18주차_자연어 처리의 시작

링크 https://velog.io/@pehye89/Euron-18주차-자연어-처리의-시작

1 more property

Getting started with NLP - A general Intro

Explore and run machine learning code with Kaggle Notebooks | Using data from Natural Language Processing with Disaster Tweets

https://www.kaggle.com/code/parulpandey/getting-started-with-nlp-a-ge...

💯 출석퀴즈

자연어처리 개요

자연어 처리 활용 분야와 트렌드

- 단어나 문장, 길 문단과 같은 글을 이해하는 natural language understanding (NLU)
- 이해하는 것이 아닌, 이런 글을 생성하는 natural language generation (NLG)
- 인공지능 및 딥러닝 기술이 가장 활발하게 이루어지는 분야 중 하나

1️⃣ Natural Language Processing NLP

- 딥러닝 기술의 발전을 선도하는 핵심적인 기술
- ACL, EMNLP, NAACL 학회 등에서 발표됨

1. Low Level parsing
    - tokenization : 문장을 단어 단위로 쪼개는 것
    - stemming : 같은 단어라도 여러 어휘의 변형들이 존재한다는 것을 컴퓨터가 이해하게 하는 것. 즉, 단어의 어근을 추출하는 것
2. Word and Phrase Level
    - Named Entity Recognition (NER)
    - POS tagging : 품사나 성분을 알아내는 태깅
3. Sentence Level
    - Sentiment Analysis : 감정분석
    - Machine Translation : 문장을 전체적으로 이해하고 각 단어별로 적절한 단어를 고르고 적절한 어순을 선택하는 것
4. Multi-sentence and Paragraph Level
    - Entailment Prediction : 여러 문장들을 보고 참/거짓을 예측하는 것. 즉, 모순이 존재하는지 확인하는 것
    - Question/Answer : 질문의 단어 하나하나의 키워드들을 나열하는 것 뿐만 아니라, 질문을 정확하게 파악해서 정확하게 제시해줄 수 있게 하는 것
    - Dialog system
    - Summarization

2️⃣ Text Mining

- KDD, The WebConf, WDSM, CIKM, ICKWSM 학회 등에서 발표됨
- 빅데이터 분석과 관련되어 있다.
- 몇백만건의 문서 데이터를 모아서 트렌드를 분석하여, 예를 들어, 특정 유명인의 이미지가 어떤 사건을 기점으로 어떻게 변화하는지 등을 분석할 수 있다.
- 이 과정에서 비슷한 의미를 갖는 키워드들을 모아 grouping 할 필요가 있고, 그러기에는 clusting을 위해 topic modeling을 활용한다.
- 또 사회과학과도 깊은 관련이 있다.
    - 예를 들면 트위터 데이터를 분석해서 특정 현상이 어떻게 시작되고 유행되는지를 분석할 수 있다.

3️⃣ Information Retrieval

- 검색엔진에서 사용되는 검색기술을 연구하는 분야
- 하지만 현재 검색기술은 어느정도 성숙해졌다고 할 수 있다. 즉, 성장 속도가 더디다.
- 현재 가장 많이 활용되고 있는 분야는 추천 시스템이 있다.

> 💡 추천 시스템에서 너무 비슷한 것만 보여주는 것이 아닌, 일부러 반대되는 내용도 같이 나오게 하면 좋을 것 같다. 필터버블을 피할 수 있는 좋은 방법일 것 같다.

자연어처리의 최근 발전 과정

- 컴퓨터 비전과 함께 인공지능과 딥러닝이 가장 활발하게 적용되고 있다
- 일반적으로 데이터가 숫자로 학습되기 때문에, 자연어처리의 경우 단어들을 벡터화할 필요가 있고, 이 과정을 **워드 임베딩** word embedding이라고 한다.
- 또한 언어 특성상 같은 단어가 문장 구조에 따라 다른 단어로 사용될 수 있기 때문에, 이런 벡터들의 시퀀스가 중요하게 된다. 이런 벡터들의 시퀀스를 학습하기 위해 RNN 모델들이 (LSTM, GRU) 자리잡게 된다.
- 이후 Transformer 모델이 기존의 RNN 모델이 할 수 있는 모든 것을 self-attention이라는 모듈로 대체할 수 있게 되었다. 이 모델이 이제 자연어 처리 분야에서 기본으로 활용되고 있다.
- 자가학습 가능한 모델을 만들기 위해서는 엄청난 데이터와 비용이 들기 때문에 일부 OpenAI, Google과 같은 큰 기업들이 이 분야를 이끌어나가고 있다.

기존 자연어 처리 기법 : Bag-of-Words

단어 및 문서를 숫자로 나타내는 가장 기본적인 방법

- 한 문장이 있다고 해보자
- 우선 여기에서 unique words를 등록한다. 즉, 중복된 단어를 제외한 단어들을 사전에 정리한다.
- 각 단어들을 one-hot encoding하여 categorical variable을 one-hot vector로 표현해준다.
- 이러한 좌표공간 사이에서 모든 단어들의 거리가 유클리디안 거리인 $\sqrt{2}$가 된다.
- 또한 각 단어의 유사도 cosine simillarity는 0이된다. 즉, 단어의 실제 관계와 상관없이, 관계가 없는 것으로 정해진다.

Naive Bayes Classifier

$$P(d|c)P(c) = P(w_1, w_2, \dots, w_n|c) \rightarrow P(c) \prod_{w_i \in W} P(w_i|c)$$

- $P(d|c)$: 특정 카테고리 c가 고정이 되었을 때, 문서 d가 나타날 확률
- 문서 d는 w_1 부터 w_n까지 동시에 나타나는 동시 사건
- 각 단어가 나타날 확률이, c가 고정되어있을 때 독립이라고 가정할 수 있다면, 곱한 형태로 나타낼 수 있다.
- 또한 naive bayes classifier는 클래스가 2개 이상, 즉 3개일 때도 사용할 수 있다.

- 만약 학습 데이터 내 특정 단어가 나타나지 않을 경우 0이 될 것이고, 그 단어가 포함된 문장이 나타나면, 다른 단어들이 해당 클래스와 밀접한 관련이 있을 경우에도 0이 된다. 이 단점을 보안하게 위해 다른 regularization 방법론들을 추가해서 사용한다.
- 또한 이 파라미터 추정 과정은 MLE를 통해 도출이 된다.

아래 예시를 보자

Data	Doc(d)	Document (words, w)	Class (c)
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	?

여기서 CV일 경우와 NLP일 경우의 확률을 계산한다면 아래와 같다.

- $P(c_{cv}) = \frac{2}{4} = \frac{1}{2}$
- $P(c_{nlp}) = \frac{2}{4} = \frac{1}{2}$

다음으로는 클래스가 둘 중에 어느 한 클래스로 고정되었을 때 각 단어가 나타날 확률을 추정해본다.

- 왼쪽의 경우는, 주어진 단어가 CV일 경우를 가정하고 계산했을 때이다.
- 여기서 14는 CV의 단어 개수, 즉 "Image recognition use ..."와 "Transformers can be used for...", 이 두 문장의 단어 개수이다.
- 여기서 task라는 단어가 1번만 등장했기 때문에 $\frac{1}{14}$가 되는 것이다.

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

- $P(c_{cv}|d_5)P(c) = P(c_{cv})\prod_{w\in W} P(w|c_{cv}) = \frac{1}{2} \times \frac{1}{14} \times \frac{1}{14} \times \frac{1}{14} \times \frac{1}{14}$
- $P(c_{NLP}|d_5)P(c) = P(c_{NLP})\prod_{w\in W} P(w|c_{NLP}) = \frac{1}{2} \times \frac{1}{10} \times \frac{2}{10} \times \frac{1}{10} \times \frac{1}{10}$

자연어 처리와 벡터

워드임베딩이란?

- 자연어가 단어들을 정보의 기본 단위로 해서, 이 정보들의 시퀀스를 볼 때에, 각 단어를 좌표공간에 최적의 벡터로 표현하는 기법이다.
- 예를 들면, kitty 와 cat은 비슷한 좌표를 갖게 될 것이고, hamburger는 더 먼 좌표를 갖게 될 것이다.

> 💡 즉, 위드 임베딩은 비슷한 의미를 갖는 단어가, 좌표공간에 비슷한 위치에 입력되게 함으로써, 단어들의 의미의 유사도를 잘 반영한 벡터 표현을 자연어처리 모델이 학습하도록 전달해주는 것이다.

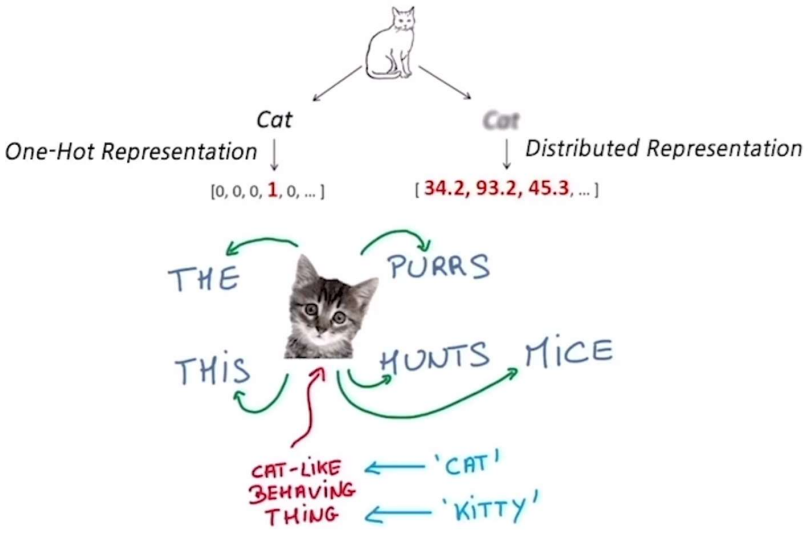
또한 감정분석을 할 때에, 아래 단어들이 비슷한 위치에 맵핑이 될 것이다.

- "기쁨", "환희"는 긍정적인 감정을 나타내는 단어들

- "분노", "증오"는 부정적인 감정을 나타내는 단어들

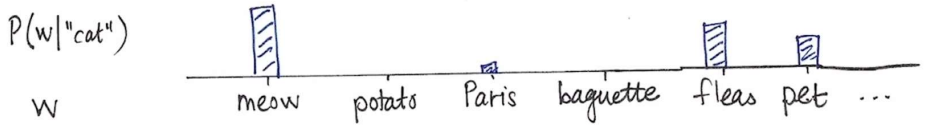
워드임베딩 : Word2Vec

- 이 알고리즘은 같은 문장에서 나타난 인접한 단어들 간에 의미가 비슷할 것이라는 가정을 한다.

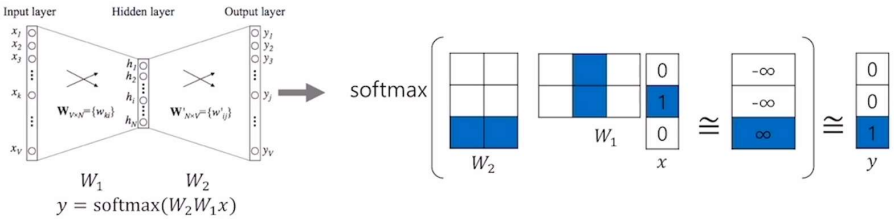


Distributed Representations of Words and Phrases and their Compositionality, NeurIPS'13

- 만약 "This cat purrs"와 "This cat hunts mice"라는 문장 두 개에서, 여기에서 나오는 단어들이 모두 cat과 연관되어 있다고 한다.



- 따라서, 이 알고리즘에서는 어떤 한 단어가 주변에 나타나는 단어들을 통해 그 의미를 알게될 수 있다는 아이디어를 통해 주변에 나타나는 단어들의 확률분포를 계산하게 된다.
- 그렇기에 위 사진과 같이, cat과 함께 나타나는 단어들이 얼마나 자주 나오는지에 대한 확률분포를 구한다.



W_1 W_2

$y = \text{softmax}(W_2 W_1 x)$

- **Sentence** : "I study math."
- **Vocabulary**: {"I", "study" "math"}
- **Input**: "study" [0, 1, 0]
- **Output**: "math" [0, 0, 1]
- Columns of W_1 and rows of W_2 represent each word
- E.g., 'study' vector : 2nd column in W_1, 'math' vector : 3rd row in W_2.
- The 'study' vector in W_1 and the 'math' vector in W_2 should have a high inner-product value.

- 이를 위해 우선 워드를 토크나이징 Tokenizing 해준 후, 유니크한 단어만 모아서 사전 Vocabulary을 만든다. 이후 문장에서 중심단어를 위주로 학습 데이터를 구축해준다.
- 예를 들어 "I study math"의 중심단어가 study 라고 한다면, **Sliding 기법**을 통해 (I, study), (study, I) , (study, math)와 같은 단어쌍을 학습 데이터로 구축한다.
- 이후 윈도우를 옮겨가며 중심단어를 바꿔가며 학습 데이터를 더 구축한다.

- 이렇게 단어쌍이 구축되고 나면, 두 개의 레이어를 갖는 신경망이 생겨나게 된다.
- 여기서 각 단어가 3차원의 원핫 벡터로 나타나기 때문에 입출력의 노드수는 3이 될 것이다.

- 그리고 은닉층의 노드 수는 사용자가 정하는 하이퍼파라미터이다.
- 위 예제는 은닉층이 2차원 레이어가 된다.

$W_1 \times x$

- 또한 W_1과 x의 행렬곱을 구한다면, 원핫벡터에 해당하는 값만 불러오는 과정이라는 것을 알 수 있다.
- 이 과정은, W_1이 Vocabulary만큼의 column 갖고 있을 때, 그 해당하는 입력 단어에 대한 2차원의 column vector를 뽑아온 것으로 생각할 수 있다.

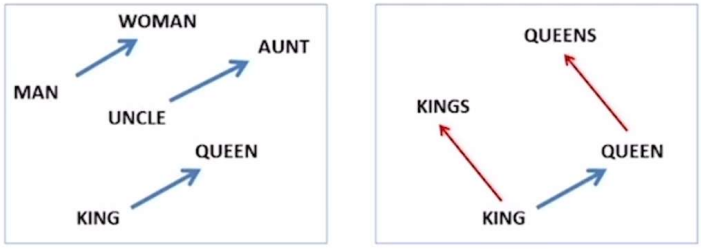
$W_2 \times x$

- 다음 W_2와의 곱을 본다면, 위 과정을 통해 이 2차원 벡터가 뽑혀왔을 때, 이 벡터와 W_2의 row vector와의 내적을 통해서 이 값들이 구해진다는 것을 알 수 있다.
- 또한 W_2에서 row vector의 사이즈는 vocabulary 만큼의 사이즈를 갖고 있다는 것을 알 수 있다.

결국, 이 W_1, W_2그리고 x가 곱해져서 나온 값들이 ground truth label, 즉 $[0, 0, 1]$과 유사한 값이 나오려면, softmax의 logit값의 측면으로 볼 때는, ground truth에 해당하는 부분의 값은 $+$ 그리고 그 외에 값을 $-$으로 설정했을 때 원하는 값이 나올 수 있다.

즉, 가장 성능이 좋게 되려면 아래와 같이 학습을 진행해야한다.

- 주워진 입력 단어의 해당하는 W_1 상에서의 벡터와 주워진 출력 단어에 해당하는 W_2 상에서의 벡터간에 내적에 기반한 유사도가 최대한 커지도록 하는 것
- 또한 그 이외에 값들의 값이 최대한 작을 수 있게 하는 것



(Mikolov et al., NAACL HLT, 2013)

- 이렇게 워드 임베딩은 단어들간의 거리를 통해 이들의 유사도를 알 수 있게 했다.
- 위 예시를 보면 man-woman, uncle-aunt, king-queen간에 거리가 일정하다는 것으로 이들의 차이가 동일한 카테고리, 즉 성별이라는 것을 알 수 있다.
    - vec[queen]-vec[king] = vec[woman]-vec[man]

Word2Vec의 활용

Word2Vec은 그 자체로도 의미가 있지만, 다양한 테스크에서 사용되고 있다.

- Machine translation : 단어 유사도를 학습하여 번역 성능을 더 높여준다

- Sentiment analysis : 감정분석, 긍부정분류
- Image Captioning : 이미지의 특성을 추출해 문장으로 표현

워드임베딩 : GloVe

차이점

- 각 입력 및 출력 단어쌍에 대해, 사전에 미리 각 단어들의 동시 등장 빈도수를 계산하고, 이 단어쌍간의 내적값과 사전에 계산된 값에 로그값을 취해서 이들의 내적값이 최대한 가까워질 수 있도록 하는 loss function을 사용한다.
- 또한 Word2Vec은 모든 연산을 반복하지만, GloVe은 사전에 계산된 값을 ground truth으로 사용하기 때문에 이러한 중복되는 계산을 줄일 수 있다.
- 즉, 학습이 더 빠르게 동작하고 더 작은 데이터에서도 잘 작동한다.