자연어 처리의 모든 것 1. 자연어 처리의 시작

자연어 처리 활용 분야와 트렌드

자연어 처리 : 단어,문장, 문단을 이해하는 NLU 과정 / 자연어 생성 NLG

- 랭귀지 모델링, 기계번역, 컨슈천 앤서링..

<자연어를 다루는 분야>

1. 자연어 처리 NLP

- ACL, EMNLP, NAACL 라는 학회에서 주로 발표

◆ Low level parsing

• **tokenization** : 문장을 이루는 단어를 정보 단위(토큰)로 생각하여 단어 단위로 쪼개기

- 문장은 토큰의 sequence
- 어미가 변해도 같은 단어임을 컴퓨터가 인지해야함

• **stemming** : 단어의 어근 추출

◆ Word and phrase level

• **named entity recognition (NER)** : 여러 단어로 이루어진 고유 명사 인식

• **part of speech tagging (POS)** : 단어가 문장 내 성분이 무엇인지 알아내는 태스크

◆ Sentence level

• 감정분석 : 긍정어조, 부정어조 분류

• 기계번역(machine translation)

◆ Multi-sentence and paragraph level

• entailment prediction

• 논리적인 내포, 모순관계 예측

• question answering : 답에 해당하는 정보를 검색 결과로 나타내도록 하는 기술

- 질문의 키워드가 포함된 문서를 검색하고 독해를 통해 주어진 질문에 대한 정답을 제시

• dialog system : 대화를 수행할 수 있는 기술

• summarization : 문서를 자동으로 한줄 요약하는 기술

2. 텍스트 마이닝 분야

- WSDM,CIKM,ICWSM 주요 학회

EX) 특정 키워드의 빈도수를 시간순으로 트렌드 분석

- 서로 다르지만 의미가 비슷한 키워드를 그루핑해서 분석
- 팝핑 모델링 / **문서 군집화 기술** : 자동으로 분석
- ex) 토픽 모델링
- 사회과학과도 밀접한 연관

  ex) 혼밥 키워드 증가

3. 정보 검색

- 검색 기술을 연구하는 분야
- 검색 기술은 성숙한 상태 => 상대적으로 느린 분야
- **추천 시스템** : 자동화하여 선제적으로 제공 / 적극적인 검색 시스템
  - 상업에서도 큰 영향 (광고, 상품 추천)

<NLP 자연어 처리 분야 최근 발전 과정>

- 컴퓨터 비전 (영상처리분야)와 딥러닝이 가장 많이 적용되는 분야
- 숫자로 이루어진 데이터를 입력 출력에 필요로 하는데, 주어진 텍스트 데이터를 **단어 단위로 분리**하고 특정 차원을 가진 **벡터로 표현하는 과정**을 가짐

  ⇨ **word embedding**
- 워드를 벡터로 나타내면 워드 벡터가 순서 정보 포함한 하나의 sequence로 볼 수 있음
- RNN 모델이 자연어 처리의 핵심 모델
- LSTM & LSTM을 단순화한 GRU 모델
- 구글에서 self attention 모듈로 대체하는 **transformer 모델**이 나옴

- 기계번역에 사용
- 구글 translate를 가능하게한 기술 (딥러닝을 기반으로 한 번역기술)
- 영상처리, 신약개발, 시계열 데이터 등에도 사용
- 이전에는 각각의 태스크에 특화한 모델이 따로 있었는데,

  모델을 키우고 **자가지도 학습**을 통해 모델 학습하면 큰 구조의 변화없이 전이 학습의 형태로 적용하면 더 좋은 성능
- **자가지도 학습** : 입력 중 일부 단어를 가리고 앞뒤 문맥을 보고 그 단어를 맞추게 하는 학습
- 사전학습의 예시 : GPT : 범용 인공지능 기술
- but 대규모의 데이터 필요

<기존의 자연어 처리 기법>

Bag-Of-Words (단어 가방 모형)

- 텍스트 데이터셋에서 유니크한 워드를 모아서 사전을 만들기

  - 중복된 단어 제거하기

  - 각 단어를 범주형 변수로 간주하고, 원핫 벡터로 나타내기

    - 가능한 워드가 8개이면 차원이 8인 벡터공간을 설정하고

    8개의 축을 사전에 등록된 단어에 매핑해서 해당 값을 1, 나머지는 0으로 할당하여 8 차원의 벡터 나타내기

    원핫 벡터 : 워드임배딩과 대비

    - 단어쌍의 유클리드 거리가 루트2
    - 내적값, 유사도 = 0 : 단어의 의미와 상관없이 동일한 관계를 가짐

  ⇨ 워드들의 원핫벡터를 더한 벡터로 문서를 나타냄 : bag of words (단어 가방 모형)

- 특정 문장에서 나타난 워드를 그에 맞는 가방에 넣은 후 차원에 해당하는 가방에 들어간 워드의 수를 세서 벡터로 나타냄

Naive Bayes Classifier for Document Classification

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}}\ P(c|d)$$ MAP is "maximum a posteriori" = most likely class

$$= \underset{c \in C}{\operatorname{argmax}}\ \frac{P(d|c)P(c)}{P(d)}$$ Bayes Rule

by 베이즈 정리

$$= \underset{c \in C}{\operatorname{argmax}}\ P(d|c)P(c)$$ Dropping the denominator

- **maximum a posteriori**
- p(d) : 특정한 문서 d가 뽑힐 확률 (상수) => argmax에서 무시 가능
- P(d|c) : c가 고정이 되었을 때 문서 d가 나타날 확률

  d는 w1 ... wn의 동시 사건

특정 클래스가 고정되었을 때 각 워드가 나올 확률을 추정하여 나이브 베이즈 분류기에 필요한 파라미터 추정

Data	Doc(d)	Document (words, w)	Class
Training	1	Image recognition used convolutional neural networks	CV
	2	Transformers can be used for image classification task	CV
	3	Language modeling uses transformer	NLP
	4	Document classification task is language task	NLP
Test	5	Classification task uses transformer	?

P(c_cv)=2/4

총 단어 개수 (14개) 중 task 라는 단어는 1/14의 확률

- **Example**
  - For each word w_i, we can calculate conditional probability for class c
    - $P(w_k|c_i) = \frac{n_k}{n}$, where n_k is occurrences of w_k in documents of topic c

Word	Prob	Word	Prob
$P(w_{\text{"classification"}}\|c_{CV})$	$\frac{1}{14}$	$P(w_{\text{"classification"}}\|c_{NLP})$	$\frac{1}{10}$
$P(w_{\text{"task"}}\|c_{CV})$	$\frac{1}{14}$	$P(w_{\text{"task"}}\|c_{NLP})$	$\frac{2}{10}$
$P(w_{\text{"uses"}}\|c_{CV})$	$\frac{1}{14}$	$P(w_{\text{"uses"}}\|c_{NLP})$	$\frac{1}{10}$
$P(w_{\text{"transformer"}}\|c_{CV})$	$\frac{1}{14}$	$P(w_{\text{"transformer"}}\|c_{NLP})$	$\frac{1}{10}$

"Classification task uses transformer"

We calculate the conditional probability of the document for each class

We can choose a class that has the highest probability for the document

$P(c_{CV}|d_5) = P(c_{CV}) \prod_{w \in W} P(w|c_{CV}) = \frac{1}{2} \times \frac{1}{14} \times \frac{1}{14} \times \frac{1}{14} \times \frac{1}{14}$

$P(c_{NLP}|d_5) = P(c_{NLP}) \prod_{w \in W} P(w|c_{NLP}) = \frac{1}{2} \times \frac{1}{10} \times \frac{2}{10} \times \frac{1}{10} \times \frac{1}{10}$

Word	Prob	Word	Prob
$P(w_{\text{"classification"}}\|c_{CV})$	$\frac{1}{14}$	$P(w_{\text{"classification"}}\|c_{NLP})$	$\frac{1}{10}$
$P(w_{\text{"task"}}\|c_{CV})$	$\frac{1}{14}$	$P(w_{\text{"task"}}\|c_{NLP})$	$\frac{2}{10}$
$P(w_{\text{"uses"}}\|c_{CV})$	$\frac{1}{14}$	$P(w_{\text{"uses"}}\|c_{NLP})$	$\frac{1}{10}$

- 클래스의 확률 x 각각의 단어가 가진 확률
- 가장 큰 확률값을 가지는 클래스를 분류 예측값의 결과
- but 학습데이터 내에 특정 단어가 없으면 확률 0 , 그 단어가 포함될 문장이 주어지면 클래스 확률이 0 으로 나타남

  ⇨ regularization 기법들이 추가 되어 활용

<Word Embedding – Word2Vec>

word embedding : 각 단어를 특정 차원을 가진 공간의 점이나 좌표의 벡터로 전환하는 기술

★ 비슷한 의미를 가지는 단어가 비슷한 좌표의 점으로 매핑되도록 하는 것

감성분석 모델 : 긍정 감정의 단어들 / 부정 감정의 단어들 을 각각 비슷한 위치에 매핑

Word2Vec

- 문장 내에서 비슷한 위치에 등장하는 단어는 유사한 의미를 가질 것이라고 생각
- 주변에 등장하는 단어들을 통해 중심 단어의 의미가 표현될 수 있다는 것으로 추정

★ 가까운 단어가 의미적으로 관련성이 높은 것이라고 생각

EX) cat 단어를 입력으로 주고 다음 단어를 예측하는 형태로 학습 진행

1. 주어진 학습 데이터를 워드별로 분리하는 토큰화

2. 사전을 구축

3. 사전의 사이즈만큼의 차원을 가지는 원핫 벡터 생성

4. sliding window 기법으로 앞뒤의 단어 쌍을 구성

   EX) 윈도우의 사이즈가 3인경우 (앞뒤의 단어 하나씩 보는 경우)

   I study math 에서는 (I, study) (study, I) (study, math)

5.

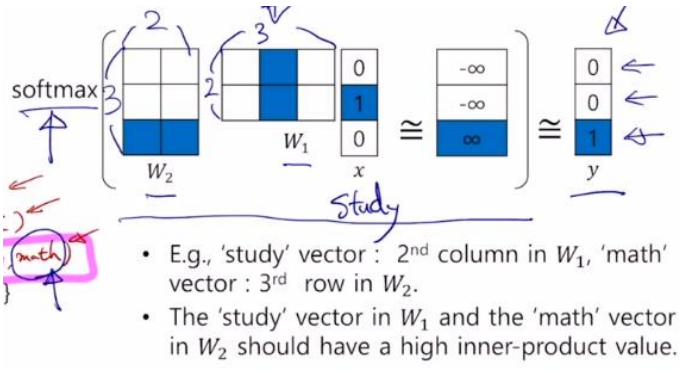
입출력 단어쌍에 대해 예측 task를 수행하는 2layer

입출력 레이어 수는 3개

가운데의 hidden layer의 노드 수 : 사용자가 정하는 하이퍼파라미터

- 좌표공간의 차원 수와 동일하게 설정

EX) 여기서는 입력노드는 3차원, 출력노드는 3차원, 히든레이어는 2차원



1. 선형변환 행렬을 곱함

★ embedding layer : 행렬곱을 수행하지 않고 원핫벡터의 1에 해당하는 인덱스의 칼럼벡터를 뽑아오는 형태

- 첫번째 row vector와 원핫벡터를 곱할 때, 원핫벡터 자리에 해당하는 칼럼벡터를 W1에서 원하는 값을 추출하는 연산 진행

- 두번째 row vector와 곱할 때도 동일하게 연산 진행

=> 입력 워드에 대한 2차원 벡터를 뽑아온 것과 같음!

2. 2차원 벡터 => 3차원 벡터 : 행렬을 곱해서 변환 (3x2)

W2에서 각각의 row vector의 내적을 통해 값들 나옴

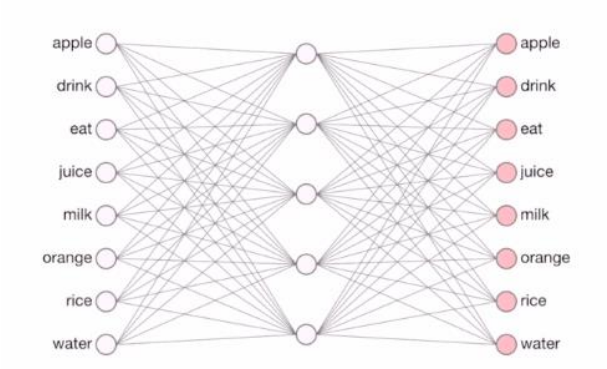
3. softmax 함수를 통해 확률 값으로 변환

- 로직값의 측면

- (math라는 단어)타겟 레이블에 해당하는 원핫벡터

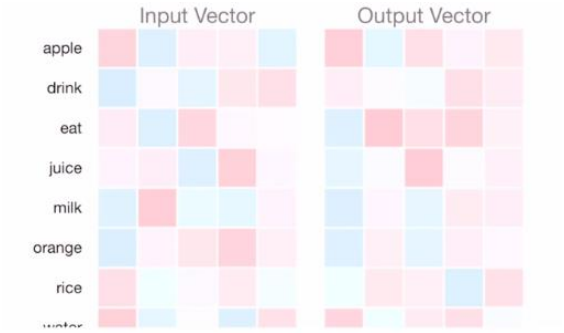
- 가장 거리가 가까워지도록하는 softmax loss 를 적용

- 주어진 입력 단어에 W1, 주어진 출력 단어에 W2의 내적의 벡터에 따른 유사도가 최대한 커지도록, 출력 단어가아닌 W2의 벡터들의 내적의 기반한 유사도는 최대한 작게 만드는 것
- W1, W2의 파라미터를 이렇게 조정함



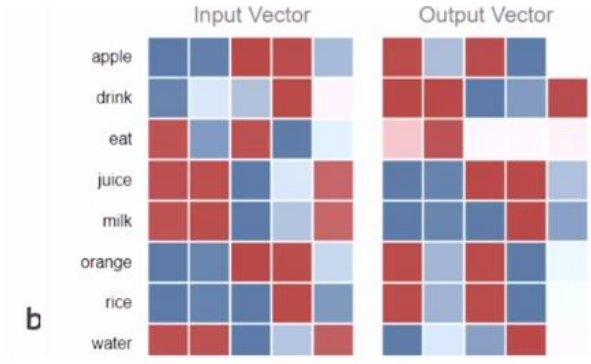
입력 출력 =8, 히든레이어 노드 수 =5

- W1은 행렬의 transpose를 했을 때 W2와 같도록



붉은색은 양수

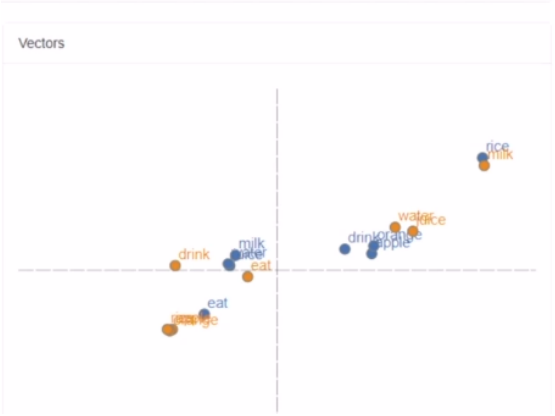
- W1에서 칼럼벡터 뽑아오는 형식으로 eat와 연결된 가중치가 eat의 1과 곱해셔서 히든 레이어의 벡터를 결정
- 두번째 layer에서 각 output노드과 연결된 가중치를 고려하여 출력벡터 W2와 내적 진행
- 출력 노드의 값 (W2의apple에 해당하는 벡터)과 W1의 eat에 해당하는 벡터의 내적값이 최대한 커지도록 학습 진행



반복학습 후

- drink, juice는 유사한 벡터 표현형을 가짐 => 내적값이 크다
- milk, water / apple, orange : 벡터들과 유사한 패턴을 가짐

- 입력과 출력 단어버전의 워드임베딩 결과 중 어느 것을 최종 워드 임베딩으로 사용할지는 상관 없음
- W1 입력 을 보통 최종 결과로 사용함



W1, W2 를 PCA를 통해 2차원으로 차원 축소를 한 이후 scatterplot으로 시각화한 예시

- king – queen 와 woman - man 과 유사하게 일관된 벡터로 유지 => 효과적으로 학습

Word Intrusion Detection : 나머지 단어와 의미가 가장 상이한 단어를 찾아내는 태스크

- Word2Vec의 embedding 기술을 이용
- 단어간의 유클리드 거리를 계산하고 평균값=> 한단어가 다른 단어와 가지는 평균 거리
- 평균거리가 가장 큰 단어 => 의미가 가장 상이한 단어

<Application of Word2Vec>

기계번역 : 서로 다른 언어간의 같은 의미를 가진 워드들의 임베딩 벡터가 쉽게 정합되도록 하여 성능을 높임

감정분석 : 긍부정의 의미를 용이하게 분류

이미지 캡셔닝 : 이비지 상황을 잘 이해하고 설명글을 자연어형태로 생성

<Word Embedding – GloVe>

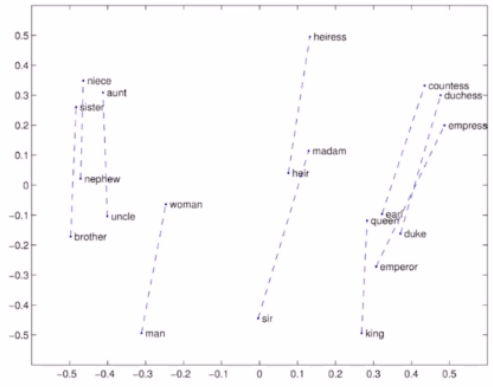
★ 각 입력 출력 쌍에 대해 두 단어가 한 윈도우 내에서 몇번 동시에 등장했는지 사전에 계산하여 로그를 취하고,

입력 워드의 임베딩 벡터 ui, vj의 내적값과 최대한 가까워질 수 있도록 하는 loss function 이용

- 특정 단어쌍이 자주 등장한 경우에 데이터 아이템이 여러 번 학습되면서 내적값이 커지도록 하는 학습 Word2Vec
- 미리 계산하고 로그를 취해서 학습 진행 => 중복되는 계산을 줄여줌

계산이 Word2Vec 보다 빠르고, 적은데이터에도 적용 가능

- 추천 시스템에 사용되는 low rank matrix factorization의 태스크로도 이해 가능
- Word2Vec과 GloVe는 텍스트 데이터에 기반하여 동일한 역할 수행하는 알고리즘, 성능도 비슷



PCA로 차원축소하여 나타낸 그래프

- 형용사의 원형과 비교급 최상급 단어들에 대해서도 일정한 크기와 방향을 가지는 벡터

wikipedia 2014, gigaword 5 : 6B token만큼 학습된 모델

- 50d, 100d ... : GloVe에서 각각 설정한 타겟 차원
- 입력, 출력워드의 차원 결정