KIPS Tr. Software and Data Eng.
Vol.5, No.11 pp.521~526 pISSN: 2287-5905

http://dx.doi.org/10.3745/KTSDE.2016.5.11.521

A Design on Informal Big Data Topic Extraction System Based on Spark Framework

Kiejin Park†

ABSTRACT

As on-line informal text data have massive in its volume and have unstructured characteristics in nature, there are limitations in applying traditional relational data model technologies for data storage and data analysis jobs. Moreover, using dynamically generating massive social data, social user's real-time reaction analysis tasks is hard to accomplish. In the paper, to capture easily the semantics of massive and informal on-line documents with unsupervised learning mechanism, we design and implement automatic topic extraction systems according to the mass of the words that consists a document. The input data set to the proposed system are generated first, using N-gram algorithm to build multiple words to capture the meaning of the sentences precisely, and Hadoop and Spark (In-memory distributed computing framework) are adopted to run topic model. In the experiment phases, TB level input data are processed for data preprocessing and proposed topic extraction steps are applied. We conclude that the proposed system shows good performance in extracting meaningful topics in time as the intermediate results come from main memories directly instead of an HDD reading.

Keywords : Topic Model, N-gram, Spark, Hadoop, Machine Learning

Spark 프레임워크 기반 비정형 빅데이터 토픽 추출 시스템 설계

박 기 진†

요 약

온라인상에서 다루어지는 비정형 텍스트 데이터는 대용량이면서 비구조적 형태의 특성을 가지고 있기 때문에, 기존 관계형 데이터 모델의 저장 방식과 분석 방법만으로는 한계가 있다. 더군다나, 동적으로 발생하는 대량의 소셜 데이터를 활용하여 이용자의 반응을 실시간으로 분석하기란 어려운 상황이다. 이에 본 논문에서는 대용량 비정형 데이터(문서)의 의미를 빠르고, 용이하게 파악하기 위하여 데이터 셋에 대한 사전 학습 없이, 문서 내 단어 비중에 따라 자동으로 토픽(주제)이 추출되는 시스템을 설계 및 구현하였다. 제안된 시스템의 토픽 모델링에 사용될 입력 단어는 N-gram 알고리즘에 의하여 도출되어 복수 개의 단어도 묶음 처리할 수 있게 했으며, 또한, 대용량 비정형 데이터 저장 및 연산을 위하여 Hadoop과 분산 인메모리 처리 프레임워크인 Spark 기반 클러스터를 구성하여, 토픽 모델 연산을 수행하였다. 성능 실험에서는 TB급의 소셜 댓글 데이터를 읽어 들여, 전체 데이터에 대한 전처리 과정과 특정 항목의 토픽 추출 작업을 수행하였으며, 대용량 데이터를 클러스터의 디스크가 아닌 메모리에 바로 적재 후, 처리함으로써 토픽 추출 성능의 우수성을 확인할 수 있었다.

키워드 : 토픽모델, N-gram, Spark, Hadoop, 기계학습

1. 서 론

온라인 소셜 미디어 기반의 비정형 데이터는 기존의 정형화된 데이터보다 훨씬 방대하고 다양한 구조를 갖고 있다. 즉 여러 형태의 뉴스기사, 블로그, 상품평 등이 혼재되어 있으며 이러한 텍스트 데이터로 구성된 문서 집합에서 의미(Semantics)를 찾아내기 위한 방법으로, 최근 토픽 모델이 각광을 받고 있다[1]. 토픽 모델은 입력된 문서 집합에 대해 통계 기법을 적용하여 "문서 내에서 특정 단어들이 어떤 의미로 쓰였는지?" 구분해주는 기계학습(Machine Learning) 알고리즘이다 특히, 토픽 모델을 사용하면 문서 내용을 간결하게 나타낼 수 있고, 단어 및 문서 간의 유사도 평가도 가능하기 때문에, 내용 구분이 모호한 비정형의 문서에서 데이터의 숨겨진 특징을 파악하는데 적합하다고 볼 수 있다.

본 논문에서는 대용량 비정형 텍스트 데이터의 의미를 빠르고, 효율적으로 파악하기 위하여 데이터 셋에 대한 사전

※ 이 논문은 아주대학교 학술연구비를 지원받아 연구되었음.
† 종신회원 : 아주대학교 융합시스템공학과 교수
Manuscript Received : October 4, 2016
Accepted : October 13, 2016
* Corresponding Author : Kiejin Park(kiejin@ajou.ac.kr)

학습 없이(Unsupervised), 문서 내 단어 비중에 따라 자동으로 토픽(주제)이 추출되는 시스템을 설계 및 구현하였다. 제안된 시스템의 토픽 모델에 사용될 단어는 N-gram 알고리즘[2]에 의하여 도출되어 복수 개의 단어도 묶음 처리할 수 있게 했으며, 또한, 대용량 비정형 데이터 저장 및 연산을 위한 클러스터 관리자로 Hadoop의 YARN(Yet Another Resource Negotiator)[3]을 채택하였다. 인메모리 기반의 Spark[4] 플랫폼을 적용하여 대용량 데이터에 대한 처리를 디스크에서 메모리로 옮겨 처리 속도를 높였다. 이는 Hadoop 클러스터에 분산 저장된 대용량 데이터 연산시 디스크 접근을 최소화하고 모든 중간 계산 데이터를 메모리에 올려 놓은 상태에서 처리하므로 고속의 처리 효과를 얻을 수 있기 때문이다. 본 논문에서 채택한 확률론적 통계 이론에 바탕을 둔 LDA(Latent Dirichlet Allocation) 토픽 모델 기법[5]은 온라인 상의 비정형 데이터의 숨겨진 특징을 파악하는 것뿐 아니라, 정보 검색, 인공지능, 바이오 인포메틱스 등과 같은 여러 분야에 다양하게 응용될 수 있다.

본 논문은 모두 5장으로 구성되어 있으며, 2장에서는 N-gram 알고리즘과 LDA 토픽 모델에 대하여 설명하였다. 3장에서는 Spark 인메모리 기반 토픽 추출 시스템의 설계 및 데이터 처리 프로세스에 대하여 기술하였고, 4장에서는 프로토타입 시스템 개발 및 실험을 통한 온라인 댓글 데이터에 대한 토픽 추출 결과를 제시하였다. 마지막으로 5장에서는 결론 및 후속 연구에 대해 언급하였다.

2. 관련 연구

문서 데이터 내에서 특정 단어 묶음이 실제적으로 전체 문맥 속에서 개별 단어와 같은 역할을 하기 때문에 “단어 묶음을 잘 찾아낼 경우 해당 문서 전체를 잘 파악할 수 있다”고 판단하였으며, 이에 본 논문에서는 토픽 모델링시 개별 단어 입력보다는 N-gram의 결과물인 단어 묶음을 입력값으로 사용함으로써 우수한 토픽 추출을 달성하고자 하였다. 특정 단어 묶음이 자주 등장한다는 것은 토픽으로 추출될 가능성이 높아지는 것을 뜻하며, 추출된 토픽의 의미 해석에도 유리한 측면이 있기 때문이다.

한편, 토픽 모델은 “문서는 다수의 토픽으로 표현되고, 토픽에 의해 단어가 생성된다”는 전제하에, “토픽의 분포 p(z)와 각 토픽 별로 단어가 생성될 확률 p(w|z)을 알 경우, 이것들을 통해 문서가 생성될 확률 p(d)을 구해낼 수 있다”는 개념이다. 확률 이론에 기반한 토픽 모델은 텍스트로 구성된 문서 내에 숨겨져 있는 토픽(의미)들을 찾기 위해 고안된 통계적 추론 기법이며 대표적으로 PLSA[6]와 LDA가 있다. 이들 중 본 논문에서는 대용량 빅데이터 처리에 적합한, 즉, 분산-병렬 처리에 적합한 LDA 기법을 채택하여, Hadoop YARN 기반 Spark 클러스터 환경에서 토픽 추출 시스템을 설계 및 구현하였다. 한편, [7]에서는 LDA 토픽 모델을 Hadoop 기반 Mapreduce 프레임워크 상에서 온라인 학습기를 개발하였으나, 디스크 기반 연산을 수행함으로 인해, 인메모리 기반 Spark 에 비해 성능에 한계가 있다.

2.1 N-gram 알고리즘

N-gram은 확률과 통계를 바탕으로 한 색인 분석 등에 널리 쓰이는 방식으로 초기 검색 사이트에서 사용한 색인 검색 알고리즘의 하나이다[2]. 음절 단위로 N-gram 알고리즘을 적용했을 경우, n개의 문자열 크기만큼의 창(Window)을 만들어 문자열을 왼쪽에서 오른쪽으로 한 단위씩 움직이며 추출되는 문자 요소 집합(Character Item Set)의 출현을 수집한다. 이러한 방식으로 두 개의 문자열 각각의 문자 요소 집합을 수집한 후 출현 빈도를 비교함으로써 두 문자열을 비교해서 그 결과를 수치로 표현한다. 이때 n 값은 의미가 있다고 생각하는 음절의 수로 지정하면 되는데, 이 값이 1인 경우는 Unigram, 2인 경우는 Bigram, 3인 경우는 Trigram이라고 한다. N-gram의 확률 계산식은 Equation (1)과 같으며, 여기서 w_1^n는 n개 단어의 순차적 나열이며, $w_1 \dots w_n$은 단어를 의미한다.

$$P(w_1^n) = P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_1^2)\dots P(w_n \mid w_1^{n-1}) = \prod_{k-1}^{n} P(w_k \mid w_1^{k-1}) \quad (1)$$

2.2 LDA(Latent Dirichlet Allocation) 토픽 모델

LDA는 하나의 문서가 여러 개의 토픽으로 구성되어 있다고 가정한 다음, 이런 토픽들과 각 문서에서의 토픽 비율을 찾아내는 방법론이며, 각 문서를 구성하는 단어가 서로 독립적이지 않다는 가정(Dirichlet Distribution)에서 단어를 생성하는 조건에 따라 사후 확률을 추론한다. 한편, PLSA 모델은 하나의 문서 내에서 각 단어들이 하나의 토픽하고만 연관되어 있기 때문에 문서 셋 전체에 걸쳐 나타나는 토픽 분포의 경향까지는 나타내지 못하며, 또한 주어진 데이터에 지나치게 맞춰지는 과적합(Over-Fitting) 등의 단점이 있다[5].

LDA 토픽 모델에서 단어들은 특정 토픽들로부터 생성되고, 해당 문서가 어떤 토픽 비율(Topic Proportion) Θ를 가질 것인지는 파라미터가 α인 Dirichlet Distribution에 의해 결정된다고 본다. n개의 문서가 주어져 있고, 모든 문서는 각각 k개의 주제 중 하나에 속할 경우, 문서 생성의 확률은 Equation (2)와 같다.

$$p(D \mid \alpha, \beta) = \prod_{d=1}^{M} \int p(\theta_d \mid \alpha) \left(\prod_{n=1}^{N_d} \sum p(z_{dn} \mid \theta_d) p(w_{dn} \mid z_{dn}, \beta) \right) d\theta_d \quad (2)$$

여기서, p(Θd|α)는 K-dimensional Dirichlet Distribution이고, α, β는 Hidden Parameter, Θ와 z는 Hidden Variable, 단어 w는 유일한 관측 데이터이다. 문서별 토픽의 분포 Θ를 Multinomial Distribution으로 두고 이에 대한 Prior Distribution을 Dirichlet Distribution으로 결정함으로써 잠재변수(hidden variable) Θ에 대한 사후확률(Posterior) 계산을 쉽게 할 수 있다.

3. Spark 인메모리 분산 처리 기반 토픽 추출 시스템

3.1 분산 처리 클러스터 시스템 구성

본 분산처리 시스템은 다양한 비정형 문서 집합에서 토픽들을 추출해 내기 위한 시스템으로 Spark 인메모리 분산 처리 플랫폼을 채택하여, 대용량 데이터에 대한 고속 분석이 가능하도록 설계하였다. 대용량 데이터의 저장을 담당하는 Hadoop 클러스터 상에 동작하는 Spark 플랫폼은, Fig. 1에 나타난 바와 같이 YARN에서는 Hadoop 클러스터 리소스 할당, 작업 스케줄링, 및 결함처리 기능을 제공하고[8], 그 위에 인터렉티브 한 방식으로 동작하는 Spark 메인 프로그램(Driver Program)을 통해 각 Slave 노드(Worker Node)에서 분산-병렬 방식으로 분석 태스크를 수행하게 된다.

Spark는 RDD(Resilient Distributed Data)[9]라는 DAG (Directed Acyclic Graph) 형식의 데이터 단위 처리 구조를 사용하기 때문에 클러스터 구성하는 모든 Slave 노드의 메모리를 효율적으로 사용할 수 있으며, 이러한 이유로 기존 빅데이터 분석 플랫폼에서 사용 중인 MapReduce[10] 기법 적용시 발생하는 디스크 병목 현상이 제거되어 분석에 소요되는 시간을 획기적으로 줄일 수 있다[11]. 제안된 시스템에서는 Master 노드(Cluster Manager)에 YARN의 RM (ResourceManager)를 두었으며, 각각의 Slave 노드에는 AM (ApplicationMaster)과 NM(NodeManager)를 두어서, AM에서는 작업 스케줄링 처리를, NM는 데이터 처리를 담당하도록 했다.

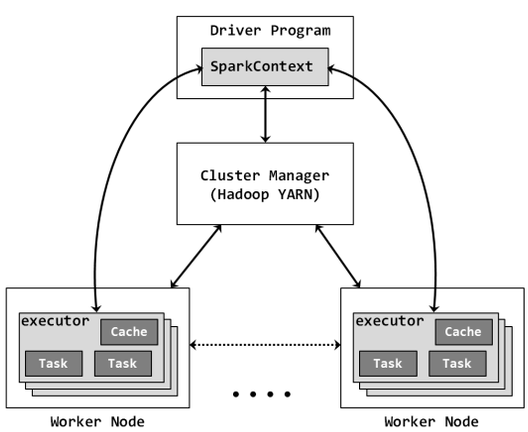

Fig. 1. The Architecture of Topic Extraction System Using Spark

3.2 토픽 추출 프로세스

비정형 텍스트로 구성된 대용량의 데이터를 사전학습 없이 각 토픽(주제)별로 단어들의 가중치로 나타내어 자동으로 추출하는 프로세스는 Fig. 2와 같다. LDA 토픽 모델의 입력 단어는 N-gram 알고리즘을 통하여 생성되며, 단순히 단어 하나를 입력 단위로 생각하지 않고 특정 단어들의 묶음을 한 단위로 보고 처리하였다. 먼저, 인터렉티브한 분석을 위하여 1) 함수적 언어의 절차적 처리와 SQL의 선언적 처리를 동시 만족할 수 있는 SparkSQL[12] 데이터 프레임으로 전체 텍스트 데이터를 읽어 들인다. 다음으로 2) 토픽 추출을 위한 전처리 과정으로 문서 별 단어 분리 작업을 수행한 후, 각 문서 당 분리된 단어들에서 불용어(Stopword) 제거를 실시한다. 그 후, 3) 정제된 단어들로만 구성된 문서에 대하여 N-gram 알고리즘을 수행한다(Fig. 3 참조). 이때, 단어 개수를 달리하여 하나의 단어 묶음으로 처리한다. 이렇게 구분된 단어 묶음을 하나의 word로 하여 구성된 문서들을 LDA 토픽 모델 과정의 입력 값으로 한다.

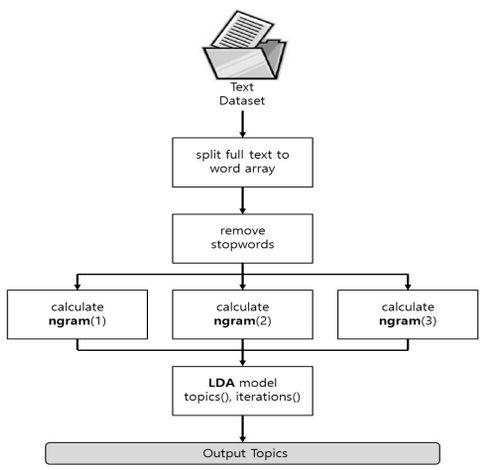

Fig. 2. Data Preprocessing and Topic Extraction Flow

처리된 문서 당 단어 묶음들이 LDA 모델을 통하여 문서 내에서 차지하는 토픽에 따라 가중치가 부여되어 전체 문서들의 잠재적인 특징을 파악할 수 있게 되며, 이를 통하여 대용량 텍스트 데이터에 대한 사전 학습 없이도 전체 문서 내의 주제가 도출될 수 있게 된다. Fig. 3에 본 데이터 처리 주요 과정의 소스코드(Scala 언어 기반)를 나타내었다.

Sample Source in Scala

```
    :
    / / Data    Frame
1)  val    df = sqlContext.read.json("Dataset")
    :
    // Stopword Removal
2)  val    remover = new StopWordsRemover().setInputCol(regexTokenizer.getOutputCol)

    // Call    n-gram
3)  val    ngram = new
    NGram().setN(2).setInputCol(remover.getOutputCol).setOutputCol("ngrams")
    :
    val    pipeline = new Pipeline().setStages(Array(regexTokenizer, remover, ngram))
    val    df1 = pipeline.fit(df).transform(df)
    :
```

Fig. 3. Sample Code for the Proposed Topic Extraction Process

3.3 Spark 에서의 토픽 모델 분산 처리

LDA 토픽 모델의 파라미터 추론 과정을 Fig. 4에 나타내었으며, E-Step은 현재의 파라미터를 이용하여 잠재변수의 기대치(Expectation)를 계산하는 과정이고, M-Step은 E-Step에서 찾아진 기대치를 잠재변수의 관찰값으로 하여 로그 우도를 최대화(Maximization)하는 파라미터를 찾는 과정이다.

Algorithm LDA: pseudo code

```
while relative improvement in loss function do
  E-Step:
  for d = 1 to D do
    repeat
      update document/topic distribution for d
      update topic/word assignments for d
    until convergence
  M-Step:
update topic/word distribution
```

Fig. 4. LDA Topic Model Parameter Inference Algorithm

M-Step을 수행하고 나면, 이전의 파라미터로부터 수정된 새로운 파라미터를 얻게 되고, 이것을 이용하여 다시 잠재변수의 기대치 를 계산하는 E-Step을 수행하게 된다. 이렇게 M-Step과 E-Step을 반복하면서 보다 정확한 파라미터를 추정할 수 있다.

Fig. 5는 텍스트로 구성된 문서 데이터가 Spark에서 분산-병렬 처리되는 LDA 과정으로, 먼저 전체 문서(d1,d2, …, dn)에 대해, 각 문서당 내용을 단어 별로 구분한 후 각각 구성된 단어 별로 잠재변수의 기대치를 계산한다. 실행 초기에는 파라미터 값이 주어지지 않았으므로, 빈도(Frequency)로써 우선 정하고, 이후에 M-step을 수행하여 얻어지는 파라미터 값을 사용하게 된다. 각 Step을 일정 횟수 반복한 후 얻어진 파라미터 값에 의해 확률밀도함수가 결정될 수 있으므로, 이를 이용하여 확률 변수의 기대치를 계산할 수 있게 된다. LDA 수행시에 각각의 입력 문서는 Hadoop 클러스터에 분산 저장되며, 각 Slave 노드에서는 독립적으로 LDA를 병렬 처리한다.

4. 성능 평가 실험

4.1 비정형 텍스트 데이터 셋

실험을 위해 구축된 프로토타입 클러스터는 1대의 Master와 총 6대의 Slave 노드로 구성되어 있다. 실험환경 세부 스펙은 Table 1과 같으며, 클러스터 전체 메모리는 384 GB이며, HDD 용량은 64 TB가 된다.

입력 소셜 댓글[13] 데이터는 크기가 약 1 TB이며, 총 레코드 수는 약 16.5억 개다. 또한 최초 파일 저장 형식은 Key와 Value로 구성된 json 포맷이며, 총 속성 항목은 22개다. 입력 데이터 셋에 대한 전처리 과정을 통해, 서브 항목(Sub Item)이 “Fitness”인 데이터들 중에서 “댓글” 속성에 대해서만 추출하였으며, 이는 “Fitness” 서브 항목의 경우, 토픽 분류가 비교적 확실할 것으로 예상하였기 때문이며, 비 포함된 다른 서브 항목들도 동일한 실험 방식을 적용하여 토픽을 추출할 수 있다.

Table 1. Experiment Environments

구분	Spec.
Data Node (Slave)	CPU: Intel Core i5 or i7(Skylake), 3.2GHz
	Memory: 384 GB (64 GB * 6 nodes)
	HDD: 64 TB = (8 TB * 6 nodes)
O.S.	Ubuntu 14.04 LTS
Big Data Platform	Hadoop 2.7 / Spark 2.0

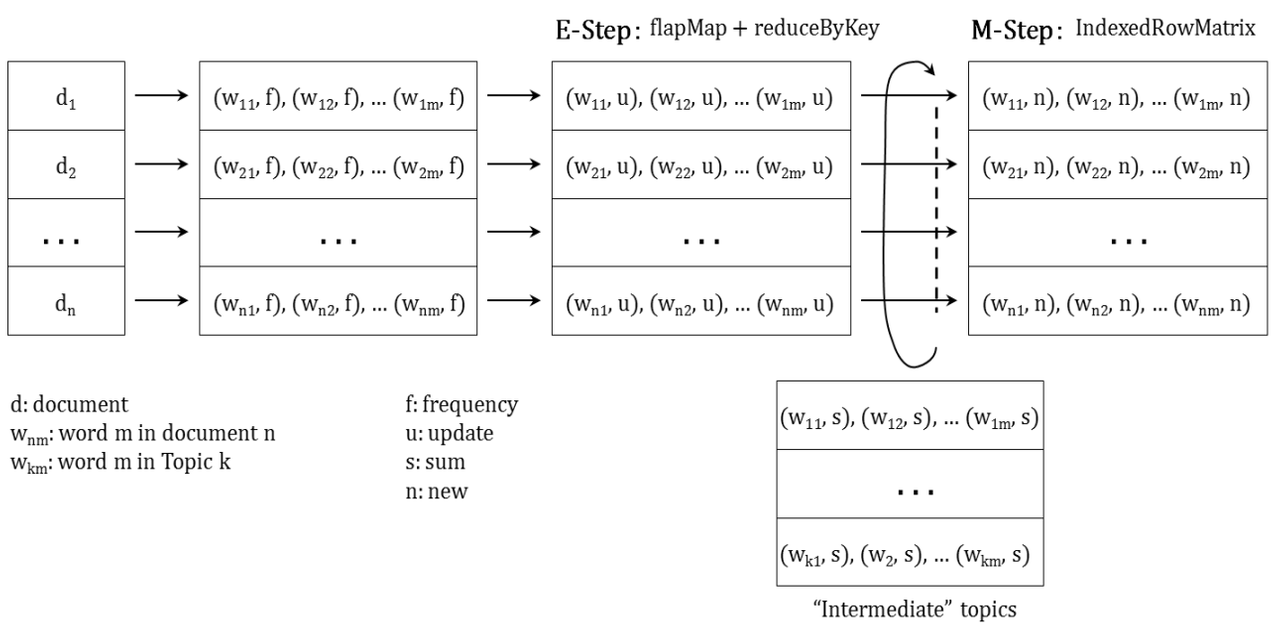

Fig. 5. LDA Calculation Process in Spark

4.2 토픽 추출 결과

실험은 먼저 전처리 과정을 거친 “Fitness” 관련 댓글 내용인 비정형 데이터 셋에 대하여 N-gram 처리하였다. 그 후 LDA 모델을 통하여 토픽을 추출했으며, 토픽 당 단어 개수는 5로, 최대 반복 횟수는 50으로 고정하였다.

Table 2는 댓글을 1-gram 알고리즘에 의하여 처리한 후, LDA 모델을 적용한 결과다. 추출된 결과는 Topic1은 체중 관련 영양소, Topic2는 트레이닝 관련, Topic3는 건강 관련 질의, Topic4는 체육관 관련, Topic5는 트레이닝 종목 관련 등의 내용임을 알 수 있고, 각 Topic 당 해당 단어들의 비중을 알 수 있다.

Table 3은 2-gram 처리를 거친 후 LDA 를 실행했을 때의 토픽 추출 결과를 보여준다. 두개의 단어를 하나의 말 묶음으로 처리하여 하나의 단어처럼 취급한 후 토픽모델에 적용된 것이다. Topic1의 경우 건강을 위한 운동 기구에 관한 내용이며, Topic2의 경우 몸무게를 줄이기 위한 내용 등이다. 이를 통하여, 한 단어로 표현된 1-gram 을 가지고 처리 했을 때보다 두개의 연속 단어 묶음이 토픽의 내용을 보다 세부적으로 나타냄을 알 수 있다.

Table 4는 3-gram처리를 거친후 LDA 를 실행했을 때의 토픽 추출 결과를 보여준다. 이번에는 3개의 연속 단어로 구성된 말묶음을 하나의 단어로 인식하여 토픽 추출을 한 것이다. 표에서 Topic1에서는 “fitness”, Topic2에서는 “youtube” 단어가 속한 3개 연속 단어 묶음들이 나타남을 알 수 있다. 하지만, 두개의 단어로 표현된 2-gram 처리 보다는 토픽별 구분이 모호해 짐으로 인해 단어 묶음이 많다고 해서 토픽 선명도를 높여준다고 볼 수 없음을 알 수 있다. 이는 대부분의 댓글 텍스트가 이용자들의 생활 용어로 구성된 구어체 구문이기 때문에 긴 단어 묶음 사용은 잘 사용하지 않기 때문으로 추정된다.

4.3 추출 시간 비교

다음으로 Fig. 6은 토픽 개수 K에 따른 추출 시간을 비교하였다. 이때, 토픽당 단어 개수는 5로, 최대 반복 횟수는 50으로 고정하였다. 토픽 추출을 위해서는 모든 단어에 대한 정보가 메모리에 로드된 상태에서 처리되어야 한다. Fig. 6에서 토픽 개수의 변화에 따라 시간이 달라짐을 확인 할 수 있다.

Table 2. Results of [Unigram + LDA]

Topic1		Topic2		Topic3		Topic4		Topic5	
term	weight	term	weight	term	weight	term	weight	term	weight
fat	0.0171	want	0.0174	getting	0.0242	gym	0.0191	doing	0.0204
eat	0.0159	know	0.0160	message	0.0228	i've	0.0137	work	0.0146
calories	0.0155	think	0.0153	started	0.0219	going	0.0126	squat	0.0117
muscle	0.0148	training	0.0116	post	0.0210	really	0.0117	bench	0.0106
protein	0.0129	look	0.0114	questions	0.0205	got	0.0082	lbs	0.0102

Table 3. Results of [Bigram + LDA]

Topic1		Topic2		Topic3		Topic4		Topic5	
term	weight	term	weight	term	weight	term	weight	term	weight
bench press	0.0017	lose weight	0.0026	sort new	0.0106	https www	0.0029	feel like	0.0022
imgur com	0.0017	body fat	0.0019	foolish friday	0.0095	wiki index	0.0028	don't know	0.0020
pull ups	0.0014	weight loss	0.0019	fitness comments	0.0074	youtube com	0.0026	it's just	0.0014
http imgur	0.0013	make sure	0.0014	questions concerns	0.0065	www youtube	0.0025	don't want	0.0013
days week	0.0011	losing weight	0.0014	subreddit message	0.0065	com watch	0.0025	don't think	0.0013

Table 4. Results of [Trigram + LDA]

Topic1		Topic2		Topic3		Topic4		Topic5	
term	weight	term	weight	term	weight	term	weight	term	weight
new amp restrict	9.0736	www youtube com	0.0023	link http www	0.0019	fitness wiki index	0.0055	http imgur com	0.0031
posts specific fitness	8.7211	youtube com watch	0.0022	questions answered wiki	0.0012	contact moderators subreddit	0.0054	www bodybuilding com	0.0031
fitness promote discussion	5.7869	com watch v	0.0022	http imgur com	0.0007	performed automatically contact	0.0052	http www bodybuilding	0.0027
specific fitness promote	5.7815	https www youtube	0.0020	faq getting started	0.0006	moderators subreddit message	0.0051	ice cream fitness	0.0025
search q flair	5.7266	medical injury advice	0.0013	feel free make	0.0006	action performed automatically	0.0051	bodybuilding com fun	0.0021

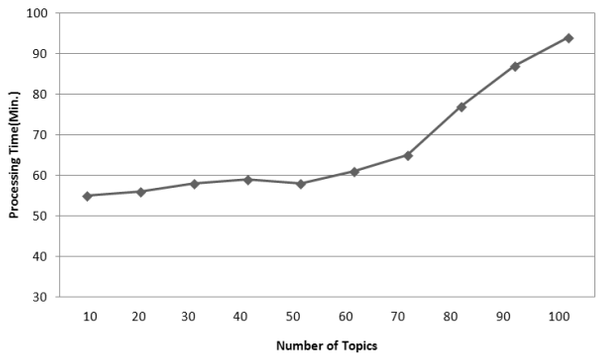

Fig. 6. Processing Time of [Unigram + LDA]

본 실험을 통해, 방대한 양의 비정형 데이터인 소셜 데이터 전체를 읽어 들여 분석하는데 인메모리 방식의 Spark를 이용할 경우 처리 속도에 대한 향상이 가능함을 확인하였고, 토픽 모델의 입력 값을 N-gram에 의하여 조정함으로써 문서 전체의 이해를 다양하게 선택할 수 있었다. 참고로, 정형 데이터를 주로 처리하는 관계형 데이터 모델에서는 TB급 용량의 레코드를 대상으로 한 간략한 SQL 질의 수행에 수 시간이 소요되는 것이 현실이다.

5. 결론 및 후속 연구

본 논문에서는 대용량 비정형 텍스트 데이터에 대한 토픽 추출 시스템을 설계 및 구축하였으며, N-gram을 이용하여 단어 묶음을 구분한 후, LDA 토픽 모델을 통한 토픽 추출 및 문서 데이터의 의미를 분석하였다. 제안된 시스템에서는 Hadoop과 Spark(분산 인메모리 처리 프레임워크) 기반으로 데이터 저장 및 연산 클러스터를 구성하였기 때문에, 데이터 처리시 디스크 접근을 최소화하고 모든 중간 산출 결과를 메모리 상에서 처리하므로 고속의 처리 성능을 얻을 수 있다. 실험에서는 약 16억 개의 레코드로 구성된 1TB 소셜 댓글 데이터를 읽어 들여 전체 데이터에 대한 전처리 과정과 토픽 분석을 수행하였으며, 분석 시간 단축을 확인하였다.

후속 연구로는 동적으로 발생하는 소셜 스트림 텍스트 데이터에 대한 처리를 돕기 위한 연산 플랫폼 설계와 이를 위한 다양한 토픽 분석 알고리즘에 대한 연구가 필요하다고 본다.

References

[1] D. M. Blei, “Probabilistic Topic Models,” *Communication of the ACM*, Vol.55, No.4, pp.77-87, 2012.

[2] P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai, “Class-Based N-gram Models of Natural Language,” *Computational Linguistics*, Vol.18, No.4, pp.467-479, 1992.

[3] V. K. Vavilapalli and A. C. Murthy, et al., “Apache Hadoop YARN: Yet Another Resource Negotiator,” in *Proceedings of the 4th annual Symposium on Cloud Computing ACM*, No.5, pp.1-16, 2013.

[4] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, “Spark: Cluster Computing with Working Sets,” in *HotCloud*, p.10, 2010.

[5] D. M. Blei, A. Y. Ng, and M. J. Jordan, “Latent Dirichlet Allocation,” *Journal of Machine Learning Research*, Vol.3, pp. 993-1022, 2003.

[6] T. Hofmann, “Probabilistic Latent Semantic Indexing,” in *Proceedings of the 22nd Annual International SIGIR Conference on Research and Development in Information Retrival*, pp.50-57, 1999.

[7] J. Park and H. Oh, “Distributed Online Learning for Topic Models,” *Communications of the Korean Institute of Information Scientists and Engineers*, Vol.32, No.7, pp.40-45, 2014.

[8] K. Shvachko, et al., “The Hadoop Distributed File System,” in *Proceedings of the 26th IEEE Transactions on Computing Symposium on Mass Storage Systems and Technologies*, pp. 1-10, 2010.

[9] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. Franklin, S. Shenker, and I. Stoica, “Resilient Distributed Datasets: A Fault-tolerant Abstraction for In-memory Cluster Computing,” *NSDI*, Apr., 2012.

[10] J. Dean and S. Ghemawat, “MapReduce: Simplified Data Processing on Large Clusters,” in *Proceedings of the 6th Symposium on Operating System Design and Implementation*, pp.137-150, 2004.

[11] K. Park, C. Baek, and L. Peng, “A Development of Streaming Big Data Analysis System Using In-memory Cluster Computing Framework: Spark,” *LNEE*, Vol.393, pp.157-163, 2016.

[12] M. Armbrust, R. S. Xin, C. Lian, Y. Huai, D. Liu, J. K. Bradley, X. Meng, T. Kaftan, M. J. Franklin, A. Ghodsi, and M. Zaharia, “Spark SQL: Relational data processing in Spark,” in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pp. 1383-1394, 2015.

[13] https://www.reddit.com/wiki/ko/reddiquette.

박 기 진

e-mail : kiejin@ajou.ac.kr

1989년 한양대학교 산업공학과(공학사)

1991년 POSTECH 산업공학과(공학석사)

1991년～1997년 삼성종합기술원/삼성전자 연구원

2001년～2002년 한국전자통신연구원 연구원

2001년 아주대학교 컴퓨터공학과(공학박사)

2002년～2004년 안양대학교 컴퓨터학과 교수(학과장)

2004년～현 재 아주대학교 산업공학과/융합시스템공학과 교수(학과장)

2009년～현 재 대한산업공학회 이사/편집위원장/총무이사

2010년～2011년 Visiting Professor, Rutgers, The State University of New Jersey, USA

관심분야 : Big Data Processing, Intelligent System, Social Data Analysis