Multilevel Event Detection, Storyline Generation, and Summarization for Tweet Streams

Poonam Goyal, Prerna Kaushik, Pranjal Gupta, Dev Vashisth, Shavak Agarwal, and Navneet Goyal

***Abstract*— Users acting as real-time sensors post information about current events on various social media sites like Twitter, Facebook, Instagram, and so on. This generates a huge amount of data requiring significant effort to process and filter it to detect events/topics. It becomes more challenging when data are generated as a tweet stream because of its speed, presence of noise, slangs, phrases, abbreviations, and so on. In recent years, many approaches have been proposed either for detecting small- or large-scale events, individually. There is a lack of a complete solution that provides analysis from different perspectives. We propose a novel approach Mythos that detects events, subevents within an event, and generates abstract summary and storyline to provide different perspectives for an event. There are three modules in Mythos. Online incremental clustering algorithm identifies small-scale events in the form of small clusters, the event hierarchy generator generates bigger events in the form of hierarchies, and the summarization module produces summary of events/subevents. The summarization module uses a long short-term memory (LSTM)-based learning model to generate summaries at different levels—from the most abstracted to the most detailed. The summaries at different levels are used to generate a storyline for the event. Our experimental analysis on a variety of twitter data sets from different domains compares Mythos against the known existing approaches for event detection and summarization. It outperforms baseline approaches for both. The generated summaries are evaluated against summaries provided by external reference sources like Guardian and Wikipedia.**

***Index Terms*— Clustering, event detection, event hierarchy, storyline, subevent detection, summarization, Twitter.**

I. Introduction

MICROBLOGGING site like Twitter, in the past few years, has attracted the attention of news agencies, political parties, commercial websites, and so on, due to its usefulness to gather and spread information [1], to get feedback from users [2], to promote their products [3], and so on. The growing number of users' voicing their views in real time generates an enormous amount of data providing treasure of information, which can be used in different domains, namely, disaster management, news [4], [5], sport [6], traffic [7], and so on. Especially Twitter stands out as a public platform due to its unique way of allowing people to witness and express about every bit of the significant events. Many times, it is ahead of newswire and reaches people as soon as events unfold. A major benefit of the event detection and summarization of tweets is the added social component that allows readers to understand the impact of the event and the people's reactions to it. Event detection in Twitter data stream thus becomes an important research problem. Despite numerous challenges posed by Twitter data, researchers have used several unsupervised and supervised approaches [1], [8]–[10] to detect events online. These approaches are limited to detect global/large-scale events and take a total number of events as input parameter. However, the detection of events from twitter stream should be an automated process rather than governed by a user-defined parameter.

An event is complex as it may have many subevents, subsubevents (SSevts), and so on, e.g., the game of Super Bowl (SB) has four quarters and each quarter can be treated as a subevent and each subevent can have further subevents such as multiple goals or touchdowns, etc., and so on. Moreover, there can be distinct subevents or SSevts, which are not part of the game such as people protesting against police brutality before the beginning of the game and then players kneeling to protest in SB 2018. Therefore, it is necessary to detect all subevents/SSevts for an event. A subevent/SSevt is an event and called so when it is part of a bigger event. The subevents of subsubevent are also referred to as SSevts.

Researchers have also worked to identify small-scale/local events [11]–[15] and subevents [16]–[20] for tweet stream. In these approaches, small-scale and fast-changing events are overlooked because only few tweets are generated for them resulting in low-term frequencies. Jasmine system was developed to detect local events using geolocation information from micro blogs [12]. It obtains a name list of locations from geotagged tweets and adds positional information to tweets by matching the location name. EventRadar [13] detects a local event in real time using a density-based algorithm. It first identifies potential topics from a time interval and later uses classification to detect whether event is local or not by considering seven days' historic data. These approaches are limited to certain specific types of event content that they can handle. Subevent detection methods identify bursts or key moments in real time. Buntain *et al.* [18] use a sliding window approach to process Twitter's unfiltered public sample stream for bursty and nonbursty tokens and uses a classifier to identify tokens experiencing these bursts. TopicSketch [20] uses a sketch-based structure to detect bursty topics in real time. These approaches either detect events of predefined topics such as weather, sports, and so on, or detect subevents and small-scale events without specifying larger context.

Manuscript received May 29, 2019; revised October 22, 2019; accepted October 31, 2019. Date of publication February 16, 2019; date of current version February 24, 2020. *(Corresponding author: Poonam Goyal.)*

The authors are with the Advanced Data Analytics and Parallel Technologies (ADAPT) Laboratory, Department of Computer Science, BITS Pilani, Pilani 333031, India (e-mail: poonam@pilani.bits-pilani.ac.in).

Digital Object Identifier 10.1109/TCSS.2019.2954116

These approaches do not deal with detecting inherent hierarchy hidden in these subevents to portrait the events at abstract level.

In order to view events, subevents and SSevts having coarser and fine details, we propose a framework, Mythos. The objective of the proposed framework is to provide a complete picture of events/subevents, including fast-changing subevents in the form of hierarchies. It uses the online clustering algorithm to capture fast-changing subevents at the earliest. These small-scale events are then merged considering temporal and semantic proximity to obtain event hierarchies.

Storyline and summarization are two ways to express an event. Existing methods of summarization give one/multiple tweet summary [21], fixed-length summary for online, and historical events [22]. We generate the summary for different levels in the hierarchy of an event, whose length depends upon the content of the events. Experimental results have been evaluated and compared with the existing methods for event detection and summarization for several benchmarked data sets. Our experimental results of event detection of whole/partial event hierarchy on these data sets are promising in comparison to baselines. The automatically generated summaries have been evaluated not only by comparing with other methods but also with external reference summaries provided by Wikipedia and Guardian.

Our main contributions in this article are as follows.

1) An event detection system, Mythos, presents events at different levels from small scale to bigger events. Small-scale events detected in online fashion in the form of primitive clusters (PCs) are merged semantically and temporally to obtain bigger events at different levels. To the best of our knowledge, this is the first study of this kind.
2) A summarizer as a part of Mythos uses a long short-term memory (LSTM)-based deep learning model and generates a summary at different levels from the most abstractive level to the most detailed level. These summaries are processed to generate a storyline of the event.

II. Related Work

Mythos incorportaes both event detection (events/subevents/SSevts) and summarization of these event clusters and eventually generating the storyline. We first present related work for online event detection followed by offline event detection and then report work on summarization and storyline generation.

A. Event Detection

Due to increasing use of Twitter for reporting breaking news, planned events, to inform about disasters, and so on, many approaches have been devised for event detection. Petrović *et al.* [8] gave an online event detection approach that uses a locality-sensitive hashing for first story detection by processing each tweet in constant time and space. It uses cosine similarity to assign a tweet to topics for which similarity is above threshold. Assignment of tweet to multiple topics thus leads to noisy creation of events. Aggarwal and Subbian [23] proposed both supervised and unsupervised algorithms that use content and graphical structure of interactions for event detection and their evolution. Another approach proposed by Yin *et al.* [24] represents each tweet by a vector of words and each word frequency is computed by term frequency-inverse document frequency (TF-IDF). A tweet is assigned to a cluster with the minimum distance to its centroid, which is then updated. BNgram approach [25] compares term frequencies in the current time slot with the previous t time slots to find trending topics. This approach becomes ineffective when there are more trending topics present in the stream. K-term hashing approach in [26] operates in constant time and space and scales to very-high-volume streams to find new events. They store terms in a lookup table on the basis of their newness. New events are then detected based on novelty score having k hyperparameters. Comito *et al.* [27] consider textual information of tweets and take into account the importance of words, hashtags, and mentions according to their frequency. The similarity measure is multiplied by a fading function to decay time similar to the approach in [28]. The approach of Comito *et al.* [27] does not deal with the topic fragmentation problem once clusters are generated. Moreover, many resulting clusters are represented by the same set of words and hashtags. A detailed survey for event detection can be found in [29].

The above-mentioned approaches deal with finding large-scale or global events. However, a large event encompasses many small subevents happening together/sequentially. To the best of our knowledge, there is no work reported in the literature, which detects events as well as smaller events within the event. Instead, we find a few approaches that find only small-scale events. TopicSketch [20] employs sketch structure for burst detection. TwitInfo [30] uses a streaming algorithm to automatically discover peaks of high tweet activity from collection of tweets where users can drill down to subevents based on geolocations. StreamCube [31] is an extension of data cube structure with spatial geographical) and temporal hierarchy. Users can explore high level or local (based on geo-location) events at different time granularities. Alsaedi *et al.* [16] provide an event detection framework that integrates classification and clustering for smaller scale disruptive events. Burst detection techniques focus on finding bursts where only a fraction of events can be detected. Most of these approaches focus on finding subevents, missing the context of larger events.

B. Summarization and Timeline Generation

Summary and timeline generation have become very important for online news portals. Many existing approaches handle automatic event summarization for events detected from Twitter stream. LexRank [32] computes sentence importance based on the concept of eigenvector centrality in a graph representation of sentences. LexRank approach works well for small data sets but its efficiency drops drastically with larger data sets. Chakrabarti and Punera [33] summarize highly structured events like sport events using hidden Markov model to learn the underlying hidden state representation of the event from the previous similar events and extract tweets that describe the interesting occurrences in the current event. However, it is not an effective approach as no two sports events are similar to each other. Sharifi *et al.* [21] used a phase reinforcement algorithm for abstractive summarization of tweets and used a single sentence to represent the summary, which is inadequate true summarization of an event. Inouye and Kalita [34] proposed hybrid TF-IDF and clustering algorithms to select top k posts to generate multiple post summaries. However, k is a hyperparameter leading to inappropriate summary of the event if not chosen carefully. Nichols *et al.* [35] used the phase graph model to generate the summary event and considered important moments in an event by identifying tweets with popular discussion points. TOWGS [36] uses a word graph along with decaying windows and pruning technique to generate summaries from tweets. However, query keywords are required to generate the word graph. Sumblr [22] uses the online tweet stream clustering algorithm to cluster tweets and maintain statistics about cluster in tweet cluster vector (TCV). It uses TCV rank summarization technique to generate online and historical summaries of the tweets related to events. This approach does not detect fast-changing subevents within an event and thus does not include them as part of summary. Its execution time increases exponentially with an increase in the size of data. These approaches are either designed to summarize events of a specific domain or use supervised learning, thereby rendering them unsuitable for identifying unknown events. Centriod-based summarizarion approach [37] uses vector representation of both the centroid and each sentence of a document. Then, the sentences closer to the centroid are selected. Generating event time line from the set of tweets is another challenging task as it gives another perspective to look at the event chronologically, which makes the event analysis easier and faster. Diakopoulos and Shamma [38] used time lines to analyze the 2008 presidential debate by Twitter sentiment. Timeline Summarization [39] framework generates an evolution time line of each date from the large collection of time stamped web documents from news web sites like CNN, BBC, and so on. These data sources have well-structured and grammatically correct sentences with very less noise compared to Twitter data. Therefore, this approach may not be able to handle tweets. Alonso *et al.* [40] use pseudorelevance feedback that is generated automatically using social data. However, they have not validated it with external evidence. Alsaedi *et al.* [16] generate time line for the subevents and events.

III. Mythos: The Proposed Framework

Our idea to design Mythos framework is based on the following observations: 1) twitter stream is a high-speed stream conversing about many different events/subevents/SSevts) and so on; 2) many of these subevents are highly related to each other and can be merged together based on their semantic and temporal closeness to create a bigger event; and 3) a big event can be organized as a hierarchy of its subevents and a stream of tweets can be seen as forest of hierarchies of different sizes depending upon underlying subevents. The proposed framework Mythos has three modules: 1) an online incremental clustering that generates clusters of small scale events; 2) a hierarchy generator that generates bigger clusters representing bigger events by iteratively merging small events sharing temporal and semantic proximity; and 3) a summarization module that generates summary for clusters at different levels of the hierarchy and a storyline by merging summaries in chronological order. The whole framework is depicted in Fig. 1.

A. Definition and Problem Model

An event is defined as a real-world occurrence event with: 1) associated time period and 2) a time-ordered stream of Twitter messages, of substantial volume, discussing its occurrence and published during that time [10]. In real world, an event is much more complex and is constituted of subevents, SSevts, and so on. In this article, we represent an event e by a cluster C. We also define the lowest scale event, referred to as a primitive event, which represents an event as a single unit that cannot be divided further. The corresponding cluster of a primitive event is called PC.

Definition 1 (Tweet Stream): A tweet stream is a continuous and temporal stream of tweet objects $T_1, T_2 \ldots T_k$ starting at initial time t_0.

Definition 2 (Tweet): A tweet T_i will be represented as an object $T_i = (T_{iId}, \text{text}_i, T_{io}, T_{iw}, T_{iv})$ where T_{iId}, text_i, T_{io}, and T_{iw} are tweet identifier, tweet text, tweet publish time, and total tweet weight, respectively. T_{iv} is a vector of n-grams and their weights, $te_{i1} : tw_{i1}, te_{i2} : tw_{i2}, \ldots, te_{ik} : tw_{ik})$, where tw_{ij} is that weight of element te_{ij} and te_{ij} is a unigram, bigram, or trigram in T_i.

We distinguish between two types of unigrams: regular weighted unigrams (RWUs) and special weighted unigrams (SWUs), which are hashtags or proper nouns. We use proper nouns such as person, location, and organization name, which are identified using Stanford NER. The unigrams other than SWUs are RWUs. Following the work in [4], we use special weight of 1.5 for SWUs and default weight 1 for RWUs. Authors have shown that the special weight of 1.5 for SWUs is the best for grouping results. The weight for a bigram/trigram (weight of $te_i j$, i.e., jth element of $T_i v$) is computed by aggregating weights of its unigrams, i.e., $tw_{ij} = \sum_{k=1}^{2\ or 3} tw_{ik\ \text{unigram}}$ such that $te_{ik} \in te_{ij}$.

Example1: Tweet (T)-"obama will win Vermont #election 2012" $T_v = \{$Obama: 1.5, win: 1, Vermont: 1.5, #election2012: 1.5, Obama win: 2.5, win Vermont: 2.5, Vermont #election 2012: 3, Obama win Vermont: 4, win Vermont #eletion2012: 4$\}$. Tweet Weight, $Tw = 21.5$.

B. Online Incremental Clustering Algorithm

Online clustering algorithm preprocesses tweets in tweet stream and generates PCs. We use concepts of core set (coreset) and extended set (extset) of n-grams for a cluster. coreset and extset contain m top-weighted n-grams and rest of the n-grams for each n (=1 (SWU, RWU), 2, 3) of the cluster, respectively, in four separate lists in sorted order of weights. We take unigrams to capture important vocabulary of the event and bigrams and trigrams for the purpose to capture the

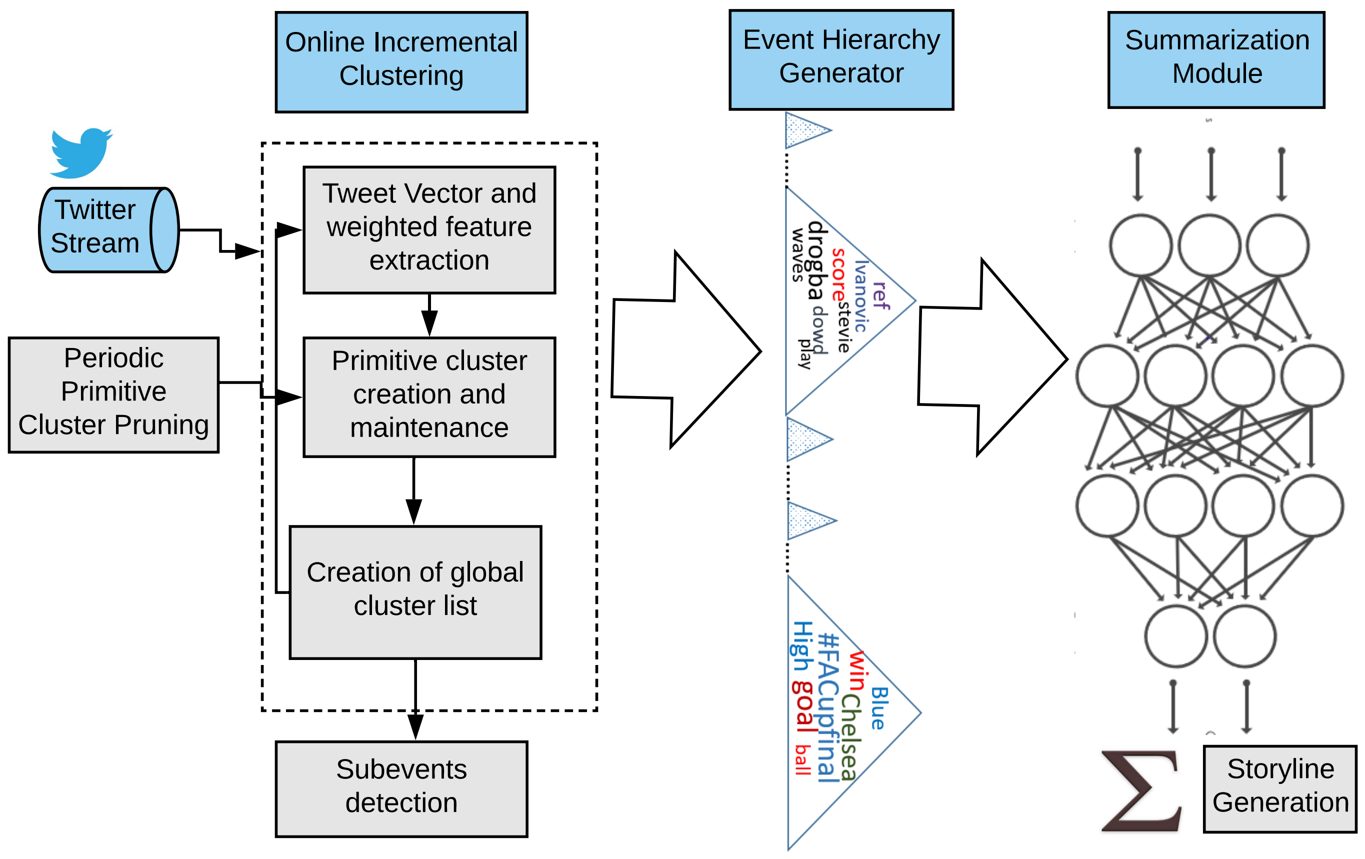

Online Incremental Clustering
Event Hierarchy Generator
Summarization Module
Twitter Stream
Tweet Vector and weighted feature extraction
Periodic Primitive Cluster Pruning
Primitive cluster creation and maintenance
Creation of global cluster list
Subevents detection
Storyline Generation
ref
Ivanovic
score stevie
drogba dowd
waves play
Blue
win Chelsea
#FACupfinal
High goal ball

Fig. 1. Mythos framework.

semantic and contextual meaning of the tweets. Approaches described in [5], [8], and [15] use term representation of the tweet and use clustering based on the nearest neighbors or local spatial distribution of keywords in sliding window. However, these do not consider semantics of the tweets and thus noisy topics. Moreover, methods in [4] and [25]–[27] use n-grams for tweet representation. These approaches use unigrams along with n-grams for $n > 1$ for merging process, which gives more weightage to entities than context and thus leads to mixed or noisy events. Our approach uses only bigrams and trigrams that ensure more weightage to tweet context rather than entities involved in the events and thus obtain pure small-scale clusters/events.

Definition 3 (PC): A PC, PC_i, is represented as an object $\mathrm{PC}_i = (\mathrm{PC}_{i\mathrm{Id}}, \mathrm{PC}_{i\mathrm{text}}, \mathrm{PC}_{iw}, \mathrm{PC}_{iO}, \mathrm{PC}_{iU}, \mathrm{PC}_{i\mathrm{coreset}}, \mathrm{PC}_{i\mathrm{extset}}, \mathrm{Num}_i, LT_i)$, where $\mathrm{PC}_{i\mathrm{Id}}$ is unique PC id, $PC_{i\mathrm{text}}$ is the representative text, PC_{iw} is cluster weight–aggregated weight of tweets in LT_i, PC_{iO} is the cluster creation time, PC_{iU} is the cluster current update time, $\mathrm{PC}_{i\mathrm{coreset}}$ is top k n-grams, and $\mathrm{PC}_{iextset}$ is remaining n-grams (n = 1,2,3) with their weights. Num_i and LT_i are the number of tweets merged in the cluster and list of these tweets, respectively.

1) Tweet Preprocessing: We perform basic text preprocessing before the tweet object is created. We perform the removal of punctuation marks, stop words, and mentions. We also perform stemming using Porter stemmer provided by Stanford's CoreNLP. Based on the observation, we find that tweets with less than three words do not contain useful information. We remove those tweets.

2) PC Creation: For a tweet in a tweet stream, we construct its object T_i. We initiate PC_j for T_i if T_i is not assigned to any of the existing PCs. The features of PC_j are initialized as $PC_{jO} = T_{iO}$, $PC_{jU} = T_{iU}$, $PC_{jw} = T_{iw}$.

$\mathrm{PC}_{j\mathrm{coreset}}$ is created by assigning top m elements of each type SWUs, RWUs, bigrams, and trigrams of the tweet. The remaining tweet n-grams of each type are assigned to $\mathrm{PC}_{j\mathrm{extset}}$. The n-grams (elements) of PC_j and their weights are denoted by e_{jk} and w_{jk}, respectively, for $k = 1, 2 \ldots n$. We keep coreset with a limited number of entries for not to deviate from the original topic of the cluster.

3) PC Updation: A PC, PC_j, is updated each time when a new tweet T_k is added to it. The features are updated as:

$\mathrm{PC}_{jw} = \mathrm{PC}_{jw} + T_{kw}$, $\mathrm{PC}_{jU} = T_{kO}$, $\mathrm{Num}_j = \mathrm{Num}_j + 1$ and $LT_j = LT_j \cup T_k$. The $\mathrm{PC}_{j\mathrm{coreset}}$ and $\mathrm{PC}_{j\mathrm{extset}}$ are reassigned with elements as stated above after updating weights of all elements. For this, the weight w_{js} of each element e_{js} of PC_j is updated as:

$w_{js} = w_{js} + \delta_{sl}.tw_{kl}$ where $\delta_{sl} = 1$ if $e_{js} = te_{kl}$ otherwise 0 and te_{kl} is lth element of T_k.

A tweet T_k is assigned to a PC, PC_j, if similarity between them is greater than the threshold σ. T_k is assigned to every such cluster. Assigning a tweet to multiple clusters helps in establishing the connections between similar topic events. We list PCs in inverted indexes of their bigrams and trigrams.

Similarity Computation Similarity between a tweet and a PC is estimated by the Jaccard coefficient on weights of features of tweet and cluster as Jaccard is considered as one of the best similarity metrics due to its simplicity and discreteness [41].

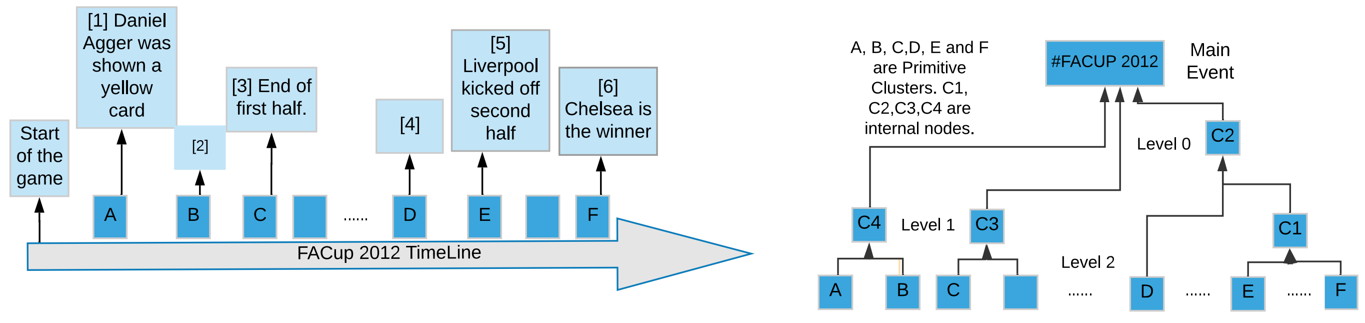

[1] Daniel Agger was shown a yellow card
[2]
[3] End of first half.
[4]
[5] Liverpool kicked off second half
[6] Chelsea is the winner
Start of the game
A
B
C
......
D
E
F
FACup 2012 TimeLine
A, B, C,D, E and F are Primitive Clusters. C1, C2,C3,C4 are internal nodes.
#FACUP 2012
Main Event
Level 0
C2
C4
Level 1
C3
C1
Level 2
A
B
C
......
D
......
E
......
F

Fig. 2. Sample PCs along the Twitter time line and corresponding cluster hierarchy.

Only presence/absence does not provide enough evidence for adding a tweet into a cluster. Therefore, we consider weights for bigrams and trigrams of tweet and the coreset of PC. We do not consider unigrams for computing similarity because tweets are noisy and unnecessarily contain nouns, hashtags, popular words, abbreviations, slangs, phrases, and so on, and thus leads to unwanted merging, thereby getting noisy clusters. Our objective is to obtain pure PCs, which convey a small event only. Unigrams are kept for later use. We, therefore, modify Jaccard for weights as follows:

$$\textsc{Sim}\,(T_i, PC_j) = \frac{\sum_{k\in A} w_{jk}}{\sum_{k\in B} w_{jk} + \sum_{k\in B'} tw_{jk}} \tag{1}$$

where A is the intersection of bigram and trigram of T_i and $\text{PC}_{j\text{coreset}}$, B is bigrams and trigrams of PC_j coreset, and B' is bigrams and trigrams of T_i, which are not present in $\text{PC}_{j\text{coreset}}$.

4) Incremental Clustering: For an object T_i, we use its bigrams and trigrams (elements) to collect all the PCs indexed on these elements. All such PCs are updated if their similarity with T_i is greater than σ or a new PC is created from T_i. All elements of T_i are added to the inverted indexes if they are not already present. The updated PC or newly created PC is indexed against all elements of its coreset. For experiments, σ is set to 0.6. In order to handle high speed and high level of noise in Twitter stream efficiently, cluster pruning is done at regular intervals to discard irrelevant and noisy PCs.

5) Intermittent Cluster Pruning: We observed that if weights of PCs are low, these do not correspond to subevents. This is because they involve noise such as personnel conversation or opinions and do not remain active for long. Therefore, their weights do not grow. Our cluster pruning procedure removes them from the list of currently active clusters. We apply intermittent pruning periodically for period, Prd. If a cluster is inactive with a period Prd and its weight is less than the threshold τ we prune it, otherwise we consider it for historical context. The PCs that are active at the time of intermittent pruning are kept in active PC list, i.e., they continue for live accumulation of tweets.

C. Event Hierarchy Generator

The intuition behind generating event hierarchy is to view subevents/SSevts under an event/subevent or structure of an event plot. This allows us to effectively extract summary and to generate storyline for the bigger events. PCs, having temporal proximity in Twitter stream, tend to have a high semantic similarity. We use this observation to create event hierarchy by merging semantically and temporally close PCs to get subevents and then merging the obtained subevents to get bigger events. The following example depicts event and its subevents and their mapping with the desired hierarchy.

TABLE I
FACup Subevents and Corresponding PCs

Subevent Text	PC Text
[1] Daniel Agger was shown a yellow card for his challenge on Mikel.	[A] Daniel Agger is the second player in Phil Dowd's notebook, for a foul on Mikel #FACupFinal
[2] The first yellow card of the match was shown to Chelsea midfielder John Obi Mikel in the 36th minute, after a mistimed tackle on Gerrard.	[B] And Agger also goes into the Referee's book for a challenge on #Nigeria John Obi Mikel
[3] First half ends with Chelsea- 1 Liverpool-0 score	[C] HT: Chelsea 1-0 Liverpool. Blues in control of 2012
[4] Carroll ran away celebrating his second goal, but after consulting with his assistant, referee Phil Dowd did not award the goal.	[D] Goal disallowed for Liverpool! Beautiful chip to Carroll but Cech makes a great save. Ball had not cross the line. Epic.
[5] Liverpool kicked off the second half.	[E] Kick off 2: Chelsea 1-0 Liverpool Keep The Blue Flag Flying High
[6] Chelsea is the winner of FA Cup 2012.	[F] THE WINNER of #FACupFinal 2012 is CHELSEA.

Example: FA cup Final 2012 (FACup) was played between Chelsea and Liverpool on May 5, 2012 whose sample subevents and corresponding PCs are given in Table I. A, B, and so on, are PCs and [1], [2], and so on, are actual subevents. Details for these can be found in Table I. PC Text is its first tweet text. Subevent text is borrowed from Wikipedia corresponding to the Ground truth, which is in the form of keywords [25].

Fig. 2 shows the PC formation along the Twitter time line for sample tweet stream of FACup and corresponding cluster hierarchy generated by Mythos. In cluster hierarchy, all the leaf nodes A, B, C, D, E, and F are PCs and C_1, C_2, C_3, and C_4 (internal nodes) are clusters. C_3 consists of major game subevents, e.g., A and B of the first half and C_4 consists of end of the first half and public opinions (people generally do discussions during break time of the games) about the first half. C_2 consists of some important moments of the game and C_1, in turn, corresponds to the end of the match and declaration of the winner of the game. It is clear from the cluster hierarchy in Fig. 2 that PCs (e.g., A, B), which share temporal proximity and semantic context, are part of same internal cluster, creating a hierarchy of the event. However, it is difficult to draw clear boundaries between subevents because of the presence of high noise, discussions, and opinions. Our PCs obtained by the proposed online incremental clustering and hierarchy generation algorithms make it feasible to get forest of events.

1) Selection of PCs for Hierarchy Generation: It has been observed that higher weight PCs indicate important subevents. In order to incorporate historical context of all major moments of the events, Mythos selects top K PCs from the current set of potential PCs whose weights are greater than and equal to τ. K is determined by the following equation:

$$K = (\lceil \ln(1 + \text{size}) \rceil)^2 * f \tag{2}$$

where size is the number of active PCs. The factor f is dependent on size for example if $100 < \text{size} \leq 500$, $f = 5$; $500 < \text{size} \leq 1000$, $f = 10$; $1000 < \text{size} \leq 5000$, $f = 15$; and $5000 < \text{size} \leq 10\,000$, $f = 20$. For a larger value of size, we increment f by 5 each time for every increment of 5000 in size. Basically, K has to be large enough so that PCs corresponding to all the key moments are selected. We observed that 60%–70% of the PCs contain all key moments in general. This formula optimizes the number of PCs used for hierarchy creation. We take all active PCs for size = 100 or lower.

We standardize all selected clusters, i.e., we normalize weights of all types of n-grams of every cluster in an interval [0, 1] and store these clusters in a global cluster list L, which is used later for cluster hierarchy generation.

2) Hierarchy Generation: The PCs are merged to create hierarchies. Internal nodes are clusters in the hierarchy represent bigger subevents. They have representation similar to PCs. The following gives the cluster definition, its creation, updation, and merging of cluster process.

Definition 4(Cluster): The ith cluster is an object $C_i = (C_{i\text{Id}}, C_{iW}, C_{iO}, C_{iU}, C_{i\text{coreset}}, C_{i\text{extset}}, \text{Num}_i, LC_i)$, where $C_{i\text{Id}}$ is the cluster Id, C_{iW} is the cluster weight, C_{iO} is the cluster creation time, C_{iU} is the cluster current update time, and $C_{i\text{coreset}}$ and $C_{i\text{extset}}$ are the core set and extended set of n-grams with their weights, and Num_i and LC_i are the number of PCs in C_i and list of these PCs, respectively.

Cluster Creation: A cluster C_{new} is created when two PCs, PC_1, and PC_2 are merged (in first iteration) or two clusters C_1 and C_2 at the same level are merged (second iteration onward). $C_{\text{new}O}$ is the given earliest creation time of merging clusters. LC_{new} is created with PC_1 and PC_2 or the union of LC_1 and LC_2, according to the iteration. New list of n-grams for $n = 1$ (SWU and RWU), 2, 3 are created by taking the union of a respective list of clusters and then performing normalization. $C_{\text{new coreset}}$ is created by assigning top m elements of each type of n-grams, similar to that of PCs.

Cluster Updation: Cluster updation happens when an existing cluster, C_U of higher level, is updated with a PC or a cluster, C_A at the same level. LC_A is appended to LC_U. The creation time, C_{UO}, is the earliest creation time of the constituent clusters and does not change while updation. This is because we merge clusters left to right on a sorted list (according to creation time) of PCs/clusters in each iteration. All type n-grams are updated in the same way as we do in cluster creation.

Algorithm 1 Generate-Event-Hierarchy

```
1  procedure GENERATE-EVENT-HIERARCHY ()
       In  : L, P, EVENTDUR
       Out: HIERARCHY OF CLUSTERS
2      while $(mergeDur <= eventDur)$ do
3          $SL \leftarrow generateSublists(L, p, mergeDur)$
           $L = \Phi$;
4          foreach sublist sl do
5              $L_{new} = \Phi$;
6              foreach $(cluster\ C_j \in sl,\ k \neq j)$ do
7                  Create a new cluster $C_{new} = \Phi$;
8                  foreach $(cluster\ C_k \in sl, k \neq j, k > j)$ do
9                      if $Sim(C_j, C_k) > \alpha$ then
10                         update $C_{new}$ $C_j \leftarrow C_{new}$;
11                     end
12                 end
13                 $L_{new}.add(C_{new})$ ;
14             end
15             $L \leftarrow L \cup L_{new}$;
16         end
17         $mergeDur \leftarrow 2 * mergeDur$;
18         $p \leftarrow p/2$;
19         $Z \leftarrow L$;
20     end
```

3) Similarity Measure for Cluster Merging: Two clusters or PCs are merged if similarity between them is high, which is estimated by Jaccard coefficient on two-term pairs of SWUs and RWUs of *Ci* and *Cj*, i.e., cluster merging is carried out on the basis of the presence or absence of two-term pairs of important unigrams. The similarity measure is given as

$$\text{SIM}(C_i, C_j) = \frac{|A|}{|B| \cup |B'|} \tag{3}$$

where A is set common two-term pairs of C_i and C_j, and B and $B\prime$ are two-term pairs of C_i and C_j, respectively. In the above measure, we do not use weights because the weight of SWUs will always be higher and thus causing the merging of subevents having similar special weighted keywords. Instead, we create context by combing terms of SWUs and RWUs. We normalize SWUs and RWUs after each merge so that broader context can be used in the next merging. We use a similarity threshold, α, for qualifying two clusters for merging.

4) Cluster Merging: Merging procedure takes into account the semantic and temporal closeness among PCs/clusters. For this, a global cluster list L is divided into p number of sublists of equal duration, mergeDur. The set of sublists is called SL. Each sublist $sl \in SL$ is arranged in descending order of creation time. In order to bring in temporal proximity, for each sublist sl, the algorithm merges the PCs having semantic similarity resulting in merged clusters in mergeDur of that list. We then double the mergeDur in every subsequent iteration and repeat merging of clusters in two consecutive lists to obtain bigger events. The pseudocode for the merging procedure is given in Algorithm 1. For simplicity, we have chosen p as 2^r, where $r = 1, 2$. However, the algorithm can be easily generalized for any value of p.

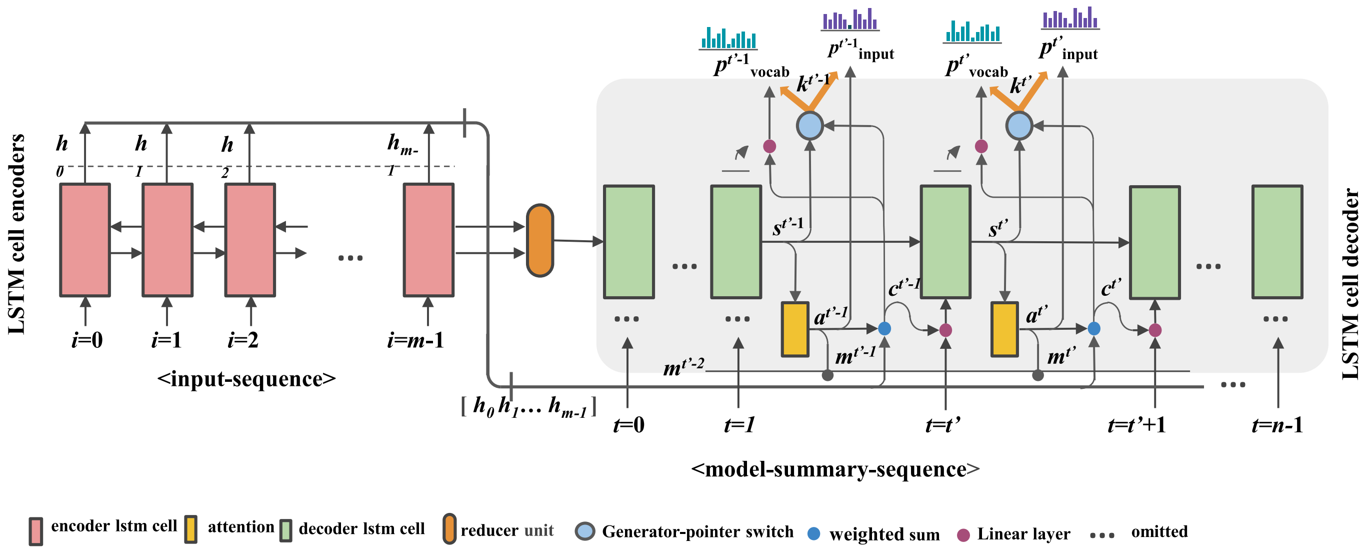


Fig. 3. Summarization model with LSTM encoder and decoder.

5) Hierarchy Structure: The repeated merging of clusters or PCs results in a forest. Each hierarchy of which represents an event and can have two or more levels. External nodes of a hierarchy are PCs or small-scale events of the corresponding event. The internal nodes and root-level nodes are the aggregation of clusters or PCs listed in the following their levels. Top m elements of SWU, RWU, bigram, and trigram capture the content and context of the events/subevents.

6) Hashtag-Based Merging: We also perform hashtag-based clustering to mark one specific event if the tweet stream is having tweets for multiple events.

D. Abstractive Summarization and Storyline Generation

We employ an LSTM-based deep learning model for abstractive text summarization to summarize each cluster in our event cluster hierarchy. The summarization module provides two types of summaries, namely, online and offline. The online summary is generated by feeding the PCs, formed by our online incremental clustering algorithm, to the summarization module. We also present offline summaries at different levels, including full-event summary. The module generates summaries at different levels of hierarchies from the most detailed level (at PCs) to the most abstract level (at root). We process the output of the levelwise summary to get the storyline for the event. Our summarizing module produces grammatically correct, short, and sometimes longer sentences. This is the strength of our summarization module. In many cases, the sentences are abstractive, i.e., new sentences appear in the output, which are formed by conjunction and rephrasing of the original input tweets. This helps us in creating richer storyline sentences. However, the current summarization module produces a good-quality summary when input size is small. It may produce more abstractive sentences if trained on domain-specific data rather than general news data. Section III-D1 represents the model architecture and working.

1) Model Architecture: As shown in Fig. 3, we have adapted the base models described by Nallapati *et al.* [42] and See *et al.* [43]. Our model is composed of an encoder, consisting of a single layer of bidirectional LSTM cells instead of gated recurrent unit (GRU) cells given in [42], and a decoder, containing a single layer of unidirectional LSTM cells. We have used an LSTM-based model. The LSTM unit uses a memory unit to control the flow of information and generates better results with large data, compared to the GRU unit. The following are the main features of our model.

1) *Encoder–Decoder Architecture With Bidirectional RNN Cells:* Vanilla model for taking sequential input (source text) and producing single or multiple sentences.
2) *Attention Mechanism:* It makes use of outputs of all hidden states of the encoder to make a better prediction of each word in each decoder state.
3) *Coverage Mechanism:* It prevents the repetition of sentences in the decoder output and increases the coverage.
4) *Switching Generator-Pointer Mechanism:* It helps in dealing with out-of-vocabulary (OOV) words by predicting if the output word of the decoder is a vocabulary word or is taken by pointing to one of the unseen words in input.
5) *Dropout:* We use a dropout layer over the encoder output while training to prevent overfitting of the model. Since we train using a well-structured set of text and use the trained model on relatively noisy data, using dropouts enhances its capacity to adapt and form better sentences.

Let x be the input document of length $m : (0 \dots m - 1)$. Model will summarize the document x. Let i be the iterator that is iterating over the tokens of the document. At each time-step i, the ith token of the document is given as input as an n-dimensional word embedding to the encoder which produces output h_i. The attention mechanism, originally given by Bahdanau *et al.* [44] for images and then extended for text summarization by Nallapati *et al.* [42] and See *et al.* [43], is used to calculate the importance of each output of hidden encoder state for predicting an output word in the decoder, conditioned over the decoder state s^t output time step t.

The importance of h_i is calculated as a normalized distribution of weights a^t as

$$e_i^t = v^T \tanh(W_h h_i + W_s s^t + b_{\text{attn}}) \tag{4}$$
$$a^t = \text{softmax}(e^t) \tag{5}$$

where v, W_h, W_s, and b_{attn} are the parameters, learned via model training. a^t is then used to calculate a weighted sum of outputs of hidden encoder states, called the context vector $c^t = \sum_{i=0}^{i=m} a_i^t.h_i$. The context vector c^{t-1} is fed to the decoder at each time step t along with the input word (i.e., the previous output word, during decoding or the word in reference summary, during training phase) to generate the decoder output.

To cater the problem of OOV words in summarization, we follow the approach described in [42] of switching generator-pointer mechanism that is essentially a switch, which decides if the output word in the decoder is to be "generated" out of the model vocabulary or "pointed" from the input sequence of words. This is considerably different from the pointer-generator mechanism employed in [43] to solve a similar hurdle. It predicts a word from the combined probability distribution of vocabulary and input words. Our approach helps us in getting sentences with a better abstraction. However, as reasoned in [42], pointer-generator helped in achieving proper structure and correctness in trade for abstraction, by essentially "copying" long sequences of words from the input. In light of our case of working with tweets, the input text is not structured, and hence, we aim to "avoid" copying stretches of obscure words. Although the abstraction we are able to achieve is less than human-curated summary, we are able to filter noise from the input and at instances and able to join sentences that share a common context. The switch k^t is modeled as

$$P(k^t = 1) = \sigma(v_p(W_{ps}s^t + W_{pe}E[o^{t-1}] + W_{pc}c^t + b_p \tag{6}$$

where $P(k^t = 1)$ is the probability at a decoder time-step for the switch to be "ON." $E[o^{t-1}]$ is the embedding of last output word in the decoding sequence. W_{ps}, W_{pe}, and W_{pc} are the learnable parameters. When the switch is "ON," the output is the word generated from the model vocabulary as it is normally supposed to be by calculating probability distribution over the words in the model vocabulary, as

$$P_{\text{vocab}}^t = \text{softmax}(v_u[s^t, c^t] + b_u) \tag{7}$$
$$o^t = \text{argmax}_j\left(P_{\text{vocab}}^t(j)\right) \ for \ j = 0, 1 \ldots |V| \tag{8}$$

where P_{vocab}^t is the probability distribution is over the words in the model vocabulary V, o^t is the index of the word in V that is output at the current time step, while v_u and b_u are the model learnable parameters. However, in the event when k^t is turned off, an OOV word from the input sequence is pointed to. The pointer value is given by the max in the probability distribution P_{input}^t that is derived from attention distribution as

$$P_{\text{input}}^t = \text{softmax}(c^{t-1}) \tag{9}$$
$$p^t = \text{argmax}_j\left(P_{\text{input}}^t(i)\right) \ for \ i = 0, 1 \ldots |m-1| \tag{10}$$

when k^t is OFF, and p^t is the index of the token in the input document. For training, the model summary includes pointers to the token in the input document only when that word is not present in the model vocabulary. The loss is calculated as a log-likelihood function, which is conditioned on whether or not a particular word in the model summary is present in model vocabulary

$$\log P\left(\frac{y}{x}\right) = \sum_t \left(g^t \log\left(P\left(\frac{y^t}{y_{\text{prev}}, x}\right) * P(k^t)\right) + (1 - g^t)\right.$$
$$\left. * \log\left(P\left(\frac{P^t}{y_{\text{prev}}, x}\right) * (1 - P(k_t))\right)\right) \tag{11}$$

where g^t is conditional and is set to 0 when the word from model summary is not present in the model vocabulary, while y_{prev} are the previous words in the model summary.

We also employ the coverage mechanism as a means to avoid redundancy in the generated output by penalizing repeated and excess attention to a particular output of the encoder hidden state at any time step t. The coverage m^t at a time step t in the decoder is given as the sum of all attention distribution $a^{t\prime}$ for previous time steps. The coverage vector is then used for improving the calculation of attention distribution $a^{t\prime}$. We model coverage as given in [43], i.e., define a coverage loss and use it along with the above-mentioned loss during training

$$m^t = \sum_{t\prime=0}^{t-1} a^{t\prime} \tag{12}$$
$$\text{coverage} - \text{loss}^t = \sum_i \min\left(a_i^t, m_i^t\right). \tag{13}$$

Hence, the loss at each time step t is equal to the sum of the negative log-likelihood function and weighted coverage loss

$$\text{loss}^t = \log P\left(\frac{y}{x}\right) + \lambda * \text{coverage} - \text{loss}^t. \tag{14}$$

We use $\lambda = 1$ in all our experiments [43]. Finally, the total loss for one pair of input sequence x, and model summary y, can be stated as the mean of loss over all the decoder time steps

$$\text{loss} = \frac{1}{n} * \sum_t \text{loss}^t. \tag{15}$$

2) Cluster Text Preprocessing: The text (tweets) present in the clusters requires preprocessing before submission to the summarization module. The preprocessing is necessary due to: 1) a large number of spelling and grammatical errors; 2) improper and casual sentence structure and; and 3) high noise that is irrelevant to the topic. Also, with the motivation of generating an unbiased summary, we tend to drop tweets that exhibit a personal opinion in first person, containing words like "i," "we," "my," "me," and very short tweets with less than five tokens. We strip reserved tags like RT (retweet) and MT (modified tweet), hashtags, and hyperlinks. The refined set of tweets in each cluster is input to our summarization model. We also postprocess the output of the summarization model to avoid topic redundancy and to get an overall connected structure of our summary.

Chunking: While generating summaries for internal nodes, a common problem is summary length for large input.

We observed that the quality of sentences decreases (syntax construction and completeness) with the increase in input size and generated summary is unable to cover all the important aspects of the input text primarily because of the architectural limitations of the model. If we increase the size of the model decoder, the model does not work well on clusters with a small number of sentences. A way around this problem is to feed data to the model in chunks for large clusters. Experimental observations determine using the chunks of sizes five–ten sentences (50–200 tokens) produces output with good cluster topic coverage.

3) Postprocessing of Summary Sentences: The summarization model generates distinctive sentences for each cluster of tweets. However, these sentences suffer from improper syntactic construction because of retweets/modified tweets, irrelevant tags, opinions, and so on, redundancy, i.e., sentences appear repeatedly in the model output, incomplete sentences, and so on. This deems it improper to be represented in the final storyline and hence has to be filtered out from the pool of probable sentences from that cluster. The postprocessing steps are as follows.

Grouping: We group the sentences that have a high similarity score based on the modified Jaccard coefficient,$|a \cup b|/\min(|a|, |b|)$, where a and b are the set of words in sentences 1 and 2, respectively. We aim to pick a sentence that is most informative and complete from sentences in a group. The threshold value for Jaccard is chosen in a way so as to balance between not losing slight variants of factually different information as well as not keeping repeated information.

Source Copy Percentage: Source copy percentage gives the normalized score of the amount of source text that is represented in the generated sentence. The high value of source-copy percentage adequately covers a large portion of the source in the generated sentence. The score is considerably less for sentences that lack good coverage or are incomplete and thus the metric is quite effective for selecting the best sentences to represent a group.

SWU and RWU Scores: This metric calculates the sum of the weights of all SWUs and RWUs present in each generated sentence. The weight of each word in SWU set and RWU set is the relative importance of that word in the cluster of tweets they appear in. Sentences with greater scores have greater importance. The score is normalized by the sum of weights of all the SWUs and RWUs, respectively, occurring in the source text of the cluster

$$\text{SWU/RWU score} = \frac{\sum_{x:\text{SWU in sentence}} \text{weight}(x)}{\sum_{x:\text{SWU in source}weight}(x)}. \tag{16}$$

Matching: Each sentence from each group of the output is tagged with creation time and the weight of the corresponding PC. We use the creation time to get the earliest reference of the topic in the tweet stream. Internal clusters in the hierarchy have a large number of tweets from different clusters under them. The generated output summary sentences are matched with the corresponding tweets in the input for assigning timestamp and weight to the sentences, thus inducing ordering in the output pool of summary sentences. After postprocessing, summary sentences corresponding to each PC/cluster are well formed and give a good representation of subevents/events. In order to increase the readability of the final summary, we also perform a couple of enhancements like replacing twitter handles with their actual account names and correct commonly misspelled dictionary words.

We generate summaries at different levels for evaluation. In order to generate a summary at a level, all the cluster's outputs at that level are combined and ordered chronologically using the earlier described time stamping to generate meaningful summary.

4) Timeline Generation: We use summary sentences of PCs to generate a timeline. The base-level clusters have the maximum information about the minutest of occurrences in an event. We use the same process that is used in summary generation. Our timeline generation module can work online as our framework is capable of continuously clustering the tweets and preserving their chronological order.

IV. Experimental Analysis

We use four data sets, given in Table II, to evaluate the Mythos framework, namely, FA Cup final, Super Tuesday (ST) Primaries, US Elections (USEs), and SB. The first three data sets are publically available with the ground truth [25]. We will publish the data set and make code available on GitHub when the manuscript is accepted.

FA Cup (FA) is the main knock out competition in English football and in 2012, and the final game was played between Chelsea and Liverpool, which was won by Chelsea with goals from Ramirez and Drogba. The available ground truth for this event has 13 topics including three goals, bookings, start, half time, and end of the game. The duration of the game is 90 min with extra 30 min, called extra time, when the deciding leg (or replay of a tie) has not produced a winner by the end of normal or full-time.

In U.S., the President candidate for each political party is selected by a series of primaries, which are elections held in individual state where party members vote for the candidate of their preference. At the end of this process, the candidate with most delegates elected by each state becomes the presidential nominee. In 2012, Alaska, Tennessee, Vermont, Georgia, Idaho, Massachusetts, North Dakota, Ohio, Oklahoma, and Virginia all voted on ST, i.e., March 6, 2012. Mitt Romney, Ron Paul, Newt Gingrich, and Rick Santorum were four republican presidential candidates. The ground truth includes 22 topics with stories like projection about a candidate win or televised speeches of several candidates.

USE held on November 6, 2012 was won by President Barack Obama and Vice President Joe Biden defeating Mitt Romney and Paul Ryan, respectively. The ground truth included announcements by U.S. television networks for outcomes of presidential race in particular states, senate race results, and Obama Victory speech.

SB 2018 is collected from public streaming API of Twitter from February 3 to February 5, 2018. The SB is the annual championship game of the National Football League (NFL). The game is the culmination of a regular season that begins in the late summer of the previous calendar year. The duration

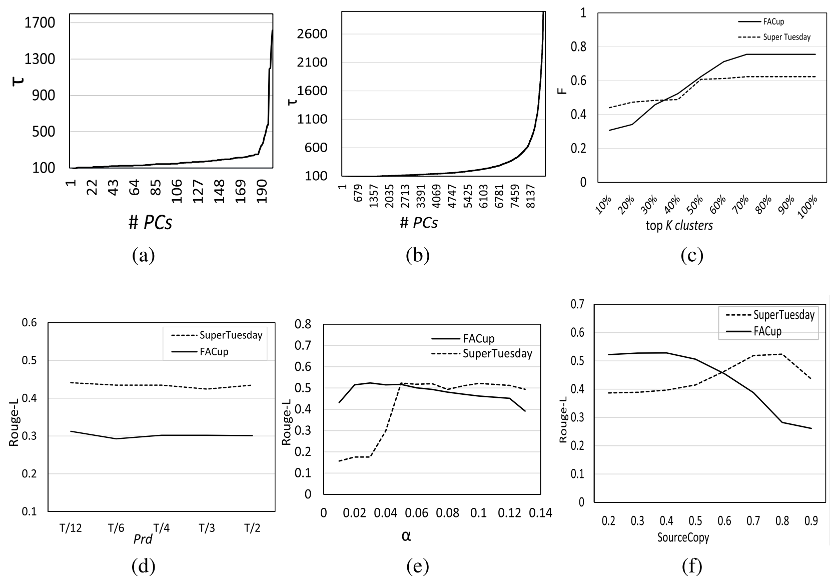


Fig. 4. Variation in weight of PCs of (a) FA Cup, (b) ST, (c) effect on F-measure with varying K, (d) effect on Rouge-L with different pruning time Prd, (e) effect of α on rouge-L score, and (f) effect of source copy on rouge-L score.

TABLE II

DATA SETS AND THEIR SIZES

Datasets	#Tweets	#Tweets used in Experiments
FA Cup	148,652	51,640
Super Tuesday	474,109	297,937
US Election	1,247,483	485,516
Super Bowl	2,151,916	195,731

of the game is 3 h and 44 min. We used reputed news web sites like Guardian, USA news, and Wikipedia to track the start and end time of the event and selected a subset of tweets corresponding to the actual time frame. Ground truth for event detection was generated using the scoring summary section available on Wikipedia [45] page for each quarter of the game.

Aiello *et al.* [25] can be referred for more details about first three data sets. Authors have used tweets corresponding to the time slots, which have at least one topic in the ground truth. Tweets other than these timeslots are not used in their experiments. We also use the same data for our experiments for fair comparison. However, we give entire data as a tweet stream and applied our proposed algorithm on it to detect events and matched them with the ground truth.

Ground Truth for Summarization: We have collected external reference summaries for FA Cup [46], ST [47], USE [48], and SB [49] event from Guardian. We have also collected external reference summaries for FA Cup [50] and SB [45] from Wikipedia. Wikipedia summary is not available for ST and USE. In general, Wikipedia summaries are shorter, well-structured, and more abstract in comparison to Guardian summaries.

A. Measures for Evaluation

We follow the approach in [25] to evaluate the proposed framework Mythos for event detection. We use keyword precision, keyword recall, and F-measure. Rouge-*1* and Rouge-L are used for evaluating summarization.

Keyword Precision (P): Correctly detected keywords/total number of detected keywords.

Keyword Recall (R): Correctly detected keywords/total number of keywords in ground truth.

F-Measure (F): $2*(P*R)/(P+R)$.

The cluster obtained is matched with ground-truth topics and the topic is chosen based on the best match. The average precision/recall/F-measure of all the events are computed by macro averaging the individual scores. A detected term matches a ground truth keyword when their Levenshtein similarity is 0.8 for handling the spelling variations.

Rouge-1 and Rouge-L: Rouge-1 refers to the overlap of *1*-gram between the generated and reference summaries. Rouge-L score is based on the longest common subsequence (LCS), which accounts for sentence-level structure using the longest co-occurring n-grams.

B. Parameter Tuning

In this section, we describe how we tune the parameters on training data sets and give an approximate value or range of values to be used for test data sets.

τ and K: These parameters are used for noise filtering during online clustering. PCs with weight lower than τ are discarded as these PCs correspond to noise or irrelevant conversations present in the tweet stream. Moreover, it has been observed that tweet streams have only a small fraction of meaningful tweets, which can be established through scree test [51] on weights of PCs. Elbow-nick present in weight curves in Fig. 4(a) and (b) shows that PCs to the right of the elbow-nick are relevant and have meaningful tweets and PCs on the left-side are noisy and irrelevant. A value of τ, which is little left side of the elbow-nick is good for not missing any meaningful PC. We choose $\tau = 200$. Although we use (2) for

TABLE III
PRECISION, RECALL F-MEASURE FOR FA CUP, ST, USE, AND SB DATA SETS

Datasets	**ST**		**USE**		**SB**		**FA**	
Method	**P/R**	**F**	**P/R**	**F**	**P/R**	**F**	**P/R**	**F**
BNgram	0.6286 / 0.6471	0.638	0.4050 / 0.5632	0.4712	0.4257 / 0.5672	0.4864	0.2989 / 0.5778	0.3939
OnlineDetect	0.3059 / 0.8998	0.457	0.5687 / 0.9116	0.7004	0.1561 / 0.4849	0.2361	0.1224 / 0.7941	0.212
KTerm	0.1819 / 0.9707	0.306	0.3708 / 0.9767	0.5375	0.0384 / 0.6327	0.0725	0.0879 / 0.8048	0.1585
Mythos	0.7954 / 0.7193	**0.755**	0.7971 / 0.6313	**0.7046**	0.7569 / 0.6794	**0.7161**	0.5536 / 0.7138	**0.6236**

selecting K, we show that our proposed equation corresponds to the best value of K. For this, we plot F-measure for event detection using different values of K. Fig. 4(c) shows that the F-measure improves when K increases. When $K >= 70\%$ of the potential cluster list, L, F-measure remains the same.

Prd: To handle high-speed Twitter stream, intermittent pruning is done to dynamically select the most important subevents for the cluster pruning duration and discard the remaining subevents. Fig. 4(d) shows the variation in Rouge Scores for summaries with different pruning times. We have considered the Rouge-L score for the most detailed (PC) level. There is no change in the summary quality with varying Prd.

α: Fig. 4(e) presents the effect of α on summaries. FA Cup data set being a fast-paced sports event has tweets written in shorthand with lots of improper grammar. However, ST is a relatively slower, well-organized, political event that calls for more descriptive tweets and thus has variations in sentences of the event. Therefore, we recommend lower values of α (0.02–0.05) for descriptive tweets and higher value of α (0.08–0.12) for sports events. Rouge-L used here is for the most abstract (root) level. Similar behavior is observed for other data sets.

Sourcecopy: Effect of sourcecopy on event summaries is depicted in Fig. 4(f). The recommended range of source copy is lower for sports data sets and it is high for political data sets (having descriptive tweets). We recommend source Copy threshold as 0.4 for FA and 0.8 for ST. Rouge-L is used for the most detailed level (PC).

V. PERFORMANCE EVALUATION

A. Event Detection

We first present the comparison of Mythos with baselines for small-scale event detection. We have considered the BNgram approach [25], K-term Hashing [26], Online clustering approach [27], and LexRank [32] for comparison. In this experiment, we use clusters at the lowest level (at PC level). Table III presents precision, recall, and F-measure of baselines and Mythos. It shows that F-measure is high in both training/testing and sports/political data sets for the proposed method. All baselines show high recall but poor precision and thus result in low F-measure. The clusters produced by Mythos possess both high precision/recall and cover all subevents. Our PC creation and updation module allows only related tweets to be merged and raise pure clusters, i.e., one cluster depicting only one small event. OnlineDetect approach and BNgram approach form bigger clusters and thus show high recall as compared to their precision. Their precision is higher for political data sets as compared to sports data sets as subevents in these political events are for longer duration and highly related to each other (because of politician names and place of election). For KTerm, we compare the output of the bloom filter against the ground truth for each event. The number of keywords obtained from the output of bloom filter is quite high and thus KTerm has large difference between its output and the ground truth. It results in high recall and very low precision. This scenario is more prominent in sports event due to their short ground truth as compared to political events.

TABLE IV
FAST HAPPENING SUBEVENTS DETECTED BY MYTHOS FOR FA AND USE

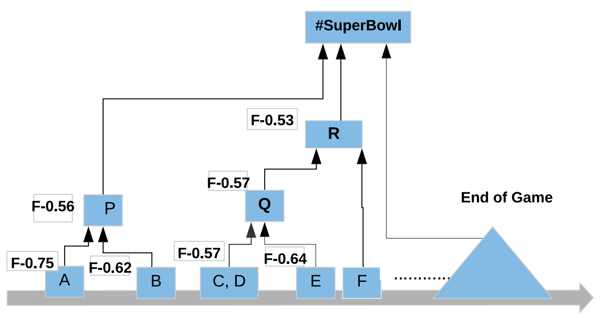
Event	**Detected subevents**	**#tweets in PC**	**PC duration**
FA	Laura Wright sings the National Anthem.	8	0 Sec
	#cfc fans sang the National Anthem. Many #lfc fans didn't, even jeering it. They jeered the National Anthem. Sad.	63	20 min
USE	BREAKING: Democrat Claire McCaskill wins Senate seat in Missouri, beating out Republican Todd Akin.	3	0 Sec
	Female Iraq vet Tammy Duckworth wins 2012 for Illinois's 8th congressional district.	46	11 min
	Tim Carper of Delaware, retains, his senate seat	3	0 Sec

B. Fast Happening Subevents

Mythos framework is able to detect all the important subevents of an event. In order to verify Mythos's capability for detecting fast-changing subevents, we have chosen publically available data sets whose ground truth is available and cover most of its subevents/SSevts. However, there are some subevents/SSevts detected by our framework whose tweeting duration was short or they are not listed in the ground truth. These subevents are available on news websites of "The Telegraph" and "The Washington Post." These subevents are detected after filtering and pruning of the generated PCs. Table IV shows such subevents along with other subevents for FA and USE, which are detected by Mythos but not listed in the ground truth. PC duration is the difference of PC_{iO} and PC_{iU}.

C. Hierarchy Evaluation

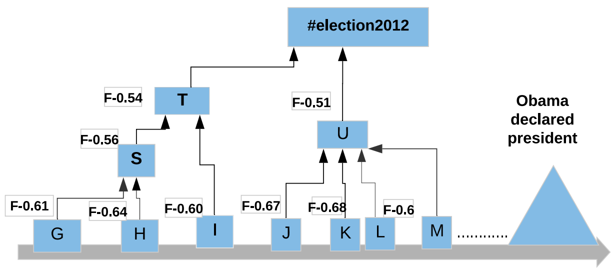
In order to evaluate the hierarchies, we present results on test data sets SB and USE. There are four quarters in SB,



Event-SuperBowl
P. Pre game sub events
A. Pink sing Nation Anthem in before start of game
B. Players not Kneel during National Anthem
Q. First-Quarter sub events
C. 25-yard field goal by Jack Elliot
D. 26-yard field goal by GostKowaski
E. Alshon Jeffery 34-yard touchdown reception from Foles. Elliott kick no good (wide right)
R. First Quarter and Discussion and Opinions
F. Discussion about good performance of Foles in game

Summarization Model Output
P. Pre game sub events
A. Pink just sang the national anthem at the SuperBowl while battling the flu .
B. players that decided to stand for our flag and national anthem today.
Q. First-Quarter sub events
C. Elliott makes all three FG tries (25, 42, 46 yards) in the Eagles ' 41-33 victory for Philly's first Super Bowl win.
D. 3-0 Philly it was Elliott ... then it was Gotti ..Stephen Gostkowski answers Elliott's 25 yarder with a 26 yarder of his own ... tying Super 52 at 3 and fans in Memphis .
E. Elliot, extra point , leaving the Eagles up , 9-3 , and upsetting thousands of people with 0-3 in their Super pools.
R. First Quarter and Discussion and Opinions
F. QB Nick has three passing touchdowns and one receiving as the Eagles beat the Patriots in the SuperBowl 41-33

Fig. 5. Internal cluster details with F-measure and corresponding summarization model output for SB.



Event-USElection
S. Prediction of win for Candidates
G. Obama projected to win Hawaii, Washington, California
H. Romney projected to win Idaho.
T. Prediction of win for Obama
I. Obama projected to win Michigan and New York.
U. Legalization of gay marriage and marijuana.
J Same-sex marriage leaglized in Maryland.
K. Marijuana legalized in Colorado and Washington
L. Elizabeth Warren first female senator from Massachusetts.
M. Discussion about election.

Summarization Model Output
S. Prediction of win for Candidates
G. Barack Obama will win California, Washington and Hawaii .
H. Mitt Romney is the projected winner in Idaho .
T. Prediction of win for Obama
I. President Obama wins Michigan , New York , and New Jersey .
U. Legalization of gay marriage and marijuana.
J Maryland votes to legalize same-sex marriage ,
K. Meanwhile , Colorado and Washington both legalized marijuana
L.Elizabeth Warren just became the first woman to represent MA in the senate .

Fig. 6. Internal cluster details with F-measure and corresponding summarization model output for USE.

which can be considered as four subevents, many SSevts under these subevents and subevents like performances by artists like Pink singing US national anthem, players kneeling during anthem, and so on. USE has subevents like prediction and conformation of winning or losing for Obama and Romney at various states, making same-sex marriage, use of marijuana legal in many states, and so on. As shown in Fig. 5, P, Q, and R are the main subevents denoting pregame subevents, subevents of the first quarter, and discussion regarding Foles's performance in the first quarter, respectively. A and B are SSevts for P, while C, D, and E are SSevts for Q subevent. F-measure is calculated using Coreset unigrams against external ground truth for subevents and SSevts. Fig. 6 shows the similar details about USE events. It can be observed from Figs. 5 and 6 that F-measure is high for all internal clusters and is the highest at the lowest level and decreases as we go up. This is because the abstraction keeps increasing for higher levels but ground truth granularity remains the same at all levels.

D. Summarization

We use the CNN/Daily Mail data set [52] to train our summarization module. CNN News has a total of 92.5k pairs of stories and summaries, while Daily mail has around 220k pairs. The in-model vocabulary contains 50 000 words for both target and source, while other words (NER and OOV) are handled by the "switching generator-pointer" mechanism. We train our model with a hidden dimension of encoder and decoder fixed at 400, while the dimension of input and output embedding is set at 200. The word embeddings are not pretrained and are learned during the complete model training. Optimizer used is AdaDelta, with initial learning rate of 0.001. Also, we use gradient clipping at the maximum gradient norm of 2, but we do not use any normalization technique on the input text. The training is done for around 20 epochs with a batch size of 32, which keeps reducing by half at appropriate intervals to provide finer tuning. We use the coverage mechanism with weightage $\lambda = 1$ and training for around 5000 iterations in the end. Our training procedures and specifications are the same as those used in [43]. For decoding, we use beam search of width 3 to generate the output at each cluster.

1) Mythos Summary Analysis: In the first set of experiments, we report the comparison of Mythos generated summaries with that of the external sources for whole events (see Table V). The whole event summary can be obtained at two levels, i.e., the most abstractive (root, denoted by R) level and at the finest (PC, i.e., the lowest level denoted by L) level. This is because some of the events can merge directly with the higher levels and may not be present at middle levels. For sports events, the F-measure is better for Wikipedia at the most abstractive level and it is better for Guardian at the lowest level because Wikipedia summaries are shorter, well-structured, and

TABLE V

EVALUATION OF MYTHOS SUMMARIES OF FULL EVENT GENERATED AT MOST DETAILED (PC)/LOWEST (L) LEVEL AND MOST ABSTRACT (ROOT) / HIGHEST LEVEL (R) AGAINST EXTERNAL REFERENCE SUMMARIES FOR BOTH TESTING AND TRAINING DATA SETS

			Rouge-1		Rouge-L	
Datasets	**External Reference**	**Level**	**P/R**	**F**	**P/R**	**F**
SB	**Guardian**	**L**	0.4127 / 0.4727	0.44066	0.4064 / 0.4654	0.4339
	Guardian	**R**	0.4131 / 0.4609	0.43569	0.4039 / 0.4506	0.4259
	Wikipedia	**L**	0.2985 / 0.6317	0.40544	0.2708 / 0.5731	0.3678
	Wikipedia	**R**	0.3075 / 0.6339	0.41415	0.2732 / 0.5630	0.3679
FA	**Guardian**	**L**	0.6062 / 0.4802	0.53596	0.5907 / 0.4679	0.5222
	Guardian	**R**	0.6921 / 0.2112	0.32362	0.6685 / 0.2039	0.3126
	Wikipedia	**L**	0.2138 / 0.6342	0.31974	0.1969 / 0.5841	0.2945
	Wikipedia	**R**	0.3685 / 0.4211	0.39305	0.3303 / 0.3774	0.3523
USE	**Guardian**	**L**	0.5211 / 0.6455	0.57664	0.5099 / 0.6318	0.5644
	Guardian	**R**	0.7082 / 0.2928	0.4143	0.6856 / 0.2835	0.4011
ST	**Guardian**	**L**	0.5216 / 0.6198	0.56644	0.4259 / 0.6799	0.5238
	Guardian	**R**	0.6074 / 0.3633	0.45464	0.5893 / 0.3525	0.4411

TABLE VI

EVALUATION OF MYTHOS GENERATED SUMMARIES OF MIDDLE-LEVEL SUBEVENTS FROM SB AND USE AGAINST EXTERNAL REFERENCE SUMMARIES FROM GAURDIAN

		Rouge-1		Rouge-L	
Datasets	**SubEvents**	**P/R**	**F**	**P/R**	**F**
SB	First quarter	0.3523 / 0.8426	0.4968	0.3381 / 0.8085	0.4767
USE	Legalise same-sex marriage in Maryland and Colorado, Washington vote to become first US states to legalize marijuana.	0.4706 / 0.5128	0.4908	0.4353 / 0.4744	0.4539

TABLE VII

COMPARISON OF SUMMARIES GENERATED BY MYTHOS AND BASELINES AGAINST EXTERNAL REFERENCE SUMMARIES FOR ALL DATA SETS. BEST SCORES ARE IN BOLD

Methods		**TOWGS**		**Centroid based**		**Sumblr**		**Mythos**	
Ref Summary	**Datasets**	**P/R**	**F**	**P/R**	**F**	**P/R**	**F**	**P/R**	**F**
Guardian	**ST**	0.3233 / 0.2309	0.2694	0.3216 / 0.5152	0.396	0.4930 / 0.3961	0.4393	0.4259 / 0.6799	**0.5238**
Guardian	**USE**	0.5841 / 0.1253	0.2063	0.5261 / 0.1498	0.2332	0.5434 / 0.2181	0.3112	0.5099 / 0.6318	**0.5644**
Guardian	**FA**	0.4406 / 0.1541	0.2283	0.5621 / 0.1690	0.2598	0.3557 / 0.1528	0.2138	0.5907 / 0.4679	**0.5222**
Guardian	**SB**	0.5676 / 0.2063	0.3026	0.3614 / 0.2039	0.2607	0.2466 / 0.4922	0.3286	0.4064 / 0.4654	**0.4339**
Wikipedia	**FA**	0.5156 / 0.1424	0.2231	0.3033 / 0.3415	0.3212	0.1917 / 0.2066	0.1988	0.3303 / 0.3774	**0.3523**
Wikipedia	**SB**	0.2676 / 0.3063	0.2856	0.3092 / 0.1618	0.2125	0.2264 / 0.4191	0.2939	0.2732 / 0.5631	**0.3679**

more abstract in comparison to Guardian summaries. Guardian summaries are more descriptive and contain tweets quoted by famous personalities regarding the event. This difference between two levels is more in FA because data in FA are lesser than SB. Moreover, the F-measure in FA for Wikipedia is less than that of Guardian. This is because FA is a short duration event and thus has less data.

We also compare Mythos generated summary for subevents (like the first quarter of SB), which map with internal clusters of the middle level. The F-measure is computed with respect to the Guardian summary and given in Table VI. The F-measure at internal clusters is less than that of at L level but more than that at R level in accordance with the expectation, i.e., abstraction increases as we go from L (finer granularity) to R (coarser granularity).

2) Comparison With Other Approaches: We compare Rouge-L of Mythos with that of baselines for full events (see Table VII). We have used our results at R level for Wikipedia and L level for Guardian summaries. Comparison with LexRank is shown in Fig. 7. LexRank approach works well for small data sets but its efficiency drops drastically with larger data sets. This method can be adapted to handle streaming data [22] by using a sliding window

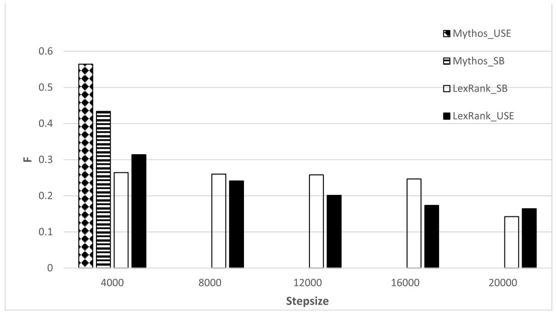


Fig. 7. Comparison of summaries generated by Mythos and LexRank for USE and SB.

TABLE VIII
GENERATED STORYLINE FOR USE. THE GROUND TRUTH TIME IS IN CENTRAL STANDARD TIME

SSEvts	**GroundTruth / Predicted Time**
First projects results in Election 2012 come in : Vermont for Obama , Kentucky & Indiana for Romney.	12:03-12:21 AM
Bernie Sanders re-elected as senator in Vermont.	12:16 AM
Romney RACE CALL : Romney wins West Virginia.	12.31 AM
Obama wins Connecticut , DC , Delaware , Illinois , and Maryland !.	1:00 AM
Romney is the projected winner of Arkansas & Tennessee lead.	1:20 AM
Romney wins Alabama but lets be honest ,there are no real winners in Alabama.	1:40 AM
President Obama wins Michigan , New York , and New Jersey.	2:00 AM
Obama wins Michigan and New Mexico ; too close to call in Colorado , Wisconsin.	2:30 AM
Romney wins Kansas , Louisiana, North Dakota , South Dakota and Wyoming.	2:00 AM
Obama is leading in Ohio , New Hampshire and Pennsylvania.	2:10-2:50 AM
CONGRATS Sherrod Brown wins Senate seat in Ohio.	2:20 AM
Barack Obama won swing state of New Hampshire election. 2012	2:50 AM
Romney wins Utah	3:00 AM
Republican Orrin Hatch wins Senate seat in Utah.	3:03 AM
Democrat McCaskill wins Senate seat in Missouri , beating out Republican Todd (via The Associated Press) Akin.	3:25 AM
News projection : Romney wins Idaho and North Carolina	3:50-4:00 AM
Obama projected winner in CALIFORNIA , WASHINGTON , HAWAII.	4:00 AM
Barack Obama has been re-elected to a second term as President of the United States.	4:10 AM
The Associated Press reports that Maine legalized same-sex marriage.	4:50 AM
Meanwhile , Colorado and Washington both legalized marijuana.	4:50 AM
The Seattle Times : Maryland , Maine approve gaymarriage and Washington is poised to.	5:20 AM
Romney won Idaho and Montana. Results 238 Obama and Romney 191.	5:40 AM
Governor Romney has called President Obama to concede.	5:50 AM
The best is yet to come " - Barack Obama tells supporters in Chicago victory speech.	6:40 AM

TABLE IX
GENERATED STORYLINE FOR SB. THE PREDICTED TIME AND ACTUAL TIME IS IN CENTRAL STANDARD TIME

SSEvts	**Predicted Time**	**Actual Time**
Pink just sang the national anthem at the SuperBowl while battling the flu.	5:30 PM	5:25 PM
Eagles 3 , Patriots 0 in the 1st Quarter.	5:47 PM	5:42 PM
First Gostkowski answers Elliott's 25 yarder with a 26 yarder of his own ... tying Super Bowl.	5:59 PM	5:54 PM
Jeffery scored the first touchdown of the SuperBowl with an outstanding contested catch.	6:05 PM	6:01 PM
The Eagles lead 9-3 after missed extra point.	6:05 PM	6:01 PM
FlyEaglesFly with a 15-3 lead and 8: 48 to go in the first half.	6:33 PM	6:33 PM
White with a 26 - yard run to cut the deficit to just 15-12 , Eagles lead.	6:54 PM	6:53 PM
QB Nick catches the TD pass on 4th down , PHI now leading 22-12 with 30 seconds left till halftime.	7:06 PM	7:05 PM
Justin Timberlake under fire for performing a duet with a projection of Prince at the Superbowl half-time.	7:15 PM	7:13 PM
Which Brady finds wide receiver Chris Hogan in the end zone to cut the Philadelphia Eagles ' lead to 29-26 late in the third quarter.	8:16 PM	8:12 PM
Tom lobs the pass TD pass up to Rob Gronkowski , giving New England a narrow 33-32 lead.	8:38 PM	8:37 PM
the Eagles ending up with Ertz scoring will go down in Super Bowl History.	8:47 PM	8:46 PM
MORE : QB Nick has three passing touchdowns and one receiving as the Eagles beat the Patriots in the SuperBowl 41-33.	9:20 PM	9:18 PM
new name on the Vince trophy ! Philadelphia Eagles are Super Bowl champions for the first time since 1933.	9:30 PM	9:30 PM

scheme to summarize tweets of the window. Window moves further by a step size and tweets in a new window are then summarized. The window size is set to 20 000 tweets and step size varies from 4000 to 20 000. Since LexRank is sensitive to step size, we compare Mythos with LexRank at different step sizes. LexRank performance decreases with the increase in step size (stepsize $>=$ 12 000). TOWGS builds word graphs and combine the decay window techniques to handle the memory requirements. The low score for TOWGS is due to the algorithm's dependability on word graphs of bigrams and trigrams along with decaying window technique that decays graph weights, count of words, and word pair occurrences. The decay will be more where updation is less, which is affected by noise (huge number of retweets relating to the discussions and personnel opinions), making these words and pairs more relevant than the words related to the important subevents. For the Centroid-based summarization approach to work, we remove all the retweets and use word embedding size as 500 with all the suggested settings. However, this approach is not efficient in handling the sheer variation of topics present in the tweet stream. Sumblr approach creates noisy clusters as it is not able to remove noise present in the tweets and generates online summary that contains more noisy tweets than meaningful information resulting in low F-measure compared to Mythos. Feng *et al.* [31] StreamCube approach also verifies the same findings for Sumblr-generated summaries.

E. Storyline Generation and Summarization Model Output

Tables VIII and IX represent the generated storylines for SB and USE events, respectively. The subevents' text and timesstamps are taken from the summarization model output. The creation time of the cluster is referred as the prediction time of the event. The timings for SB and USE subevents have been confirmed with the Guardian summary and time provided with ground truth, respectively. For USE, the events/subevents in the storyline are exactly the same as the timings of the ground truth events (referred to as ground truth time). However, in SB, few events are identified not later than 4 min from the actual time (time provided by Guardian). The quality of sentences and abstraction obtained by Mythos mainly depend on the quality of tweets. The quality and abstraction are better in the case of well-structured sentences. If tweet is not a well-structured sentence, then the deep learning model outputs poor quality sentences. Refer Table X for some examples of

TABLE X
COMPARISON BETWEEN CLUSTER TEXT AND SUMMARIZATION MODEL OUTPUT

Tweet Text	Summarization Model Output
#FACUpFinal now underway! Drogba leading the firing power, Torres on the bench. John Terry will keep Suarez in check! Com'on Chelsea!!	***Big occasion and Drogba always shines ! 2-0 ! Come on CHELSEA***.
Stevie G goes down in the box after falling with Ivanovic but ref Dowd waves play on #CFCWembley #FACupFinal (SL)	Stevie G goes down in the box after falling with Ivanovic waves. ref Dowd waves play on CFCWembley FACupFinal (SL).
Didier Drogba becomes the first player to score in 4 FA Cup Finals! #CFCWembley #FACupFinal	Didier Drogba becomes the first player to score in 4 FA Cup finals.
Mikel goes into the book on 37 mins for late tackle on Stevie G. #CFCWembley #FACupFinal (SL)	Mikel goes into the book on 37 mins for late tackle on Stevie G.
Great mazy run by Kalou into the box but he gets ambushed by the Liverpool defence before he can shoot #CFCWembley #FACupFinal (SL)	he gets ambushed by the Liverpool defence before Kalou into the box but he can shoot Liverpool FACupFinal.
THE WINNER of #FACup 2012 is CHELSEA FC!!! WEMBLEY is BLUE! TODAY is a BLUE DAY! Proud of you, guys! :'] #CU	***THE WINNER of FACup 2012 is a BLUE DAY ! Proud of you , guys ! :']***

output sentences produced by the model for FA Cup event. In Table X, the first and last rows show abstraction performed by the model (in bold and italic), while the second last row is an example of poor quality sentence when tweet text is not well formed.

VI. CONCLUSION

We proposed a framework, Mythos, which presents small-scale to large events in the form of PCs and hierarchies, respectively. Its online clustering module is not only able to identify small-scale events but also fast happening subevents, which have been substantiated through our experimental analysis. Event hierarchy generator module of Mythos generates bigger events in the form of hierarchies. These hierarchies are generated by aggregating small-scale events that share temporal and semantic proximity. Our experiments show that the internal nodes of the hierarchies represent subevents of an event. The summarization module of Mythos produces abstractive summaries of the events at different levels of granularity from the most abstract to the most detailed level. We have evaluated the Mythos summaries with externally available summaries of Wikipedia and Guardian for both granularities. In all the cases, Mythos outperforms the baselines approaches. Mythos also generates storyline for an event using summaries at different levels. The generated storyline is very close to the actual storyline of the event. For future work, a separate deep learning module can be trained for weight-specific *n*-grams to give more weightage to tweets with substantial information and produce better sentences to combat high levels of noise in tweets. These high-quality tweets can be used with summarization model to generate better summary and timeline of events.

REFERENCES

[1] T. Sakaki, M. Okazaki, and Y. Matsuo, "Earthquake shakes Twitter users: Real-time event detection by social sensors," in *Proc. 19th Int. Conf. World Wide Web*, 2010, pp. 851–860.

[2] R. McCreadie *et al.*, "SUPER: Towards the use of social sensors for security assessments and proactive management of emergencies," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 1217–1220.

[3] W. He, S. Zha, and L. Li, "Social media competitive analysis and text mining: A case study in the pizza industry," *Int. J. Inf. Manage.*, vol. 33, no. 3, pp. 464–472, 2013.

[4] S. Phuvipadawat and T. Murata, "Detecting a multi-level content similarity from microblogs based on community structures and named entities," *J. Emerg. Technol. Web Intell.*, vol. 3, no. 1, pp. 11–19, 2011.

[5] M. Osborne, S. Petrović, and R. McCreadie, "Bieber no more: First story detection using Twitter and Wikipedia," in *Proc. SIGIR*, 2012, pp. 1–4.

[6] M. Adedoyin-Olowe, M. M. Gaber, C. M. Dancausa, F. Stahl, and J. B. Gomes, "A rule dynamics approach to event detection in twitter with its application to sports and politics," *Expert Syst. Appl.*, vol. 55, pp. 351–360, Aug. 2016.

[7] Y. Gu, Z. S. Qian, and F. Chen, "From Twitter to detector: Real-time traffic incident detection using social media data," *Transp. Res. C, Emerg. Technol.*, vol. 67, pp. 321–342, Jun. 2016.

[8] S. Petrović, M. Osborne, and V. Lavrenko, "Streaming first story detection with application to Twitter," in *Proc. Hum. Lang. Technol., Annu. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, Jun. 2010, pp. 181–189.

[9] S. Katragadda, S. Virani, R. Benton, and V. Raghavan, "Detection of event onset using Twitter," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 1539–1546.

[10] H. Becker, M. Naaman, and L. Gravano, "Beyond trending topics: Real-world event identification on twitter," in *Proc. 5th Int. AAAI Conf. Weblogs Social Media*, 2011, pp. 1–4.

[11] M. Walther and M. Kaisser, "Geo-spatial event detection in the Twitter stream," in *Advances in Information Retrieval* (Lecture Notes in Computer Science). Springer, 2013, pp. 356–367.

[12] K. Watanabe, M. Ochi, M. Okabe, and R. Onai, "Jasmine: A real-time local-event detection system based on geolocation information propagated to microblogs," in *Proc. 20th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2011, pp. 2541–2544.

[13] A. Boettcher and D. Lee, "Eventradar: A real-time local event detection scheme using twitter stream," in *Proc. IEEE Int. Conf. Green Comput. Commun.*, Nov. 2012, pp. 358–367.

[14] S. B. Ranneries, M. E. Kalør, S. A. Nielsen, L. N. Dalgaard, L. D. Christensen, and N. Kanhabua, "Wisdom of the local crowd: Detecting local events using social media data," in *Proc. 8th ACM Conf. Web Sci.*, 2016, pp. 352–354.

[15] H. Abdelhaq, C. Sengstock, and M. Gertz, "EvenTweet: Online localized event detection from Twitter," *Proc. VLDB Endowment*, vol. 6, no. 12, pp. 1326–1329, 2013.

[16] N. Alsaedi, P. Burnap, and O. Rana, "Can we predict a riot? Disruptive event detection using Twitter," *ACM Trans. Internet Technol.*, vol. 17, no. 2, pp. 1–26, 2017.

[17] D. Pohl, A. Bouchachia, and H. Hellwagner, "Automatic identification of crisis-related sub-events using clustering," in *Proc. 11th Int. Conf. Mach. Learn. Appl.*, vol. 2, Dec. 2012, pp. 333–338.

[18] C. Buntain, J. Lin, and J. Golbeck, "Discovering key moments in social media streams," in *Proc. 13th IEEE Annu. Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2016, pp. 366–374.

[19] P. Meladianos, G. Nikolentzos, F. Rousseau, Y. Stavrakas, and M. Vazirgiannis, "Degeneracy-based real-time sub-event detection in Twitter stream," in *Proc. 9th Int. AAAI Conf. Web Social Media*, 2015, pp. 248–257.

[20] W. Xie, F. Zhu, J. Jiang, E.-P. Lim, and K. Wang, "Topicsketch: Real-time bursty topic detection from Twitter," in *Proc. IEEE 13th Int. Conf. Data Mining*, Dec. 2013, vol. 28, no. 8, pp. 837–846.

[21] B. Sharifi, M. A. Hutton, and J. Kalita, "Automatic summarization of twitter topics," in *Proc. Nat. Workshop Design Anal. Algorithm*, Tezpur, India, 2010, pp. 121–128.

[22] Z. Wang, L. Shou, K. Chen, G. Chen, and S. Mehrotra, "On summarization and timeline generation for evolutionary Tweet streams," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 5, pp. 1301–1315, May 2015.

[23] C. C. Aggarwal and K. Subbian, "Event detection in social streams," in *Proc. SIAM Int. Conf. Data Mining*, 2012, pp. 624–635.

[24] J. Yin, A. Lampert, M. Cameron, B. Robinson, and R. Power, "Using social media to enhance emergency situation awareness," *IEEE Intell. Syst.*, vol. 27, no. 6, pp. 52–59, Nov./Dec. 2012.

[25] L. M. Aiello *et al.*, "Sensing trending topics in Twitter," *IEEE Trans. Multimeida*, vol. 15, no. 6, pp. 1268–1282, Oct. 2013.

[26] D. Wurzer, V. Lavrenko, and M. Osborne, "Twitter-scale new event detection via K-term hashing," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Sep. 2015, pp. 2584–2589.
[27] C. Comito, C. Pizzuti, N. Procopio, and V. P. Bucci, "Online clustering for topic detection in social data streams," in *Proc. IEEE 28th Int. Conf. Tools Artif. Intell. (ICTAI)*, Nov. 2016, pp. 362–369.
[28] C. C. Aggarwal and P. S. Yu, "On clustering massive text and categorical data streams," *Knowl. Inf. Syst.*, vol. 24, no. 2, pp. 171–196, 2010.
[29] F. Atefeh and W. Khreich, "A survey of techniques for event detection in Twitter," *Comput. Intell.*, vol. 31, no. 1, pp. 132–164, 2015.
[30] A. Marcus, M. S. Bernstein, O. Badar, D. R. Karger, S. Madden, and R. C. Miller, "Twitinfo: Aggregating and visualizing microblogs for event exploration," in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, May 2011, pp. 227–236.
[31] W. Feng *et al.*, "STREAMCUBE: hierarchical spatio-temporal hashtag clustering for event exploration over the Twitter stream," in *Proc. IEEE 31st Int. Conf. Data Eng.*, Apr. 2015, pp. 1561–1572.
[32] D. R. Radev, "Lexrank: Graph-based lexical centrality as salience in text summarization," *J. Artif. Intell. Res.*, vol. 22, pp. 457–479, Dec. 2004.
[33] D. Chakrabarti and K. Punera, "Event summarization using tweets," in *Proc. 5th Int. AAAI Conf. Weblogs Social Media*, 2011, pp. 66–73.
[34] D. Inouye and J. K. Kalita, "Comparing Twitter summarization algorithms for multiple post summaries," in *Proc. IEEE 3rd Int. Conf. Privacy, Secur., Risk Trust IEEE 3rd Int. Conf. Social Comput.*, Oct. 2011, pp. 298–306.
[35] J. Nichols, J. Mahmud, and C. Drews, "Summarizing sporting events using Twitter," in *Proc. ACM Int. Conf. Intell. Interfaces*, 2012, pp. 189–198.
[36] A. Olariu, "Efficient online summarization of microblogging streams," in *Proc. 14th Conf. Eur. Chapter Assoc. Comput. Linguistics*, 2014, pp. 236–240.
[37] G. Rossiello, P. Basile, and G. Semeraro, "Centroid-based text summarization through compositionality of word embeddings," in *Proc. MultiLing*, 2017, pp. 12–21.
[38] N. A. Diakopoulos and D. A. Shamma, "Characterizing debate performance via aggregated twitter sentiment," in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, Apr. 2010, pp. 1195–1198.
[39] W. X. Zhao, Y. Guo, R. Yan, Y. He, and X. Li, "Timeline generation with social attention," in *Proc. 36th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Aug. 2013, pp. 1061–1064.
[40] O. Alonso, S.-E. Tremblay, and F. Diaz, "Automatic generation of event timelines from social data," in *Proc. ACM Web Sci. Conf.*, 2017, pp. 207–211.
[41] S. Wang *et al.*, "StoryLine: Unsupervised geo-event demultiplexing in social spaces without location information," in *Proc. 2nd Int. Conf. Internet–Things Design Implement.*, 2017, pp. 83–93.
[42] R. Nallapati, B. Zhou, C. N. dos Santos, C. Gulcehre, and B. Xiang, "Abstractive text summarization using sequence-to-sequence RNNs and beyond," 2016, *arXiv:1602.06023*. [Online]. Available: https://arxiv.org/abs/1602.06023
[43] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," 2017, *arXiv:1704.04368*. [Online]. Available: https://arxiv.org/abs/1704.04368
[44] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*. [Online]. Available: https://arxiv.org/abs/1409.0473
[45] *SuperBowl 2018-Wikipedia*. Accessed: Dec. 7, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Super_Bowl_LII
[46] *FACup 2012-Guardian*. Accessed: Dec. 7, 2019. [Online]. Available: https://www.theguardian.com/football/2012/may/05/fa-cup-final-chelsea-liverpool-live
[47] *SuperTuesday-Guardian*. Accessed: Dec. 7, 2019. [Online]. Available: https://www.theguardian.com/world/2012/mar/06/super-tuesday-results-live
[48] *US Election-Guardian*. Accessed: May 2019. [Online]. Available: https://www.theguardian.com/world/2012/nov/06/us-election-2012-results-live-blog
[49] *SuperBowl 2018-Guardian*. Accessed: Dec. 7, 2019. [Online]. Available: https://www.theguardian.com/sport/2018/feb/04/super-bowl-2018-philadelp hia-eagles-new-england-patriots-nfl
[50] *FACup 2012-Wikipedia*. Accessed: May 2019. [Online]. Available: https://en.wikipedia.org/wiki/2012_FA_Cup_Final
[51] P.-N. Tan, S. Steinbach, A. Karpatne, and V. Kumar, *Introduction to Data Mining*, 2nd ed. London, U.K.: Pearson, 2018.
[52] *CNNn DailyMail Dataset*. Accessed: May 2019. [Online]. Available: https://cs.nyu.edu/~kcho/DMQA/