15기 추천시스템 세미나

Tobig's 15기 강의자
장아연 & 이성범

Airbnb Paper Review

Real – time Personalization using Embedding for Search Ranking at Airbnb

Contents

Unit 01 | INTRODUCTION

Unit 02 | RELATED WORK

Unit 03 | METHODOLOGY

Unit 04 | EXPERIMENTS

Unit 05 | CONCLUSION

01 INTRODUCTION

Unit 01 | INTRODUCTION

Unique Search and Recommendation

Host와 Guest의 선호를 만족/최적화해야 하는
two-sided marketplace

같은 item을 두 번 이상 구매하는 경우가 적음

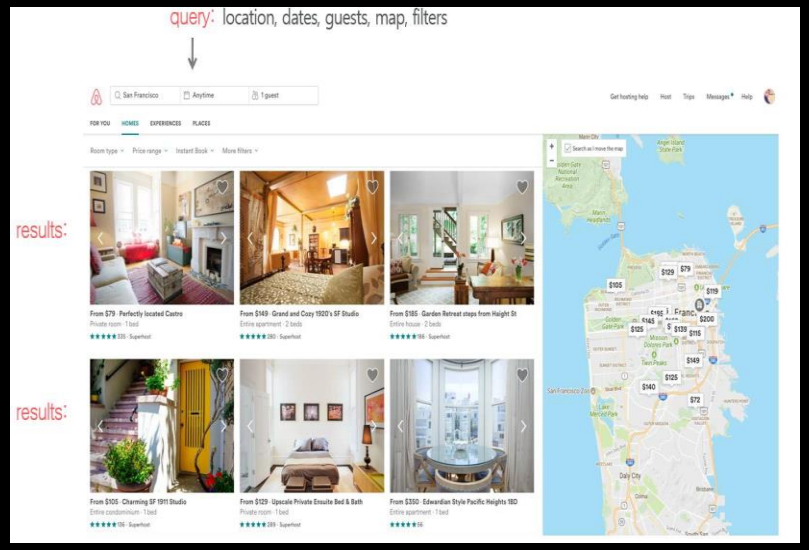
특정 기간 동안 한 명의 Guest만 받을 수 있음

Unit 01 | INTRODUCTION

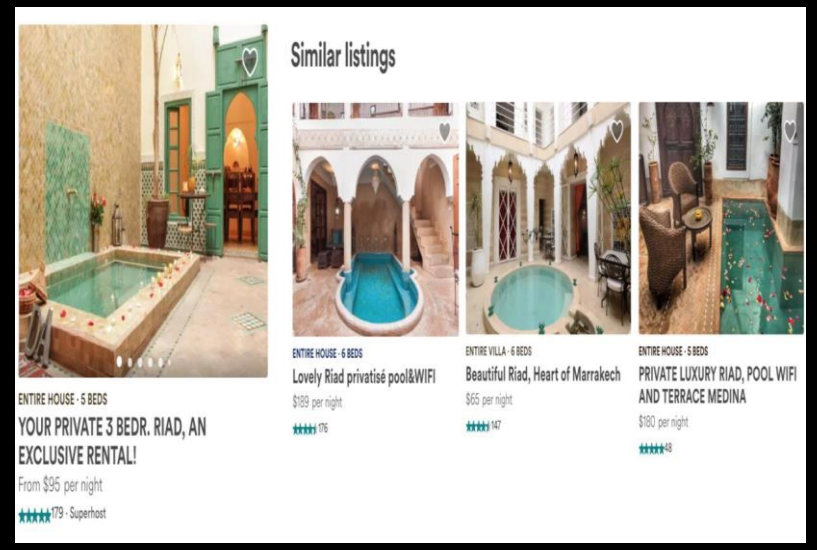
Goal

Host와 Guest 양측을 만족시키는 Search Ranking과 Recommendation 서비스 제공

Real-time Personalization in Search Ranking



Similar Listing Recommendations



Unit 01 | INTRODUCTION

Real-time Personalization

Adapting Training for Congregated Search

User Type Embeddings

Listing 과 User Embedding을 이용한 추천 시스템 구현

Leveraging Conversions as Global Context

Rejections as Explicit Negatives

Unit 01 | INTRODUCTION

1. Real-time Personalization

User의 최근 상호 작용한 항목 & 순위를 매겨야 하는 후보 항목들
Online 방식으로 결합해 유사성 계산

2. Adapting Training for Congregated Search

User가 특정한 시장/범위 내에서만 검색하는 Congregated Search함
negative sampling 진행

3. Leveraging Conversions as Global Context

Session에 따른 window가 움직일 때마다
Booked list를 Global(전역)으로 취급

4. User Type Embeddings

User 개개인의 User ID 아님
User Type으로 Embedding 진행
Same Type of User = Same value of Embedding

5. Rejections as Explicit Negatives

Host의 거절로 추천 항목에 대한 Booking이 좌절된 경우 방지
Explicit Negative로 Host의 preference signal 중 Rejection을 반영

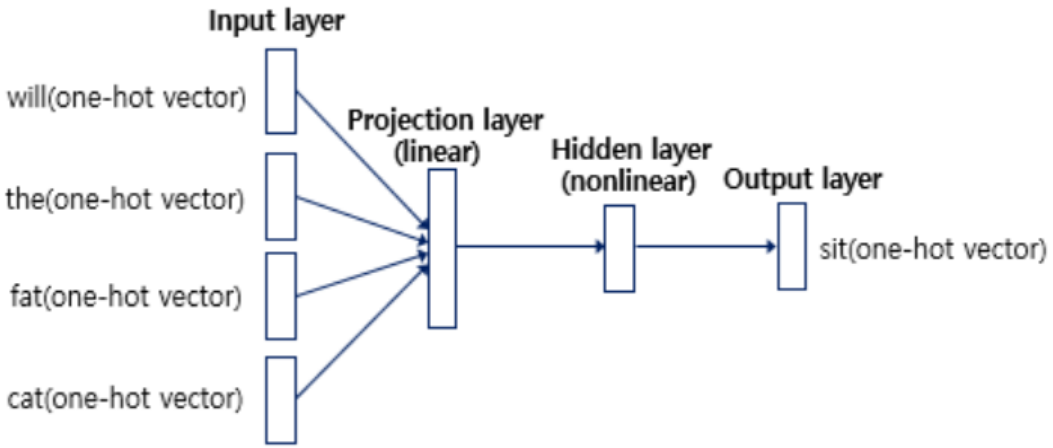
02 RELATED WORK

Neural Language Model

Neural Network 사용
Word Embedding 학습
Low Dimension Representation 표현

Efficient Estimation of Word Representations in Vector Space
Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean

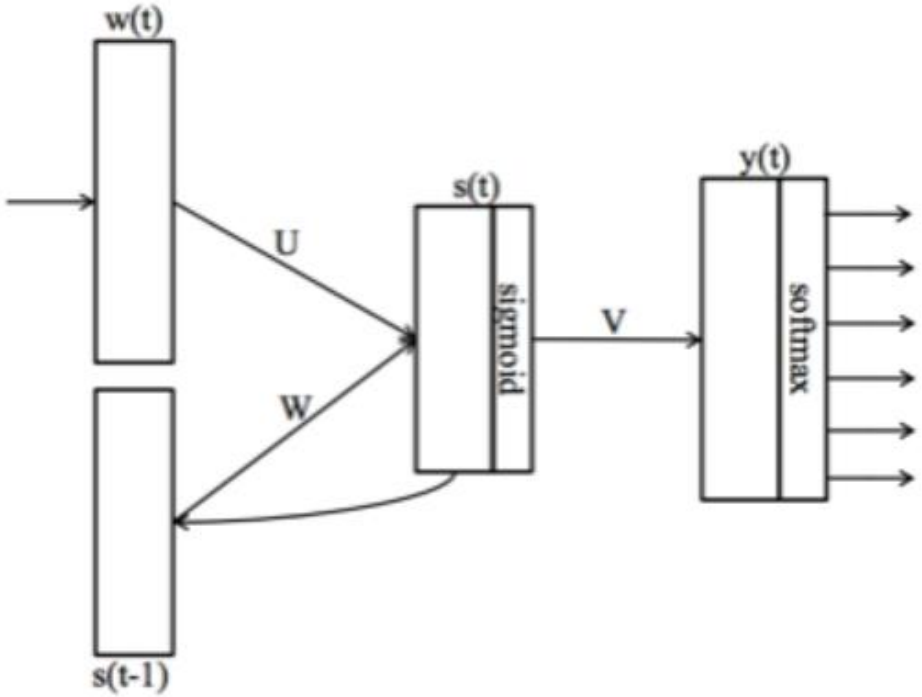
NNLM : Feed – Forward Neural Net Language Model



한계

1. 단어 수 지정 필수
2. 중심 단어 바로 앞 단어만 고려
3. 높은 계산 복잡도
4. 느림

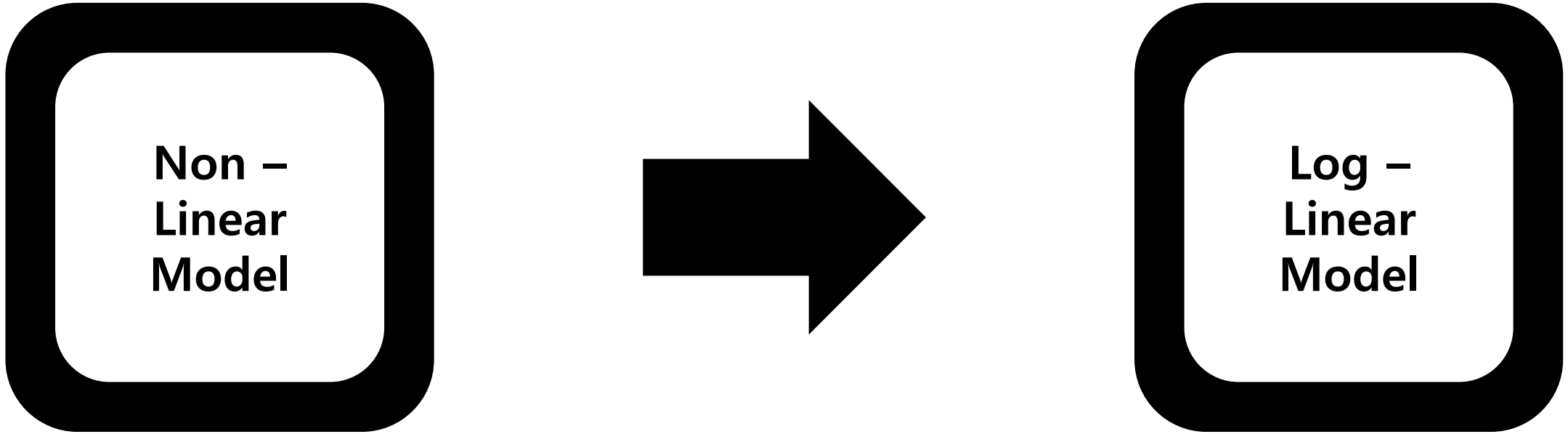
RNNLM : Recurrent Neural Net Language Model



NNLM 한계 극복

1. 단어 수 지정 필수 X
2. Projection Layer 없음
   -> Recurrent 연결

Neural Network 기반의 단어 벡터화의 계산 복잡도 줄이기



Unit 02 | RELATED WORK

You shall know a word by the company it keeps

- 언어 학자 J.R.Firth (1957)

Word Order
Word Co-Occurrence

이용한 training 진행

Word2vec

CBoW : Continuous Bag of Words

SG : skip gram language

Word Representation의 NLP 영역을 넘어 다양한 분야로 확장 적용됨

무엇을 맥락으로 적용하냐에 따라 확장 가능해짐

Sentence 속 Word Sequence

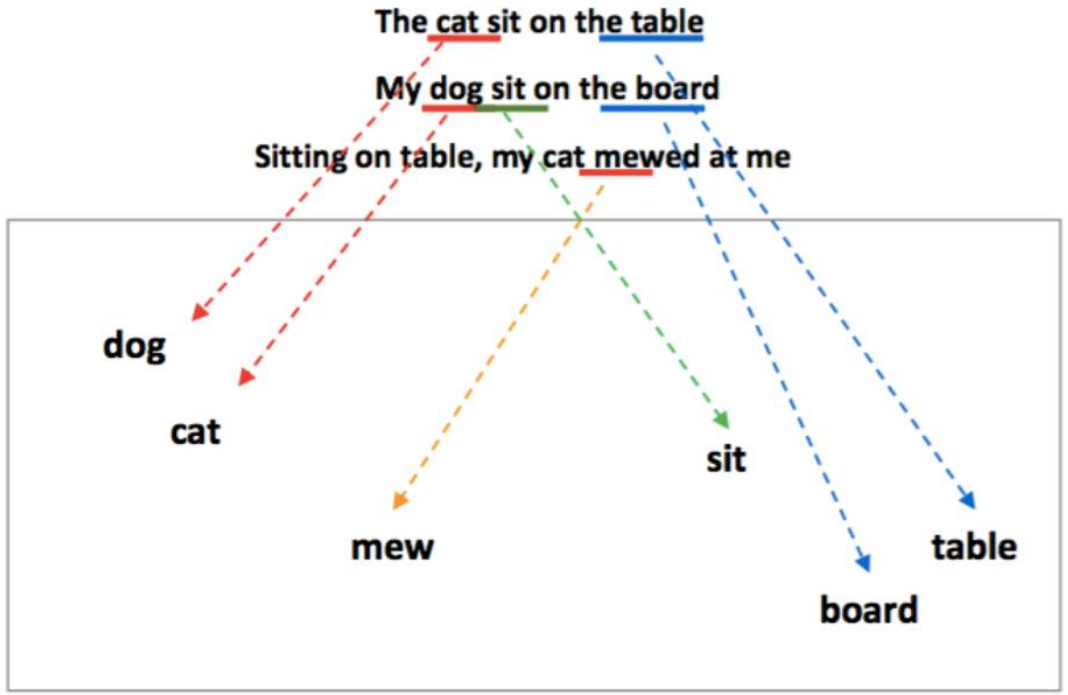
Word Embedding

User action Sequence

User action Embedding

Word2Vec : Softmax regression

: 고차원의 의미공간에서 각 단어의 좌표값 부여



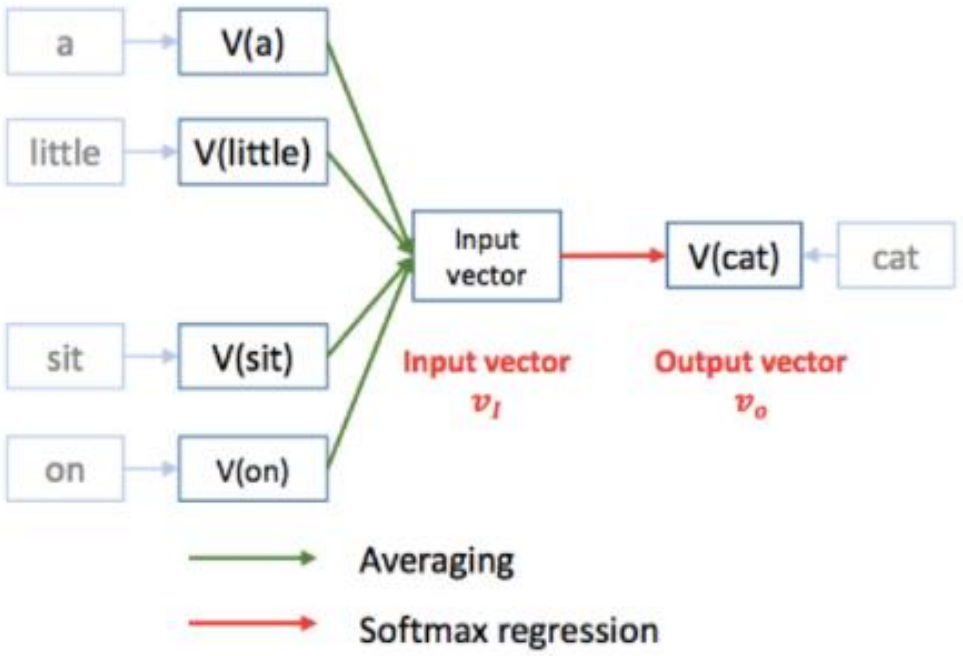
$$maximize P(y_k|x) = \frac{exp(\beta_{y_k}^T x)}{\sum_j exp(\beta_j^T x)}$$

X에 대해 class y가 될 확률 최대화 시키는 방향으로
클래스 y의 대표 벡터인 coefficient β에 학습 진행

최대가 되는 경우
: X와 이에 해당하는 β가 같은 방향

Word2Vec : Softmax regression

Window Classification



문장 : a little cat sit on the table

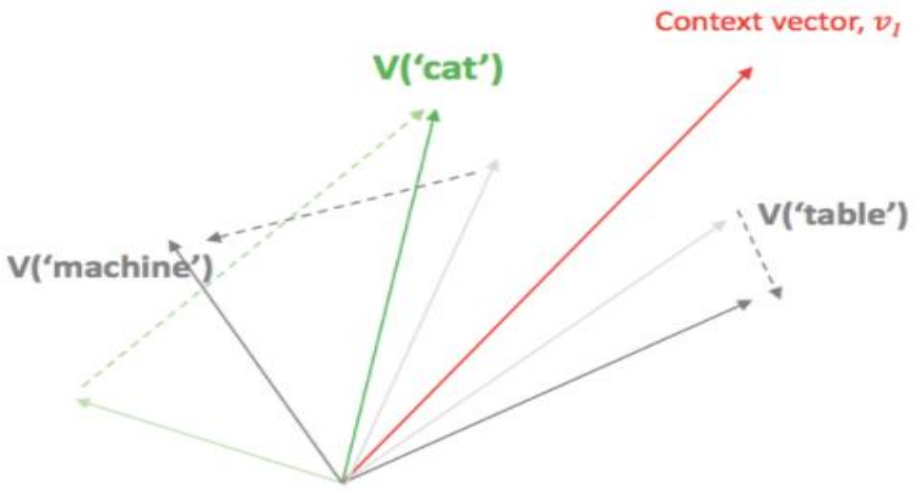
X : [a, little, sit, on], Y : cat
스캐너가 [a, little, sit, on] 스캔 Cat 예측하는 regression 진행

X : [little, cat, on, the], Y : sit
스캐너가 [little, cat, on, the] 스캔 Sit 예측하는 regression 진행

Word2Vec : Softmax regression

: 의미공간에서 각 단어의 위치 좌표를 수정하며 Word2vec의 학습 진행

$$h_\theta(x) = \begin{bmatrix} P(w_{(t)} = cat) \\ P(w_{(t)} = dog) \\ P(w_{(t)} = table) \\ \dots \\ P(w_{(t)} = Vocab) \end{bmatrix} = \frac{1}{\sum_{j=1}^{|V|} \exp(\theta^{(j)T} x)} \begin{bmatrix} \exp(v(cat)^T v_l) \\ \exp(v(dog)^T v_l) \\ \exp(v(table)^T v_l) \\ \dots \\ \exp(v(Vocab)^T v_l) \end{bmatrix}$$

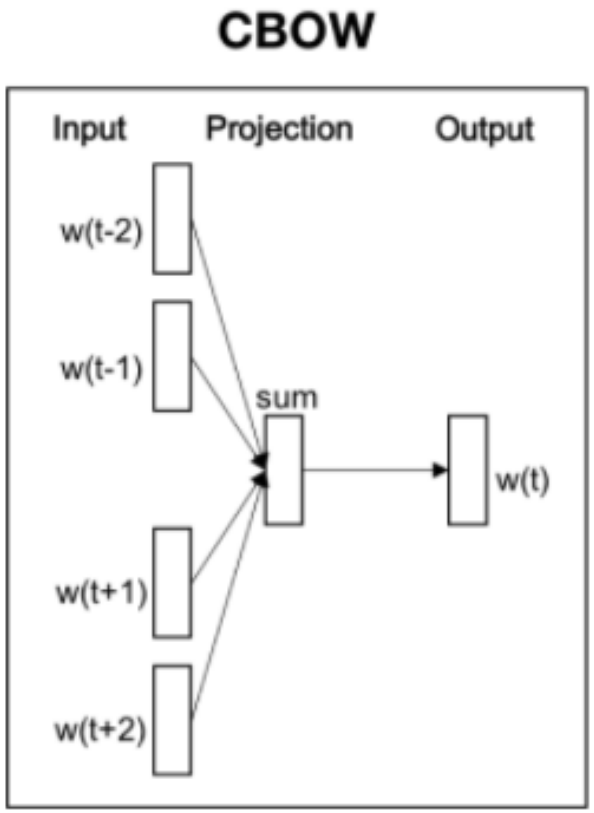


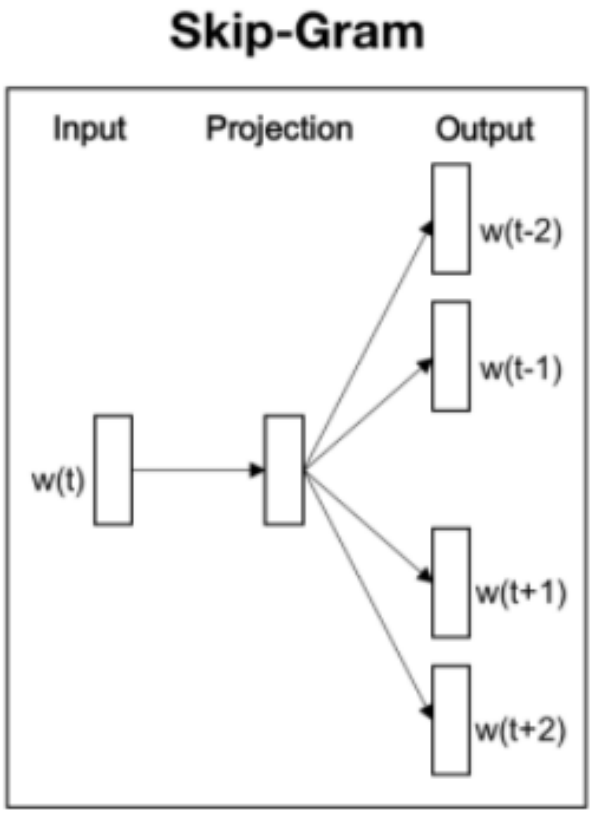
문장 : a little cat sit on the table

$$maximize P(y_k|x) = \frac{exp(\beta_{y_k}^T x)}{\sum_j exp(\beta_j^T x)}$$

V(cat)과 V(sit)이 서로 멀어지는 방향으로 Vector를 조정하며 학습 진행

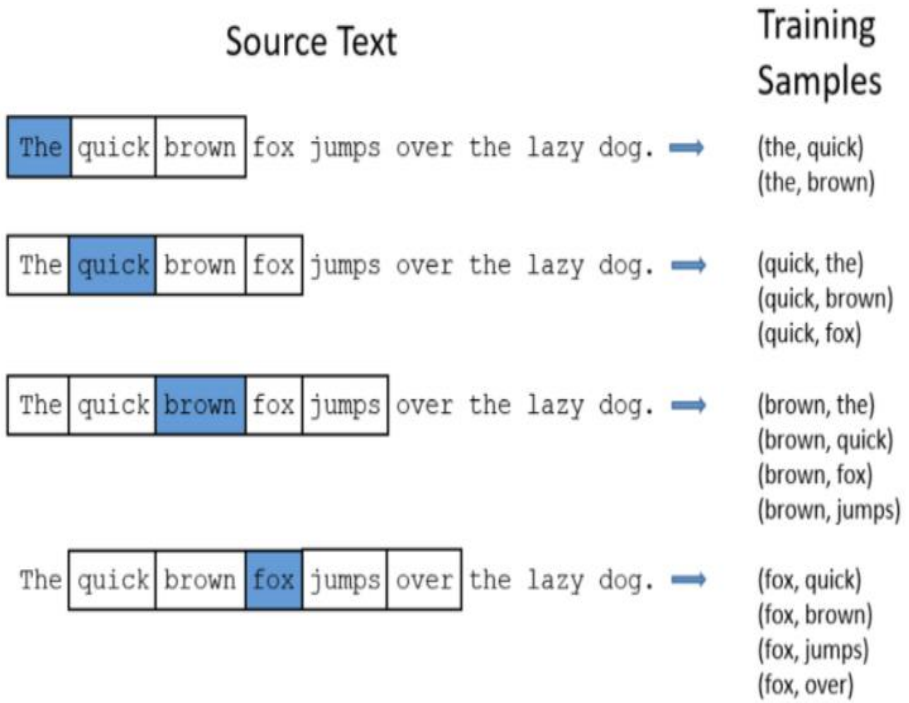
Unit 02 | RELATED WORK



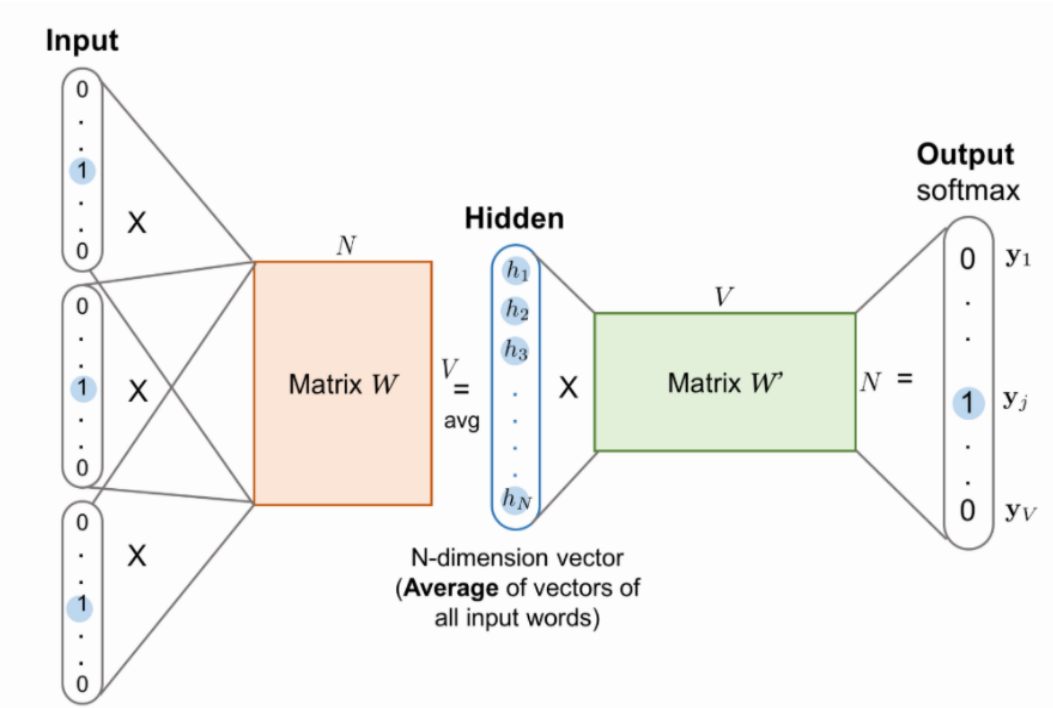


Unit 02 | RELATED WORK

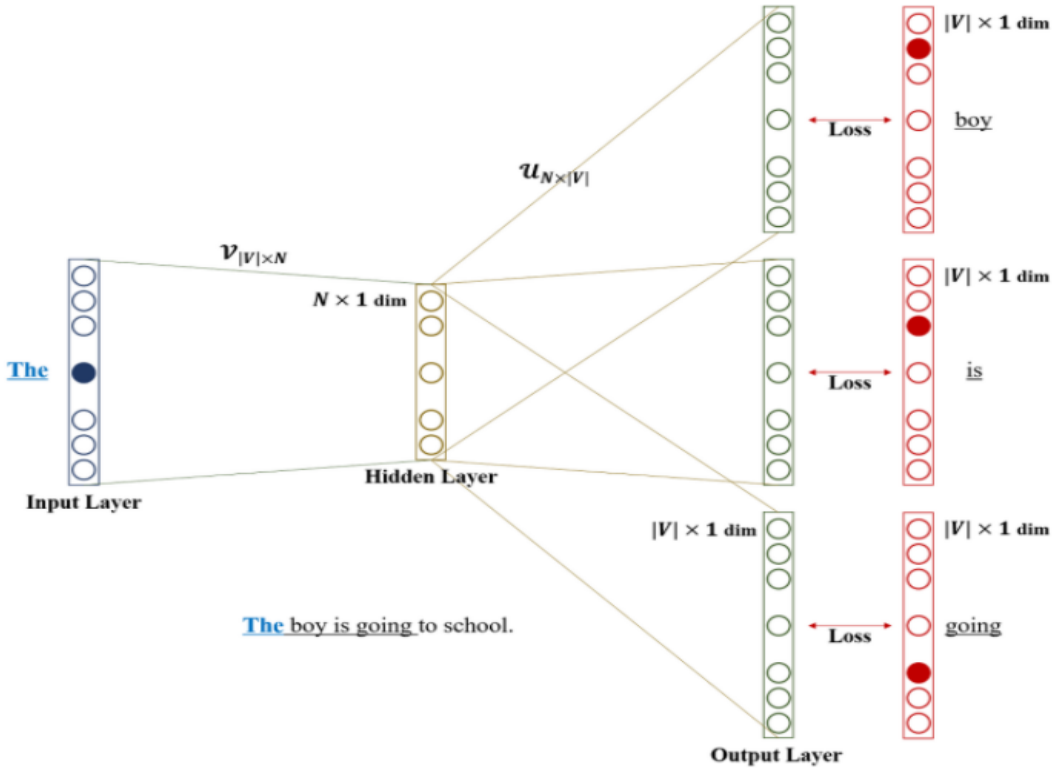
center word	context words
[1,0,0,0,0,0,0]	[0,1,0,0,0,0,0] [0,0,1,0,0,0,0]
[0,1,0,0,0,0,0]	[1,0,0,0,0,0,0] [0,0,1,0,0,0,0] [0,0,0,1,0,0,0]
[0,0,1,0,0,0,0]	[1,0,0,0,0,0,0] [0,1,0,0,0,0,0] [0,0,0,1,0,0,0] [0,0,0,0,1,0,0]
[0,0,0,1,0,0,0]	[0,1,0,0,0,0,0] [0,0,1,0,0,0,0] [0,0,0,0,1,0,0] [0,0,0,0,0,1,0]
[0,0,0,0,1,0,0]	[0,1,0,0,0,0,0] [0,0,0,1,0,0,0] [0,0,0,0,0,1,0] [0,0,0,0,0,0,1]
[0,0,0,0,0,1,0]	[1,0,0,1,0,0,0] [0,0,0,0,1,0,0] [0,0,0,0,0,0,1]
[0,0,0,0,0,0,1]	[0,0,0,0,1,0,0] [0,0,0,0,0,1,0]



1. CBoW : Continuous Bag of Word



2. SG : Skip Gram



Skip – Gram : Vector Representation Quality 높이기

1. Phase 기반 학습
2. 빈출 단어 기반으로 subsampling
3. Train 과정에서 negative sampling 기법 도입

Distributed Representations of Words and Phrases and their Compositionality
Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean

Unit 02 | RELATED WORK

1. Phase 기반 학습

$$\text{score}(w_i, w_j) = \frac{\text{count}(w_i w_j) - \delta}{\text{count}(w_i) \times \text{count}(w_j)}$$

Boston	Globe	Boston Globe

하나의 단어가 여러 단어로 이루어져 있지만
각 각 단어에 대한 뜻의 합이 그 단어의 의미가 아닌 경우

여러 단어가 묶여 함께 자주 등장하지만
각 각은 잘 나타나지 않는 단어를 Score를 계산해 찾음

2. 빈출단어 기반으로 subsampling 진행

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}}$$

In	of	At	their	that	with

자주 등장하지만 이보다 빈도수가 적은 단어보다 포함하는 정보량이 적음

모집단을 그룹으로 나누고
각 그룹에서 표본 추출하여 해당 표본만 이용

3. Train 과정에서 negative sampling 기법 도입

$$p(w_O|w_I) = \frac{\exp\left({v'_{w_O}}^{\top} v_{w_I}\right)}{\sum_{w=1}^{W} \exp\left({v'_w}^{\top} v_{w_I}\right)}$$

Log probability Value 향상 vs Vector Representation Quality 향상

Hierachical Softmax	NCE : Noise Contrastive Estimation	NEG : Negative Sampling

$$p(w|w_I) = \prod_{j=1}^{L(w)-1} \sigma\left(\left[\left[n(w,j+1) = \mathrm{ch}(n(w,j))\right]\right] \cdot {v'_{n(w,j)}}^{\top} v_{w_I}\right)$$

$$\log \sigma({v'_{w_O}}^{\top} v_{w_I}) + \sum_{i=1}^{k} \mathbb{E}_{w_i \sim P_n(w)} \left[\log \sigma(-{v'_{w_i}}^{\top} v_{w_I})\right]$$

$$P_n(w) = U(w)^{\frac{3}{4}}/z$$

03 METHODOLOGY

Airbnb의 개인화를 위한 Embedding 종류

- Listing Embeddings -> 고객의 단기 관심사를 개인화
- User-type & Listing-type Embeddings -> 고객의 장기 관심사를 개인화

Airbnb를 사용하는 고객의 분류

- 어떤 목록을 클릭 후, 검색 결과로 다시 돌아가는 사람
- 현재 목록과 관련된 추천 정보를 클릭하는 사람

클릭이 모인 세션들을 활용하여 Embedding을 하자!

Listing Embeddings

- Data

$$s = (l_1, \ldots, l_M) \in \mathcal{S}$$

- N명의 유저로부터 수집된 Click sessions인 s
- M개의 Listing id로 구성된 Click sessions인 s
- 30분을 간격으로 수집된 s의 집합인 S

Airbnb는 8억개 이상의 검색 클릭 세션을 사용

Listing Embeddings

- Model

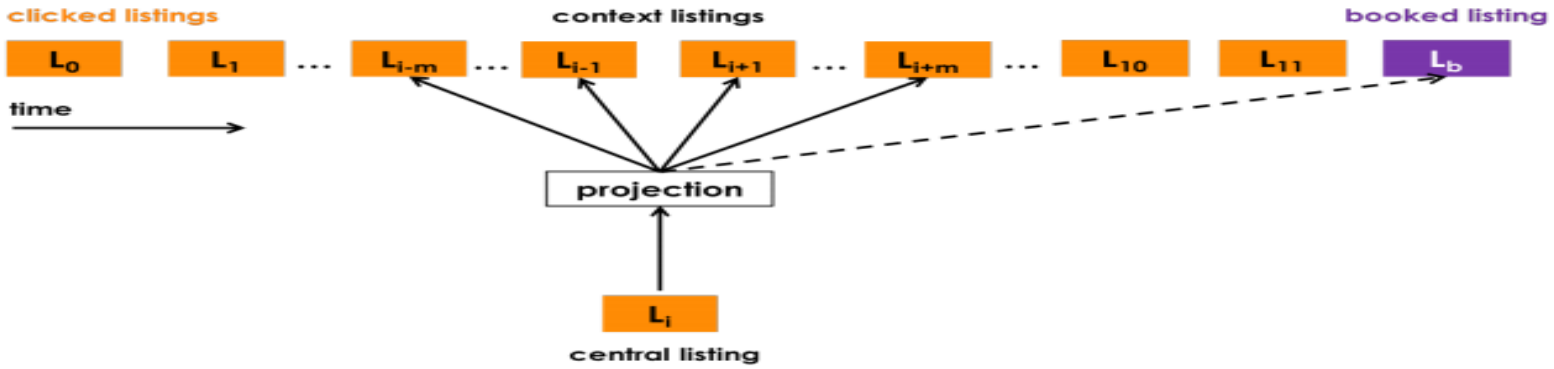


Figure 1: Skip-gram model for Listing Embeddings

- Negative Sampling을 사용한 Skip-gram model을 활용
- D-차원의 Listing Representation을 나타내는 Embedding Space를 구하자

Airbnb의 경우 컴퓨터 비용과 성과 사이에 Trade-off를 통해서 32차원으로 결정

Listing Embeddings

- Model
  - Negative Sampling 이란?
    - 각 단어의 빈도수를 고려하여 자주 등장하는 단어를 높을 확률로 샘플링 하는 방법
    - 자주 등장하는 단어만큼은 제대로 학습하자!

pos와 유사한 단어가 neg로 샘플링 되더라도 neg로 샘플링 된 단어의 빈도는 워낙 많아

Context Vector로 부터 아주 조금만 멀어져 학습에는 큰 지장이 없다!

왜냐하면 Softmax regression 입장에서 조금 떨어진 단어는 영향력이 작기 때문에!

Listing Embeddings

- Model
  - Embedding을 Random Vector로 초기화
  - Session내의 Listing들을 Window 단위로 학습
  - 업데이트 방식은 SGD를 활용
  - Central Listing의 근처에 있는 pos와는 가깝게
  - 동시에 Window내에 없는 neg와는 멀게 학습

Airbnb는 위 방법을 약간 수정하여 적용!

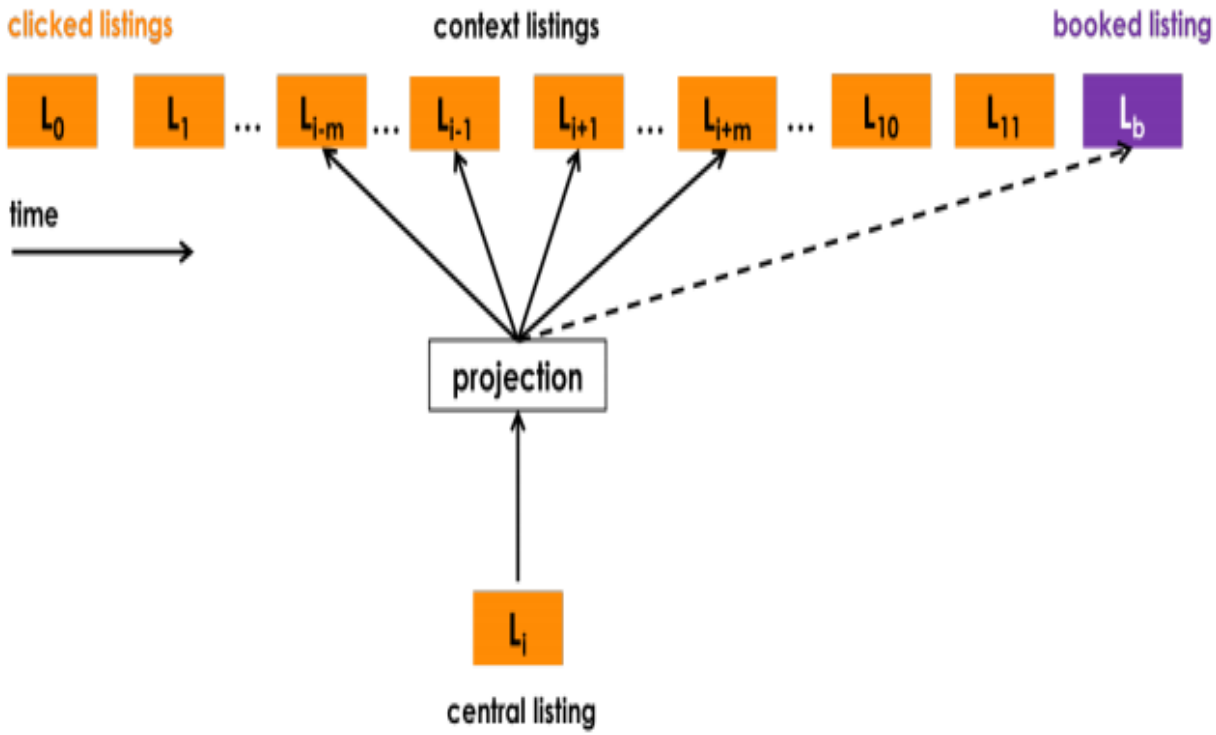


Figure 1: Skip-gram model for Listing Embeddings

Unit 03 | METHODOLOGY

Listing Embeddings

- Model
  - **Using Booked Listing as Global Context**
    - 예약으로 종료되는 Session 만을 이용하여 학습
    - Booked listing을 Window 단위로 모델이 학습하는 동안 Central Listing의 업데이트에 반영

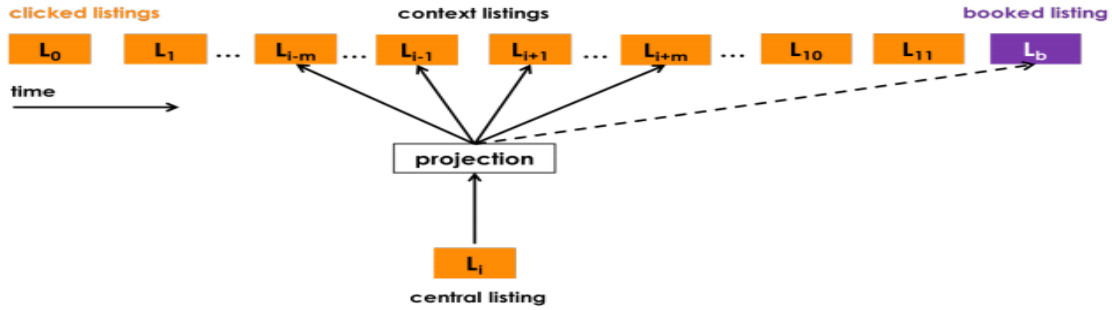


Figure 1: Skip-gram model for Listing Embeddings

최종 예약 목록까지 잘 예측할 수 있도록 최적화가 가능!

  - **Adapting to Congregated Search**
    - pos의 경우 대게 같은 지역내에 존재하는 listing, neg의 경우 다른 지역에 존재하는 listing
    - Data의 Sampling 자체가 매우 Imbalance함
    - Central listing의 지역에서 Random 하게 추출한 neg Dmm set을 추가하여 학습

지역내 유사성을 조금 더 잘 반영할 수 있게 학습!

Listing Embeddings

- Optimization Objective

$$\operatorname*{argmax}_{\theta} \sum_{(l,c)\in\mathcal{D}_p} \log \frac{1}{1+e^{-\mathbf{v}'_c \mathbf{v}_l}} + \sum_{(l,c)\in\mathcal{D}_n} \log \frac{1}{1+e^{\mathbf{v}'_c \mathbf{v}_l}} + \log \frac{1}{1+e^{-\mathbf{v}'_{l_b} \mathbf{v}_l}} + \sum_{(l,m_n)\in\mathcal{D}_{m_n}} \log \frac{1}{1+e^{\mathbf{v}'_{m_n} \mathbf{v}_l}}.$$

- l은 central listing을 의미, vector v(l)은 업데이트 됨
- Dp는 central listing과 pos context listing의 pair (l, c)의 set, 서로 가깝게 Embedding 되도록 학습
- Dn는 central listing과 neg context listing의 pair (l, c)의 set, 서로 멀게 Embedding 되도록 학습
- lb는 Global context로 central listing vector와 서로 가깝게 Embedding 되도록 학습
- Dmn는 central listing과 동일 지역내의 neg context listing의 pair (l, mn)의 set, 서로 멀게 Embedding 되도록 학습

Listing Embeddings

- Evaluate
  - K-means clustering <- 지리적 유사성이 잘 학습되었는지 판단
  - Cosine similarity <- Embedding에 listing의 특징이 잘 반영되었는지 판단
  - Tool <- 빠르게 Embedding의 결과를 확인하기 위해

Unit 03 | METHODOLOGY

Listing Embeddings

- K-means clustering
  - California에 있는 listing들을 100개로 군집화
  - listing간의 지리적 유사성을 판단

그림과 같이 지리적 유사성이 잘 학습됨

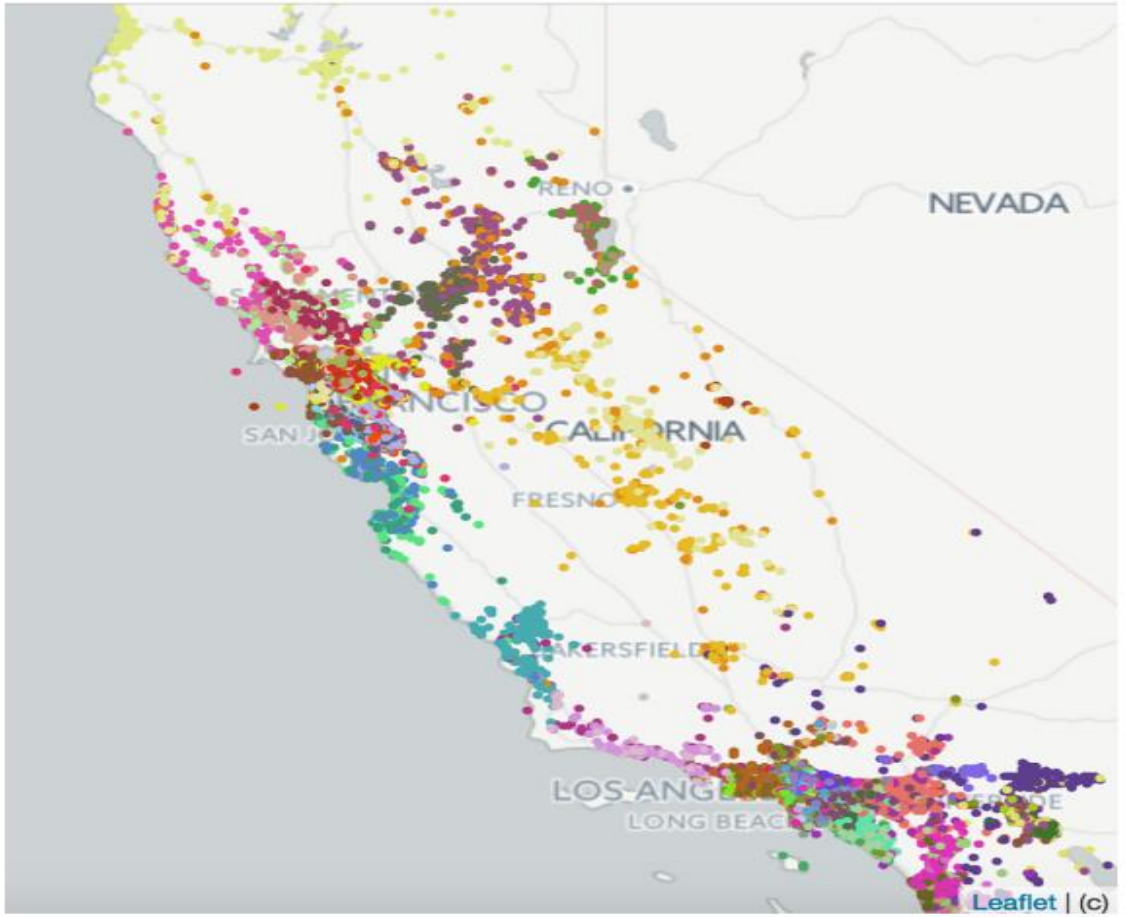


Figure 2: California Listing Embedding Clusters

Listing Embeddings

- Cosine similarity
  - listing 간의 유사성을 판단

그림과 같이 같은 type, 동일한 가격 범위 내의 listing 들이 서로 유사한 것을 확인 가능

따라서 listing의 특징이 Embedding에 잘 학습됨

Table 1: Cosine similarities between different Listing Types

Room Type	Entire Home	Private Room	Shared Room
Entire Home	**0.895**	0.875	0.848
Private Room		**0.901**	0.865
Shared Room			**0.896**

Table 2: Cosine similarities between different Price Ranges

Price Range	<$30	$30-$60	$60-$90	$90-$120	$120+
<$30	**0.916**	0.887	0.882	0.871	0.854
$30-$60		**0.906**	0.889	0.876	0.865
$60-$90			**0.902**	0.883	0.880
$90-$120				**0.898**	0.890
$120+					**0.909**

Listing Embeddings

- Tool

그림과 같은 Tool을 만들어

listing 간의 특징이 잘 학습 되었는지

빠르게 판단함

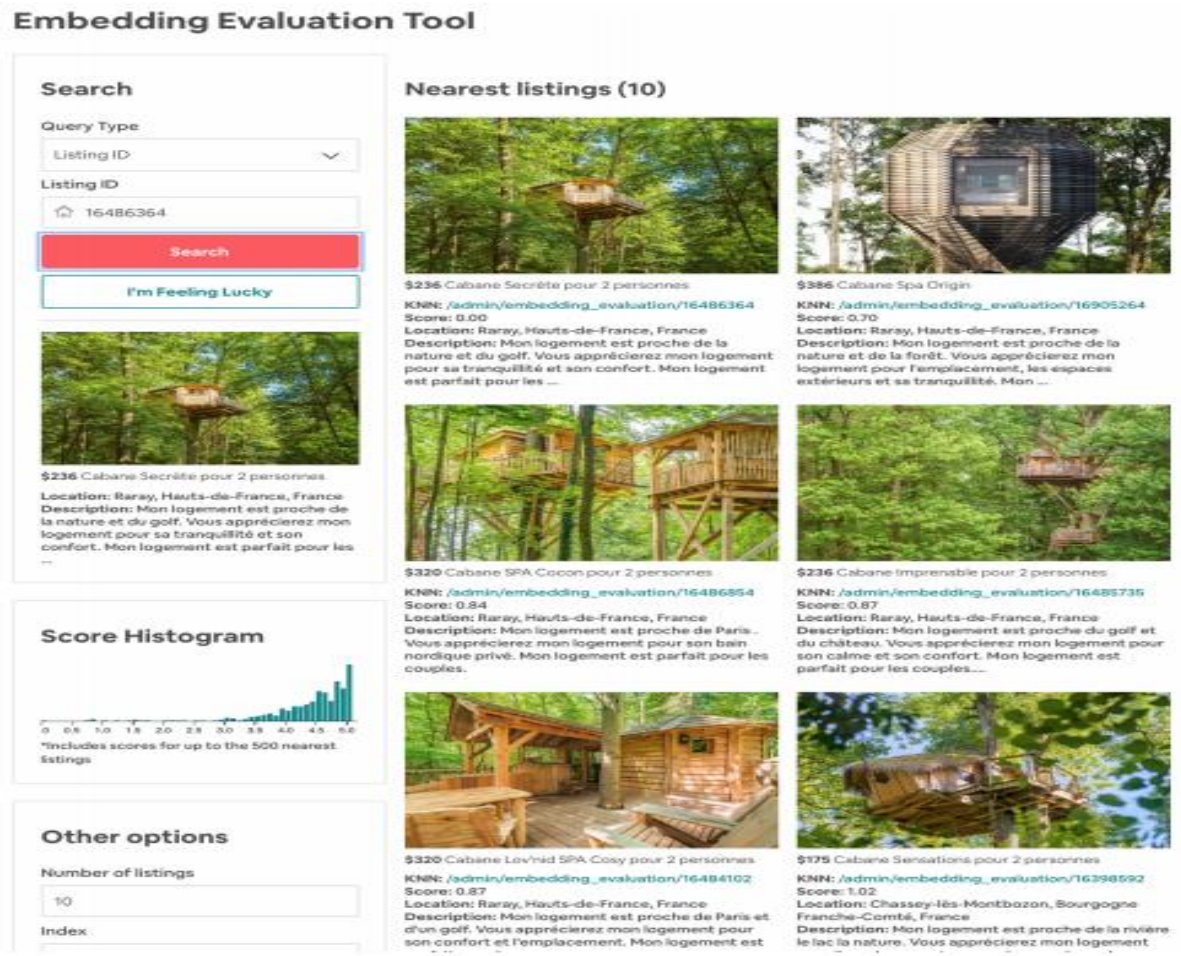


Figure 4: Embeddings Evaluation Tool

Listing Embeddings

- Airbnb는 본 모델을 통하여 450만 개의 listing을 Embedding 함

- Airbnb의 Cold start listing embedding 해결법
  - 새롭게 등록된 listing과
  - 지역적으로 가장 가깝고,
  - 동일한 type이며,
  - 동일한 가격 범위를 가지는
  - 3개의 listing의 Vector를 구해서 평균을 내어 활용함

User-type & Listing-type Embeddings

- Data

먼저 Listing Embedding과

비슷한 방식으로 데이터를 구성해보자!

User-type & Listing-type Embeddings

- Data

$$s_b = (l_{b1}, \ldots, l_{bM})$$

N명의 유저로부터 수집된 M개의 예약 Listing id으로 sessions을 구성하자!

그런데 이렇게 데이터를 구성하니 문제점이 존재!

User-type & Listing-type Embeddings

- Data
  - 예약이라는 이벤트는 매우 드물기 때문에 Sb보다 훨씬 작다!
  - 많은 사용자가 과거에 하나만 예약했기 때문에 세션의 길이가 1인 경우가 대부분!
  - 사용자의 연속적인 예약 사이에 긴 시간이 존재하고 그 시간 동안 사용자의 선호가 변화가능!

그래서 Airbnb는 데이터를 다르게 구성하여 학습!

Unit 03 | METHODOLOGY

User-type & Listing-type Embeddings

- Data

Table 3: Mappings of listing meta data to listing type buckets

Buckets	1	2	3	4	5	6	7	8
Country	US	CA	GB	FR	MX	AU	ES	...
Listing Type	Ent	Priv	Share					
$ per Night	<40	40-55	56-69	70-83	84-100	101-129	130-189	190+
$ per Guest	<21	21-27	28-34	35-42	43-52	53-75	76+	
Num Reviews	0	1	2-5	6-10	11-35	35+		
Listing 5 Star %	0-40	41-60	61-90	90+				
Capacity	1	2	3	4	5	6+		
Num Beds	1	2	3	4+				
Num Bedrooms	0	1	2	3	4+			
Num Bathroom	0	1	2	3+				
New Guest Acc %	<60	61-90	>91					

Table 4: Mappings of user meta data to user type buckets

Buckets	1	2	3	4	5	6	7	8
Market	SF	NYC	LA	HK	PHL	AUS	LV	...
Language	en	es	fr	jp	ru	ko	de	...
Device Type	Mac	Msft	Andr	Ipad	Tablet	Iphone	...	
Full Profile	Yes	No						
Profile Photo	Yes	No						
Num Bookings	0	1	2-7	8+				
$ per Night	<40	40-55	56-69	70-83	84-100	101-129	130-189	190+
$ per Guest	<21	21-27	28-34	35-42	43-52	53-75	76+	
Capacity	<2	2-2.6	2.7-3	3.1-4	4.1-6	6.1+		
Num Reviews	<1	1-3.5	3.6-10	> 10				
Listing 5 Star %	0-40	41-60	61-90	90+				
Guest 5 Star %	0-40	41-60	61-90	90+				

각각의 id가 아닌 id를 그림과 같은 type에 mapping하여 학습하자!

User-type & Listing-type Embeddings

- Data

$$s_b = (u_{type_1} l_{type_1}, \ldots, u_{type_M} l_{type_M}) \in S_b$$

  - N명의 User_type에게 수집된 M개의 예약 Listing type을 튜플로 구성한 sessions인 sb
  - 서로 다른 sb의 집합인 Sb

시간이 지남에 따라 해당 listing type을 선호하는 user가 달라지면 listing type 변경

User-type & Listing-type Embeddings

- Model

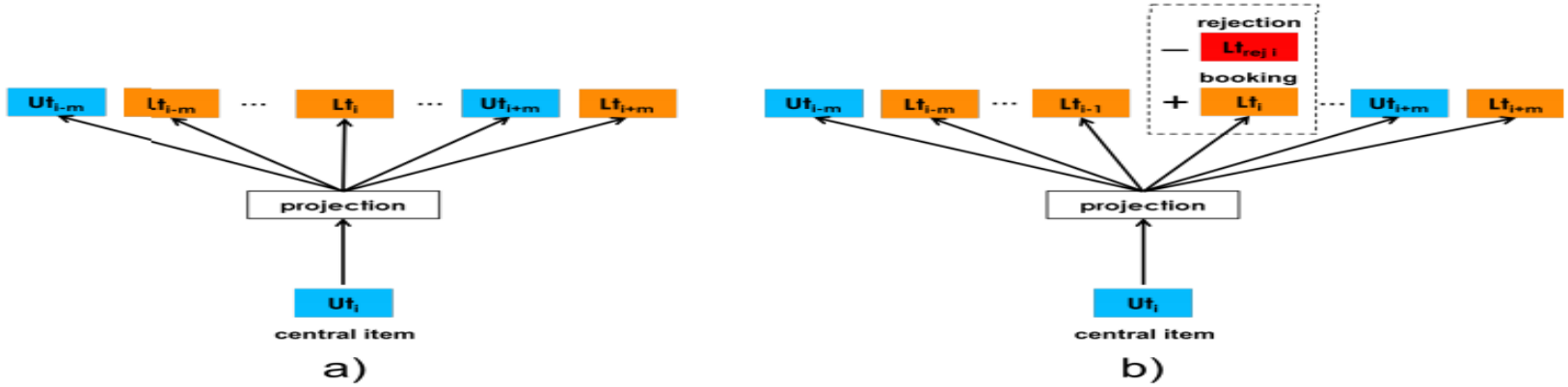


Figure 5: Listing Type and User Type Skip-gram model

- Listing Embeddings 처럼 같은 지역 내의 neg를 추가하여 학습할 필요 X
- 대신 호스트에 대한 선호를 반영할 필요 O
- 학습 시 rejection이 생기면 –, 예약이 잘되면 +

User-type & Listing-type Embeddings

- Optimization Objective

$$\underset{\theta}{\operatorname{argmax}} \sum_{(u_t,c)\in\mathcal{D}_{book}} \log\frac{1}{1+\exp^{-\mathbf{v}'_c\mathbf{v}_{u_t}}} + \sum_{(u_t,c)\in\mathcal{D}_{neg}} \log\frac{1}{1+\exp^{\mathbf{v}'_c\mathbf{v}_{u_t}}} + \sum_{(u_t,l_t)\in\mathcal{D}_{reject}} \log\frac{1}{1+\exp^{\mathbf{v}'_{l_t}\mathbf{v}_{u_t}}}. \quad (8)$$

User-type Embeddings

$$\underset{\theta}{\operatorname{argmax}} \sum_{(l_t,c)\in\mathcal{D}_{book}} \log\frac{1}{1+\exp^{-\mathbf{v}'_c\mathbf{v}_{l_t}}} + \sum_{(l_t,c)\in\mathcal{D}_{neg}} \log\frac{1}{1+\exp^{\mathbf{v}'_c\mathbf{v}_{l_t}}} + \sum_{(l_t,u_t)\in\mathcal{D}_{reject}} \log\frac{1}{1+\exp^{\mathbf{v}'_{u_t}\mathbf{v}_{l_t}}}. \quad (9)$$

Listing-type Embeddings

User-type & Listing-type Embeddings

- Evaluate

Table 5: Recommendations based on type embeddings

User Type	
$SF\_lg_1\_dt_1\_fp_1\_pp_1\_nb_3\_ppn_5\_ppg_5\_c_4\_nr_3\_l5s_3\_g5s_3$	
Listing Type	**Sim**
$US\_lt_1\_pn_4\_pg_5\_r_5\_5s_4\_c_2\_b_1\_bd_3\_bt_3\_nu_3$ (large, good reviews)	0.629
$US\_lt_1\_pn_3\_pg_3\_r_5\_5s_2\_c_3\_b_1\_bd_2\_bt_2\_nu_3$ (cheaper, bad reviews)	0.350
$US\_lt_2\_pn_3\_pg_3\_r_5\_5s_4\_c_1\_b_1\_bd_2\_bt_2\_nu_3$ (priv room, good reviews)	0.241
$US\_lt_2\_pn_2\_pg_2\_r_5\_5s_2\_c_1\_b_1\_bd_2\_bt_2\_nu_3$ (cheaper, bad reviews)	0.169
$US\_lt_3\_pn_1\_pg_1\_r_5\_5s_3\_c_1\_b_1\_bd_2\_bt_2\_nu_3$ (shared room, bad reviews)	0.121

User type과 유사한 Listing type을 확인 가능

04 EXPERIMENTS

Training Listing Embeddings

- 8억개의 클릭 세션을 사용
- 30초 미만 머무른 페이지 사용 X
- 2개 이상의 클릭이 발생한 세션만 사용
- 예약 세션의 경우 5배 오버 샘플링 하여 사용
- Embedding은 32차원으로 설정
- Widow는 5로 설정, 10 번 반복으로 설정

Offline Evaluation of Listing Embeddings

- 추천된 listing 중에서 가장 최근 클릭을 바탕으로 얼마나 많이 예약하는지 확인
- 표는 Embedding의 유사성을 기반으로 re-ranking 했을 때의 평가
- Y축은 예약된 Listing의 평균 순위
- X축은 예약까지의 클릭 수
- D32, booking global, market neg를 활용한 버전의 성능이 가장 높음

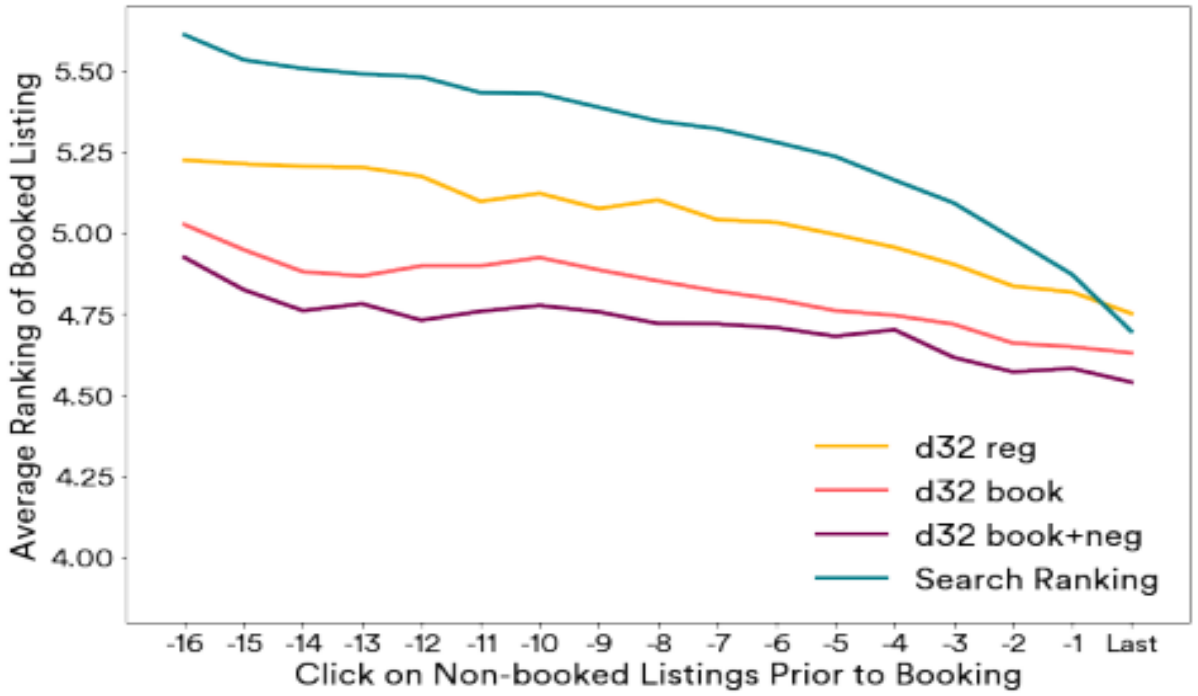


Figure 6: Offline evaluation of Listing Embeddings

Similar Listings using Embeddings

- Airbnb의 페이지는 그림과 같이 동일한 날짜에 사용할 수 있는 유사 추천 목록을 제공
- Embedding Listing을 K-NN 알고리즘을 활용하여 cosine similarity를 계산
- Similar listing들의 CTR 측면에서 21%, 예약 측면에서 4.9% 증가

Figure 3: Similar Listings using Embeddings

Unit 04 | EXPERIMENTS

Real time personalization in Search Ranking using Embeddings

- 사용자가 좋아하는 Listing과 유사한 Listing은 많이 보여주고
- 사용자가 덜 좋아하는 Listing과 유사한 Listing을 덜 보여주고
- 위와 같은 목적을 수행하기 위해 다음 event들의 set을 실시간 수집

(1) H_c: **clicked listing_ids** - listings that user clicked on in last 2 weeks.
(2) H_{lc}: **long-clicked listing_ids** - listing that user clicked and stayed on the listing page for longer than 60 sec.
(3) H_s: **skipped listing_ids** - listings that user skipped in favor of a click on a lower positioned listing
(4) H_w: **wishlisted listing_ids** - listings that user added to a wishlist in last 2 weeks.
(5) H_i: **inquired listing_ids** - listings that user contacted in last 2 weeks but did not book.
(6) H_b: **booked listing_ids** - listings that user booked in last 2 weeks.

Table 6: Embedding Features for Search Ranking

Feature Name	Description
EmbClickSim	similarity to clicked listings in H_c
EmbSkipSim	similarity to skipped listings H_s
EmbLongClickSim	similarity to long clicked listings H_{lc}
EmbWishlistSim	similarity to wishlisted listings H_w
EmbInqSim	similarity to contacted listings H_i
EmbBookSim	similarity to booked listing H_b
EmbLastLongClickSim	similarity to last long clicked listing
UserTypeListingTypeSim	user type and listing type similarity

Real time personalization in Search Ranking using Embeddings

- Hc와 candidate listing 들에 대하여 유사도를 구함 (사용자가 2주간 Click한 listing의 집합)
- Hs와 candidate listing 들에 대하여 유사도를 구함 (사용자가 2주간 Skip한 listing의 집합)

$$EmbClickSim(l_i, H_c) = \max_{m \in M} cos(\mathbf{v}_{l_i}, \sum_{l_h \in m, l_h \in H_c} \mathbf{v}_{l_h})$$

$$EmbSkipSim(l_i, H_s) = \max_{m \in M} cos(\mathbf{v}_{l_i}, \sum_{l_h \in m, l_h \in H_s} \mathbf{v}_{l_h})$$

Real time personalization in Search Ranking using Embeddings

- 앞에서 만든 유사도들(이외에도 많음)을 New GBDT Search Ranking Model의 feature로 활용

Table 7: Embedding Features Coverage and Importances

Feature Name	Coverage	Feature Importance
EmbClickSim	76.16%	**5/104**
EmbSkipSim	78.64%	**8/104**
EmbLongClickSim	51.05%	20/104
EmbWishlistSim	36.50%	47/104
EmbInqSim	20.61%	12/104
EmbBookSim	8.06%	46/104
EmbLastLongClickSim	48.28%	11/104
UserTypeListingTypeSim	86.11%	22/104

Real time personalization in Search Ranking using Embeddings

- EmbClick 값이 클 수록 모델의 점수는 높아짐
- EmbSkipSim 값이 클 수록 모델의 점수가 낮아짐
- UserTypeListingTypeSim 값이 클 수록 모델의 점수가 높아짐

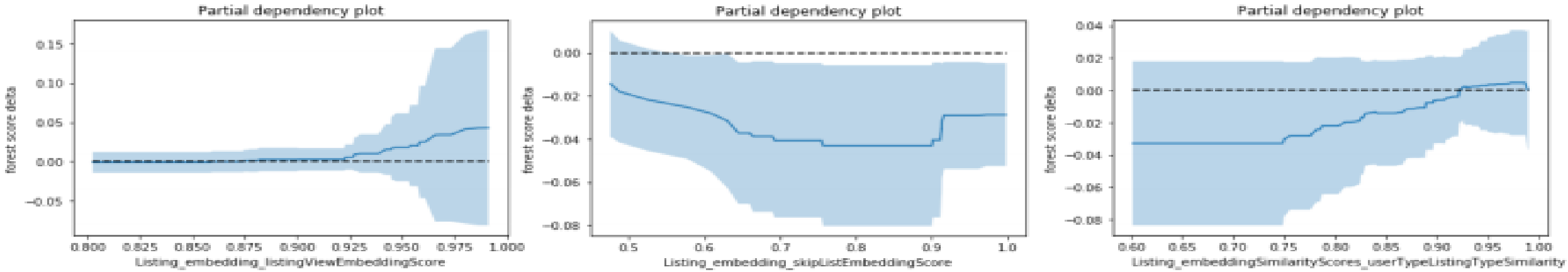


Figure 7: Partial Dependency Plots for EmbClickSim, EmbSkipSim and UserTypeListTypeSim

05 CONCLUSION

Unit 05 | CONCLUSION

- 검색 랭킹에서 실시간 개인화를 위한 새로운 방법 제안
- 저차원에 listing과 user을 표현
- 학습 시에 global context, explicit neg signal라는 개념을 정립
- 성공적인 Test 후 2017년에 Application에 배포하여 실시간 개인화 가능을 탑재

들어주셔서 감사합니다.

참고 자료

- **Mihajlo Grbovic & Haibin Cheng. (2018). Real – time Personalization using Embedding for Search Ranking at Airbnb. *KDD***
- **https://brunch.co.kr/@andrewhwan/54#comment**
- **https://medium.com/mighty-data-science-bootcamp/airbnb-listing-embeddings-in-search-ranking-d0f39116fc44**
- **https://papers.nips.cc/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf**
- **https://lovit.github.io/nlp/representation/2018/03/26/word_doc_embedding/**
- **https://wooono.tistory.com/244**