인공지능과 신약개발을 위한 파이썬

12주차 DeepChem QSAR

홍 성 은

sungkenh@gmail.com

목차

Word Embedding

- 워드 임베딩(Word Embedding)은 단어를 벡터로 표현하는 방법으로, 단어를 밀집 표현으로 변환합니다. 이번 챕터에서는 희소 표현, 밀집 표현, 그리고 워드 임베딩에 대한 개념을 소개함

- 희소 표현 : 원 핫 인코딩을 통해 생성된 단어 벡터, 벡터 또는 행렬의 값이 대부분 0으로 표현되는 방법을 희소 표현(Sparse representation)이라고 함

- 희소 벡터의 문제점은 단어의 개수가 늘어나면 차원이 한없이 커짐(공간적 낭비)
  - Ex) 강아지 = [0 0 0 0 1 0 0 0 0 0 0 0 … 중략 … 0] # 이 때 1 뒤의 0의 수는 9995개.
- 밀집 표현(Dense Representation): 벡터의 차원을 단어 집합의 크기로 상정하지 않고, 일정한 크기로 단어 벡터의 차원을 맞춤 (0과 1만을 갖는 것이 아니라 실수 값을 가짐)
  - Ex) 강아지 = [0.2 1.8 1.1 -2.1 1.1 2.8 … 중략 …] # 이 벡터의 차원은 128

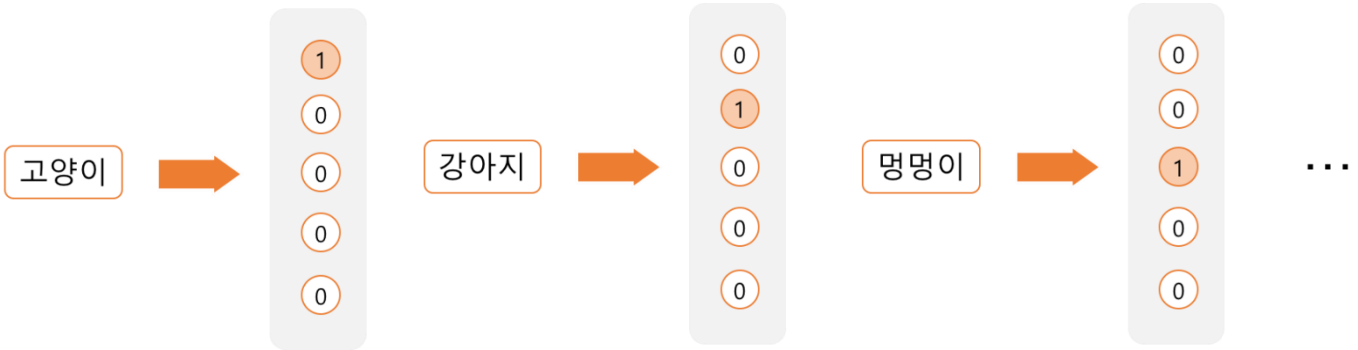
Word Embedding

- 단어를 밀집 벡터(dense vector)의 형태로 표현하는 방법을 워드 임베딩(word embedding)이라고 함
- 이 밀집 벡터를 워드 임베딩 과정을 통해 나온 결과라고 하여 임베딩 벡터(embedding vector)라고도 함

-	원-핫 벡터	임베딩 벡터
차원	고차원(단어 집합의 크기)	저차원
다른 표현	희소 벡터의 일종	밀집 벡터의 일종
표현 방법	수동	훈련 데이터로부터 학습함
값의 타입	1과 0	실수

Word2Vec

- 텍스트 기반의 모델 만들기는 텍스트를 숫자로 바꾸는 과정을 말함
- 숫자로 바꿔야만 알고리즘에 넣고 계산을 한 후 결과값을 낼 수 있기 때문(아래처럼 0과1로된 벡터로 쉽게 바꿀 수 있음)
- 단점, 유사한 의미를 갖고 어떤 단어가 반대의 의미를 갖는지 등 단어 간의 관계는 반영하지 못함
- 단어를 벡터로 바꾸는 모델을 word embedding model이라고 부르며, word2vec이 대표 모델임
- 한국어 워드 임베딩 : https://word2vec.kr/search/



Word2Vec

- 분산 표현(Distibuted Representation)
  - 분산 표현(distributed representation) 방법은 기본적으로 분포 가설(distributional hypothesis)이라는 가정 하에 만들어진 표현 방법임
  - 이 가정은 **'비슷한 위치에서 등장하는 단어들은 비슷한 의미를 가진다'**라는 가정임

  - 강아지란 단어는 귀엽다, 예쁘다, 애교 등의 단어가 주로 함께 등장하는데 분포 가설에 따라서 저런 내용을 가진 텍스트를 벡터화한다면 저 단어들은 의미적으로 가까운 단어가 됨
  - Word2Vec로 임베딩 된 벡터는 굳이 벡터의 차원이 단어 집합의 크기가 될 필요가 없습니다. 강아지란 단어를 표현하기 위해 사용자가 설정한 차원을 가지는 벡터가 되면서 각 차원은 실수형의 값을 가짐
    - Ex) 강아지 = [0.2 0.3 0.5 0.7 0.2 ... 중략 ... 0.2]
  - 요약하면 희소 표현이 고차원에 각 차원이 분리된 표현 방법이었다면, 분산 표현은 저차원에 **단어의 의미를 여러 차원에다가 분산**하여 표현함
  - 이런 표현 방법을 사용하면 **단어 간 유사도**를 계산할 수 있음

Word2Vec

- Feature Representation
  - 데이터는 대상의 속성을 표현해 놓은 자료로, 그것을 바탕으로 모델을 만듦
  - 데이터 분석(기계학습)을 떠올려보면 속성을 사용하여 모델을 만드는 방식을 이해할 수 있음
  - 대상의 속성을 표현하는 방식을 feature representation이라고 부름(독립적인 속성(차원)을 사용)
  - 자연어 처리의 경우 대상은 텍스트이고, 이 텍스트의 속성을 표현한 것이 데이터
  - 속성은 품사, 앞단어 뒷단어, 단어의 길이 등 언어적 정보를 추출하여 feature representation을 하는 방법이 word2vec임 (각각의 속성을 독립적인 차원에 나타내지 않음 고정한 크기에 대응시켜 표현)

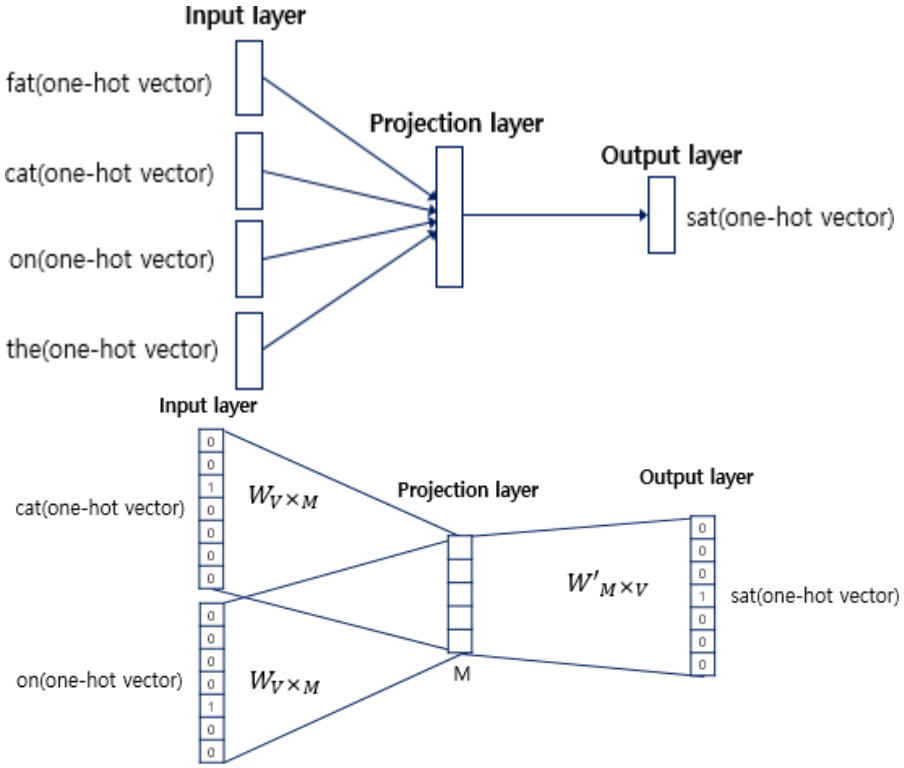
CBOW

- Word2Vec에는 CBOW(Continuous Bag of Words)와 Skip-Gram 두 가지 방식이 있음
  - CBOW는 **주변에 있는 단어들을** 가지고, **중간에 있는 단어들을 예측**하는 방법
  - Skip-Gram은 **중간에 있는 단어로 주변 단어들을 예측**하는 방법
  - **예문 : "The fat cat sat on the mat“,** {"The", "fat", "cat", "on", "the", "mat"}으로부터 sat을 예측
- 중심 단어를 예측하기 위해서 앞, 뒤로 몇 개의 단어를 볼지를 결정했다면 이 범위가 **윈도우(window)**
  - 윈도우 크기가 n이면, 중심 단어 예측을 위한 주변 단어 개수는 2n
- 윈도우 크기를 정했다면, 윈도우를 계속 움직여서 주변 단어와 중심 단어 선택을 바꿔가며 학습을 위한 데이터 셋을 만들 수 있는데, 이 방법을 **슬라이딩 윈도우(sliding window)**라고 함

중심 단어	주변 단어
[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, 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, 1]	[0, 0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 0, 1, 0]

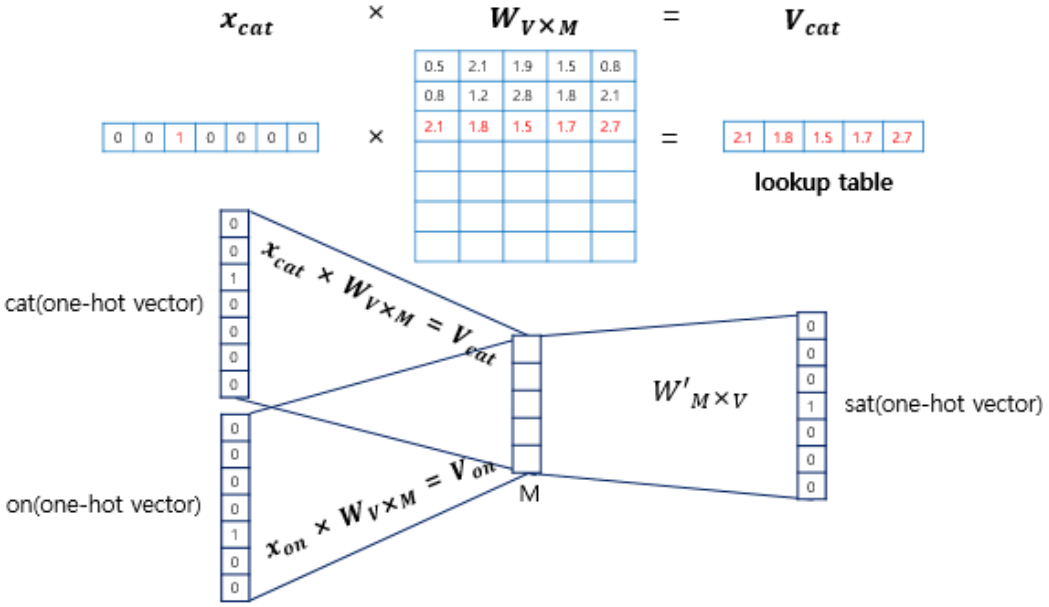
CBOW

- 입력층(Input layer)의 입력으로서 앞, 뒤로 사용자가 정한 윈도우 크기 범위 안에 있는 주변 단어들의 원-핫 벡터가 들어가게 되고, 출력층(Output layer)에서 예측하고자 하는 중간 단어의 원-핫 벡터가 필요함
- Word2Vec은 딥 러닝 모델(Deep Learning Model)은 아니라는 점
- 이렇게 은닉층(hidden Layer)이 1개인 경우에는 일반적으로 심층신경망(Deep Neural Network)이 아니라 얕은신경망(Shallow Neural Network)이라고 부름



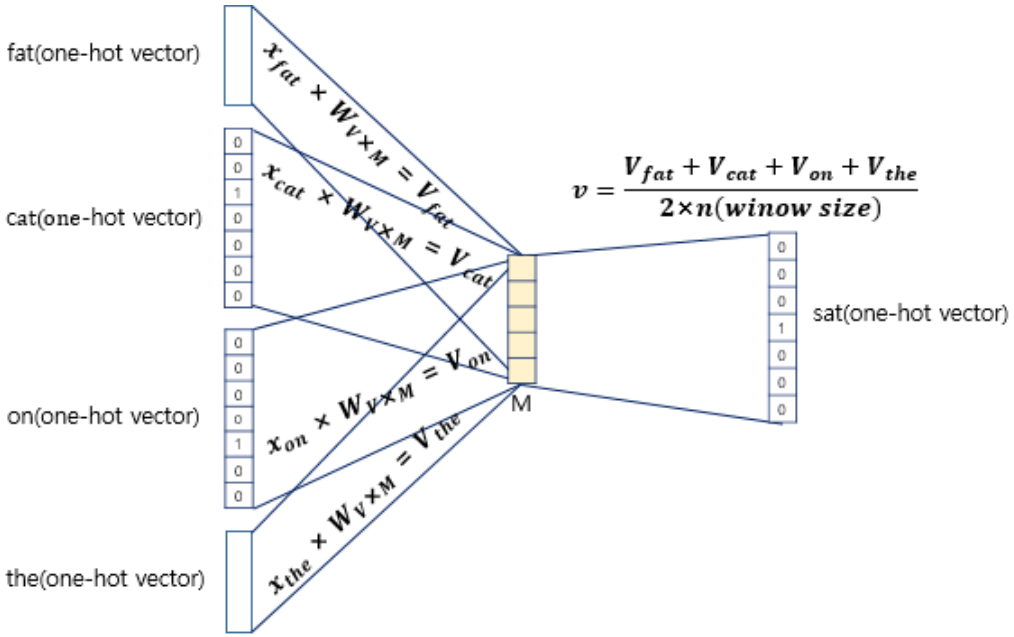
CBOW

- 주목해야할 것은 두 가지임
  - 하나는 **투사층의 크기가 M**이라는 점
    - CBOW에서 투사층의 크기 M은 임베딩하고 난 벡터의 차원이 됨
    - 위의 그림에서 투사층의 크기는 M=5이기 때문에 CBOW를 수행하고나서 얻는 각 단어의 임베딩 벡터의 차원은 5가 될 것
  - 두번째는 **입력층과 투사층 사이의 가중치 W는 V × M 행렬이며, 투사층에서 출력층사이의 가중치 W'는 M × V 행렬이라는 점**
    - V는 단어 집합의 크기 (가중치 W는 7 × 5 행렬), (W'은 5 × 7 행렬) 서로 다른 행렬
    - 인공 신경망의 훈련 전에 이 가중치 행렬 W와 W'는 대게 굉장히 작은 랜덤 값을 가지게 됨
    - CBOW는 주변 단어로 중심 단어를 더 정확히 맞추기 위해 계속해서 이 W와 W'를 학습해가는 구조입니다



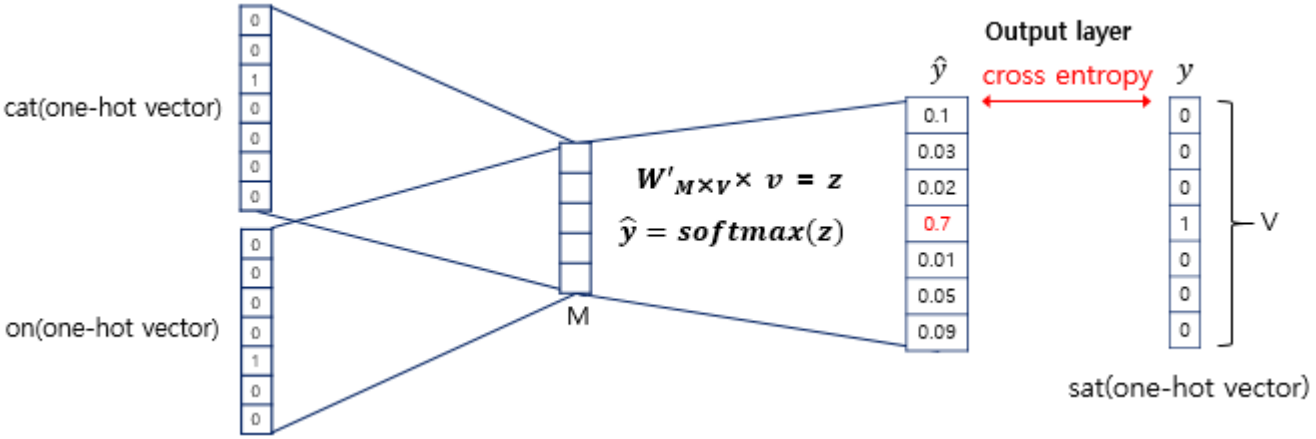
CBOW

- 각 주변 단어의 원-핫 벡터에 대해서 가중치 W가 곱해서 생겨진 결과 벡터들은 투사층에서 만나 이 벡터들의 평균인 벡터를 구함
- 윈도우 크기 n=2라면, 입력 벡터의 총 개수는 2n이므로 중간 단어를 예측하기 위해서는 총 4개가 입력 벡터로 사용됨
- 평균을 구할 때는 4개의 결과 벡터에 대해서 평균을 구하게 됨



CBOW

- 구해진 평균 벡터는 두번째 가중치 행렬 W'와 곱해짐
- 곱셈의 결과로는 원-핫 벡터들과 차원이 V로 동일한 벡터가 나옴
- 이 벡터에 CBOW는 소프트맥스(softmax) 함수를 취하는데, 소프트맥스 함수로 인한 출력값은 0과 1사이의 실수로, 각 원소의 총 합은 1이 되는 상태로 바뀜
  - 스코어 벡터의 가장 높은 값을 갖는 인덱스의 위치가 중심 단어일 확률 나타냄
  - 이 스코어 벡터는 우리가 실제로 값을 알고있는 벡터인 중심 단어 원-핫 벡터의 값에 가까워져야 함
  - CBOW는 손실 함수(loss function)로 cross-entropy 함수를 사용함



Mol2Vec

- NLP의 Word2Vec모델을 활용하여 Molecular 대량의 DB에서 가져온 구조 정보(ZINCv15와 ChEMBLv23의 약 10million의 compound)를 사전 학습하여 pretrained embedding vector를 생성한 연구(FP의 Sparse vector 문제 해결)

https://pubs.acs.org/doi/10.1021/acs.jcim.7b00616

RETURN TO ISSUE | < PREV ARTICLE NEXT >

Mol2vec: Unsupervised Machine Learning Approach with Chemical Intuition

Sabrina Jaeger, Simone Fulle, and Samo Turk

View Author Information

Cite this: *J. Chem. Inf. Model.* 2018, 58, 1, 27−35
Publication Date: December 22, 2017
https://doi.org/10.1021/acs.jcim.7b00616

Article Views 5156 | Altmetric 36 | Citations 52
LEARN ABOUT THESE METRICS

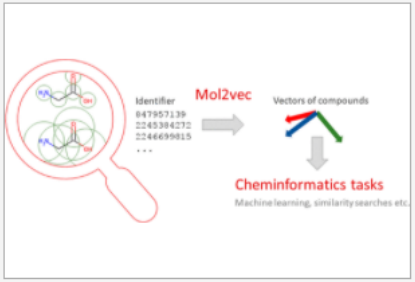
Share Add to Export

Read Online | PDF (2 MB) | Supporting Info (1) »

SUBJECTS: Embedding, Peptides and proteins, Molecular modeling, Monomers, Molecules

Journal of Chemical Information and Modeling

Abstract



Inspired by natural language processing techniques, we here introduce Mol2vec, which is an unsupervised machine learning approach to learn vector representations of molecular substructures. Like the Word2vec models, where vectors of closely related words are in close proximity in the vector space, Mol2vec learns vector representations of molecular substructures that point in similar directions for chemically related substructures. Compounds can finally be encoded as vectors by summing the vectors of the individual substructures and, for instance, be fed into supervised machine learning approaches to predict compound properties. The underlying substructure vector embeddings are obtained by training an unsupervised machine learning approach on a so-called corpus of compounds that consists of all available chemical matter. The resulting Mol2vec model is pretrained once, yields dense vector representations, and overcomes drawbacks of common compound feature representations such as sparseness and bit collisions. The prediction capabilities are demonstrated on several compound property and bioactivity data sets and compared with results obtained for Morgan fingerprints as a reference compound representation. Mol2vec can be easily combined with ProtVec, which employs the same Word2vec concept on protein sequences, resulting in a proteochemometric approach that is alignment-independent and thus can also be easily used for proteins with low sequence similarities.

Mol2Vec

- Vector 생성 방법
  - RDkit을 이용하여 SMILES 문자열 데이터를 radius=1인 Morgan FP로 substructure 추출하고 생성된 데이터를 python의 word2vec가 구현되어 있는 라이브러리인 gensim을 활용하여 CBOW(w=5& 10) 및 Skip-gram(w=10 & 20), Dimension(100D& 300D)로 학습하여 최상의 모델인 Skip-gram, w=10, 300D의 벡터를 추출하였음
- Embedding Vector 성능 검증
  - 회귀 및 분류 task에 모두 적용해보았으며, Random Forest (RF), Gradient Boosted Machines (GBM) 및 Deep Neural Networks (DNN)를 사용하여 모델을 생성하여 회귀모델은 MSE, R2을 metric으로 분류 모델은 auc, sensitivity를 사용하여 검증
  - 검증에는 ESOL, Ames, Tox21, Kinase Dataset을 사용하였음
  - ESOL: predict aqueous solubility of 1,144 compounds
  - Ames Mutagenicity Dataset: 3,481 mutagens + 2,990 non-mutagens
  - Tox21 Dataset: 12 targets associated with human toxicity, 8,192 compounds
  - Kinase Dataset: IC50, Kd, Ki ChEMBL bioassays with confidence ≥ 8, converted to pIC50 values, thresholder at 6.3
- 결과
  - Mol2vec-GBM의 경우 **R^2=0.86의** 결과를 얻었으며, Morgan FP-GBM 모델의 경우 0.66
  - Morgan Fingerprint의 기본 크기인 2048bit보다 작은 크기의 300D으로 학습 시 메모리를 절약할 수 있으며, 학습 속도 또한 빠르다는 결론을 얻음