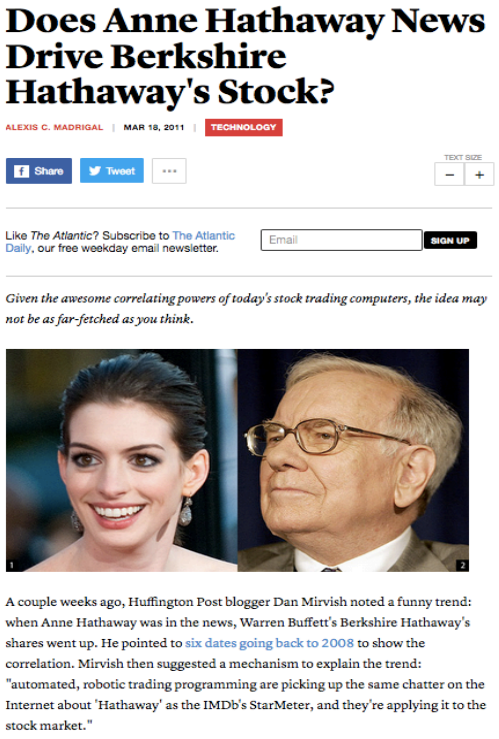
Type	Focus	Example
Topic	Subject matter	Sports vs Technology
Sentiment/opinion	Emotion (current state)	Negative vs Positive
Intent	Action (future state)	Order vs Inquiry

คืนนี้จะได้ดูตอนใหม่แล้ว #ออเจ้า #อดใจไม่ไหว

Topic: บุพเพสันนิวาส
Sentiment: positive
Action: watch

อยากจะสั่งพิซซ่าหน้าฮาวายเอี่ยนหน่อยครับ

Action: order_hawaiian



Does Anne Hathaway News Drive Berkshire Hathaway's Stock?

ALEXIS C. MADRIGAL | MAR 18, 2011 | TECHNOLOGY

Share Tweet ...

TEXT SIZE − +

Like *The Atlantic*? Subscribe to The Atlantic Daily, our free weekday email newsletter. Email SIGN UP

Given the awesome correlating powers of today's stock trading computers, the idea may not be as far-fetched as you think.

A couple weeks ago, Huffington Post blogger Dan Mirvish noted a funny trend: when Anne Hathaway was in the news, Warren Buffett's Berkshire Hathaway's shares went up. He pointed to six dates going back to 2008 to show the correlation. Mirvish then suggested a mechanism to explain the trend: "automated, robotic trading programming are picking up the same chatter on the Internet about 'Hathaway' as the IMDb's StarMeter, and they're applying it to the stock market."

https://www.theatlantic.com/technology/archive/2011/03/does-anne-hathaway-news-drive-berkshire-hathaways-stock/72661/

Other classification application

- Spam filtering
- Authorship id
  - Age id
  - Gender id
  - Native speaker id
  - First language id
- Auto tagging (information retrieval)
- Trend analysis
  - Patent landscape

Text classification definition

- Input
  - Set of documents: $D = \{d_1, d_2, d_3, \ldots, d_M\}$
    - Each document is composed of words
      - $d1 = [w_{11}, w_{12}, \ldots w_{1N}]$
  - Set of classes: $C = \{c_1, c_2, c_3, .., c_J\}$

- Output
  - The predicted class c from the set C

Rule-based classification

- Rules based on phrases or other features
  - Wongnai Rating
    - แมลงสาบ -> 2 ดาว
    - อร่อย -> 4 ดาว
    - ไม่อร่อย -> 2 ดาว
    - …
  - What if the phrase is ไม่ค่อยอร่อย
  - New rule
    - อร่อย โดยที่ไม่มีคำว่าไม่อยู่แถวๆนั้น -> 4 ดาว
  - What if the phrase is ไม่ถูกแต่อร่อย
- This can yield very good results but…
- Building and maintaining rules is expensive!
- Or keep a word list of positive and negative words

Supervised text classification definition

- Input
  - Set of documents: D = $\{d_1, d_2, d_3, \ldots, d_M\}$
  - And labels $Y = \{y_1, y_2, y_3, \ldots, y_M\}$
    - Each document is composed of words
      - d1 = $[w_{11}, w_{12}, \ldots w_{1N}]$
  - Set of classes: C = $\{c_1, c_2, c_3, .., c_J\}$

- Output
  - A classifier H: d -> c

What classifier?

- Any classifier you like
- k-NN
- Naïve Bayes
- Logistic regression
- SVM
- Neural networks

We use this kind of classifier before, in homework and midterm

Outline

- Naïve Bayes
- Topic Models
  - Latent topic models (LDA)
- LDA2vec

Bag of words representation

★★★☆☆ 1 check-in

ร้องเรียน

ก๊วยเตี๋ยวอร่อย ราคาถูก นั่งทานในห้องแอร์

เมนูเด็ด: บะหมี่ต้มยำหมูแดง

ส่วนตัวชอบก๋วยเตี๋ยวต้มยำมากค่ะ รสชาติอร่อยไม่ต้องปรุงเลย แต่ซุปเปอร์ตีนไก่ใส่ถั่วงอกมาด้วย เหมือนเป็นเกาเหลาตีนไก่มากกว่าซุปเปอร์ตีนไก่ รสชาติก็ยังไม่กลมกล่อมเท่าก๋วยเตี๋ยวต้มยำ ตีนไก่เปื่อยดี ทานง่าย

H [ส่วนตัวชอบก๋วยเตี๋ยวต้มยำมากค่ะ รสชาติอร่อยไม่ต้องปรุงเลย แต่ซุปเปอร์ตีนไก่ใส่ถั่วงอกมาด้วย เหมือนเป็นเกาเหลาตีนไก่มากกว่าซุปเปอร์ตีนไก่ รสชาติก็ยังไม่กลมกล่อมเท่าก๋วยเตี๋ยวต้มยำ ตีนไก่เปื่อยดี ทานง่าย] = 3

Bag of words representation

H [ส่วนตัวชอบก๋วยเตี๋ยวต้มยำมากค่ะ รสชาติอร่อยไม่ต้องปรุงเลย แต่ซุปเปอร์ตีนไก่ใส่ถั่วงอกมาด้วย เหมือนเป็นเกาเหลาตีนไก่มากกว่าซุปเปอร์ตีนไก่ รสชาติก็ยังไม่กลมกล่อมเท่าก๋วยเตี๋ยวต้มยำ ตีนไก่เปื่อยดี ทานง่าย] = 3

Bag of words on care the presence of words or features but ignore word position and context

H [ชอบ, อร่อย, ไม่, ไม่, กลมกล่อม, ทานง่าย] = 3

Bag of words representation

H [ส่วนตัวชอบก๋วยเตี๋ยวต้มยำมากค่ะ รสชาติอร่อยไม่ต้องปรุงเลย แต่ซุปเปอร์ตีนไก่ใส่ถั่วงอกมาด้วย เหมือนเป็นเกาเหลาตีนไก่มากกว่าซุปเปอร์ตีนไก่ รสชาติก็ยังไม่กลมกล่อมเท่าก๋วยเตี๋ยวต้มยำ ตีนไก่เปื่อยดี ทานง่าย] = 3

Bag of words on care the presence of words or features but ignore word position and context

H [

Word	Count
ชอบ	1
อร่อย	1
ไม่	2
กลมกล่อม	1
ทานง่าย	1

] = 3

Bag of words for classification intuition

Bag of words for classification intuition

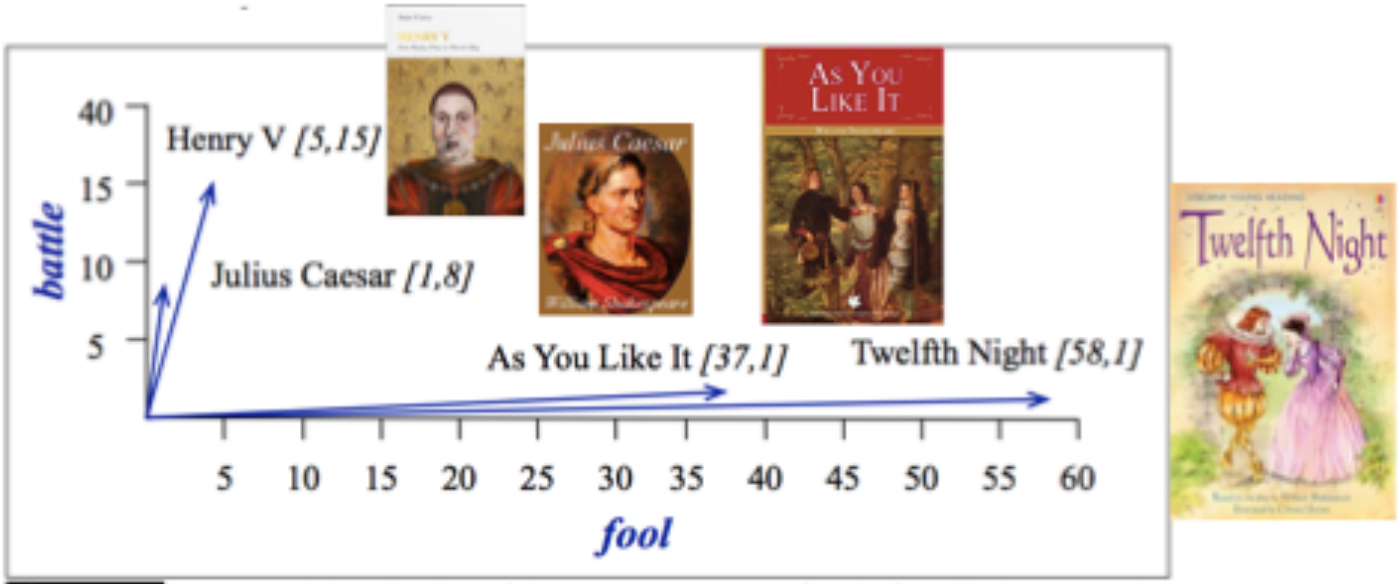


Figure 15.3 A spatial visualization of the document vectors for the four Shakespeare play documents, showing just two of the dimensions, corresponding to the words *battle* and *fool*. The comedies have high values for the *fool* dimension and low values for the *battle* dimension.

Reference: Jurafsky, Dan, and James H. Martin. Speech and language processing. 3rd edition draft, https://web.stanford.edu/~jurafsky/slp3/, August 2017

Bayes' Rule for classification

- A simple classification model
- Given document d, find the class c
  - Argmax P(c|d)
    c

$$= \underset{c}{\text{Argmax}} \frac{P(d|c)\,P(c)}{P(d)}$$

Bayes' Rule

$$= \underset{c}{\text{Argmax}}\, P(d|c)\,P(c)$$

P(d) is constant wrt to c

$$= \underset{c}{\text{Argmax}}\, P(x_1, x_2, \ldots, x_n \mid c)\,P(c)$$

The document is represented by features $x_1, x_2, \ldots, x_n$

Bayes' Rule for classification

- A simple classification model
- Given document d, find the class c
  - Argmax P(c|d)
    c

    $= \text{Argmax}_c \frac{P(d|c)\,P(c)}{P(d)}$

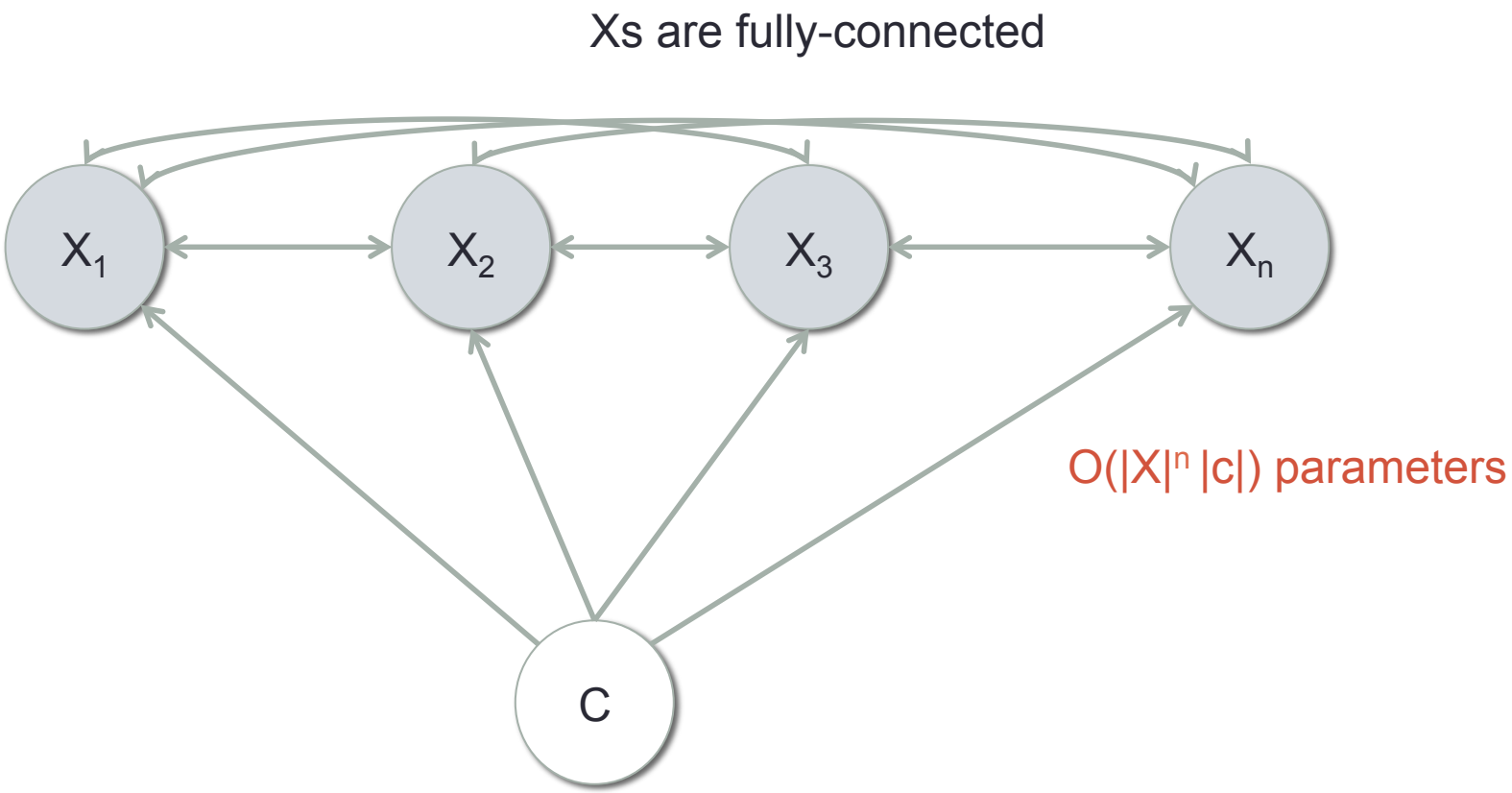
    $= \text{Argmax}_c P(d|c)\,P(c)$

    $= \text{Argmax}_c P(x_1, x_2, \ldots, x_n \mid c)\,P(c)$

$P(x_1, x_2, \ldots, x_n \mid c)$ requires $O(|X|^n\,|c|)$ parameters. Cannot train

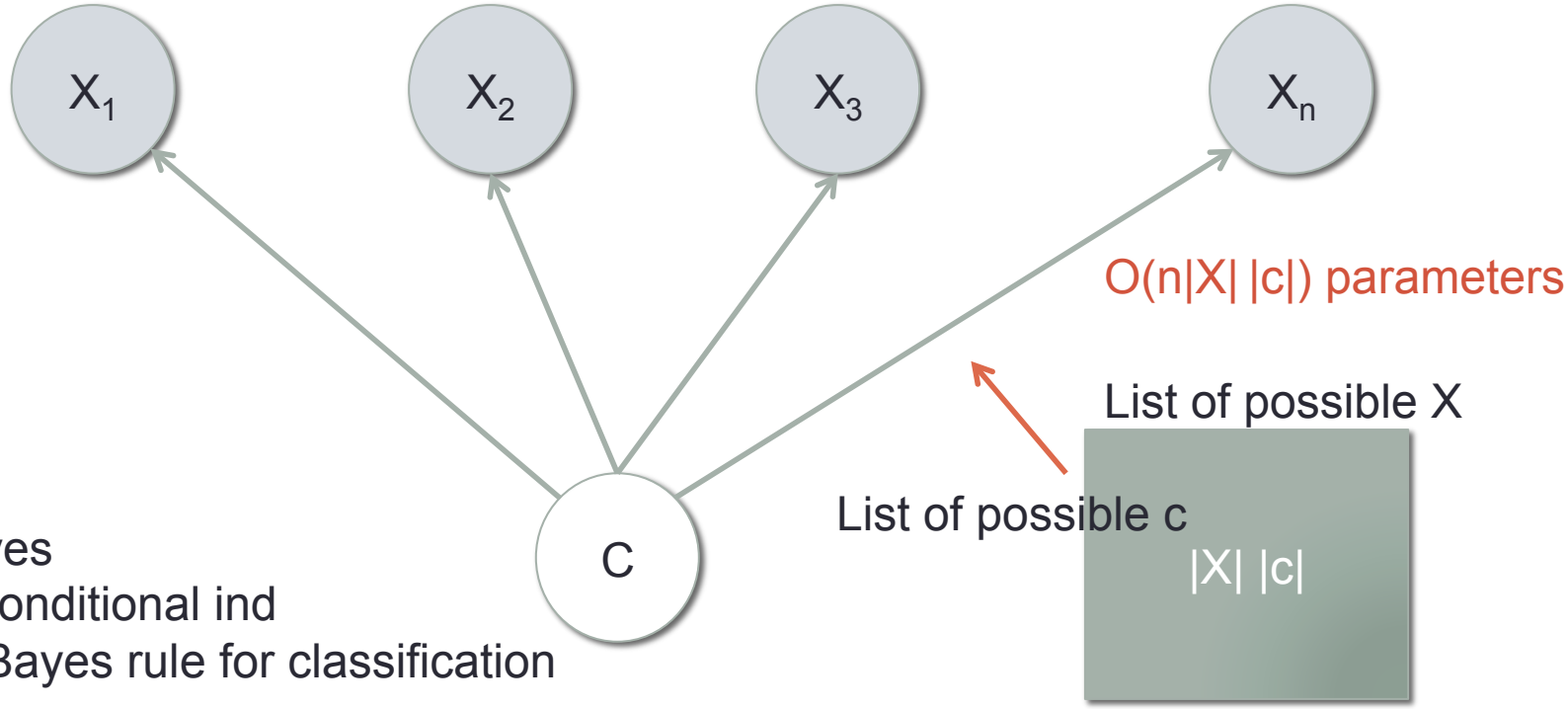
Graphical view

- $P(x_1, x_2, \ldots, x_n \mid c)\, P(c)$



Bag of words assumption

- $P(x_1, x_2, \ldots, x_n \mid c)\, P(c) = P(x_1|c)\, P(x_2|c)\, P(x_3|c) \ldots P(x_n|c)\, P(c)$
  - Conditional independence



Naïve Bayes
Naïve – conditional ind
Bayes – Bayes rule for classification

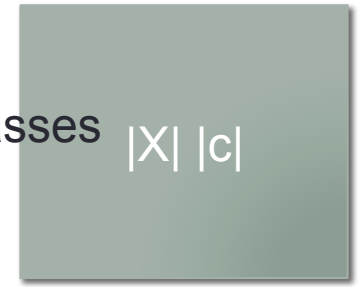
Learning the Naïve Bayes model

- As usual counts
- $P(x_1|c)$
- $P(x_1 = 1| c=5) = \frac{count(x_1 = 1 , c = 5)}{count(c = 5)}$
- $P(c)$
- $P(c = 5) = \frac{count (c = 5)}{count (all\ reviews)}$

x_n is a feature that counts word occurrence
x_1 how many times ยอด appear

List of possible counts

List of classes



- This is the Maximum Likelihood Estimate (MLE)

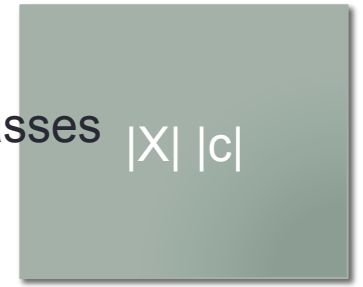
Learning the Naïve Bayes model

- What if we encounter zeroes in our table
- $P(x_1|c)$
- $P(x_1 = 1| c=2) = \frac{count(x_1 = 1 , c = 2)}{count(c = 2)}$

x_n is a feature that counts word occurrence
x_1 how many times ยอด appear

List of possible counts

List of classes



$= 0$

- P('ร้าน นี้ ราด หน้า ยอด ผัก ไม่ อร่อย'|c = 2)

$= P(x_1=1|c=2)*\ldots$

$= 0$

Zero probability regardless of other words

One solution: add-1 smoothing

Naives Bayes

- Can use other features beside word counts
  - Feature engineering – restaurant name, location, price range, reviewer id, date of review
  - Tedious but very powerful
    - Features > 10000
- Pros: very fast, very small model
- Robust especially for small training data (hand-crafted rules)
- A good fast baseline. Always try Naive Bayes or logistic regression in model search.
- Even with lots of data and rich features, Naives Bayes can be very competitive and fast!

Fast and good classification using n-grams

- Features: n-grams (bag of phrases)
- Model: logistic regression
- Very competitive results

Model	Yelp'13	Yelp'14	Yelp'15	IMDB
SVM+TF	59.8	61.8	62.4	40.5
CNN	59.7	61.0	61.5	37.5
Conv-GRNN	63.7	65.5	66.0	42.5
LSTM-GRNN	65.1	67.1	67.6	45.3
`fastText`	64.2	66.2	66.6	45.2

Table 3: Comparision with Tang et al. (2015). The hyper-parameters are chosen on the validation set. We report the test accuracy.

Bag of Tricks for Efficient Text Classification
https://arxiv.org/pdf/1607.01759.pdf

Fast and good classification using n-grams

- Features: n-grams (bag of phrases)
- Model: logistic regression
- Very competitive results

	Zhang and LeCun (2015)		Conneau et al. (2016)			fastText
	small char-CNN	big char-CNN	depth=9	depth=17	depth=29	$h = 10$, bigram
AG	1h	3h	24m	37m	51m	1s
Sogou	-	-	25m	41m	56m	7s
DBpedia	2h	5h	27m	44m	1h	2s
Yelp P.	-	-	28m	43m	1h09	3s
Yelp F.	-	-	29m	45m	1h12	4s
Yah. A.	8h	1d	1h	1h33	2h	5s
Amz. F.	2d	5d	2h45	4h20	7h	9s
Amz. P.	2d	5d	2h45	4h25	7h	10s

Table 2: Training time for a single epoch on sentiment analysis datasets compared to char-CNN and VDCNN.

Bag of Tricks for Efficient Text Classification
https://arxiv.org/pdf/1607.01759.pdf

Tag prediction

Model	prec@1	Running time	
		Train	Test
Freq. baseline	2.2	-	-
Tagspace, $h = 50$	30.1	3h8	6h
Tagspace, $h = 200$	35.6	5h32	15h
`fastText`, $h = 50$	31.2	6m40	48s
`fastText`, $h = 50$, bigram	36.7	7m47	50s
`fastText`, $h = 200$	41.1	10m34	1m29
`fastText`, $h = 200$, bigram	46.1	13m38	1m37

Table 5: Prec@1 on the test set for tag prediction on YFCC100M. We also report the training time and test time. Test time is reported for a single thread, while training uses 20 threads for both models.

Bag of Tricks for Efficient Text Classification
https://arxiv.org/pdf/1607.01759.pdf

Naïve Bayes tricks for text classification

- **Domain specific features**
- Count words after “not” as a different word
  - I don’t go there. -> I don’t go_not there_not
- Upweighting: double counting words at important locations
  - Words in titles
  - First sentence of each paragraph
  - Sentences that contain title words

Context-Sensitive Learning Methods for Text Categorization
https://www.researchgate.net/publication/2478208_Context-Sensitive_Learning_Methods_for_Text_Categorization

Information retrieval using location and category information
https://www.jstage.jst.go.jp/article/jnlp1994/7/2/7_2_141/_article

Automatic text categorization using the importance of sentences
https://dl.acm.org/citation.cfm?id=1072331

Relationship to language modeling

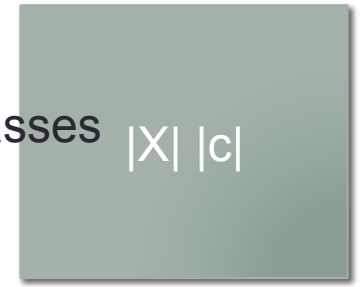
- .
- $P(x_1|c)$

x_1 is a feature that looks at the first word

- $P(x_1 = \text{ยอด}| c=5) = \frac{count(x_1 = \text{ยอด} , c = 5)}{count(c = 5)}$
- $P(c)$
- $P(c = 5) = \frac{count (c = 5)}{count (all\ reviews)}$

List of words

List of classes



This looks like… n-grams, but instead of conditioning on the past, we condition on the topic –
bag of words model for topic modeling (unigram with topic)

Language modeling view

- Which class is this review
- P(w|c)

อร่อย แต่ ไม่ ถูก

Class= 1	
อร่อย	0.01
แต่	0.4
ไม่	0.4
ถูก	0.03
…	

Class= 5	
อร่อย	0.4
แต่	0.05
ไม่	0.25
ถูก	0.15
…	

P(s|c =1) = 0.01*0.4*0.4*0.03 = 0.000048
P(s|c = 5) = 0.4*0.05*0.25*0.15 = 0.00075

Topic modeling

- Sometimes you want to model the topic of a document

Class=
บรรยากาศ

อร่อย	0.01
แต่	0.4
ไม่	0.4
ถูก	0.03
…	

Class=
อาหาร

อร่อย	0.4
แต่	0.05
ไม่	0.25
ถูก	0.15
…	

อาหารที่นี่ไม่ค่อยอร่อย แต่ขนมใช้ได้เลย ถ้าว่างอาจจะกลับมากินอีก แนะนำให้สั่งเค้กใบเตย

ด้านบรรยากาศ มีเสียงก่อสร้างมาจากตึกข้างๆ แต่นอกนั้นตกแต่งโอเค แต่ยังขาดอะไรไปหลายๆอย่าง

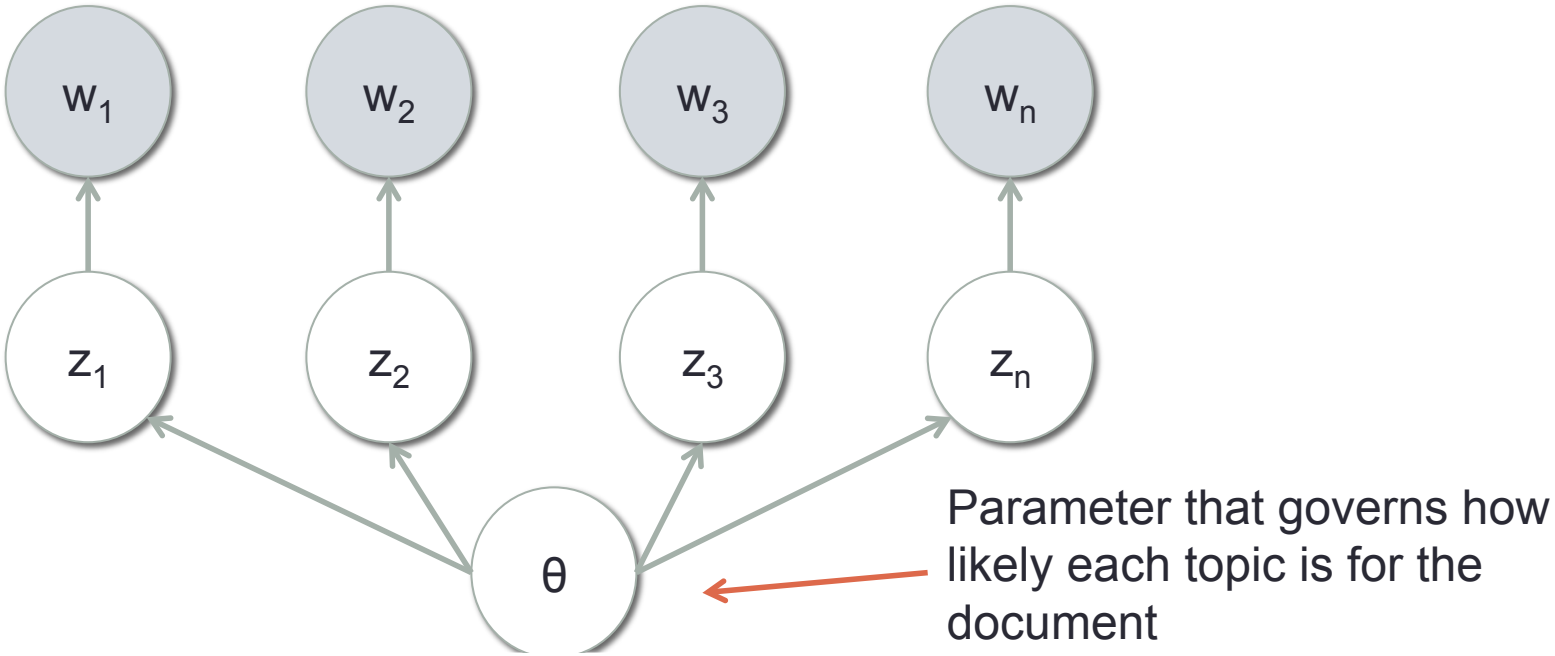
P(s|c=บรรยากาศ) = ?
P(s|c=อาหาร) = ?

Naïve Bayes Topic modeling issues

- Most document have multiple topics (multi-label).
  - Our model assumes 1 document 1 topic.
- Solution: Let a document be a mixture of topics (language model interpolation). Each word has its own topic, z.
  - P(w) = P(w is topic A) P(w | topicA) + P(w is topic B) P(w | topicB)
  - P(w is topic A) + P(w is topic B) = 1

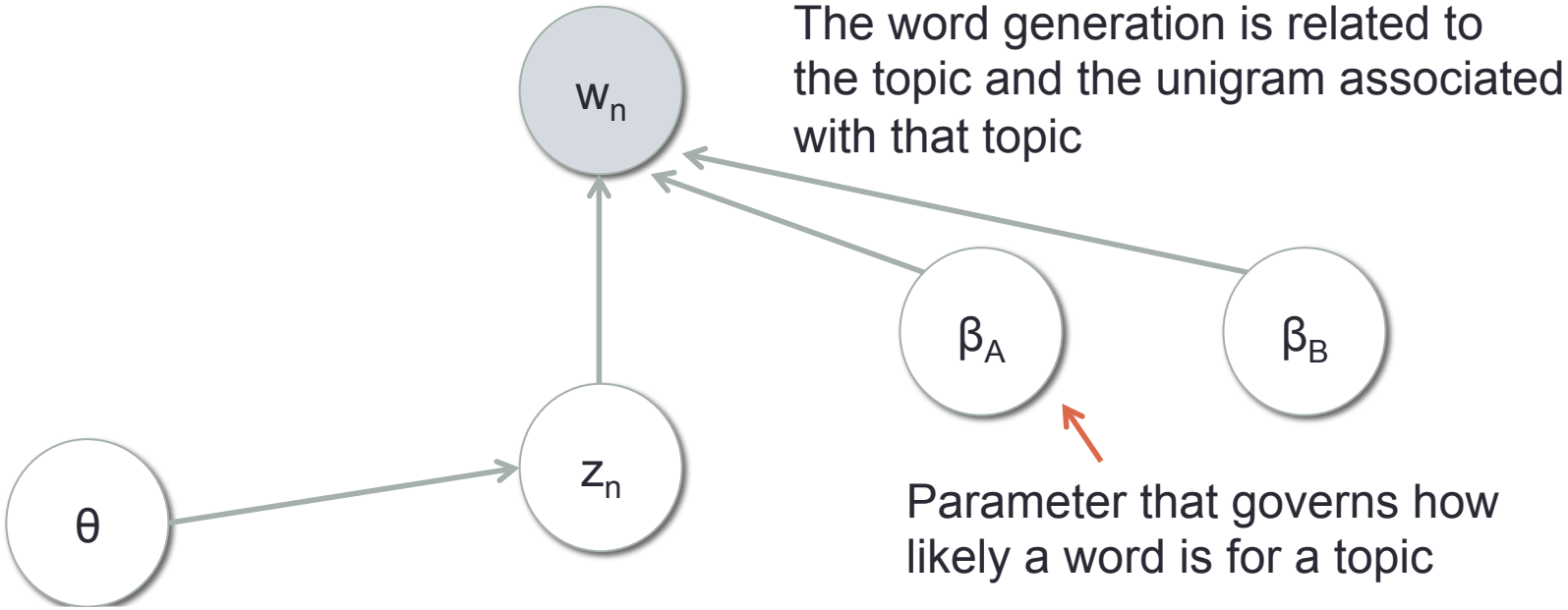
Naïve Bayes Topic modeling issues

- Most document have multiple topics. Our model assumes 1 document 1 topic.
  - Let a document be a mixture of topics (language model interpolation). Each word has its own topic, z.
    - P(w) = P(z = A) P(w | z = A) + P(z = B) P(w | z = B)
    - P(z = A) + P(z = B) = 1, θ = P(z = A)



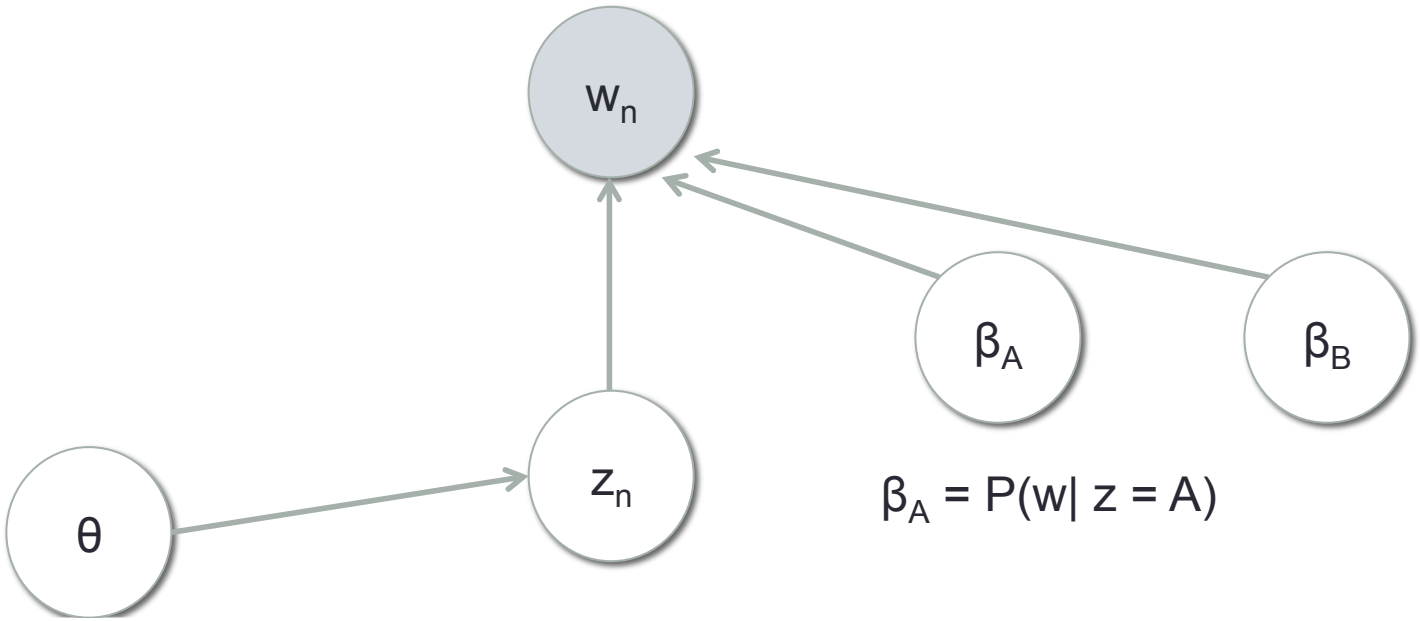
Topic modeling

- Most document have multiple topics. Our model assumes 1 document 1 topic.
  - Let a document be a mixture of topics (language model interpolation). Each word has its own topic, z.
    - P(w) = P(z = A) P(w | z = A) + P(z = B) P(w | z = B)
    - P(z = A) + P(z = B) = 1, $\theta = P(z = A)$ $\beta_A = P(w| z = A)$



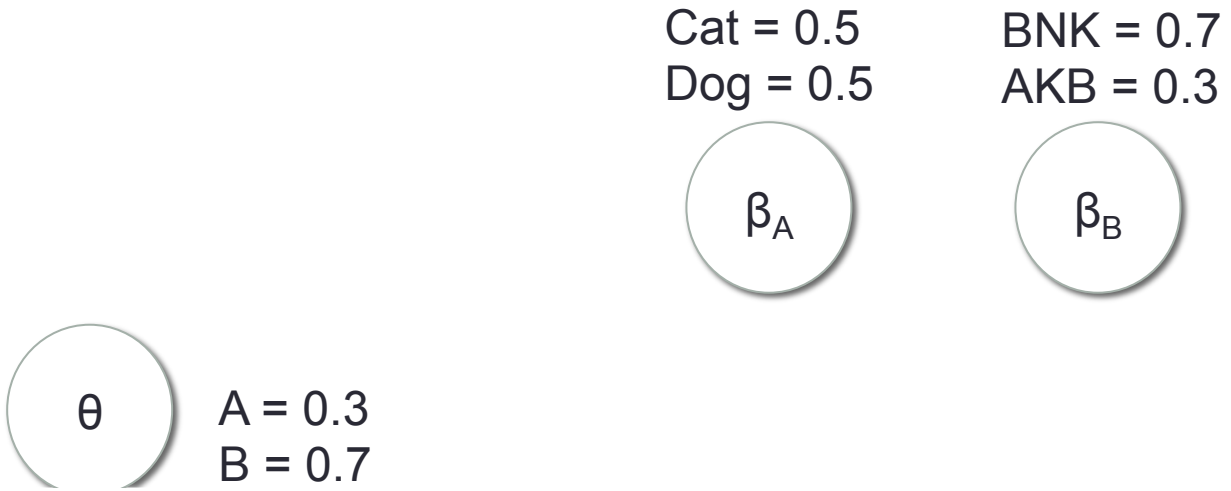
Graphical model and generation

- You can generate a sample from a graphical model by following the arrows



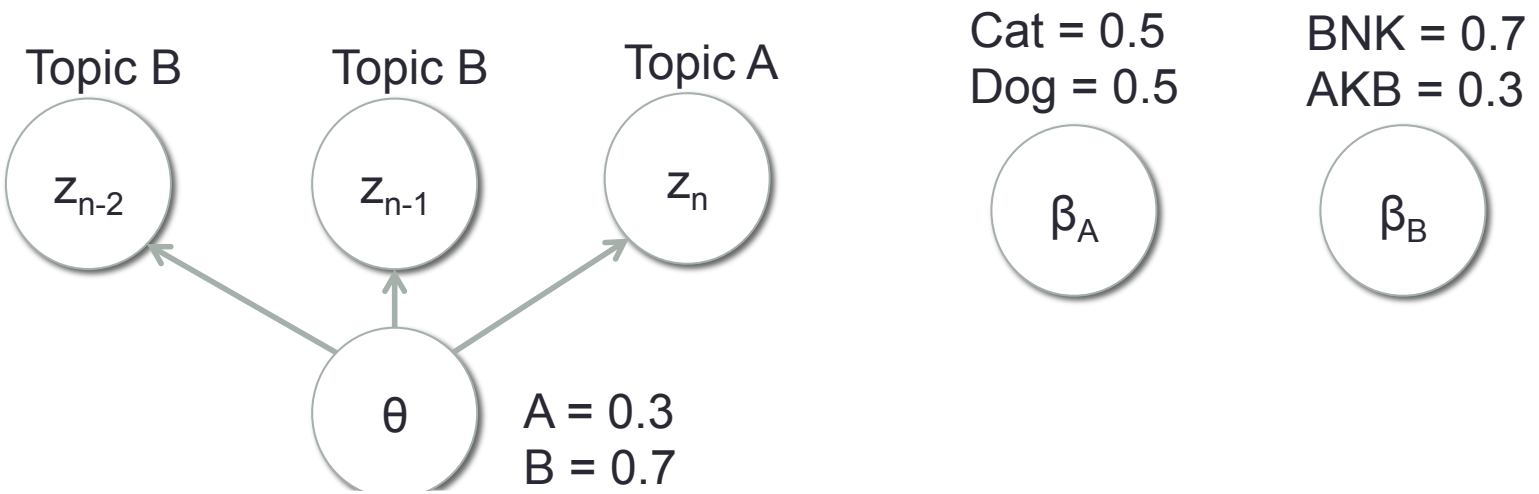
Graphical model and generation

- Given θ, β_A, β_B



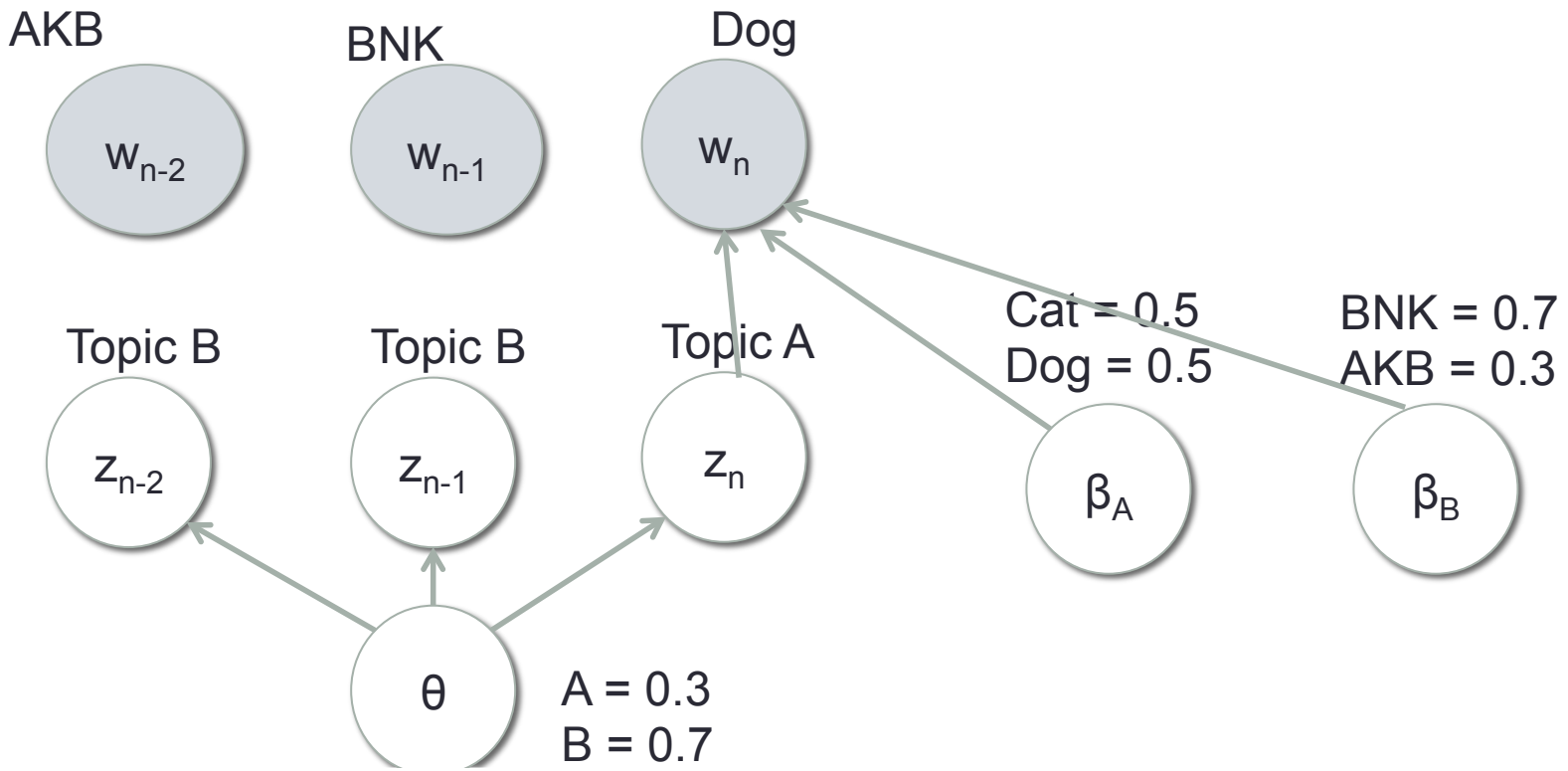
Graphical model and generation

- Given θ, β_A, β_B
- Generate (randomly create) z from θ



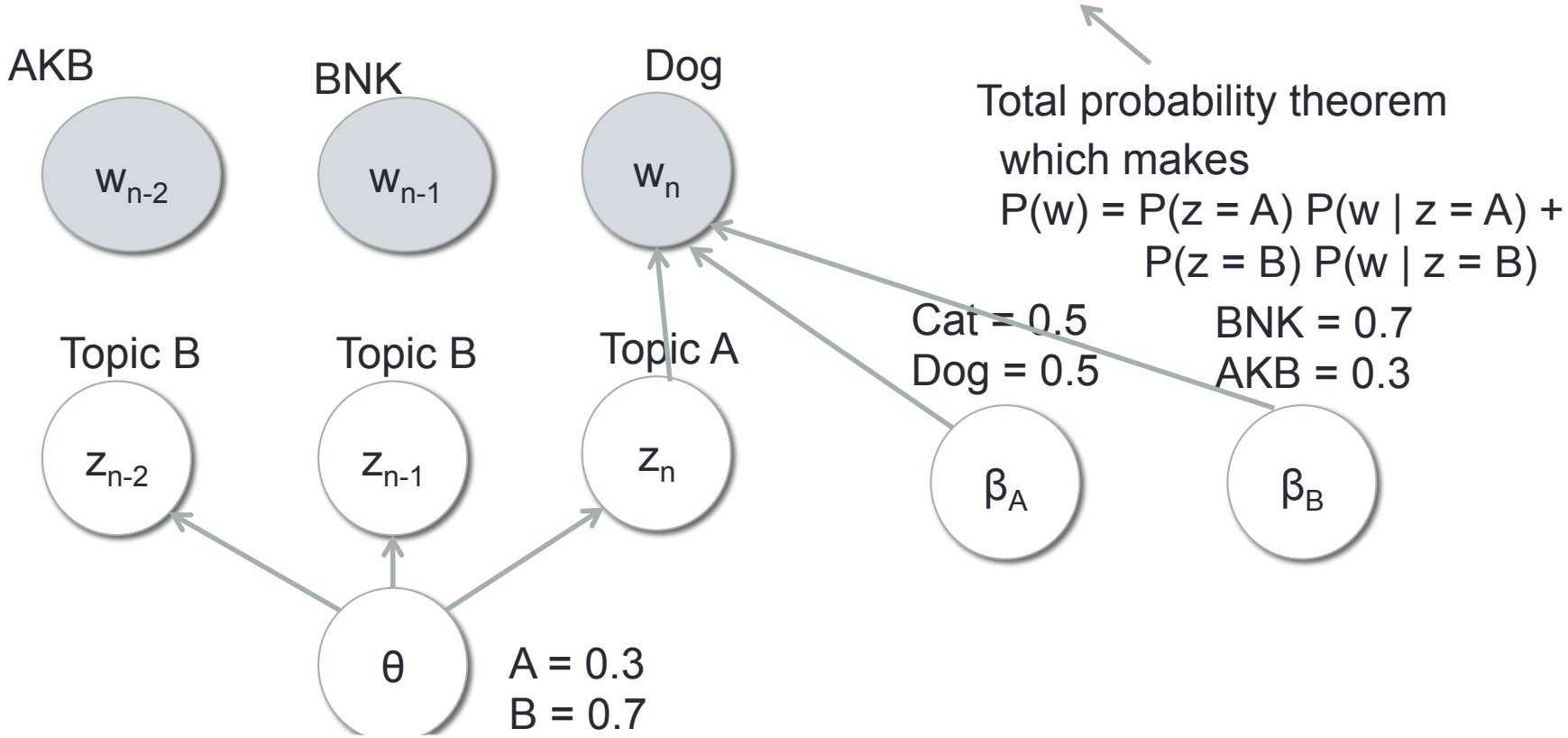
Graphical model and generation

- Given θ, β_A, β_B
- Generate (randomly create) z from θ
- Generate w_n from z_n β_A, β_B



Graphical model and generation

How likely a sentence is likely to be generated follows this generation process
P(AKB,BNK,Dog,B,B,A) = P(B)P(B)P(A)P(AKB|B)P(BNK|B)P(Dog|A)
Note
P(AKB,BNK,Dog) = P(AKB,BNK,Dog,A,A,A) + P(AKB,BNK,Dog,A,A,B)
P(AKB,BNK,Dog,A,B,A) + P(AKB,BNK,Dog,A,B,B) +...



Graphical models and plate notation

- Sometimes you have too many relationships.

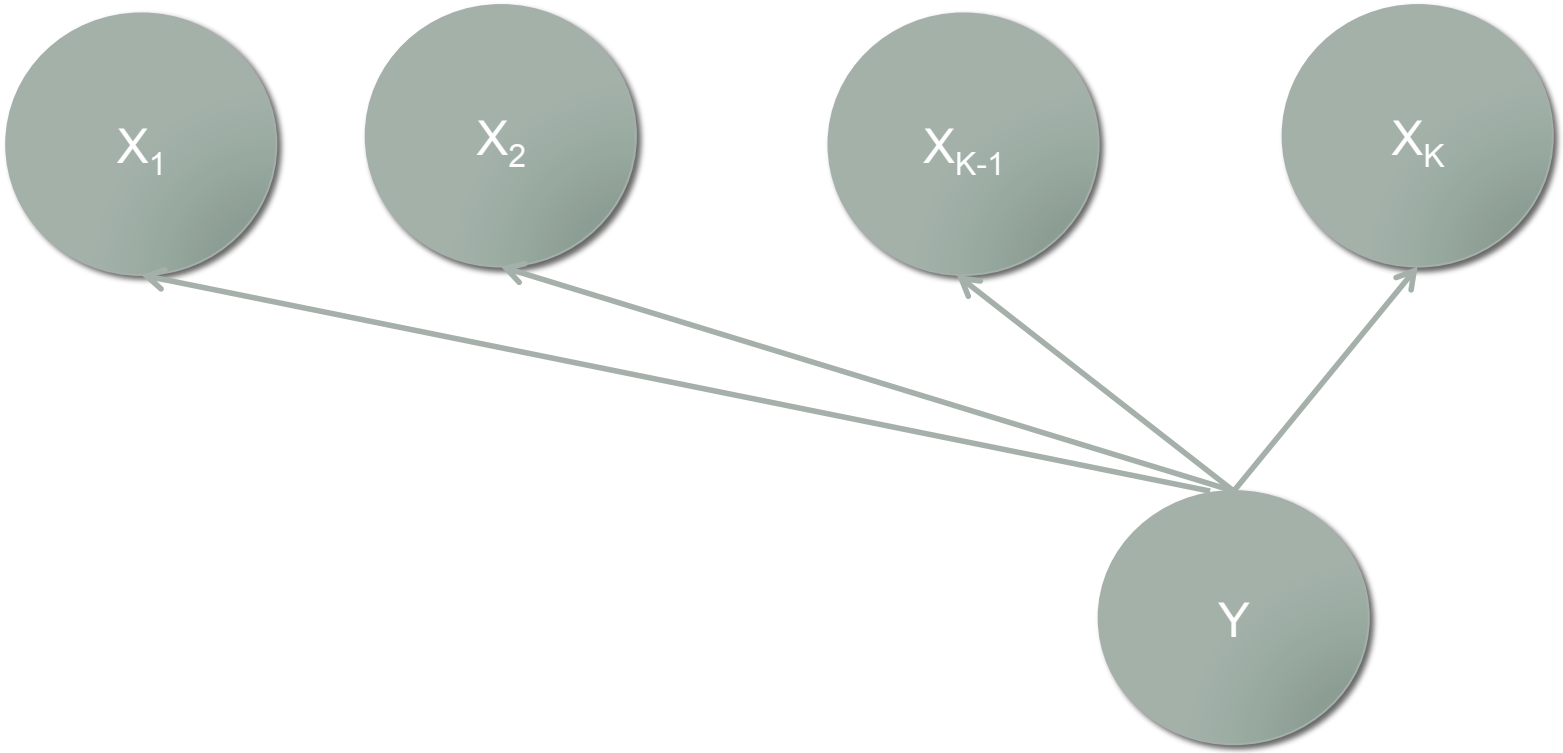
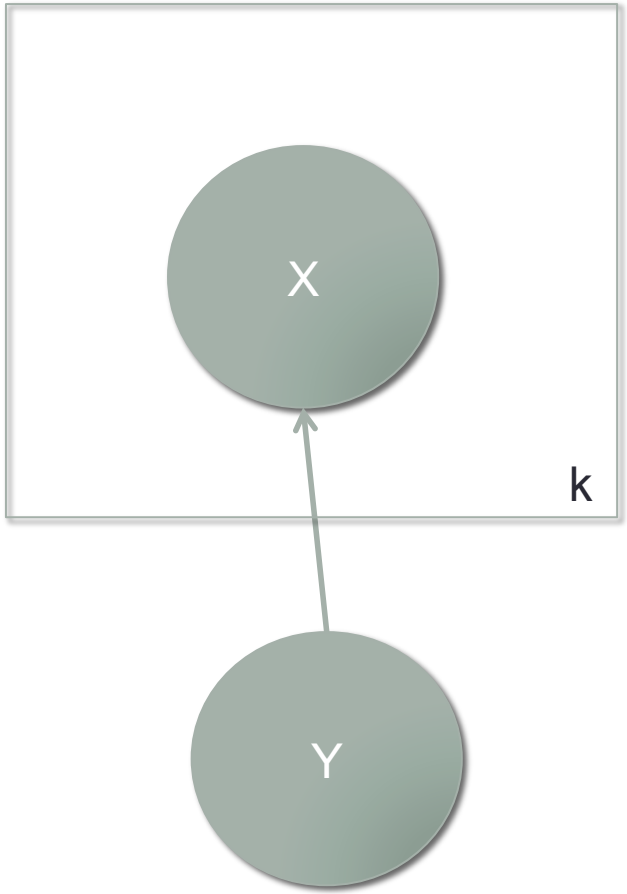
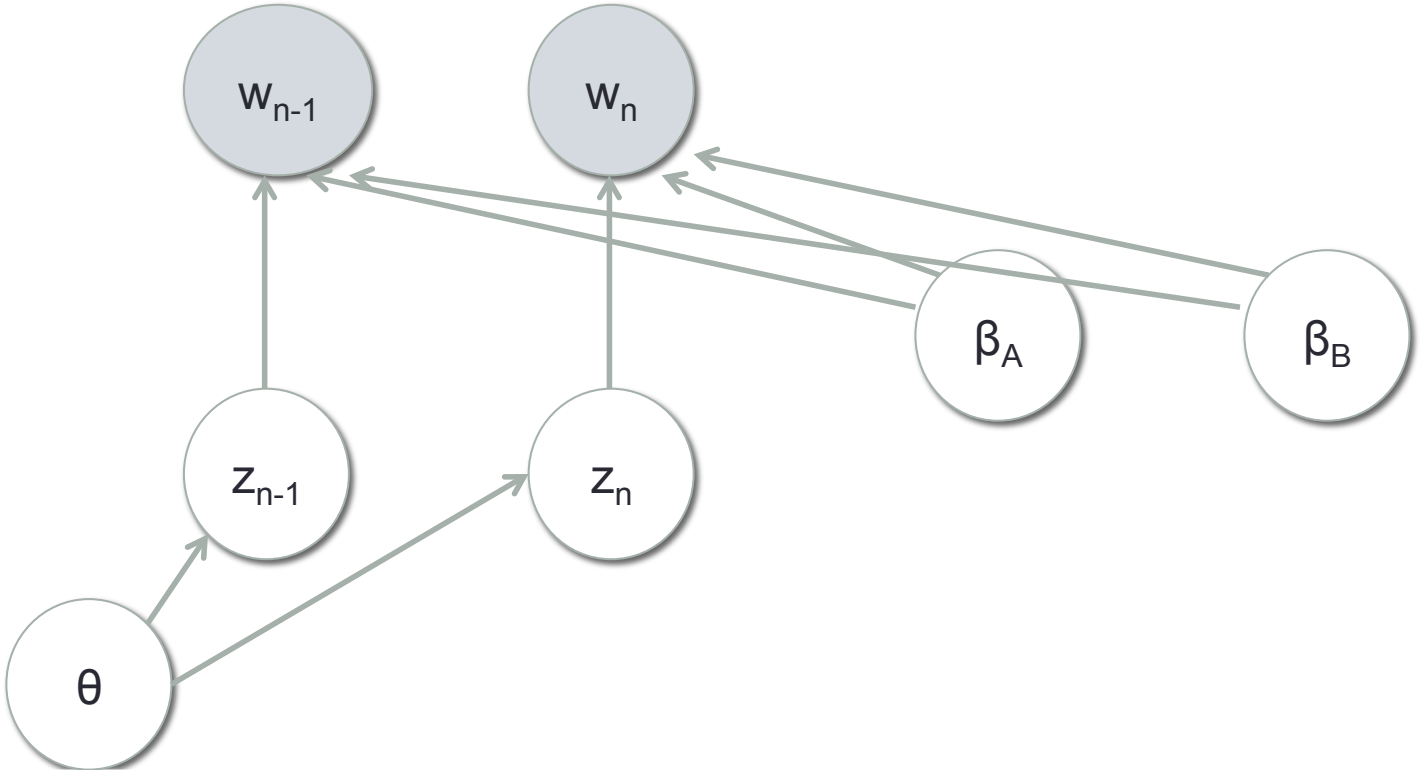


Plate notation

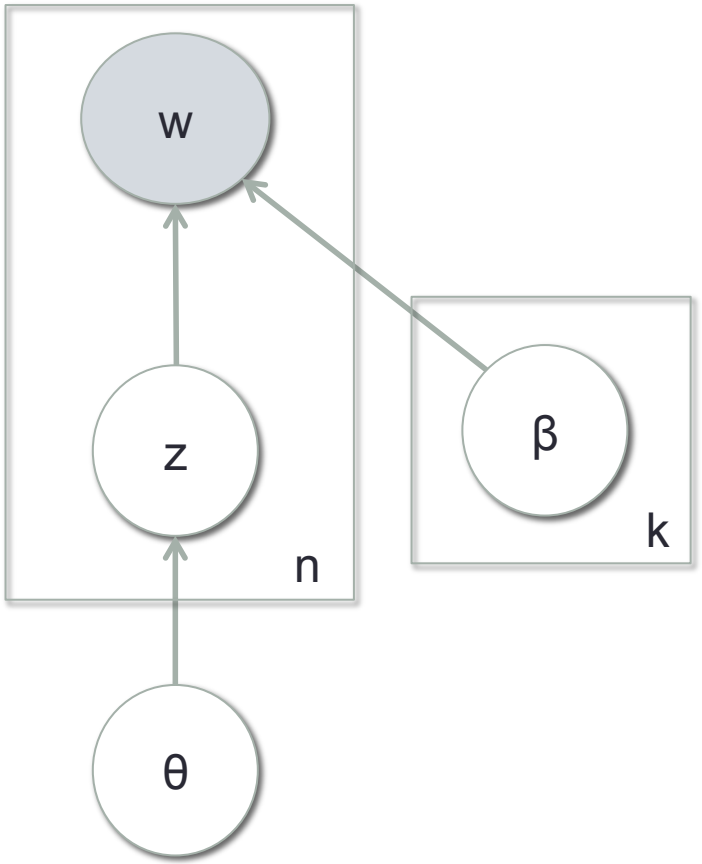
- Summarize by using a square box with number



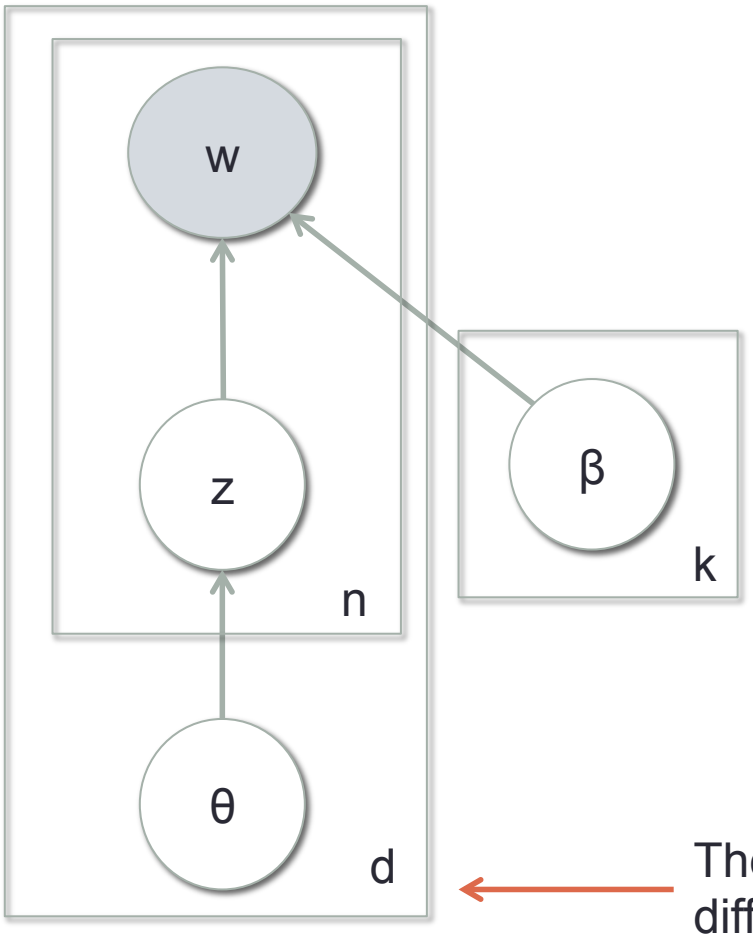
Topic modeling with plate notation



Topic modeling with plate notation

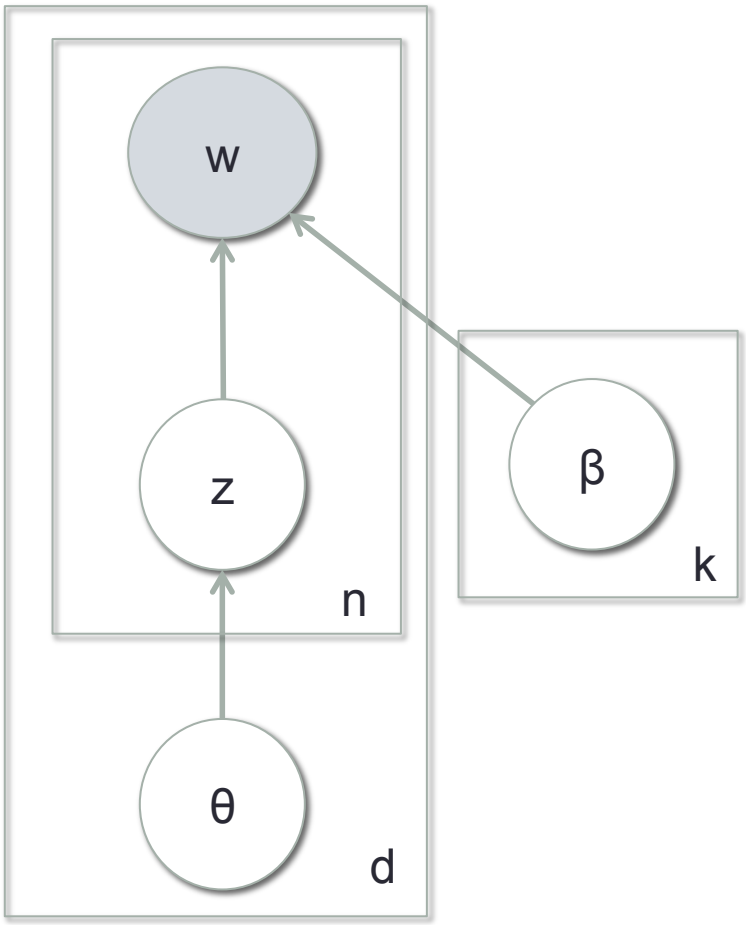


Topic modeling with plate notation



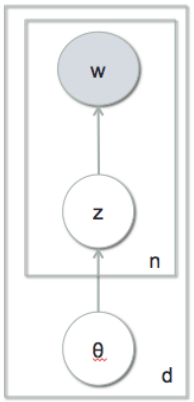
There are d documents each with different topic distributions

Topic modeling with plate notation



Called pLSA
probabilistic Latent Semantic Analysis

Note: if you look at other textbooks, you will see a slightly different picture



Learning topic latent model parameters

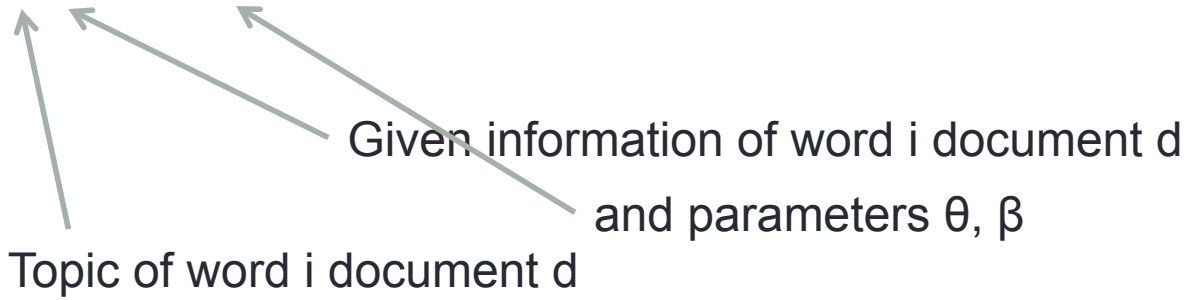
- How to find θ and β?
  - “Cat,Dog,BNK”, “Cat, cat, cat, BNK”, “Dog, dog, BNK, dog, AKB”
- If we know, the latent topic z for each word we can use the counts
  - “Cat_A, Dog_A, BNK_B”
- $P(Cat|A) = \frac{count(Cat\_A)}{count(A)}$
- $P_1(A) = \frac{count(A)}{count(\text{all words in document } 1)}$
- But we don’t know the topic z for each word

Expectation maximization (EM)

- A method to iteratively maximize the likelihood of a model on training data
  - Expectation step (E step) – guess latent variables from model parameters (get soft counts)
  - Maximization step (M step) – re-estimate model parameters from latent variables (counts)

E-step

- Find an estimate for the latent variable given parameters θ, β
- $p(z_{di} \mid w_{di}, \theta, \beta)$



E-step

- Find an estimate for the latent variable given parameters θ, β

$$p(z_{di}|w_{di},\theta,\beta) = \frac{p(z_{di},w_{di},\theta,\beta)}{\sum_{z'=1}^{k} p(z'_{di},w_{di},\theta,\beta)}$$

$p(w, \theta, \beta)$

E-step

- Find an estimate for the latent variable given parameters θ, β

$$p(z_{di}|w_{di},\theta,\beta) = \frac{p(z_{di},w_{di},\theta,\beta)}{\sum_{z'=1}^{k} p(z'_{di},w_{di},\theta,\beta)}$$

$$= \frac{\theta_{z|d}\beta_{w|z}}{\sum_{z'=1}^{k} \theta_{z'|d}\beta_{w|z'}}$$

Index di for z and w dropped for clarity

E-step

- Find an estimate for the latent variable given parameters θ, β

$$p(z_{di}|w_{di},\theta,\beta) = \frac{p(z_{di},w_{di},\theta,\beta)}{\sum_{z'=1}^{k} p(z'_{di},w_{di},\theta,\beta)}$$

$$= \frac{\theta_{z|d}\beta_{w|z}}{\sum_{z'=1}^{k} \theta_{z'|d}\beta_{w|z'}}$$

P(Word is from topic A| word is cat from document 1)
Probability that the word is from each topic. Use as counts

M-Step

- Instead of real counts use $P(z_{di})$ as the topic label
  - $P(\text{Cat}|A) = \frac{\text{count}(\text{Cat\_A})}{\text{count}(A)}$
  - $P_1(A) = \frac{\text{count}(A)}{\text{count}(\text{all words in document 1})}$

Expectation maximization (EM)

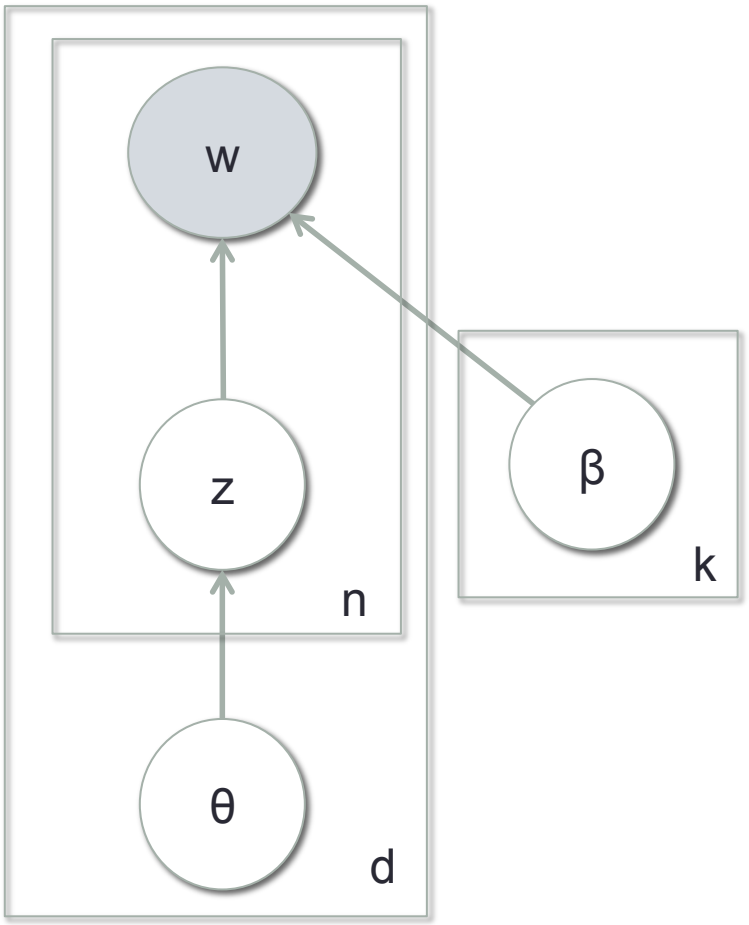
- Initialize θ, β
- Expectation step (E step) – guess latent variables from model parameters (get soft counts)
- Maximization step (M step) – re-estimate model parameters from latent variables (counts)
- Repeat E and M step until satisfied (Likelihood of the whole training set using the model does not change much)

- For more info on EM see pattern recognition lecture 6

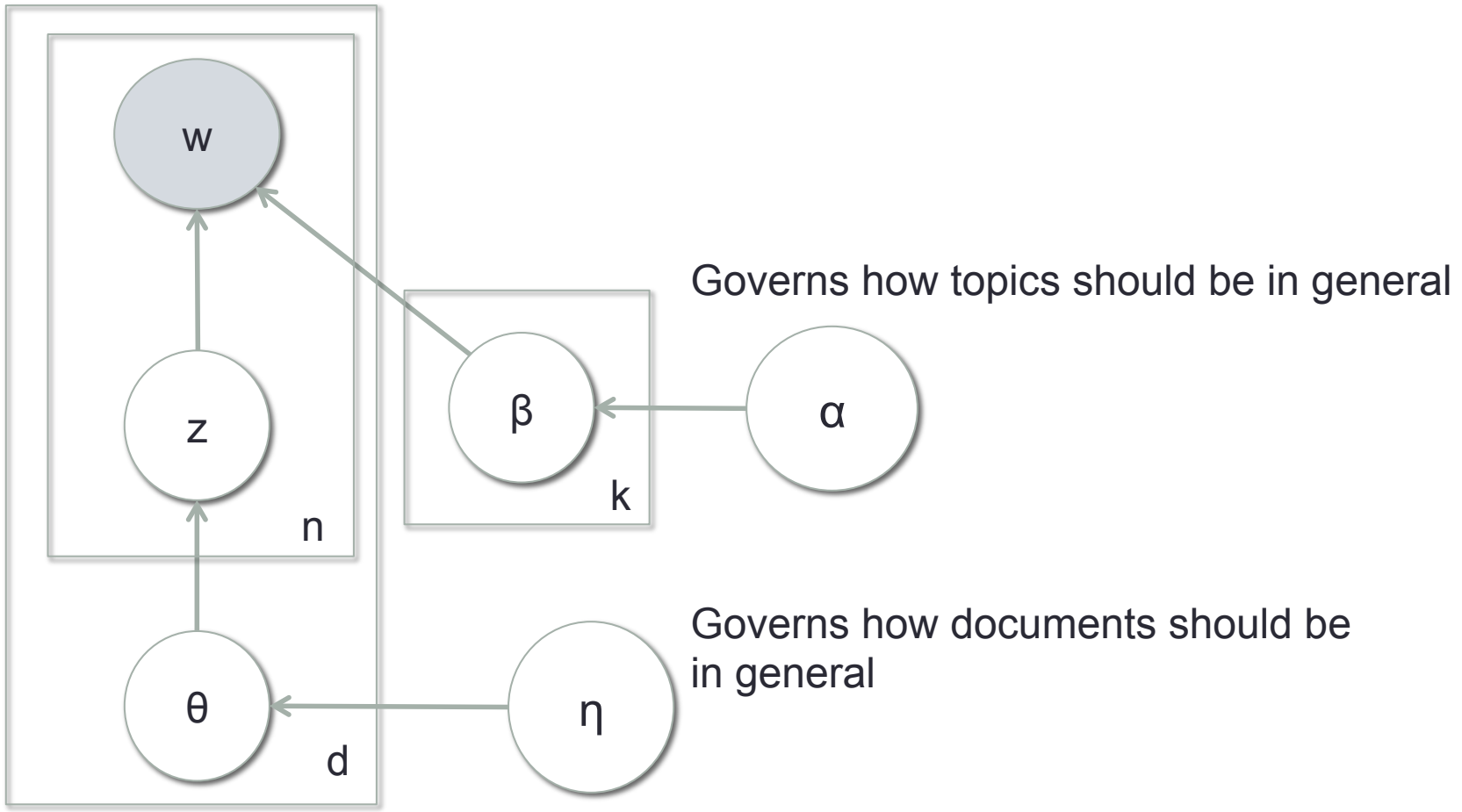
pLSA

- pLSA automatically clusters words into topic unigrams
  - Requires user to specify number of topics
- Automatically learn document representation based on the learned topics
  - DocA = [0.7 0.3] DocB = [0.2 0.8] DocC = [0.5 0.5]
- Overfits easily to data outside of the training set
  - Nothing that ties all document together
  - A document from a document collection should be have topic distributions that are similar
- Solution: LDA (Latent Dirchlet Allocation)

pLSA



LDA


w
z
n
β
k
α
θ
d
η
Governs how topics should be in general
Governs how documents should be in general

α and η

- α is a Dirichlet distribution
- Or a distribution of distributions…

- Example: rolling a die
- P(x=1) = 0.3, P(x=2) = 0.1,…
- This is a distribution. (Multinomial distribution)
- But what if I have a bag of dices, each with different distributions.
- α tells me the probability of what kind of die I will get

Dirichlet distribution

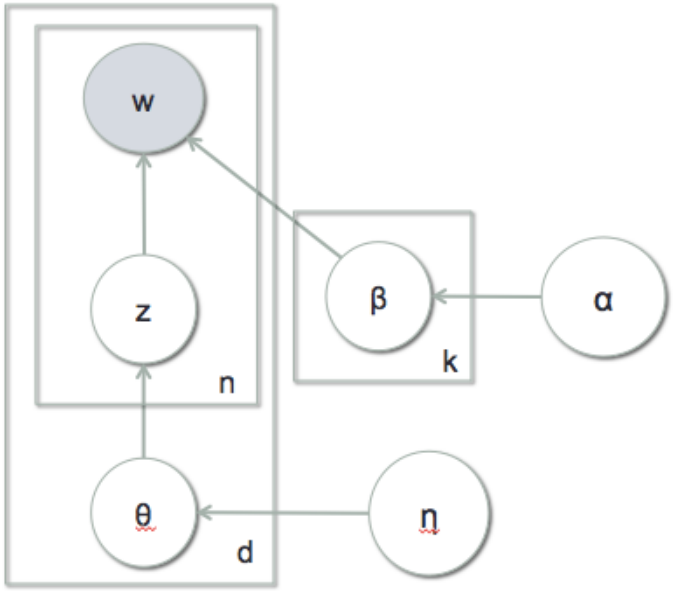
- pdf

Parameters giving preference to each side of die/topic

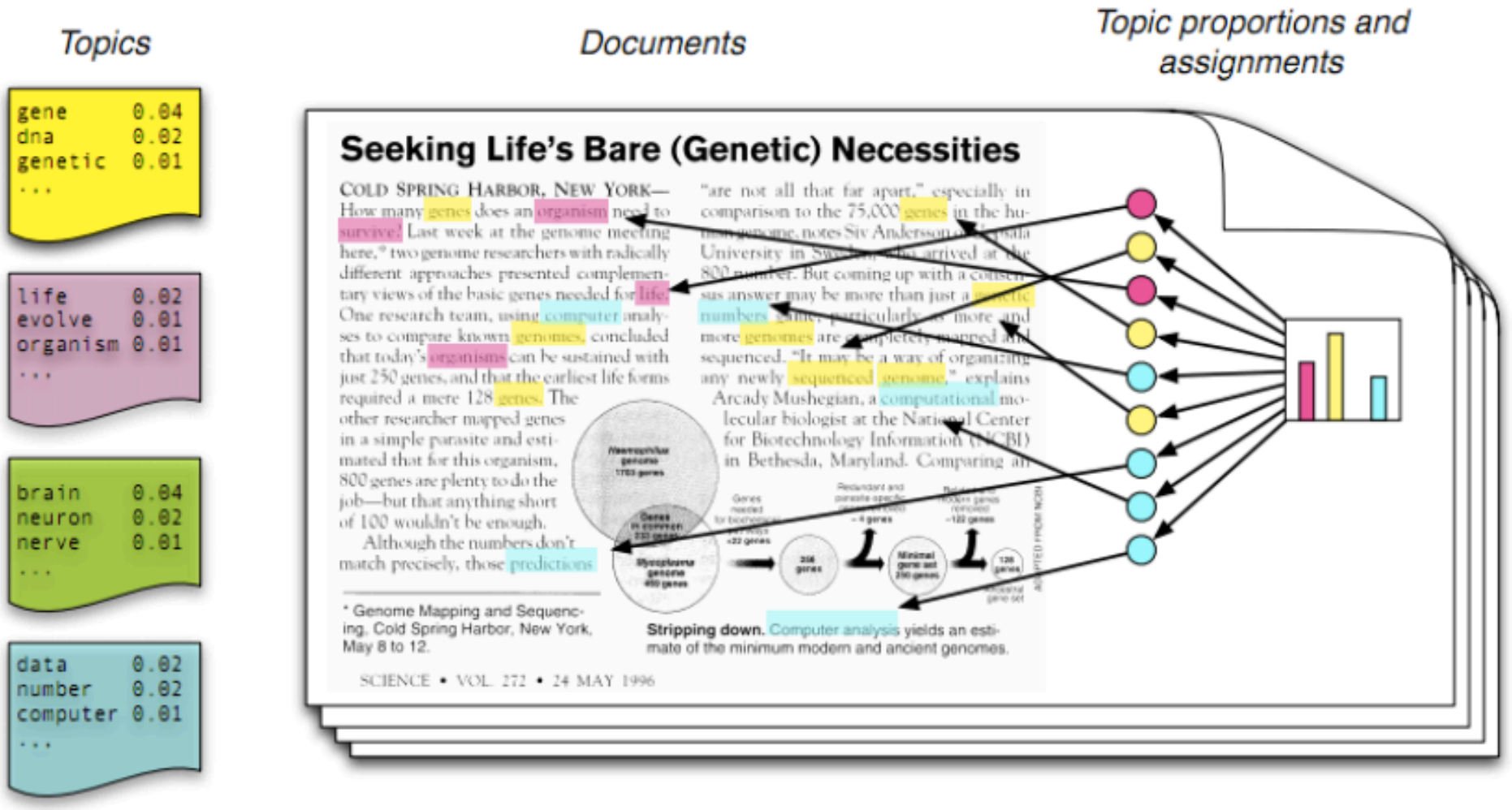
$$f(x_1, \ldots, x_K; \alpha_1, \ldots, \alpha_K) = \frac{1}{\mathrm{B}(\boldsymbol{\alpha})} \prod_{i=1}^{K} x_i^{\alpha_i - 1}$$

Probability of each side of die, probability of each topic

Generative process



- For each topic k ∈ {1,...,K}:
  - Draw word distributions β_k ~ Dir(η)
- For each document d ∈ {1,...,D}:
  - Draw topic proportions θ_d ~ Dir(α)
  - For each word in a document n ∈ {1,...,N}:
    - Draw a topic index z_{dn} ~ Mult (θ)
    - Generate word from chosen topic
      - w_{dn} ~ Mult(β_{zdn})



Introduction to Probabilistic Topic Models, Blei 2011
http://menome.com/wp/wp-content/uploads/2014/12/Blei2011.pdf

Parameter learning

- How do we actually learn these parameters and latent variables?
  - Gibbs sampling
  - Variational methods
- Totally beyond the scope of this class

LDA

- Automatically learns topics, and the word distribution of each topic
  - Just give a bunch of documents!
  - Each document is given a mixture of topics
    - Dirichlet prior prefers sparse topics – each document only have probability in few topics – easy for interpretibility
- Requires user to pick number of topic
- Requires user to make sense of the learned topics

Example usage

- Lots of lectures. Can we discover the topics? Can we classify the lectures? No topic labels given

Top 10 high probability nouns in word probabilities of Topics 3 ($\theta_{k=3}$), 15 ($\theta_{k=15}$), and 26 ($\theta_{k=26}$).

Topic 3 (~ classical mechanics)	Topic 15 (~astronomy)	Topic 26 (~ (time) unit)
m	Light	Percent
Energy	Degrees	Time
Force	Angle	Dollars
Mass	Frequency	Times
Point	Energy	Minutes
Velocity	Direction	Day
Direction	Waves	Bit
v	Sun	Year
Times	Star	Hour
Speed	Speed	Half

http://www.sciencedirect.com/science/article/pii/S0885230810000641

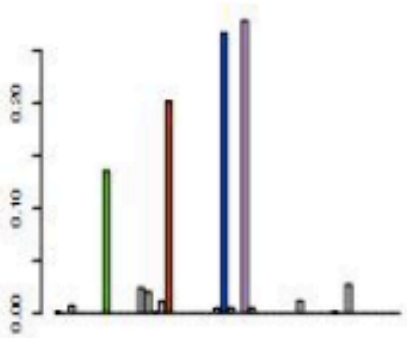
Chance and Statistical Significance in Protein and DNA Sequence Analysis

Samuel Karlin and Volker Brendel

Top words from the top topics (by term score)

sequence	measured	residues	computer
region	average	binding	methods
pcr	range	domains	number
identified	values	helix	two
fragments	different	cys	principle
two	size	regions	design
genes	three	structure	access
three	calculated	terminus	processing
cdna	two	terminal	advantage
analysis	low	site	important

Expected topic proportions



Abstract with the most likely topic assignments

Statistical approaches help in the determination of significant configurations in protein and nucleic acid sequence data. Three recent statistical methods are discussed: (i) score-based sequence analysis that provides a means for characterizing anomalies in local sequence text and for evaluating sequence comparisons; (ii) quantile distributions of amino acid usage that reveal general compositional biases in proteins and evolutionary relations; and (iii) *r*-scan statistics that can be applied to the analysis of spacings of sequence markers.

Top Ten Similar Documents

Exhaustive Matching of the Entire Protein Sequence Database
How Big Is the Universe of Exons?
Counting and Discounting the Universe of Exons
Detecting Subtle Sequence Signals: A Gibbs Sampling Strategy for Multiple Alignment
Ancient Conserved Regions in New Gene Sequences and the Protein Databases
A Method to Identify Protein Sequences that Fold into a Known Three- Dimensional Structure
Testing the Exon Theory of Genes: The Evidence from Protein Structure
Predicting Coiled Coils from Protein Sequences
Genome Sequence of the Nematode C. elegans: A Platform for Investigating Biology

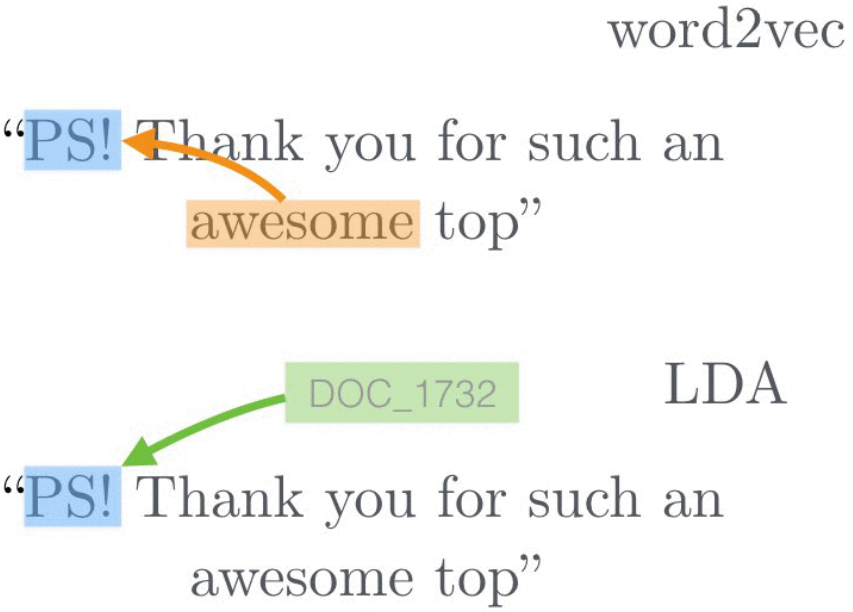
http://jialu.info/technical_notes/Topic_Model.pdf

LDA with deep learning

- LDA was develop on discrete inputs (words)
  - Modified to work with dense representation (word vectors)
    - “Gaussian LDA for Topic Models with Word Embeddings”
    - http://www.aclweb.org/anthology/P15-1077
- Modified network structure and loss function to include LDA traits
  - Use like a neural network(just like how we use crf in neural networks)
  - LDA2vec

LDA2Vec

- Word2Vec estimates words around a pivot word (local information)
- LDA estimates words from a document topic (global information)
- LDA2Vec estimates words using both information



https://multithreaded.stitchfix.com/blog/2016/05/27/lda2vec/

LDA2Vec

- Combines advantages from LDA and Word2Vec
  - Combine global info with local info

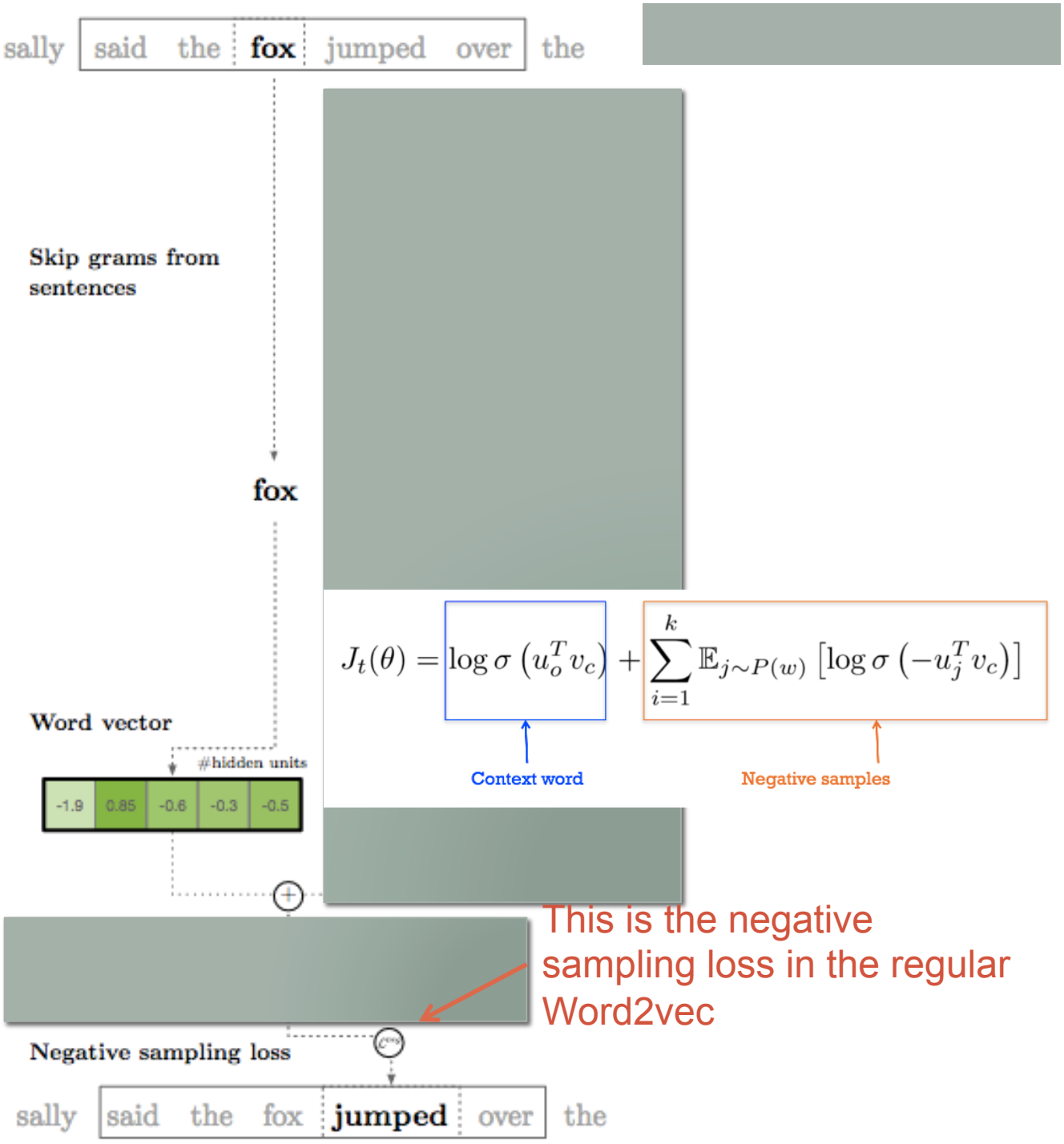
  Word German + airline topic = Lufthansa
  - Dense word vector but sparse document vectors

  Apple = [0.2 -1.2 -2.1 1.5]

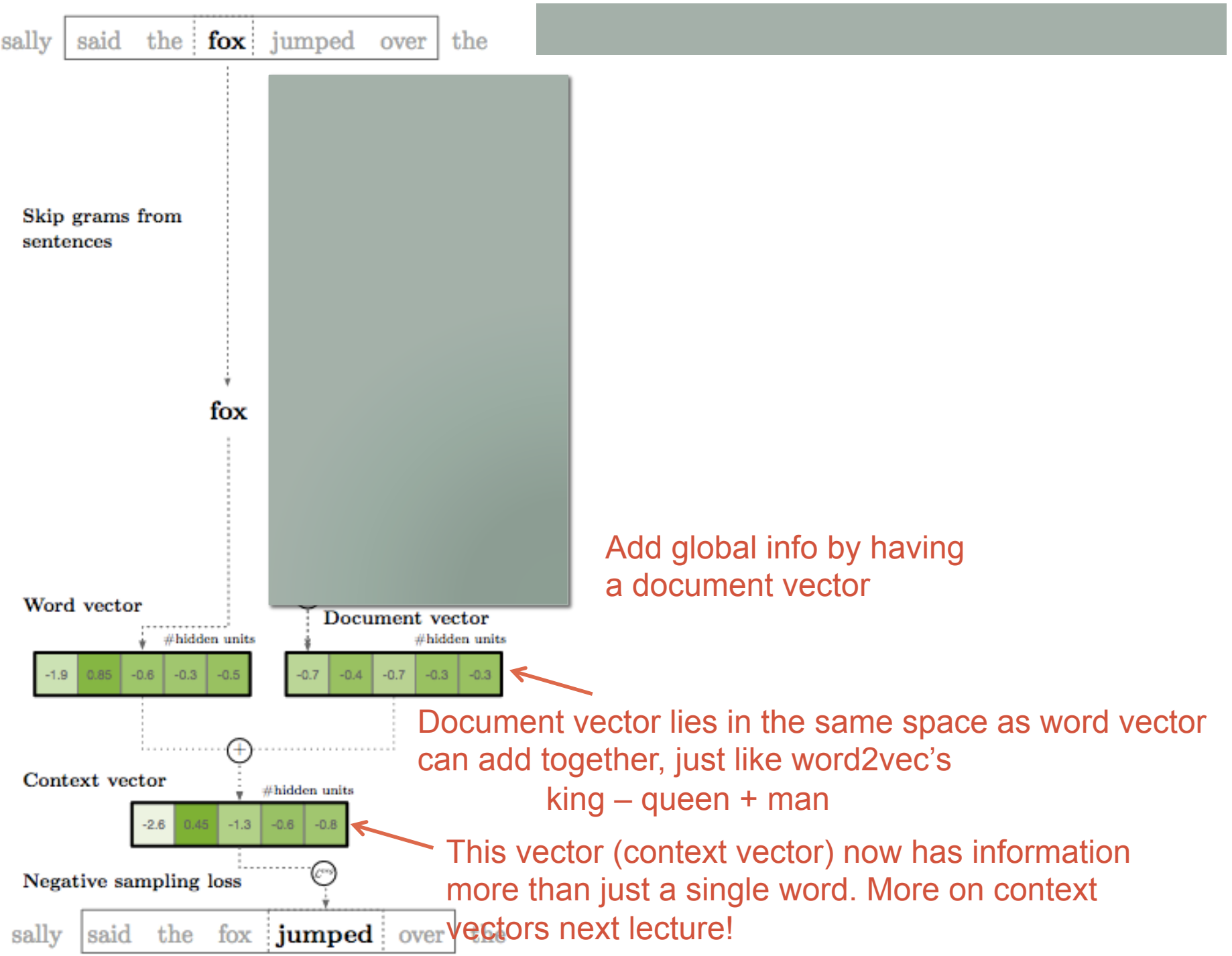
  Doc1 = [80% 20% 0% 0%]
  - Mixture of topics for each document vector give interpretability for humans

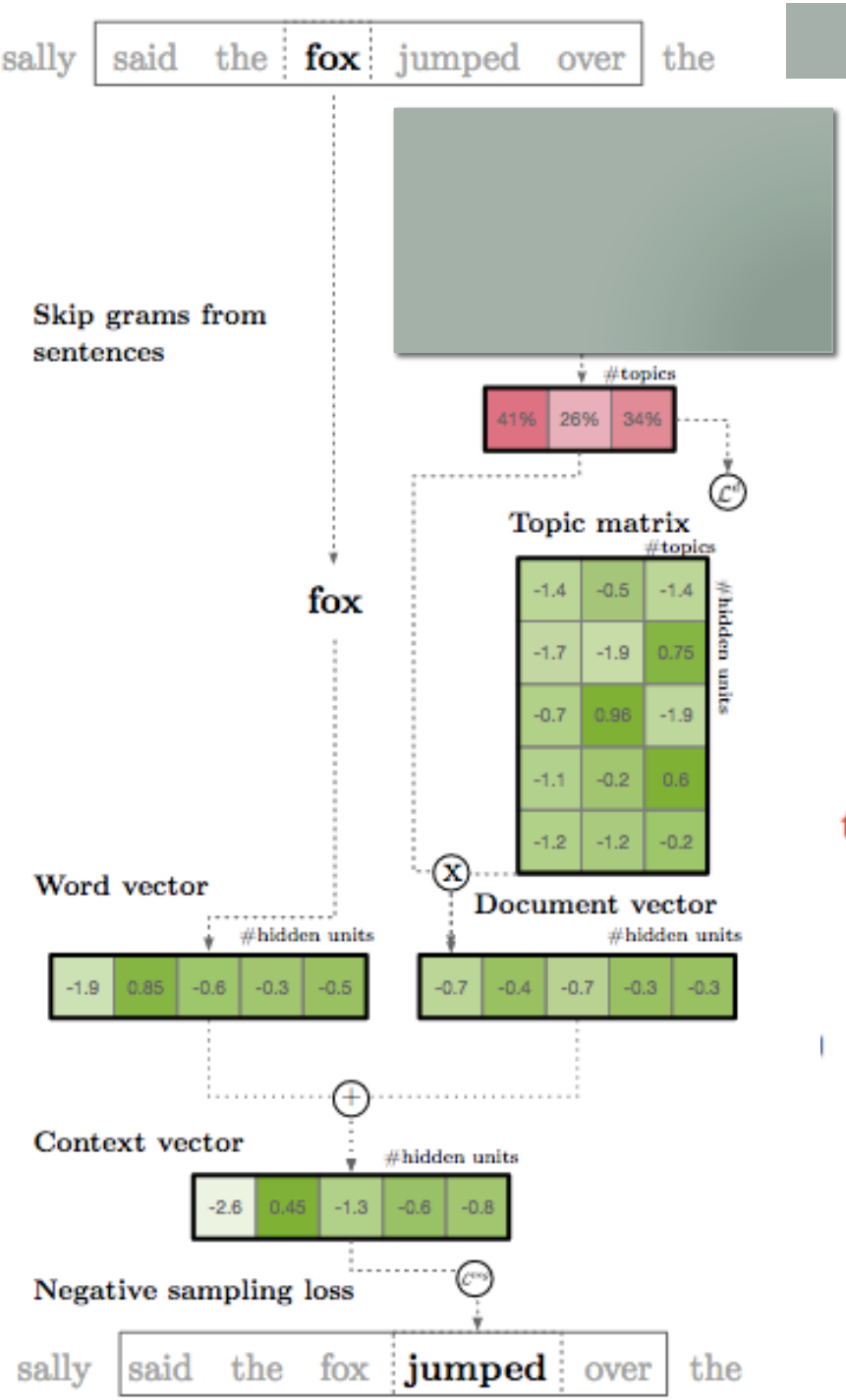
A typical skipgram
with negative sampling
A word is transformed to
a word vector

Maximize dot product
between pivot word and
context words
while reducing dot
product between pivot
word and negative
sampled words



This is the negative sampling loss in the regular Word2vec


sally said the fox jumped over the
Skip grams from sentences
fox
Word vector
#hidden units
-1.9 0.85 -0.6 -0.3 -0.5
Document vector
#hidden units
-0.7 -0.4 -0.7 -0.3 -0.3
Context vector
#hidden units
-2.6 0.45 -1.3 -0.6 -0.8
Negative sampling loss
sally said the fox jumped over the
Add global info by having a document vector
Document vector lies in the same space as word vector can add together, just like word2vec's
king – queen + man
This vector (context vector) now has information more than just a single word. More on context vectors next lecture!



Just like how we model a topic mixture
p(w) = Σp(topic)p(w|topic)
We can model a document vector as
Σp(topic)*TopicVector

Topic vector lies in the same space as word vectors

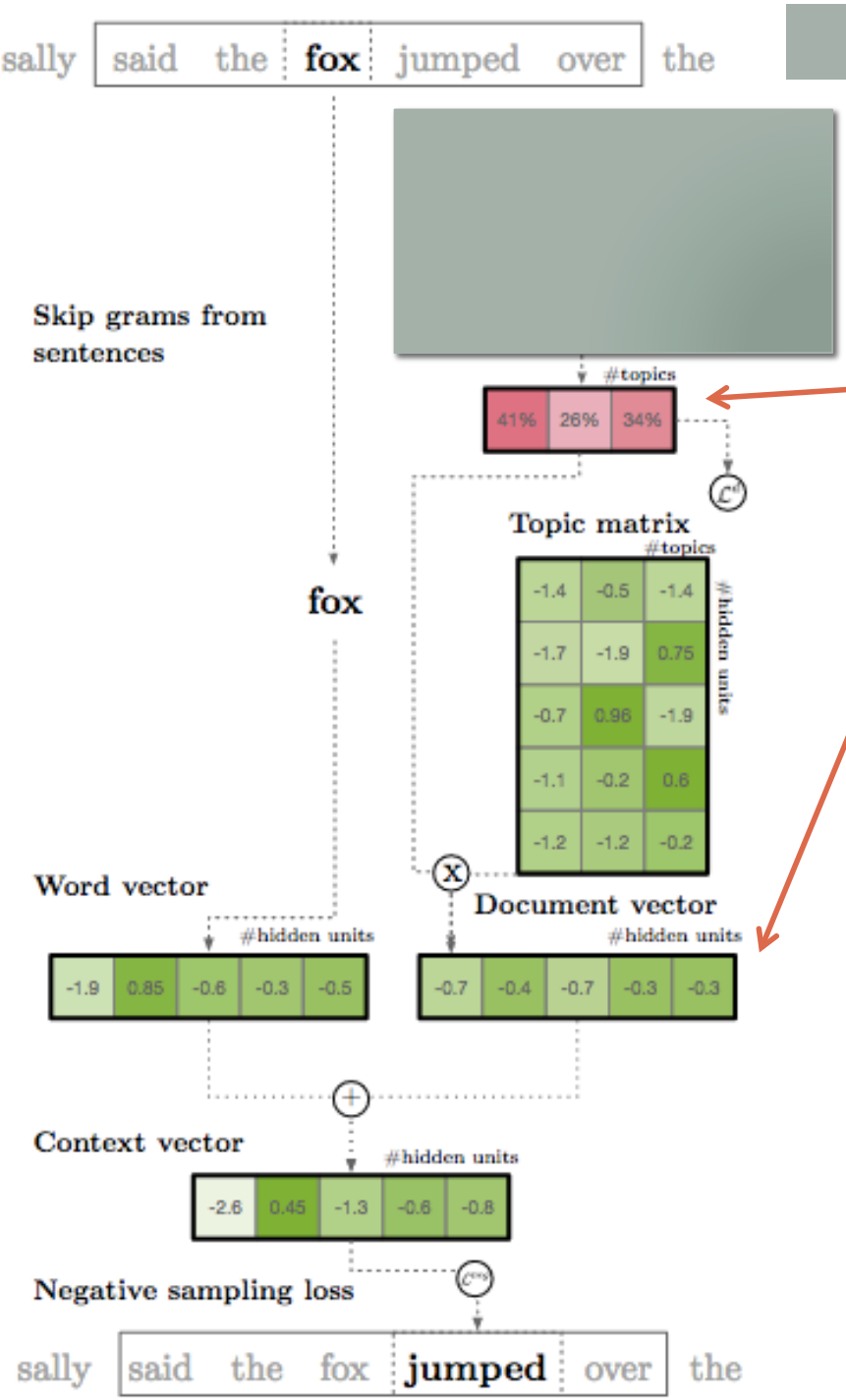
Adding word vectors yields document vectors that lies in the word vector space

topic 2 = "politics"

Milosevic
absentee
Indonesia
Lebanese
Isrealis
Karadzic

topic 1 = "religion"

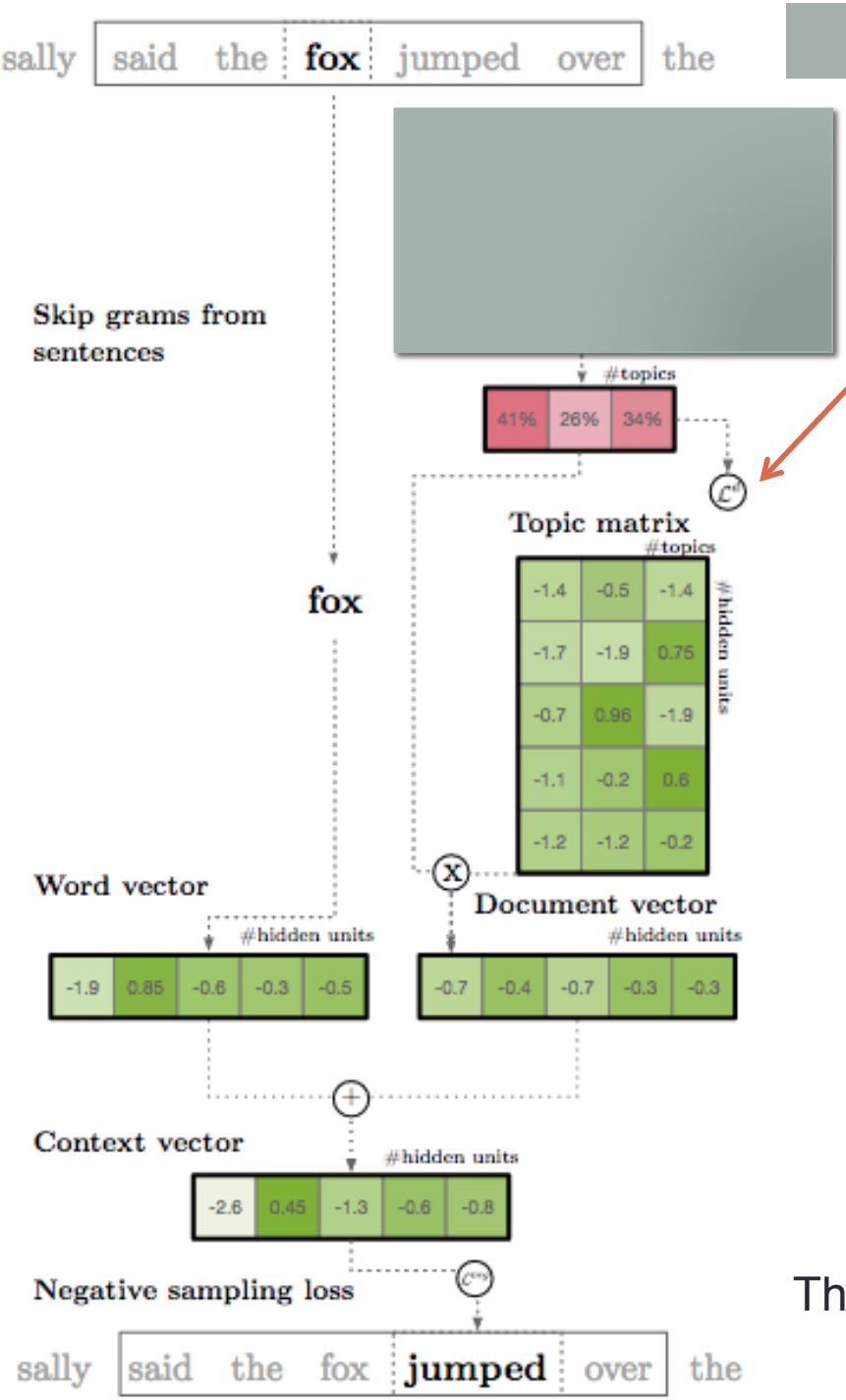
Trinitarian
baptismal
Pentecostals
Bede
schismatics
excommunication



This construction give explicit meaning to document vectors (valid probabilities)

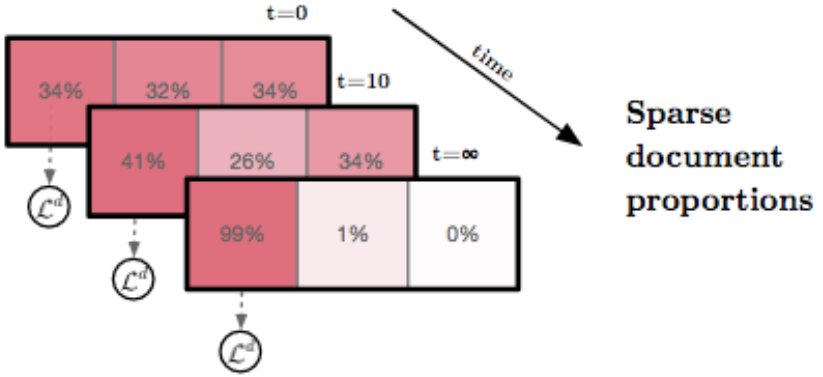
Instead of a bunch of numbers

However, we further prefer sparse topic probabilities for even better interpretability



Add a loss term to include the Dirichlet loss which prefers sparse topics

Mixture prob becomes spares as training evolves

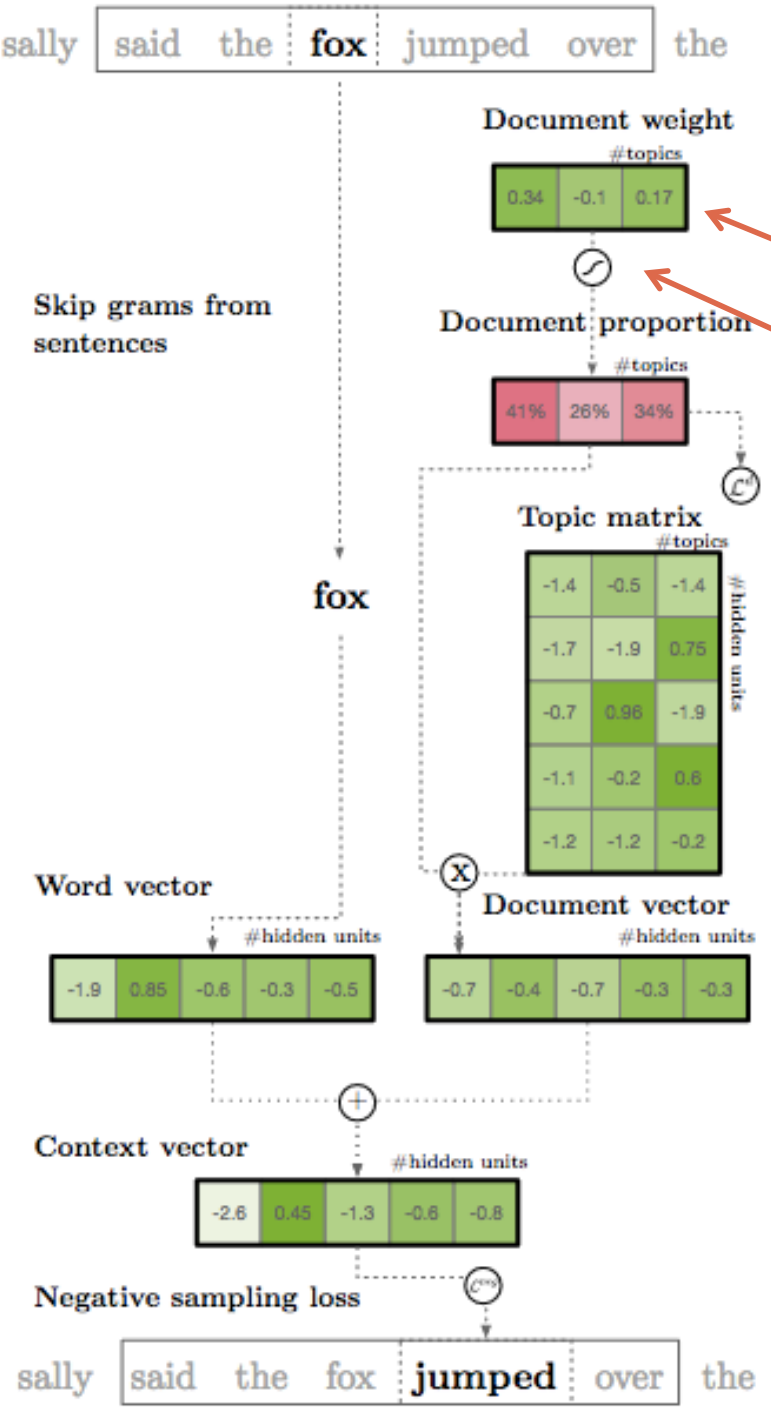


$$\mathcal{L}^d = \lambda \Sigma_{jk} (\alpha - 1) \log p_{jk}$$

$$\alpha = n^{-1}$$

n is number of topics

This loss prefers a couple of high value p



To ensure the topics proportion is a prob, Document weight is initialized and learned

The weight is transformed to prob using a softmax

Overall loss function

- Loss from skipgram and Dirichlet

Dirichlet loss

$$\mathcal{L} = \Sigma_{word\ pairs\ (i,j)}[\sigma(\vec{c_j} \cdot \vec{w_i}) + \sigma(-\vec{c_j} \cdot \vec{w_{negative}})] + \Sigma_{documents\ k}\ log\ q(d_k|\alpha)$$

Skipgram loss

$$\vec{c_j} = \vec{w_j} + \vec{d_j}$$

Context = word + document vector

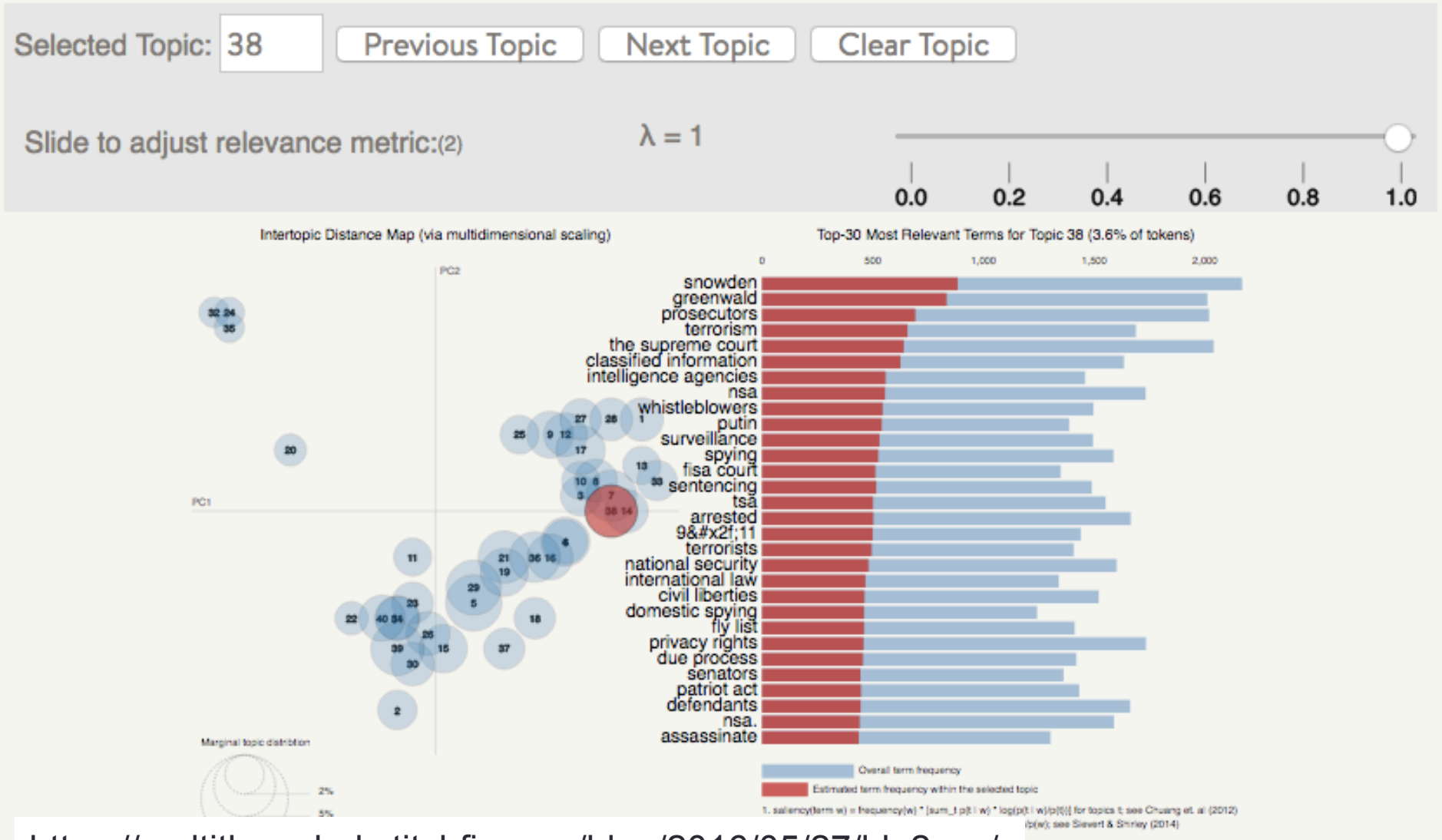
$$\vec{d_j} = a_{j0} \cdot \vec{t_0} + a_{j1} \cdot \vec{t_1} + \ldots$$

Document vector is mixture of topics

$$q(d_k|\alpha) = \Sigma_k(\alpha_k - 1)loga_k$$

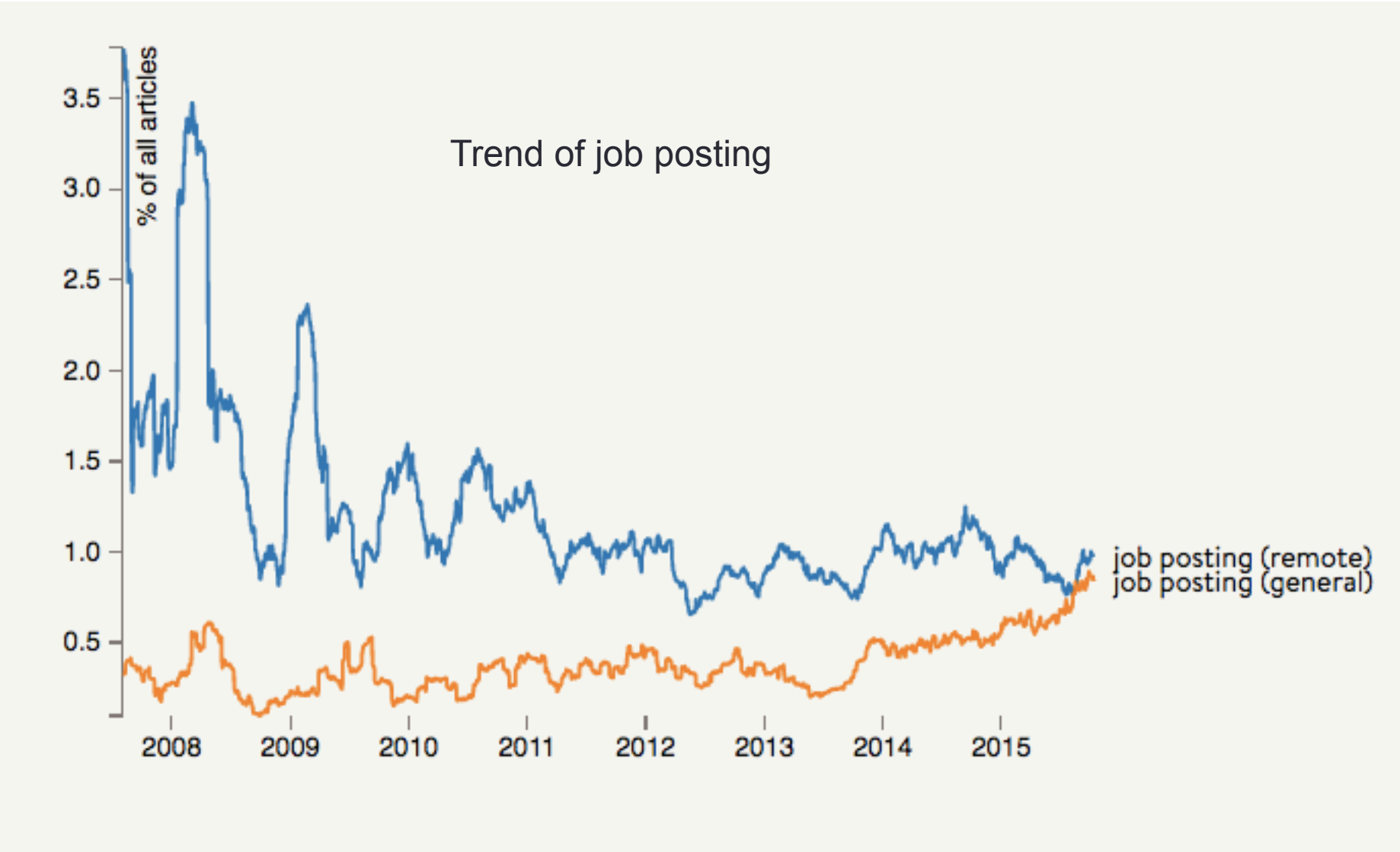
Dirichlet likelihood term

LDA2Vec example

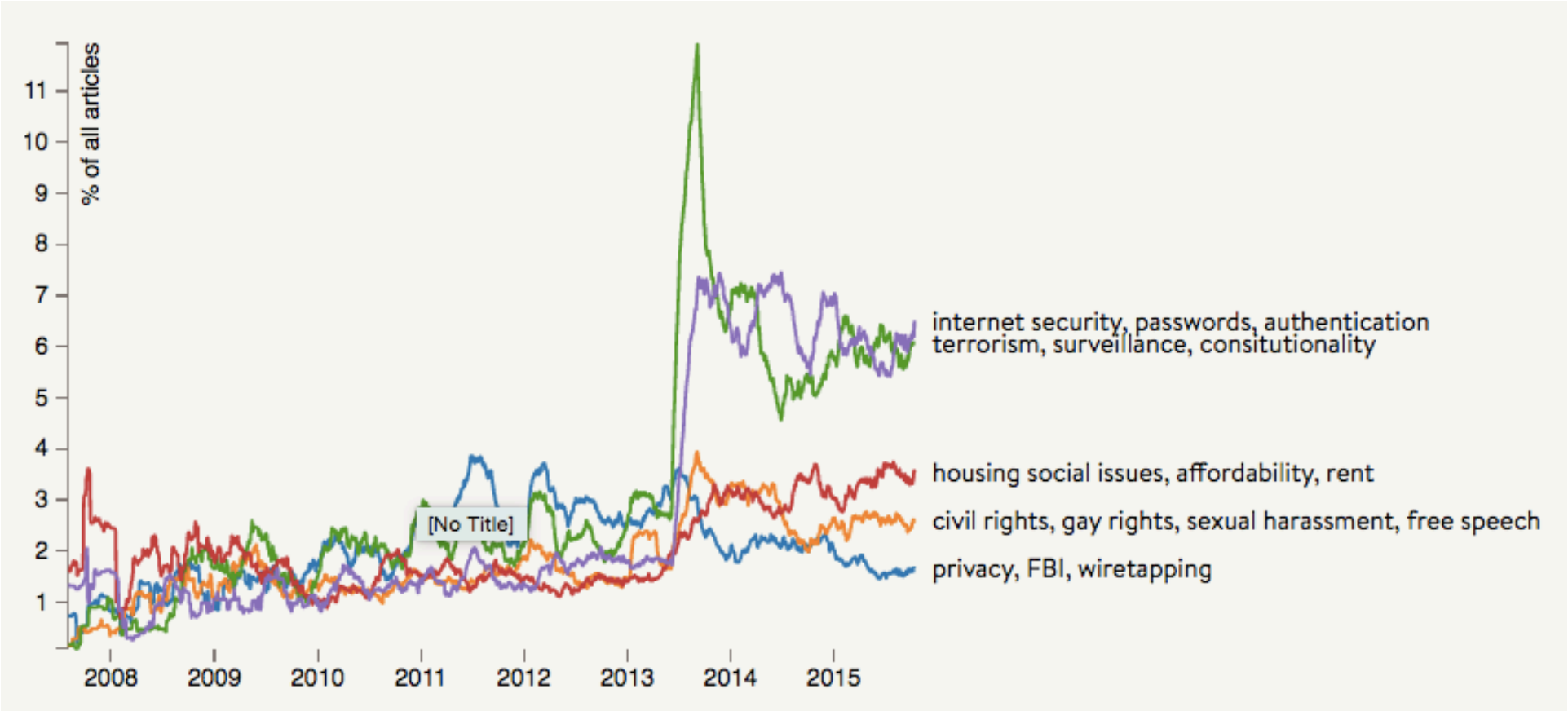


https://multithreaded.stitchfix.com/blog/2016/05/27/lda2vec/

LDA2Vec example



Politics trend



LDA2Vec as a classification model

- Maybe not…
  - Model that offers human interpretability usually is worse
- But it's great for
  - Extract insights from trends by visualizations
  - Completely unsupervised model
    - Except for the need to specify number of topics and look through each topic words to determine the topic label
- This model gives a good example of
  - The design decisions for model building can be complex but logically motivated
  - Combining loss functions to get the desired behavior

Summary

- Text classification task
- Bag of words model
  - Naïve Bayes
  - Topic Models
    - Latent topic models (LDA)
  - LDA2vec
- Context vectors
  - More next time