키워드 집합 데이터를 위한 효과적인 군집화 알고리즘

2016년

서강대학교 대학원

컴퓨터공학과

이 세 희

키워드 집합 데이터를 위한 효과적인 군집화 알고리즘

지도교수 정 성 원

이 논문을 공학석사 학위논문으로 제출함

2017년 1월 3일

서강대학교 대학원

컴퓨터공학과

이 세 희

논 문 인 준 서

이세희의 공학석사 학위논문을 인준함

2017 년 1 월 3 일

주심 박 석 (인)

부심 정 성 원 (인)

부심 박 운 상 (인)

감사의 글

2013년 초, 6학기가 막 시작되기 직전에 연구실에 들어와서 석사를 졸업하기까지 무려 4년이라는 시간이 흘렀습니다. 연구실과 학교를 떠나는 이 시점에서 이 지면을 빌어 감사의 말씀을 전하고자 합니다.

가장 먼저 아무것도 모르고 너무나 부족한 저를 학부생 때부터 하나하나 가르쳐 주시면서 키워주신 정성원 교수님께 감사의 말씀을 드립니다. 교수님께서 때로는 따끔하게, 때로는 부드럽게 지도해 주신 덕분에 아직 부족하긴 해도 당당하게 사회로 나아갈 수 있게 되었습니다. 다시 한번 감사합니다. 또한, 수업과 논문지도 과정에서 자세하고 정확하게 지도해주신 박석 교수님과 박운상 교수님께도 감사의 말씀을 드립니다.

제가 처음 연구실에 왔을 때부터 자상하게 또 날카롭게 옆에서 도와주고 조언을 아끼지 않았던 연구실 선배님이신 범준이 형과 형철이 형! 정말 감사합니다. 그리고 연구실 동기 상근이형, 준홍이. 혼자였으면 너무 힘들고 어려웠겠지만 든든한 동기가 있어서 정말 너무나 즐겁게 보낼 수 있었습니다. 그리고 새롭게 들어온 초등학교 때부터 같이한 용담이에게도 고맙고 힘내라는 말을 전하고 싶습니다.

항상 힘들 때 제가 어떠한 결정을 하더라도 묵묵하게 지지해주고 응원해준 부모님, 그리고 동생 현수. 모두 고맙습니다. 더욱더 믿음직스럽고 자랑스러운 아들이자 형이 되도록 노력하겠습니다.

마지막으로 5년이 넘는 시간 동안 제 옆에서 항상 저를 응원해주고 힘들 때마다 다시 기운 나게 해준, 여자친구 선희에게도 고맙다는 말을 하고 싶습니다.

서강대학교 아담샬관 816호 맨 끝자리. 이곳에서 있었던 4년간의 추억과 배움은 저에겐 가장 큰 행운이었고 기쁨이었고 또 축복이었습니다. 다시 한번 감사합니다.

2017년 1월

이 세 희

목 차

그림 목차

표 목차

요 약

최근 정보화기술의 발전과 대중화 그리고 스마트폰을 비롯한 첨단 기기들의 발달로 인하여 다양한 종류의 데이터가 지속적으로 누적되고 있다. 이로 인하여 군집화 기술에 대한 수요가 증가하고 있고 다양한 알고리즘이 개발되었다. 그러나 최근까지 연구되어진 군집화 알고리즘들은 최근 증가하고 있는 이미지, 동영상, 가상현실, 초고화질 영상, 3D 등 다양한 종류의 데이터를 효과적으로 군집화 하지 못한다. 따라서 이러한 데이터를 효과적으로 군집화하는 방법이 필요하다.

그러나 지금까지의 키워드 기반 데이터 군집화 알고리즘은 고정된 차원에 기반을 두고 있고, 키워드 공간이 한정되어 있기 때문에 차원이 고정되어 있지 않고 키워드 공간이 무한한 키워드 기반 데이터를 효과적으로 군집화 하지 못한다는 문제를 지니고 있다.

이러한 문제를 해결하기 위해 본 논문에서는 각각의 데이터의 차원의 크기가 다르고, 각 차원이 가지는 의미가 무색해진 데이터인 키워드 집합 데이터를 새롭게 정의하고 이 데이터를 위한 유사도 측정법을 새롭게 제안한다. 새롭게 정의한 유사도 측정법은 다양한 종류의 키워드를 고려하여 유사도를 측정한다. 그리고 키워드 집합 데이터로 이루어진 군집을 대표하는 대푯값을 새롭게 정의하고, 새롭게 정의한 데이터와 유사도 측정법을 바탕으로 초기에 군집의 개수를 알지 못해도 군집화가 가능한 키워드 집합 데이터 군집화 알고리즘인 CASK(Clustering Algorithm for Set of Keywords)를 제안한다. CASK는 각 군집의 대푯값을 기반으로 반복을 통해서 유사한 키워드 집합 데이터를 한 군집으로 모아준다.

Abstract

Due to the recent development and popularization of information technology and the development of advanced devices including smart phones, various types of data are continuously accumulating. As a result, the demand for clustering technology is increasing and various algorithms have been developed. However, clustering algorithms that have been investigated until recently cannot efficiently clustering various kinds of data such as image, video, virtual reality, ultra-high-resolution image and 3D. Therefore, there is a need for a way to effectively clustering these data.

However, since this keyword-based data clustering algorithms are based on a fixed dimension, and the keyword space is limited, it do not effectively cluster the keyword-based data in which the dimension is not fixed and the keyword space is infinite.

In order to solve this problem, we newly define the keyword set data, which is the data in which the dimension of each data is different and the meaning of each dimension is unclear, and newly proposes a similarity measurement method for this data. The newly defined similarity measure measures similarity by considering various kinds of keywords. We also define CASK(Clustering Algorithm for Set of Keyowords), which is a keyword aggregation data clustering algorithm that can group clusters even if we do not know the number of clusters initially based on newly defined data and similarity measure method. CASK collects similar keyword aggregation data through repetition based on the representative value of each cluster.

제 1 장

서론

인터넷을 비롯한 정보통신기술(Information Technology)의 발달로 인하여 수 많은 사람들이 정보를 생산하고 공유하고있다. 이러한 상황에 맞춰 야후(Yahoo), 구글(Google), 페이스북(Facebook), 인스타그램(Instagram), 트위터(Twitter), 위키피디아(Wikipedia) 등 많은 회사들이 사용자들이 공유한 데이터를 분석하고 재가공하여 새로운 가치를 창출하는 서비스들을 내놨고, 이들은 현재 세계적인 기업으로 성장하여 막대한 수익을 얻고 있다.[11] 이렇게 정보를 분석하고 재가공하는 것은 막대한 가치를 창출해내는 매우 중요한 기술이다. 특히 매일 새로운 정보가 쌓이는 최

근 데이터의 특성상 새로운 지식의 발견이나 확장을 위한 군집화 기술 개발에 대한 요구가 증가하고 있고 군집화 기술에 대한 연구가 많이 진행되고 있다. 군집화(clustering) 기술은 인공지능(ArtificialIntelligence), 자연어처리(Natural Language Processing) 등을 위한 전처리 과정으로 사용되는 중요한 기술로 현재 다양한 분야에서 사용하고 있으며 활발하게 연구되고 있다. 군집 분석법은 구체적인 특성을 공유하는 군집을 찾아준다. 즉 이미 알려져 있는 클래스의 레이블을 이용하지 않고 객체들 분석한다. 객체들의 분포에 대한 사전 정보가 없이 분석을 하기 때문에 보다 높은 복잡도를 갖는다.

특히 최근에는 스마트폰(smart phone)을 비롯한 전자기기와 센서의 발달, 그리고 인터넷 서비스의 발전으로 인하여 정형화 되어있지 않은 다양한 종류의 데이터가 생성되고 있고 이러한 데이터를 분석할 수 있는 군집화 기술에 대한 요구가 늘어나고 있다. 실제 삼성전자에서는 사용자 누구나 쉽게 360도 영상을 촬영할 수있는 기기를 내놨고, 페이스북에서는 이러한 360도 영상을 공유할 수 있고, 다른 사람이 컴퓨터나 일반 스마트폰 뿐만 아니라 가상현실(virtual Reality) 머리장착디스플레이(Head MountedDisplay)에서도 생동감 있게 볼 수 있도록 하고 있다. 구글의 유튜브(Youtube)에서는 일반 고화질 동영상을 넘어 4K 영상을 공유하고 시청할 수 있게 하고 있고 소니(SONY)에서는 집에서도 가상현실 게임이 가능한 차세대 콘솔 게임기를 출시했다. 이렇듯 최근에는 일반 사진, 영상 컨텐츠 뿐만 아니라 초고화질의 멀티미디어 컨텐츠와 가상현실 컨텐츠까지 생산하고 공유하고 있다. 이러한 컨텐츠들은 개수가 조금만 있는 경우에는 큰 문제가 되지 않는다. 하지만 구글, 페이스북과 같은 세계적인 인터넷 서비스들은 하루에 수 억 명이 사용하며 다양한 종류의 수많은 컨텐츠를 생산한다. 따라서 현재 우리는 매일매일 다양한 종류의 데이터를 정확하게 분석해야 하는 문제에 직면하고 있다.

이러한 문제를 해결하는 방법 중 하나는 “키워드”를 사용하여 데이터를 관리하고 분석하는 것 이다. 각각의 컨텐츠에 자신을 설명 할 수 있는 키워드를 여러 개 달아두고 키워드를 검색 및 분석에 사용한다. 키워드를 사용하면 상당한 시간과 계산이

필요한 이미지 처리(image processing)이나 영상처리(video processing) 없이 키워드만을 가지고 분석 및 재가공을 할 수 있다.[2,3,12]

키워드와 같이 연속되지 않는 값을 속성으로 가지는 데이터를 범주 속성 데이터라고 한다. 범주속성 데이터는 수치 속성과는 다르게 서로의 값을 통해 크기 비교가 불가능하다. 색상이나 직업 등과 같이 값들이 같은지 아닌지만 판별이 가능할 뿐, 각각의 거리간의 크기 비교가 불가능하다. 따라서 범주 속성상의 값들이 얼마나 유사한지를 판별하기 위한 측정 기법들이 발전되어 왔는데, 범주 속성을 가지는 데이터의 거리 측정기법으로 해밍 거리(hamming distance), 리벤슈타인 거리(levenshteindistance), 자카드 계수(jaccard coefficient), 코사인 유사도 등이있다[13,14]. 또한 이러한 유사도 측정법을 바탕으로 k-mode, ROCK, TBKMedoid와 같은 범주속성 데이터 군집화 알고리즘이 개발되었다. 이러한 범주속성데이터들은 차원이 고정 되어있고 각 차원이 나타내는 정보가 정해져 있다. 예를 들어 [표 1]은 총 25마리 말들의 의학기록 데이터를 나타내고 있다.

ID	Surgery	Age	Pulse	.....	Temperature	Outcome
1	2	1	60-80	.....	38-39	2
2	1	1	80-100	.....	39-40	2
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
24	1	2	120-140	.....	38-39	1
25	1	1	60-80	.....	38-39	1

[표 1] 25마리 말의 의학 데이터

이때 1차원 ID는 관리 번호를 나타내고, 2차원 “Surgery”는 수술 여부를 표시하고 있다. 값이 1이면 수술을 한적이 있는 말이고, 2이면 수술 한적이 없는 말이다. 그리고 3차원의 Age는 1이면 6개월 이상 된 성인 말, 2이면 6개월 미만인 어린 말을 의미한다. 4차원 Pulse는 분당 심박수의 범위를 20단위로 끊어서 나타내고 있고,

Temperature는 체온의 범위를 나타낸다. 마지막 차원인 Outcome은 말이 현재 죽었는지 살아있는지를 나타내는데 값이 1이면 살아있다는 의미이고 2이면 죽었다는 것을 의미한다. 이처럼 범주속성의 데이터는 차원이 고정되어 있으며 각 차원이 의미하는 바가 명확하다는 특징을 지닌다. 이러한 데이터를 군집화 하는 대표적인 알고리즘이 k-mode[1]이다. 그러나 소셜미디어를 비롯한 키워드(혹은 태그)로 이루어진 데이터는 차원이 고정되어있지 않을 뿐만 아니라 각 차원이 나타내는 정보가 정해져 있지 않다. 즉 차원의 개념이 무색해졌다는 점에서 기존의 범주속성 데이터와 큰 차이를 보인다.

ID	Keyword
1	서울, 여행, 광화문, 경복궁, 한식, 한복, 한국
2	커피, 아메리카노, 스타벅스, 광화문
3	디즈니, 디즈니랜드, 미키마우스, 미니마우스, 일본

[표 2] 키워드 데이터 예시

예를 들어 [표 2] 에는 총 세 개의 데이터가 있으나 각 데이터가 가지는 키워드의 개수가 모두 다르며 동시에 각 값들이 의미하는바가 차원에 구속되어있지 않는 집합의 형태를 가진다. 이러한 차이로 인하여 선행연구 되었던 k-mode를 비롯한 범주 속성 데이터 거리 측정 기법뿐만 아니라 범주속성 군집화 알고리즘으로는 최근의 키워드 데이터를 군집화하지 못한다.

[표 2]와 같은 키워드 데이터를 군집화 하기 위하여 다양한 연구가 진행되었다. 특히 소셜미디어 데이터에 대한 알고리즘이 많이 연구되고 있다. 소셜미디어 데이터는 그래프 구조로 전체 데이터가 구성되어있고 이 그래프를 분석하여 사용자가 관심이 있을 것이라고 예상되는 데이터와 사용자를 추천해준다. 그러나 이렇게 그래프를 이용한 분석 말고도 각 데이터의 키워드를 활용하여 새로운 정보를 추출하는 알고리즘에 대한 연구가 활발하게 진행되고 있다.

[16]에 따르면 실시간 트위터 분석을 통해서 지진에 관한 정보를 수집하고 예측할 수 있다. 또 [17]에 의하면 사용자들의 기분과 상태를 바탕으로하여 주식의 등락을 예측할 수 있다고 한다. 이렇듯 소셜미디어 데이터는 사용자들의 감정과 의도가 키워드로 나타나게 되고, 이를 분석하여 다양한 분야에서 새로운 가치를 창출할 수 있다는 점에서 활발하게 연구가 이루어지고 있다. 대표적인 알고리즘으로 STREAMCUBE[5], Event Photo mining[2], SMSC[15], COSA[6]가 있다.

STREAMCUBE와 Event Photo mining은 소셜미디어 데이터를 군집화 하는 대표적인 알고리즘이다. 그러나 소셜미디어 데이터의 키워드는 그 종류가 다양하고 키워드 공간이 무한대이기 때문에 키워드에 다른 정보를 추가하여 군집화한다. STREAMCUBE는 키워드와 위도, 경도, 시간을 추가하여 군집화 하고 있고, Event Photo mining은 동일한 이벤트 키워드를 지닌 데이터를 모은 뒤에 각 군집별로 이미지 처리를 하여 군집화 한다. 이 두 알고리즘은 키워드만을 활용하여 군집화 하는 알고리즘은 아니다. 그리고 위도, 경도, 시간, 이미지 처리 등 다른 특징을 추가로 사용하여 군집화 하기 때문에 군집화 할 수 있는 데이터가 한정되어있고, 키워드를 제거하는 과정에서 정보의 누락이 발생한다는 단점을 지니고 있다.

SMSC와 COSA는 다른 정보없이 키워드만을 가지고 군집화하는 대표적인 알고리즘이다. COSA는 문서를 키워드로 군집화하는 대표적인 알고리즘이다. 온톨로지(Ontology)를 활용하여 키워드 공간을 줄이고, 키워드가 많아 데이터의 밀도가 낮아지는 문제를 해결했다. 그리고 키워드가 가지는 의미를 바탕으로 전처리 후 군집화하기 때문에 군집화 성능이 높다. 그러나 온톨로지를 항상 최신으로 유지하는데 많은 비용이 들기 때문에 실시간으로 다양한 키워드가 사용되는 최근의 데이터를 효과적으로 군집화 하지 못한다. SMSC는 마이크로 블로그와 트위터, 인스타그램 등 키워드를 지닌 모든 데이터가 대상인 알고리즘이다. SMSC는 키워드의 공간이 늘어나면 늘어날수록 데이터의 밀도가 낮아지는 문제를 해결하기 위하여 키워드 공간을 한정하고, 공간 안의 모든 키워드에 대하여 가상의 데이터를 만들어 군집화 하는 방

법을 사용하고 있다. 밀도가 낮은 실제 데이터가 아닌 높은 밀도의 가상의 데이터를 기반으로 1차 군집화를 하고, 이 정보로 실제 데이터를 군집화 하기 때문에 키워드 데이터의 밀도가 낮아 생기는 문제를 잘 해결하고 있다. 하지만 이 알고리즘은 키워드의 공간이 전체 데이터의 개수보다 매우 작다는 가정하에 고안되었다. 따라서 키워드 공간이 매우 작고, 이로인하여 고려되지 않는 키워드가 늘어난다. 이는 정보의 누락을 의미하고 정확도에 매우 큰 악영향을 미친다.

키워드를 기반으로 하는 데이터는 키워드의 공간에 따라서 정확도가 매우 크게 떨어지는 문제를 지니고 있다. 따라서 대부분의 알고리즘이 이러한 문제를 해결하기 위하여 다른 속성을 추가하거나, 키워드 공간을 한정하여 문제를 해결하고 있다. 이는 키워드 공간이 열린 공간(open space)에서는 잘 동작하지 않는다는 문제를 야기한다. 최근 수집되는 데이터들의 키워드 공간이 점차 커지고 있어 열린 공간에 대한 데이터를 군집화하는 알고리즘이 반드시 필요하다.

따라서 이와 같은 단점을 보완하기 위하여 본 논문에서는 닫힌 키워드 공간 뿐만 아니라 열린 키워드 공간에서도 효과적으로 군집화 하는 알고리즘을 제안하여 기존의 군집화 알고리즘이 지니는 문제를 해결하고자 한다. 본 논문에서는 이러한 문제를 해결하기 위하여 새롭게 키워드 집합 데이터를 정의한다. 그리고 이 데이터에 대한 새로운 유사도 측정법을 고안하고, 효과적인군집화 알고리즘인 CASK(Clustering Algorithm for Set of Keyword data)를 제안한다.

이후 본 논문은 2장에서 기존의 범주속성 데이터 유사도 측정법과 키워드를 활용하여 대표적으로 키워드를 통해 관리되는 문서와 소셜미디어 데이터를 군집화하는 기존의 알고리즘에 대해 설명한다. 그리고 3장에서는 키워드 집합 데이터와 키워드 집합 데이터로 이루어진 군집의 대푯값을 정의하고 키워드 집합 데이터를 위한 유사도 측정법을 새롭게 정의한다. 그리고 4장에서는 키워드 집합 데이터를 군집화 하는 CASK를 소개하고 5장에서는 실제 데이터셋을 사용하여 성능을 분석한다. 마지막으로 6장에서본 논문에 대한 결론을 내리며 마무리한다.

제 2 장

관련연구

이 장에서는 범주속성 데이터 유사도를 측정하는 기존의 측정법과 키워드를 사용하여군집화하는 알고리즘에 대하여 알아본다. 그 중, 대표적인 키워드 기반 데이터인 소셜미디어 데이터를 군집화 하는 알고리즘인 STREAMCUBE와 Event Photo mining을 소개한다. 그리고 키워드를 통해 문서를 군집화하는 COSA를 소개한다.

2.1 기존의 범주속성 데이터 유사도 측정법

기존의 범주 속성 데이터 유사도 측정법에는 해밍 거리(hamming distance), 레벤슈타인 거리(levenshteindistance), 코사인 유사도(cosine similarity), 자카드 계수(jaccard coefficient)가 있는데 이러한 키워드 집합 데이터를 측정하는데 부족한 부분이 있다. 먼저 해밍 거리는 차원이 같은 두 데이터의 각 차원끼리 비교하여 값이 다른 차원이 총 몇 개인지 찾아서 이를 거리로 한다. 예를 들어 “t,o,n,e,d”와 “r,o,s,e,s”의 해밍 거리는 3이다. 해밍 거리는 두 데이터의 차원이 같아야만 거리를 계산할 수 있기 때문에 차원이 고정되어 있지 않은 데이터들의 유사도는 측정할 수 없다. 레벤슈타인 거리는 두 문자열사이의 거리를 나타낸다. 두 문자열이 같아지기 위해서 삭제, 삽입, 수정 해야하는 경우를 찾아서 거리로 나타낸다. 예를 들면 “kitten”과 “sitting”의 레벤슈타인 거리는 “k”를 “s”로 한번, “e”를 “i”로 두 번 수정한 뒤에 “g”를 한번 삽입하면 되기 때문에 거리는 총 3이 된다. 이는 데이터의 차원의 개수가 달라도 거리를 계산할 수 있지만 문자열을 위한 거리이기 때문에 위치(차원)가 중요하다. 따라서 위치(차원)의 의미가 없는 데이터의 거리는 측정할 수 없다. 다음으로 코사인 유사도는 두 백터 사이의 각도를 계산하여 거리를 구한다. 두 백터 A, B의 코사인 유사도 $cs(A, B)$는 다음과 같다.

$$cs(A, B) = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}$$

코사인 유사도는 각 단어 하나를 하나의 차원으로 하여 백터를 구성하고 유사도를 측정한다. 코사인 유사도는 양수 공간이라는 것만 만족하면 얼마나 많은 차원이던 거리를 측정할 수 있다는 큰 장점을 지닌다. 이는 단어의 종류가 매우 다양하고 해당 값의 가중치까지 고려하여 유사도를 측정할 수 있다. 그러나 키워드의 종류가 매우 다양한 데이터의 경우 데이터의 차원이 너무 커진다. 즉 이때 각 데이터에 0의 값이 많아지고 계산 시간도 증가하는 등 매우 비효율적이기 때문에 키워드의 종류가 다양한 데이터를 효율적으로 다룰 수 없다. 마지막 유사도 측정법은 자카드 계수

이다. 두 데이터 A, B의 자카드 계수 $jc(A, B)$는 다음과 같다.

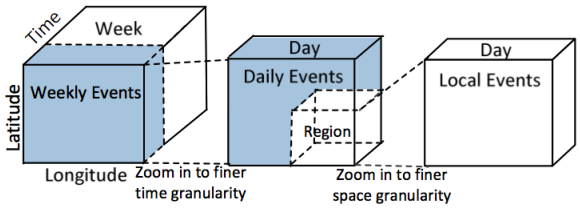
$$jc(A,B) = \frac{N_{A \cap B}}{N_{A \cup B}}$$

자카드 계수는 두 데이터를 집합으로 가정하고 계산한다는 점에서 차원이 고정되어 있지 않고 다양한 종류의 키워드를 가지는 데이터에 가장 알맞은 유사도 측정법이다. 그러나 코사인 유사도와 다르게 각 키워드의 가중치는 고려하지 않는다는 단점이 있다.

2.2 키워드를 활용한 소셜미디어 군집화 알고리즘

2.2.1 STREAMCUBE

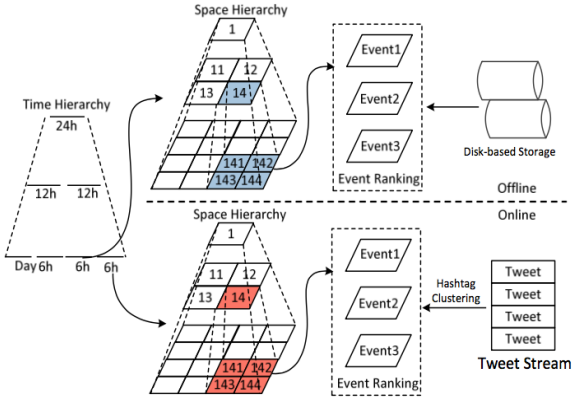
최근 가장 이슈가 되는 데이터는 소셜미디어 데이터이다. 소셜미디어 데이터는 데이터의 형태가 기존의 데이터와는 다르게 다양한 종류의 값을 가지고 있다. 대표적인 소셜미디어 데이터인 트위터는 위치정보, 시간, 텍스트, 이미지를 하나의 데이터가 가지고 있다. 물론 모든 데이터가 이 값을 다 가지고 있는 것은 아니다. 데이터를 생성하고 공유한 유저가 선택한 종류의 값만 가진다. 즉 어떤 데이터에는 위치정보와 이미지가 없을 수도 있다. 이러한 다양한 종류의 데이터로 이루어진 소셜미디어 데이터를 군집화하는 대표적인 알고리즘이 STREAMCUBE이다. STREAMCUBE는 위치정보, 시간, 그리고 태그(키워드)를 계층적으로 구성하여 서로 연관성이 높은 데이터를 하나의 군집으로 만들어 준다. 즉 STREAMCUBE는 특정 지역에서 특정 시간에 발생한 이벤트를 찾는데 그 목적이 있다. 이때 사용하는 구조는 [그림 1]에 나온 Zoomable Event Cube다.



[그림 1] Zoomable Event Cube

Zoomable Event Cube는 위도, 경도, 시간을 각각 축으로 만들어진 큐브다. 위도와 경도 축을 축소하면 지역이 좁아지고, 시간을 축소하면 특정 시간대를 설정할 수 있다. 또한 이 구조는 위치, 시간, 키워드 모두 사용이 가능하다는 장점을 지니고 있다. 데이터를 이러한 구조로 관리하여 군집화 하는 것이 STREAMCUBE의 기본 아이디어다.

STREAMCUBE는 가장 먼저 시간에 대한 계층구조를 만들어 데이터를 나눈다. 그리고 그 안에서 지역을 세분화하여 시간-공간 계층 구조를 만들고 특정 지역에서의 태그를 바탕으로 군집화 하고 이벤트 순위를 확인한다.



[그림 2] STREAMCUBE의 Framework

[그림 2]는 STREAMCUBE의 전체적인 구조를 보여준다. [그림 2]를 보면 먼저 시간에 대한 계층구조를 만든다. 하루 24시간을 두 번째 층에서는 12시간, 세 번째 층에서는 6시간 단위로 나눠 계층구조를 만든다. 그 후에 6시간 단위로 공간에 대한 계층구조를 만든다. 하나의 큰 공간을 네 곳을 나누어 계층화한다. 즉 첫 번째 층에서는 인덱스가 1인 하나의 공간만 존재하고, 두 번째 층에서는 인덱스가 11,12,13,14인 총 네 개의 공간이 존재하게 된다. 이렇게 시간-공간 계층을 만들고 난 후에 태그를 군집화 한다. 군집화 하는 아이디어는 동일한 이벤트를 나타내는 데이터를 하나의 군집으로 모아주는 것 이다. 예를 들어 “Election2016”, “Clinton”, “Trump” 태그들은 모두 2016 미국 대선이라는 동일한 이벤트를 나타내는 태그이다. 따라서 동일한 이벤트를 지칭하는 데이터를 하나의 군집으로 보고 군집의 대푯값은 이벤트가 된다.

STREAMCUBE는 다양한 종류의 데이터를 모두 활용하여 군집화 하는 매우 효과적인 알고리즘이다. 시간과 장소, 그리고 태크(키워드)를 모두 활용한다는 점에서 실시간 이벤트를 확인하고 이를 활용하는 분야에 매우 적합하다. 그러나 지역과 시간을 한정지어서 처리한다는 점은 이벤트를 분석 하는데 효과적일지 모르지만 이벤트 분석이 아닌 다른 목적으로는 활용하기 힘들다. 또한 태그를 군집화 하기 위하여 백오프워드를 사용했는데, 이는 키워드의 종류가 매우 다양한 최근의 데이터를 모두 다루기에는 하나의 데이터 크기가 커지기 때문에 매우 비효율적이다. 즉 STREAMCUBE는 소셜미디어 데이터를 한정적인 시간과 지역에서 태크를 분석하기에는 매우 훌륭하지만 키워드로 되어있는 소셜미디어 데이터가 아닌 다른 데이터(이미지, 문서, 동영상)등은 분석하기 어려워 적용이 힘들다는 단점을 가지고 있다.

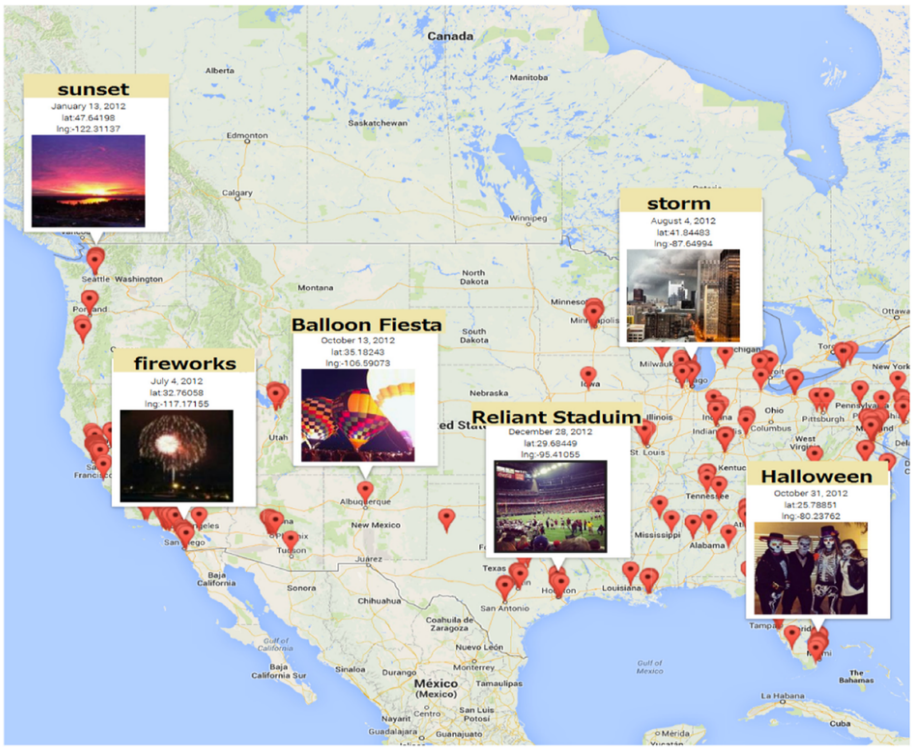
2.2.2 Event Photo mining

Event Photo mining은 STREAMCUBE와 비슷하다. 소셜미디어 데이터의 위치정보를 바탕으로 데이터를 나눈 뒤에, 특정 위치, 특정 시간에 발생한 이벤트를 찾는

다. 다만 STREAMCUBE와 다른점은 Event Photo mining은 이미지를 위치 정보과 키워드를 바탕으로 군집화 한다는 점이다. 따라서 이미지와 시간, 위지정보, 그리고 키워드를 포함하는 데이터를 군집화한다.

Event Photo mining은 우선 데이터의 키워드를 분석하여 이벤트 키워드 하나를 추출해낸다. 예를 들어 “비”, “태풍” 이라는 키워드를 가지고 있는 데이터의 이벤트 키워드는 “태풍”이 된다. 이렇게 하나의 이벤트 키워드만을 가지기 때문에 백오브 워드를 사용하는 STREAMCUBE와 비교하여 데이터 하나의 크기가 매우 작아지고 구조가 단순해진다. 이로 인하여 Event Photo mining은 유사도를 계산하는 시간이 줄어들고 데이터의 크기가 매우 작아진다는 장점을 가진다. 하나의 이벤트 키워드를 추출하고 난 뒤에는 동일한 이벤트 키워드인 데이터를 하나로 모아준다. 이 과정이 끝나면 Event Photo mining은 특정 위치에서 특정 시간에 나타난 이벤트를 기준으로 하는 군집을 얻게된다. 그 후 같은 군집에 있는 데이터의 이미지를 비교하여 대표 이미지를 얻어낸다. 그러면 다음 [그림 3]과 같은 결과를 얻게 된다.

[그림 3]을 보면 위치와 시간대 별로 미국에서 가장 이슈가 된 이벤트를 사진과 함께 확인할 수 있다. Event Photo mining은 이벤트 키워드를 활용하여 소셜미디어 데이터를 군집화 했고, 그 안에서 이미지 처리를 통해 대표 이미지 까지 분석했다. 이는 소셜미디어의 이미지 데이터를 군집화하는데 매우 훌륭한 결과를 보여줬다. 그러나 Event Photo mining역시 위치정보와 시간 정보를 활용하기 때문에 순수 이미지를 키워드로 군집화 했다고 보기에는 어렵다. 또한 키워드의 종류는 매우 다양한데 이벤트 키워드로 한정했다는 점과, 각각의 키워드가 나타내는 이벤트를 정의하기에는 소셜미디어 데이터의 키워드가 너무 다양하다는 점은 단점이다.



[그림 3] Event Photo mining 결과

2.3 키워드를 활용한 문서 군집화 알고리즘

소셜미디어 데이터 외에도 키워드를 통해서 관리하는 대표적인 데이터에는 문서 데이터가 있다. 하나의 문서 데이터는 여러 단어로 이루어져있기 때문에 이 단어 중에서 해당 문서 데이터를 가장 잘 나타내는 것을 추출하여 문서를 관리한다. 이때 추출된 단어들이 해당 문서의 키워드가 된다. 문서 데이터를 군집화는 알고리즘들 역시 키워드를 활용하여 군집화한다. 일반적인 문서 군집화 알고리즘은 동일한 키워드가 몇 개인지 자카드 계수를 구하여 유사한 문서들을 계층 구조로 군집화 한다. 혹은 온톨로지를 통해서 계층적으로 군집화한다. 이러한 방법을 사용하는 대표적인

군집화 알고리즘이 COSA[6]다.

COSA는 먼저 Core Ontology를 정의한다. Core Ontology는 어휘, 컨셉, 관계함수, 그리고 heterarchy로 구성되어있다. 예를 들어 어휘는 {Hotel, Grand Hotel, Hotel Schwarzer Adler, Accommodation, ……}가 되고 컨셉은 {ROOT, HOTEL, ACCOMMODATION, …..}처럼 어휘의 상위 계층의 키워드이다. 관계함수는 어휘와 컨셉 사이의 관계로, {(Hotel, HOTEL), (Grand Hotel, HOTEL), ……}가 된다. 마지막 heterarchy는 컨셉 키워드와 컨셉 키워드 간의 관계로 예를 들면 {(HOTEL, ACCOMMODATION), (ACCOMMODATION,ROOT),……}가 heterarchy이다. 이렇게 Core Ontology를 만들어 둔 다음에는 문서를 분석하여 키워드를 추출한다. COSA는 문서의 데이터를 Core Ontology를 활용하여 컨셉 키워드와 맵핑을 시킨다. 매핑과정이 끝나면 각각의 문서 데이터는 컨셉 키워드를 값으로 가지는 벡터가 된다. 이렇게 만들어진 벡터를 가지고 군집화 알고리즘을 사용하여 문서를 군집화한다.

COSA는 Core Ontology를 활용하여 매우 많은 단어로 이루어진 문서를 대표하는 키워드를 추출하고 추출한 키워드를 활용하여 군집화 한다. COSA는 온톨로지를 사용하기 때문에 매우 정확한 키워드를 추출해낼 수 있을 뿐만 아니라 키워드 공간을 줄여 백오브워드처럼 데이터가 낭비되는 것을 막았다. 그리고 온톨로지를 사용하기 때문에 데이터의 계층구조를 고려하여 군집화가 가능하다는 장점을 지닌다. 그러나 문서의 주제가 다양하고 그로 인하여 키워드 역시 다양한데 이를 모두 온톨로지로 관리하는 것은 매우 비효율적이다. 또한 온톨로지를 사용하면 놓치게 되는 키워드가 생기고 새롭게 추가되는 키워드를 추가로 관리해줘야 한다는 단점을 가지고 있다.

제 3 장

키워드 집합 데이터와 유사도 측정법

이 장에서는 최근 소셜미디어를 비롯한 키워드 기반의 데이터의 특징을 알아보고 이러한 특징을 반영한 키워드 집합 데이터를 새롭게 정의한다. 그리고 새롭게 정의한 집합 데이터간의 유사도를 측정하는 측정법을 제안한다.

3.1 키워드 집합 데이터

데이터의 종류는 매우 다양하다. 익숙한 수치속성 데이터와 범주속성 데이터뿐만 아니라 이미지, 동영상, 음악과 같은 멀티미디어 데이터, 그리고 논문, 신문, 책 등과 같은 텍스트 데이터도 존재한다. 이러한 데이터를 분석하기 위한 여러 연구가 진행 되었고, 각 데이터 도메인에 맞는 군집화 알고리즘과 데이터 유사도(거리) 측정법들이 개발되었다. 현재까지 개발된 알고리즘들은 특정 도메인에서 빅데이터가 아닌 작은 데이터셋을 분석하는데는 매우 뛰어나다.

그러나 멀티미디어 데이터는 초고차원 데이터이므로 데이터간의 거리를 유사도(거리)를 계산하는데 매우 많은 시간이 필요하다. 그래서 대용량의 멀티미디어 데이터를 군집화하는데 상당한 시간이 걸리고 다양한 종류의 도매인이 혼재되어있는 데이터셋을 군집화하는데 한계가있다. 이러한 경향을 텍스트 데이터 역시 마찬가지이다. 텍스트 데이터는 매우 많은 단어로 이루어진 데이터이기 때문에 온톨로지(ontology)를 사용하거나 백 오브 워드(Bag-of-word)를 사용하여 데이터를 가공한 뒤에 군집화를 진행한다. 온톨로지와 백 오브 워드는 데이터를 가공해야 하는데 많은 시간이 필요하다는 점과 다양한 종류의 단어를 모두 다루기 힘들다는 단점을 지니고 있다.

이처럼 하나의 종류인 데이터를 분석하는 것도 상당히 어렵고 많은 시간이 필요하다. 그런데 최근에는 여러 종류의 데이터가 혼합되어 하나의 데이터를 이루는 데이터가 많아지고 있다. 대표적이 예로 인스타그램 데이터를 들 수 있다. 인스타그램 데이터는 기본이 이미지 혹은 동영상 이지만, 각 이미지에 태그(키워드)가 달려있고, 이미지를 포스팅한 유저의 생각을 적은 텍스트도 존재한다. 또한 위치정보도 같이 포함되기도 한다. 이러한 데이터를 분석하는 대표적인 방법은 도매인에 맞게 특정 데이터를 추출하여 분석한다. 예를 들어 위치정보만을 추출하여 위치에 따른 정보를 분석하거나, 이미지만을 추출하여 분석한다. 그러나 이러한 방법을 사용하면 추출하

지 않은 데이터에 대한 손실이 발생한다는 문제를 가지고 있다. 따라서 이러한 데이터를 관리하기 위하여 키워드(태그)를 사용한다. 모든 데이터를 나타내는 키워드를 달아서 검색 및 분석에 사용한다. 이러한 키워드 데이터는 Definition 1과 같다.

Definition 1. 키워드 데이터

어떤 키워드 데이터 kd_i는 총 n개의 키워드 $k_1, k_2, \cdots, k_n$ 으로 이루어져있다. 이때 키워드 k_j에 대한 특정 keyword space가 정해져 있지 않다. 즉, k_j의 종류는 무한집합이다. 그리고 키워드의 개수 n은 각각의 데이터마다 다를 수 있다.

Definition 1에서 정의된 키워드 데이터를 분석하기 위하여 현재까지는 특정 키워드를 추출하거나 키워드 스페이스를 정해서 사용하는 등 키워드 데이터를 그대로 사용하지 않고 한번 가공한 뒤에 사용한다. 키워드 데이터를 바로 사용하지 않는 이유는 이 키워드 데이터가 차원이 고정되어있지 않다는 집합의 특징을 가지고 있기 때문이다. 현재 집합 데이터에 대한 정의가 되어있지 않고 유사도(거리) 측정법 역시 없다. 그래서 우리는 키워드 데이터에 빈도수를 넣어 키워드 집합 데이터를 새롭게 정의한다.

Definition 2. 키워드 집합 데이터

떤 키워드 집합 데이터 sd_i는 총 n개의 속성 $A_1, A_2, \cdots, A_n$ 으로 이루어져있다. 속성 A_l은 2차원 벡터로써 키워드 k_l과 키워드의 빈도수 f_l으로 구성되어 있다.

Definition 2에서 정의된 것처럼 키워드 집합 데이터는 총 n개의 속성으로 이루어져있고 n은 고정되어 있지 않다. 또한, 각 차원이 속성을 지니는 기존의 범주 속성 데이터와는 다르게 이 키워드 집합 데이터는 차원이라는 개념이 적용되지 않는다. 예를 들어 서울 여행 중 찍은 사진 하나의 이미지에 대한 기존의 범주 속성 데이터는 (위도, 경도, 시간, 날짜)와 같은 4차원데이터로 이루어져 있다면, 키워드 집합 데이터는 “(비,1), (경복궁,1), (서울,1), (여행,1), (한국,1), (한복,1)”과 같이 여러 개의 키워드와 빈도수가 모여 집합을 이루고 있다. 이때 하나의 속성은 2차원 벡터

로써 1차원은 키워드이고 2차원은 해당 키워드의 빈도수이다. 예를 들어 “비, 경복궁, 서울, 여행, 한국, 한복” 이라는 키워드를 가지는 집합 데이터 sd_1는 n=6이고 A_1=(비,1), A_2=(경복궁,1), A_3=(서울,1), A_4=(여행,1), A_5=(한국,1), A_6=(한복,1) 총 6개의 속성을 가진다. 속성에 빈도수를 넣는 이유는 키워드를 강조하기 위하여 두 번 이상 사용하는 경우가 있기 때문이다. 예를 들어 위 키워드 집합 데이터의 예제에서 “경복궁”이라는 키워드를 강조하기 위하여 A_2의 빈도수를 3으로 할 수도 있다.

3.2 키워드 집합 데이터로 이루어진 군집의 대푯값

수치속성 데이터로 이루어진 군집의 대푯값으로는 중심점(centroid)를 사용한다. 그리고 이 중심점과 데이터간의 거리를 비교하여 데이터가 해당 군집과 얼마나 유사한지 판단한다. 이처럼 키워드 집합 데이터를 군집화 하기 위해서는 군집의 대푯값을 정의해야한다. 군집의 대푯값은 군집에 속한 키워드 집합 데이터들의 보편적인 특성을 반영해야한다. 그렇기 때문에 군집에 속한 키워드 집합 데이터의 모든 키워드와 빈도수를 대푯값으로 정의한다. 즉, 모든 키워드 집합 데이터의 합집합이 대푯값이 된다. 그러나 키워드 집합 데이터의 모든 키워드를 대푯값으로 사용하면 그 크기가 너무 커진다. 크기가 커지면 유사도 계산을 하는데 상당히 많은 시간이 걸리게 되고 관리하기도 쉽지 않다. 또한 전체 키워드 집합 데이터 중에서 단 한번만 사용하여 그 빈도가 매우 낮은 키워드 역시 대푯값이 되면 대푯값의 의미가 무색해진다. 이러한 문제를 해결하기 위해서 각 키워드의 빈도를 바탕으로 대푯값을 정한다. 이때 임계값 t를 사용한다. 임계값 t는 최소 비율을 의미한다. 즉 어떤 군집 C_i의 대푯값 Q_i의 어떤 키워드 k_j를 사용하는 키워드 집합 데이터가 적어도 군집 C_i 안의 전체 키워드 집합 데이터 중 t 만큼은 된다는 것을 의미한다. 따라서 대푯값으로 사

용되는 키워드들의 빈도수는 최소한 임계값 조건을 만족해야한다. 임계값 조건에 대한 정의는 다음과 같다.

Definition 3. 임계값 조건

임계값 t가 주어졌을 때, 어떤 군집 C_i의 모든 키워드 집합 데이터 $\{sd_1, sd_2, \cdots, sd_n\}$에 대하여 어떤 키워드 k_l이 총 $t \times n$개 이상의 키워드 집합 데이터에서 사용되었으면 키워드 k_l은 임계값 조건을 만족 했다고한다.

CASK는 어떤 군집에 대해서 Definition 3에서 정의한 임계값 조건을 만족하는 모든 키워드를 군집의 대푯값으로 하고, 이를 통해서 유사도를 측정하고 군집화한다. 자세한 군집의 대푯값은 다음과 같이 정의한다.

Definition 4. 키워드 집합 데이터 군집의 대푯값

어떤 키워드 집합 데이터들의 군집 $C_i = \{sd_1, sd_2, \cdots, sd_n\}$가 있을 때 이 집합 C_i의 대푯값은 $Q_i = \{q_1, q_2, \cdots, q_m\}$이다. 대푯값의 속성 q_j는 2차원 벡터로 키워드 k_j과 태그의 빈도수 f_j로 구성되어있다. 이때 대푯값의 속성 q_j의 키워드 k_j는 임계값 조건을 만족한다. 즉 $\frac{f_j}{n_i}$이 임계값 t보다 크거나 같다. 이를 수식으로 나타내면 다음과 같다.

$$\frac{f_j}{n_i} \geq t \quad \forall f_j \, of \, q_j \in Q_i$$

Definition 3과 Definition 4를 통해 정의된 군집의 대푯값에 대한 예를 들어보면, $t = 0.5$이고 $A_1 = (\text{비}, 1)$, $A_2 = (\text{경복궁}, 1)$, $A_3 = (\text{서울}, 1)$, $A_4 = (\text{여행}, 1)$, $A_5 = (\text{한국}, 1)$, $A_6 = (\text{한복}, 1)$인 sd_1과 $A_1 = (\text{한식}, 1)$, $A_2 = (\text{서울}, 1)$, $A_3 = (\text{여행}, 1)$, $A_4 = (\text{한국}, 1)$인 sd_2인 이루어진 군집 C_1의 대푯값 Q_1은 $\{q_1 = (\text{서울}, 2), q_2 = (\text{여행}, 2), q_3 = (\text{한국}, 2)\}$이 된다.

3.3 유사도 측정법

3.3.1 키워드 집합 데이터간의 유사도 측정

일반적으로 범주 속성데이터는 서로 같은 값(키워드)을 많이 포함하면 할수록 유사하다고볼 수 있다. 그러나 키워드 집합 데이터가 포함하는 키워드의 개수가 고정되어있지 않기 때문에 단순하게 같은 키워드를 많이 공유하고 있다고 유사한 것은 아니다. 예를 들어 sd_1와 sd_2 모두 포함하고 있는 키워드가 10개이고 sd_1 또는 sd_2에 속한 키워드가 100개이면, 전체적으로 10%만 같은 키워드를 가지고 있다고 볼 수 있다. 반면 sd_1 또는 sd_2에 속한 키워드가 10개이면, 두 데이터는100% 유사하다고 볼 수 있다. 키워드 집합 데이터 간의 정확한 유사도를 측정하기 위해서는 자카드 계수처럼 두 키워드 집합 데이터 sd_1와 sd_2에 대해서 데이터 sd_i와 sd_j 모두 속한 키워드뿐만 아니라 sd_i와 sd_j에 대하여 적어도한 쪽에 속하는 키워드의 개수 역시 고려해야한다. 그리고 키워드 집합 데이터에는 각 키워드의 빈도수(가중치) 값이 존재한다. 너무 많은 키워드가 있는 경우 좀 더 대표성을 지니는 키워드를 나타내기 위한 값이므로 유사도 측정에 있어서 이 가중치 역시 고려돼야 한다. 그래서 우리는 먼저 두 집합 데이터 sd_i와 sd_j 모두 속한 키워드 데이터 집합을 sd_i와 sd_j의 교집합 I_{ij}라고 하고 다음과 같이 정의한다.

Definition 5. 키워드 집합 데이터의 교집합

임의의 두 집합 데이터 sd_i와 sd_j에 대하여 sd_i와 sd_j 모두 속하는 태그를 가진 속성들로 이루어진 집합을 sd_i와 sd_j의 교집합이라 하고 I_{ij}로 표시한다. 즉, 두 속성 $A_k \in sd_i$와 $A_l \in sd_j$에 대하여

$$A_r = (k_k,\ f_k + f_l) \in I_{ij}$$

이다.

예를 들어 $A_1 = (\text{비}, 1)$, $A_2 = (\text{경복궁}, 1)$, $A_3 = (\text{서울}, 1)$, $A_4 = (\text{여행}, 1)$, $A_5 = (\text{한국}, 1)$,

$A_6 =$(한복,1)인 sd_1과 $A_1 =$(한식,1), $A_2 =$(서울,1), $A_3 =$(여행,1), $A_4 =$(한국,1)인 sd_2의 교집합 I_{ij}는 $\{A_1 =$(서울, 2), $A_2 =$(한국, 2), $A_3 =$(여행, 2)$\}$이 된다. 그리고 두 데이터 sd_i와 sd_j중 적어도 어느 한쪽에 속한 키워드 데이터 집합을 합집합 U_{ij}이라 하고 다음과 같이 정의한다.

Definition 6. 키워드 집합 데이터의 합집합

임의의 두 집합 데이터 sd_i와 sd_j에 대하여 적어도 sd_i 또는 sd_j 한 쪽에 속하는 태그를 가진 속성들로 이루어진 집합을 sd_i와 sd_j의 합집합이라 하고 U_{ij}로 표시한다. 즉, 두 속성 $A_k \in sd_i$와 $A_l \in sd_j$에 대하여

$$A_r = (k_k, f_k + f_l) \in U_{ij},\ where\ k_k = k_l$$

$$A_k \in U_{ij} \text{ and } A_l \in U_{ij},\ where\ k_k \neq k_l$$

이다.

예를 들어 $A_1 =$(비,1), $A_2 =$(경복궁,1), $A_3 =$(서울,1), $A_4 =$(여행,1), $A_5 =$(한국,1), $A_6 =$(한복,1)인 sd_1과 $A_1 =$(한식,1), $A_2 =$(서울,1), $A_3 =$(여행,1), $A_4 =$(한국,1)인 sd_2의 합집합 U_{ij}는 다음과 같다.

$\{A_1 =$(비,1), $A_2 =$(경복궁,1), $A_3 =$(서울,2), $A_4 =$(여행,2), $A_5 =$(한국,2), $A_6 =$(한복,1), $A_7 =$(한식,1)$\}$

Definition 5와 Definition 6에서 정의한 교집합과 합집합은 기존의 집합의 개념에 빈도수라는 가중치를 부여한 집합의 형태이다. 키워드 집합 데이터의 교집합과 합집합을 통해서 두개의 집합데이터 sd_i와 sd_j의 유사도를 측정한다. 새롭게 정의한 두 키워드 집합 데이터 sd_i와 sd_j의 유사도는 자카드 계수를 활용하였고, 자세한 정의는 다음과 같다.

Definition 7. 키워드 집합 데이터의 유사도

임의의 두 집합 데이터 sd_i와 sd_j에 대하여 두 집합 데이터의 유사도 $s(sd_i, sd_j)$는 다음과 같다.

$$s(sd_i, sd_j) = \frac{\sum_{l=1}^{n(I_{ij})} f_l}{\sum_{l=1}^{n(U_{ij})} f_l}$$

Definition 7을 보면 본 논문에서 새롭게 제시한 유사도는 키워드 집합 데이터의 합집합과 교집합의 빈도수의 합으로 계산한다. 즉, 두 키워드 집합 데이터의 총 키워드 중에서 공통된 것의 비율을 유사도로 한다. 우리는 이 유사도 측정법에 대하여 다음과 같은 정리를 얻을 수 있다.

Theorem 1. 유사도 범위

공집합이 아닌 임의의 두 집합 데이터 sd_i와 sd_j의 유사도 $s(sd_i, sd_j)$의 범위는 $[0,1]$이다.

Proof)
case 1) $I_{ij} = \varnothing$
임의의 두 집합 데이터 sd_i와 sd_j의 속성 중 공통된 키워드 k_l이 하나도 없으면, Definition 3에 의하여 교집합 I_{ij}는 공집합이 된다. 이때 $\sum_{l=1}^{n(I_{ij})} f_l$은 0이 된다. 이때 유사도 $s(sd_i, sd_j)$는 0이 된다.
case 2) $I_{ij} \neq \varnothing$
임의의 두 집합 데이터 sd_i와 sd_j의 모든 속성의 빈도수 f_l은 0보다 크다. 그러므로 sd_i와 sd_j의 교집합 I_{ij}의 모든 속성의 빈도수 역시 0보다 크다. 따라서 $0 < \sum_{l=1}^{n(I_{ij})} f_l$는 항상 만족한다. 그리고 Definition 5과 Definition 6에 의하여 $I_{ij} \subseteq U_{ij}$는 항상 만족 하므로 $\sum_{l=1}^{n(I_{ij})} f_l \leq \sum_{l=1}^{n(U_{ij})} f_l$ 이다.

따라서 $0 < \sum_{l=1}^{n(I_{ij})} f_l \leq \sum_{l=1}^{n(U_{ij})} f_l$ *는 교집합* I_{ij}*가 공집합이 아니면 항상 만족한다. 즉,* sd_i*와* sd_j*의 유사도* $s(sd_i, sd_j)$*는* $\frac{\sum_{l=1}^{n(I_{ij})} f_l}{\sum_{l=1}^{n(U_{ij})} f_l}$ *이므로 다음과 같은 수식을 얻을 수 있다.*

$$0 < s(sd_i, sd_j) \leq 1$$

case 1), 2)에 의하여 Definition 5에서 정의한 임의의 두 집합 데이터 sd_i*와* sd_j*의 유사도* $s(sd_i, sd_j)$*의 범위는* $[0, 1]$ *이다.* □

Theorem 1에서 증명한 것처럼 유사도 값의 범위는 $[0, 1]$인데, 0이면 교집합이 공집합, 즉 공통된 것이 하나도 없다는 의미이다. 유사도 값이 1이면 교집합과 합집합이 같아 두 키워드 집합 데이터는 키워드뿐만 아니라 빈도수 까지 완벽하게 같다는 의미이다. 즉 임의의 두 키워드 집합 데이터 sd_i와 sd_j는 유사도 $s(sd_i, sd_j)$가 1에 가까울수록 서로 유사하다는 것을 의미한다. 예를 들어 A_1=(비,1), A_2=(경복궁,1), A_3=(서울,1), A_4=(여행,1), A_5=(한국,1), A_6=(한복,1)인 sd_1과 A_1=(한식,1), A_2=(서울,1), A_3=(여행,1), A_4=(한국,1)인 sd_2의 유사도 $s(sd_1, sd_2)$는 0.6이 된다.

3.3.2 키워드 집합 데이터와 군집의 유사도 측정

3.3.1장에서는 두 키워드 집합 데이터 간의 유사도를 어떻게 측정하는지 알아봤다. 이번 3.3.2장에서는 키워드 집합 데이터 sd_1과 키워드 집합 데이터로 이루어진 군집 C_1 간의 유사도 $s(sd_1, C_1)$을 어떻게 구하는지 설명한다.

데이터와 군집 사이의 유사도 혹은 거리를 측정하는 방법에는 크게 두 가지 방법이 존재한다. 먼저 첫 번째는 군집의 대푯값을 구한 뒤에 대푯값과 데이터와의 유사도를 측정하여 나온 값을 데이터와 군집 사이의 유사도로 보는 방법이다. 그리고 두

번째는 해당 데이터가 군집에 속했을 때 군집의 밀도나 군집에 속한 데이터들 간의 거리 등 군집의 특성이 얼마나 잘 유지되는 지를 확인하는 방법이다. 이때, 군집에 데이터를 넣었을 때 특성의 변화가 가장 적은 군집이 가장 유사한 군집이 된다. 두 방법 모두 장단점이 존재한다. 첫 번째 방법의 경우 군집의 대푯값을 정의해야하는 문제가 존재하지만, 대푯값을 정의한 뒤에는 군집에 속한 모든 데이터를 다 확인할 필요 없이 대푯값만을 사용하여 유사도를 측정할 수 있다는 큰 장점을 가지고 있다. 두 번째 방법은 대푯값을 정의할 필요가 없다는 장점을 가지고 있다. 또한 군집에 속한 모든 데이터를 확인하기 때문에 좀 더 정확한 판단을 할 수 있다는 장점을 지닌다. 그러나 유사도를 측정하기 위하여 군집에 속한 모든 데이터를 확인하고 계산해야한다는 점과 군집에 충분한 데이터가 없는 경우 정확한 특성 파악이 힘들다는 단점을 지니고 있다. 물론 이러한 단점을 임계값으로 해결할 수 있으나 그러면 정확한 임계값은 어떤 값인지에 대한 문제가 새롭게 생긴다.

본 논문에서는 이미 3.2장에서 키워드 집합 데이터로 이루어진 군집의 대푯값을 정의하였기 때문에 데이터와 군집 사이의 거리를 측정하는 방법으로 빠르고 정확하게 유사도를 측정할 수 있는 첫 번째 방법을 사용했다. 자세한 유사도 측정법은 다음과 같이 정의한다.

Definition 8. 키워드 집합 데이터와 군집의 유사도

어떤 키워드 집합 데이터 sd_i와 어떤 키워드 집합 데이터들의 군집 $C_j = \{sd_1, sd_2, \cdots, sd_n\}$의 대푯값을 $Q_j = \{q_1, q_2, \cdots, q_m\}$라 할 때, *유사도* $s(sd_i, C_j)$*는 다음과 같다.*

$$s(sd_i,\ C_j) = s(sd_i,\ Q_j)$$

Definition 4에서 정의한 어떤 키워드 집합 데이터들의 군집 $C_j = \{sd_1, sd_2, \cdots, sd_n\}$의 대푯값 $Q_j = \{q_1, q_2, \cdots, q_m\}$를 보면 Definition 2에서 정의한 키워드 집합 데이터와 형태가 동일하다. 그래서 어떤 키워드 집합 데이터 sd_i와 어떤 키워드 집합 데이터들의 군집 $C_j = \{sd_1, sd_2, \cdots, sd_n\}$의 유사는 두 키워드 집합

데이터 sd_i와 Q_j의 유사도와 같다. 예를 들어 $t=0.5$이고 $A_1=$(비,1), $A_2=$(경복궁,1), $A_3=$(서울,1), $A_4=$(여행,1), $A_5=$(한국,1), $A_6=$(한복,1)인 sd_1과 $A_1=$(한식,1), $A_2=$(서울,1), $A_3=$(여행,1), $A_4=$(한국,1)인 sd_2인 이루어진 군집 C_1의 대푯값 Q_1은 $\{q_1=$(서울, 2), $q_2=$(여행, 2), $q_3=$(한국,2)$\}$이다. 이때 군집 C_1과 $A_1=$(한식,1), $A_2=$(음식,1), $A_3=$(불고기,1), $A_4=$(한국,1)인 sd_3의 유사도는 $s(sd_3, Q_1)$이고 유사도 값은 0.3이 된다. 즉 군집 C_1와 키워드 집합 데이터 sd_3의 유사도는 0.3이다.

제 4 장

키워드 집합 데이터를 위한 군집화 알고리즘

최근 키워드와 태그를 기반으로 하는 군집화 알고리즘에 대한 연구가 활발하게 진행되고 있다. 이 장에서는 제 2 장에서 제안한 키워드 집합 데이터를 효과적으로 군집화하는 알고리즘을 소개한다.

4.1 CASK 설명

4.1.1 CASK 알고리즘

기존의 키워드 데이터 군집화 알고리즘으로는 COSA, STREAMCUBE, Event Photo mining가 있다. 이러한 알고리즘들은 키워드 데이터의 키워드 중 일부만을 사용하거나 그 키워드 공간을 정의한 뒤에 키워드 데이터를 군집화한다. 그러나 이는 키워드의 종류가 다양하고 데이터 하나가 가지는 키워드의 개수가 고정되어있지 않은 키워드 집합 데이터를 효율적으로 군집화 하지 못한다.

그래서 본 논문에서는 가장 대표적인 군집화 방법인 분할기반 군집화 방법(partitional based clustering method)을 사용하여 제안한 키워드 집합 데이터를 위한 새로운 군집화 알고리즘인 CASK(Clustering Algorithm for Set of Keyword data)을 제안한다.

CASK는 반복을 통해 전체 데이터셋을 분할하여 군집화한다. 이처럼 반복을 통해 군집화하는 이유는 간편하게 다양한 도메인에 쉽게 적용할 수 있고 정확도 역시 높기 때문이다. CASK는 각 군집의 대푯값을 기준으로 각각의 키워드 집합 데이터를 배치한다. 이때 각 키워드 집합 데이터는 자신과 가장 유사한 군집에 속하게 된다. 이때 가장 유사한 군집의 의미는 군집의 대푯값과 키워드 집합 데이터의 유사도가 다른 군집들의 대푯값과의 유사도보다 크다는 것을 말한다. 그렇기 때문에 모든 키워드 집합 데이터는 반드시 하나의 군집에만 속하게 된다. 이 과정을 모든 군집이 안정 될 때까지 하는데, 군집이 안정되었다는 의미는 반복해도 군집의 대푯값이 변하지 않고 각 군집에 속한 키워드 집합 데이터 역시 변하지 않는다는 것을 말한다. 이러한 기본적인 아이디어를 기반으로 한 CASK 알고리즘은 다음과 같다.

Algorithm 1: CASK

```
1:  function CASK(D, k, t, maxIter);
    Input: D = {sd_1, sd_2, ..., sd_n} (Dataset)
           k (Maximum number of clusters)
           t (Threshold)
           maxIter (Maximum Iteration)
    Output: Q = {Q_1, Q_2, ..., Q_k} (Set of Representative value of the clusters)
            L = {l(sd_i) | i = 1,2,...,n} (Set of cluster labels of D.)
2:  foreach c_i ∈ C do:
3:      c_i = sd_j ∈ D (e.g. Random Sampling)
4:  end
5:  foreach sd_i ∈ D do:
6:      l(sd_i) = maxSimilarity(sd_i, Q_j) j ∈ {1,2,...,k};
7:  end
8:  change = F;
9:  iteration = 0;
10: repeat :
11:     foreach Q_i ∈ Q:
12:         UpdateRepresentative(Q_i, L);
13:         if Q_i = ∅ then:
14:             remove(Q_i);
15:             k = k - 1;
16:         end
17:     end
18:     UpdateThreshold(Q, t);
19:     foreach sd_i ∈ D do:
20:         maxSim = maxSimilarity(sd_i, Q_j) j ∈ {1,2,...,k};
21:         if maxSim ≠ l(sd_i) then:
22:             l(sd_i) = maxSim;
23:             change = T;
24:         end
25:     end
26:     iteration += 1;
27: until change is T and iteration ≤ maxIter;
```

Algorithm 1을 보면 CASK는 초기에 전체 데이터셋과 군집의 최대 개수 k, 임계값 t, 그리고 최대 반복횟수인 $maxIter$를 입력받는다. 가장 먼저 2번에서 4번줄까지 나와 있는 것처럼 총 k의 키워드 집합 데이터를 무작위로 선택하여 대푯값으로

한다. 그 후 5번에서 7번까지처럼 각 키워드 집합 데이터와 각 군집의 대푯값과 유사도를 계산 하여 가장 유사도가 높은 대푯값의 군집의 라벨을 *maxSimilarity*함수를 통해 찾아 데이터에 표시한다. 초기 대푯값에 대하여 모든 데이터의 라벨링이 끝나면 군집의 변화를 확인하는 변수 *change*를 F로 초기화 하고(8번) 반복 횟수를 나타내는 *iteration* 변수의 값을 0으로 초기화(9번) 한다. 이제 각 군집의 대푯값을 업데이트한다(12번). 이때 13번에서 16번까지처럼 대푯값이 공집합이 되면 해당 대푯값은 삭제하고 군집의 개수 k를 하나 줄인다. 대푯값 업데이트가 끝나면 임계값을 업데이트한다(18번). 그 후 19번에서 25번까지의 반복을 통해서 전체 키워드 집합 데이터를 다시 라벨링 한다. 이때 어떤 키워드 집합 데이터 sd_i에 대하여 새롭게 계산된 가장 유사도가 높은 군집의 라벨 *maxSim*이 기존의 라벨 $l(sd_i)$와 다르면 군집에 변화가 생긴 것이기 때문에 sd_i의 라벨을 *maxSim*으로 변경하고(23번), 변수 *change*를 T로 변경한다(24번). 모든 키워드 집합 데이터에 대한 라벨링이 끝나면 반복이 끝났다는 표시로 *iteration*값을 하나 증가하고(26번) 멈춤 조건을 확인한다(27번). 만일 멈춤조건에 만족하면 반복을 멈추고 라벨링 결과인 $L = \{l(sd_i) \mid i = 1,2,\cdots,n\}$와 군집의 대푯값의 집합인 $Q = \{Q_1, Q_2, \cdots, Q_k\}$를 산출하며 알고리즘은 끝이 난다.

Algorithm 1에서 자세한 설명과 증명이 필요한 군집의 대푯값을 업데이트 하는 방법(12번)과 임계값 업데이트 방법(18번)은 4.1.2장에서 설명하고 Algorithm 1의 멈춤 조건(27번)은 4.1.3장 에서 설명한다.

4.1.2 키워드 집합 데이터 군집의 대푯값 업데이트

본 논문은 Definition 4에서 키워드 집합 데이터로 이루오진 군집에 대한 대푯값을 정의하였다. 이 정의에 따라서 전체 키워드 집합 데이터가 라벨링되면 각 대푯값의 속성들 역시 새롭게 계산이 된다. 이때 우리는 임계값 조건을 만족하여 만들어진 대푯값에 대하여 다음과 같은 정리를 얻을 수 있다.

Theorem 2. 임계값 업데이트

임계값을 t로하여 군집의 대푯값을 계산했을 때, k개의 군집 $C_1, C_2, \cdots, C_k$에 대하여 $t' = \min_{1 \le i \le k}(\frac{f_j}{n_i})$, where f_j of $q_j \in Q_i$의 값이 임계값 t보다 크면, t'을 임계값으로 하여 대푯값을 계산해도 임계값을 t로하여 군집의 대푯값을 계산했을 때와 모든 군집의 대푯값이 같다.

Proof)
Theorem 2를 귀류법(Proof by Contradiction)으로 증명할 수 있다.

k개의 군집 $C_1, C_2, \cdots, C_k$에 대하여 임계값을 t로하여 군집의 대푯값을 계산하면 $t < t' = \min_{1 \le i \le k}(\frac{f_j}{n_i})$, where f_j of $q_j \in Q_i$ 이다. 이때의 t'을 임계값으로 하여 모든 군집의 대푯값을 계산하면 어떤 군집의 대푯값은 변한다고 가정하자.

t'은 t보다 크기 때문에 t'를 임계값으로 하면 t를 임계값으로 했을 때의 어떤 군집 C_i의 대푯값의 속성 q_l는 대푯값의 속성이 되지 못한다. 즉 이 q_l의 f_l은 $\frac{f_l}{n_i} < t' \quad \forall f_l$ of $q_l \in Q_i$ 를 만족한다. 즉, 위 가정을 만족하려면 t를 임계값으로 하여 계산된 어떤 군집 C_i의 대푯값 $Q_i = \{q_1, q_2, \cdots, q_m\}$중 $t \le \frac{f_l}{n_i} < t'$인 어떤 속성 q_l이 있어야하고 이 $\frac{f_l}{n_i}$값은 $\min_{1 \le i \le k}(\frac{f_j}{n_i})$, where f_j of $q_j \in Q_i$가 된다.

그러나 $\min_{1 \le i \le k}(\frac{f_j}{n_i})$의 값은 t'이기 때문에 $t \le \frac{f_l}{n_i} < t'$인 어떤 속성 q_l은 존재할 수 없고 대푯값이 변한다는 가정은 모순이다. 따라서 Theorem 2의 명제는 참이다. □

Theorem 2에 의하여 우리는 대푯값을 구한 뒤에 임계값을 업데이트 할 수 있다. 새로운 임계값 t_n은 $\min_{1 \le i \le k}(\frac{f_j}{n_i})$, where f_j of $q_j \in Q_i$이 된다. 그리고 이 새로운 임계값은 다음 반복 후 대푯값을 계산할 때 활용된다. 새로운 임계값 t_n은 항상 직전 임계값인 t보다 크거나 같기 때문에 반복이 계속 될수록 각 군집의 대푯값들은

높은 비율의 키워드를 가지게 되고 이로 인해 매우 유사한 키워드 데이터 들이 하나의 군집에 속하게 되고 이는 각 군집의 응집력은 더 강하게 만들어 준다.

Definition 4에서처럼 각 군집의 대푯값을 계산하다 보면 총 k개의 군집이 있을 때 어떤 키워드 집합 데이터들의 군집 $C_i = \{sd_1, sd_2, \cdots, sd_n\}$가 있을 때 이 군집 C_i의 대푯값 $Q_i = \{q_1, q_2, \cdots, q_m\}$에 대하여 $\frac{f_j}{n_i} < t \quad \forall f_j \; of \; q_j \in Q_i$ 이면 군집 C_i의 대푯값 $Q_i = \varnothing$ 이다. 대푯값이 공집합 이라는 것은 C_i의 키워드 집합 데이터가 포함한 모든 키워드들의 비율이 임계값 t보다 낮다는 것을 의미한다. 즉, 해당 군집에 속한 키워드 집합 데이터들은 서로 유사도가 매우 낮아 군집으로써 의미가 없다. 이 경우 CASK는 해당 군집을 제외하고 군집화를 이어간다. 즉 $Q_i \neq \varnothing$ 인 Q_i들만 사용하여 키워드 집합 데이터를 재배치한다. 이렇게 되면 총 군집의 개수는 k에서 $Q_i = \varnothing$인 C_i의 개수를 뺀 수가 된다. 이러한 경우 때문에 CASK는 다른 분할 기반 군집화 알고리즘과 다르게 군집의 개수가 아닌 군집의 개수의 최댓값을 입력받아 효과적인 군집의 개수를 찾아가며 군집화 한다.

4.1.3 CASK 멈춤 조건

CASK는 Algorithm 1에서처럼 멈춤 조건에 만족할 때 까지 대푯값을 업데이트하고 업데이트 된 대푯값과 키워드 집합 데이터 간의 유사도를 계산하여 군집화 한다. 때문에 이 알고리즘이 멈추기 위한 조건이 반드시 필요하다. 좋은 멈춤 조건은 적당한 횟수로 반복을 진행하여 적절한 시간에 군집화가 완료되어야한다. 또한 모든 키워드 집합 데이터가 적절한 군집에 배치되어야한다. 이를 가능하게 하기 위하여 CASK는 Definition 8과 같은 멈춤 조건을 설정하였다.

Definition 8. CASK의 멈춤 조건

CASK는 다음 두 조건 중 하나의 조건만 만족하면 반복을 멈춘다.

1) 사용자가 입력한 최대 반복 횟수 r에 대하여 CASK가 r번 반복한 경우

2) CASK는 모든 군집의 변화가 없는 경우

Definition 8에 정의된 첫 번째 조건은 사용자가 최대 반복 횟수를 입력하고 이 안에 군집화를 끝내는 것이다. 최대 반복 횟수를 정하고 이를 기준으로 한다는 것은 정확한 반복 횟수를 알 수 없고, 데이터셋에 따라 반복 되는 횟수가 다르기 때문에 항상 좋은 군집화 결과를 얻을 수 없다는 매우 큰 단점이 있다. 그러나 군집화를 하는 컴퓨터의 성능과, 시간 등 다른 요인에 의하여 특정 횟수 이상 반복이 진행되면 더 이상 의미가 없다고 판단되면 멈추는 것이 더 현명하다. 예를 들어 사용자가 판단하여 100번의 반복 이상이 되면 너무 많은 시간이 소요되어 멈추길 원하는 경우 r의 값을 100으로 설정할 수 있다. 두 번째 조건이 100번 안에 만족되어 멈추면 괜찮지만 그렇지 않은 경우 CASK는 100번 반복하고 나온 군집화 결과를 최종 결과로 한다.

Definition 8에 정의된 두 번째 조건은 모든 분할 기반 군집화 기법의 멈춤 조건이다. 이 조건이 만족한다는 증명을 하기 전에 모든 군집의 변화가 없다는 것을 다음과 같이 정의한다.

Definition 9. 군집화 결과가 같다.

p번 반복 후 나온 군집화 결과 R_p와 $p+1$번 반복 후 나온 군집화 결과 R_{p+1}은 다음을 모두 만족할 때 서로 같다, 즉 모든 군집의 변화가 없다고 한다.

1) R_p에 속한 군집의 개수 k_p와 R_{p+1}에 속한 군집의 개수 k_{p+1}은 같다.

2) 모든 R_{p+1}의 군집 들은 자신과 대푯값이 동일하고, 군집에 속한 키워드 집합 데이터가 동일한 군집이 R_p에 하나 존재한다.

Definition 9에 정의된 것처럼 p+1번 반복 후 나온 군집화 결과가 p번 반복 후 나온 군집화 결과와 동일하다면, p+2번 반복 후 나온 군집화 결과 역시 동일하다. 즉 모든 군집이 안정화되었다는 것을 의미한다. 따라서 더 이상 반복을 할 의미가 없기 때문에 반복을 멈추고 군집화 결과를 도출한다.

CASK가 반복되면서 어떤 군집 C_i에 대해 대푯값과 군집에 속한 키워드 집합 데이터간의 평균 거리인 $\frac{\sum_{j=1}^{n} s(sd_j, Q_i)}{n}$의 값이 일정해 진다는 것은 해당 군집이 안정화 되고 있다는 것을 의미한다. 만일 Definition 8에서 첫 번째 조건이 없다면 CASK가 멈출지, 그리고 반복이 계속되면서 군집화 결과가 안정화 되는지에 대한 의문이 생길 수 있다. 그래서 우리는 다음과 같은 정리를 통해서 CASK는 반드시 언젠가는 멈춘다는 것과 반복이 계속 되면서 군집화 결과가 안정화된다는 것을 확인 하고자한다.

Theorem 3. CASK는 Definiton 8의 두 번째 조건을 언젠가 반드시 만족한다.

Proof)

Definition 7을 통해서 새롭게 계산된 대푯값들을 바탕으로 모든 키워드 집합 데이터를 재배열한다. 따라서 어떤 군집 C_i에 대해서 p번째 반복 후 계산 된 대푯값을 Q_i^p가 있다. CASK는 p+1번 반복에서 이 Q_i^p를 기준으로 데이터를 재배치한다. 재배치한 후에 대푯값 Q_i^{p+1}를 구하면 Q_i^{p+1}는 Q_i^p보다 더 군집을 잘 대표한다. 즉 $\frac{\sum_{j=1}^{n^p} s(sd_j, Q_i^p)}{n^p}$의 값과 재배치한 후에 새롭게 계산된 Q_i^{p+1}을 바탕으로 하는 $\frac{\sum_{j=1}^{n^{p+1}} s(sd_j, Q_i^{p+1})}{n^{p+1}}$의 값을 비교하면 항상 다음을 만족한다.

$$\frac{\sum_{j=1}^{n^p} s(sd_j, Q_i^p)}{n^p} \leq \frac{\sum_{j=1}^{n^{p+1}} s(sd_j, Q_i^{p+1})}{n^{p+1}}$$

이는 재배치 된 키워드 집합 데이터를 고려하여 Q_i^{p+1}*를 구했기 때문에 당연한 사실이다. 그러나 Definition 5에 정의된 유사도 측정법의 범위는* $[0,1]$*이기 때문에* $\frac{\sum_{j=1}^{n^{p+1}} s(sd_j, Q_i^{p+1})}{n^{p+1}}$*의 범위 역시* $[0,1]$*이다. 즉*

$$0 \leq \frac{\sum_{j=1}^{n^p} s(sd_j, Q_i^p)}{n^p} \leq \frac{\sum_{j=1}^{n^{p+1}} s(sd_j, Q_i^{p+1})}{n^{p+1}} \leq 1$$

이 된다. 반복이 진행되면서 대푯값과 키워드 집합 데이터 간의 평균 거리는 1에 수렴하고 결국 $\frac{\sum_{j=1}^{n^p} s(sd_j, Q_i^p)}{n^p}$*와* $\frac{\sum_{j=1}^{n^{p+1}} s(sd_j, Q_i^{p+1})}{n^{p+1}}$*는 1에 가까워지다가 같아진다. 이 값이 같아진다는 것은 군집* C_i*가 안정화 되었다는 것을 뜻하므로* C_i*의 대푯값과* C_i*에 속한 키워드 집합 데이터의 변화가 없다는 것을 의미한다. 따라서 CASK의 반복이 계속되면 계속될수록 군집은 안정화되고 Definition 8의 두 번째 조건을 만족하여 멈추게된다.* □

이번 장을 통해서 CASK가 반드시 멈춘다는 사실을 확인했다. 또한 CASK는 계속 반복을 하며 군집들의 안정성을 높이다가 반복을 해도 군집화 결과가 변하기 않을 때 즉, 안정되었을 때 멈춘다는 것을 확인할 수 있었다.

4.2 CASK 예제

4.1절에서 설명한 키워드 집합 데이터를 위한 군집화 알고리즘인 CASK의 예제를 보이도록 하겠다. 이번 예제를 위해서 대표적인 키워드 집합 데이터인 인스타그램 (instagram) 데이터를 수집하고, 그 중에서 일부를 가져와 예제에 사용했다. 이 데

이터를 바탕으로 하여 임계값 업데이트, 군집의 개수 조정 등 CASK가 키워드 집합 데이터를 군집화 하는 방법을 자세하게 설명한다.

sd_i	Keyword
sd_1	서울, 여행, 광화문, 경복궁, 한식, 한복, 한국
sd_2	커피, 아메리카노, 스타벅스, 광화문, 경복궁
sd_3	디즈니, 디즈니랜드, 미키마우스, 미니마우스, 일본
sd_4	한식, 서울, 여행, 한국, 광화문
sd_5	히라주쿠, 오모테산도, 일본, 디즈니샵
sd_6	디즈니, 미녀와야수, 장난감, 인형
sd_7	미키마우스, 미키, 캐릭터, 인형
sd_8	엘사, 안나, 디즈니, 겨울왕국, 디즈니랜드, 인형
sd_9	한국, 서울나들이, 서울, 삼청동, 경복궁, 광화문, 여행
sd_{10}	광화문, 테라로사, 브런치, 서울
sd_{11}	광화문, 공연, 경복궁, 가을, 음악, 노래
sd_{12}	비빔밥, 한식, 나물, 광화문, 경복궁, 서울
sd_{13}	겨울슬리퍼, 캐릭터, 디즈니샵
sd_{14}	디즈니, 미키마우스, 미니마우스, 푸우
sd_{15}	에뛰드하우스, 디즈니, 미니마우스, 문구, 핑크
sd_{16}	미니마우스, 미키마우스, 디즈니, 장난감, 인형, 인형가게
sd_{17}	겨울, 사무실, 슬리퍼, 미키마우스, 디즈니, 겨울슬리퍼
sd_{18}	경복궁, 맛집, 한우, 한식, 등심
sd_{19}	맛집, 광화문, 해장, 갈비탕, 서울, 한식
sd_{20}	광화문, 해치, 서울, 여행, 경복궁
sd_{21}	한우, 등심, 광화문, 저녁, 회식, 서울
sd_{22}	서울, 저녁, 맛집, 한식
sd_{23}	광화문, 등심, 저녁, 청계천, 시청
sd_{24}	세종문화회관, 먹방, 경복궁, 한식
sd_{25}	세종문화회관, 경복궁, 시청, 청계천, 야경, 맛집

[표 3] 예제 데이터셋

[표 3]는 수집합 인스타그램 키워드 집합 데이터 중 총 25개의 데이터를 설명한 것 이다. 각 데이터는 독립적이며, 키워드는 한국어로 한정하였다. 그리고 각 데이터의 키워드 개수역시 고정되어있지 않다.

Q_i	q_j
Q_1	(서울,1), (여행,1), (광화문,1), (경복궁,1), (한식,1), (한복,1), (한국,1)
Q_2	(겨울슬리퍼,1), (캐릭터,1), (디즈니샵,1)
Q_3	(디즈니,1), (디즈니랜드,1), (미키마우스,1), (미니마우스,1), (일본,1)
Q_4	(경복궁,1), (맛집,1), (한우,1), (한식,1), (등심,1)

[표 4] 무작위로 선택한 각 군집의 초기 대푯값

먼저 초기 값으로 최대 군집의 개수 k의 값은 4이고, 임계값 t는 0.6, 그리고 최대 반복 횟수인 *maxIter*의 값은 50이다. 그리고 Algorithm 1에서처럼 총 k개의 데이터를 무작위로 선택하여 군집의 대푯값으로 한다. 그리고 모든 키워드 집합 데이터에 대하여 가장 유사도가 높은 대푯값을 찾아 라벨링한다. 첫 번째 라벨링 결과는 [표 5]와 같다.

sd_i	$l(sd_i)$	sd_i	$l(sd_i)$
sd_1	1	sd_{14}	3
sd_2	1	sd_{15}	3
sd_3	3	sd_{16}	3
sd_4	1	sd_{17}	3
sd_5	2	sd_{18}	4
sd_6	3	sd_{19}	4
sd_7	3	sd_{20}	1
sd_8	3	sd_{21}	4
sd_9	1	sd_{22}	4
sd_{10}	1	sd_{23}	4
sd_{11}	1	sd_{24}	4
sd_{12}	1	sd_{25}	4
sd_{13}	2		

[표 5] 첫 번째 군집화 결과

모든 키워드 집합 데이터에 대하여 라벨링이 끝났으면 각 군집의 대푯값의 업데이트한다. 먼저 군집 C_1에 속한 $l(sd_i)=1$인 데이터는 총 8개이다. 그리고 이 8개의 키워드 집합 데이터에 속한 모든 키워드는 “서울”, “여행”, “광화문”, “경복궁”, “한

식”, “한복”, “한국”, “커피”, “아메리카노”, “스타벅스”, “서울나들이”, “삼청동”, “테라로라”, “브런치”, “공연”, “가을”, “음악”, “노래”, “비빔밥”, “나물”, “해치”이다. 이중에서 임계값인 0.6을 만족하여 C_1의 대푯값인 Q_1의 속성이 되기 위해서는 키워드의 빈도수가 5회($\lceil t\times 8 \rceil$)이상이어야 한다. 이 중에서 5회 이상인 키워드는 “서울”, “광화문”, “경복궁”이다. 따라서 $Q_1 = \{(\text{서울},6), (\text{광화문},8), (\text{경복궁},5)\}$ 이다. 이러한 방식으로 Q_2에서 Q_4까지 모두 업데이트 하면 다음과 같다.

Q_i	q_j
Q_1	(서울,6), (광화문,8), (경복궁,6)
Q_2	(디즈니샵, 2)
Q_3	(디즈니,7), (미키마우스,5)
Q_4	$\varnothing$

[표 6] 첫 번째 군집화 후 재계산된 각 군집의 대푯값

[표 6]를 보면 C_4의 대푯값인 Q_4는 공집합이다. 이렇게 된 이유는 모든 키워드들이 임계값 조건을 만족하지 못했지 때문이다. C_4에 포함된 키워드 집합 데이터는 총 7개인데 이때 이 7개의 키워드 집합 데이터에 포함된 키워드와 빈도수는 “경복궁” 4번, “맛집” 4번, “한우” 2번, “한식” 4번, “등심” 3번, “광화문” 4번, “해장” 1번, “갈비탕” 1번, “서울” 4번, “여행” 1번, “해치” 1번, “저녁” 3번, “회식” 1번, “청계천” 2번, “시청” 2번, “세종문화회관” 2번, “먹방” 1번, “야경” 1번이다. C_4에는 총 7개의 키워드 집합 데이터가 있기 때문에 대푯값 Q_4가 될 수 있는 키워드는 빈도가 5회($\lceil t\times 7 \rceil$)이상 이어야 한다. 그러나 5회 이상 사용된 키워드가 없기 때문에 Q_4는 공집합이 된다. 대푯값이 공집합이기 때문에 군집 C_4를 제거하고 군집의 개수 k의 값은 4에서 3이 된다. 이렇게 대푯값을 재계산하고 군집의 개수를 조정하였으면 다음단계로 임계값을 업데이트 해준다. 각 군집의 대푯값들의 속성 중 빈도수의 비율이 가장 낮은 경우는 Q_3의 “미키마우스”로 0.625이다. 따라서 임계값 t를 0.6에서 0.625로 업데이트한 뒤에 다시 Q_1, Q_2, Q_3과 비교하며 키워드 집합 데이터를

군집화한다. 다시 반복한 결과는 [표 7]와 같다. 이때

sd_i	$l(sd_i)$	sd_i	$l(sd_i)$
sd_1	1	sd_{14}	3
sd_2	1	sd_{15}	3
sd_3	3	sd_{16}	3
sd_4	1	sd_{17}	3
sd_5	2	sd_{18}	1
sd_6	3	sd_{19}	1
sd_7	3	sd_{20}	1
sd_8	3	sd_{21}	1
sd_9	1	sd_{22}	1
sd_{10}	1	sd_{23}	1
sd_{11}	1	sd_{24}	1
sd_{12}	1	sd_{25}	1
sd_{13}	2		

[표 7] 두 번째 군집화 결과

이때 sd_{19}를 포함한 군집 C_4에 포함되어 있었던 모든 키워드 집합 데이터가 C_1으로 라벨이 변경됐다. 그러므로 다시 군집의 대푯값을 업데이트 하면 다음과 같아진다.

Q_i	q_j
Q_1	(서울,9), (광화문,11), (경복궁,9)
Q_2	(디즈니샵, 2)
Q_3	(디즈니,7), (미키마우스,5)

[표 8] 두 번째 군집화 후 재계산된 각 군집의 대푯값

이 결과를 바탕으로 임계값을 업데이트 하면 그대로 0.625가 된다. 군집의 변화가 있었기 때문에 한번 더 반복을 한다. 추가로 반복한 뒤의 군집화를 하게 되면 두 번째 군집화 결과와 똑같다. 군집화 결과 변화가 전혀 발견되지 않았기 때문에 CASK 알고리즘은 [표 7]의 결과를 반환하며 종료한다.

제 5 장

성능 평가

이 장에서는 본 논문에서 제안한 키워드 집합 데이터를 위한 효과적인 군집화 알고리즘인 CASK의 성능을 평가하기 위하여 키워드 데이터를 군집화 하는 알고리즘인 COSA[6]와 함께 비교 성능 실험을 수행하였다. COSA는 논문에서 실험한 것처럼 k-means를 적용하여 실험하였고, 온톨로지는 [10]에서 제공하는 키워드 온톨로지 정보를 수집하여 사용하였다. 실험은 모두 파이썬(Python)언어로 진행하였으며, 실험은 Core 4 2.5GHz와 32GB의 메인 메모리를 갖는 환경에서 진행하였다. 실험

을 위하여 사용한 데이터는 구글 학술 검색[9]에서 수집한 2014년에서 2015년까지 출판된 논문 50만편의 데이터와 인스타그램[8]에서 수집한 총 50만개의 포스트를 사용하여 실험하였다. 군집화 알고리즘의 실험 결과 정확도를 측정하기 위한 방법으로는 순수도(purity)와 회수도(recall)를 주로 사용한다. 그러나 이는 데이터셋의 모든 데이터가 라벨링 되어있을 경우에만 정확도를 측정할 수 있기 때문에 본 논문에서 실험에 사용한 데이터는 순수도와 회수도를 통한 정확도 측정이 불가능하다. 그래서 정확도 측정을 위하여 라벨링 되어있지 않은 데이터셋의 군집화 결과를 판단하는 Silhouette Coefficient[7]를 계산하여 정확도를 측정하였다. Silhouette Coefficient를 계산하는 식은 다음과 같다.

$$S_i = \frac{D_{\min_i^{OUT}} - D_{avg_i^{IN}}}{\max\left\{D_{\min_i^{OUT}},\ D_{avg_i^{IN}}\right\}}$$

위 식에서 $D_{avg_i^{IN}}$ 은 하나의 데이터 sd_i와 동일한 군집 내의 모든 데이터와의 평균 거리를 의미한다. 그리고 $D_{\min_i^{OUT}}$ 은 어떤 데이터 sd_i와 이 데이터가 속하지 않은 다른 군집의 모든 데이터와의 평균 거리 중 최소값을 의미한다. Silhouette Coefficient는 i번째 데이터 하나에 대해서 S_i를 계산하고 범위는 (-1, 1)이다. 1에 가까운 값이 나왔다는 것은 군집화 결과 군집들이 서로 잘 분리되어 있다는 것을 의미한다. 반대로 −1에 가까운 값이 나오면 한 군집에 서로 다른 라벨의 데이터 들이 혼합되어 있다는 의미이다. 어떤 데이터 d_i에 대한 S_i값을 구한 뒤에 이의 평균을 구하여 정확도를 판단하였다.

5.1 군집의 개수에 따른 성능 평가

군집화 알고리즘의 정확도에 가장 큰 영향을 미치는 것은 군집의 개수이다. 그러나 실제 어플리케이션에서 군집화를 원하는 데이터는 라벨링 정보를 포함한 군집의 정보가 없기 때문에 군집화 개수역시 정확하게 알지 못하는 경우가 많다. 따라서 이번 장에서는 구글 학술 데이터와 인스타그램 데이터를 바탕으로 군집의 개수를 2개에서 100개까지 증가하며 정확도가 어떻게 변하는지 비교하며 확인하고자 한다.

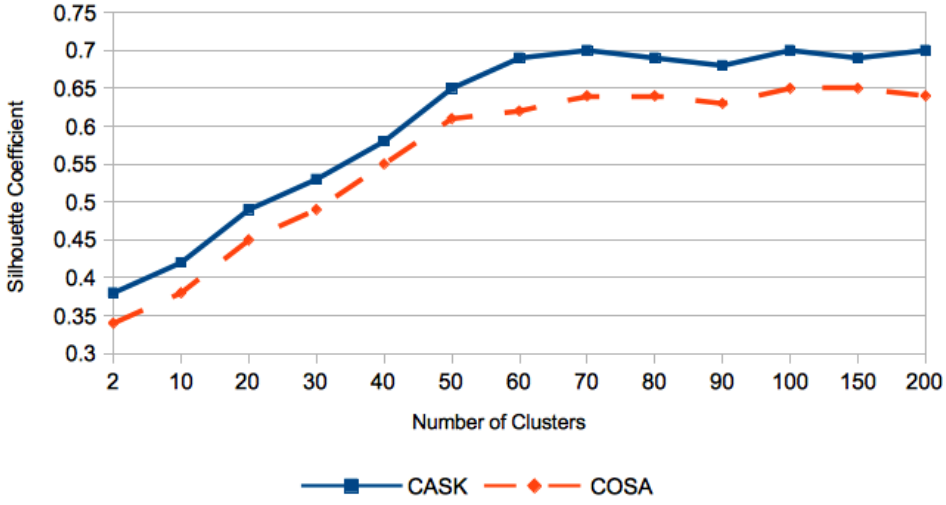
[그림 4](a)는 구글 학술 데이터를 바탕으로 군집의 개수 증가에 따른 정확도를 보여주고 있다. 실험 결과를 보면 군집의 개수가 증가함에 따라서 정확도가 증가하다가 특정 개수 이후에는 일정해지는 것을 확인할 수 있다. 이렇게 군집의 개수가 증가하면서 정확도가 올라가는 이유는 군집의 개수가 적으면 하나의 군집에 속하는 데이터들간의 유사도가 매우 낮아지고 또한 군집에 속하지 않게 되는 데이터가 나온다. 따라서 군집의 개수가 증가하면서 정확도가 증가하다가 적정 개수 이후로는 유지되는 경향을 보인다.

[그림 4](a)를 보면 COSA의 경우 k가 50일 때부터 일정해지는데 CASK는 60에서부터 일정해진다. 이는 CASK는 초기에 입력받은 군집의 개수가 고정된 값이 아니라 최대값을 받은 뒤에 군집의 개수를 줄이면서 군집화 하기 때문에 이러한 차이가 나타난다. 또한 실험결과 CASK는 60이후 군집화 결과 군집의 개수가 모두 55 ~ 58개 사이로 나왔다. 이 값은 다른 알고리즘에서도 적절한 군집의 개수임이 실험을 통해서 확인 되었다. 따라서 CASK는 다른군집화 알고리즘 보다 정확도도 높으면서 적당히 큰 군집의 최대 개수가 초기에 주어지면 적절한 군집의 개수로 알아서 찾아가는 것을 확인할 수 있었다.

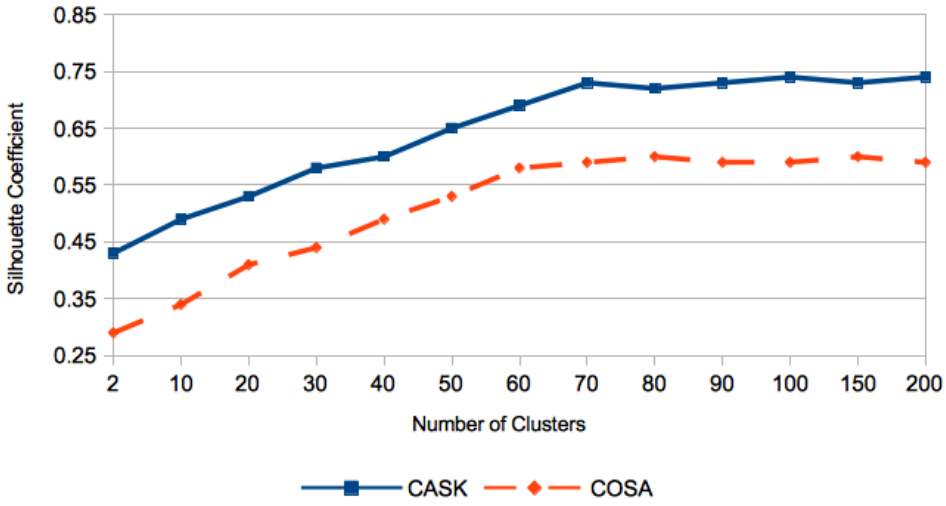
이러한 경향은 [그림 4](b)에서도 확인할 수 있었다. CASK를 통해 군집화 한 결과를 보면 k가 70 이후에는 일정하게 유지되는 사실을 확인할 수 있다. 그리고 구글 학술 데이터를 활용한 실험과 마찬가지로 70 이후에는 군집화 결과 군집의 개수

가 63 ~ 67로 실제 다른 군집화 알고리즘에서 최적의 군집 개수로 판단할 수 있는 값이 나왔다.

이번 실험은 군집의 개수에 따른 정확도를 비교 분석하는 실험이었다. 그 결과 CASK는 정확한 군집의 개수 없이도 적절히 큰 값이 주어지면 다른 알고리즘 보다 정확하게 군집화가 가능하다는 사실을 확인하였다. 또한 군집의 개수에 상관없이 상한 CASK의 정확도가 높았는데 이 이유는 다음 장에서 자세한 실험 결과를 바탕으로 자세하게 설명한다.



(a) 구글 학술 데이터



(b) 인스타그램 데이터

[그림 4] 군집의 개수에 따른 성능평가

5.2 실제 데이터를 통한 정확도 평가

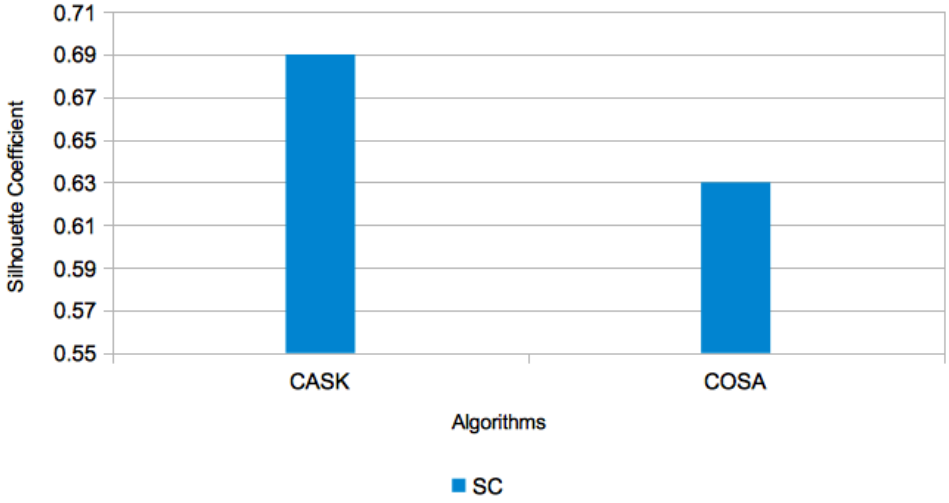
실험에 사용한 데이터셋들은 모두 군집의 개수도 알 수 없다. 그런데 CASK, COSA 모두 초기에 군집의 개수가 반드시 필요하다. 그렇기 때문에 먼저 CASK의 k값을 100으로 하여 실험한 뒤에 CASK 결과 나온 군집의 개수를 COSA의 k값으로 하여 실험하였다. CASK는 실험을 위해서 초기에 최대 군집의 개수를 100, 임계값을 0.6으로 하여 실험하였고, 그 결과 [그림 5]과 같은 정확도가 나왔다.

[그림 5](a)는 구글 학술 데이터를 군집화 한 결과이다. CASK를 통해서 군집화 한 결과 최종 군집의 개수는 53개가 나왔다. 이렇게 나온 값인 53을 COSA의 초기 k값으로 하여 실험을 진행한 결과 [그림 5](a)와 같이 나왔다. [그림 5](a)에서의 정확도를 보게 되면 CASK, COSA 순으로 정확도가 높은 것을 확인할 수 있다. 이렇게 나오는 이유는 CASK는 키워드 집합 데이터의 모든 키워드를 다 고려하며 반복이 진행될수록 임계값을 통해 군집의 대푯값을 좀 더 강하게 만든다. 즉 유사도가 높은 데이터를 하나의 군집에 더 잘 모아준다. 반면 COSA는 온톨로지 기반이기 때문에 온톨로지의 정보가 없는 키워드는 정확하게 고려하지 못하기 때문에 누락되는 정보가 많아지고 이로인하여 CASK보다 정확도가 낮아졌다.

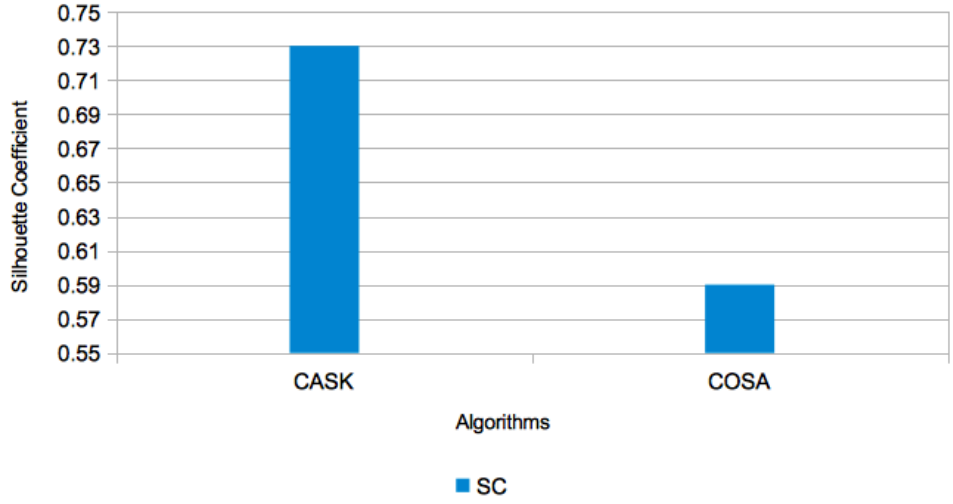
[그림 5](a)와 [그림 5](b)의 COSA결과를 보면 [그림 5](b)의 COSA의 정확도가 더 낮은데 이는 인스타그램의 키워드가 훨씬 다양하여 온톨로지에 반영되지 않은 키워드가 많기 때문이다.

[그림 5](b)는 인스타그램 데이터를 군집화 한 결과를 나타내고 있다. CASK를 통해 군집화 한 결과 최종 군집의 개수는 67개가 나왔다. 실험 결과를 보면 [그림](a)와 동일한 결과가 나타났다. [그림 5](a)와 [그림 5](b)를 비교하면 CASK와 다른 알고리즘 간의 정확도 차이가 인스타그램 데이터를 활용한 [그림 5](b)에서 더 확연하게 나타났다. 이는 CASK는 다양하고 새로운 키워드가 존재하는 데이터역시 정확하게 군집화 하는 반면 COSA는 온톨로지를 활용하기 때문에 다양한 키워드를 처리하

지 못한다는 한계를 확인할 수 있다. 이번 실험을 통해서 CASK가 키워드가 비교적 정형화된 학술 데이터 뿐만 아니라 키워드의 종류가 다양한 인스타그램 데이터에서도 다른 군집화 알고리즘 보다 정확하게 군집화 한다는 것을 확인하였다.



(a) 구글 학술 데이터



(b) 인스타그램 데이터

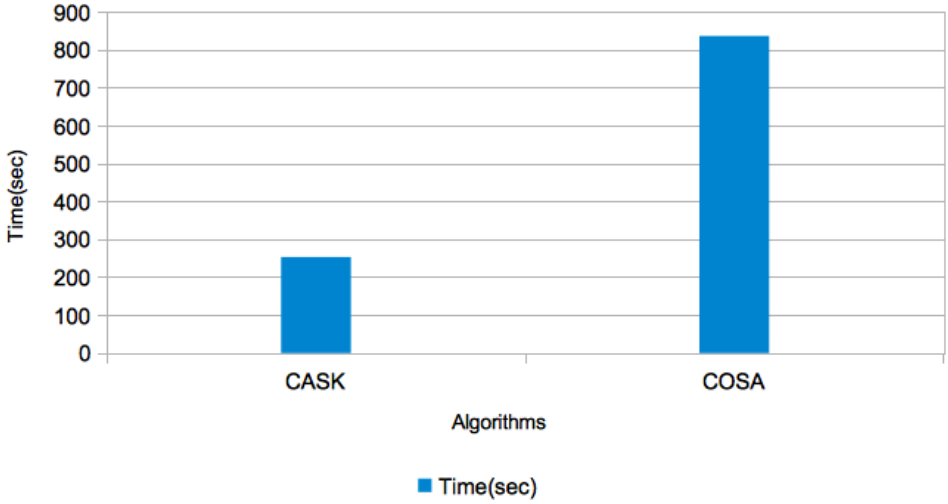
[그림 5] 실제데이터를 활용한 정확도 실험 결과

5.3 군집화 소요시간

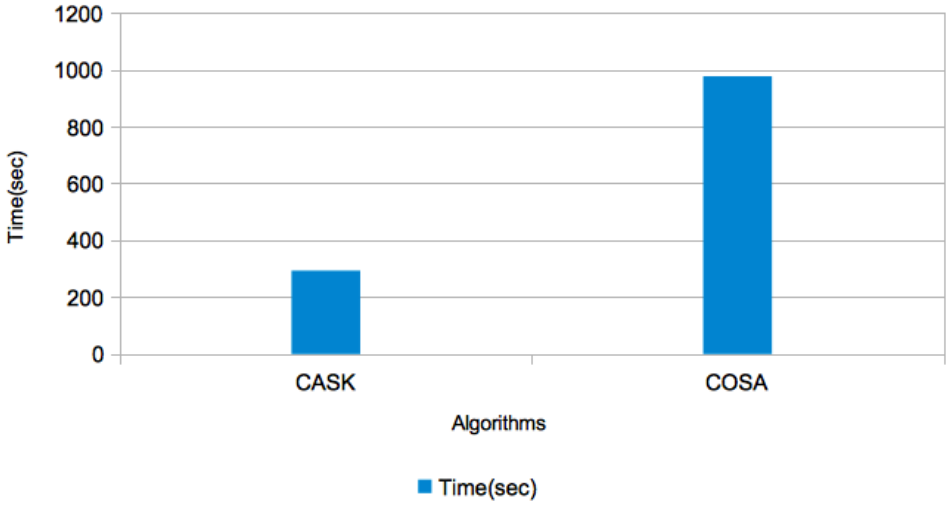
이번 장에서는 군집화하는데 걸리는 총 소요시간에 대한 실험 결과에 대하여 논의하겠다. [그림 6](a)와 [그림 6](b)는 각각 구글 학술 데이터와 인스타그램 데이터를 군집화 하는데 걸리는 시간을 나타내고 있는 그래프이다.

CASK와 COSA를 비교하면, CASK가 구글 학술 데이터, 인스타그램 데이터 모두 2~3배이상 속도가 빨랐다. 이는 COSA는 온톨로지를 사용하기 때문에 군집화를 위한 시간에 데이터를 처리하는 추가적인 시간이 소요되었기 때문이다. 또한 온톨로지에 없는 키워드가 생기다 보니 데이터의 크기도 커지고 k-means가 반복되는 시간이 증가하여 시간이 많이 걸렸다.

반면 CASK는 구글 학술 데이터는 총 4번, 인스타그램 데이터는 총 6번 반복 후 알고리즘이 멈췄다. 임계값을 사용하여 군집의 대푯값을 강화하고, 이로인해 유사도가 높은 데이터가 빠르게 같은 군집에 속하도록 해주기 때문에 빠른 시간에 군집화 결과를 얻을 수 있었다. 실험 결과를 보면 모든 알고리즘이 인스타그램 데이터를 군집화 하는데 더 많은 시간이 걸리는데 이는 데이터의 양이 많고 키워드의 종류가 다양하기 때문에 군집의 개수도 더 많기 때문이다. 여기서 중요한 점은 정확도 때와 마찬가지로 속도 역시 인스타그램 데이터에서 더 큰 차이를 보이고 있다. 이는 COSA가 다양하고 새로운 키워드를 잘 다루지 못하기 때문이다.



(a) 구글 학술 데이터



(b) 인스타그램 데이터

[그림 6] 수행 속도 비교 실험 결과

제 6 장

결 론

본 논문에서는 다양한 종류의 키워드를 원소로 하는 집합 형태의 키워드 집합 데이터와 이를 위한 유사도 기법을 새롭게 정의하였다. 그리고 이를 효과적으로 군집화 하는 알고리즘을 제안하였다. 최근 데이터는 키워드 공간이 무한대이기 때문에 다양한 종류의 키워드를 고려할 수 있도록 키워드와 빈도수로 이루어진 벡터를 원소로 하는 키워드 집합 데이터를 정의하였다. 그리고 이러한 키워드 집합 데이터로 이루어진 군집을 대표하는 대푯값 역시 새롭게 정의하였고, 이들 간의 유사도를 측

정하는 측정법 역시 새롭게 정의하였다. 그리고 이를 사용하여 정확한 군집의 개수 없이도 군집화가 가능한 CASK 알고리즘을 제안했다.

기존의 알고리즘들은 차원이 고정되어 있는 범주속성 데이터만 가능하거나 키워드 데이터를 백오브워드 혹은 온톨로지를 활용하여 군집화한다. 그렇기 때문에 키워드의 종류가 다양하고 차원이 고정되어 있지 않은 최근의 키워드 기반 데이터를 군집화 하지 못한다.

그러나 본 논문에서는 키워드 집합 데이터라는 새로운 데이터 정의를 바탕으로 하여 고정된 차원이 아닌 데이터를 군집화 할 수 있도록 해줬다. 또한 새로운 유사도 측정법을 고안하여 키워드 공간의 제약 없이 다양한 키워드를 고려하여 유사도를 측정할 수 있다. 이를 바탕으로 제안된 키워드 집합 데이터 군집화 알고리즘인 CASK는 키워드의 종류가 다양하고 차원이 고정되어 있지 않은 데이터를 빠른 시간에 정확하게 군집화 한다.

본 논문에서 제안한 CASK를 기존의 군집화 알고리즘과 실제 데이터를 사용하여 비교 분석한 결과 키워드가 비교적 정형화 되어있고 종류가 한정적인 구글 학술 데이터를 활용한 실험에서는 COSA 보다 0.05 높게 나왔다. 또한 키워드가 매우 다양한 인스타그램 데이터에서는 COAS보다 0.13 높은 정확도를 보였다. 군집화 속도 역시 온톨로지를 기반으로 하는 COSA보다 매우 빠르게 군집화가 이루어졌다. 또한 정확한 군집의 개수가 필요한 다른 알고리즘과는 다르게 CASK는 적당히 큰 값이 초기에 주어면 정당한 개수의 군집으로 군집화가 가능했다. 이는 군집의 개수를 알기 힘든 실제 데이터를 보다 효과적으로 군집화 할 수 있다는 점에서 매우 큰 장점이다.

현재는 단일 장비에서 군집화가 가능한 알고리즘을 제안하였다. 향후 이를 바탕으로 하여 분산 환경을 활용한 군집화 알고리즘을 연구하면, 보다 빠르고 정확하게 빅데이터를 군집화할 수 있을 것으로 보인다.

참고문헌

[1] Huang, Z; Ng, MKP, “A fuzzy k-modes algorithm for clustering categorical data”, IEEE Transactions on Fuzzy Systems, v. 7 n. 4, p. 446-452, 1999

[2] Takamu Kaneko, Keiji Yanai, "Event photo mining from Twitter using keyword bursts and image clustering", Neurocomputing 172, pp. 143-158, 2016.

[3] Oren Tsur, Adi Littman, Ari Rappoport, "Efficient Clustering of Short Messages into General Domains", ICWSM, 2013

[4] Saeyoung Kim, Jaehwan Lee, Sehee Lee, Sungwon Jung, “A MapReduce-based Clustering Algorithm for Social Media Data with Tags”, KIISE, 2015

[5] Feng, Wei, et al. "STREAMCUBE: hierarchical spatio-temporal hashtag clustering for event exploration over the twitter stream." 2015 IEEE 31st International Conference on Data Engineering. IEEE, 2015.

[6] Hotho, Andreas, Alexander Maedche, and Steffen Staab. "Ontology-based text document clustering." KI 16.4 (2002): 48-54.

[7] Aggarwal, Charu C. Data mining: the textbook. Springer, 2015.

[8] http://www.instagram.com

[9] http://scholar.google.co.kr

[10] http://www.daml.org

[11] Fahad, Adil, et al. "A survey of clustering algorithms for big data: Taxonomy and empirical analysis." IEEE transactions on emerging topics in computing 2.3 (2014): 267-279.

[12] Tang, Xiaoyu, and Qingtian Zeng. "Keyword clustering for user interest profiling refinement within paper recommender systems." Journal of Systems and Software 85.1 (2012): 87-101.

[13] Boriah, Shyam, Varun Chandola, and Vipin Kumar. "Similarity measures for categorical data: A comparative evaluation." red 30.2 (2008): 3.

[14] Xu, Rui, and Don Wunsch. Clustering. Vol. 10. John Wiley & Sons, 2008.

[15] Feng, Wei, et al. "STREAMCUBE: hierarchical spatio-temporal hashtag clustering for event exploration over the twitter stream." 2015 IEEE 31st International Conference on Data Engineering. IEEE, 2015.

[16] Sakaki, Takeshi, Makoto Okazaki, and Yutaka Matsuo. "Earthquake shakes Twitter users: real-time event detection by social sensors." Proceedings of the 19th international conference on World wide web. ACM, 2010.

[17] Bollen, Johan, Huina Mao, and Xiaojun Zeng. "Twitter mood predicts the stock market." Journal of Computational Science 2.1 (2011): 1-8.