ISSN 1598-0170 (Print)
ISSN 2287-1136 (Online)
http://www.jksii.or.kr

트위터 트랜딩 토픽을 이용한 HBase 기반 자동 요약 시스템

HBase-based Automatic Summary System using Twitter Trending Topics

이 상 훈[1] 문 승 진[2*]
Sanghoon Lee Seung-Jin Moon

요 약

트위터는 사용자들이 140개 정도의 문자들로 이루어진 짧은 메시지를 웹에 포스팅 할 수 있도록 제공하는 인기 있는 소셜 미디어 플랫폼이다. 해시태그는 이러한 트위터 사용자들이 특정한 주제에 대해서 토론을 하거나 높은 트랜딩을 가지는 이슈를 나타내고자 할 때 사용하는 특정한 단어나 두음문자이다. 하지만 동일한 해시태그를 포함하는 포스트들은 관련 있는 문장이 아닌 시간 순서에 의해서 처리되기 때문에, 처음 사용자가 그 해시태그와 관련된 내용을 이해하기 위해서는 다른 불필요한 내용까지 읽어야 하는 어려움이 있다. 본 논문에서는, 이러한 문제점을 해소하기 위한 HBase 기반 자동 요약 시스템을 제안한다. 제안된 시스템은 트위터 API에서 제공하는 스트리밍 데이터를 HBase에 저장한 후 퍼지 시스템과 접목하여 자동 요약 방법을 시행하였다. 이를 통해서 해시태그를 포함한 포스트내의 중복된 내용을 제거하고, 각 포스트들의 중요도를 계산해서 사용자가 트랜딩 토픽내에 관련 있는 주제에 쉽게 접근할 수 있도록 하였다.

☞ 주제어 : 트위터 트렌딩 토픽, 자동 요약 시스템, 퍼지 이론, HBase, NoSQL, 트위터 API

ABSTRACT

Twitter has been a popular social media platform where people post short messages of 140 characters or less via the web. A hashtag is a word or acronym created by Twitter users to open a discussion about certain topics and issues that have a very high percentage of trending. Since the hashtag posts are sorted by time, not relevancy, people who firstly use Twitter have had difficulty understanding their context. In this paper, we propose a HBase-based automatic summary system in order to reduce the difficulty of understanding. The proposed system combines an automatic summary method with a fuzzy system after storing the streaming data provided by Twitter API to the HBase. Throughout this procedure, we have eliminated the duplicate of contents in the hashtag posts and have computed scores between posts so that the users can access to the trending topics with relevancy.

☞ keyword : Twitter trending topics, Automatic summary system, Fuzzy theory, HBase, NoSQL, Twitter API

1. 서 론

트위터는 1억 명의 사용자들이 매일 2억개의 짧은 메시지를 주고받는 가장 인기 있는 마이크로 블로깅 사이트 중에 하나로, 2009년 이후 여러 소셜 매체 중 하나로써 사람들에게 많은 영향을 끼쳐오고 있다. 최근에, 사용자가 직접 포스트내에 # 사인을 추가해서 생성한 해시태그는 트위터에서 정보를 공유하는 하나의 수단으로 인식되어 오고 있고, 이러한 해시태그를 사용해서 사람들은 특정한 이슈와 주제에 대해서 실시간으로 토론을 할 수 있게 되었고, 또한 쉽게 공통된 주제에 대해서 정보를 공유할 수 있게 되었다. 하지만 이러한 해시태그를 사용한 트랜딩 이슈들은 동일한 해시태그를 포함하는 포스트의 내용이 달라지면 그 전까지 이슈가 되었던 내용들을 파악하기 힘든 단점이 있는 뿐만 아니라, 포스트들이 시간에 의해 업데이트가 되기 때문에 처음 그 해시태그에 대한 이슈를 알고자 하는 사람들은 그 전 내용을 파악하기 위해 다른 여러 포스트도 읽어봐야 하는 수고를 하게 된다.

자동 요약 기술은 어떤 문서에서 내용이 중복되는 문장을 제거하고, 각 문장에 중요도를 부여함으로써 사용자가 쉽게 그 문서의 내용을 파악하는데 장점을 가지고 있다. 전통적으로 Document Understanding Conference (DUC)[1]와 같이 전문가들이 미리 요약을 해둔 데이터를

[1] Department of Computer Science, Georgia State University, Atlanta, 30303, USA
[2] Department of Computer Science, University of Suwon, Gyeonggi-do, 445-743, Korea
* Corresponding author (sjmoon103@hotmail.com)
[Received 20 June 2014, Reviewed 09 July 2014, Accepted 01 August 2014]

가지고 실험 평가를 해왔고, 평가된 시스템의 성능을 더욱 객관적이고 정확하게 하기위한 많은 노력들이 이루어져 왔다. 최근 트위나 페이스북과 같은 마이크로 블로깅 사이트내의 포스트들에 대해서 이러한 자동 요약 기술을 접목하고자 하는 움직임이 많았는데, Sharifi[2]은 이러한 자동 요약 기술을 마이크로 블로깅 사이트에 사용하여, 문장내 구문에 기반한 자동 요약 알고리즘을 제안했고, 이후 Inouye[3]는 Sharifi의 방법을 기존의 여러 자동 요약 방법들(MEAD[4], LexRank[5], 그리고 TextRank[6])과 비교 분석을 하여 그 관심을 나타내었다.

하지만 이러한 방법들은 포스팅 시간에 대한 고려 없이 오직 순수한 문장만을 사용해서 연구하였기 때문에, 시간의 흐름에 따른 포스트 내의 정보를 나타내지 못하였고, 해시태그에 대한 고려가 없었기 때문에 실제 트랜딩 주제에 대한 정확한 정보를 나타내기에는 한계를 가지고 있었다. 또한 요약을 실행할 때 실행시간에 대한 고려 없이 순수한 요약 방법에만 관심을 두고 있었기 때문에 실시간으로 들어오는 데이터에 대한 요약을 시행할 때 그 한계를 가지고 있었다.

본 논문에서는 해시태그를 사용해서 특정 트랜딩 주제를 선별해서 포스팅 시간에 따른 자동 요약 시스템을 제안하고 요약 방법의 실행시간에 대해서도 고려한다. 본 논문이 기여하는 부분은 크게 두 가지로 볼 수 있다. 첫째, 포스트내에 해시태그를 사용한 특정 트랜딩 토픽을 자동 요약하여 사용자에게 쉽게 그 트랜딩 토픽에 접근하도록 하였다. 이를 위해서 실시간으로 업데이트 되는 데이터 정보를 처리하기 위해서 HBase내 timestamp를 이용하였고, 기존의 요약방법에서 주로 사용된 TF-IDF 방법, 그리고 포스트내 길이 정보를 함께 처리하기 위해서 퍼지 시스템을 이용해 최종적으로 요약문을 도출하였다. 둘째, HBase 내의 테이블 의 장점을 이용해서 자동 요약 시간의 단축을 시도하였다. HBase테이블은 재사용이 가능하기 때문에 HBase 테이블을 사용했을 때와 사용하지 않았을 때를 비교하여 실험을 하였다.

본론의 1절에서는 HBase에 대한 간단한 설명과 함께 제시된 자동 요약 시스템의 데이트 처리 과정에 대해서 설명할 것이다. 본론의 2절에서는 퍼지 시스템에 어떻게 제시된 요약 방법을 접목하였는지 기술할 것이고, 3장에서는 실험과 실험결과에 대해서 설명하고. 마지막으로 4장에서는 결론 및 향후 연구과제에 대해서 설명하도록 한다.

2. 본 론

2.1 HBASE 기반 자동 요약 시스템

이번 장에서는 HBase 의 배경에 대해서 간단히 설명한 후, 본 논문에서 사용된 HBase 테이블 스키마와 함께 제안된 자동 요약 시스템의 설계에 대해서 설명하도록 한다.

2.1.1 HBase의 배경설명

최근 빅 데이터에 대한 관심과 요구의 증가로 인해서 다양한 영역에서 방대한 데이터를 처리하는 기술이 많은 관심을 받고 있다. 특히 구글은 구글 파일 시스템[7], 맵리듀스[8], 빅테이블[9] 등 다양한 신 기술들을 제시했고, 이러한 기술들은 종종 NoSQL 시스템으로 알려져 왔다. NoSQL은 전통적인 관계형 데이터베이스 보다 훨씬 더 대용량의 데이터를 저장할 수 있고, 분산형의 구조를 갖기 때문에, 데이터를 여러 개의 노드에 분산해 저장하고, 분산시에 데이터를 복제해서 특정한 노드에 장애가 발생했을 때에도 데이터의 유실이나 서비스 중지가 없는 형태의 구조를 가지고 있어 여러 곳에서 사용되고 있는 추세이다. 본 논문에서 사용되는 NoSQL은 HBase에 기반하고 있다. 다른 NoSQL과 비교해서 HBase의 경우 Search engine 이나 Log 데이터를 분석하는데 사용되어 왔고, 트위터와 같이 실시간으로 계속 업데이트가 되는 텍스트를 처리하는데 HBase가 주로 사용되어왔고, 또한 본 시스템은 Hadoop 기반으로 Map/Reduce 작업을 처리하기에 가장 알맞은 NoSQL이 HBbase라고 알려져 왔기 때문에 HBase를 선택하여 사용하였다.

HBase는 이러한 NoSQL 데이터베이스 중에 하나로, 최근 구조화되지 않은 방대한 양의 데이터를 처리하는데 널리 사용되고 있는데, 페이스북은 메시지들을 저장하기 위해서 HBase를 사용하고 있고, 트위터 또한 모든 트윗들을 저장하는데 사용하고 있다. 기본적으로 HBase 는 클러스터 상에서 운용되도록 설계되었고, 클러스터의 각 노드는 데이터를 저장할 수 있는 공간이나 캐시 그리고 데이터 처리 등을 제공한다. HBase 의 장점은 대단히 유동적이라는데 있다. 즉 어떠한 노드도 특별하지 않기 때문에, 한 노드가 없어지더라도 다른 노드로 대체할 수 있는 장점이 있다. HBase에는 세 가지 모드가 있는데, Standalone 모드, Pseudo-distributed 모드, 그리고 Full-distributed 모드이다. Standalone 모드에서 HBase는 오직

하나의 자바 프로세스에서만 실행이 된다. 반면에 Pseudo-distributed 모드에서 하나의 노드는 많은 자바 프로세스들 상에서 실행 될 수 있다. Standalone 모드나 Pseudo-distributed 모드와는 다르게 Full-distributed 모드에서는 실제 많은 노드들을 가지고 있는 상태에서 HBase가 클러스터를 통해 실행되게 된다. 본 논문에서 제안된 시스템은 시스템 환경의 제약으로 인해서 Pseudo-distributed 모드를 사용하기로 한다. 하지만 추후에 실제 많은 노드를 사용할 수 있는 환경이 갖추어 진다면 Full-distributed로 확장할 것이다.

2.1.2 HBase에서 테이블 설계

HBase에서 테이블은 row와 column으로 구성된다. 각 Row는 rowkey에 의해 구분이 되고 column은 column family로 그룹이 지어진다. 관계형 데이터베이스와는 다르게 HBase은 테이블 상에 각 셀들이 version 이 된다. 즉 셀이 테이블에 삽입 될 때 마다 HBase에 의해 할당된 timestamp가 시간을 체크해서 각 row를 구분한다.

(표 1) HBase 테이블 스키마 예
(Table 1) HBase table schema example

Rowkey	ColumnFamily	Column	Timestamp	Value
UserID/time	UserInfo	Name	135658526	example
UserID/time	UserInfo	Post	135658526	Hello, #everyone
UserID/time	UserInfo	Hashtag	135658526	everyone
Hashtag	Hashtag_Count	Count	135658526	1

Table 1은 HBase테이블 스키마의 한 예를 보여준다. 모든 Column family 멤버는 공통의 prefix를 갖는다. 예를 들어, column에서 Name, Post, 그리고 Hashtag 는UserInfo의 멤버가 되는 것이다. Column family 는 테이블 스키마를 정의하기에 앞서 반드시 명기되어야 하지만, 새로운 Column family 멤버는 요구에 의해 추가될 수 있다. 예를 들어, 새로운 column 인 Hashtag 가 업데이트 될 때, Column family 가 해당 테이블에 존재 하는 한 사용자의 요구에 의해 추가될 수 있다. 또한 테이블이 증가함에 따라, 영역의 수도 증가한다. 여기서 영역은 HBase 클러스터 상에서 분산 처리 되는 유닛이다. 이러한 방법으로, 한 서버 상에서 데이터가 너무 많아지면 서버들의 클러스터 상으로 옮겨간다.

본 논문에서 제안된 시스템은 이러한 Hbase테이블 스키마의 규칙을 따른다. Table 2는 본 논문에서 사용된 Column family의 이름과 설명을 보여준다. 사용된 Column family는 총 다섯 개로, 각각 UserInfo, HashTag_Count, HashTag_Contens, HashTag_Vectorized, 그리고 TwoStep_Summaries으로 구분하였다. Column family는 column을 포함하고 있고, row는 rowkey에 의해 구분된다. 본 논문에서 사용된 rowkey는 name과 time의 조합이다. 예를 들어, rowkey인 “hashtag name / time”는 hashtag가 트위터로부터 수집된 사용자 코드와 시간을 나타낸다.

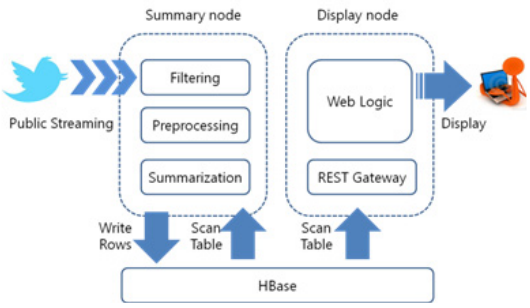
(표 2) 자동 요약 시스템에 사용된 column families
(Table 2) Column families used for automatic summary system

Column Families	Descriptions
UserInfo	Receive information from Twitter and spread the information to other column families. Columns: username, id, location, post, and hashtags
HashTag_Count	Count total number of every hashtag. Column: count
HashTag_Content	Store posts of top hashtags. Column: content
HashTag_Vectorized	Store vectorized posts. Each post is used to compute summary weights. Column: vector
TwoStep_Summaries	Represent summary contents. Column: summary

2.1.3 HBase기반 자동 요약 시스템

본 논문에서 제안된 자동 요약 시스템은 Summary node, Display node, 그리고 HBase 총 세 부분으로 구성이 된다. 그림 1 에서 보는 것과 같이 Summary node는 트위터로 부터 스트리밍 데이터를 받아서 요약화 과정을 거친 후 HBase와 정보를 교환한다. 요약화 과정은 트위터로 부터 데이터 정보가 처음 Summary node에 도착할 때 필터링 과정을 거쳐야 하는데, 그 이유는 데이터 정보가 모든 자연 언어를 포함하기 때문에 사용할 언어를 선별할 필요가 있기 때문이다. 본 논문에서는 가장 많은 부분을 구성하고 있는 언어인 영어만이 필터링이 될 수 있도록 하였다. 필터가 된 데이터 정보는 Column family 인

UserInfo에 저장되고, 한번 UserInfor에 정보가 도착하면 수집 된 정보 내에 동일한 해시태그가 존재한다면 그 수를 계산하여 HashTag_Count에 업데이트 하게 된다. 전처리 과정과 요약 과정은 이 과정을 거친 후 수행하게 된다. 이 과정에 대해서는 3장에서 자세히 이야기 하도록 한다.



(그림 1) 제안된 시스템의 데이터 흐름도
(Figure 1) Data flow of proposed system

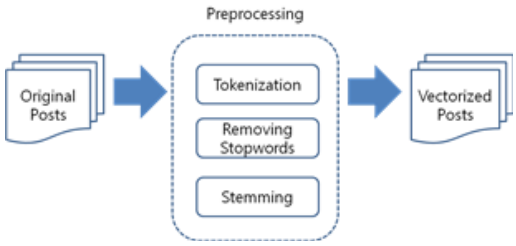
HBase는 REST API를 제공하는데 이는 분산 시스템을 위한 소프트웨어 아키텍처를 말한다. REST 는 최근 급격하게 사용이 많아진 웹 API중에 하나로서 XML 혹은 JSON을 지원하고, HBase에 있는 모든 테이블들의 리스트를 검색해서 XML 형식으로 보여준다. 하지만 사용자가 HBase REST 게이트웨이를 통해서 직접 요약문을 볼 수 없기 때문에, 본 눈문에서는 사용자와 REST 게이트웨이를 연결 할 수 있는 서블릿 프로그램을 만들었다. REST API를 사용하기 위해서는 먼저 HBase 데몬들이 실행되어야 하는데, NameNode, SecondaryNameNode, DataNode, JobTracker, TaskTracker, 그리고 Hmaster라고 불리는데 이러한 데몬들은 각각 Hadoop과 HBase가 정해놓은 순서에 따라 시작이 된다. NameNode는 파일 시스템에서 모든 파일의 트리 구조를 유지하는 역할을 하는데, 클라이언트가 파일을 위치하고자 원할 때 그 NameNode는 파일시스템에 데이터를 저장하는 DataNode를 리턴해서 그 요청에 답장을 준다. SecondaryNameNode는 주기적으로 체크포인트를 수행한다. 즉 주기적으로 현재 NameNode 이미지를 다운로드해서 로그파일을 수정한다. 그래서 NameNode가 일시적으로 작동하지 않는다고 해도, DataNode를 다운 시킬 필요가 없이 해당 NameNode만 재시작을 하면 된다. JobTracker는 Hadoop내에 위치한 서비스 중에 하나로 클라이언트가 JobTracker에 job을 보내고 그 JobTracker들은 데이터의 위치를 결정하기 위해서 NameNode와 서로 통신을 한다. TaskTracker는 task를 받아서 그 task의 진행상황을 추적한다. 마지막으로 HMaster는 HBase를 위한 마스터 서버이다. 일단 한번 모든 데몬들이 올바르게 작동하면, REST gateway는 시작 준비를 하게 되고 Oracle Web Logic Server 를 통해서 요약 결과를 사용자에게 보여주게 된다.

2.2 퍼지 시스템을 이용한 트위터 요약 방법

이번 장에서, 우리는 본 논문에서 사용된 퍼지 시스템을 이용한 요약방법에 대해서 기술 하도록 한다. 일반적으로 문서 요약은 문서의 일관성을 유지시키면서 중복을 피하는 압축된 정보를 생산하는데 목적을 가지고 있다. 이러한 문서 요약은 크게 두 가지 방법으로 연구되고 있는데 요약문을 만들 때 원본 문장을 변형 없이 그대로 사용하는 추출식 요약 방법과 원본 문장을 변형하는 추상적 요약 방법이 있다. 본 논문에서는 현재 가장 많이 사용하고 있는 요약 방법인 추출식 요약 방법에 근거하고 있다.

2.2.1 전처리 단계

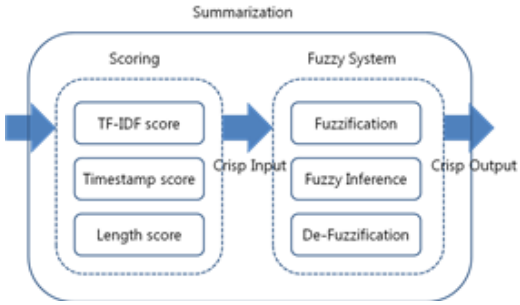
트위터로 부터 스트리밍 데이터가 Summary node에 들어오게 되면, 데이터는 필터링 과정을 거쳐 전처리 단계로 들어가게 된다. 그림 2는 본 논문에서 사용된 전처리 단계를 보여준다. 필터링을 거친 원본 포스트는 토큰화 작업을 거치게 되는데 이것은 입력된 텍스트를 단어로 나누는 작업을 말한다. 이 과정에서 선택된 원본 포스트들은 토큰화가 된다. 이 후에, 각 단어들 중에 대명사나 전치사 같은 문장에 크게 영향을 주지 않는 단어들을 제거하게 되는데 이를 StopWord 제거 단계라고 부른다. 일반적으로 이 단계를 통해서 요약문의 성능을 높이게 된다. Stemming은 각 단어에서 어근을 제외하고 나머지를 제거하는 단계이다. 이 단계에서 우리는 현재 가장 많이 쓰이는 알고리즘인 Porter Stemmer[10]를 사용하였다. 또한 효율적인 요약문을 위해서는 중복된 문장을 제거하는 것이 필요한 작업으로 인식되어 왔다. 본 논문에서는 이러한 중복된 문장을 제거하기 위해서 서로 다른 문장사이의 유사도를 계산해서 유사도가 정해서 Threshord 보다 크다면 새로운 문장을 무시되고, 기존의 문장은 또 다른 새로운 문장과 비교하도록 하였다. 유사도 계산을 위해서 Cosine 유사도 방법이 사용되었다. 이렇게 전처리 과정을 거친 포스트들은 HBase 내 Column family 인 HashTag_Vectorized 에 저장되게 된다.



(그림 2) 전처리 단계
(Figure 2) Preprocessing steps

2.2.2 퍼지 시스템을 이용한 요약

퍼지 이론이란 L.A. Zadeh[11]에의 도입된 퍼지 집합의 사고방식에 기초를 둔 것으로 자연 언어의 애매함을 정량적으로 표현하기 위해서 도입된 이론이다. 퍼지 시스템에서 입력 값은 퍼지 집합이라고 불리는 소속 함수들 중 하나의 집합에 매핑이 되고, 그 입력 값은 퍼지화라고 불리는 단계 동안 퍼지 값으로 변환된다. 그 퍼지 값은 퍼지 추론 이라고 불리는 과정에서 지정한 규칙에 의거해서 출력 값으로 변환되는데, 이때 퍼지 시스템은 역퍼지화라고 불리는 과정을 거쳐서 처음의 입력 값에 대한 출력 값을 보여준다. 본 논문에서는 자동 요약 시스템의 성능을 높이기 위해서 이러한 퍼지 이론을 접목하였다. 즉 동일한 해시태그를 포함하는 포스트내에서, 각 문장의 중요도를 측정하는 방법을 세 가지로 구분하였고, 이를 각각 하나의 퍼지 집합으로 정의하였다. 그림 3은 제안된 요약 방법의 작업 흐름도 보여준다. 전처리 작업을 거친 각 문장들은 Term Frequency-Inverses Document Frequency (TF-IDF), Timestamp, 그리고 Length 에 근거해서 정규화를 거친 후, 퍼지 시스템을 거쳐 출력 값을 나타내게 된다.



(그림 3) 퍼지 시스템을 이용한 요약 작업 흐름도
(Figure 3) Summary work flow using fuzzy system

본 논문에서는 각 퍼지 집합에 대한 중요도를 다음과 같이 정의한다. 일반적으로 퍼지 집합은 불확실한 의미를 표현하는 집합으로서 멤버쉽 함수, 즉 퍼지 집합에 속하는 정도를 $\mu_s(w) \in [0,1]$ 로 표현해 왔다. 본 논문에서는 이러한 퍼지 집합의 중요도를 각 Post의 중요도를 계산하는데 사용하도록 한다.

$$Score = \{(w, \mu_s(w)) \mid w \in W\} \quad (1)$$

여기서, Score는 하나의 퍼지 집합의 중요도를 말한다. W는 Score에 대한 집합을 나타내고, $\mu_s(w) \in [0,1]$ 는 소속함수를 나타낸다.

TF-IDF는 텍스트 마이닝에서 가장 널리 알려진 문서의 중요도를 측정하는 방법으로 TF는 문장내의 단어의 빈도수를 나타내고, IDF는 단어가 문서 내에서 나타나는 빈도수를 역으로 한 값으로 TF와 IDF를 곱한 값으로 그 단어가 특정 문서 내에서 얼마나 중요한지를 나타낼 수 있다. 여기서 우리는 TF-IDF를 다음과 같이 정의한다.

$$TF \cdot IDF = TF(w,s) \times IDF(w) \quad (2a)$$

$$IDF = 1 + \log \frac{total\,Content}{num\,Content + 1} \quad (2b)$$

$$Score_{TF \cdot IDF} = \frac{\sum_{k=1}^{n} TF_k \cdot IDF_k}{Max(\sum_{k=1}^{n} TF_k^1 \cdot IDF_k^1)} \quad (2c)$$

여기서 TF(w,s)는 포스트에 타나난 단어 w의 수를 나타내고, IDF(w)는 단어 w에 대한 역 문장 빈도 수를 나타낸다. totalContent는 포스트의 총 수를 나타내고, numContent는 단어w 가 나타나는 포스트의 총 수를 나타낸다.

트위터내에 해시태그를 포함하는 포스트들은 시간대별로 다른 의미를 가지고 있다. 예를 들면, 2013년 5월 15일 부터 19일까지의 해시태그 #Oklahoma를 포함한 포스트의 주된 내용은 대학 미식축구와 관련된 내용이었지만, 2013년 5월 19일부터 해시태그 #Oklahoma를 포함한 포스트는 Oklahoma 도시에 불어온 엄청난 토네이도에 대한 내용으로 메인을 이루었고, 2013년 5월 22일 부터는 도네이션 캠페인이나 토네이도의 희생자들에 관한 내용이 포스트의 대다수를 이루는 등. 시간이 지남에 따라 그 내용 또한 달라진다. 이러한 이유로, 본 논문에서는 포스트내의 내용이 시간에 따라 변화한다고 가정하였다. 시

간에 대한 정보는 2.2장에서 언급했듯이 HBase테이블에서 각 row는 cell에 값이 도착할 때 마지막 시간을 timestamp로 저장하기 때문에, 우리는 이 정보로부터 시간에 대한 정보를 추출할 수 있다. 그림4에 보듯이 본 논문에서는 이러한 timestamp를 하나의 퍼지 집합으로 정의하였다. 우리는 timestamp를 아래와 같이 정의한다.

$$Score_{timestamp,0\,to\,1} = \frac{X_{timestamp} - X_{Min}}{X_{Max} - X_{Min}} \quad (3)$$

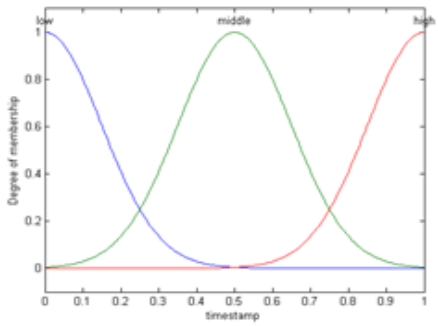
여기서 $X_{timestamp}$는 각각의 timestamp를 나타내고, X_{Min}와 X_{Max}는 각각 모든 timestamp 중에 최소값과 최대값을 나타내고, 최종 $Score_{timestamp,0\,to\,1}$는 0과 1사이의 값을 나타낸다.

본 논문에서는 포스트의 길이도 내용의 중요도에 영향을 끼친다고 가정하였다. 트위터의 경우 최대 140개의 문자까지 적을 수 있기 때문에 사용자들은 그들의 의견을 짧고 간결하게 나타내는 경향이 있다. 즉 포스트 내에 단어가 많아질수록 길이의 중요도는 감소하고, 단어가 적어질수록 길이의 중요도는 높아진다.

$$Score_{length} = \frac{L}{Max(L)} \quad (4)$$

여기서 L은 해당 포스트 내에 존재하는 단어의 총 수를 나타내고, $Max(L)$는 수집된 포스트 중에 단어 개수의 최대값을 나타낸다.

퍼지 추론은 퍼지 논리를 사용해서 주어진 입력 값을 출력 값으로 매핑하는 작업이다. 본 논문에서는 퍼지 추론을 계산하기 위해서 최소 t-norm을 결정하기 위해서 다음과 같이 정의한다.



(그림 4) Timestamp 퍼지 집합
(Figure 4) Timestamp fuzzy set

$$\mu_{s1}\,AND\,\mu_{s2} = Min\,\{\mu_{s1}, \mu_{s2}\} \quad (5)$$

여기서 μ_{s1}와 μ_{s2}는 퍼지 집합이다.

퍼지 추론을 계산하면서 사용된 규칙은 교집합을 사용하였다. 즉 사용된 규칙은 다음과 같다. "IF TF-IPF 의 중요도가 매우 높고 AND timestamp의 중요도가 높고 AND 길이의 중요도가 높다면 THEN 그 중요도는 매우 높다." 마지막으로 역퍼지화를 위해서 최대-최소 추론에 근거한 centroid defuzzification 방법을 사용하였고, 다음과 같이 정의하였다.

$$Center\,of\,Gravity = \frac{\sum_{w=1}^{n}(\mu_s(w) \times L_w)}{\sum_{w=1}^{n}\mu_s(w)} \quad (6)$$

여기서 $\mu_s(w)$는 중요도의 상태를 나타내고 L_w은 그 상태의 위치 값을 나타낸다.

3. 실 험

3.1 실험평가

이번 장에서는, 제안된 HBase 기반 자동 문서 요약 시스템을 실험을 통해서 기존의 TF-IDF에 근거한 방법과 비교한다. 첫째로 우리는 자동 문서 요약 기법에서 가장 많이 알려진 Recall Oriented Understudy for Gisting Evaluation (ROUGE)[12]을 사용해서 실험을 하도록 한다. 다시 말해서, ROUGE 메트릭을 사용해서, 퍼지 시스템을 이용한 자동 문서 요약 방법과 기존의 TF-IDF 만을 이용한 방법과 비교하도록 한다. 두번째로, 우리는 제안된 시스템의 실행시간을 비교하기 위해서 HBase 를 사용한 방법과 아닌 방법을 비교하도록 한다. 본 장은 크게 두 부분으로 구성이 되는데, 먼저 실험에 사용 될 데이터 집합과 ROUGE 메트릭에 대해서 설명하고, 그 다음 실험결과에 대해서 설명하도록 한다.

3.1.1 실험 평가 방법

먼저 실험을 위해서 우리는 오후 12시 부터 1시까지 7일 동안 트위터로 부터 스트리밍 데이터 정보를 받았다. 이렇게 데이터 정보에 시간을 정한 이유는 첫째, 아침이나 저녁 이후의 경우에는 너무 많은 광고성 글이 전달이 되었기 때문에 시간을 점심으로 정하였고, 두 번째 이유

는 트위터의 스트리밍 데이터 정보는 매 0.1초 간격으로 전달되기 때문에 평가를 위한 데이터 정보로 충분했기 때문이다. 평가를 위해서 우리는 총 7시간 동안 35,359 개의 해시태그의 정보를 얻었고, 그중에 가장 많은 포스트를 포함하는 해시태그 50개를 선별하였다. 이렇게 50 개를 선별한 이유는 가능한 많은 포스트를 포함하는 해시태그를 요약해서 평가하고자 했기 때문이다. 수집된 포스트의 수는 총 75,317 개로, 35,359개의 해시태그를 포함하고 있었고, 그중 50개의 해시태그를 포함하는 포스트의 수는 2,512개 였다.

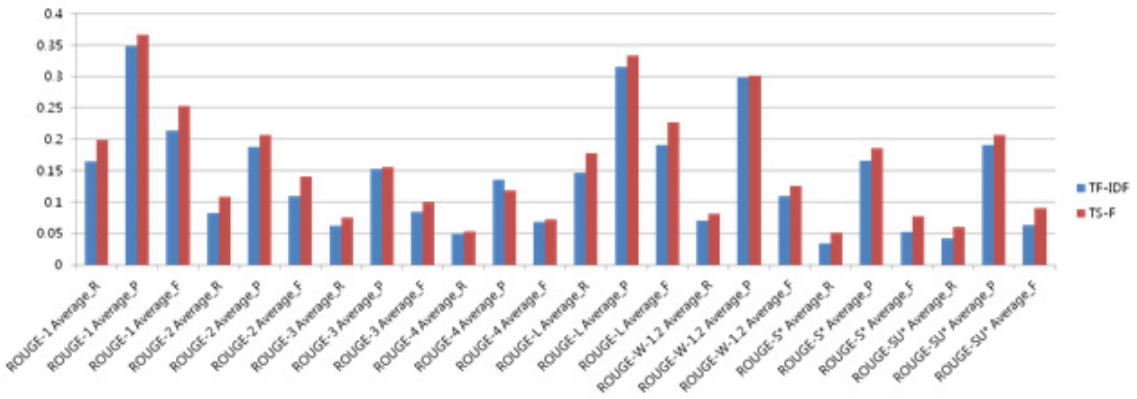
실험평가를 위해서, 앞으로 TF-IDF, timestamp, 그리고 length 의 조합을 사용한 본 논문에서 제안한 요약 방법을Twitter Summarization using Fuzzy technique (TS-F) 라고 부르기로 한다. 그리고 ROUGE 매트릭을 사용될 Model Sumary를 위해서 두 명의 학생에게 동일한 해시태그를 가지고 있는 포스트들을 네 개의 문장으로 요약해 달라고 요청하였다. 이 Model summary 는 제안된 System summary와 비교할 것이다. 또한 전통적인 방법인TF-IDF와 제안된 TS-F를 비교하기 위해서TF-IDF 방법을 사용한 System summary 역시 추출하였다.

ROUGE 매트릭은 현재 요약 평가를 위한 방법으로 가장 많이 쓰이는 방법이다. 이 매트릭을 이용한 방법은 Model summary가 어떤 문서의 집합에 가장 적당한 요약이라고 가정하고, 다른 시스템에서 만들어진 System summary와 비교해서 그 성능을 평가하는 방법이다. 즉 Model summary는 인간이 요약한 결과를 말하고, System summary는 시스템이 요약한 결과를 말한다. 이 방법을 이용해서 본 논문에서는 제안된 TS-F와 전통적인 방법인 TF-IDF를 ROUGE-N 매트릭을 사용해서 비교하도록 한다. 여기서 N은 Model summary 와 System summary 사이의 n-gram recall을 말한다. n-gram 에서 n은 연속적인 n개의 단어의 길이를 나타낸다. ROUGE-N은 Model summary의 집합과 한 System summary 사이의 n-gram recall 을 계산하고, 다음과 같이 정의된다.

$$ROUGE-N=\frac{\sum_{S\in m}\sum_{gram_n\in S}N_{match}(gram_n)}{\sum_{S\in m}\sum_{gram_n\in S}N(gram_n)} \quad (7)$$

여기서 m은 Model summary 그리고 n은 $gram_n$의 길이를 나타내고, $N_{match(gram_n)}$은 Model summary의 집합과 하나의 System summary에서 발생하는 최대 n-gram의 수를 나타낸다.

Longest common subsequence (LCS) 는 두개의 문자열이 있을 때 두 문자열의 최대 공통 문자열을 찾는 문제를 말한다. 이와 관련해서 Lin et al.[13]은 System summary와 Model summary사이에 LCS 평가 방법을 제시했는데, 본 논문에서의 평가 방법 역시 ROUGE-N 매트릭 뿐 아니라 ROUGE-L 매트릭을 포함하도록 한다. 하지만 기본적으로 LCS는 문자열안에 부분적인 관계를 설명하지 못하기 때문에, 본 논문에서는 ROUGE-W-1.2 매트릭 역시 사용하기로 한다. ROUGE-W-1.2 는 1.2의 weighting을 가지는 weighted longest common subsequence를 평가하는 방법을 말한다. 또한 일련의 문장에서 한 쌍의 단어를 평가하는 skip-bigram에 근거한 ROUGE-S와 unigram에 기반한 방법



(그림 5) TF-IDF 와 TS-F의 Recall, Precision, F-Measure 비교
(Figure 5) Recall, Precision, F-Measure Comparison of TF-IDF and TS-F

인 ROUGE-SU도 역시 사용하도록 한다. 따라서 사용된 ROUGE 방법은 총 8개로 ROUGE-1, ROUGE-2, ROUGE-3, ROUGE-4, ROUGE-L, ROUGE-W-1.2, ROUGE-S, 그리고 ROUGE-SU 매트릭이다. 이러한 매트릭들은 recall, precision, 그리고 F-measure로 계산되고 다음과 같이 정의한다.

$$Recall(system|\text{model}) = \frac{|system \cap \text{model}|}{\text{model}} \quad (7a)$$

$$\text{Precision}(\text{model}|system) = \frac{|system \cap \text{model}|}{system} \quad (7b)$$

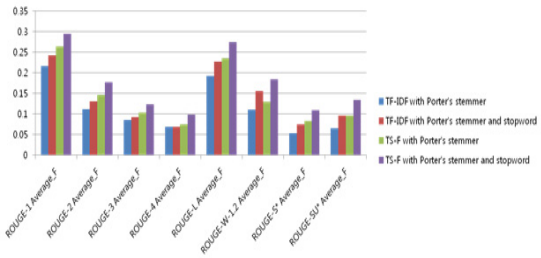
$$F_\beta = \frac{(1+\beta^2) \times precision \times recall}{\beta^2 \times precision \times recall} \quad (7c)$$

여기서 System과 Model 사이의 교집합은 두 요약문이 공유하는 단어의 수를 말한다. β는 System summary와 Model summary사이의 비율이다. 즉 β가 2일 경우 precision보다 recall에 더 가중치를 두는 것이고, β가 0.5일 경우 recall보다 precision에 더 가중치를 두는 것이다. 본 논문에서는 F1-score를 사용했기 때문에 사용된 β는 1이다.

3.1.2 실험결과

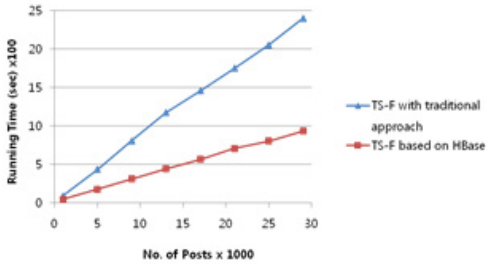
본 논문에서 사용된 실험은 ROUGE 매트릭을 사용해서 두개의 System summary: TS-F와 TF-IDF와 두 개의 Model summary를 가지고 실험하였다. 먼저 실험은 TF-IDF와 TS-F를 비교하는데, 이 외에도 Porter's stemmer와 stop-word를 사용했을 때와 안 했을 때의 실험도 병행하였다. 또한 동일한 환경에서 자동 요약을 하는데 걸리는 시간을 평가하기 위해서 HBase를 사용했을 경우와 안했을 경우 성능의 차이를 비교하였다. 첫 번째와 두 번째 비교는Recall, Precision, 그리고 F-Measure를 계산해서 비교하였고. 마지막 비교는 실행 시간에 근거해서 비교하였다.

그림 5는 TF-IDF와 TS-F의 계산된 Recall, Precision, F-Measure를 보여준다. X축은 ROUGE 매트릭들을 나타내고 Y축은 계산된 값을 나타낸다. 값은 0과 1사이에 위치하며 1에 가까울수록 좋은 성능을 보인다. R, P, 과 F는 각각Recall, Precision, 그리고 F-Measure의 약어를 나타낸다. 그림을 보면 ROUGE-1 Average F의 값이 ROUGE-L Average F 값보다 큰 것을 볼 수가 있는데, 이것은 트위터에서 사용자들이 쓸 수 있는 단어의 수가 140 개로 제한 되어있기 때문에 연속적인 문자열을 선호하기 보다는 짧은 단어를 통해서 내용을 전달하기 때문이라고 보여진다. 결과적으로 ROUGE-4 Average R과 ROUGE-4 Average F에서 계산된 결과가 비슷하거나 TF-IDF가 더 나은 결과를 보이지만, 이것을 제외한 나머지 TS-F에서 TF-IDF보다 좋은 결과를 보임을 볼 수 있다. 그림 6 은 TF-IDF와 TS-F를 다른 관점에서 비교한 것이다. 즉 전처리 과정에서 Porter's stemmer와 stop-word를 각각 사용했을 때와 안했을 때도 비교하였다. 이 결과 역시 TS-F가 TF-IDF보다 더 나은 결과를 보임을 볼 수 있다.



(그림 6) Porter's stemmer 와 stop-word 대한 TF-IDF와 TS-F 비교

(Figure 6) Comparison of TF-IDF and TS-F with Porter's stemmer and stop-word



(그림 7) 증가하는 데이터에 대한 HBase 사용에 대한 TS-F의 비교

(Figure 7) Comparison of TS-F with or without HBase for the incremental data

트위터의 스트리밍 데이터는 지속적으로 늘어나기 때문에, 증가하는 데이터를 관리할 필요가 있다. 본 논문에서는 이러한 증가하는 데이터에 대해서 HBase를 사용하여 실행시간을 감소시키고자 하였다. 실험은 UBUNTU 12.04 64bit, double dual-core 2.1 GHz Intel core processors와 4GB RAM를 가지는 환경에서 실행하였다. 그림 7은 증가하는 데이터에서 TS-F를 실행했을 때 HBase를 사용

한 경우와 사용하지 않은 경우를 구분하여 비교한 것이다. X축은 포스트의 수를 나타내고, Y축은 실행시간을 나타낸다. 그림에서 보듯이 HBase를 사용한 것이 사용하지 않은 것보다 더 완만한 선을 유지하는 것을 볼 수 있다. 이것은 HBase를 사용한 TS-F 가 데이터베이스 스키마에서 column family가 재사용이 가능하기 때문에, 주기마다 요약된 내용을 다시 사용해서 그 실행시간을 줄였기 때문이다. 예를 들어, 한 column family인 HashTag_Vectorized 는 요약 실행 전에 벡터화 된 포스트들을 미리 저장을 해서 실행시간을 줄이고, 그 다음 그 column에 저장된 포스트들은 새로운 포스트가 들어왔을 때 재사용이 되는 것이다.

3.1.3 실험에 대한 성찰

본 논문은 제안된 시스템의 실험 및 평가는 몇가지 제한된 요소를 가지고 있다. 첫 번째 사용된 Model summary가 두 명의 학생으로 부터 나온 것이다. Lin C.Y. 는 ROUGE 메트릭을 사용하는데 있어[14]에서 전문가로 이루어진 가능한 많은 사람들이 Model을 만들 경우 더 높은 결과를 기대할 수 있다고 하였다. 하지만 본 논문에서 사용된 트위터 데이터 정보의 경우 전문가를 선정하기에는 어려운 부분이 많았다. 그리고 현재는 Model summary가 두 명의 학생으로부터 이루어 졌지만, 이 후에는 추가적으로 더 다양한 연령대의 사람들을 대상이나 전문적인 토픽을 선정해서 해당 전문가들의 Model summary를 만든다면 더욱 객관적으로 정확한 결과를 기대할 수 있을 것이다. 그리고, 트위터를 사용하는 유저들의 토픽에 대한 접근성을 평가하기 위한 하나의 방법으로 사용자의 Log를 이용해서 해당 토픽에 접근한 사용자의 기록을 분석해서 평가 할 수 있지만, 본 논문에서는 이러한 사용자의 접근성 평가를 위한 사용자 Log를 분석하는 데 충분한 데이터를 얻지 못하였기 때문에, 이 후 충분한 데이터를 모아서 접근성에 대한 평가를 진행할 예정이다. 또한 HBase와 관련해서 그림 7의 실험에 사용된 최대 Post의 수는 30,000 이었고, 그 이후의 실행시간에 대해서는 Twitter가 제공하는 Post의 수와 실험장비의 제약으로 인해서 한계를 가질 수밖에 없었다. 하지만 Twitter 유료 서비스를 이용한다면 더 많은 데이터와 함께 다른 NoSQL를 비교해서 분석한다면 빅 데이터를 처리하는 데이터베이스를 이용하는 문서관련 분야에 기여를 할 수 있을 것이라고 판단된다.

4. 결 론

본 논문에서는 퍼지 시스템을 이용한 자동 요약 시스템을 HBase 기반으로 설계 및 구현 하였다. 이 시스템은 많은 데이터를 처리하는데 효과적인 HBase 기반으로 설계 되었으며, 기존의 벡터 스페이스 모델인 TF-IDF에 퍼지 모델을 접목시켜 자동 요약 시스템을 구현하였다. 실험 평가는 ROUGE 매트릭에 의해서 제안된 시스템의 요약문을 TF-IDF로 추출한 요약문과 비교하였고, 대부분의 경우 좋은 결과를 볼 수 있었다. 또한 재사용이 가능한 HBase 테이블의 특징을 이용해서 요약문을 실행하는데 HBase의 장점을 사용하지 않은 경우와 비교해서 시간을 감소시키는 결과를 가져왔다. 제안된 시스템은 적절한 Model summary 의 선택과 실제적인 HBase 사용을 위한 Hadoop 실행 환경 구축을 위한 과제를 가지고 있어 이를 계속 발전시킬 계획이다.

참고문헌(Reference)

[1] Over, P. and J. Yen. “An Introduction to DUC 2003 – Intrinsic Evaluation of Generic News Text Summarization Systems.” Available for: http://duc.nist.gov, 2003.

[2] Sharifi, B., Hutton, M.A., and Kalita, J. “Summarizing microblogs automatically.” In Proc. HLT/NAACL-10. pp. 685-688, 2010.

[3] Inouye, D. “Multiple post microblog summarization” Research Final Rep. Colorado Springs, GA: University of Colorado at Colorado Springs, 2010

[4] Radev, D., Jing, H., Sty, M., and Tam, D. “Centroid-based summarization of multiple documents” Information Processing and Management. vol. 40, pp. 919-938, 2004.

[5] Erikan, G. and Radev, D. “LexRank: Graph-based centrality as salience in text summarization. J. Artif. Intell. Res. vol. 22, pp. 457-479, 2004.

[6] Mihalcea, R. and Tarau, P. “TextRank: Bringing order into texts” In Proceedings of EMNLP-04. pp. 404-411, 2004.

[7] Ghemawat, S., Gobioff , H., and Leung, S.-T. “The Google File System” In Proceedings of SOSP ’03. pp. 29-43, 2003.

[8] Dean, J., Ghemawat, S. “MapReduce: Simplied Data Processing on Large Clusters, Communications of the ACM. 51, 1 (Jan. 2008), pp. 107-113, 2008.

[9] Chang, F., Dean, J., Ghemawat, S., Hsieh, W.C., Wallach, D.A., Burrows, M., Chandra, T., Fikes, A., and Gruber, R.E. “Bigtable: A Distributed Storage System for Structured Data” ACM Trans. Comput. Syst. 26, 2 (June 2008), pp. 1-26, 2008. DOI=http://doi.acm.org/10.1145/1365815.1365816.

[10] Porter, M. F. “An Algorithm for Suffix Stripping. Program. vol. 14, no. 3, pp. 130-137, 1980.

[11] Zadeh, L.A. “Fuzzy sets” In Information and Control. vol. 8, no. 3, pp. 338-393, 1965.

[12] Lin, C.Y. “ROUGE: A Package for Automatic Evaluation of Summaries” In Proceedings of the Workshop on Text Summarization. Branches Out (WAS 2004). pp. 74-81, 2004.

[13] Lin, C.Y. and Josef, F. “Automatic Evaluation of Machine Translation Quality Using Longest Common Subsequence and Skip-Bigram Statistics” In Proceedings of the 42th Annual Meeting of the Association for Computational Linguistic (ACL 2004). pp. 605-612, 2004.

[14] Lin, C.Y. “Looking for a Few Good Metrics: Automatic Summarization Evaluation – How Many Samples Are Enough?” In Proceedings of NTCIR Workshop 4, Tokyo, Japan, June 2-4, 2004.

◐ 저 자 소 개 ◑

이 상 훈 (Sanghoon Lee)

2004년 수원대학교 컴퓨터학과 졸업(학사)
2006년 수원대학교 대학원 컴퓨터학과 졸업(석사)
2013년 조지아 주립대학교 대학원 컴퓨터학과 졸업(석사)
2013 ~ 현재 조지아 주립대학교 대학원 컴퓨터학과 박사과정
관심분야 : 데이터 마이닝, 시멘틱 웹, 빅 데이터기반 텍스트 마이닝, etc.
E-mail : shlee8020@gmail.com

문 승 진 (Seung-Jin Moon)

1986년 텍사스주립(오스틴)대학교 컴퓨터학과 졸업(학사)
1991년 플로리다 주립대학교 대학원 컴퓨터학과 졸업(석사)
1997년 플로리다 주립대학교 대학원 컴퓨터학과 졸업(박사)
1997 ~ 현재 수원대학교 컴퓨터학과 교수
관심분야 : 데이터베이스, 실시간 데이터베이스, 모바일 데이터베이스, 임베디드 DB, 실시간 시스템, etc.
E-mail : sjmoon103@hotmail.com