# [GNN] DeepWalk: Online Learning of Social Representations - Paper Review

4 minute read

> 🌌
> *자연어처리 단어 임베딩에 사용된 skip-gram의 아이디어를 차용해, Deep Learning Graph Embedding을 처음 시도해 본 DeepWalk 알고리즘에 대해 배워봅니다!*

## 0. 들어가며

안녕하세요, 배우는 기계 러닝머신입니다!


오늘 살펴볼 논문은 DeepWalk: Online Learning of Social Representations 입니다. 해당 논문의 원문은 이 링크(https://arxiv.org/pdf/1403.6652.pdf)에서 확인할 수 있습니다.

논문의 핵심 아이디어는 다음과 같습니다.

> 1. *Graph 데이터를 저차원 dense representation으로 Embedding*
> 2. *Graph를 자연어의 일반화된 형태로 간주, 자연어 처리의 방법론을 차용해 Embedding 학습*
> 3. *Graph 내를 Random한 패턴으로 일정한 길이만큼 순회하여, 자연어의 sequence와 같은 random walk sequence를 생성*
> 4. *이렇게 획득한 random walk sequence에 Skip-Gram 알고리즘을 적용해 Node Embedding 학습*

### Contents



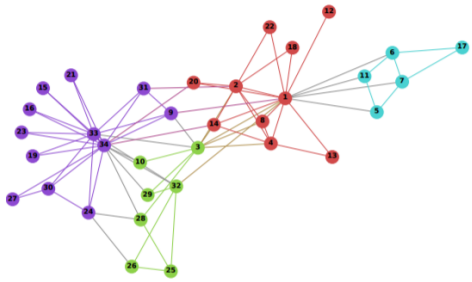
## 1. Introduction

**Introduction**

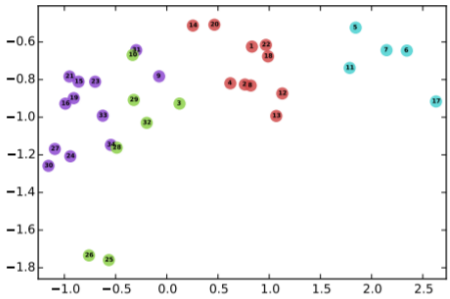
graph 표현은 처리 전의 자연어 형태처럼 sparse한 성질을 가지고 있습니다. 이를 다른 downstream task에 활용할 수 있는 형태로 만들기 위해서는 graph의 low-dimensional dense representation을 얻는 것이 필요합니다. 자연어 처리 임베딩 기법 분야에서 딥러닝의 응용이 성공을 거둠에 따라, graph의 dense representation 학습을 위해 자연어 처리 영역의 알고리즘을 적용할 수 있을 것이라고 판단, 이를 적용해 성공을 거둔 최초의 논문입니다.

이 논문은 short random walk의 sequence를 사용해, 그래프의 꼭지점(vertex, node)의 social representation을 학습하는 DeepWalk라는 알고리즘을 제안합니다.
여기서 social representaion은 꼭지점의 이웃간 유사도와, 집단적인 특성을 포착해 내는 latent feature입니다. 가령, 학회라는 society가 있을 때 각 학자들을 꼭지점(vertex, node)이라고 간주하면, 학자간의 유사한 정도, 그리고 학자의 집단(학회 혹은 같은 학파)의 특성을 (인간이 직접적으로 해석할 수는 없는)수치 벡터로 표현해낸 것이 social representation입니다.

DeepWalk는 그래프 내의 노드간의 특징을 잘 포착해 낼 수 있도록 임베딩을 학습합니다. 이 결과를 시각화한 예시는 다음과 같습니다.



(a) Input: Karate Graph




(b) Output: Representation

왼쪽의 그림은 Zachary's karate club으로 알려진, 가라테 클럽 내의 34명 멤버들의 interaction을 표현한 그래프입니다. 각각의 노드(멤버)는 4개의 클래스에 속해 있습니다.
오른쪽의 결과는 2차원 latent space에 각각 노드를 표현한 결과입니다. 4개의 클래스에 해당하는 노드를 잘 구분되도록 임베딩한 것을 확인할 수 있습니다.

이처럼 DeepWalk를 사용하면 저차원 공간 내에 쉽게 표현할 수 없던 그래프 노드간의 관계를, 2차원과 같은 저차원 공간 내에 의미를 최대한 보존하며 표현해 낼 수 있다는 것입니다.

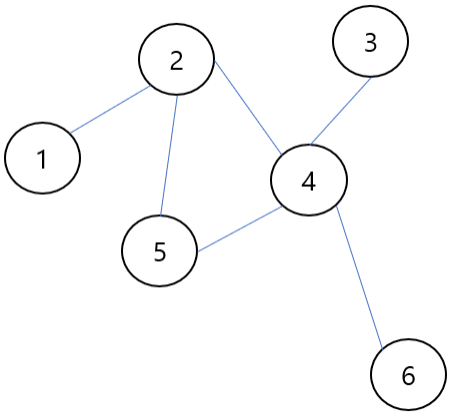
특히 그래프의 label이 sparse함에도 불구하고 예측 성능이 타 알고리즘에 비해 높다는 것, 그리고 쉽게 병렬화 가능하기 때문에 높은 scalability를 보장받을 수 있다는 것을 DeepWalk의 강점으로 꼽았습니다.

# 2. Random Walk

**random-walk**

DeepWalk 알고리즘의 핵심은, 그래프에서 sequence를 생성해, 자연어처리의 Skip-Gram 방식으로 임베딩을 학습하자! 입니다.
이 때, "그래프에서 sequence를 생성"하는 과정이 바로 이 부분이라고 할 수 있습니다.

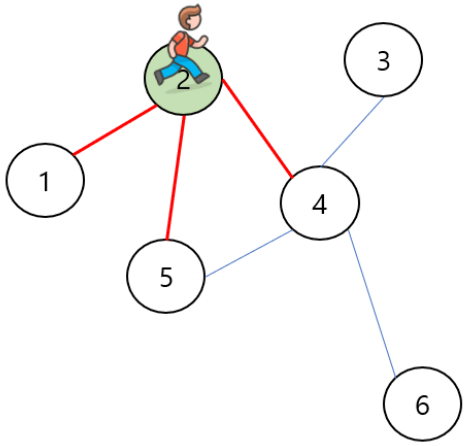
간단한 그림으로 설명하고자 합니다.
먼저, 다음과 같은 그래프가 있습니다.



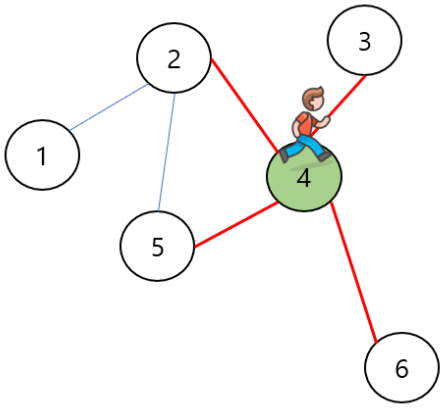
이 위의 한 점을 선택해, root node로 사용합니다. 이 root node에 연결된 노드 중에 무작위적으로(random) 하나를 선택해, 이동(walk) 합니다.
여기서는 root node가 2번 노드가 되었고, 이제 연결된 1,4,5 노드 중 1개를 선택합니다. 주사위를 굴린다고 생각하면 좋을 것 같습니다. random_walk=[] 라는 리스트에, 이동하는 행보를 기록합니다.

> *random_walk=[2]*

이 때 미리 지정해 놓은 random walk의 길이(k라고 하겠습니다.)에 도달할 때까지 계속 무작위적인 이동을 계속하게 됩니다.




주사위를 굴린 결과 4가 나왔네요. 그렇다면 다음처럼 이동하게 되겠지요?




이제 우리의 sequence는 다음처럼 갱신됩니다.

> *random_walk=[2,4]*

이를 k의 길이에 도달할 때까지 반복해, random walk의 sequence를 만듭니다.

이 sequence는, 자연어에서의 단어 토큰의 sequence에 대응됩니다.

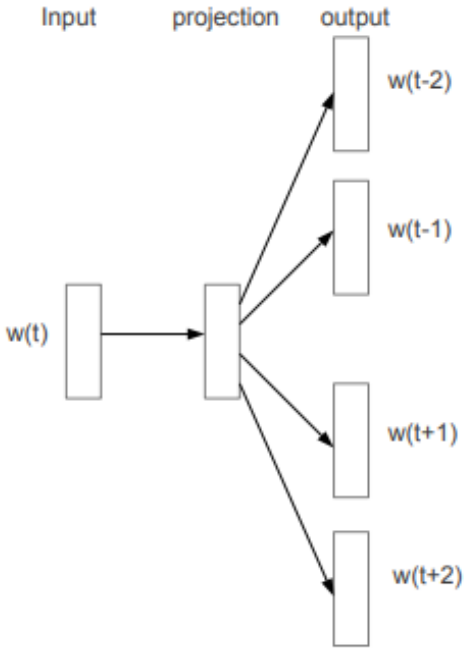
반복 결과, 다음과 같은 sequence를 얻었다고 하겠습니다.

> *random_walk=[2,4,3,4,5,2,5,4,6,4]*

# 3. Skip-Gram

### skip-gram

이렇게 마련한 sequence를 사용해, skip-gram 알고리즘을 적용해 보려고 합니다.
skip-gram은 다음과 같은 그림으로 설명할 수 있습니다.



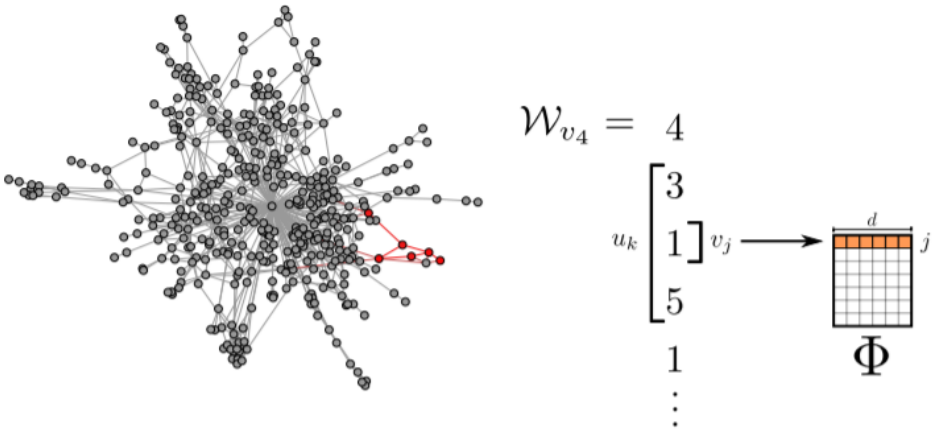

만약 "나는 오늘 도서관에 갔다가 졸려서 죽을 뻔했다." 라는 문장이 있을 때,
[나는, 오늘, 독서실에, 갔다가, 졸려서, 죽을, 뻔했다]라는 식으로 토큰화를 해봅시다(보통은 형태소 단위로 더욱 쪼개어 줍니다).
이 때, 위 그림에서 w(t)가 "독서실에"라는 토큰이라면,
그 주변에 "나는", "오늘", "갔다가","졸려서" 와 같은 단어가 빈번하게 등장할 것입니다.
("정글북","호모에렉투스","정복하다" 등과 같은 단어에 비해서 말이지요.)

이러한 점에 착안해서, 비슷한 위치에 등장하는 단어는 비슷한 의미를 가질 것이라는 가정 하에,
특정 단어가 주어졌을 때 주변의 유관한 단어가 등장할 확률을 극대화할 수 있도록 임베딩을 학습하는 방법이 바로 skip-gram입니다.
그리고 이러한 학습은 특정 단어가 주어졌을 때 주변의 단어를 예측하는 과정에서 이루어 집니다.

앞서 구한 다음 sequence를 보면,

> *random_walk=[2,4,3,4,5,2,5,4,6,4]*

2가 주어졌을 때 주변에 어떤 node가 나올지 예측함에 따라 node의 social한 representation을 학습할 수 있습니다.




(a) Random walk generation. (b) Representation mapping.

아래는, 논문에 첨부된 DeepWalk 알고리즘의 수도 코드입니다.

**Algorithm 1** DeepWalk($G$, $w$, $d$, $\gamma$, $t$)

**Input:** graph $G(V, E)$
window size $w$
embedding size $d$
walks per vertex $\gamma$
walk length $t$
**Output:** matrix of vertex representations $\Phi \in \mathbb{R}^{|V| \times d}$

Initialization: Sample $\Phi$ from $\mathcal{U}^{|V| \times d}$
Build a binary Tree $T$ from $V$
**for** $i = 0$ to $\gamma$ **do**
  $\mathcal{O} = \text{Shuffle}(V)$
  **for each** $v_i \in \mathcal{O}$ **do**
    $\mathcal{W}_{v_i} = RandomWalk(G, v_i, t)$
    SkipGram($\Phi$, $\mathcal{W}_{v_i}$, $w$)
  **end for**
**end for**

## 4. 마치며

-

이번 시간에는 DeepWalk에 대해 배워보았습니다. 자연어를 그래프의 특수한 형태로 간주하고, 자연어처리에서 성공을 거둔 임베딩 기법을 적용한 아이디어와, random walk를 사용해 빈도로써 순차적 정보를 반영할 수 있도록 sequence를 만들어 낸 방법이 인상적이었습니다.

다음 시간에는, 다른 그래프 임베딩 방법론을 소개하고자 합니다!

**Tags:** Graph Neural Networks | paper review

**Categories:** Graph Neural Networks

**Updated:** September 25, 2020