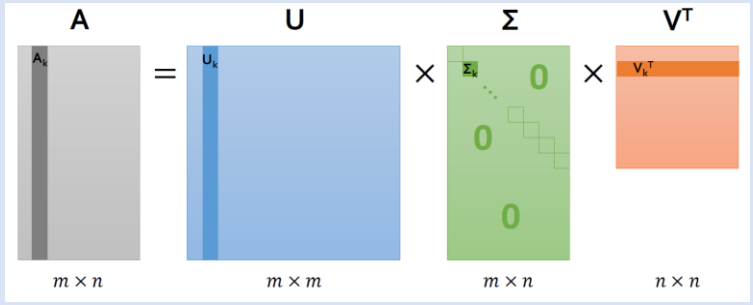
Sentence Embeddings Using Korean Corpora

단어 수준 임베딩 : Glove, Swivel

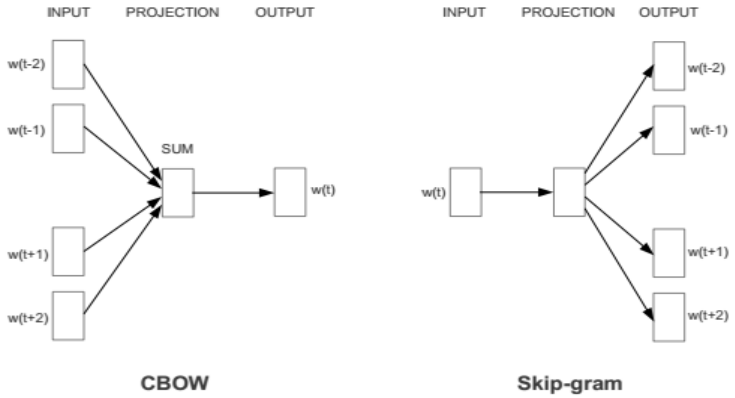
Glove(Global Word Vectors)

Word2Vec과 잠재 의미 분석 두 기법의 단점을 극복하기 위해 제안된 단어 임베딩 기법

LSA: Latent Semantic Analysis



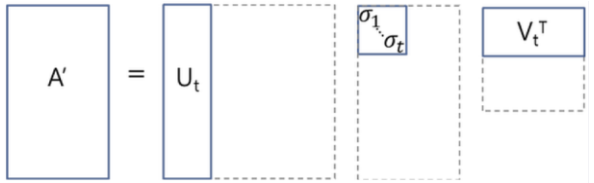
Word2Vec



➢ LSA(잠재 의미 분석)

- 단어행렬 A(tf-idf, PMI, ...)를 "분해" 하는 것
- Why? 차원축소를 통한 효율성 증진, text에 숨어있는 의미를 도출
- How?
  - m개 단어 x n개 문서로 이루어진 행렬 A
  - 특이값 분해(SVD) : (mxn) = (mxm) * (mxn) * (nxn)
  - Truncated SVD : 특이값(∑의 대각성분)중 가장 큰 d개 선택

(mxn) = (mxd) * (dxd) * (dxn)



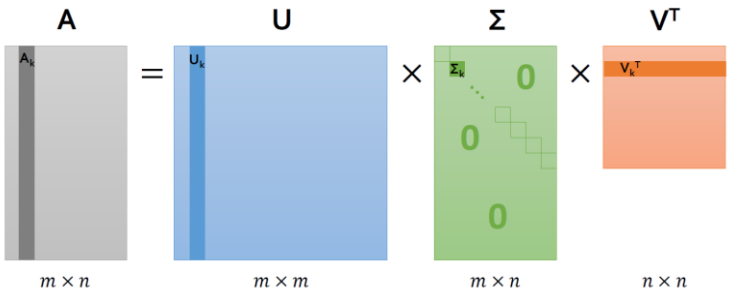
  - U는 단어 임베딩(m개 단어, emb_size=d)
  - V는 문서 임베딩(n개 문서, emb_size=d)
  - **emb_size=n의 단어 임베딩이 emb_size=d(n>d)로 축소**
  - **emb_size=m의 단어 임베딩이 emb_size=d(m>d)로 축소**

- 단어와 문맥 간의 내재적인 의미(Latent/Hidden meaning)을 효과적으로 보존하면서 차원 축소를 통해 효율성 증진(각종 연구)
- However, 단어 간 유사도 측정이 힘듦

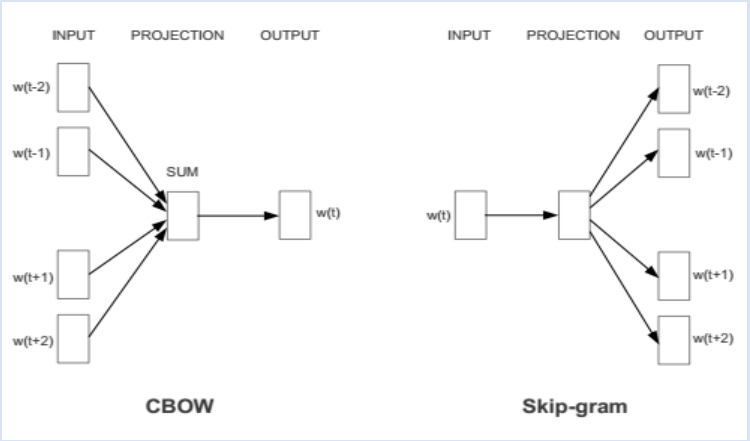
Glove(Global Word Vectors)

Word2Vec과 잠재 의미 분석 두 기법의 단점을 극복하기 위해 제안된 단어 임베딩 기법

LSA: Latent Semantic Analysis



Word2Vec



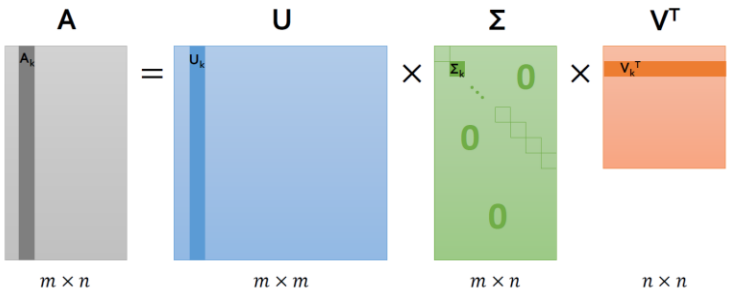
➢ Word2Vec

- CBOW : 문맥 단어를 통해 타깃 단어를 예측
- Skip-gram : 타깃 단어를 통해 문맥 단어를 예측
- Word2Vec 임베딩의 학습은 타깃 단어 & 문맥 단어(positive sample) 사이의 내적을 키우는 방향으로 진행
  - 벡터간 내적은 코사인 유사도와 비례(p. 124)
  - 따라서 Word2Vec 학습시 positive sample과의 코사인 유사도는 높게 산출되며, 이에 따라 단어간 유사도 산출이 용이함
  - However, window size에 따른 로컬 문맥(local context)만 학습에 이용되기 때문에 말뭉치 전체의 통계정보는 반영되기 어려움

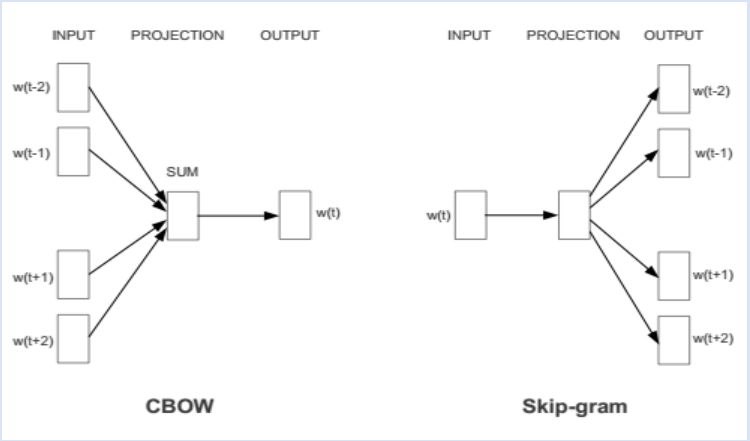
Glove(Global Word Vectors)

Word2Vec과 잠재 의미 분석 두 기법의 단점을 극복하기 위해 제안된 단어 임베딩 기법

LSA: Latent Semantic Analysis



Word2Vec



➢ Word2Vec

- CBOW : 문맥 단어를 통해 타깃 단어를 예측
- Skip-gram : 타깃 단어를 통해 문맥 단어를 예측
- Word2Vec 임베딩의 학습은 타깃 단어 & 문맥 단어(positive sample) 사이의 내적을 키우는 방향으로 진행
  - 벡터간 내적은 코사인 유사도와 비례(p. 124)
  - 따라서 Word2Vec 학습시 positive sample과의 코사인 유사도는 높게 산출되며, 이에 따라 단어간 유사도 산출이 용이함
  - However, window size에 따른 로컬 문맥(local context)만 학습에 이용되기 때문에 말뭉치 전체의 통계정보는 반영되기 어려움

Summary

LSA(잠재 의미 분석)
- pros : 말뭉치 전체의 통계량이 반영되어 학습, 문맥간 유사도 측정 용이
- cons : 단어간 유사도 측정이 힘듦

Word2Vec
- pros : 단어간 유사도 측정이 용이
- cons : window size 내의 통계정보만 반영(말뭉치 전체의 통계정보 반영 x)

Glove
- 말뭉치 전체의 통계정보를 반영하면서 & 단어간 유사도 측정이 가능토록 설계

Glove(Global Word Vectors)

Glove의 목표 : 임베딩된 단어 벡터간 유사도 측정을 수월하게 하면서 말뭉치 전체의 통계적보를 반영

모델 목적함수(loss func)

$$loss = \sum_{i,j=1}^{|V|} f(A_{ij})(U_i \cdot V_j + b_i + b_i - \log(A_{ij}))^2 \qquad where\ F(x) = \begin{cases} (\frac{x}{x_{max}})^{\alpha} & if\ \ x < x_{max} \\ 1 & otherwise \end{cases}$$

- $U_i \cdot V_j$: 각 단어의 임베딩 벡터(초기값은 $random$)
- $log(A_{ij})^2$: log(두 단어의 동시 등장 빈도)

  ** window size k에서 i,j 동시등장빈도*
- $U_i \cdot V_j$ - $log(A_{ij})^2$

- $f(A_{ij})$: 동시 등장 빈도가 높을 수록 가중치를 두되, 한계 설정

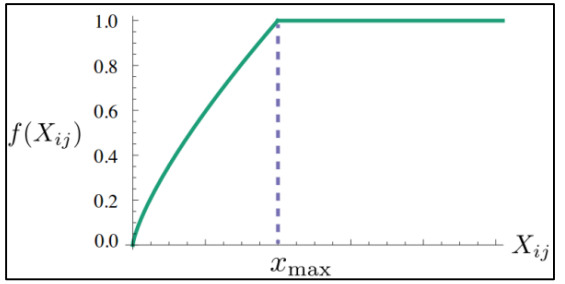
  es) it is , 수 있다

- $b_i,\ b_i$: 목적함수 최소화를 위한 상수항

✓ 두 벡터의 내적은 코사인 유사도와 비례

✓ A_{ij} 는 두 단어의 window size k에서의 동시등장 정도

✓ ∴ 동시 등장 빈도($log(A_{ij})^2$)가 큰 단어의 벡터간 내적이 높게 설정되도록 학습. 내적이 높을 수록 코사인 유사도는 올라감



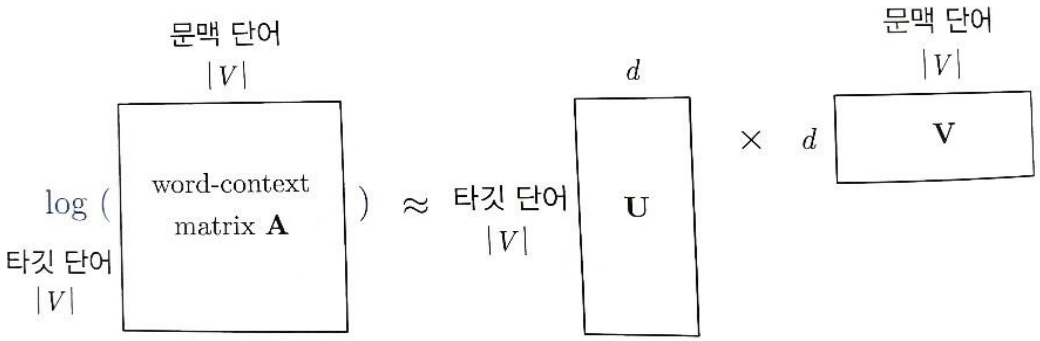
Xmax는 임의의 기준

Glove(Global Word Vectors)

Glove의 목표 : 임베딩된 단어 벡터간 유사도 측정을 수월하게 하면서 말뭉치 전체의 통계적보를 반영

모델 목적함수(loss func)

$$loss = \sum_{i,j=1}^{|V|} f(A_{ij})(U_i \cdot V_j + b_i + b_i - \log(A_{ij}))^2 \qquad where\ F(x) = \begin{cases} (\frac{x}{x_{max}})^{\alpha} & if\ \ x < x_{max} \\ 1 & otherwise \end{cases}$$



특이값 분해를 통해 얻은
d차원 단어 임베딩(전체)

* Glove 학습은 랜덤 초기회된 U/V를 조금씩 update하여 loss 최소화 방향으로 진행

Glove는 W2V의 단점인 말뭉치 전체의 통계치를 반영하지 못한다는 점을 개선하기 위해 개발된 방법이나, Skip-gram 모델에서 window size를 설정해 학습하는 것이 결국 말뭉치 전체 통계량 행렬(SPMI)을 분해하는 것과 동치임이 증명됨 (p. 139~240, 144)

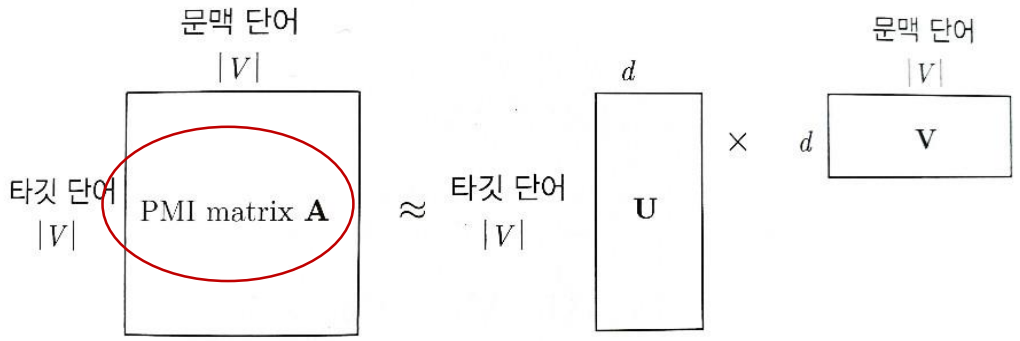
단어 유추 평가(word analogy test)에서 Glove와 Skip-gram 비교

성 능	: 67.1% vs 68.3%	(ACC)
학습 시간	: 4h12m vs 8h38m	
메모리 사용	: 9414mb vs 628mb	(peak RAM)

출처 : https://rare-technologies.com/making-sense-of-word2vec/

Swivel(Submatrix-Wise Vector Embedding Learner)

PMI 행렬을 차원 축소하여 단어 임베딩을 수행



➢ Swivel

- Google 연구팀 발표(2016)
- PMI행렬을 분해해 단어 임베딩에 활용한다는 점에서 단어-문맥 행렬을 분해하는 Glove와 차이를 보임
  (단어-문맥 행렬 → 단어간 상관성 적용 → PMI 행렬)
  (p. 74 ~ 76)
- PMI의 단점(상관성이 매우 떨어지는 단어간 PMI 값은 음의 무한대로 수렴)을 극복하는 방향으로 목적함수 설정

$$if\ P(i,j) > 0:$$
$$loss = \frac{1}{2} f(x_{ij})(U_i \cdot V_i - PMI(i,j))^2$$
$$else:$$
$$loss = \log[1 + \exp(U_i \cdot V_i - PMI^*(i,j))]$$

$if\ P(i,j) > 0:$

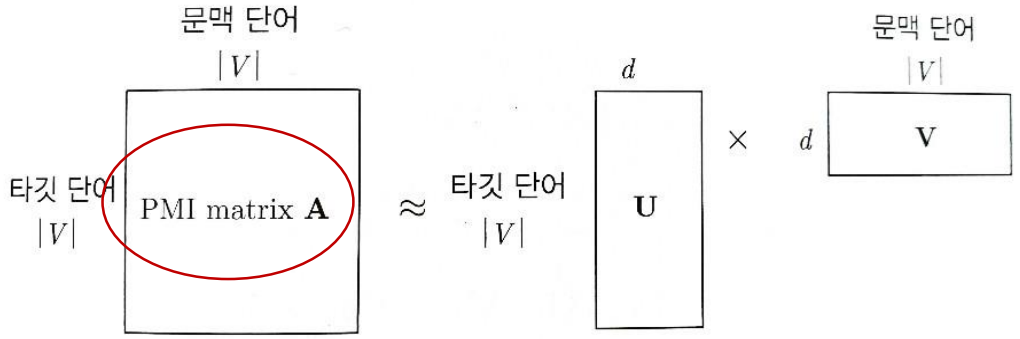
- i, j의 임베딩 벡터 U_i, V_j 의 내적이 두 단어의 PMI값에 근사하도록 벡터를 업데이트
  - PMI(분포 가정에 따른 동시출현 가중치)가 높을 수록 벡터간 내적 증가, 코사인 유사도 상승
- $f(x_{ij})$: *사용자 지정 적절한 보정함수* : $\sqrt{x_{ij}}$

the difference between $w_i^\top \tilde{w}_j$ and $\mathbf{pmi}(i;j)$, tempered by a monotonically increasing weighting function of the observed co-occurrence count, $f(x_{ij})$:

https://arxiv.org/pdf/1602.02215.pdf

Swivel(Submatrix-Wise Vector Embedding Learner)

PMI 행렬을 차원 축소하여 단어 임베딩을 수행



➢ Swivel

- Google 연구팀 발표(2016)
- PMI행렬을 분해해 단어 임베딩에 활용한다는 점에서 단어-문맥 행렬을 분해하는 Glove와 차이를 보임
  (단어-문맥 행렬 → 단어간 상관성 적용 → PMI 행렬)
  (p. 74 ~ 76)
- PMI의 단점(상관성이 매우 떨어지는 단어간 PMI 값은 음의 무한대로 수렴)을 극복하는 방향으로 목적함수 설정

$$if\ P(i,j) > 0:$$
$$loss = \frac{1}{2} f(x_{ij})(U_i \cdot V_i - PMI(i,j))^2$$
$$else:$$
$$loss = \log[1 + \exp(U_i \cdot V_i - PMI^*(i,j))]$$

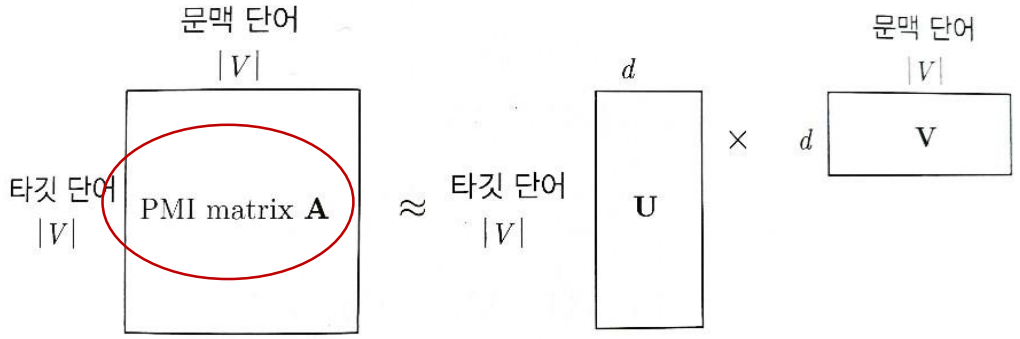
else :

- *P(i,j)*가 0일 경우 log 값인 PMI는 음의 무한대로 발산
  - 동시 출현 빈도가 0인 단어간 loss 함수를 따로 설정
- $PMI^*(i,j)$ → 동시등장 빈도를 0에서 1로 조정한 값.
- *PMI 값이 작기 때문에, 벡터간 내적도 작게 학습됨*

- *i, j가 고빈출 단어라면?*
  - $PMI^*(i,j)$ *감소, 내적이 상대적으로 작게 학습*
  - *고빈출 단어들인데도 불구하고 동시출현이 적은 것이면 정말 상관성이 떨어지는 단어들이다. (종이, 운전))*

Swivel(Submatrix-Wise Vector Embedding Learner)

PMI 행렬을 차원 축소하여 단어 임베딩을 수행



➢ Swivel

- Google 연구팀 발표(2016)
- PMI행렬을 분해해 단어 임베딩에 활용한다는 점에서 단어-문맥 행렬을 분해하는 Glove와 차이를 보임
  (단어-문맥 행렬 → 단어간 상관성 적용 → PMI 행렬)
  (p. 74 ~ 76)
- PMI의 단점(상관성이 매우 떨어지는 단어간 PMI 값은 음의 무한대로 수렴)을 극복하는 방향으로 목적함수 설정

$$if\ P(i,j) > 0:$$
$$loss = \frac{1}{2} f(x_{ij})(U_i \cdot V_i - PMI(i,j))^2$$
$$else:$$
$$loss = \log[1 + \exp(U_i \cdot V_i - PMI^*(i,j))]$$

else :

- *P(i,j)*가 0일 경우 log 값인 PMI는 음의 무한대로 발산
  - 동시 출현 빈도가 0인 단어간 loss 함수를 따로 설정
- $PMI^*(i,j)$ → 동시등장 빈도를 0에서 1로 조정한 값.
- *PMI 값이 작기 때문에, 벡터간 내적도 작게 학습됨*

- *i, j가 저빈출 단어라면?*
  - *$PMI^*(i,j)$ 증가, 내적이 상대적으로 크게 학습*
  - *저빈출 단어들끼리라면 동시출현이 적더라도 의미가 있을 수 있다.*
  - *corpus 자체가 작음, 우연 등으로 인해. (확률, 분포)*