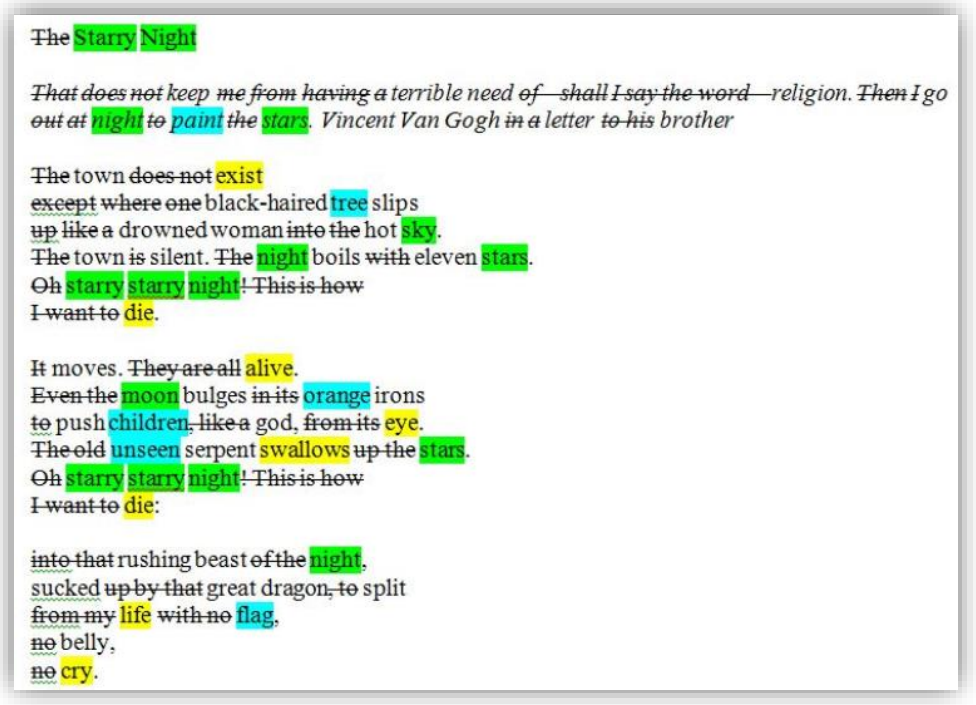
Topic Modeling Method

Why Topic Modeling

Topic modeling

- Topic modeling의 필요성
  - ➢ Text 문서의 의미적인 분석을 하기 위함



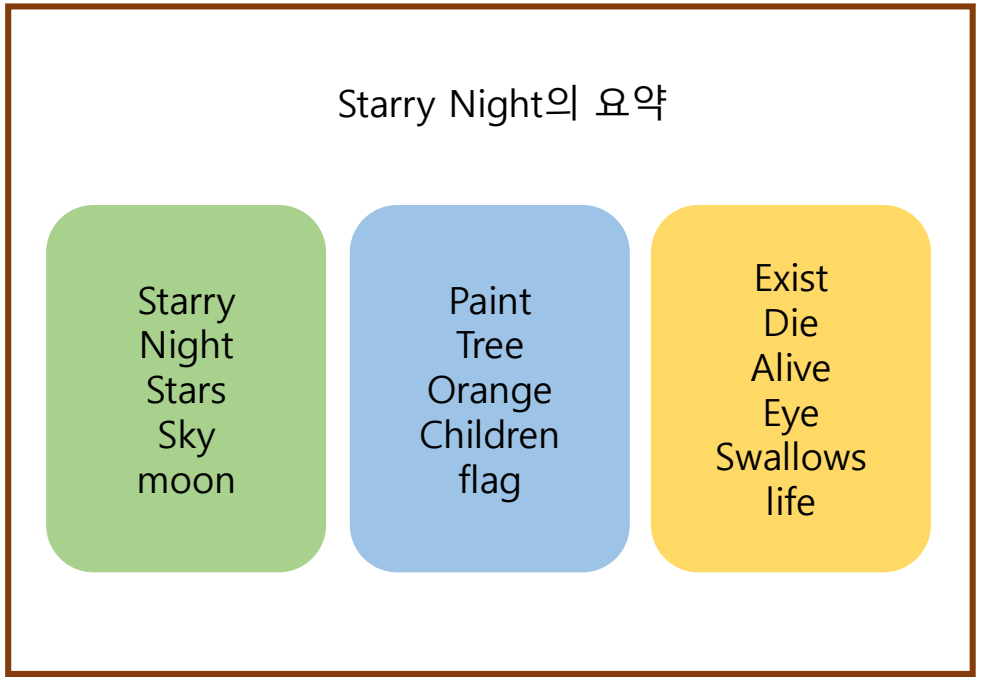
The Starry Night

That does not keep me from having a terrible need of—shall I say the word—religion. Then I go out at night to paint the stars. Vincent Van Gogh in a letter to his brother

The town does not exist
except where one black-haired tree slips
up like a drowned woman into the hot sky.
The town is silent. The night boils with eleven stars.
Oh starry starry night! This is how
I want to die.

It moves. They are all alive.
Even the moon bulges in its orange irons
to push children, like a god, from its eye.
The old unseen serpent swallows up the stars.
Oh starry starry night! This is how
I want to die:

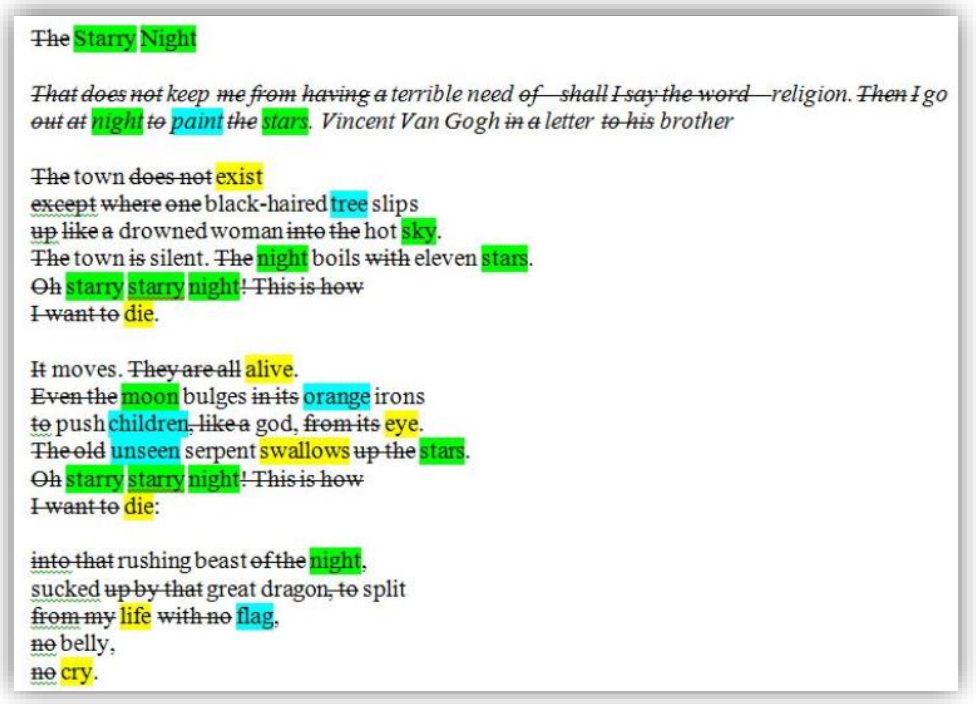
into that rushing beast of the night,
sucked up by that great dragon, to split
from my life with no flag,
no belly,
no cry.



Why Topic Modeling

Topic modeling

- Topic modeling의 필요성
  - ➢ Text 문서의 의미적인 분석을 하기 위함



The Starry Night

That does not keep me from having a terrible need of—shall I say the word—religion. Then I go out at night to paint the stars. Vincent Van Gogh in a letter to his brother

The town does not exist
except where one black-haired tree slips
up like a drowned woman into the hot sky.
The town is silent. The night boils with eleven stars.
Oh starry starry night! This is how
I want to die.

It moves. They are all alive.
Even the moon bulges in its orange irons
to push children, like a god, from its eye.
The old unseen serpent swallows up the stars.
Oh starry starry night! This is how
I want to die:

into that rushing beast of the night,
sucked up by that great dragon, to split
from my life with no flag,
no belly,
no cry.

Why Topic Modeling

Topic modeling

- Topic modeling의 필요성
  - ➢ Text 문서 집합에서 의미적으로 요약

What Is Topic Modeling

Topic modeling

- Topic modeling
  - ➢ 문서 집합의 추상적인 '토픽'를 발견하기 위한 통계적 모델
  - ➢ 각 텍스트 본문의 숨겨진 의미구조를 발견
  - ➢ 의미 구조
    - ➢ 의미를 탐색하기 위해 문서 내 단어들의 co-occurrence를 분석

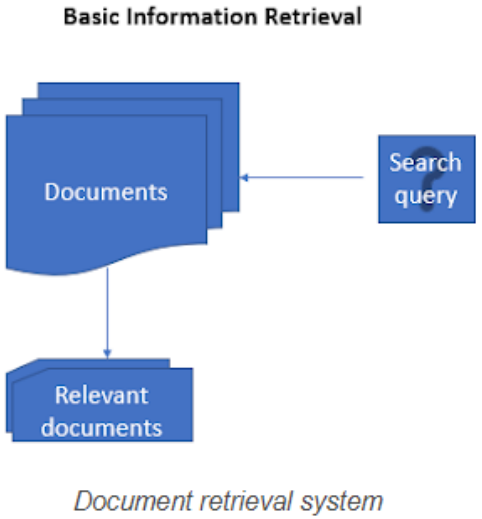
배				
	단어 1	단어 2	단어 3	단어 4
문서 1	**배**	타다		화물
문서 2	**배**		먹다	맛있게

중국집 메뉴				
	단어 1	단어 2	단어 3	단어 4
문서 1	중국집	메뉴	짜장면	짬뽕
문서 2	?	?	짜장면	짬뽕

Topic-based Vector Space Model

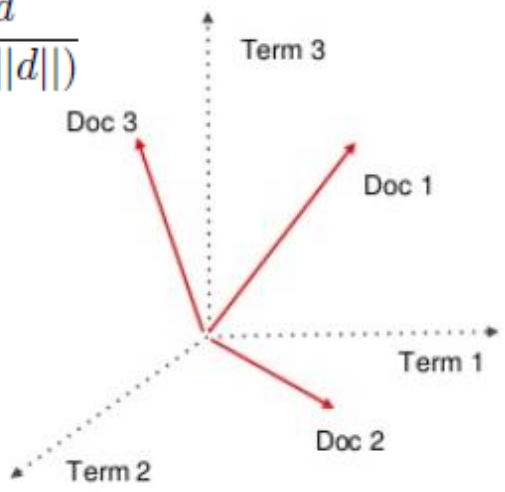
Keyword Search

- 텍스트 문서를 벡터로 표현하는 모델
  - 대량의 문서들이 주어지고, term과 비슷한 문서를 찾는 것이 목적
  - 문서들을 벡터화
    - Co-occurrence 개념을 사용하여 의미적으로 유사한 문서 벡터들을 가깝게 위치
    - Document-term matrix로부터 산출된 문서 벡터들의 거리를 Cosine Similarity 로 구함
    - 한계점: 너무 sparse 한 벡터를 사용하여 정보 손실



	Term 1	Term 2	Term 3
Doc 1	0.4	0.1	0.6
Doc 2	0.3	0.5	0.0
Doc 3	0.0	0.2	0.6

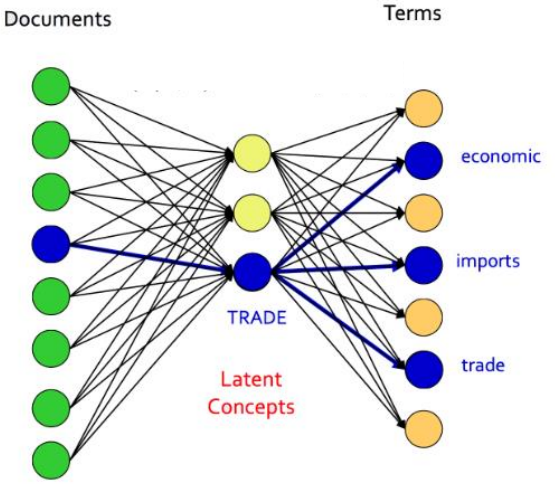
$$cos(q,d) = \frac{q.d}{(||q||.||d||)}$$



Latent Semantic Indexing

Dimension Reduction

- Term-Document matrix 중 핵심이 되는 차원을 뽑아내어 차원을 낮추는 기법
  - SVD(Singular Value Decomposition) 기법을 사용하여 행렬 분해: Topic 열은 직교
  - 값이 작은 요소들을 잘라내어 원래 크기보다 적은 matrix로 표현 가능
  - 문서와 단어들을 잇는 잠재 토픽들(latent topic)이 있음을 가정



$$A_{m\times n} = U_{m\times r}\,\Sigma_{r\times r}\,V^{T}_{r\times n}$$

U: $m\times m$ 직교행렬 $(AA^T=U(\Sigma\Sigma^T)U^T)$
V: $n\times n$ 직교행렬 $(A^TA=V(\Sigma^T\Sigma)V^T)$
Σ: $m\times n$ 직사각 대각행렬

	term						
doc	1	1	0	0	0	0	0
	0	0	1	1	1	0	0
	0	1	0	0	0	1	0
	0	0	1	0	0	0	1
	0	0	0	1	1	0	1
	1	0	1	0	1	1	1

6×6

=

	topic					
doc	0.12	0.57	-0.32	0.00	-0.71	-0.24
	0.44	-0.36	-0.41	0.71	0.00	-0.08
	0.12	0.57	-0.32	0.00	0.71	-0.24
	0.33	-0.07	0.56	0.00	0.00	-0.75
	0.44	-0.36	-0.41	-0.71	0.00	-0.08
	0.69	0.30	0.37	0.00	0.00	0.55

6×6

	topic					
	2.98	0.00	0.00	0.00	0.00	0.00
	0.00	1.88	0.00	0.00	0.00	0.00
	0.00	0.00	1.36	0.00	0.00	0.00
	0.00	0.00	0.00	1.00	0.00	0.00
	0.00	0.00	0.00	0.00	1.00	0.00
	0.00	0.00	0.00	0.00	0.00	0.87

6×6
내림차순 정렬

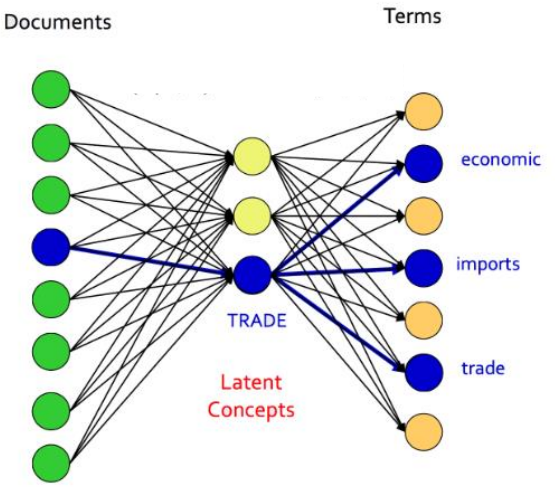
	doc						
topic	0.27	0.46	0.04	0.00	-0.71	0.35	0.30
	0.08	0.61	-0.47	0.00	0.00	-0.56	-0.30
	0.49	-0.07	0.38	0.71	0.00	-0.33	0.00
	0.30	-0.38	-0.61	0.00	0.00	-0.18	0.60
	0.53	-0.22	-0.34	0.00	0.00	0.44	-0.60
	0.27	0.46	0.04	0.00	0.71	0.35	0.30
	0.49	-0.07	0.38	-0.71	0.00	-0.33	0.00

6×6

Latent Semantic Indexing

Dimension Reduction

- Term-Document matrix 중 핵심이 되는 차원을 뽑아내어 차원을 낮추는 기법
  - SVD(Singular Value Decomposition) 기법을 사용하여 행렬 분해: Topic 열은 independent
  - 값이 작은 요소들을 잘라내어 원래 크기보다 적은 matrix로 표현 가능
  - 문서와 단어들을 잇는 잠재 토픽들(latent topic)이 있음을 가정



$$A_{m\times n} = U_{m\times r}\,\Sigma_{r\times r}\,V^{T}_{r\times n}$$

U: $m\times m$ 직교행렬 $(AA^T=U(\Sigma\Sigma^T)U^T)$
V: $n\times n$ 직교행렬 $(A^TA=V(\Sigma^T\Sigma)V^T)$
Σ: $m\times n$ 직사각 대각행렬

doc \ term							
	1	1	0	0	0	0	0
	0	0	1	1	1	0	0
	0	1	0	0	0	1	0
	0	0	1	0	0	0	1
	0	0	0	1	1	0	1
	1	0	1	0	1	1	1

6×6

=

doc \ topic						
	0.12	0.57	-0.32	0.00	-0.71	-0.24
	0.44	-0.36	-0.41	0.71	0.00	-0.08
	0.12	0.57	-0.32	0.00	0.71	-0.24
	0.33	-0.07	0.56	0.00	0.00	-0.75
	0.44	-0.36	-0.41	-0.71	0.00	-0.08
	0.69	0.30	0.37	0.00	0.00	0.55

6×6

topic \ topic						
	2.98	0.00	0.00	0.00	0.00	0.00
	0.00	1.88	0.00	0.00	0.00	0.00
	0.00	0.00	1.36	0.00	0.00	0.00
	0.00	0.00	0.00	1.00	0.00	0.00
	0.00	0.00	0.00	0.00	1.00	0.00
	0.00	0.00	0.00	0.00	0.00	0.87

6×6

내림차순 정렬

topic \ term							
	0.27	0.46	0.04	0.00	-0.71	0.35	0.30
	0.08	0.61	-0.47	0.00	0.00	-0.56	-0.30
	0.49	-0.07	0.38	0.71	0.00	-0.33	0.00
	0.30	-0.38	-0.61	0.00	0.00	-0.18	0.60
	0.53	-0.22	-0.34	0.00	0.00	0.44	-0.60
	0.27	0.46	0.04	0.00	0.71	0.35	0.30
	0.49	-0.07	0.38	-0.71	0.00	-0.33	0.00

6×6

Latent Semantic Indexing

Dimension Reduction

- Term-Document matrix 중 핵심이 되는 차원을 뽑아내어 차원을 낮추는 기법
  - SVD(Singular Value Decomposition) 기법을 사용하여 행렬 분해
  - 값이 작은 요소들을 잘라내어 원래 크기보다 적은 matrix로 표현 가능
  - 문서와 단어들을 잇는 잠재 토픽들(latent topic)이 있음을 가정

U2*Σ2 : 각 문서에 있는 토픽의 분포

$$\begin{vmatrix} 0.12 & 0.57 \\ 0.44 & -0.36 \\ 0.12 & 0.57 \\ 0.33 & -0.07 \\ 0.44 & -0.36 \\ 0.69 & 0.30 \end{vmatrix} \begin{vmatrix} 2.98 & 0.00 \\ 0.00 & 1.88 \end{vmatrix} =$$

	Topic 1	Topic 2
Doc 1	0.3573	1.0707
Doc 2	1.31	-0.6762
Doc 3	0.3573	1.0707
Doc 4	0.9825	-0.1315
Doc 5	1.31	-0.6762
Doc 6	2.0543	0.5635

Σ2*V2T: 각 topic에 있는 단어들의 분포

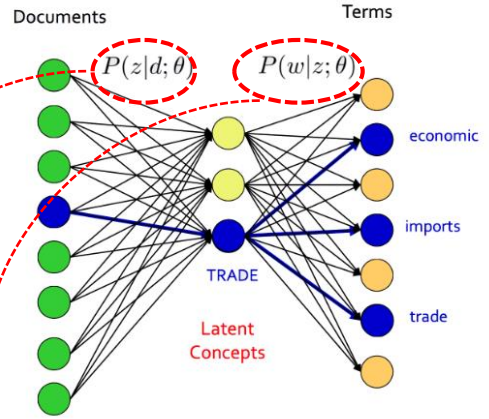
$$\begin{vmatrix} 2.98 & 0.00 \\ 0.00 & 1.88 \end{vmatrix} \begin{vmatrix} 0.27 & 0.46 \\ 0.08 & 0.61 \\ 0.49 & -0.07 \\ 0.30 & -0.38 \\ 0.53 & -0.22 \\ 0.27 & 0.46 \\ 0.49 & -0.07 \end{vmatrix}$$

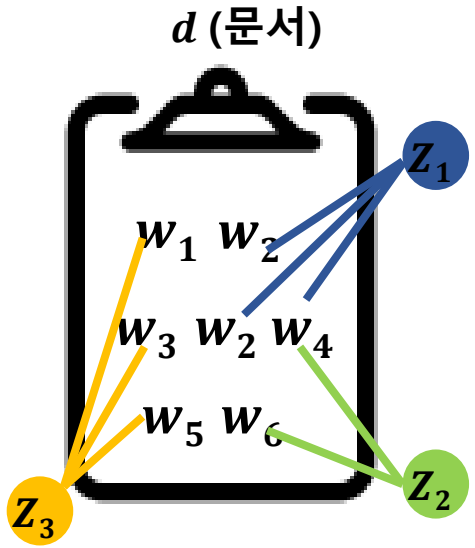
	cute	kitty	eat	rice	cake	hamster	bread
#1	0.8038	0.2382	1.4588	0.8932	1.5779	0.8038	1.4588
#2	0.8641	1.1458	-0.1315	-0.7138	-0.4132	0.8641	-0.1315

Probabilistic Latent Semantic Indexing

Topic Modeling

- 문서 내에 특정 용어가 등장한 **확률**을 기반으로 하여 구축
  - LSA의 SVD(Singular Value Decomposition) 기법은 행렬 요소에 음수 값이 나오므로 사용 불가
  - 문서와 단어들을 잇는 잠재 토픽들(latent topic)이 있음을 가정
  - 단어와 문서의 동시 출현 확률 모델링
  - 문서 및 단어 출현 확률: 한 문서 등장 시 특정 단어 출현 확률×토픽 등장 시 특정 단어 출현 확률





$$P(w,d) = p(d) * P(w|d) \quad \cdots ①$$

$$P(w|d) = p(z|d) * p(w|z)$$

$$\sum p(z|d) * p(w|z)$$

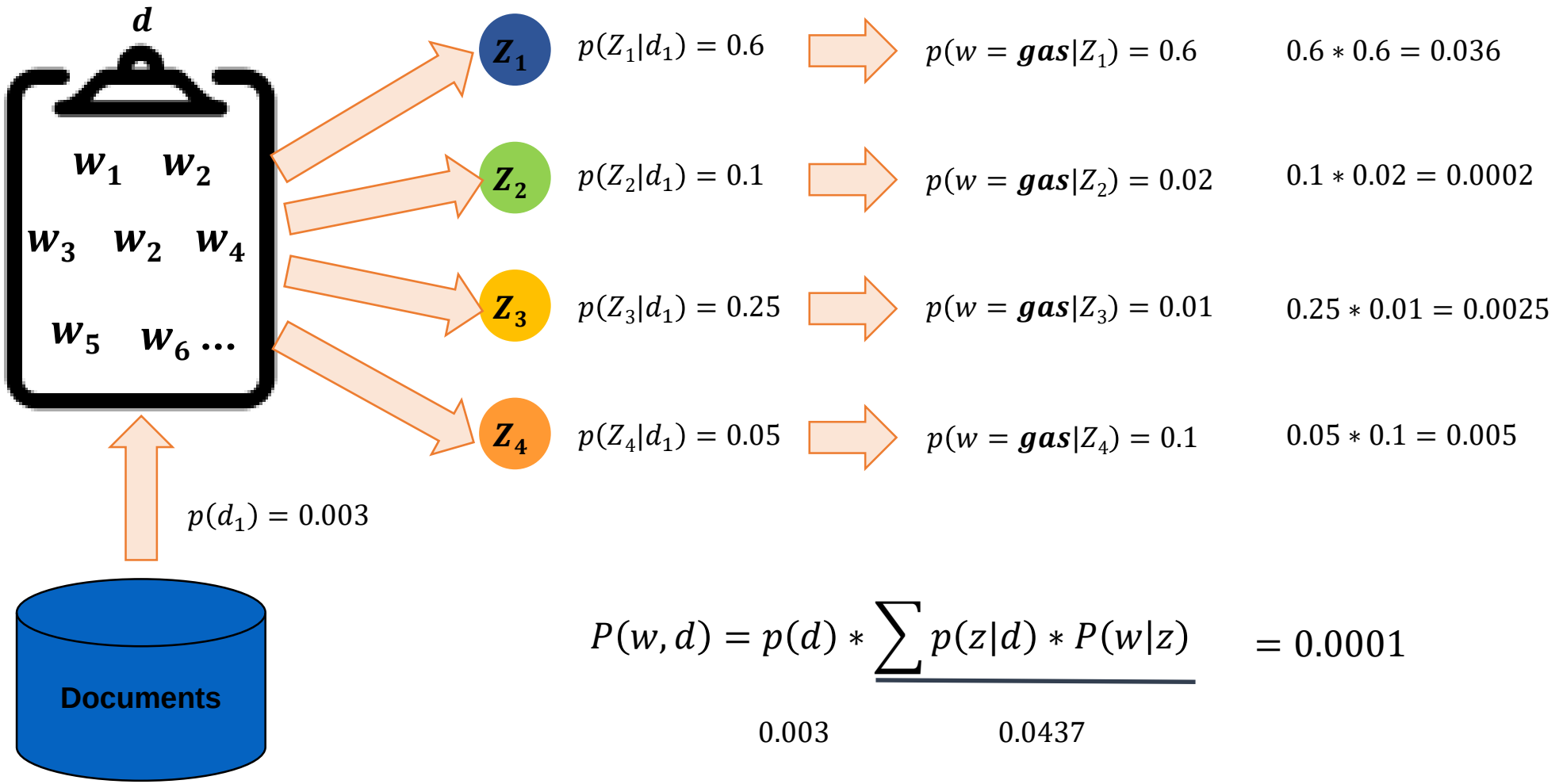
$$P(w,d) = p(d) * \sum p(z|d) * p(w|z) \quad \cdots ②$$

- P(z|d): 문서 하나가 주어졌을 때 특정 토픽이 나타날 확률
- P(w|z): 토픽 하나가 정해졌을 때 특정 단어가 나타날 확률

Probabilistic Latent Semantic Indexing

Topic Modeling

- 예시



$$P(w,d) = p(d) * \sum p(z|d) * P(w|z) = 0.0001$$

$p(d)$: 0.003

$\sum p(z|d) * P(w|z)$: 0.0437

Probabilistic Latent Semantic Indexing

Topic Modeling

- 목적식 최대화: 각 문서에 대해 특정 단어가 나올 확률분포를 최대한 최적화
- EM 알고리즘 활용 (모든 값 랜덤으로 초기화)

$$L = \prod_{i=1}^{m}\prod_{j=1}^{n} p(w_i, d_j)^{n(w_i,d_j)}$$

$$= \prod_{i=1}^{m}\prod_{j=1}^{n}\left\{\sum_{l=1}^{k} p(d_j|z_l)p(z_l)p(w_i|z_l)\right\}^{n(w_i,d_j)}$$

n(w_i, d_j) =j번째 문서에 i번째 단어가 등장한 횟수

m개 단어, n개 문서, k개 토픽(토픽)

- E-Step: Posterior probability of latent variables (concepts)

$$p(z|d,w) = \frac{P(z)P(d|z)P(w|z)}{\sum_{z' \in Z} P(z')P(d|z')P(w|z')}$$

Probability that the occurence of term w in document d can be "explained" by concept z

- M-Step: Parameter estimation based on "completed" statistics

$$P(w|z) = \frac{\sum_{d \in D} n(d,w)P(z|d,w)}{\sum_{d \in D, w' \in W} n(d,w')P(z|d,w')}$$

how often is term w associated with concept z ?

$$P(d|z) = \frac{\sum_{w \in W} n(d,w)P(z|d,w)}{\sum_{d' \in D, w \in W} n(d',w)P(z|d',w)}$$

how often is document d associated with concept z ?

$$P(z) = \frac{\sum_{d \in D, w \in W} n(d,w)P(z|d,w)}{\sum_{d \in D, w \in W} n(d,w)}$$

how prevalent is the concept z ?

Probabilistic Latent Semantic Indexing

Topic Modeling

- PLSA 업데이트 및 결과
- 한계
  - ➢ 문서 내 토픽 분포는 고려 하지 않음

	Doc 1	Doc 2	Doc 3	Doc 4	Doc 5	Doc 6
Baseball	1	2	0	0	0	0
Basketball	3	1	0	0	0	0
Boxing	2	0	0	0	0	0
Money	3	3	2	3	2	4
Interest	0	0	3	2	0	0
Rate	0	0	4	1	0	0
Democrat	0	0	0	0	4	3
Republican	0	0	0	0	2	1
Cocus	0	0	0	0	3	2
President	0	0	1	0	2	3

Input

$P(z)$

Topic 1	Topic 2	Topic 3
0.525	0.407	0.068

$P(d|z)$

	Topic 1	Topic 2	Topic 3
Doc 1	0.020	0.008	0.048
Doc 2	0.294	0.255	0.329
Doc 3	0.204	0.138	0.178
Doc 4	0.200	0.146	0.007
Doc 5	0.186	0.196	0.233
Doc 6	0.096	0.257	0.205

$P(w|z)$

	Topic 1	Topic 2	Topic 3
Term 1	0.022	0.016	0.010
Term 2	0.018	0.133	0.166
Term 3	0.242	0.058	0.133
Term 4	0.123	0.088	0.145
Term 5	0.016	0.030	0.044
Term 6	0.020	0.167	0.056
Term 7	0.147	0.129	0.201
Term 8	0.188	0.156	0.039
Term 9	0.146	0.114	0.008
Term 10	0.077	0.110	0.199

Topic 1	Topic 2	Topic 3
0.456	0.281	0.263

	Topic 1	Topic 2	Topic 3
Doc 1	0.000	0.000	0.600
Doc 2	0.000	0.000	0.400
Doc 3	0.000	0.625	0.000
Doc 4	0.000	0.375	0.000
Doc 5	0.500	0.000	0.000
Doc 6	0.500	0.000	0.000

	Topic 1	Topic 2	Topic 3
Baseball	0.000	0.000	0.200
Basketball	0.000	0.000	0.267
Boxing	0.000	0.000	0.133
Money	0.231	0.313	0.400
Interest	0.000	0.312	0.000
Rate	0.000	0.312	0.000
Democrat	0.269	0.000	0.000
Republican	0.115	0.000	0.000
Cocus	0.192	0.000	0.000
President	0.192	0.063	0.000

Latent Dirichlet Allocation

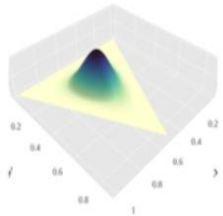
Topic Modeling

- LDA: 주어진 문서에 대하여 각 문서에 어떤 토픽들이 어떤 분포로 존재하는지에 대한 확률 모형
  - ➢ 현재 문서의 단어들로 토픽별 단어의 분포, 문서별 토픽의 분포 두 가지 모두 추정
  - ➢ 디리클레-다항 분포 사용 (k개의 토픽- hyperparameter)
  - ➢ 깁스 샘플링 활용

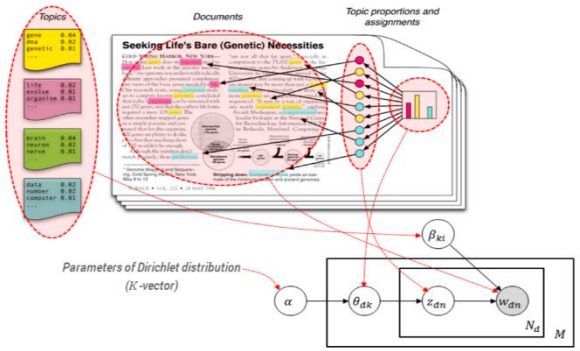
문서 생성 과정

1. 문서의 토픽을 정한다 (문서의 토픽 분포)
2. 어떤 단어를 쓸지 정한다 (토픽의 단어 분포)
3. 문서 생성

디리클레 분포



- 1에 가까워질 수록 문서에 많은 토픽이 포함 (a=0.1)
- 1에 가까워질 수록 토픽에 많은 단어 포함(a=0.01)



단어 생성 가정

- LDA inference
  - ➢ 문서 내 단어를 가지고 토픽의 단어 분포, 문서의 토픽 분포 추정
- 토픽의 단어 분포와 문서의 토픽 분포의 결합 확률이 커지도록 함
- 결과: 문서 내 단어들의 확률 분포 최대한 최적화

깁스 샘플링

- 나머지 단어는 고정 시킨 채 한 단어만을 빼고 분포 추론
- 단어 하나씩 제외 시키고 추론하면 제외된 단어에 대해 전체 분포 추정
- 모든 단어에 대하여 두 가지 분포 업데이트

Latent Dirichlet Allocation

Topic Modeling

- 6개의 문서와 해당 단어

A	Cute	Kitty				
B	Eat	Rice	Or	Cake		
C	Kitty	And	Hamster			
D	Eat	Bread				
E	Rice	Bread	And	Cake		
F	Cute	Hamster	Eats	Bread	And	Cake

- 문서 별 단어

Latent Dirichlet Allocation

Topic Modeling

- 1. 문서의 각 단어에 대해 임의로 토픽 선정 (k=2 : 2개의 토픽)

W	cute	kit	eat	rice	cake	kit	ham	eat	bre	rice	bre	cake	cute	ham	eat	bre	cake
Z	#1	#2	#1	#2	#1	#2	#2	#1	#1	#1	#2	#1	#2	#2	#2	#1	#1

- 2. 문서별 토픽 분포 형성 (∂=**0.1**)

A	Cute	Kitty				
B	Eat	Rice	Or	Cake		
C	Kitty	And	Hamster			
D	Eat	Bread				
E	Rice	Bread	And	Cake		
F	Cute	Hamster	Eats	Bread	And	Cake

θ	A	B	C	D	E	F
#1	1.1	2.1	0.1	2.1	2.1	2.1
#2	1.1	1.1	2.1	0.1	1.1	3.1

- 3. 토픽별 단어 분포 형성(∂=**0.01**)

φ	cute	kit	eat	rice	cake	ham	bre	SUM
#1	1.001	0.001	2.001	1.001	3.001	0.001	2.001	9.007
#2	1.001	2.001	1.001	1.001	0.001	2.001	1.001	8.007

Latent Dirichlet Allocation

Topic Modeling

문서 A 에 속하는 어떤 단어 m이 토픽 j에 속할 확률

= 문서 d에 속하는 모든 토픽 중 토픽 j가 차지하는 비중 * 토픽 j에 속하는 모든 단어 중 단어 m이 차지하는 비중

- 4. 문서의 첫 번째 단어 cute를 골라 뺀다

W	cute	kit	eat	rice	cake	kit	ham	eat	bre	rice	bre	cake	cute	ham	eat	bre	cake
Z	?	#2	#1	#2	#1	#2	#2	#1	#1	#1	#2	#1	#2	#2	#2	#1	#1

- 4-1. 문서별 토픽 분포 형성 (***∂*=0.1**)

θ	A	B	C	D	E	F
#1	**0.1**	2.1	0.1	2.1	2.1	2.1
#2	1.1	1.1	2.1	0.1	1.1	3.1

- 4-2 . 토픽별 단어 분포 형성(***∂*=0.01**)

A	?	Kitty				
B	Eat	Rice	Or	Cake		
C	Kitty	And	Hamster			
D	Eat	Bread				
E	Rice	Bread	And	Cake		
F	?	Hamster	Eats	Bread	And	Cake

φ	cute	kit	eat	rice	cake	ham	bre	SUM
#1	**0.001**	0.001	2.001	1.001	3.001	0.001	2.001	8.007
#2	1.001	2.001	1.001	1.001	0.001	2.001	1.001	8.007

Latent Dirichlet Allocation

Topic Modeling

문서 A 에 속하는 어떤 단어 m이 토픽 j에 속할 확률

= 문서 d에 속하는 모든 토픽 중 토픽 j가 차지하는 비중 * 토픽 j에 속하는 모든 단어 중 단어 m이 차지하는 비중

- 5. 새로운 토픽 배정

θ	A	B	C	D	E	F
#1	**0.1**	2.1	0.1	2.1	2.1	2.1
#2	1.1	1.1	2.1	0.1	1.1	3.1

φ	cute	kit	eat	rice	cake	ham	bre	SUM
#1	**0.001**	0.001	2.001	1.001	3.001	0.001	2.001	8.007
#2	1.001	2.001	1.001	1.001	0.001	2.001	1.001	8.007

- 문서 A 에 속하는 단어 cute가 토픽 #1에 속할지 토픽 #2에 속할지 판단
- 문서 A 내에 토픽 #1이 있을 확률 = 0.1 / (0.1+1.1) = 0.083 , 토픽 #1 내의 단어가 cute일 확률 = 0.001 / 8.007 = 0.00012

  문서 A 내에 토픽 #1이 있을 확률 × 토픽 #1 내의 단어가 cute일 확률 = 0.083 * 0.00012 = 0.00008

  문서 A 내에 토픽 #2이 있을 확률 × 토픽 #2 내의 단어가 cute일 확률 = 0.916 * 0.125 = **0.114**

W	cute	kit	eat	rice	cake	kit	ham	eat	bre	rice	bre	cake	cute	ham	eat	bre	cake
Z	#2	#2	#1	#2	#1	#2	#2	#1	#1	#1	#2	#1	#2	#2	#2	#1	#1

Latent Dirichlet Allocation

Topic Modeling

- 6. 문서의 모든 단어에 대해 반복
  - ➢ 모든 단어 한번 업데이트 하면 1회 깁스 샘플링 마무리, 여러 번 반복하면 모든 단어가 적절하게 자기 토픽을 찾아 배정

	cute	kit	eat	rice	cake	kit	ham	eat	bre	rice	bre	cake	cute	ham	eat	bre	cake
Z	#2	#2	#1	#1	#1	#2	#2	#1	#1	#1	#1	#1	#2	#2	#1	#1	#1

  - ➢ 최종 결과: 문서별 토픽분포, 최종 토픽별 단어 분포

θ	A	B	C	D	E	F
#1	0.1	3.1	0.1	2.1	3.1	3.1
#2	2.1	0.1	2.1	0.1	0.1	2.1

φ	cute	kit	eat	rice	cake	ham	bre	SUM
#1	0.001	0.001	3.001	2.001	3.001	0.001	3.001	11.007
#2	2.001	2.001	0.001	0.001	0.001	2.001	0.001	6.007

- 한계
  - ➢ 샘플링을 이용하기 때문에 실행시마다 결과가 달라질 수 있음(문서 집합이 작을수록, sparse 할수록 실행 결과 좌우)
  - ➢ 단어의 분포만을 가지고 토픽을 묶기 대문에 실제 토픽과는 다를 수 있음
  - ➢ 한 문서 내에서 각 토픽 간의 연관성은 찾을 수 없음

Term-weighted LDA

Topic Modeling

- 단어의 상대적인 중요성을 고려한 LDA 토픽 모델링 (깁스 샘플링 변동)
- 단어마다 가중치를 부여하여 확률분포을 계산함
- 중요하지 않은 단어에는 가중치를 덜 부여하여 토픽을 보다 더 잘 찾을 수 있도록 유도
  - 예시: the vs topic

W	cute	kit	eat	rice	cake	kit	ham	eat	bre	rice	bre	cake	cute	ham	eat	bre	cake
Z	#1	#2	#1	#2	#1	#2	#2	#1	#1	#1	#2	#1	#2	#2	#2	#1	#1

φ	cute	kit	eat	rice	cake	ham	bre	SUM
#1	1.001 +가중치	0.001	2.001	1.001	3.001	0.001	2.001	9.007
#2	1.001 +가중치	2.001	1.001	1.001	0.001	2.001	1.001	8.007

Term-weighted LDA

Topic Modeling

- 가중치
  - 이진값 (0 or 1)
  - 정보량(idf): 한 단어가 문서 집합 전체에서 얼마나 공통적으로 나타나는지

$$I = -\log P(x) = \log \frac{N}{df}$$

df: 해당 단어가 출현하는 문서의 수
N: 전체 문서의 수

  - PMI : 문서 별로 단어의 가중치가 다를 수 있음을 고려하여 문서와 단어의 관계성을 가중치로 선정
    - 특정 문서에 특정 단어가 집중적으로 분포할 경우 그 문서와 단어의 PMI 증가
    - 특정 문서와 단어가 상관이 없는 경우: PMI 감소
    - 특정 문서가 특정 단어를 피하는 경우: 음의 PMI -> 보통 가중치를 0으로 변경

$$PMI = \log \frac{P(w|d)}{P(w)} = \log \frac{C_{md} C}{C_d C_m^W}$$

C_{md}: 문서 d의 단어 m의 개수
C_d: 문서 d의 전체 단어의 개수
C: 전체 단어의 개수
C_m : 단어 m의 전체 개수

Term-weighted LDA

Topic Modeling

- 결과 및 성능
  - ➢ Average Precision(정확도): 도출된 토픽과 reference 토픽

	Tokenization	
Weighting Scheme	Word	Morpheme
Unweighted	0.505	0.544
$\log p(w\|L)$	0.616	0.641
PMI	0.612	**0.686**

	Weighting Scheme									
	LDA (no weighting)					PMI-WLDA				
Topic	1	2	3	4	5	1	2	3	4	5
Terms	the	the	vanité	as	cárcel	under	city	coeur	sat	colère
	et	de	vanidad	comme	prison	sous	ville	heart	assis	ira
	and	et	vanity	como	السجن	под	ciudad	corazón	vent	wrath
	los	of	باطل	как	prison	تحت	لمدينة	сердце	wind	anger
	и	and	суета	un	темницу	debajo	город	сердца	viento	furor
	y	y	aflicción	a	prisonniers	ombre	twelve	قلبه	sentado	гнев
	les	de	poursuite	one	темницы	bases	douze	قلب	ветер	fureur
	á	и	الباطل	لها	bound	basas	doce	قلبي	الريح	غضب
	de	la	prédicateur	une	prisión	sombra	دينة	قلبك	sitting	гнева
	of	la	وقبض	واحد	prisoners	dessous	города	сердцем	сел	contre

Table 3: Top 10 terms within top 5 topics for each of LDA and PMI-WLDA. Terms that appear twice within the same topic (e.g. 'la' in LDA topic 2) are words from different languages with the same spelling (here Spanish and French).

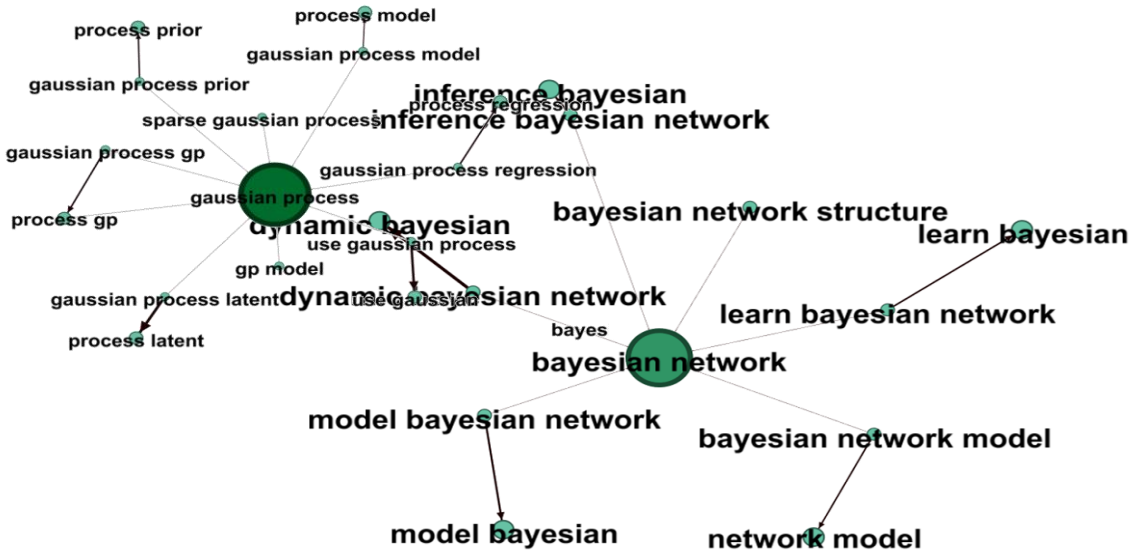
Project

Topic Modeling

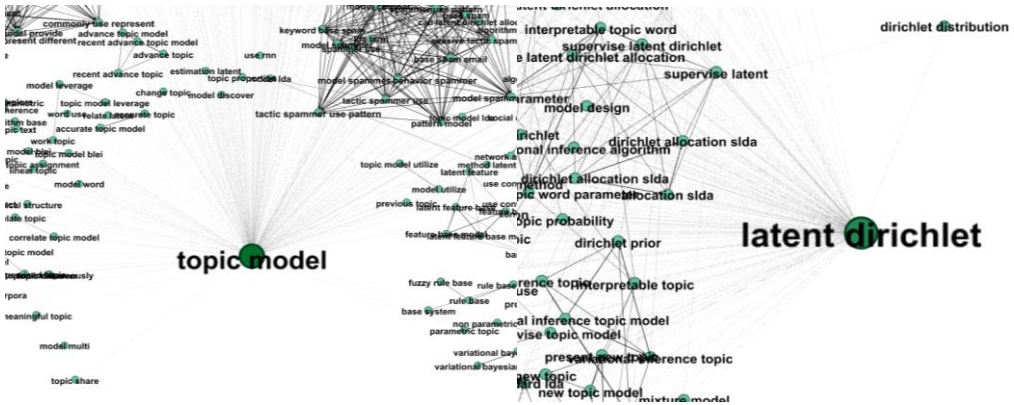
- 목표
  - ➢ 여러 문서 집합의 토픽 도출 후 문서 내 key word들의 network를 그려보자
  - ➢ 토픽 도출 시 LDA 사용

Data: arxiv 논문 데이터

Topic 1



Topic 2



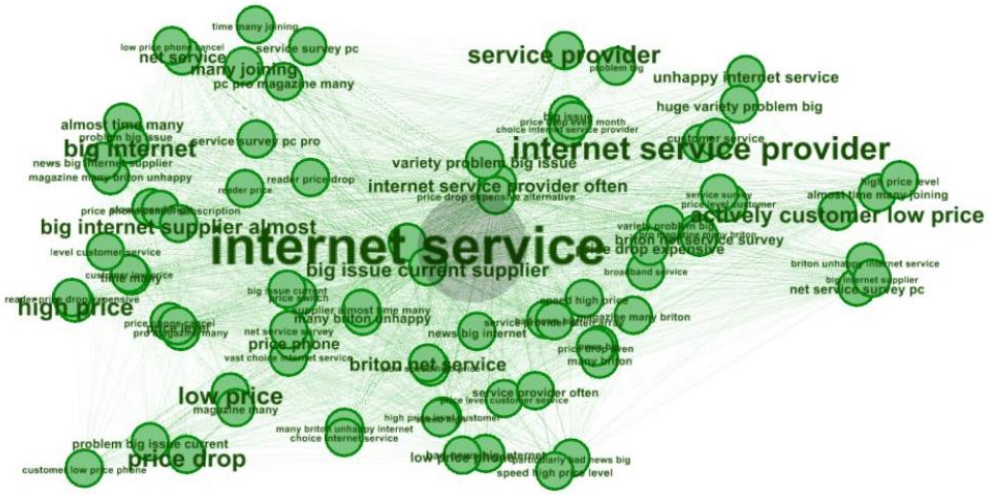
Project

Topic Modeling

- 목표
  - ➢ 여러 문서 집합의 토픽 도출 후 문서 내 key word들의 network를 그려보자
  - ➢ 토픽 도출 시 LDA 사용

Data: BBC News 기사 데이터

Topic 1



Topic 2

결론

Topic Modeling

- 대량의 비정형 데이터를 분석하기 위해 토픽 모델링을 위한 연구가 진행
- 토픽 모델링이란 문서 내 단어들의 분포를 활용하여 토픽을 도출하는 것
- 토픽 모델링의 기반이 되는 차원 축소 기법, 확률 분포를 이용한 기법, 변형된 확률 모델 존재
- 텍스트 요약, 번역, 의미론 분석에서 널리 활용되고 있음
- 토픽의 labeling에 대한 한계는 아직까지 연구가 미흡하며 딥러닝을 활용한 토픽 모델링에 대한 기법으로 해결 시도
  - LDA2vec