AI6122 Text Data Management & Analysis

Topic: Event detection

Event detection

- Topic detection and tracking
- Event detection
  - Document-pivot techniques
  - Feature-pivot techniques
- Case study
  - Event detection on Twitter
  - Event detection by queries and documents
  - Event popularity prediction

A bit of history about event detection

- Topic Detection and Tracking (TDT) is a DARPA-sponsored initiative
  - to investigate the state of the art in finding and following new events in a stream of broadcast news stories.
  - TDT Pilot study ran from Sep 1996 to Oct 1997 by DARPA, CMU, Dragon Systems, UMass.

- Three tasks
  - **Segmenting** a stream of data, especially recognized speech, into distinct stories
  - Identifying those news stories that are the first to discuss a new event occurring in the news → **New Event Detection**
  - Given a small number of sample news stories about an event, finding all following stories in the stream → **Event Tracking**

TDT and Event

- TDT: detecting the appearance of new topics and for tracking the reappearance and evolution of them
  - Notion of a “topic” is modified to be an “event” during the study

- **Event**: some unique thing that happens at some point in time.
  - Emphasis more on the “topic” of the event and time, rather than spatial/temporal localization.
    - Example: the eruption of Mount Pinatubo on June 15th, 1991 is considered to be an event, whereas volcanic eruption in general is considered to be a class of events.
  - Events might be unexpected, such as the eruption of a volcano, or expected, such as a political election, or periodical like new year celebration

Where to detect events?

- TDT study assumes **multiple sources of information**, for example various newswires and various news broadcast programs

- The information flowing from each source is assumed to be divided into a **sequence of stories**, which may provide information on one or more events.
  - The general task is to identify the events being discussed in these stories, in terms of the stories that discuss them.
  - Stories that discuss *unexpected events* will of course follow the event,
  - Stories on *expected events* can both precede and follow the event.

TDT Tasks

- *The Segmentation Task*: the task of segmenting a continuous stream of text (including transcribed speech) into its constituent stories.

- *The Detection Task*: **Retrospective Event Detection**
  - The task of identifying **all of the events in a corpus of stories.**
    - Discovering previously unidentified events in an accumulated collection
  - Events are defined by their association with stories.
  - The task is to group the stories in the corpus into clusters. Each cluster represents an event, and the stories in the cluster discuss the event.
  - It will be assumed that each story discusses at most one event. Therefore each story may be included in at most one cluster

TDT Tasks

- *The Detection Task*: **Online New Event Detection**
  - The task of identifying new events in a stream of stories.
  - Each story is processed in sequence, and a decision is made whether or not a **new event** is discussed in the story after processing the story.
  - The decision is made before processing any subsequent stories (cannot access subsequent stories in online setting)
  - The first story to discuss an event should be flagged YES. If the story doesn't discuss any new events, then it should be flagged NO.

- ***The Tracking Task***: associating incoming stories with events known to the system.
  - An event is defined ("known") by its association with stories that discuss the event. Thus each target event is defined by a list of stories that discuss it.

Event detection

- Event detection nowadays typically refers to the detection of new event and its subsequent stories (i.e., tracking)
  - **Retrospective** Event Detection vs **Online** New Event Detection

- **Document-Pivot Technique**s: event detection is to cluster documents into clusters (events)
  - A document is a data point; event is a cluster
  - Retrospective event detection can use clustering algorithms to access the entire document collection, and to organize the documents into topic clusters, e.g., hierarchical agglomerative clustering (HAC)
  - New event detection: incremental clustering algorithms to process the input streams sequentially,
    - Merge an event with the most similar one,
    - Create a new cluster if the similarity measure exceeds a predefined threshold

Document-Pivot Technique: Incremental clustering

- Take a document d from the document stream (information source)
  - Computer similarity between d and the known events $e \in E$ (i.e., document clusters)
    - If $sim(d, e) \geq \theta$, assign d to e with the highest similarity
    - If $sim(d, e) < \theta$, consider d as a new event (a new cluster with a single document for now)
  - Till all documents processed in the stream.

- Parameters to consider:
  - Similarity function, e.g., cosine similarity
  - Document and event representation
    - Tfidf vector? Recent documents in an event be given more weight?
  - Threshold θ
  - Filtering of events: only consider recent events when computing $sim(d, e)$?

Feature-Pivot Techniques

- Identify topic areas that were previously unseen or rapidly growing in importance within the corpus, **bursty topics**

- Feature-pivot techniques model an event in text streams as a bursty activity, with certain features rising sharply in frequency as the event emerges.
  - An event is therefore conventionally represented by a number of keywords showing burst in appearance counts
  - The underlying assumption is that some related words would show an increased usage as an event occurs.

- These techniques analyze feature distributions and discover events by grouping bursty features with identical trends.

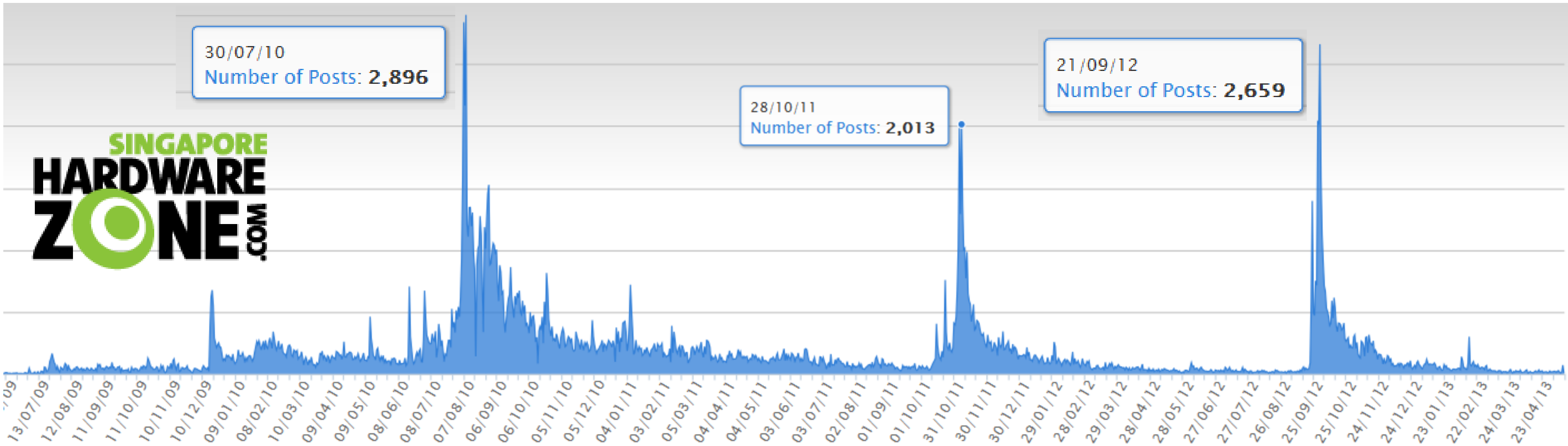
Feature-Pivot Techniques

- “Bursty and hierarchical structure in streams” by Kleinberg (2002)
  – A formal approach for modeling such “bursts”: An infinite-state automaton; Bursts appear as state transitions
  – A nested representation of the set of bursts that imposes a hierarchical structure on the overall stream.
- “Parameter free bursty events detection in text streams” by Fung et al. (2005)
  – Modeled word appearance as binomial distribution, identified the bursty words according to a heuristic-based threshold, and grouped bursty features to find bursty events.
- “Analyzing feature trajectories for event detection” by He, Chang, Lim, and Zhang (2007)
  – Use discrete Fourier transformation (DFT) to categorize features for different event characteristics (e.g., important or not, and periodic or aperiodic events).
  – DFT converts the signals from the time domain into the frequency domain, such that a burst in the time domain corresponds to a spike in the frequency domain

Feature-Pivot Event Detection

- Detect bursty features based on certain models or statistics.
- Events are then detected by maximizing the co-occurrences among documents and the consistence of the frequency distributions for all bursty features within an event.
- The timestamp for an event is calculated based on the bursty periods of the bursty features related to that event

What if a “major event” happens in social media?



Sub-forum: mobile communication technology (2009 – 2013)

Singapore launch date for **iPhone 4** is July 30 **...** - iMerlion
www.imerlion.com/2010/07/**singapore-launch-date**-for-**iphone-4**-is.html

iPhone 4S releases in **Singapore** on 28 Oct 2011 **...**
sgtransport.blogspot.com/.../**iphone-4s-releases-in-singapore**-on-28.html

SingTel to offer **iPhone 5** in **Singapore** on September 21 **...**
info.singtel.com › About Us › NewsRoom

Event detection applies to Twitter (social media)

- Feature-Pivot Event Detection
  - Identify the bursty words based on certain statistics or model.
  - Grouped burty words into events based on their co-occurrences.

- Document-Pivot Event Detection
  - Cluster tweets into events. The tweets similar to each other are grouped as events.
  - Certain terms like "named entities" could be assigned with high weights. Named entities need to be recognized by a Named Entity Recognizer

  - Efficient clustering can be achieved through locality sensitive hashing (LSH). Example LSH is MinHash algorithm.
    - The simplest version of the minhash scheme uses k different hash functions, where k is a fixed integer parameter, and represents each set S by the k values of $h_{min}(S)$ for these k functions.

Case study: Segment-based Event Detection from Tweets

- Twitter
  - A message written by the users, up to **140 characters** with **free writing styles**
    - information updates/sharing at low cost
  - A real-time information network that connects you to **the latest information** in your world.

- Event detection in Twitter
  - Events attracted user attentions
  - Events can be more timely detected

- Event detection in Twitter is challenging

Event detection in Twitter: Challenges

- Large data volume
  - 500 million tweets per day in 2019
- Diverse and fast changing topics
- Short and noisy content

PAP POSTERS ARE EVERYWHERE! AND FOR SOME LAMP POLES THERE ARE BOTH **NSP** AND **PAP** POSTERS! #whathappentosavingtheearth

ya la! some of them gg to **potong pasir**. I'm gg to **yio chu kang**

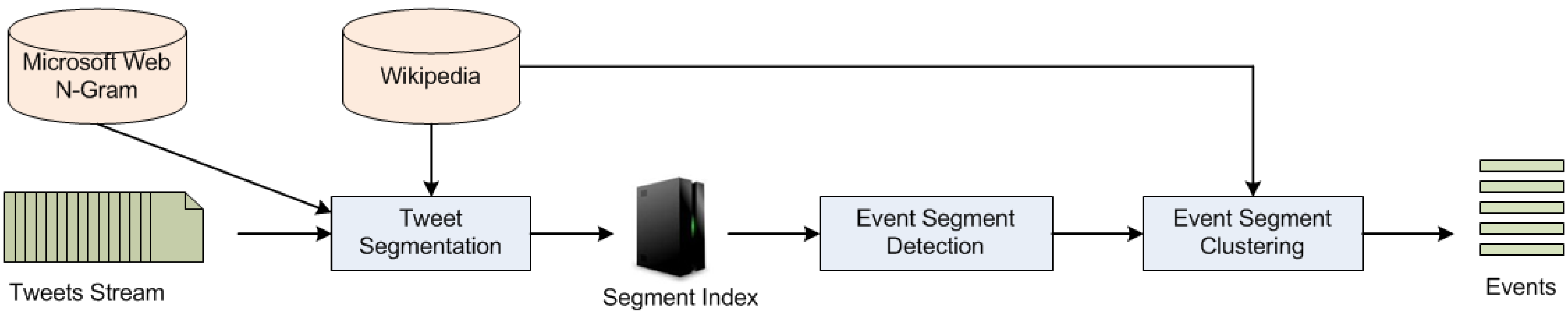
Principle of Least Effort [Zipf49]: People used to communicate information with the least context, especially in the situation where a short message with free style is allowed.

- Miss spellings
- Informal abbreviations

Event detection in Twitter: Our approach

- Tweet Segmentation → **Informative keyphrases**
  - Reduce noise for further processing.
- **User** Frequency
  - Robust to the negative impact of Spam & Self-Promotion tweets.
- External Knowledge Base (**Wikipedia**)
  - Resist to the adverse impact of Pointless Babble tweets.
  - Derive interpretable event descriptions.

TwEvent: System architecture



Tweets → Tweet segments → Event segments → Events

Iphone 4g's coming out on 4th july as according t @zoewasabi hmm. My birthday's on the 10th july. I can use iphone as a present!!! :D

iphone 4g |s |**coming out** |4th july |according |t |hmm |birthday |10th july |use |**iphone** |present |d

iPhone 4G is been officially announced today at WDC

iphone 4g |officially announced |today |wdc

Iphone 4g, iphone, coming out

Tweet segmentation

- Each segment (unigram/multi-gram) may represent a semantic unit
  - Example segments: **Steve Jobs, MTV Movie Awards**
  - Implemented by maximizing the sum of **stickiness of all segments**

- External resources for calculating the stickiness of a segment.
  - Microsoft Web N-Gram: A prior **probability for each segment** in the index of English web pages
  - Wikipedia: the likelihood that a segment **being an anchor text** in Wikipedia pages.

Tweet segmentation: Example

Example Tweet Portion

youth olympic games sailing competition

Possible segmentation 1

(youth) | (olympic games) | (sailing competition)

Possible segmentation 2

(youth olympic games) | (sailing competition)

Possible segmentation 3

(youth) | (olympic games sailing competition)

Segment burstyness

- **Bursty Segment**: A segment *s* is a bursty segment in time window *t* if its tweet frequency $f_{s,t} > E[s|t]$

- **Bursty Probability**: $P_b(s,t) \in (0,1]$ indicates the degree of busrtyness of a segment *s* with frequency $f_{s,t}$.

  – $P_b(s,t) = 1$

  $$f_{s,t} \geq E[s|t] + 2\sigma[s|t]$$

  – $P_b(s,t) = sigmoid\left(10 \times \frac{f_{s,t} - (E[S|t] + \sigma[S|t])}{\sigma[S|t]}\right)$

  $$f_{s,t} \in (E[s|t], E[s|t] + 2\sigma[s|t])$$

Misspelling words and informal abbrev. are detected as bursty segments

Event segment detection

- **User frequency**
  - The number of users who post tweets containing segment s during the time window t.

- Weight each bursty segment: $w_b(\mathrm{s},\mathrm{t}) = P_b(s,t)\log(u_{s,t})$

- **Event segment**: A bursty segment s is a potential event segment in time window t if it is ranked among top-*K* bursty segments by $w_b(s,\mathrm{t})$ where $K = \sqrt{N_t}$
  - N_t is number of tweets in time window t

Event segment similarity

- For each time window t, we further divide the period evenly into M sub-time-window with a weight: $w_t(s,m) = \frac{f_t(s,m)}{\sum_{m'=1}^{M} f_t(s,m')}$

- A pseudo document $T_t(s,m)$ is built for each segment s at sub-time window m by concatenating all tweets containing s in that window

- The similarity between a pair of segments s_a and s_b is the weighted cosine similarity with tf.idf scheme.

$$sim_t(s_a, s_b) = \sum_{m=1}^{M} w_t(s_a, m) w_t(s_b, m) sim\left(T_t(s_a, m), T_t(s_b, m)\right)$$

The semantic of a segment is defined by the tweets containing it.

Event segment clustering: k-Nearest Neighbor graph

- Two event segments are in the same cluster if they appear in each others' k-nearest neighbors.
  - An edge between two event segments is retained if and only if they appear in each others' k-nearest neighbors.
  - The resulted connected components are considered as **Candidate Events**.

- **Event**: "anything that happens, especially something important and unusual" --- Cambridge Dictionaries Online
  - Example candidate event: [***Friday night, Friday, weekends, trip, enjoy***] → plans or schedule for weekends
  - **Important and unusual event?**

Event newsworthiness

- **Segment newsworthiness**: the probability that a sub-phrase in the segment appear as anchor text in Wikipedia articles that containing the segment:

$$\mu(s) = max_{l \in s} e^{Q(l)} - 1$$

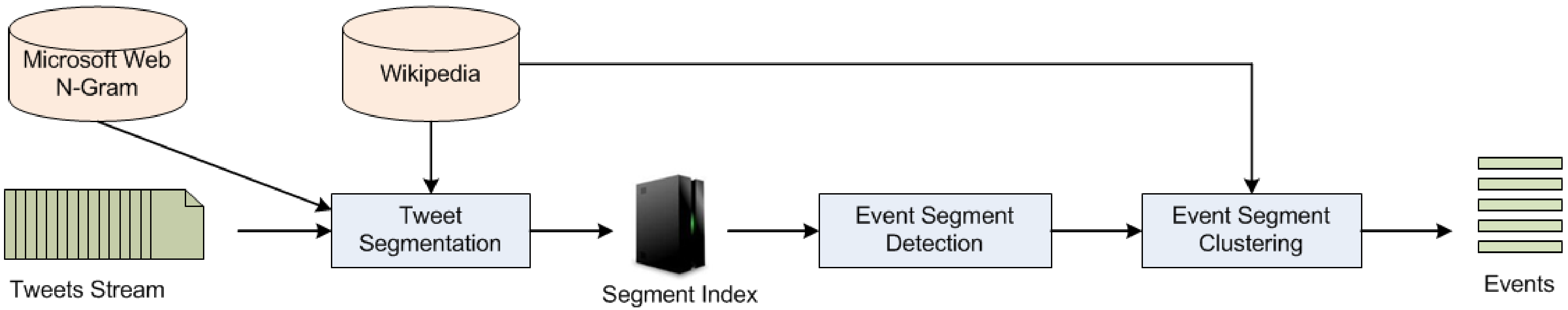
  - $Q(l)$ is the prior probability that l appears as anchor text in Wikipedia articles that contain l, and l is any sub-phrase of s

- **Event newsworthiness** :
  - Often used as anchor text in Wikipedia (well known entities)
  - Well connected

$$\mu(e) = \frac{\sum_{s \in e_s} \mu(s)}{|e_s|} \cdot \frac{\sum_{g \in E_e} sim(g)}{|e_s|}$$

Threshold-based event selection

- Events are selected based on event newsworthiness score from all candidate events in the time window



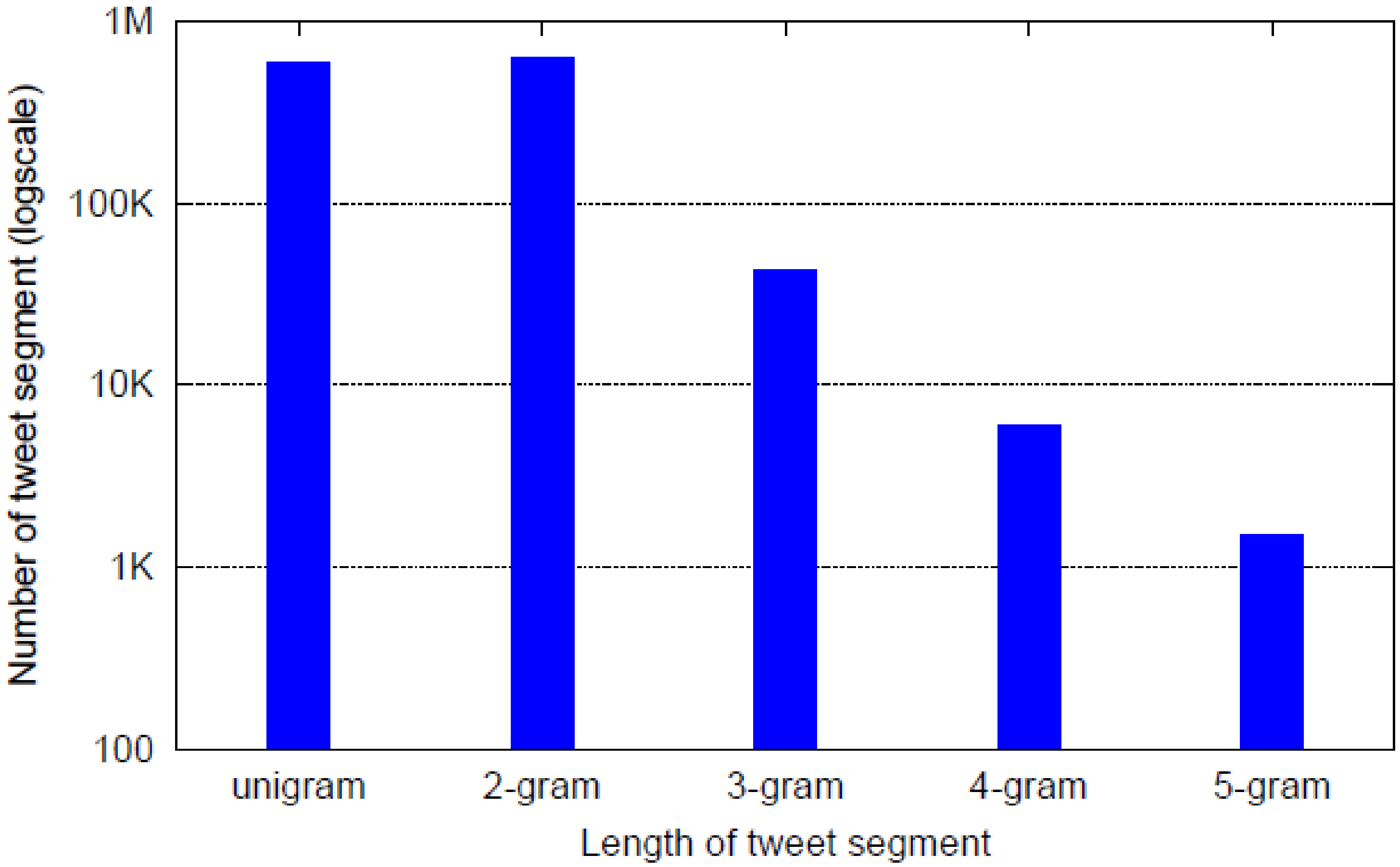
Experiments: Dataset

- Wikipedia:
  - English Wikipedia Dump 2010

- Tweets:
  - 4,331,937 tweets posted in June 2010 by Singapore-based Users.

- Realistic events in data collection period:
  - FIFA World Cup 2010;
  - WWDC 2010;
  - MTV Movie Awards 2010.

Experiments: statistics on segments



Experiments: example events detected

Day	e_{ID}	[Event Segments]: Event Description
7	e_1.	**[steve jobs, imovie, wwdc, iphone, wifi]**: iPhone4 was released during WWDC 2010.
	e_2.	**[mtv movie awards, mtv, new moon, twilight, robe]**: The movie *The Twilight Saga: New Moon* was the biggest winner in MTV Movie Awards 2010; it took 4 out of 10 "Best" Awards.
	e_3.	**[yesung, yesung oppa, kyuhyun, oppa, kyu]**: Korean popular band Super Junior's showcase was held on June 6, 2010 at Singapore. Yesung Oppa and Kyuhyun Oppa are members of Super Junior.
8	e_4.	**[lady gaga, music video, gaga, mv, alejandro]**: The music video *Alejandro* by Lady GaGa was premiered officially on June 8, 2010.
	e_5.	**[ss501, indonesia, ariel, sama, trend]**: No clear corresponding real-life event.
	e_6.	**[singapore, iphone 4g, iphone 3gs, iphone, coming out]**: Related to event e_1. People started to talk about the release date of iPhone 4 in Singapore.
9	e_7.	**[lady gaga, youtube, youtube video, music video, gaga]**: Related to event e_4.
	e_8.	**[twitter, whale, stupid, capacity, over again]**: A number of users complained they could not use twitter due to over-capacity. A logo with whale is usually used to denote over-capacity.
	e_9.	**[ipad, iphone, apple, new]**: Related to event e_1.
	e_{10}.	**[watching glee, glee, season finale, season, channel]**: The season finale of the American TV series *Glee* was broadcasted on June 8, 2010.
10	e_{11}.	**[lady gaga, youtube, youtube video, music video, amber]**: Related to event e_7.
	e_{12}.	**[justin bieber, try, pa, took, each]**: Related to event e_{15}. The song *Never Say Never* by Justin Bieber serves as the theme song for the movie *The Karate Kid*, which was released on June 10, 2010 in Singapore.
	e_{13}.	**[yesung, tweeted]**: Super Junior's Yesung posted a photo about his pet turtles.
	e_{14}.	**[twitter, whale, stupid, capacity, over]**: Related to event e_8.
	e_{15}	**[karate kid, watch movie, movie]**: The movie *The Karate Kid* was released on June 10, 2010 in Singapore.
11	e_{16}.	**[uruguay vs france, uruguay, france, vs]**: A match between Uruguay and France in World Cup 2010.
	e_{17}.	**[south africa, vs mexico, mexico, goal, first goal]**: A match between South Africa and Mexico in World Cup 2010. And the first goal of the 2010 World Cup was scored in the match.

All events: http://www.cais.ntu.edu.sg/~lichenliang/twevent/eventlist.txt

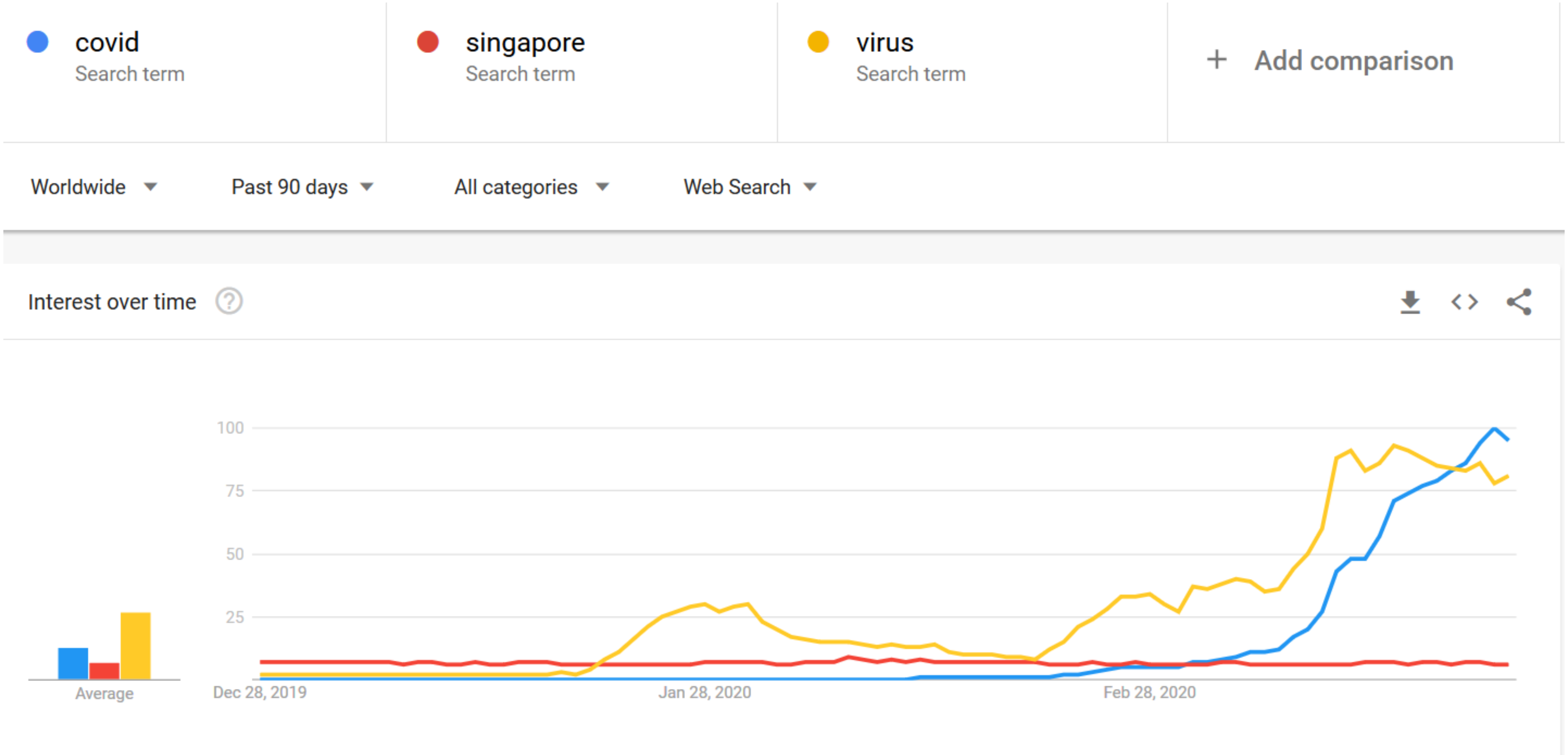
Summary on event detection from Twitter

- Tweet Segmentation → **Informative keyphrases**
  - Reduce noise for further processing.

- **User** Frequency
  - Robust to the negative impact of Spam & Self-Promotion tweets.

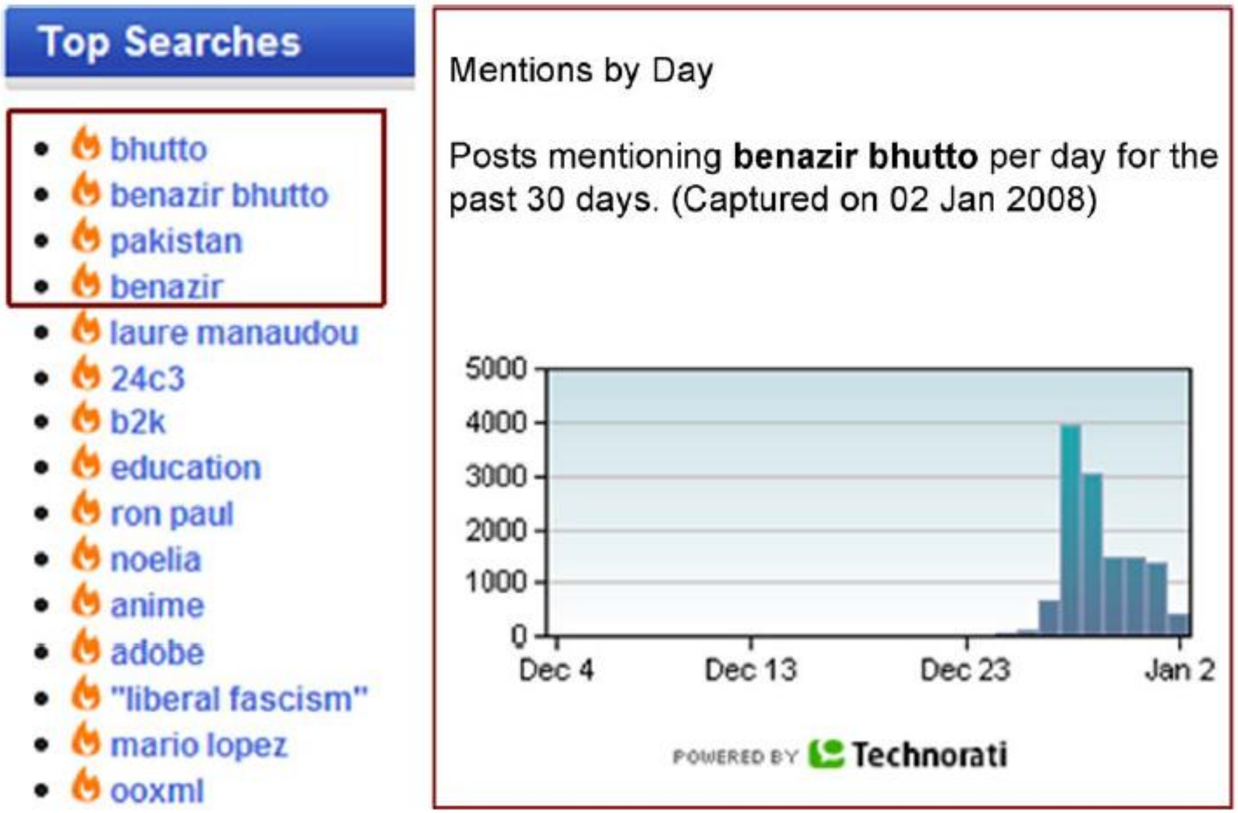
- External Knowledge Base (**Wikipedia**)
  - Resist to the adverse impact of Pointless Babble tweets.
  - Derive interpretable event descriptions.

Case study: event detection of common interest



Example

- Top-15 popular searches from Technorati.com captured on December 28, 2007 and statistics on blog posts captured on January 2, 2008



Case study: event detection of common interest

- Events of common interest to many users
  - A large volume of event-related queries are issued to news/blog search engines, making them popular queries during the event period.
  - A large number of news articles and blog posts are published by journalists and bloggers containing updated facts, commentary or discussions about the event.

- From the updated information, web users may formulate new queries (e.g., another person involved in the event) which may subsequently become popular queries.
  - The changes in the queries at different time points become good indications of event evolution.

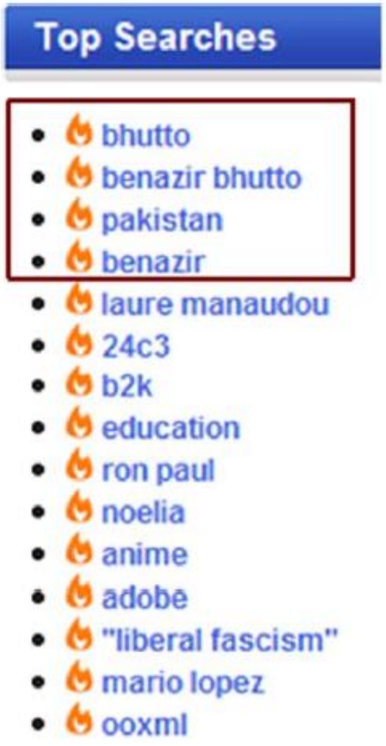
User in the loop

- Event detection to consider interactions between
  - *query streams*: what people search for
  - *news streams:* what are reported
  - *blog streams:* what are written by users
- Basically: what web users *want to know about* and what they *talk about*.

- Event detection guided by user queries

Challenges

- Not all popular queries issued by masses of web users are event-related.
  - Event-related queries increase dramatically when an event happens does not necessarily imply that all popular queries are event-related.
  - Many extremely popular queries are likely to be website names, such as Google, MySpace, and YouTube, and they are often not event-related.
- Multiple query keywords may be related to the same event.
  - The same query keyword Pakistan issued at different time points may refer to different events happened in that country.
- Computational cost
  - Consider the large number of news articles and blog posts accessible online



Query profile

- **Profile** of a query q at time point t is the set of most recently published documents from news (or blog) stream that match q.
  – Consider to use the most recent 50 matching documents to define the meaning of a query term at the current time (in news search or web search)

- **Recency**: query profile *recency* is the averaged time difference between documents in the query profile to the query's issuing time

- ***Clarity**:* how indicative the words are in the query profile
  – If a query is event-related, its word features describing the event can often distinguish it from the background.

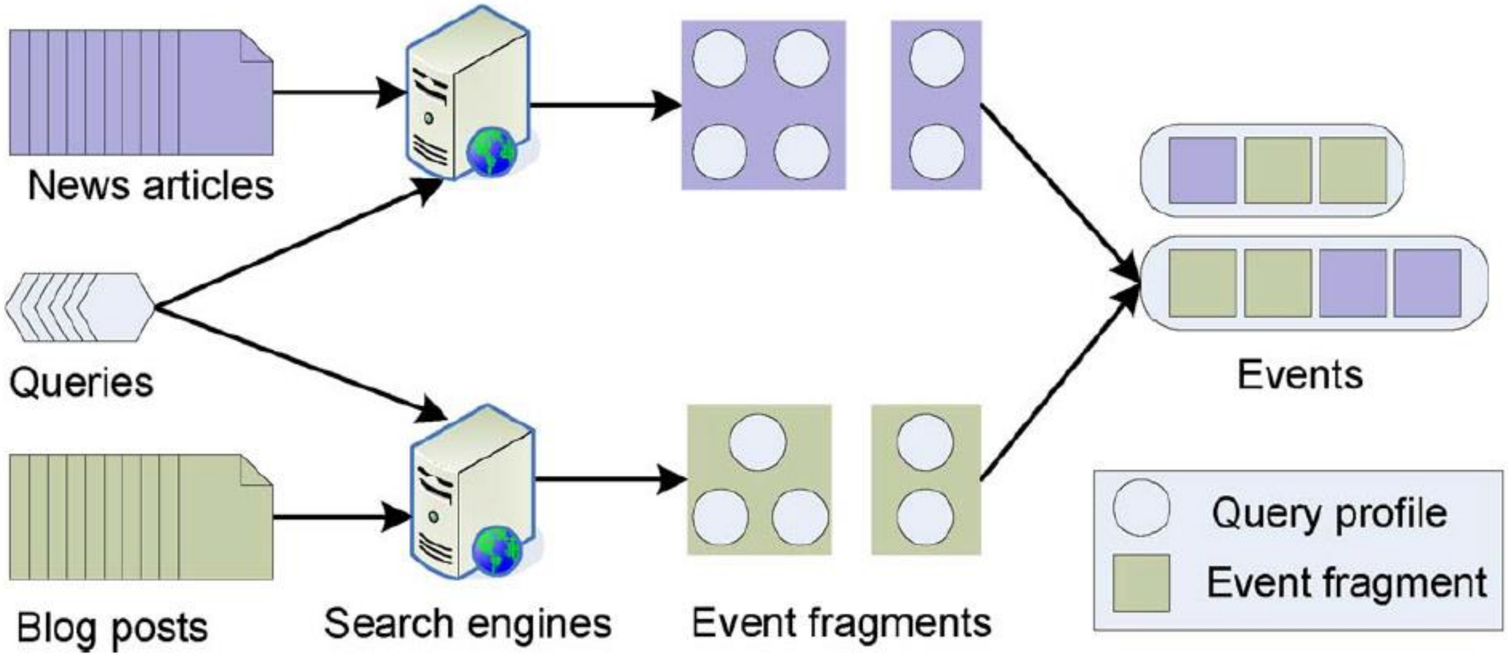
Query Profile Clarify

- The word distribution of the query profile, compared to that of a general document collection, measured through Kullback–Leibler (KL) divergence.
  - Divergence between the language model of a query profile, and the language model of a general document collection (i.e., background collection)

$$KL(C_q, S) = \sum_{w \in C_q} P(w|C_q) \log_2 \frac{P(w|C_q)}{P(w|S)}$$

- Assuming query q is about a particular event, then the set of words that are frequently observed among documents in C_q are likely to be describing the event.

Event detection process



Query profile → Event fragment → Event

- For each query in query stream, two query profiles are constructed from the news and blog streams, respectively
  - The event-related query profiles are further passed to event fragment detection.
  - The non-event related query profiles are dropped.

- An event fragment is a set of query profiles that is about the same event received from the same document stream within a predefined time window T.
  - Documents from different streams may demonstrate different properties (e.g., blog posts are noisier than news articles in general),
  - Event fragment detection may require different parameter settings for different document streams.

- Event fragments from both document streams are grouped into events.
  - An event is a sequence of event fragments from both news and blog streams. An event fragment contains query profiles that each contains documents.
  - A detected event contains queries and news articles/blog posts matching them.

Event Fragment Detection

- Given the query profiles in time window T, event fragment detection is to group the query profiles related to the same event into one event fragment (i.e., a small cluster).

- To perform the grouping
  - An appropriate distance metric between any pair of query profiles; and
  - an appropriate clustering algorithm

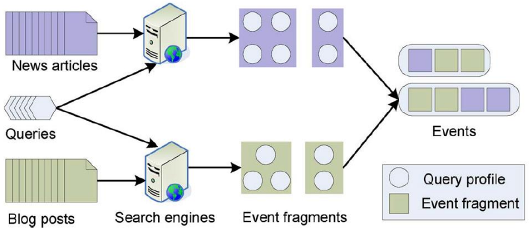
- Distance between two query profiles
  - Cosine similarity
  - Divergence between the language models of the two (sets of) documents (e.g., square root of Jensen-Shannon divergence or *JSD)*

Event fragment detection as clustering

- A clustering algorithm satisfying the following requirements:
  - The algorithm requires no prior knowledge on the number of event fragments to be detected from a set of query profiles,
    - It is unreasonable to guess how many events would happen in a given time window;
  - The algorithm should be able to filter away noise, as predicting whether a query is event-related can never be perfect;
  - The algorithm should be able to handle large data set with reasonable space and time complexity.

- DBSCAN is the choice here

Grouping event fragments into events



- Semantic distance
  - Event fragments in the same event show talk about similar things.

- Query distance
  - Semantic distance worked well for event fragments received from the same stream but not across streams
  - Differences in vocabulary and writing style between news and blogs.
  - Event fragments received within a short time period are likely related to the same event if they share common query keywords.

- Temporal distance
  - An event may last for a long time period and evolve at a fast pace, event fragments of the same event but are temporally far apart may not be similar to each other.
  - Only compute the distance between a newly detected event fragment to those recently detected within 5 days.
  - The timestamp of the event fragment is derived from its query profiles

Case study: Hashtag popularity prediction

- Predict the popularity of new hashtags in the near future (e.g., one day).
  - Popularity range: <25, [25, 50), [50, 100), [100, 200), >200
- Classification and feature engineering approach
  - Classifiers: NB, kNN, SVM, Logic Regression
  - Features: 7 content features and 11 context features
- Main findings
  - Context features are more effective than content features
  - More effective on bursty tags than continuous tags

7 Content Features and 11 Context Features

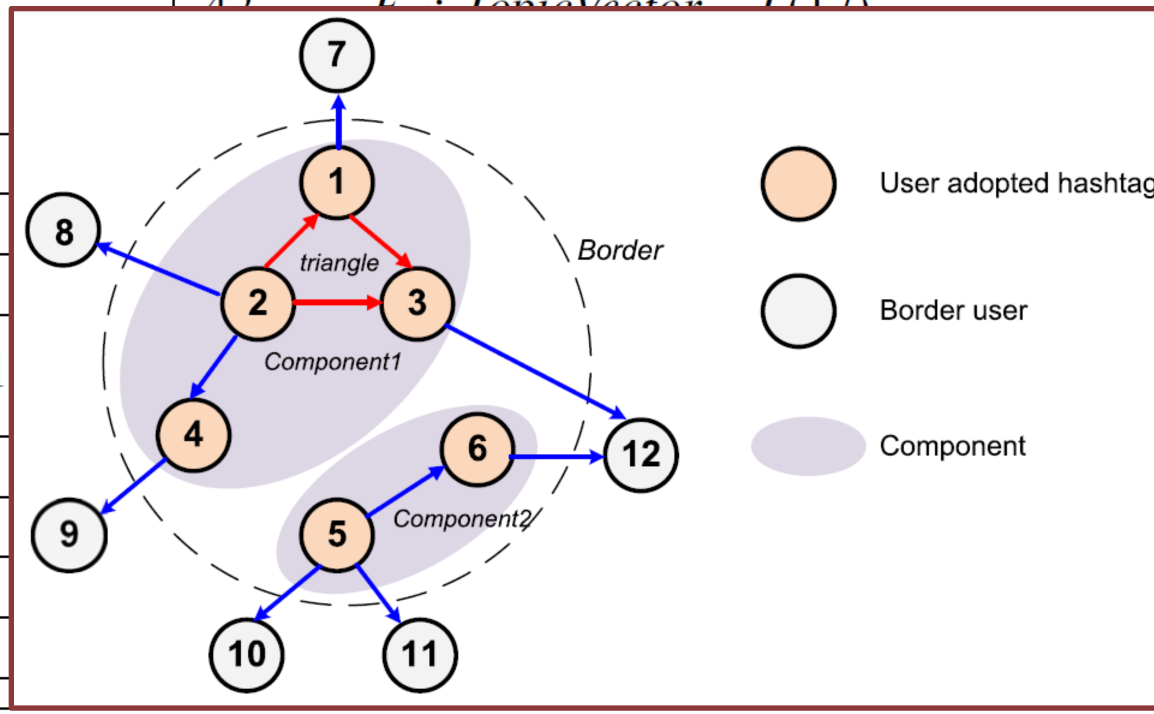
Feature		Description
F_{c1}	*ContainingDigits*	Binary attribute checking whether or not a hashtag contains digits
F_{c2}	*SegWordNum*	Number of segment words from a hashtag
F_{c3}	*URLFrac*	Fraction of tweets containing URL in T_t^h
F_{c4}	*SentimentVector*	3-dimension vector: ratio of neutral, positive and negative tweets in T_t^h
F_{c5}	*TopicVector*	20-dimension topic distribution vector derived from T_t^h using Topic Model
F_{c6}	*HashtagClarity*	KL-divergence of word distribution between T_t^h and tweets collection $\mathcal{T}$
F_{c7}	*SegWordClarity*	KL-divergence of word distribution between tweets containing any segment word in h and tweet collection $\mathcal{T}$
F_{x1}	*UserCount*	Number of users $\|U_t^h\|$
F_{x2}	*TweetsNum*	Number of tweets $\|T_t^h\|$
F_{x3}	*ReplyFrac*	Fraction of tweets containing mention @
F_{x4}	*RetweetFrac*	Fraction of tweets containing *RT*
F_{x5}	*AveAuthority*	Average authority of users in G_t^h
F_{x6}	*TriangleFrac*	Fraction of users forming triangles in G_t^h
F_{x7}	*GraphDensity*	Density of G_t^h
F_{x8}	*ComponentRatio*	Ratio between number of connected components and number of nodes in G_t^h
F_{x9}	*AveEdgeStrength*	Average edge weights in G_t^h
F_{x10}	*BorderUserCount*	Number of border users
F_{x11}	*ExposureVector*	15-dimension vector of exposure probability $P(k)$

The most and least effective 15 features

Rank	Feature	Rank	Feature
1	F_{x1}: $UserCount$	39	F_{c5}: $TopicVector - T(2)$
2	F_{x10}: $BorderUserCount$	40	F_{c5}: $TopicVector - T(14)$
3	F_{x2}: $TweetsNum$	41	F_{x9}: $AveEdgeStrength$
4	F_{c6}: $HashtagClarity$	42	F_{c5}: $TopicVector - T(17)$
5	F_{x6}: $TriangleFrac$	43	F_{x8}: $ComponentRatio$
6	F_{x11}: $ExposureVector - P(15)$	44	F_{c5}: $TopicVector - T(20)$
7	F_{x11}: $ExposureVector - P(14)$	45	F_{c5}: $TopicVector - T(9)$
8	F_{x11}: $ExposureVector - P(9)$	46	F_{c5}: $TopicVector - T(1)$
9	F_{x11}: $ExposureVector - P(10)$	47	F_{c4} :$PosRatio$
10	F_{c5}: $TopicVector - T(13)$	48	F_{x5} :$AveAuthority$
11	F_{x11}: $ExposureVector - P(11)$	49	F_{c4}: $NegRatio$
12	F_{x11}: $ExposureVector - P(5)$	50	F_{c7}: $SegWordClarity$
13	F_{x11}: $ExposureVector - P(8)$	51	F_{c4}: $NeuRatio$
14	F_{x11}: $ExposureVector - P(7)$	52	F_{c2}: $SegWordNum$
15	F_{x11}: $ExposureVector - P(12)$	53	F_{c1}: $ContainingDigits$

The most and least effective 15 features

Rank	Feature	Rank	Feature
1	F_{x1}: *UserCount*	39	F_{c5}: *TopicVector* $- T(2)$
2	F_{x10}: *BorderUserCount*	40	F_{c5}: *TopicVector* $- T(14)$
3	F_{x2}: *TweetsNum*	41	F_{x9}: *AveEdgeStrength*
4	F_{c6}: *HashtagClarity*	42	[illegible]
5	F_{x6}: *TriangleFrac*		
6	F_{x11}: *ExposureVector* $-$		
7	F_{x11}: *ExposureVector* $-$		
8	F_{x11}: *ExposureVector* $-$		
9	F_{x11}: *ExposureVector* $-$		
10	F_{c5}: *TopicVector* $- T(1$		
11	F_{x11}: *ExposureVector* $-$		
12	F_{x11}: *ExposureVector* $-$		
13	F_{x11}: *ExposureVector* $-$		
14	F_{x11}: *ExposureVector* $-$		
15	F_{x11}: *ExposureVector* $-$		



Event detection

- Topic detection and tracking

- Event detection
  - Document-pivot techniques
  - Feature-pivot techniques

- Case study
  - Event detection on Twitter
  - Event detection by queries and documents
  - Event popularity prediction

Reference

- Topic Detection and Tracking Pilot Study Final Report
  http://ciir.cs.umass.edu/pubfiles/ir-137.pdf

- A Survey of Techniques for Event Detection in Twitter.
  https://doi.org/10.1111/coin.12017

- Case studies:
  - *Twevent: segment-based event detection from tweets*
  - *Query-Guided Event Detection From News and Blog Streams*
  - *On predicting the popularity of newly emerging hashtags in Twitter*