IT 논문 토픽모델링과 키워드 클러스터링을 통한 트렌드 분석

I. 서론

데이터 분석을 통해 트렌드를 분석하고 나아가 미래를 예측하려는 시도는 빅 데이터 기반 연구가 활성화된 이후 경제, 정치 등 여러 분야에서 활발하게 이루어졌다. 하지만 구체적으로 활용하는 데이터와 그 분석 방법에 따라 그 결과는 달라지기 마련이다. 매우 빠르게 변화하는 IT 분야에서 현재 어떤 기술이 트렌드로 자리잡았는지 객관적으로 규정하는 것은 쉽지 않은 일이지만, 우리 ITREND팀은 컴퓨터 과학 분야의 논문들을 기반으로 어떤 Keyword가 오늘날 IT분야의 트렌드가 되고 있는지, 해당 Keyword들은 시대에 따라 어떤 양상으로 그 영향력이 변화하였는지 밝히고자 했다. ITREND의 이러한 시도는 두 가지 가정 하에서 이루어졌는데, 첫째, 학계에서 활발하게 연구되는 분야가 곧 IT업계의 트렌드가 된다는 것이고, 둘째, 논문들이 담고 있는 Keyword들의 빈도수가 해당 시기의 학계 흐름이 나타낸다는 것이다.

프로젝트 진행은 1) 데이터 크롤링 2) 데이터 클러스터링 및 토픽 모델링 3) 결과 시각화 및 트렌드 분석의 단계를 통해 이루어졌다.

II. 실험방법

1. 논문 크롤링

IEEE에서 논문을 크롤링하기로 정하였다. 크롤링할 논문마다 논문이 등재된 날짜(pubDate), 논문의 키워드들(keywords), 제목(title), 초록(abstract) 을 크롤링하기로 하였다.
처음에 URL 기반 크롤링으로 시도했다가 중간에 사이트가 개편되는 바람에 URL 기반 크롤링이 불가능하게 되었고, 그래서 셀레니움이랑 GraphQL으로도 시도해 보았다. 그 중 GraphQL을 통해서 시도해보았을 때 결과가 가장 안정적이고 좋았기에 GraphQL로 크롤링을 진행하였다.

가. URL 기반 크롤링

requests와 beautifulSoup를 기반으로 하여서 크롤링 하였다. url을 통해서 해당 사이트에 html 파일을 request 하고, 받은 html 파일을 파싱하고 분석하면서 진행하는 방식이다.
이 방법은 IEEE 사이트가 중간에 개편되기 전에는 효과적이었지만 사이트가 개편된 후 사이트가 스크립트 코드를 통해 부분부분 불러오는 방식으로 바뀌면서 크롤링이 불가능하게 되었다.

나. Selenium

selenium과 chromedriver를 기반으로 하여서 크롤링 하는 방식이다. 유저가 일반적으로 브라우저를 통해 서칭하는 것과 비슷한 방식으로 진행할 수 있기에 스크립트 코드로 바뀌었더라도 진행이 가능하다.

URL 기반 크롤링이 막히고 나서 바로 시도해보았던 방식으로, 확실한 방법이기는 하나 속도가 느렸고, 역시 selector를 지정해가며 서칭해나가야했지만, 사이트가 개편되고 나서 selector를 특정지어주기 어렵다는 문제도 있었기에 크롤링이 가능하기는 하지만 너무 복잡하고, 특이 케이스마다 어떻게 해결할지를 일일이 다 지정해줘야한다는 문제, 그리고 크롤링이 너무 불안정하다는 문제가 있었다.

다. GraphQL

GraphQL은 API에 사용되는 query language(질의어)이다. 클라이언트는 필요한 데이터의 구조를 지정할 수 있으며, 서버는 정확히 동일한 구조로 데이터를 반환해준다. GraphQL은 서버에서 등록해준

API를 이용하여 query문을 작성하여 보내고 원하는 결과값을 받아오기에 GraphQL을 이용하는 방식이 사이트에 구성만 되어있다면 가장 빠르고 효율적인 방법이다.
우리는 나중에 IEEE 사이트에 GraphQL 방식을 제공하는 API가 있다는 것을 알게 되었고, 이를 사용하여 원하는 값들을 크롤링할 수 있었다.

2. LDA

가. 전처리

자연어 처리에서 전처리는 매우 중요한 단계로, 수집한 데이터를 모델에 넣기전에 정제하는 과정이다. 데이터 전처리는 정제(Normalizaiont), 토큰화(Tokenization), 불용어제거(Stop and Elimination) 그리고 어간추출(Stemming) 순으로 진행된다.

전처리의 첫번째 과정으로 정제과정(Normalization)과 토큰화(Tokenization) 을 걸쳤다. 토큰화 작업에 방해가 되는 부분들을 줄이고자 토큰화 작업 수행하기 전에 앞서 정제 과정이 이루어졌다. 정제 과정으로 정규 표현식을 이용해 영문자가 아닌 문자는 공백으로 치환하고 모든 문자를 소문자로 통합하였다. 그 후, 정제된 자료를 가지고 띄어쓰기를 기준으로 단어 토큰화를 진행하였다.
두번째로, 갖고 있는 데이터 속에는 큰 의미가 없는 단어 역시 존재하기 때문에 이를 제거하는 작업이 필요하다. 여기서 큰 의미가 없는 단어란 I, my, me와 같이 문장 속에서 빈번하게 등장하지만 문장의 의미를 분석하는 데에 있어서 큰 도움이 되지 않는 단어들을 의미하며 이는 불용어라고 불린다. 이러한 단어는 nltk에 이미 정의된 불용어 패키지를 이용하여 쉽게 제거할 수 있다.
마지막으로 어간 추출(Stemming) 단계가 있다. 변화된 단어의 접미사나 어미를 제거하여 같은 의미를 가지는 형태소의 기본형을 찾는 방법이다. 우리는 NLTK의 SnowballStemmer을 이용하여 어간을 추출하였다.
위의 과정을 IEEE에서 크롤링한 초록에 적용하여 Dataframe 형식으로 만들었다. 다음 예시는 실제로 전처리한 과정 순서대로 직접 크롤링한 논문 초록 일부를 나열한 것이다.

전처리 과정 전
Re-use of patients' health records can provide tremendous benefits for clinical research.

정제
Re use of patients health records can provide tremendous benefits for clinical research

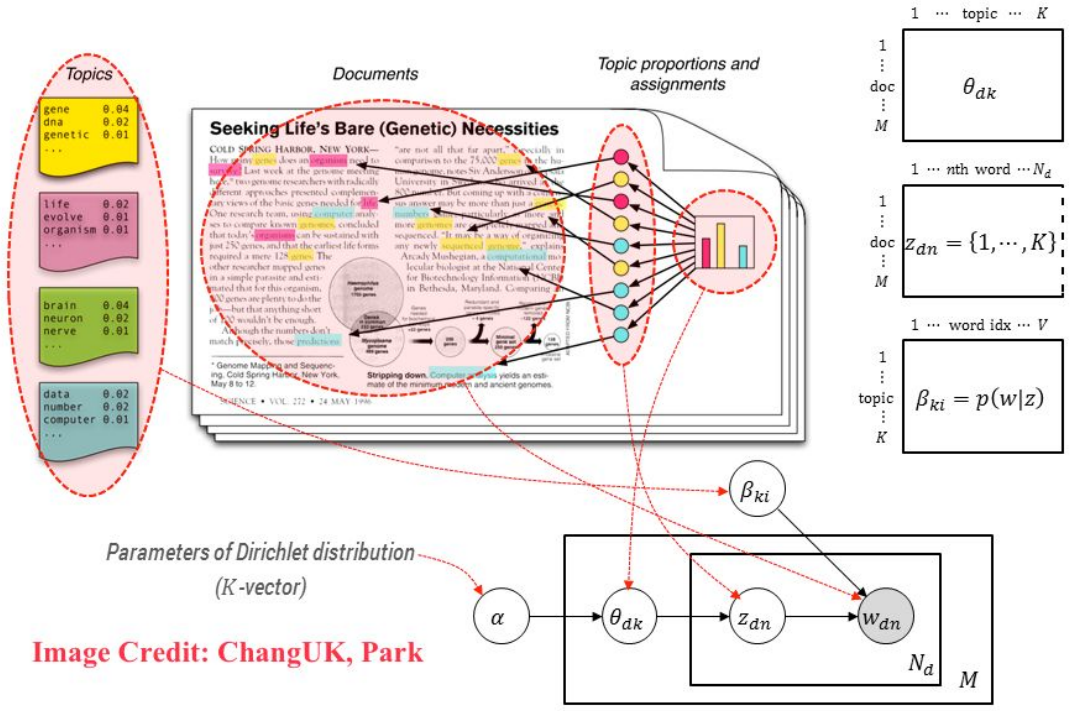
토큰화
['re', 'use', 'of', 'patients', 'health', 'records', 'can', 'provide', 'tremendous', 'benefits', 'for', 'clinical', 'research']

불용어 제거
['use', 'patients', 'health', 'records', 'provide', 'tremendous', 'benefits', 'clinical', 'research']

어간 추출
['use', 'patient', 'health', 'record', 'provid', 'tremend', 'benefit', 'clinic', 'research', 'yet', 'research']

나. LDA 모델 구축

전처리된 초록을 바탕으로 토픽을 추출하기 위하여, 토픽 모델링 기법 중 하나인 잠재디리클레할당(Latent Dirichlet Allocation, LDA) 모델을 사용하였다. LDA란 주어진 문서에 대해 각 문서에 어떤 주제들이 존재하는지를 서술하는 확률적 토픽 모델링 방법이다. 주어진 코퍼스에서의 토픽별 단어 분포도를 바탕으로 주어진 문서에서 발견된 단어 분포를 분석하여 문서가 어떤 주제를 다루고 있을지 예측한다. 모델의 개요도는 다음과 같다.



전처리된 초록을 **sent_to_words** 함수에 입력값으로 주면 문장 별 토큰이 리스트로 반환된다. 'machine learning', 'supervised learning'과 같은 통상적으로 함께 쓰이지는 않지만 본 프로젝트에서 중요시 여겨지는 연어(collocation)를 결과에 반영하기 위하여 반환된 리스트에 bigram을 적용한다. 자주 함께 등장하는 단어라면 'machine_learning'과 같이 반환하여 'machine' 또는 'learning'이라는 단어와는 다른 단어 단위로 사용된다.

```
bigram = gensim.models.Phrases(data_words, min_count=5, threshold=20)
bigram_mod = gensim.models.phrases.Phraser(bigram)
data_words_bigrams = [bigram_mod[word] for word in data_words]
```

bigram을 적용한 초록 토큰을 이용하여 Dictionary를 구축한다. id2word는 각각의 word에 id를 부여하는 함수의 출력값이다. Dictionary는 id와 word를 맵핑하기 위해 필요하며 gensim의 **corpora.Dictionary** 메소드를 이용해서 만든다. 앞의 과정은 corpus를 구축하기 위한 준비 과정으로, 각 문장의 토큰들은 **doc2bow**를 통해 튜플 (토큰, 토큰의 빈도수)의 리스트를 반환하고 id2word를 이용해 토큰을 id에 맵핑한다. 결과물로 LDA 모델 구축에 필요한 id2word와 corpus를 얻었다.

```
id2word = corpora.Dictionary(data_words_bigrams)
texts = data_words_bigrams
corpus = [id2word.doc2bow(text) for text in texts]
```

LDA 모델을 구축하기 위해 gensim의 models.ldamodel.LdaModel을 이용하며 다양한 파라미터를 조절할 수 있다. corpus와 id2word는 앞서 구축한 것을 사용하고, 다른 변수들은 모델의 적합도와 프로젝트의 목적을 적절히 반영하는지를 따져 조정하였다. 이에 대한 설명은 iii-2에 자세히 나타나 있다.

```
lda_model = gensim.models.ldamodel.LdaModel(corpus=corpus,
id2word=id2word,
num_topics=20,
random_state=100,
```

```
update_every=1,
chunksize=100,
passes=10,
alpha='auto',
per_word_topics=True)
```

다. 성능 평가 및 개선

(1) Coherence와 Perplexity

LDA 모델의 내부 적합도를 판단하기 위해 내재적(intrinsic) 평가 방법으로 coherence(일관성)와 perplexity(불확실성)를 이용한다. coherence는 외부의 코퍼스를 참조하여 토픽의 의미론적 일관성(semantic interpretablity)의 척도로 사용되고, perplexity는 모델이 텍스트를 예측하는 데 생기는 불확실성의 척도로 사용된다.

주어진 문서에서 자동으로 생성된 토픽들의 coherence를 판단하기 위해 토픽에서 N개의 상위 단어들을 대상으로 측정한다. PMI와 같은 토픽 내 단어들의 단어 유사도 평균 또는 중간값으로 정의될 수 있다. 따라서 모델이 높은 coherence를 나타낼수록 좋은 모델이라고 볼 수 있다.
Perplexity는 확률 모델이 실제로 관측되는 값을 잘 예측하고 있는지 평가할 때 사용된다. 수식으로 나타내면 다음과 같다.

$$Perp = 2^{-\frac{\sum_{w} LL(w)}{N}}$$

지수부분은 정보 엔트로피를 나타내며, LL(w)는 log likelihood, 즉 특정 단어가 특정 토픽으로 배정될 확률 값에 로그를 씌운 값이다. 대개 깁스 샘플링을 반복할수록 perplexity는 감소하며 특정 시점 이후로 더이상 감소하지 않고 증감을 반복한다. 이를 수렴 지점으로 보며 확률 모델의 perplexity라고 정의할 수 있다. 모델이 낮은 perplexity를 나타낼수록 성공적으로 학습되었다고 볼 수 있다.

본 프로젝트에서는 **CoherenceModel** 메소드를 이용해 각각의 조건에 대한 모델의 coherence를 측정하고, 이 값이 가장 최대가 되는 조건 하의 모델을 최종 모델로 채택하였다.

```
coherence_model_lda = CoherenceModel(model=lda_model, texts=data_words_bigrams,
dictionary=id2word, coherence='c_v')
coherence_lda = coherence_model_lda.get_coherence()
print('\nCoherence Score: ', coherence_lda)
```

(2) 다양한 요소 변경 : ngram/lda 모델 파라미터, 길이제한

1) 길이 제한

크롤링한 초록들 중 길이가 짧은 초록들은 초록의 의미를 제대로 내포하지 못한다고 생각하여 길이가 100 이하인 초록들을 대상에서 제외하였다. 이 결과, 단어들만 나열된 초록, 그리고 논문의 초록이 아닌, Index의 초록 같은 엉뚱한 대상들이 제외되었다.

2) ngram의 parameter

ngram은 문자열을 N개 기준의 단위로 절단하는 방법이다. 이 방법을 쓴 이유는 가령 big data의 경우 big data라는 단어는 의미가 있으나 big, data 별개로는 특별한 의미가 없다. 초록을 토큰화 하였을 때, 이러한 문제가 발생하지 않도록 하기 위해서 ngram중 bigram(2개 기준의 단위로 구분)을 적용하였다. bigram은 gensim 패키지의 **gensim.models.Phrases**를 이용하여 구현할 수 있다.

```
gensim.models.Phrases(data_words, min_count=5, threshold=100)
```

이 함수는 적용할 words list와, min_count, threshold 세 가지의 parameter로 이루어져 있는데, min_count는 2개의 연속된 단어가 나와야하는 최소 횟수이고, threshold는 단어들로 학습하면서

넘어야 하는 임계치이다. 이 임계값이 높을 수록 bigram으로 되는 것이 어렵다. 우리는 min_count와 threshold 값을 변경하면서 적합한 값을 찾아내려고 노력하였다.

3) lda 모델 파라미터

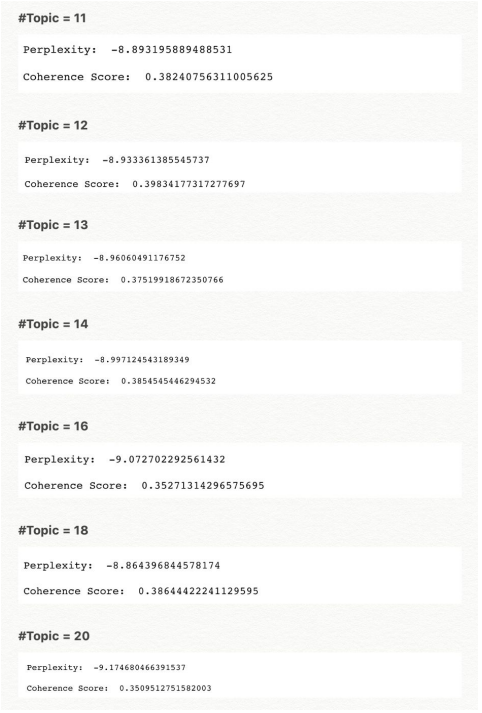
lda model 은 bigram을 만들 때와 마찬가지로, gensim 패키지의 함수를 이용하여 만들 수 있다.

```
lda_model = gensim.models.ldamodel.LdaModel(corpus=corpus,
id2word=id2word,
num_topics=20,
random_state=100,
update_every=1,
chunksize=100,
passes=10,
alpha='auto',
per_word_topics=True)
```

- **corpus** : corpus는 document 별 (word_id, word_frequency)로 이루어진 2차원 list이다. 가령 (0,1)은 단어ID 0이 문서에서 한번 발생한다는 것을 의미한다.
- **id2word** : id2word는 단어들을 id별로 정리해둔 사전입니다. id2word[wordID]는 wordID에 해당하는 word를 보여준다.
- **num_topics** : 뽑을 토픽의 수를 설정한다.
- **update_every :** 각 업데이트에 대해 반복되는 문서 수이다. 일괄 학습의 경우 0으로 설정하고 온라인 반복 학습의 경우 1 이상으로 설정한다.
- **chuncksize :** 학습에 사용될 document chunck의 크기이다.
- **passes :** 학습 중 corpus의 pass 수이다.
- **apha :** 'asymmetric'과 'auto'가 있는데, 'asymmetric'은 고정된 값을 사용하는 것이고, auto는 corpus를 바탕으로 학습한다.
- **per_word_topics :** True이면 각 단어에 대해 가장 가능성이 높은 항목의 내림차순으로 정렬한 주제 목록을 계산한다.

이 parameter들 중 가장 큰 영향을 주는 num_topics값을 변경하여 토픽 수에 따른 모델의 적합성(coherence/perplexity)을 평가하였다. coherence 점수가 높아 지다가 낮아지는 점을 최적의 토픽 수로 삼는다.

최종적으로, bigram에서는 min_count = 2, threshold=20 인 것이 성능이 제일 좋았고, 이 값을 기준으로 가장 성능이 좋은 topic의 수는 12였다.



3. 클러스터링

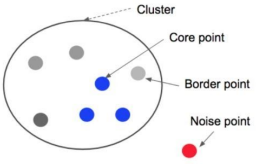
가. 전처리

앞서 진행한 논문 초록을 바탕으로 트렌드를 분석하는 것과 별개로 논문에 등록된 키워드를 바탕으로 클러스터링하여 트렌드를 분석해보았다. 이를 위해 각각의 키워드를 워드 벡터로 임베딩하는 과정이 필요한데, 이를 위해 Wikipedia2vec(https://wikipedia2vec.github.io/wikipedia2vec/) 패키지를 활용하였다. 위키피디아를 코퍼스로 이미 학습이 완료된 word, entity 벡터를 가져와 사용하는 방식을 채택했다.
이를 위해 구축된 키워드 리스트에서 너무 적게 나타난 키워드(10회 미만)는 삭제하고, 중복을 제거하여 총 116,870개의 키워드에서 1,407개의 키워드를 추출하였고, 다시 Wikipedia2vec의 워드 벡터와 맵핑하여 1,065개 키워드를 대상으로 클러스터링을 진행하였다.

나. Mean-Shift

Mean-Shift 클러스터링은 데이터 포인트의 밀집된 영역을 찾기 위해 시도하는 슬라이딩 윈도우 기반 알고리즘이다. K-means 클러스터링과는 대조적으로 클러스터 수를 정해줄 필요가 없다.
우리의 키워드를 가지고 Mean-Shift를 실행해보았으나 우리의 키워드들을 하나의 큰 클러스터 덩이로 인식하여 99% 이상의 키워드들이 하나의 클러스터에 들어가는 결과가 나왔다.

다. DBSCAN



DBSCAN 은 밀도 기반 클러스터링으로, 특정 지점에서 e(epsilon)의 거리 내에 m(minPts)숫자만큼의 점이 있으면 해당 지점을 Core point로 간주하며, 범위를 공유하는 모든 Core point들을 하나의 클러스터로 묶는 방식이다. 초기에 e, m을 어떻게 지정하는지에 따라 클러스터의 갯수와 어느 클러스터에도 포함되지 못하는 Noise point의 비율이 달라지는데, Word2Vec 의 결과가 가운데 지점에만 밀도가 몰린 형태로 나왔기에 e,m값을 다양하게 조정해보아도 클러스터가 1개로 나오거나, 그렇지 않을 경우 noise값이 지나치게 많아지는 결과가 나왔다.

라. K-means

K-means 클러스터링은 클러스터 몇개로 묶을지 k값을 정해주면 k 개의 클러스터로 묶어주는 알고리즘이다. 각 클러스터와 거리 차이의 분산을 최소화하는 방식으로 동작한다.
우리의 키워드를 가지고 실험한 결과 클러스터 별로 키워드 수도 비교적 고루 분포되었고, 클러스터 안의 키워드들도 비슷한 단어끼리 묶였다 판단되었기에 K-means 방법을 선택하였다.

a. 트렌드 분석하기
   i. 시계열 그래프 그리기

gensim 패키지를 이용하여 lda model을 만들고, 내장 함수인 get_documnet_topics를 이용하여, 해당 document가 각 토픽에 속할 확률들을 구하여 데이터프레임으로 만들었다.

```
for i,doc in enumerate(corpus):
    for freq in lda_model.get_document_topics(doc):
        absTopicTable.loc[i,freq[0]] = freq[1]
```

	0	1	2	3	4	5	6	7	8	9	10	11
0	0.020531	0.029747	0.241805	0.172464	0.023211	0.011867	0.191602	0.013301	0.097525	0.309500	0.077546	0.010668
1	0.103145	0.016687	0.115218	0.147445	0.015073	0.014108	0.014510	0.029326	0.022410	0.361029	0.136884	0.024164
2	0.117440	0.170243	0.234308	0.110922	0.000000	0.000000	0.014382	0.016160	0.014899	0.043534	0.267562	0.000000
3	0.196721	0.040128	0.098700	0.249598	0.013371	0.000000	0.000000	0.046572	0.000000	0.102683	0.232052	0.000000
4	0.029044	0.054693	0.213725	0.042339	0.000000	0.000000	0.000000	0.027080	0.048049	0.217357	0.348338	0.000000

그리고 pubDate를 알기 위해 기존 abstract 데이터프레임의 pubDate를 가져온 후, pubDate를 index로 만들었다.

```
sorted_by_date = sorted_by_date.set_index('pubDate', inplace=False)
sorted_by_date.head()
```

pubDate	0	1	2	3	4	5	6	7	8	9	10	11
2004-01-01	0.122025	0.063812	0.111958	0.208369	0.000000	0.000000	0.010760	0.107579	0.000000	0.046298	0.253237	0.064776
2004-01-01	0.048664	0.282993	0.293096	0.042974	0.014000	0.000000	0.000000	0.042676	0.037607	0.115100	0.110963	0.000000
2004-01-01	0.052467	0.197789	0.247565	0.080344	0.013221	0.021024	0.032485	0.095776	0.000000	0.069303	0.170522	0.012322
2004-01-01	0.022311	0.207719	0.111501	0.228563	0.000000	0.000000	0.055874	0.043676	0.000000	0.088588	0.198808	0.025080
2004-01-01	0.029906	0.210118	0.171445	0.018147	0.013608	0.000000	0.040615	0.037048	0.000000	0.034574	0.288348	0.145807

실험중 abstract의 pubDate가 1900년으로, 엉뚱한 값이 들어가 있는 경우는 제외하였다. 그리고 제외한 후 2004년 이전의 논문들 수가 각 년도별로 10개 내외였기 때문에, 실험의 정확성을 위해서 2004년 전의 데이터도 제외하였다.

<처리 후 각 년도별 논문의 수>

```
pubDate
2004-12-31    374
2005-12-31    456
2006-12-31    726
2007-12-31    787
2008-12-31    526
2009-12-31    577
2010-12-31    506
2011-12-31    712
2012-12-31    720
2013-12-31    435
2014-12-31    480
2015-12-31    551
2016-12-31    562
2017-12-31    649
2018-12-31    594
Freq: A-DEC, dtype: int64
```

```
sorted_by_date.resample('Y').mean()
```

2004년 전의 데이터를 제외한 후 먼저 날짜 별로 정렬하였다. 그리고 pandas.DataFrame에 내장된 날짜 별로 데이터를 묶어주는 함수 resample을 사용하여 년도별로 데이터프레임을 묶어주었다. 마지막으로 평균을 구해주는 내장함수인 mean()을 이용하여 년도별 토픽의 확률 평균값을 구하고 plot 함수를 이용해 나타내었다.

4. 회귀분석

년도별 각 토픽의 비중 값이 담긴 데이터프레임을 기반으로 근래에 활발하게 연구되고 있는 연구 토픽인 Hot Topic과 점차 연구되지 않고 있는 연구 토픽인 Cold Topic을 구분하기 위해서 회귀분석을 하였다. 유의수준 5%에서 유의한 확률을 가지는 토픽들과 Durbin-Watson 값이 1.36054 이상 2.63946 이하인 토픽들만을 대상으로 회귀계수값이 양수(+)이면 Hot 핀테크 기술(토픽), 음수(-)이면 Cold 핀테크 기술(토픽)로 구분하였다.(S. H. Seo, 2016)

독립변수는 시기로, 종속변수는 토픽의 시기별 비중 평균값으로 한 후 Statsmodel의 OLS 클래스를 이용해 토픽별로 단순회귀를 적용하였다. 적용 후 선형회귀의 기울기와 y 절편은 각각 year의 coef와 const의 coef에 해당한다.

그 외에도 중점적으로 봐야할 값이 몇가지 정해져있는데 바로 p-value와 Durbin-Waston 검정 값이다. 먼저 p-value는 유의수준으로 귀무가설이 맞는데 잘못해서 기각할 확률의 최대값이다. 유의수준이 5%라면 이는 “선형 회귀 일차식 기울기가 0이다”라는 귀무가설을 기각하는 검절결과의 5%는 잘못될 가능성이 있고 95%는 신뢰할 수 있다는 것이다. 즉, 선형 회귀식 기울기가 0이 아니므로 시기와 토픽 비중이 관련이 있음을 말할 수 있다. 유의 수준 값이 낮을수록 정밀한 검정인데 이번 실험에서는 유의수준을 5%로 설정하였다.

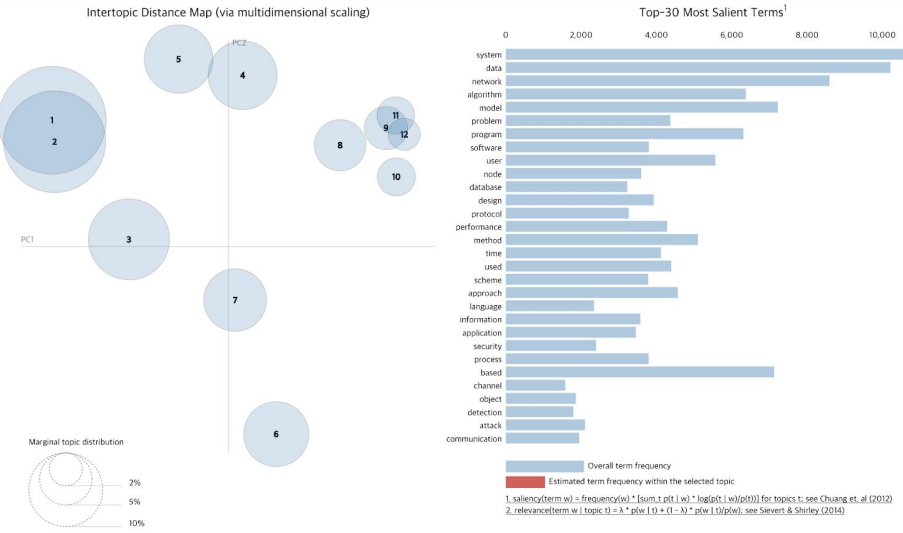
그 다음으로 봐야할 값은 Durbin-Watson 값으로 선형회귀분석 또는 다중회귀분석을 실시할 때 잔차의 독립성을 검증하기 위해 쓰인다. (잔차의 독립성이란 회귀분석 결과의 타당성을 위해 가지는 기본 가정으로 실제 자료 값에서 예측값을 뺀 나머지 잔차끼리 서로 상관을 보여서는 안된다는 의미이다.) 더빈 왓슨의 수치가 0에 가까울수록 양의 상관관계 4에 가까울수록 음의 상관관계를 나타낸다고 볼 수 있는데, 0과 4에 가까워질수록 자기상관이 존재하는 것이고 2에 가까울수록 잔차의 독립성이 보장되는 것이다.

III. 실험 결과

1. LDA 모델을 이용한 트렌드 분석

<각 토픽 별 분포와 가장 많이 속한 단어>

pyLDAvis로 구현하였다. 가로,세로 축은 의미가 없으며, 원의 크기로 분포도를 파악할 수 있다.

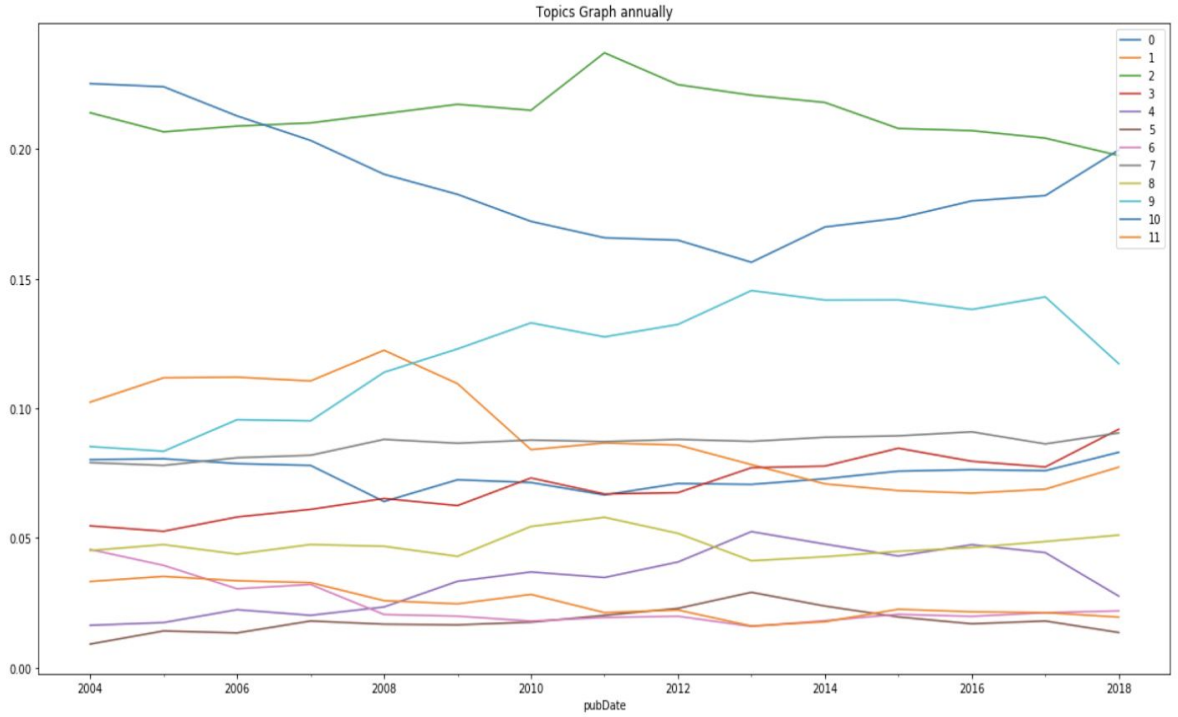


<Topic 별 관련높은 keyword>

	0	1	2	3	4	5	6	7	8	9
topic0	('system', 0.1295)	('software', 0.0449)	('design', 0.0379)	('application', 0.0231)	('communication', 0.023)	('implementation', 0.0218)	('reliability', 0.0146)	('distributed', 0.0143)	('testing', 0.0143)	('computer', 0.0138)
topic1	('network', 0.0914)	('algorithm', 0.0678)	('problem', 0.0465)	('node', 0.0383)	('graph', 0.0188)	('path', 0.013)	('traffic', 0.0114)	('number', 0.0112)	('optimal', 0.01)	('tree', 0.0098)
topic2	('model', 0.0309)	('based', 0.0201)	('method', 0.0198)	('approach', 0.0191)	('used', 0.0188)	('process', 0.0162)	('technique', 0.016)	('using', 0.0144)	('two', 0.0142)	('set', 0.0141)
topic3	('data', 0.1298)	('database', 0.041)	('language', 0.0298)	('framework', 0.0206)	('query', 0.0191)	('user', 0.0156)	('support', 0.0139)	('application', 0.0138)	('mobile', 0.0133)	('management', 0.0125)
topic4	('display', 0.016)	('malware', 0.0125)	('bound', 0.0123)	('feedback', 0.0114)	('human', 0.011)	('prediction', 0.0109)	('life_cycle', 0.0099)	('contact', 0.0095)	('author', 0.0083)	('grammar', 0.0083)
topic5	('virtual', 0.034)	('broadcast', 0.0231)	('distance', 0.0188)	('password', 0.0151)	('page', 0.0091)	('check', 0.0087)	('command', 0.0085)	('predicate', 0.0084)	('paper_examines', 0.0078)	('secret', 0.0072)
topic6	('concurrent', 0.0165)	('minimum', 0.0138)	('programmer', 0.0138)	('logic', 0.0129)	('design', 0.0118)	('statement', 0.0117)	('designer', 0.0105)	('fault_tolerance', 0.0103)	('permit', 0.0097)	('cycle', 0.0091)
topic7	('protocol', 0.0343)	('security', 0.0252)	('attack', 0.0221)	('service', 0.0185)	('mechanism', 0.0136)	('based', 0.0131)	('key', 0.0112)	('secure', 0.0084)	('privacy', 0.0065)	('scenario', 0.0057)
topic8	('object', 0.0351)	('detection', 0.0339)	('feature', 0.0292)	('accuracy', 0.0174)	('image', 0.0165)	('protection', 0.0138)	('vector', 0.0113)	('sample', 0.0102)	('estimate', 0.0102)	('surface', 0.0099)
topic9	('user', 0.033)	('information', 0.0246)	('study', 0.0161)	('development', 0.0128)	('environment', 0.0107)	('task', 0.0098)	('paper', 0.0068)	('interaction', 0.0063)	('context', 0.0063)	('may', 0.006)
topic10	('program', 0.0303)	('performance', 0.0206)	('time', 0.0198)	('scheme', 0.0182)	('control', 0.0124)	('proposed', 0.012)	('error', 0.0093)	('access', 0.0084)	('show', 0.0078)	('distributed', 0.0077)
topic11	('channel', 0.0562)	('wireless', 0.0381)	('power', 0.0351)	('sensor', 0.0324)	('transmission', 0.0236)	('mobility', 0.0194)	('frequency', 0.0145)	('wireless_sensor', 0.0139)	('signal', 0.0132)	('sensing', 0.0121)

<토픽당 년도별 언급량의 평균값 그래프>

```
In [191]: sorted_by_date.resample('Y').mean().plot(title='Topics Graph annually',figsize=(20,10))
Out[191]: <matplotlib.axes._subplots.AxesSubplot at 0x1a2cd2f470>
```



2004년부터 2018년까지의 년도별 평균 언급량을 토픽별로 나타내었다.

<2004~2018년의 토픽별 언급량 평균값>

```
sorted_by_date.mean().sort_values(ascending=False)
```

```
2     0.213648
10    0.186294
9     0.121007
1     0.090669
7     0.086030
0     0.074386
3     0.069871
8     0.047804
4     0.033629
11    0.025023
6     0.023801
5     0.017934
dtype: float64
```

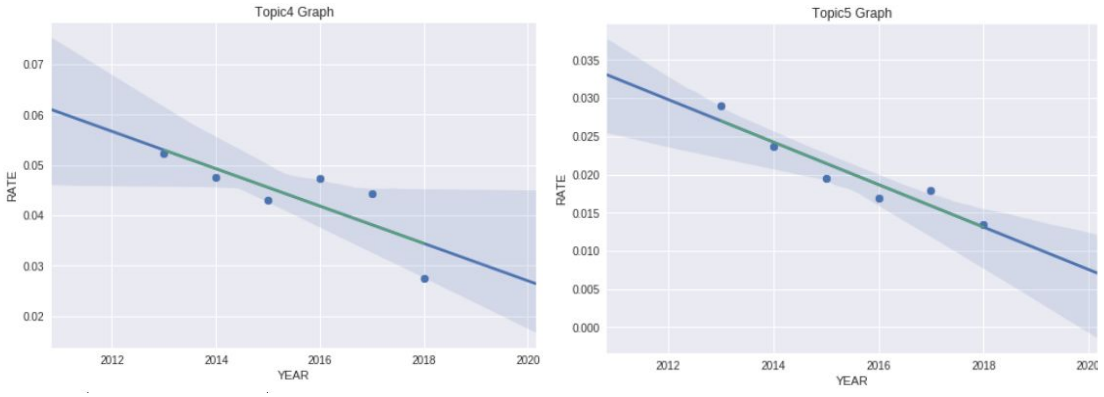
그리고 10년간의 언급량을 평균내보면 위 순서대로 전반적인 언급량 순위를 알 수 있다. 2,10번 토픽의 언급량이 각각 20%씩으로, 전체 언급량에서 많은 비중을 차지하는, 보편적인 주제임을 알 수 있다.

<토픽별 선형 회귀 분석 결과>

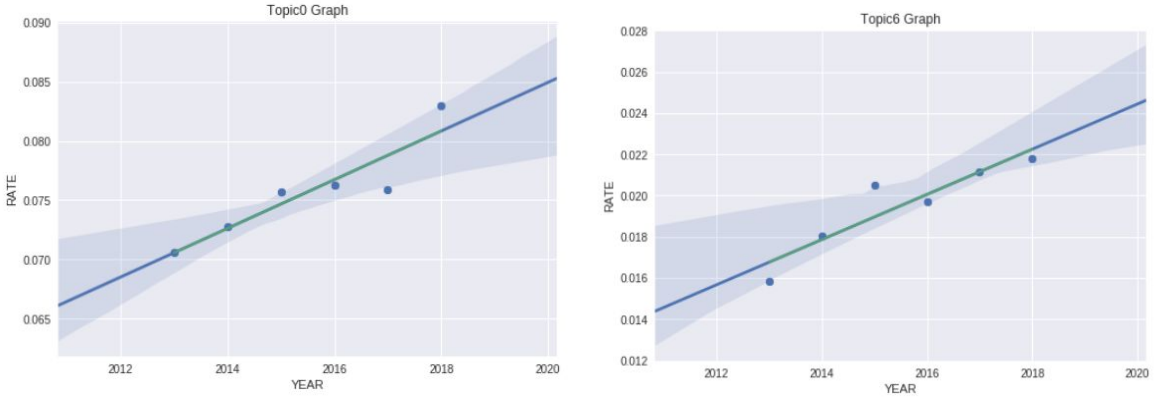
	coef	intersect	p-value	Durbin_Waston	Hot/Cold
0	0.00205269	-4.0615	0.0112621	2.40665	Hot
1	-0.000341148	0.759331	0.781819	1.2145	-
2	-0.00452563	9.33057	0.000961063	2.74422	-
3	0.00194965	-3.8482	0.196643	2.35636	-
4	-0.00371135	7.52391	0.0495808	1.89655	Cold
5	-0.00278336	5.62994	0.00415377	1.65924	Cold
6	0.00109631	-2.19011	0.00861768	1.92897	Hot
7	0.00028756	-0.490773	0.627113	2.87068	-
8	0.00195694	-3.89845	2.28416e-05	1.62542	Hot
9	-0.00403143	8.26316	0.098929	2.21545	-
10	0.00741721	-14.7726	0.00212733	2.49219	Hot
11	0.000765342	-1.52286	0.237225	1.34382	-

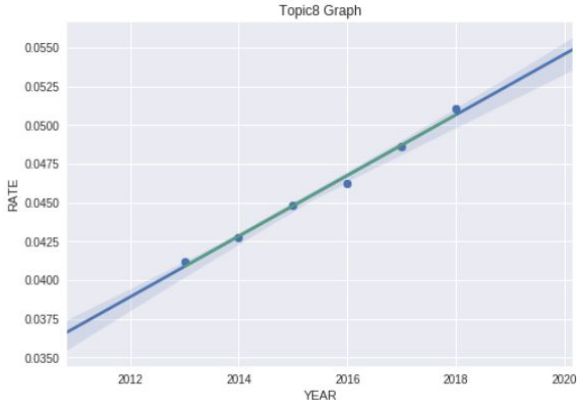
트렌드 분석을 위해 전체 기간 보다는 최근 6년간의 데이터를 가지고 선형 회귀를 해보았다. 이를 통해 토픽 0, 토픽 6, 토픽8은 기울기가 약 0.001 ~ 0.002로 연구가 증가하는 추세임을 알 수 있다. 특히, 토픽 10은 기울기가 0.007로 다른 토픽에 비해 확연히 높은 증가율을 보임으로써 트렌드임을 알 수 있다.

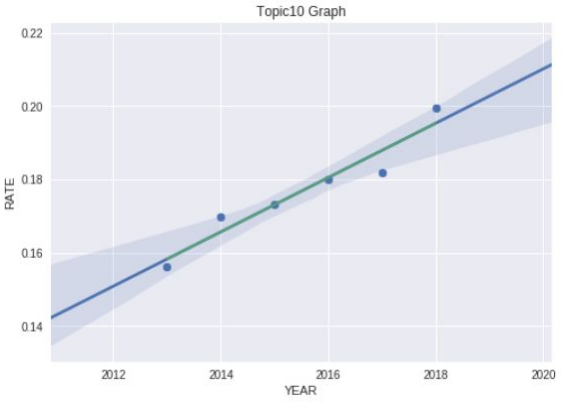
<Cold Topic 선형 회귀 결과>



<Hot Topic 선형 회귀 결과>







2. 클러스터링을 이용한 트렌드 분석

<Topic 별 관련높은 keyword>

	0	1	2	3	4	5	6	7	8	9
topic0	['Mobile communication'	Cellular networks'	Mobile Computing'	Electronic Mail'	Wireless Network'	Mobile handsets'	Intelligent network'	Interconnect'	Code Division Multiple Access'	Wireless Local Area Network'
topic1	['Computer Displays'	Crosstalk'	Acoustics'	Synchronisation'	Broadcast'	Frequency'	Displays'	Noise'	Radio networks'	Cameras'
topic2	['Distributed applications'	Hadoop'	Computer errors'	Virtual machine monitor'	Cloud Storage'	Checkpointing'	Software As A Service'	Distributed system'	Fault Tolerance'	Peer to peer network'
topic3	['Market Research'	Prototypes'	Reputation'	Trojan Horses'	Logistics'	Education course'	Licenses'	Consensus'	Advertising'	Anonymity'
topic4	['CUDA'	Computer Graphics'	Mesh generation'	Graph visualization'	Graphics hardware'	Finite Element Analysis'	Image texture'	Ray Tracing'	GPGPU'	Parallel coordinates'
topic5	['Systems Biology'	Phylogenetics'	Proteomics'	Gene regulatory networks'	Gene selection'	Bioinformatics'	Protein sequence'	Molecular Biophysics'	Biochemistry'	Protein structure prediction'
topic6	['Oscillators'	Sun'	Light scattering'	Friction'	Surface texture'	Accuracy'	Kinematics'	Fluid Dynamics'	Time measurement'	Storms'
topic7	['Support Vector Machine'	Knowledge discovery'	Unsupervised learning'	Training data'	Object Recognition'	Face recognition'	Affective Computing'	Emotion Recognition'	Reinforcement Learning'	Hidden Markov Models'
topic8	['Logic design'	Field programmable gate array'	Embedded processor'	Programmable logic devices'	System On Chip'	Sequential Circuits'	Cache Memory'	Computer Architecture'	Switching circuit'	Threshold Voltage'
topic9	['Automatic control'	Manifolds'	Network synthesis'	Queueing Theory'	Optimal Control'	Asymptotic stability'	Controllability'	Transient analysis'	Proportional Control'	Control Systems'

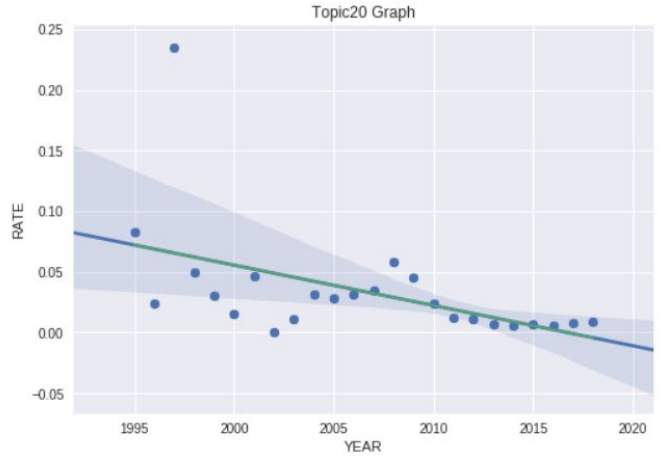
topic10	['Probability'	Estimation'	Regression Analysis'	Stochastic systems'	Parameter estimation'	Histograms'	Monte Carlo Methods'	Time Series Analysis'	Matrix Algebra'	Distribution functions'
topic11	['Avatars'	Search Engines'	Google'	Speech Recognition'	Computer Games'	Virtual Reality'	Force feedback'	Video Games'	User interfaces'	Mobile Cloud Computing'
topic12	['Machine learning algorithms']									
topic13	['Remote laboratories'	Online Learning'	Data collection'	Evaluation'	Information Management'	Information Visualization'	Abstracts'	Medical Simulation'	Social Networks'	Electronic Publishing'
topic14	['Facial expressions'	Skin'	Visual Perception'	Prosthetics'	Tumors'	Cancer'	EEG'	Haptic perception'	Fingers'	Joints'
topic15	['Table lookup'	Singular Value Decomposition'	Video coding'	quantization'	Digital Signal Processing'	Galois fields'	Convolutional codes'	Convolution'	Compressed Sensing'	Data Compression'
topic16	['Telerobotics'	Computer Security'	Mobile Robots'	Context awareness'	Space Technology'	Sensors'	Nanotechnology'	Detectors'	Humanoid robots'	Human Computer Interaction'
topic17	['Traffic modeling'	Approximation algorithm'	Decision Trees'	Tree data structure'	Boolean functions'	Biclustering'	Graph Theory'	Homomorphic Encryption'	Concurrent computing'	Circuit complexity'
topic18	['latches'	visualization'	wires'	modeling'	cache'	registers'	aggregates'	tunneling'	coherence'	modelling'
topic19	['History'	Algebra'	Economics'	Pragmatics'	Terminology'	Dictionaries'	Behavioral Science'	Ethics'	Psychology'	Vocabulary'
topic20	['Internetworking'	Ad hoc wireless network'	Congestion Control'	Packet Switching'	Scheduling algorithms'	Bit error rate'	Automatic Repeat Request'	Channel allocation'	Data communication'	Mobile ad hoc network'
topic21	['Usability'	Authorization'	Unified Modeling Language'	Network capacity'	Planning'	Human Factors'	Data Processing'	Integrity'	Decision Support Systems'	Fault injection'
topic22	['Vegetation'	Energy saving'	Urban areas'	Power demand'	Power consumption'	Distributed network'	Energy consumption'	Power Generation'	Trees'	Heating'
topic23	['Complex Networks'	Artificial'	Ontologies'	Problem Solving'	Procedural content generation'	Game Theory'	Analysis'	Evolutionary algorithms'	Numerical models'	Simulation'
topic24	['Engines'	Silicon'	Low Voltage'	Pins'	Voltage'	Dc motors'	Couplings'	Temperature sensor'	Gold'	Power supplies'
topic25	['Writing'	Performance'	Design'	Art'	Motion Pictures'	Bridges'	Floods'	Music'	Chemicals'	Concrete'

<토픽별 선형 회귀 분석 결과>

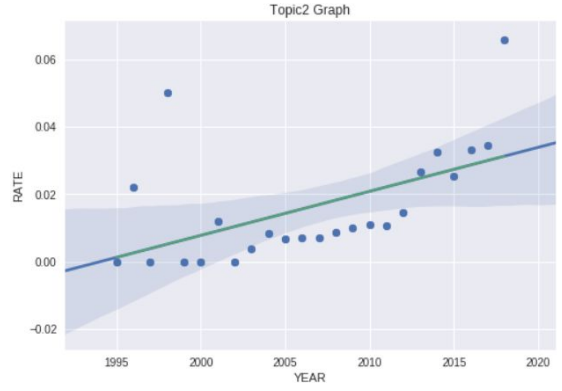
	coef	intersect	p-value	Durbin_Waston	Hot/Cold
0	0.00081647	-1.62519	0.0011162	1.05693	-
1	-0.00069678	1.41003	0.0723905	1.27612	-
2	0.00130647	-2.60519	0.00647692	1.58456	Hot
3	0.000274186	-0.543208	0.402614	2.0692	-
4	-0.000176162	0.356677	0.218537	2.69382	-
5	0.00234691	-4.69379	1.06907e-06	0.638043	-
6	0.000350712	-0.694342	0.305994	2.17791	-
7	0.00133166	-2.66173	4.65344e-10	1.29789	-
8	-0.00550127	11.0839	0.0113323	2.65189	-
9	-0.000258703	0.531888	0.456553	1.79471	-
10	0.000633906	-1.26623	1.85703e-09	1.20124	-
11	0.000711136	-1.4213	1.32118e-08	0.513763	-
12	0	0	NaN	NaN	-
13	0.000207736	-0.412259	0.177095	1.75074	-
14	0.000833049	-1.66346	0.000511583	1.45871	Hot
15	-0.000273226	0.551347	0.112929	2.30214	-
16	-0.000610736	1.23981	0.137596	1.10074	-
17	0.000466399	-0.927046	0.00965554	1.88343	Hot
18	2.76664e-05	-0.0528246	0.837119	1.22395	-
19	-0.000217118	0.446664	0.506524	1.53738	-
20	-0.00332395	6.7035	0.0132228	2.25151	Cold
21	0.000268783	-0.520625	0.544504	2.74062	-
22	0.000117588	-0.235019	5.38553e-07	1.77757	Hot
23	0.000453521	-0.903585	0.002318	1.96877	Hot
24	-0.000506493	1.02596	0.156588	2.02196	-
25	-0.000306212	0.628887	0.639525	2.44711	-

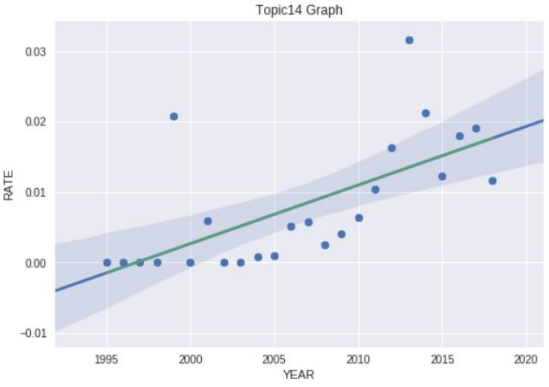
이를 통해 토픽 2, 토픽 14는 기울기가 약1.4 ~ 1.5로 연구가 증가하는 추세임을 알 수 있다. 특히, 토픽17, 토픽 22, 토픽23은 기울기가 1.7 ~ 2.0로 다른 토픽에 비해 약간 높은 증가율을 보이고 있다.

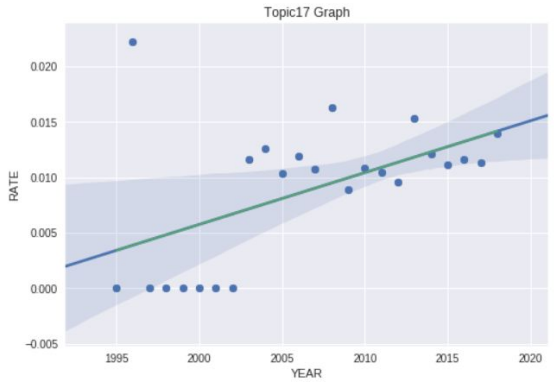
<Cold Topic 선형 회귀 결과>

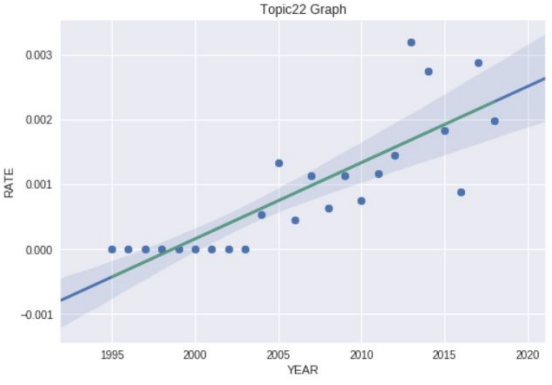


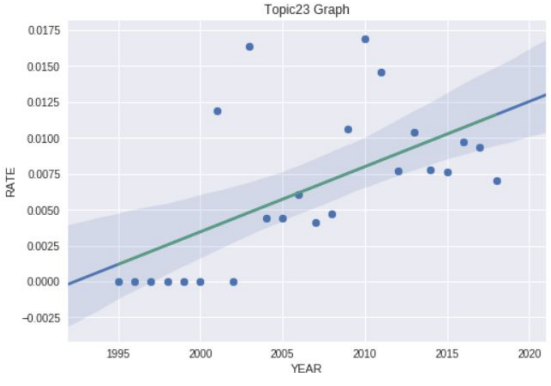
<Hot Topic 선형 회귀 결과>











IV. 토의 및 결론

컴퓨터 과학 분야의 논문들을 기반으로 어떤 Keyword가 오늘날 IT분야의 트렌드가 되고 있는지를 알기위해서 IEEE Computer Science 분야 논문 초록을 크롤링하였다. 그 후 크게 2가지 방법을 이용하여 토픽을 추출하였다. 첫번째는 LDA 모델을 이용하여 직접 토픽을 뽑은 후 선형 회귀를 이용하여 트렌드를 분석하였고 두 번째는 컨퍼런스 저자가 뽑은 키워드를 워드 클러스터링통해 묶은 토픽 대상으로 선형 회귀를 이용하여 트렌드를 분석하였다.

LDA 모델을 이용한 트렌드 분석 결과, 토픽 0, 토픽 6, 토픽 8, 토픽 10 분야 연구가 증가하는 추세임을 알 수 있으며, 특히 토픽 10이 가장 높은 증가율로 보아 학계에서 가장 활발히 연구되는 분야임을 알 수 있다.

Topic	Keywords
0	'system','software','design','application','communication','communication','reliability'...
6	'concurrent','minimum','programmer','logic','design','statement','designer'...
8	'object','detection','feature','accuracy','image','protection','vector','sample','estimate'...
10	'program','performance','time','scheme','control','proposed','error','access'...

워드 클러스터링을 이용한 트렌드 분석 결과, 토픽 2, 토픽 14, 토픽17, 토픽 22, 토픽23 분야 연구가 증가하는 추세임을 알 수 있으며, 특히 토픽17, 토픽 22, 토픽23이 상대적으로 높은 증가율로 보아 학계에서 가장 활발히 연구되는 분야임을 알 수 있다.

Topic	Keywords
2	'Distributed applications', 'Hadoop', 'Computer errors', 'Virtual machine monitor', 'Cloud Storage'...
14	'Facial expressions', 'Skin', 'Visual Perception', 'Prosthetics', 'Tumors', 'Cancer', 'EEG',...
17	‘Traffic modeling', 'Approximation algorithm', 'Decision Trees', 'Tree data structure',...
22	'Vegetation', 'Energy saving', 'Urban areas', 'Power demand', 'Power consumption', ...
23	'Complex Networks', 'Artificial', 'Ontologies', 'Problem Solving', 'Procedural content generation', ...

본 연구는 한계점으로는 다음과 같이 크게 2가지가 존재한다. 첫번째로는 짧은 논문 초록으로 인한 LDA 모델의 한계점이다 .논문 초록은 단문으로써 전처리 과정에서 불용어를 제거하면 의미를 내포하는 단어가 매우 희소해진다. 하지만 LDA는 디리클레 분포를 기반으로 각 문서 내의 단어들이 특정 주제에 포함될 확률을 계산하기 때문에 주제를 추론하고자 하는 문서가 장문일 때만 높은 정확도를 보인다. 즉, 단문인 논문 초록을 대상으로 할 때는 낮은 정확도를 보이는 한계점을 가진다.

또한, 위키백과의 word2vec 모델을 이용해 클러스터링을 하기 위해서는 모델 표제어에 크롤링한 키워드들이 등록되어있어야 한다. 10개이상 나온 키워드들이 총 1407개인데 위키 표제어에 등록되어 있는 키워드 수는 1065개이다. 위키 표제어에 등록되지 못했다는 이유로

337개의 키워드는 10번 이상 나옴에도 불구하고 워드 클러스터링에 표현할 수 없는 한계점을 가지고 있다.

향후 연구에서는 LDA 모델 대신 BTM(Biterm Topic Model)을 이용하여 토픽 추출을 진행하고자 한다. BTM은 특정 주제의 의미를 내포한 단어가 한번 이상 재출현 하는 경우가 드문 단문을 대상으로, biterm이라는 두 단어의 조합을 통해 주제를 추론하는 알고리즘이다. BTM은 biterm을 이용하기 때문에 단문이라는 초록 특징으로 인한 LDA 모델의 한계점을 극복할 수 있다.

참고문헌

잠재 디리클레 할당, 위키피디아.
https://ko.wikipedia.org/wiki/%EC%9E%A0%EC%9E%AC_%EB%94%94%EB%A6%AC%ED%81%B4%EB%A0%88_%ED%95%A0%EB%8B%B9
Topic Modeling, LDA, 2017.06.01.
https://ratsgo.github.io/from%20frequency%20to%20semantics/2017/06/01/LDA/
Reactive LDA Library, Yingjie Miao, 2014.09.19.
https://medium.com/kifi-engineering/reactive-lda-library-d495ed2a6342
Newman et al. Automatic Evaluation of Topic Coherence, Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL, 2010.06.
Callaghan et al. An Analysis of the Coherence of Descriptors in Topic Modeling, Expert Systems with Applications, 2015.02.12.
What is topic coherence?, 2016.11.08. https://www.quora.com/What-is-topic-coherence
인프런 <[NLP] IMDB 영화리뷰 감정 분석을 통한 파이썬 자연어 처리>
https://www.inflearn.com/course/nlp-imdb-%ED%8C%8C%EC%9D%B4%EC%8D%AC-%EC%9E%90%EC%97%B0%EC%96%B4-%EC%B2%98%EB%A6%AC/
Topic Modeling with Gensim(Python), 2018.05.23.
http://www.engear.net/wp/topic-modeling-gensimpython/
Seo, S. H.(2016). *Fintech trend analysis using topic modeling of BM patents* (Graduate School of Seoul National University of Science and Technology).
https://web.stanford.edu/~clint/bench/dwcrit.htm
Pandas를 통한 시간별 시각화, 2017.10.07. http://enjoyiot.tistory.com/category/Python