Ⅰ. Introduction

Advent and development of Internet technology greatly affected human society. Internet technology made production and consumption of data easier. This affected the whole environment of media including news media. Before wide spread of internet, only mass consumption of media was available through unidirectional transmission of signal. However, as writing and sharing of data became easier, mass production also becoming available as era of internet began.

Also, Internet technology enhanced dependency of news media. In the past, news media were distributed by producers. For example, broadcasting station creates its own news and broadcasts it. On the internet however, news data are typically distributed by portal website except for some big and authorized publishing corporation (Wi-Geun Kim, 2014). And this greatly affected journalism ecosystem.

Of course, portal playing a role of distributor cuts expense for producing news since producers have no concern on distribution expense. Due to reduction of cost, news media once were exclusive and gate-keeping, now started to become gate-opening (Kyungmo Kim, 2012). As a result, news creators increased, and news media became exceedingly abundant.

Due to changed environment, there are lots of news sources and contents on web nowadays. And most of them are dependent to portal websites (Yeo-kwang Yoon, 2104). Dependency of news creators on portal website created power imbalance between producers and distributor (Haeyeop Song, Jay Yang, 2017). From power imbalance, producers had to cater the distributors' need. On website, more click means more money. So, all those news producers tend to create short, quick news to compete with their competitors; to get more clicks from users. "Competition of click" led to creation of abundant and low-quality news contents focusing on temporary issues. Therefore, with online news it's becoming harder to see the whole picture of an event. For example, there are lots of news regarding "Burning sun" in South Korea. "Burning Sun" is a club in Gangnam, South Korea. "Burning Sun" started to get spotlight by assault incident. And problem got worse as more scandals like "Selling drug", "Sexual favors", "Hidden camera", "Back-scratching relationship between politician and police" were revealed to public (Joan, 2019). Since "Burning Sun" scandal has so many sub-issues, it's actually impossible to follow what happened from start since news media only focuses on particulated "sub-issues" of "Burning sun". There is a test case with this "Burning Sun" issue and would be able to check out how much this algorithm helps to understand the "Burning Sun".

There is a need for news contents that tells more specific stories about certain event. These kinds of news are called "story telling news" (Kim, Ji-Yeon, Yun and Jae Young, 2015). To suit those needs, this research proposes "Timeline news algorithm", sewing particulated news into one whole story line. "Timeline news algorithm" uses "linear storytelling" structure (KANG EUNYOUNG, 2010) that was suggested by journal above. "Linear story telling" structure is a structure that tells what happened in time series. Also, it uses Open API made by Korean Press Foundation to get several news data. Algorithm covers the "sub-issues" from time to time about

certain events and reader may be able to follow with updating issues and see the whole picture of event. There is a test case of this algorithm with keyword "Burning Sun" and would be able to evaluate how much it would help the users.

"Timeline news algorithm" collects related keywords and its weights with given word on daily basis and compares related keywords between dates. If keywords between two dates overlaps much, this algorithm considers those dates deal with same "sub-issue". If keywords between two dates don't overlap much, this algorithm considers "sub-issue" changed and divides the time interval. After all, multiple time intervals will be made and search news with related keywords on time interval basis.

Ⅱ. Algorithm Design

1. Creating a Story line of a News Keyword

Getnews function works as main function for creating timeline news. This function returns timeline news data. To be more specific, there will be multiple time intervals and there will be 3 related keywords and news articles in each time interval. Timeline is divided as new affair arises in regard to keyword; dividing phase.

This function will return given number (option) of time intervals. If you set option as -1, function will return all the chunks it found. This ability was made to prevent the timeline from being long-winded. If you use this option, function will return period chunks that were spotlighted most by using sum of each time interval's weight sum.

To follow the flow of function, refer Fig 1 and Table 1. After **keywordextract** function in Fig1, will return list of elements and each element has date and relevant keyword of "Burning Sun". Of course, related keywords will change as new sub-issues are posted. After **Phasedivide** function, all the dates will be combined to multiple numbers of time intervals. Intervals are made if adjacent dates seem to have same sub-issue. Within a single time interval, since dates in the interval deal with common issue, it is likely that interval contains same related keywords from multiple dates. To handle this, **Keyword_Integrate** function combines overlapping related keyword within time interval (weights will be added).

If rephrasing option is set to -1(off), then algorithm will jump to **NewsSearch** function. **NewsSearch** will search news for every time interval. When searching news, time interval, KEYWORD, 3 related keywords of the interval will be given. By adding 3 related keywords, search result will reflect the sub-issue of that period.

If there is a rephrasing option (positive integer N), algorithm will add up all the related keywords' weight in each time interval and compare the value of weight sum of each interval. N time intervals with highest weight sum will be the input of **NewsSearch** function.

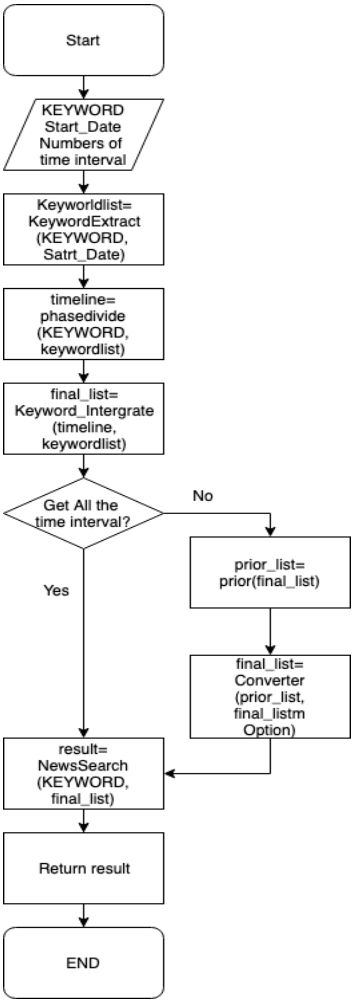

Figure 1. Flow of whole algorithm

Parameter:

KEYWORD: Single phrase. Must be phrase that one wants to know about.

START_DATE: Date that one wants to start_date search about keyword.

OPTION: Timeline option. Can limit the number of phases. -1 to set this option off.

Table 1. Creating "timeline news"; main function

1.GetNews function
Input: KEYWORD, START_DATE, OPTION
1: keywordlist = **keywordextract**(KEYWORD,START_DATE)
2: timeline = **phasedivide**(KEYWORD, keywordlist)
3: final_list=**keyword_Integrate**(timeline, keywordlist)
4: if (OPTION != -1)
5: prior_list=**prior**(final_list)
6: input_list=**converter**(prior_list,final_list, option)
7: article=**NewsSearch**(KEYWORD, input_list)

```
 8: else
 9:     article=NewsSearch(KEYWORD, final_list)
10: return article
```

2. Extracting Related Keywords of Date

This is the function that collects related daily keywords and its weights. This function used "timeline API" and "wordcloud API". "Timeline API" checks number of news posted on one day with given KEYWORD. "Wordcloud API" uses both "Term Frequency" algorithm and "Inverse Document Frequency" algorithm to get related keywords and its weight. And date with less than 10 news with KEYWORD was not included (Considering that date has less or no relevance with KEYWORD). Also getting rid of dates like this will enable timeline to end if case is closed.

*At Table 2, **detect function** will return dates in list format. If less than 10 news were posted with KEYWORD, that date will not be included in the list. This function uses "timeline API" in Table 3.

*At table 2, **getwordcloud function** collects related keywords and its weight. This function uses "wordcloud API" in the Table 4.

Table 2. Extracting relate keywords on daily basis

2.Keywordextract function

```
    Input: KEYWORD, START_DATE
1: labels=detect(KEYWORD, START_DATE)
2: final=[]
3: for i in labels:
4:     date=get date from i
5:     result=getwordcloud(KEYWORD, date)
6:     final.append(result)
7: return final
```

2-1. Detecting Relevant Dates with Keyword

Use "timeline API". The variable timeline will have list object with dates and number of new. posted as a return value of API request.

Table 3. Getting number of news posted with given KEYWORD "timeline API"

2-1.detect function

```
    Input: KEYWORD, START_DATE
1: timeline = timeline API request(KEYWORD, START_DATE to today)
2: result = []
3: for i in timeline"
4:      if (news posted >= 10):
5:          result.append(i)
6: return result
```

2-2. Getting Related Keyword with word cloud API

This function uses "wordcloud API". Function will return list of elements. Each element has date and related keywords and its weights.

Table 4. Getting related keywords using "wordcloud API"

2-2. getwordcloud function

```
    Input: KEYWROD, date
1: wordcloud = wordcloud API request(KEYWORD, date)
2: wordcloud = sort related keywords in descending order with weights
3: if(more that 10 keywords):
4:          remove rest of keywords
5: return wordcloud
```

3. Merging Dates with relevant "sub-issue" in time interval

Reminder: keywordlist is list of "dates, related keywords and its weights". Related keywords on certain date will be sorted in descending order of weight.

At line 3 of Table 5, Phasedivide function iterates through the list elements and checks if top 3 related keywords of date "t" is in the related keywords of date "t-1". If all 3 related keywords are not in the related keywords of date "t-1", date t is classified as new time interval. By comparing related keywords, algorithm tries to

Goal of this function is to divide whole events into minor stages (or sub-issues). This will enable users to see the whole event in several significant chunks. This will enhance understanding of whole event. To give an example with "Burning Sun", this algorithm should distinguish the dates into two time intervals if main sub-issue changed from "hidden camera" to "drug selling" as new scandal arises.

Table 5. Merging dates into time intervals with relate keywords of each date

3. Phasedivide function

```
    Input: keywordlist
 1: initialize counter to 7 and initialize resultlist=[];
 2: for i in keywordlist:
 3:     if (one of i th date's top 3 keyword is in the i-1 the date's keywordlist):
 4:         reduce counter
 5:         if (counter is zero):
 6:             resultlist.append(i th date)
 7:     else:
 8:         resultlist.append(i th date)
 9:         reset counter
10: return resultlist
```

4. Integrating Daily Related Keywords into period related keywords

timeline is return value of phasedivide function (previous function). Goal of this function is to integrate daily dates and keywords into time interval unit keywords and dates. Therefore, return value will be a list of periods (not date) and that period's related keywordlist. In this process, list of multiple dates" related keywordlists will be integrated and ordered in descending order of weight again.

For example, if there is a time interval from 2019-01-01 to 2019-01-03.
2019-01-01's related keywords were {"a":132.1, "b":33 }
2019-01-02's related keywords were {"b":100, "c":7.3 }
2019-01-03's related keywords were {"b":150.4 , "a":14.1 }
Then, this time interval will be 2019-01-01 to 2019-01-03 related keywords of {"b":283.4, "a":146.2, "b":133}

Table 6. Integrating multiple daily related keywords into time interval related keyword

4. Keyword_Integrate function

```
    Input: timeline, keywordlist
 1: result=[]
 2: for i in range(1, len(timeline)):
 3:     start = timeline[i-1]
 4:     end = timeline[i]
 5:     weight={}
 6:     for j in range(start+1, end, 2):        #temp will be the daily keywords
```

```
 7:          temp=keywordlist[j]                #and weights between start and end.
 8:          for k in range(0, len(temp)):       # iterates through daily keywords
 9:              if (temp[k] in list):
10:                  add weight(value) to pre-existing keyword key
11:              else:
12:                  add new keyword(key)
13:      sort key values of weight in descending order of weight
14:      result.append(weight)
15: return result
```

5. Ranking Period Blocks with Sum of Weight Value

Each time interval's weights of keyword are added in this function temporarily. Then function will compare the sum value of multiple number of intervals of weight and rank time intervals. Weight is a value that reflects how much a certain word was referred with given KEYWORD. So, related keyword with higher value can be seen as frequently mentioned sub-issue. So, weight could be interpreted as peoples' interest about certain keyword. By comparing sum of these value, function will identify which time interval's keyword had more attention. With this, algorithm will prioritize the multiple time intervals considering higher weight sum dragged more attention to public.

If there is OPTION value of positive integer (rephrasing function), algorithm will go through this function to check which time interval to return. Time interval block with relatively low sum value would be excluded at return value.

This will be executed through two functions; Prior() and Converter() at Table 7 and 8. Prior() will add the weight values and Converter() will delete the time intervals that are not in the rank.

Return value of Keyword_Integrate function becomes the input value of Table 7's function.

Table 7. Adding all the weight of related keywords within each time interval

5. Prior function

```
    Input: final_list
 1: prior=[None]* len(final_list)
 2: for i in range(0, len(final_list)):
 3:      if(i==even):
 4:          prior[i] = final_list[i]     #copy period value
 5:      else:
 6:          prioir[i] = sum of weights in final_list[i]
 7: result =copy "prior"
 8: for i in range(0, len(result_list))
```

```
 9:     set first(1st value) and last(len(result_list)th value) to 0
10:     rank sum of weight and replace sum with its ranking
11: return result
```

6. Deleting Period Blocks that are not in the rank (option)

Prior_list is return value of prior function at Table 7. Like mentioned above, this function deletes the time intervals that are not in the rank.

Table 8. Remaking final_list by prioritizing time interval

6. Converter function

```
    Input: prior_list, final_list, option
 1: iterator = len(final_option)
 2: iterator = change to option if option is smaller than len(final_option)
 3: result_list=[None]*(2*iterator)          #initilalize list with enough size
 4: save first two element of final_list to first two space of result_list
 5: save last two element of final_list to last two space of result_list
 6: index=3
 7: for i in range (3, len(final_list)-2, 2):
 8:     if(prior_list[i] < number-2):          #checks if ranking of each period suits the option
 9:         result_list[index] = final_list[i]
10:         result_list[index-1] = final_list[i-1]
11:         index+=2
12: return result_list
```

7. Searching News with Given Period, Keywords

This function uses "Search API" in the line 9 of Table 7. Purpose of this function is to search news with each time interval. NewsSearch function will search news with given period, KEYWORD, three relate keywords with big weight. Then, news that contains those keywords will be found. News that contain those keywords well are considered to reflect time interval's sub-issue.

Table 9. Searching News with given input

7. NewsSearch function

```
    Input: KEYWORD, final_list
 1: start=""
 2: end=""
 3: for i in final_list:
 4:     if (i is date):
 5:         start = period(i)'s start date
 6:         end = period(i)'s end date
 7:     else:
 8:         word = [KEYWORD, period(i)"s top 3 keywords]
 9:         article = search(start, end, words)
10:         result.append(article)
11: return result
```

7-1. Searching News with Searching API

Used "Search API" to search for news with given period and keywords. In this algorithm, API request is sent with period, KEYWORD, other three keywords. Then API is set to return most related 2 news. When doing test case, all other information like publisher and name of reporter was excluded since the core purpose of this algorithm is to understand what happened (contents of news were what mattered the most when understanding a topic).

Ⅲ. Case Study

Flow of Algorithm

To give a bird's eye view, about this algorithm, algorithm gets KEYWORD(that you want to know about), Start_Date, Option as input. Option has rephrasing ability and if you set option on (typing positive integer), algorithm will return that number of period chunks. Setting -1 on option means getting all the period chunks as return value.

Algorithm will check how many news were posted on portal website starting from given date. Then, algorithm will gather all the dates that more than 10 news were posted considering those date had some issue about given keywords.

With stored dates, algorithm will collect daily related keywords and weights with given keyword. With those date, algorithm will check if day(i)'s top ranked related keyword is in day(i-1)'s related keywords. If there are overlapping keywords between the two, algorithm considers those two dates deals with same "sub-issue" and bind those two days into single period (or chunk).

After this process, algorithm merges all the related keywords of multiple dates (single period) into single list. Then it chooses top 3 related keywords of each period based on weights. If there is an option input of positive integer, algorithm adds all the weights of

each period and ranks them to find periods with that input range.

Lastly, algorithm searches news with given keyword and top related keywords in the period. There are 2 test cases with keyword "Burning Sun". First case result is input of algorithm with "Burning Sun" starting from Jan/1st/2019 and option value of 10. Second case result is input of algorithm with "Burning Sun" starting from Jan/1st/2019 and option value of -1.

About test cases

There are two test cases about "Burning Sun" below. First one is making timeline without rephrasing. Second one is making timeline with 10 time intervals. Return value of this algorithm should have multiple time intervals with 3 related keywords and 2 news articles each. But for several reasons, article was skipped in this paper. Algorithm was set to return two news articles each that contains keywords well. And program was set to return only two articles because search result (news articles) seems to overlap and no need to store many news.

Firstly, the purpose of test case is to check if this algorithm reorganizes news well for the understanding of an issue (using timeline). And article is the simple result of news search using KEYWORD and related keywords. So to check the effectiveness of this algorithm, it is important to check the related keywords of each time interval. Secondly, since each time interval has two articles each, it takes up too much space to write on this paper. With following test case results below, there are evaluations about each test case.

1. Case 1

Input: 버닝썬("Burning Sun"), 2019-01-01, -1

Table 10 is the result of given input above (without news article). According to table 10, there were 35 time intervals in total until the test day. This means that "Burning Sun" can be divided into 35 meaningful intervals since each intervals has similar sub-issue in it.

Table 10. Return value of Input "Burning Sun" with no option (Only keywords)

Period	Keywords	Period	Keywords
1	폭행 사건, 성추행, 성폭행	19	멤버 승리, 정준영, 삼합회
2	성관계 동영상, 성폭행, 이문호 대표	20	정준영, 연예인, 마약 투약 혐의
3	성관계 동영상, VIP룸, 참고인 신분	21	현직 경찰관, 중고차, 전직 경찰
4	마약 투약, 애나, 유착 의혹	22	정준영, 성관계 동영상, 김상교
5	경찰관, 유착 의혹, 압수수색	23	유리홀딩스, 전원산업, 가수 승리
6	성접대, 이문호 대표, 사내이사	24	이승현, 몽키 뮤지엄, 가수 승리

7	성범죄, 해시태그, 물뽕	25	마약 투약 혐의, 구속 영장, 애나
8	이문호 대표, 유착 의혹, 마약 투약	26	JYP, 엔터테인먼으 업종 주가, 가수 승리
9	유착 의혹, 마약 투약, 탈세 의혹	27	성폭행, 정준영, 박한별
10	성접대, 성범죄, 각종 의혹	28	마약 투약 혐의, 전원산업, 이문호
11	이승현, 아레나, 성접대 논란	29	장자연 사건, VIP, 등지 소각팀
12	정준영, 유착 의혹, 경찰총장	30	경찰 안팎, 경찰청, 마약대응 조직
13	현직 경찰관, 피의자, 직무유기 혐의	31	연예인들, 구속영장, 불법촬영
14	김학의, 구속 영장, 정준영	32	전원산업, 정준영, 관련자
15	폭행 사건, 경리실장, 경리업무	33	성매매, 외국인 마약사범, 집중 단속
16	린사모, 김학의, 지창욱	34	유리 홀딩스 대표, 주점 몽키뮤지엄, 브랜드 음료 사용료
17	유착 의혹, 로스쿨 교수들, 서강대	35	한효주, 화장품 브랜드, JM솔루션
18	김학의, 유착 의혹, 황하나		

Test Case Evaluation

In the table 10, each interval has at least on unique related keyword comparing with adjacent time intervals. From this, it can be inferred that unique keyword represents the new event of that interval. And first related keyword can be seen as mostly spotlighted keyword during that period.

However, there are some overlapping keywords between intervals. This is inevitable result since people could be still interested in the past sub-issue along with new sub-issue. In this case both new keywords and old keywords can be on a list. It is also inevitable because this algorithm reorganizes the news articles that are already posted. This means that it's quite unrealistic to expect time period of news articles to be divided cleanly.

However, it's very clear that return result (including articles) is more helpful to understand about whole event than normal, scattered news articles. This algorithm at least removes unnecessary overlapping new articles and organizes similar sub-issues which enhances users' understandability.

2. Case 2

Input: 버닝썬("Burning Sun"), 2019-01-01, 10

Table 11 is result of algorithm with given input mentioned below. Period with most weight (8 chunks) were selected from 2nd to 34th time interval (In Table 10, there are 35 time intervals). And first and last time intervals were included without ranking them to make full

10 time intervals of news "sub-issues". The reason why first and last time intervals were added is to show the starting point (or starting scandal) of KEYWORD and to keep the recent "sub-issue" updated.
Input: 버닝썬, 2019-01-01, 10

Table 11. Return value of Input "Burning Sun" with top 10 option (Only keywords)

Time Interval	Related Keywords
1	폭행 사건 (Assault case), 성추행(Sexual harassment), 성폭행(Sexual assault)
2	마약 투약(Drug administration), 애나(Anna), 유착 의혹(Suspicion of collusion)
3	경찰관(Police officer), 유착 의혹(Suspicion of collusion), 압수수색(Search and seizure)
4	성접대(Sexual favor), 이문호 대표(CEO Lee), 사내이사(In-house director)
5	이문호 대표(CEO Lee), 유착 의혹(Suspicion of collusion), 마약 투약(Drug administration)
6	정준영(Jeong Jun-young), 유착 의혹(Suspicion of collusion), 경찰총장(Police chief)
7	김학의(Kim Hak Yi), 구속영장(Arrest warrant), 정준영(Jeong Jun-young)
8	린사모(Linsamo), 김학의(Kim Hak Yi), 지창욱(Ji Chang-wook)
9	마약 투약 혐의(Suspicion of drug administration), 구속영장(Arrest warrent), 애나(Anna)
10	한효주(Han Hyo-joo), JM솔루션(JM Solution), 광고모델(Advertising model)

Test Case Evaluation
Table 11 is result of using KEYWORD "Burning Sun ". Since this algorithm with "option 10" only ranks the time intervals with same KEYWORD, relate keywords of time intervals can be also found in Table 10. Third time interval of Table 11 is same with fifth interval of Table 10. All other related keywords of each interval can be found in Table 10 since Table 11 is just rephrasing of table 10. But last time interval (most recent sub-issue) is different because test case2 was done on different date and therefore, most "recent" interval may slightly differ as new news were kept updated.
Other than that, effectiveness of result seemed pretty much similar with test case 1. But to compare characteristics between the two (One with option and one without option), One with option is much shorter than one without option so it would save time. But some issues may be lost because algorithm tried to pick the most spotlighted sub-issues.

3. Case Discussion

Basic goal of this algorithm is to see the whole story line of given keyword topic. To suit this need, algorithm should detect all the "sub-issues" with periodic units. Also, unit should not be too similar each other because it means it has multiple chunks with same "sub-issue".

Based on those two cases, this algorithm seems to be partially successful. There were no periods with same top keywords in the result. Although table11's 20th and 22nd's period has same top keyword, we could say that "sub-issue" slightly changed since other two top related keywords differs.

Since result value completely depends on the news data, it's hard to make clear story line news like chronology. But strength of this algorithm is that it can sensitively reflects what people were interested at that period since algorithm uses weight to process(sort) the related keywords. Also even though it doesn't give timeline news like chronology, it's definitely helps to understand the whole flow of an event without searching or looking at all the news all along.

IV. Reference

Rearrange in alphabetic order!

Haeyeop, Song Jay, Yang 2017, *Online News Portal Service and changes in News Distribution: Big data Analysis of Naver News in 2010-2017*, Korean Journal of journalism & Communication Studies 61(4), Available at: http://www.dbpia.co.kr/journal/articleDetail?nodeId=NODE07230005

Joan, M 2019, *How The Burning Sun Scandal Affected Korea's Drama World*, *Forbes*, Available at: http://www.citethisforme.com/harvard-referencing.

KANG, EUNYOUNG 2010, *A Study on the Cases of Applying Interactive Storytelling on the Web Site*, Department of Media Broadcasting Graduate School of Information Sciences 2010.2, Available at: http://www.riss.kr/search/detail/DetailView.do?p_mat_type=be54d9b8bc7cdb09&control_no=aadbdcbc32da4d9cffe0bdc3ef48d419#redirect

Kim, Ji-Yeon, Yun, Jae Young 2015, *A Study on Interactive Stroytelling Methods for Online News*, International Design Conference of KSDS and ADADA with Cumulus, Available at http://www.dbpia.co.kr/journal/articleDetail?nodeId=NODE06634808.

Kyungmo, Kim 2012, *Online News Production in the New Journalism Environment Between Tradition and Change*, Journal of communication research 49(1), Available at http://www.dbpia.co.kr/journal/articleDetail?nodeId=NODE01793877.

Wi-Geun, Kim 2014, *The Influence of Portal Site News Services on Online Journalism in Korea: The Structural Transformation or the Power Change in the News Distribution*, Korean Journal of Communication & Information (2014.5), Available at http://www.dbpia.co.kr/journal/articleDetail?nodeId=NODE02452231.

Yeo-Kwang, Yoon 2014, *A Study on Contents Curation of Portal Sites*, Journal of the Korea Entertainment Industry Association 8(4), Available at http://www.dbpia.co.kr/journal/articleDetail?nodeId=NODE06069176.