CUAI Standford Univ. CS224n 스터디(NLP)

Lecture 2 : Word Vectors and Word Senses

발표자 : 황지민

목차

1. Word2Vec
- Gradient Descent
- Stochastic Gradient Descent
- CBOW (Continuous Bag of Words)
- Skip-gram

2. Co-occurrence Matrix
- Window based
- Full document based
- SVD (Singular Value Decomposition)

3. GloVe : Global Vectors for Word Representation

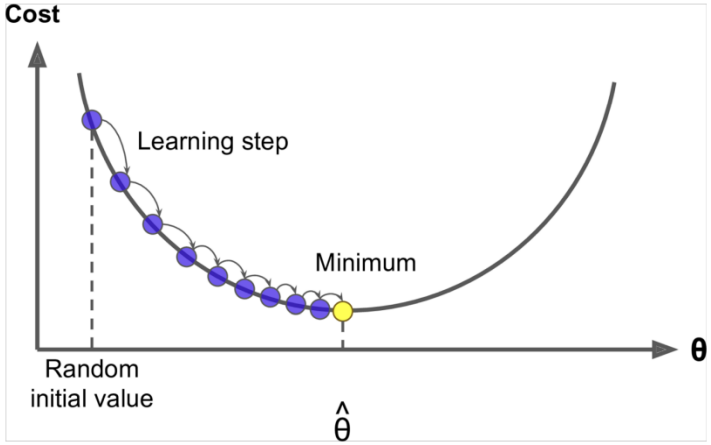
4. Word Embedding Evaluation
- Extrinsic Evaluation
- Intrinsic Evaluation
  1) word analogy
  2) correlation
- Word senses and word sense ambiguity

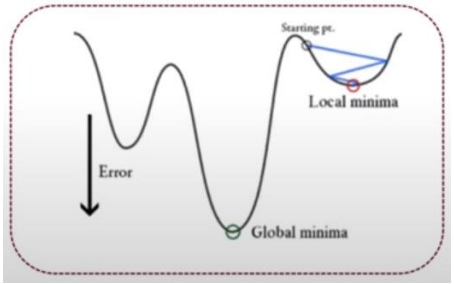
1. Word2Vec

Gradient Descent

Gradient Descent (GD) :
함수의 최소값을 찾기 위한 최적화 알고리즘

→ 랜덤한 θ에서 시작
→ 현재 θ에서 $J(\theta)$의 gradient 계산
→ gradient 절댓값의 감소 방향으로 이동
→ Minimum 값을 갖는 θ 값 찾을 때까지 반복





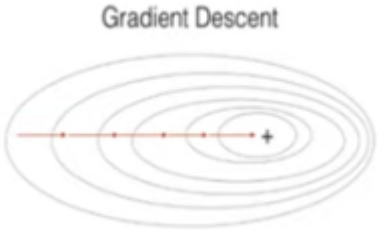
Local minima 에 빠질 수 있음

$$\theta^{new} = \theta^{old} - \alpha \nabla_\theta J(\theta)$$

$$\theta_j^{new} = \theta_j^{old} - \alpha \frac{\partial}{\partial \theta_j^{old}} J(\theta)$$

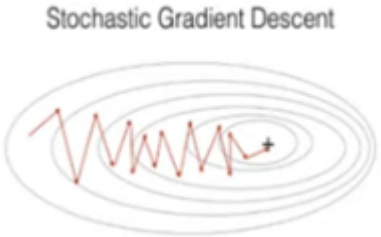
1. Word2Vec

Stochastic Gradient Descent



Gradient Descent의 단점 :
전체 데이터에 대한 계산이 이루어져 계산량이 너무 많음

Stochastic Gradient Descent :
임의의 작은 데이터 샘플 (윈도우) 을 사용해 parameter를 update하는 기법
→ 하나의 데이터만을 고려하므로 길을 헤맬 수 있음
→ $\nabla_\theta J_t(\theta)$ 가 희소행렬 형태




```
while True:
    theta_grad = evaluate_gradient(J,corpus,theta)
    theta = theta - alpha * theta_grad
```

Gradient Descent 알고리즘

```
while True:
    window = sample_window(corpus)
    theta_grad = evaluate_gradient(J,window,theta)
    theta = theta - alpha * theta_grad
```

Stochastic Gradient Descent 알고리즘

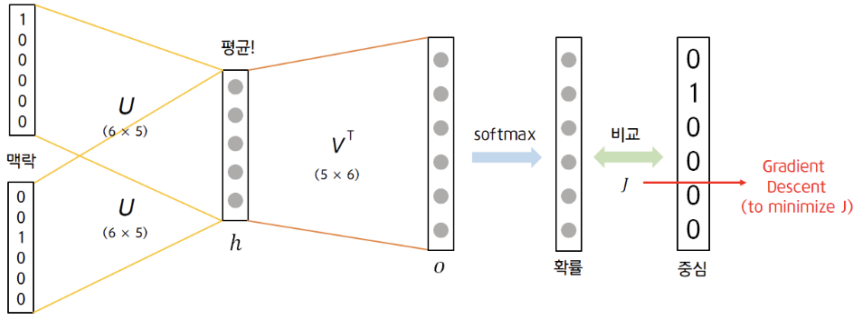
1. Word2Vec

CBOW (Continuous Bag of Words)

입력 벡터 : context vector

출력 벡터 : center vector

→ 맥락 단어들의 맥락에서 가장 잘 맞는 '중심 단어' 예측



1. 맥락 단어의 수집

→ 주어진 텍스트 데이터에서 중심 단어의 주변 단어들을 수집

Ex) I love learning deep learning // window size=2

중심 단어 : learning

맥락 단어 : I, love, deep, learning

2. 중심 단어 예측

→ 모든 맥락 단어들의 벡터를 평균으로 결합하여, 그 결합된 벡터가 중심 단어와 유사해지도록 학습

$$v_{\text{context}} = \frac{1}{2m} \sum_{j \neq t} v_{w_{t+j}}$$

→ hidden layer의 벡터

3. 최종

→ hidden layer의 벡터와 중심 벡터 사이에 있는 가중치 벡터를 곱하여 만든 output layer의 벡터에 softmax 함수 적용

$$P(w_t | w_{t-m}, \ldots, w_{t+m}) = \text{softmax}(v_{\text{context}}^T v_{w_t})$$

→ 중심 벡터와 비교하여 모델의 Loss 계산

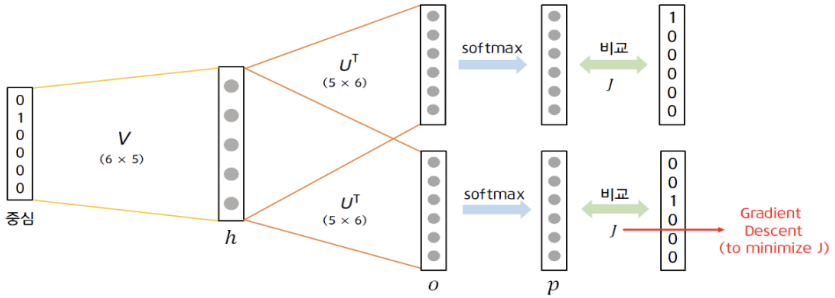
1. Word2Vec

Skip-gram

입력 벡터 : center vector

출력 벡터 : context vector

→ 중심 단어로부터 그 주변에 등장할 수 있는 '맥락 단어' 예측



1. 학습 방식

→ 각 단어를 '중심 단어'로 설정하고, 그 주변의 단어들을 '맥락 단어'로 설정

→ '맥락 단어'는 윈도우 크기에 의해 결정

Ex) I love learning deep learning
// window size = 2
→ 중심 단어와 맥락 단어의 쌍
(learning, I)
(learning, love)
(learning, deep)
(learning, learning)
→ 학습 과정에서 모델이 예측해야 하는 목표

2. 최종

→ 중심 단어 w_c로부터 주변 맥락 단어 w_o가 나올 확률을 최대화하는 것이 목표

$$P(w_o|w_c) = \frac{\exp(\mathbf{u}_o^T \mathbf{v}_c)}{\sum_{w \in V} \exp(\mathbf{u}_w^T \mathbf{v}_c)}$$

1. Word2Vec

CBOW와 Skip-gram의 장단점

- **CBOW :** 여러 맥락 단어들을 사용해 중심 단어 예측

장점 : 학습 속도가 빠르고, 자주 등장하는 단어에 적합
단점 : 희귀한 단어에 대한 학습에 덜 효과적

- **Skip-gram :** 중심 단어를 사용해 여러 개의 맥락 단어들 예측

장점 : 희귀한 단어 학습에 강함, 복잡한 관계 학습 가능
단점 : 계산량이 많고, 학습 속도가 느림

2. Co-occurrence Matrix

Window based

Count-based의 Co-occurrence Matrix 등장 배경 :

① 앞에서 다룬 Word2Vec은 '전체 단어의 동반출현 빈도수'와 같은 통계 정보 내포 불가능

② 벡터의 값들은 중심 단어가 주어졌을 때 각 값의 개별적인 등장 확률을 의미하기 때문

→ Count-based의 Co-occurrence Matrix 등장

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

'I like'
'like I' 'like deep'
'deep like' 'deep learning'
'learning deep' 'learning .'
'. learning'

Window based co-occurrence matrix (단어-문맥 행렬) :

한 문장을 기준으로 윈도우에 각 단어가 몇 번 등장하는 지를 세어서 행렬로 구성

→ 위의 예시는 window length가 1, symmetric matrix

2. Co-occurrence Matrix

Full document based

Word-Document matrix (단어-문서 행렬) :

한 문서를 기준으로 각 단어가 몇 번 등장하는 지를 세어서 행렬로 구성

-	doc1	doc2	doc3
나	1	0	0
는	1	1	2
학교	1	1	0
에	1	1	0
가	1	1	0
ㄴ	1	0	0
다	1	0	1
영희	0	1	1
좋	0	0	1

Count-based matrix의 치명적인 단점 :

단어의 개수가 증가할수록 차원이 기하급수적으로 증가

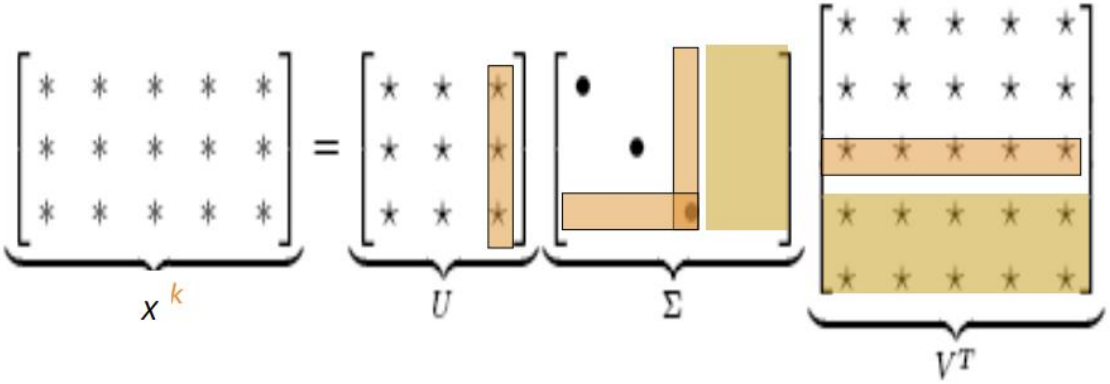
→ SVD, LSA 등을 이용해서 **차원을 축소**시켜야 함

2. Co-occurrence Matrix

SVD (Singular Value Decomposition)

SVD (Singular Value Decomposition) :

임의의 mxn 차원의 행렬 X에 대하여 세 개의 행렬로 분해하는 방법



X : 원본 공동 발생 행렬 (mxn)

U : 단어 벡터들을 나타내는 직교 행렬 (mxm, orthogonal)

Σ : 특이값들로 구성된 대각 행렬 (mxn, diagonal)

V^T : 문맥 벡터를 나타내는 직교 행렬 (nxn, orthogonal)

2. Co-occurrence Matrix

SVD (Singular Value Decomposition)

Example of SVD

$$X = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}$$

X의 크기 : 3 x 2

---

1. 특이값 Σ 계산 (3x2)

→ $X^T X$ 계산

→ $X^T X$ 의 고유값 계산

→ 특이값은 고유값의 제곱근

$$X^T X = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

$$\det(X^T X - \lambda I) = 0 \Rightarrow \det \begin{pmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{pmatrix} = 0$$

$$\Sigma = \begin{pmatrix} \sqrt{3} & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$$

2. Co-occurrence Matrix

SVD (Singular Value Decomposition)

Example of SVD

2. 행렬 V^T 계산 (2x2)

→ X^TX 의 고유벡터 계산

$$\lambda_1 = 3$$

$$(X^TX - 3I)v = 0$$

$$\begin{pmatrix} 2-3 & 1 \\ 1 & 2-3 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$v_1 = \frac{1}{\sqrt{2}},\ v_2 = \frac{1}{\sqrt{2}}$$

$$\lambda_2 = 1$$

$$(X^TX - I)v = 0$$

$$\begin{pmatrix} 2-1 & 1 \\ 1 & 2-1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$v_1 = -\frac{1}{\sqrt{2}},\ v_2 = \frac{1}{\sqrt{2}}$$

$$V^T = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}$$

2. Co-occurrence Matrix

SVD (Singular Value Decomposition)

Example of SVD

3. 행렬 U 계산 (3x3)

→ XX^T 계산

→ XX^T 의 고유값과 고유벡터 계산

$$XX^T = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 2 \end{pmatrix}$$

$$U = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$X = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} \quad U = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \Sigma = \begin{pmatrix} \sqrt{3} & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix} \quad V^T = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}$$

2. Co-occurrence Matrix

SVD (Singular Value Decomposition)

Example of SVD

$$X_{\text{reduced}} = U_{\text{reduced}} \Sigma_{\text{reduced}} V^T_{\text{reduced}}$$

4. 차원 축소

→ Σ 행렬에서 상위 1개 특이값만 유지
→ U 행렬에서 상위 1개 행만 유지
→ V^T 행렬에서 상위 1개 행만 유지
→ 행렬 곱을 통해 근사된 행렬 계산

$$X_{\text{reduced}} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \\ 0 \end{pmatrix} \begin{pmatrix} \sqrt{3} & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}$$

5. 장점

→ 원래 행렬의 중요한 정보를 보존하면서 데이터가 근사됨
→ 데이터의 크기를 줄이면서 중요한 패턴을 유지할 수 있음

$$X_{\text{reduced}} = \begin{pmatrix} \frac{\sqrt{3}}{2} & \frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & \frac{\sqrt{3}}{2} \\ 0 & 0 \end{pmatrix}$$

3. GloVe : Global Vectors for Word Representation

Count based vs direct prediction

- LSA, HAL (Lund & Burgess),
- COALS, Hellinger-PCA (Rohde et al, Lebret & Collobert)

- Fast training
- Efficient usage of statistics
- Primarily used to capture word similarity
- Disproportionate importance given to large counts

- Skip-gram/CBOW (Mikolov et al)
- NNLM, HLBL, RNN (Bengio et al; Collobert & Weston; Huang et al; Mnih & Hinton)

- Scales with corpus size
- Inefficient usage of statistics
- Generate improved performance on other tasks
- Can capture complex patterns beyond word similarity

GloVe

통계정보를 포함한
Direct prediction embedding

Count based : 빠른 훈련 속도 + 효율적으로 통계정보 사용 / 단어 간 관계 파악은 불가

Direct prediction : 높은 수준의 성능 + 단어 유사성 이상의 복잡한 패턴 파악 / 말뭉치 크기가 성능에 영향

3. GloVe : Global Vectors for Word Representation

Original paper analysis

GloVe: Global Vectors for Word Representation

Jeffrey Pennington, Richard Socher, Christopher D. Manning
Computer Science Department, Stanford University, Stanford, CA 94305
jpennin@stanford.edu, richard@socher.org, manning@stanford.edu

GloVe의 basic idea :

→ 임베딩된 단어벡터 간 유사도 측정을 수월하게 한다 (word2vec의 장점)

→ 말뭉치 전체의 통계 정보를 반영한다 (co-occurrence matrix의 장점)

→ **임베딩된 두 단어벡터의 내적이 말뭉치 전체에서의 동시 등장확률 로그값이 되도록 목적함수 정의**

3. GloVe : Global Vectors for Word Representation

Original paper analysis

Notation 정리 :

X_{ik} : 전체 말뭉치 중에서 사용자가 정한 window 내에, i번째 단어와 k번째 단어가 동시에 등장하는 횟수 (앞서 co-occurrence matrix에서 본 행렬)

X_i : 전체 말뭉치 중에서 사용자가 정한 window 내에, i번째 단어가 등장하는 횟수

$P_{ik} = P(k|i) = \frac{X_{ik}}{X_i}$: i번째 단어 주변에 k번째 단어가 등장할 조건부 확률

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

순수한 확률 수치 정보보다는 'ratio'가 relevant와 irrelevant를 구별하기 쉽다

3. GloVe : Global Vectors for Word Representation

Original paper analysis

→ 임베딩된 두 단어벡터의 내적이 말뭉치 전체에서의 동시 등장확률 로그값이 되도록 목적함수 정의

① 단어들의 동시 등장 비율 정보인 $\frac{P_{ik}}{P_{jk}}$ 를 인코딩하여 단어 벡터 공간 안에 표현

$$F(w_i, w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

② 단어 i, j가 각각 단어 k와 동시 등장할 확률의 차이를 보는 것이 목적이므로 Vector difference를 사용한 식으로 목적함수 F 수정

$$F(w_i - w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

③ 좌변은 벡터, 우변은 스칼라 값을 가지므로, 내적을 사용하게끔 목적함수 F 수정

$$F\left((w_i - w_j)^T \tilde{w}_k\right) = \frac{P_{ik}}{P_{jk}}$$

3. GloVe : Global Vectors for Word Representation

Original paper analysis

→ 임베딩된 두 단어벡터의 내적이 말뭉치 전체에서의 동시 등장확률 로그값이 되도록 목적함수 정의

GloVe 조건 :

① Center word는 context word 로도 등장할 수 있기 때문에 단어 벡터간 교환법칙이 성립해야 한다.

$$w \leftrightarrow \tilde{w}$$

② Co-occurrence Matrix 는 Symmetric 하다.

$$X \leftrightarrow X^T$$

③ 목적함수 F 는 Homomorphism 조건에 만족한다.

$$F(X + Y) = F(X) \times F(Y)$$

→ 논문에서는 $F = e^x$ 를 사용

3. GloVe : Global Vectors for Word Representation

Original paper analysis

→ 임베딩된 두 단어벡터의 내적이 말뭉치 전체에서의 동시 등장확률 로그값이 되도록 목적함수 정의

$$F\left((w_i - w_j)^T \tilde{w}_k\right) = \frac{P_{ik}}{P_{jk}}$$

$$\boldsymbol{w_i^T \widetilde{w_k} = logP_{ik} = logP(k|i) = logX_{ik} - logX_i}$$

(F는 지수함수이므로)

$$\boldsymbol{w_i^T \widetilde{w_k} = \log(P_{ik}) = \log(X_{ik}) - \log(X_i)}$$
$$\boldsymbol{w_k^T \widetilde{w_i} = \log(P_{ki}) = \log(X_{ki}) - \log(X_k)}$$

교환 법칙 성립해야 하므로

$$\boldsymbol{w_i^T \widetilde{w_k} + b_i + \widetilde{b_k} = \log(X_{ik})}$$

Bias term을 추가한다

3. GloVe : Global Vectors for Word Representation

Original paper analysis

→ 임베딩된 두 단어벡터의 내적이 말뭉치 전체에서의 동시 등장확률 로그값이 되도록 목적함수 정의

$$J = \sum_{i,j=1}^{V} f\left(X_{ij}\right)\left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij}\right)^2$$

→ 만약 i와 k가 동시에 나타나지 않는다면 $X_{ik} = 0$, 그러면 $log(X_{ik})$는 무한대로 발산

Weighting Function :

① f(0) = 0

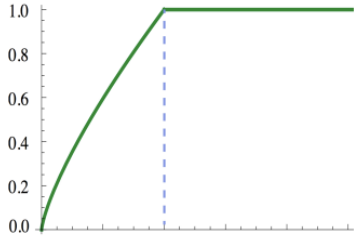
② f(x)는 감소함수가 되면 안 됨

③ f(x)는 너무 자주 나타나는 x에게 너무 큰 가중치를 주면 안 됨

$$f(X_{ij}) = \begin{cases} \left(\frac{x}{x_{max}}\right)^{\alpha} & if\ x < x_{max} \\ 1, & otherwise \end{cases}$$

→ $x_{max} = 100, \alpha = \frac{3}{4}$ 일 때 좋은 성능 보임

$f \sim$

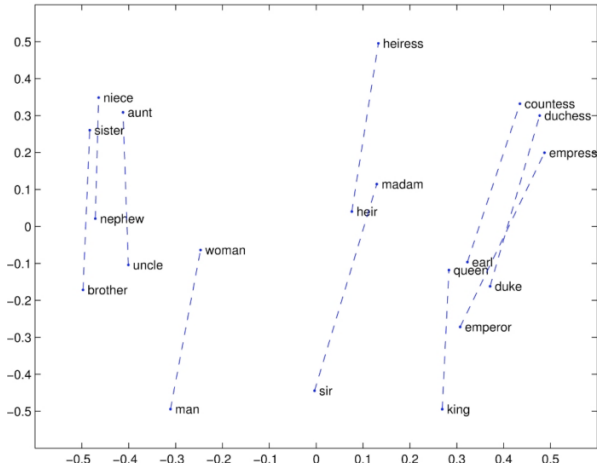


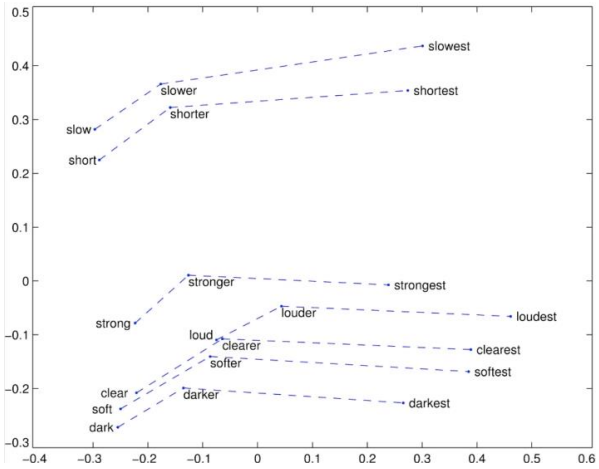
3. GloVe : Global Vectors for Word Representation

Results

Nearest words to frog:

1. frogs
2. toad
3. litoria
4. leptodactylidae
5. rana
6. lizard
7. eleutherodactylus





① frog와 형태적으로 또는 의미적으로 비슷한 단어를 잘 선택
② 반의어 관계에 있는 여러 단어쌍들이 비슷한 간격으로 2차원의 공간 내에 위치
③ 단어의 원형-비교급-최상급을 2차원 공간에 표현

4. Word Embedding Evaluation

Extrinsic Evalution

Extrinsic Evaluation :
실제 현재 문제에서 직접 적용해서 성능을 평가하는 방식

Model	Dev	Test	ACE	MUC7
Discrete	91.0	85.4	77.4	73.4
SVD	90.8	85.7	77.3	73.7
SVD-S	91.0	85.5	77.6	74.3
SVD-L	90.5	84.8	73.6	71.5
HPCA	92.6	**88.7**	81.7	80.7
HSMN	90.5	85.7	78.7	74.7
CW	92.2	87.4	81.7	80.2
CBOW	93.1	88.2	82.2	81.1
GloVe	**93.2**	88.3	**82.9**	**82.2**

→ GloVe 모델이 NER(이름 인식) 작업에 대한 높은 F1 Score 을 보임

4. Word Embedding Evaluation

Intrinsic Evalution

Intrinsic Evaluation :

평가를 위한 데이터(subtask)에 적용하여 성능을 평가

1) Word Analogy :

a : b :: c : ? 에서 ?에 들어갈 단어를 유추하는 문제

$$d = \arg\max_i \frac{(x_b - x_a + x_c)^T x_i}{||x_b - x_a + x_c||}$$

→ 단어 임베딩 모델에서 단어 간의 관계를 파악하고 유추 문제를 해결할 때 사용

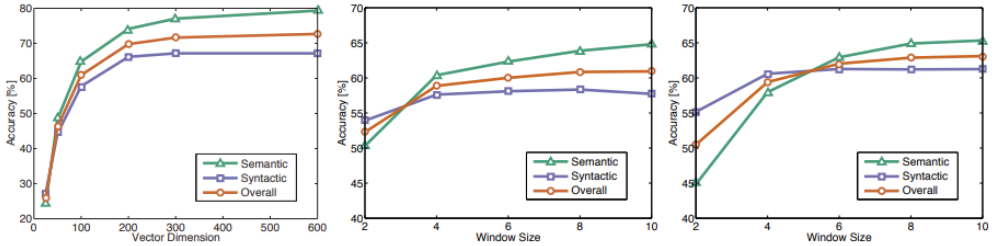
: city-in-state
Chicago Illinois Houston Texas
Chicago Illinois Philadelphia Pennsylvania
Chicago Illinois Phoenix Arizona
Chicago Illinois Dallas Texas
Chicago Illinois Jacksonville Florida
Chicago Illinois Indianapolis Indiana
Chicago Illinois Austin Texas
Chicago Illinois Detroit Michigan
Chicago Illinois Memphis Tennessee
Chicago Illinois Boston Massachusetts

: gram4-superlative
bad worst big biggest
bad worst bright brightest
bad worst cold coldest
bad worst cool coolest
bad worst dark darkest
bad worst easy easiest
bad worst fast fastest
bad worst good best
bad worst great greatest

4. Word Embedding Evaluation

Intrinsic Evalution

Model	Dim.	Size	Sem.	Syn.	Tot.
ivLBL	100	1.5B	55.9	50.1	53.2
HPCA	100	1.6B	4.2	16.4	10.8
GloVe	100	1.6B	67.5	54.3	60.3
SG	300	1B	61	61	61
CBOW	300	1.6B	16.1	52.6	36.1
vLBL	300	1.5B	54.2	64.8	60.0
ivLBL	300	1.5B	65.2	63.0	64.0
GloVe	300	1.6B	80.8	61.5	70.3
SVD	300	6B	6.3	8.1	7.3
SVD-S	300	6B	36.7	46.6	42.1
SVD-L	300	6B	56.6	63.0	60.1
CBOW†	300	6B	63.6	67.4	65.7
SG†	300	6B	73.0	66.0	69.1
GloVe	300	6B	77.4	67.0	71.7
CBOW	1000	6B	57.3	68.9	63.7
SG	1000	6B	66.1	65.1	65.6
SVD-L	300	42B	38.4	58.2	49.2
GloVe	300	42B	**81.9**	**69.3**	**75.0**



(a) Symmetric context (b) Symmetric context (c) Asymmetric context

Dimensionality Window size Window size

→ 여러 임베딩 모델에 대해서 analogy 분석을 진행한 결과, GloVe가 좋은 성능을 보임

→ Semantic 관계가 '벡터 차원'과 '윈도우 크기'에 더 민감하게 반응

4. Word Embedding Evaluation

Correlation

Correlation :

일련의 단어 쌍을 미리 구성한 후에 사람이 평가한 점수와, 단어 벡터 간 코사인 유사도 사이의 상관관계를 계산해 단어 임베딩의 품질을 평가하는 방식

Word 1	Word 2	Human (mean)
tiger	cat	7.35
tiger	tiger	10
book	paper	7.46
computer	internet	7.58
plane	car	5.77
professor	doctor	6.62
stock	phone	1.62
stock	CD	1.31
stock	jaguar	0.92

Model	Size	WS353	MC	RG	SCWS	RW
SVD	6B	35.3	35.1	42.5	38.3	25.6
SVD-S	6B	56.5	71.5	71.0	53.6	34.7
SVD-L	6B	65.7	72.7	75.1	56.5	37.0
CBOW†	6B	57.2	65.6	68.2	57.0	32.5
SG†	6B	62.8	65.2	69.7	58.1	37.2
GloVe	6B	65.8	72.7	77.8	53.9	38.1
SVD-L	42B	74.0	76.4	74.1	58.3	39.9
GloVe	42B	**75.9**	**83.6**	**82.9**	**59.6**	**47.8**
CBOW*	100B	68.4	79.6	75.4	59.4	45.5

→ GloVe 모델이 좋은 성능을 보임

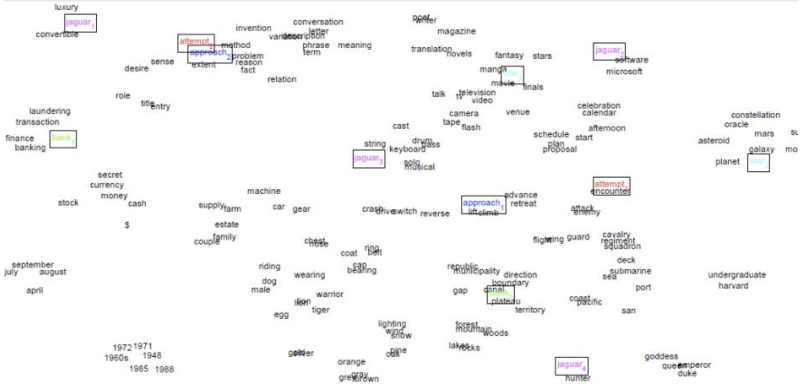
4. Word Embedding Evaluation

Word senses and word sense ambiguity

다의어를 해결하는 방법 2가지 제시 :

① Improving Word Representations Via Global Context And Multiple Word Prototypes (Huang et al. 2012)

② Linear Algebraic Structure of Word Senses, with Applications to Polysemy (Arora, ⋯, Ma, ⋯, TACL 2018)



$$v_{\text{pike}} = \alpha_1 v_{\text{pike}_1} + \alpha_2 v_{\text{pike}_2} + \alpha_3 v_{\text{pike}_3}$$

$$\text{Where } \alpha_1 = \frac{f_1}{f_1+f_2+f_3}\text{, etc., for frequency } f$$

tie				
trousers	season	scoreline	wires	operatic
blouse	teams	goalless	cables	soprano
waistcoat	winning	equaliser	wiring	mezzo
skirt	league	clinching	electrical	contralto
sleeved	finished	scoreless	wire	baritone
pants	championship	replay	cable	coloratura

→ 특정 단어의 윈도우들을 클러스팅 후, 단어 중심으로 다시 임베딩

→ 각 의미에 가중치를 부여하고 선형결합을 통해 새로운 단어 벡터 생성