What just happened? A Framework for Social Event Detection and Contextualisation

Prashant Khare
Insight Centre for Data Analytics
NUI Galway, Ireland
prashant.khare@insight-centre.org

Pablo Torres
Insight Centre for Data Analytics
NUI Galway, Ireland
pablo.torres@insight-centre.org

Bahareh R. Heravi
Insight Centre for Data Analytics
NUI Galway, Ireland
bahareh.heravi@insight-centre.org

Abstract

In course of a breaking news event, such as natural calamity, political uproar etc., a massive crowd sourced data is generated over social media which makes social media platforms an important source of information in such scenarios. The value of the information being propagated via social media is being increasingly realised by the news organisations and the journalists. Better tools and methodologies are needed to facilitate them in utilising this information for news production. A lot of analysis over social media, by the journalists, is performed via rigorous manual labour. However, the sheer volume of the data produced on social media is overwhelming and acts as a major obstacle for manual inspection of the streaming data for finding, aggregating and contextualising the emerging event in a short time span. This is a day-to-day challenge for journalists and media organisations. This paper addresses the above problem for journalist in handling the voluminous social media data, viewing it from an information retrieval perspective, by proposing an 'event detection and contextualisation' framework that processes an input stream of social media data into the clusters of likely events.

1 Introduction

Social media platforms have become a prominent medium for sharing information in real-time and have gradually evolved to more than just being a user-to-user interaction channel. Now even the normal citizens share and broadcast the real life 'events', and not just the professional journalists. This phenomenon has turned the former consumer (only) of the news into (also) a broadcaster of the news, and thereby turning the social media platforms into an invaluable source of newsworthy information. The news organisations, these days, monitor and harvest the user generated content (UGC) shared on the social media to gather real-time news, images and videos. This process, however, is laborious and time consuming and requires a considerable amount of manual effort. It is crucial for journalists and news organisations to monitor and filter the streams of data for potential newsworthy content and then to analyse, aggregate, contextualise and verify them in a timely manner.

The concept of Social Semantic Journalism is introduced by Heravi et al. [5], and addresses the above problems encountered by the media organisations. The Social Semantic Journalism framework [6] provides an integrated view for enhancing newsworthy information discovery, filtering, aggregation, verification and publication by utilising social and semantic web technologies.

Building upon the ideas of Social Semantic Journalism, to aid journalists in utilising UGC in an efficient manner, this paper proposes a framework that implements an event detection pipeline, which *clusters the data into different events,* and *determines the context of the events based on entities* (mentions particular to any person, place, event, or thing) *related to the events*. Information flows on the social media often via textual medium, and hence in the proposed framework, we leverage text mining and Natural Language Processing (NLP) technologies to extract the information. This paper elaborates the work in more detail and extends its earlier version [11] by testing the framework at a primitive level with experiments.

The remainder of this paper is organised as follows. Section 2 provides a background to the problem and briefly reviews related work. Section 3 presents our proposed Event Detection and Contextualisation framework and gives a detailed overview of its components and phases. Section 4

concludes the paper and discusses directions for future research.

2 Background and Related work

Identifying new events from data, within the context of news reporting, has been an area of interest for researchers for a long time. An important line of work in this area is Topic Detection and Tracking (TDT) [1], which focuses on breaking down an incoming streaming newswire text into smaller cohesive pieces of news and determining if any new piece has been earlier reported or not. An event detection cycle has been categorised as a subtask within TDT [1]. Millions of social media users share information about various events through social media platforms. The voluminous and streaming nature of social media data warrants the usage of streaming algorithm models, where the streaming data is in a chronological order [16] and this data is processed in a bounded space and time, i.e. as every entry arrives it gets processed. Traditional approaches for identifying new information (an event) were to compare each new entry in the data with the previously arrived entries. Petrovic et al. [19] applied clustering mechanism to determine nearest neighbours in the text data and thereby identified the tweets that were first to report the occurrence of an event. This work motivated many other contemporary researches to progress in a related direction.

Osborne et al. [17] used Petrovic et al. [19] approach as a baseline model and investigated the ways with which the event detection on Twitter data can be improved. They matched the frequency of newly occurring events from tweets with the activity of the corresponding pages of entities from Wikipedia and analysed if there was a similar pattern while determining an event. Frequency based analysis encouraged Parikh and Karlapalem [18] to develop an event detection system, that extracts events from tweets by examining frequencies in the temporal blocks of streaming data.

NLP techniques can be leveraged in detecting the emerging events from voluminous data from social media platforms. Events are associated with entities and NLP techniques can be applied to extract the entities mentioned in the text. Ritter et al. [20] redeveloped the taggers and segmenters of Stanford NLP[1] library [4] to perform named entity recognition (NER) from tweets. Ritter et al. [21] extended the above work in creating an application *Twical,* to extract an open domain calendar for events shared on Twitter.

It is also critical to determine the context of a piece of text/information in an event detection system. There is often a question asked, " What is this about?" and contextualisation is about answering this question. One of the suggested ways to answer such a problem is by aggregating information from knowledge base such as Wikipedia [22]. Context can likely be inferred by extracting set of topics that bound the text. Hulpus et al. [10] proposes an approach that automatically extracted the topic labels from the corpus by linking the topics inherent to a text with the concepts in DBpedia[2]. With respect to the context of a microblog post, Meij et al. [13] derived underlying concepts from a large knowledge base of Wikipedia articles by applying a supervised learning using a Naive Bayes (NB), Support Vector Machines (SVM), and a C4.5 decision tree classifier. Hoffart et al. [9] used a large knowledge base 'YAGO'[3] to explore the inherent relationship between extracted entities and disambiguate them to derive the context. Taking insights from various research works briefed above, we propose a framework that incorporates ideas from different works in the next section.

3 The Framework for Event Detection & Contextualisation

While there are diverse approaches to extract information from data, in form of clustering, entity extraction and contextualisation, there is no proposed pipeline that utilises different methods and brings them under one framework to generate insights from social streaming data. We aim to address this gap, by proposing a framework that performs the aforesaid functionalities under one system. A complete illustration of the framework is visualised in Figure 1. The pipeline of the framework incorporates several components, each followed by another phase that uses the output from the previous one. The data could potentially come from various social media APIs; however, in our test analysis we relied on sample collected from Twitter streaming API[4].

The following explains the phases of our proposed framework in details.

[1] http://nlp.stanford.edu/software/corenlp.shtml

[2] http://dbpedia.org/About

[3] http://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/

[4] https://dev.twitter.com/docs/api/streaming

3.1 Indexing and Clustering

The purpose of this phase is to pre-process the data, break it into set of keywords and generate an index that maps words against their corresponding document. Once indexed, this data is clustered as sets of word vectors occurring together prominently. These clusters tend to represent the events being discussed in the data.

Indexing

In this sub-phase the data is indexed. The incoming data stream is initially stored and then the data is divided in slabs of time windows (say of 10 minutes each). This is done to analyse the data based on regular time intervals, which may result in inferring only the highly dominating events/clusters present in the data. Before generating the index, the data is pre-processed where the stop words are removed and take off the non-lexical characters. An index [12], between terms and corresponding documents (that initially contained those terms), is generated for this slab of data using standard libraries such as Lucene and Solr (built over Lucene).

An index is, in short, a data structure that is built after the text is parsed and is used to answer the search queries. The index is a mapping done between the terms occurring in the text of a document and the document itself. A basic explanation of index is given in the following with an example.

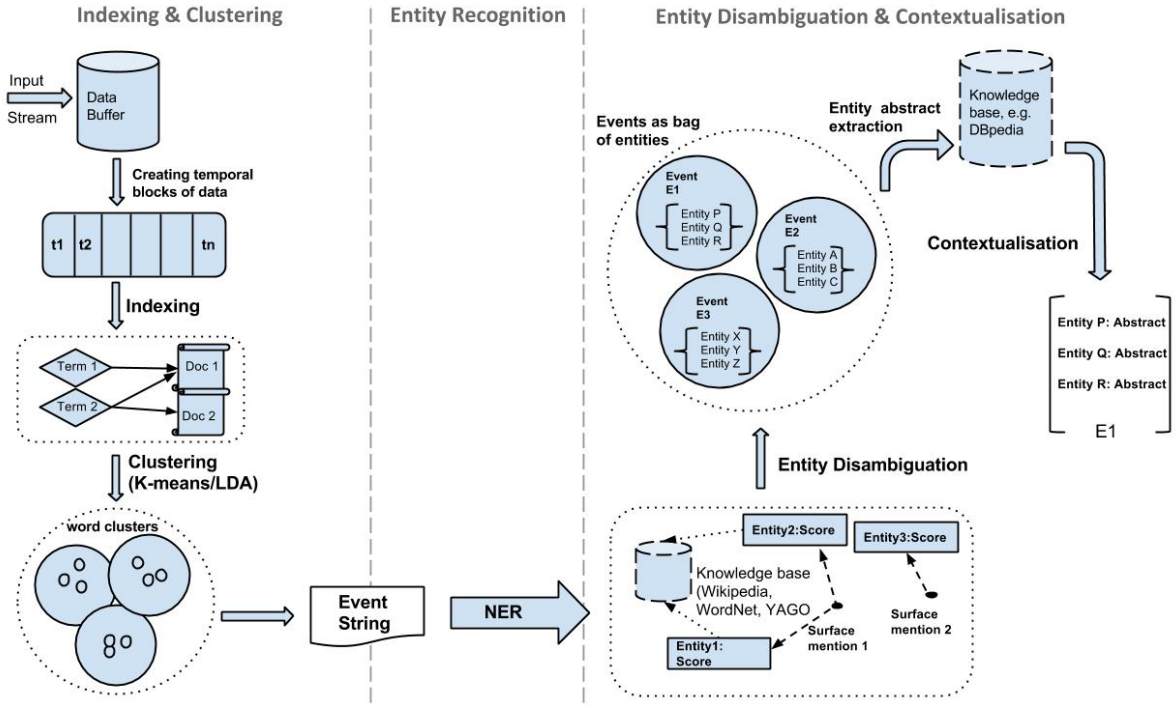

Figure 1. Event detection and Contextualisation framework

Assume that there is a corpus of documents and each document contains text in form of sentences, paragraphs etc. Let us consider the following documents:

- Document 1: Digital Humanities & Journalism and tradition Journalism
- Document 2: Social Semantic Journalism
- Document 3: Digital Repository Digital Humanities

The posting list, which contains list of documents a term occurred in, of the term 'Digital' would be the list (1,3), that means that the term 'Digital' occurs in document with IDs 1 and 3. Likewise for the term 'Journalism' the posting list would be (1,2). This information can be further extended with details such as number of times a term occurs in a document (*Term Frequency* also abbreviate as 'tf' [12]), and the number of documents a term occurs in (*Document frequency*, the inverse of which is called *Inverse document frequency* 'idf' [12]), and may be also the position of a term in a document. This document

corpus is initially parsed which removes the stop words, lower cases the documents and generate tokens (they can be generated by applying several rules such as space separated, non-alphanumeric character separated or even more rules). After parsing the documents a posting list of all the terms is created which contains the document IDs and the positioning of the terms in each document.

The posting list for the term 'Digital' would be ((1, (0)), (3, (0, 2))), which means that the term 'Digital' occurs in document with ID 1 at 0th position and in document with ID 3 at 0th and 2nd position. Similarly for the term 'Journalism' the posting list would be ((1, (3, 6)), (2, (2))). So, ideally a posting list is created for each document and later on merged with others hence a basic index may look something like this

{*'Digital'*: ((1, (0)), (3, (0,2))), *'Journalism'*: ((1, (3, 6)), (2, (2))), '': ... and so on}

This is an index (as in Figure 2) and can be saved at some place in a format more easily processed for querying such as: *term* | documentID*x* : position1, position2; documentID*y* : position1, position2;...and so on

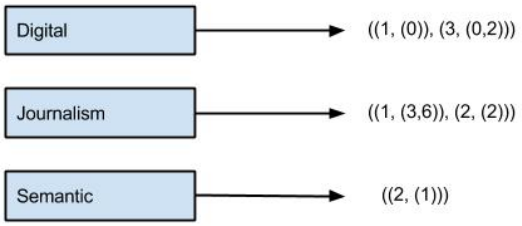

Figure 2. Keyword-document mapping

Earlier in the paper there was a mention of *tf* and *idf*, the term frequency and inverse document frequency (tf-idf) is a scheme to assign weights to each term occurring in a document based on the term frequency of the term and inverse document frequency in the whole data index. The term that carries a higher tf-idf weight is considered to be important for that document. Let's look at how *tf* 'term frequency' is calculated.

$$tf_{t,d} = N_{t,d}$$

A vector can be used to represent a document, for instance 'Digital Humanities & Journalism and traditional journalism' can be represented as (2 1 1 1) corresponding to 'journalism' (occurring two times), 'humanities', 'digital', and 'traditional'. In fact the complete data corpus can be represented as a vector matrix of *K* unique words where each document within the corpus is a k-dimensional vector. Such type of representation of documents is also termed as *vector space model*[5]. Now, if the term frequency is just weighed based on number of occurrences the longer documents will have a dominance, which ideally should not be the case as a shorter length document can be equally important with respect to a term. Therefore, to remedy such an effect of the length of the document the *term frequency* is normalised by the *Euclidean norm*[6] of the document.

$$tf_{t,d} = N_{t,d} / \|D\|$$

$\|D\|$ is the Euclidean norm of the document and is evaluated by squaring each value in a document vector, summing them together and then doing its square root [12].

The *inverse document frequency* (idf) is the inverse of the number of document a term appears in a corpus divided by the total number of documents. Why is it important? Consider if only the term frequency is used as a weighing factor, tf evaluates all the terms on same scale and if some terms occur more rarely in the documents despite being unique or more discriminative than rest of the terms they will have a lower *tf* score and hence be ranked lower while searching for a particular query/phrase. Consider a phrase 'Semantic Journalism', the expected search should have results pertaining to *semantic journalism* and not just about *journalism*, but if only term frequencies are taken into consideration then its highly likely that the term *journalism* will have a dominating value over a large data corpus (like that of Wikipedia), hence to negate this effect an inverse document frequency is determined. In a corpus of N documents the inverse document frequency of a term *t* is given as follows

$$id_{t,f} = N/N_t$$

where N_t is the number of documents term *t* occurs in. Further in order to scale this value a logarithm of it is taken. After having both the *tf* and *idf* values the *tf-idf* weight is calculated of a term *t* in a document *d* is calculated by multiplying both the values [12].

$$tf\text{-}idf_{t,d} = tf_{t,d}\, idf_t$$

[5] http://en.wikipedia.org/wiki/Vector_space_model

[6] http://en.wikipedia.org/wiki/Euclidean_norm#Euclidean_norm

As earlier stated, a document will be a k-dimensional vector. Ideally a document vector would contain the k terms that occur in it and not all the terms of the corpus. So, as there is a vector of one document, likewise vectors of all the documents in the corpus shall be created and a matrix of those vectors shall be stored at a place, which will represent the *index* of the data.

Clustering:

The next stage is to derive the preliminary clusters of the data, which are likely to assimilate the most related content within the data slab that was earlier created. Some of the clustering algorithms that can be hired to cluster the data are k-means[7], pLSA [8] and LDA [2].

Once the clusters are formed, the prominent occurring terms of the clusters can be used to query the index to retrieve the top relevant documents according to a particular threshold relevance score. The retrieval of the documents is based on the relevance score derived from term frequency and inverse document frequency (*tf-idf*) [12] value. The text from those top scored documents can now be extracted and merged into one string, hereafter called *event string*, which tend to represent the information stored against a particular cluster or event. This *event string* is further used for NER and disambiguation.

3.2 Entity recognition

This phase is required to retrieve more information about a given cluster. The key terms generated in the clusters are not the only existing piece of information for a cluster i.e. key terms in a cluster do not entirely reflect the context of a cluster as they are based on frequency of their occurrence in a given data sample. Hence, it is important to also retrieve the other key terms mentioned or co-occurring along with the terms occurring in the clusters, which might have very low score in the cluster formation and thereby getting ignored. The extra key terms retrieved from NER process will assist in determining the context much better.

The *event string* is annotated with entities by applying NER techniques. NER is an information extraction task to extract key elements, hereafter referred to as *Entities,* from a text and categorise them into person, location and organisation There is already a considerable amount of research been done (and continuously improving) on recognising the named entities in a natural language text. We will rely on libraries such as Stanford NER [4], which implements a linear chain Conditional Random Field (CRF) [4], or other wrappers to this library, which implement it to extract named entities. However, there are more libraries, for instance, Open NLP[8], Open Calais[9] etc. NER models are well explained in the research work by Sang and Meulder [24], and Finkel et al. [4].

A given text is tagged with Parts of Speech (POS) as they occur in it. The tagged terms could be nouns, verbs, predicates etc. The purpose of performing such an operation on the data is to identify and extract the potential key terms from the text which are likely to be the participating named entities (which are identified as POS tagged phrases after NER operation) with respect to the context of the information given in the text. NER operation extracts named entities (also to be further referred to as 'mention' and 'surface-mention') strictly from the data. However, these mentions would need to convey more information about themselves, as it still does not convey precisely which entities are the being referred to in terms of a global web of all the possible entities that have a certain identity and information linked with them. Hence, NER phase extracts the potential mentions of the named entities from the text and that is further analysed in disambiguation phase to map those mentions to most relevant entity that has an identity and some knowledge (in the knowledge base).

3.3 Entity disambiguation and Contextualisation

There can be multiple candidate entities (a candidate entity is a potential entity, from a knowledge source, mapped for a mention) for each mention (entities annotated post NER phase) in the *event string*. Those entities need to be further disambiguated. It can be explained with an example, such as in "David was playing for Manchester United when Victoria gave her auditions. Victoria later became part of band Spice Girls", how can we say that Victoria refers to a person (particularly Victoria Beckham) and not 'Victoria'- a place or Queen Victoria, and David refers to David Beckham and not David - a figure in religious text/history. To establish such a mapping is termed as *named entity disambiguation* process.

Post the NER phase an input text (web page, language paragraph, sentence, article) is resolved into various mentions of entities (surface forms- that

[7] http://en.wikipedia.org/wiki/K-means_clustering

[8] https://opennlp.apache.org/

[9] http://www.opencalais.com/

means its just a mention with no associated knowledge i.e. it has not been linked to an entity, yet, with a definite identification). For each mention, knowledge sources such as DBpedia and/or Yago [9, 7] are harvested to extract potential entity mentions. Each mention would then be mapped for numerous potential entity candidates, for example 'Paris' can have potential candidates as Paris (city), Paris Hilton (person), University of Paris etc. After getting the candidate entities, a relevance score can be assigned to each based on features such as *a prior for candidate entity popularity* (that means how popular a particular entity is, which can be determined by considering the number of redirects to a particular entity on Wikipedia for instance), *mutual information* (number of keywords from a keyphrase occurring in the description of an entity thereby determining a similarity between keyphrase and an entity) a related aspect of *context-article similarity* [3], *syntax based similarity* [23] - learn about the sort of entities that occur as subject to a particular predicate and vice versa, for example if there is a phrase 'Paul plays exceptional guitar' then what sort of entities occur as a subject to the verb 'play' thereby ranking the potential entities based on similarity score of syntax of phrase and description, *Keyphraseness*- the probability of a term being selected as a keyword against it overall occurrences in the corpus [14], *entity-entity coherence* (quantifying the number of similar incoming links on a knowledge base as Wikipedia). Milne and Witten [15] extended few similarity measures defined by Bunescu and Pasca [3], which compared the context of a given text to the entities mention in Wikipedia and they also capitalised on the valuable information gathered from overlapping incoming links in Wikipedia, thereby determining a semantic relatedness between a *mention's* potential candidate entities and mentions in the textual content.

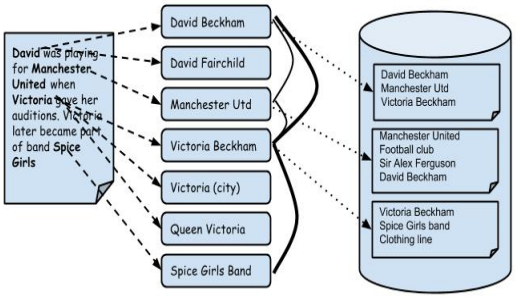

Figure 3. Mention-Entity mapping

Built on the above features, a graph of mentions and candidate entities, edges as weights, could be created, as shown in Figure 3. A greedy approach, by iteratively removing the low weight nodes can be adopted to filter out the candidates to disambiguate the entities [7]. This will generate disambiguated entities (to a high degree) corresponding to the surface mentions of the input text and represent entities as per the context of the input. Once the disambiguated entities are generated, the knowledge resource can be used to generate a brief description about the prominent entities (such as their *abstract/description* and type), and for contextualising the complete input phrase as a bag of entities and their description.

4 Preliminary Experiments

Two main questions under investigation in this paper are: 1) Do we get a correct/approximate formation of cluster, i.e. events (whether or not the key terms associated to a cluster subscribe to a common event)? 2) Whether the system is able to resolve those key terms to proper entities that have additional information about them from a knowledge base? An implementation of our proposed framework, the methods and tools used and applied to get these first preliminary results are discussed in the following.

4.1 Dataset

For test purposes we created a sample data of 2150 tweets, which constituted of tweets from three known events. The tweets for those known events were earlier collected, when they happened live, via Twitter streaming API. Three events, which constituted the sample data, were Iran Election 2013, Japan Earthquake of 7.3 magnitude in 2013, and a football match between Manchester United Football club and Liverpool Football club played in September 2013. The keywords used to collect tweets for those events were,

- Iran Election 2013 - '*Iran'*, '*IranElection'*, '*Iran Election'*
- Japan Earthquake of 7.3 magnitude in 2013 - *'Japan'*, *'JapanEarthquake'*, *'JapanTsunami'*, '*Earthquake'*
- Manchester United Football club Vs Liverpool Football club- *'mfc'*, *'lfc'*, *'mfcVSlfc'*, *'manu'*, *'manchesterutd'*, *'liverpoolfc'*

The tweets were randomly chosen from all the three different event datasets and collected in a single dataset.

4.2 Tools/Libraries

For indexing the data we chose Apache Solr (built on Apache Lucene). The library tokenises the data and generates an index. Further, the data was clustered using the *k-means* algorithm, which initialises by selecting '*k*' number of cluster seeds. Each object (in the Euclidean space) has a value which would be used in calculating the distance between the mean of the objects and the mean of the cluster. If there are a set of objects (observation) in a space (x_1, x_2, x_3,... x_n) and each object is a *n-dimensional vector*, then each observation is matched for its minimum distance against the center of a cluster. The function '*j*' of minimising the distance is [25]:

$$j = arg\ min_i\ (\|x_j - u_j^t\|^2)$$

where $\|x_j - u_j^t\|^2$ is the distance between the observation point (object) and the cluster center. The object is assigned to the cluster if the distance between the two is shorter than distance between the object and center of any other cluster. The moment an object is included in a cluster the value the centre of the cluster is recalculated and a mean of all the observations of the cluster (including the newest entry) is calculated and a new value of the centre of the cluster is generated; the same process is repeated for all the objects in the set. After an iteration is performed over all the observations, each observation is assigned to a cluster which has its center nearest to the observation. The process is repeated for some iterations (this value is given as an input while running *k-means* algorithm), to minimise the distance between the centroid of a cluster and the associated observations. In other words, the process needs to be repeated to stabilise the position of the centroid..

We chose Apache Mahout as the library which implements *k-means* clustering algorithm. Our aim here was to analyse how the *k-means* would perform if we were to generate exactly the same number of clusters as the number of known events, hence we initiated the algorithm by choosing 3 cluster seeds from the data.

Once the clusters were formed, we created the *event string* (by querying index using key terms from the clusters) for each cluster and gave them as input to entity recognition phase (to recognise potential named entities which play a role in creating the context of the information) and entity disambiguation phase of the framework. We chose the AIDA framework [7], to analyse the outcome of the disambiguated entities. AIDA is an end-to-end framework built in Java, which performs NER and entity disambiguation for a given input text. AIDA uses YAGO entity base as the knowledge base to match the candidate entities. The output of the AIDA framework are the resulting disambiguated entities of the input text that are most likely to establish the precise entities that are being mentioned in the text. This library implements many of the disambiguation parameters as discussed in section 3.3. AIDA framework took care of both the phases *viz. entity recognition* and *entity disambiguation*.

4.3 Results

The dataset of tweets, described in section 4.1, was given as an input to the pipeline that executed *k-means* algorithm and disambiguated the entities after entity recognition through AIDA framework. The three clusters that were formed post clustering were as follows,

- *Cluster 1-* '*lfc*', '*mufc*', '*come*', '*manutd*', '*liverpool*', '*man*', '*united*'
- *Cluster 2-* '*iran*', '*t.co*', '*election*', '*iran's*', '*rt*', '*nuclear*', '*voters*', '*US*', '*program*', '*unites*'
- *Cluster 3-* '*tsunami*', '*fukushima*', '*http*', '*7.3*', '*earthquake*', '*japan*', '*alerta*', '*japón*', '*magnitude*', '*terremoto*', '*quake*'

The above terms from each cluster were used to query the index and create *event string.* As earlier discussed, this string is generated from top relevant documents with respect to the key terms collected from a cluster. The top relevant documents are collected based on an assumption that they are likely to assimilate more and more information about a concerned cluster (hence an event). However, so as to reduce the redundancy in the *event string* and maximise the diversity of the collected tweets (since many a times there are quite a lot of *re-tweets* present in a streaming tweet data) we also perform a cosine similarity of the text of a document against the text of other collected documents in this retrieval phase and discard those which are found with a similarity score, based on experimental observation, of more than 0.8. The *event string* was further processed using AIDA framework and the outcome of the entire process were the Wikipedia links for the following entities for each cluster.

- *Cluster* 1- Wilfried Zaha, David Moyes, Andre Marriner, Andres Lindegaard, Anderson Luis de Abreu Oliveira. *No related links found for Manchester United Football Club or Liverpool Club with high scores*
- *Cluster* 2- Iran, Iranian peoples, Persian Language, Mahmoud Ahmadinejad, Hezbollah
- *Cluster* 3- Japan, Fukushima Perfecture, Honshu, Hawaii, United States Geological Survey, Jesus

4.4 Discussion

We performed a basic run of the framework so as to assess the viability of the underlying processes and framework as a whole. The preliminary experiment helped in understanding the essential improvements and challenges with the current proposed framework. The output of the pipeline of this framework has certain challenges to address in future, however the initial test run on the sample data gives encouraging signs. The results for two clusters out of three (as we explicitly created three clusters) were positive with respect to the context of two known events *viz. Cluster 2 for Iran Election* and *Cluster 3 for Japan earthquake.* The third cluster, *Cluster 1*, while it was discovered in the clustering phase and was formed of relevant key terms (as assessed manually), did not yield any strong score entities post disambiguation process for Manchester United Football club or Liverpool Football club despite several mentions of *#MUFC* and *#LFC*; A possible reason for this could be that YAGO knowledge base (that was used for entity disambiguation by AIDA) did not contain mapping for entities recognised for that particular cluster with the surface mentions such as '*lfc'*, '*mufc'*, '*liverpool'*. '*lfc'*, '*mufc'* are the Twitter slangs and not the kind of mentions that might exist as an entity in a knowledge base like Wikipedia or DBpedia which forms the base of YAGO entity base. This is one of the challenges that is to be explored in future research that how a certain slang (abbreviated form) for a known entity could be related to its actual match.

An issue, which we aim to address in our future work, is to have an optimal solution for generating the number of clusters of events actually occurring in a live streaming data. In the current methodology, an initial parameter has to be given as an input so as to create the number of clusters (this is same requirement for LDA, *k-means*, and PLSA). A possible heuristic, that needs to be tested, is to initialise the algorithm with a *high* seed value (in a data slab of short time interval, there are less chances of a very high number of events- a heuristic) and later assess the similarity between different clusters and bring the number of clusters down by merging highly similar clusters.

In this section we made an attempt to analyse an early phase performance of the proposed framework. However, this is not a standard statistical evaluation of the entire pipeline incorporated since we lacked a gold standard data. But, so as to assess the viability of the underlying processes and the framework as a whole we performed some early level tests on a sample data collected. This work, however, takes us forwards towards generating insights from UGC with a journalistic approach and lays an encouraging foundation of an end-to-end framework by incorporating various research milestones that are already achieved by contemporary researchers, within a single pipeline.

5 Conclusion and Future Work

This paper presents a framework, which has the potential to assist journalists in dealing with the ever-flooding UGC. A preliminary test of the framework has been performed that looks encouraging with respect to the purpose of creating such a framework. An end-to-end statistical evaluation is yet to be performed to analyse the results of every phase i.e evaluating the correct number of clusters (that also involves taking our research forward to optimise clustering formation and cluster matching), key terms of any cluster, and the efficiency of the contextualisation mechanism.

The framework incorporates some of the state of the art techniques in the overall pipeline. The various phases of the framework have often been utilised in different domains but with a journalistic viewpoint it is novel to bring and knit them together to assess the extent to which it can aid the journalism processes. The aim of the framework is to come up with a tool that can simplify the rigorous processes the journalists practice while monitoring social media platforms to explore emerging events. This framework can aid in visualising the emerging events without needing to manually analyse various technical attributes of a piece of information that is propagating on the social media, so as to term it as an event and it can also help in generating the context of the information at the same time.

As briefed in previous section a few issues that are to be explored in future research are identifying the slang (or abbreviated forms) and checking for a possibility of mapping them to relevant entities from wider knowledge bases, and improvising with already

existing clustering mechanisms to help in getting closer to determining correct number of clusters. There are more foreseen challenges such as noise filtering from exhaustive social media streams, the non-lexical nature of the data, and the verity of the content. As a future work we aim to address these challenges by exploring how information extraction techniques can be customised for syntactically and lexically inefficient data and thereby refine the information gathering processes for journalists.

Acknowledgement

This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant Numbers 12/TIDA/I2389, SFI/12/RC/2289 and 13/TIDA/I2743.

References

[1] Allan, J. (2002). Introduction to topic detection and tracking. In Topic detection and tracking (pp. 1-16). Springer US.

[2] Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. the Journal of machine Learning research, 3, 993-1022.

[3] Bunescu, R. C., & Pasca, M. (2006, April). Using Encyclopedic Knowledge for Named entity Disambiguation. In EACL (Vol. 6, pp. 9-16).

[4] Finkel, J. R., Grenager, T., & Manning, C. (2005, June). Incorporating non-local information into information extraction systems by gibbs sampling. In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (pp. 363-370). Association for Computational Linguistics.

[5] Heravi, B. R., Boran, M., & Breslin, J. (2012, May). Towards Social Semantic Journalism. In Sixth International AAAI Conference on Weblogs and Social Media.

[6] Heravi, B. R., & McGinnis, J. (2013). A Framework for Social Semantic Journalism. In First International IFIP Working Conference on Value-Driven Social & Semantic Collective Intelligence (VaSCo), at ACM Web Science.

[7] Hoffart, J., Yosef, M. A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., ... & Weikum, G. (2011, July). Robust disambiguation of named entities in text. InProceedings of the Conference on Empirical Methods in Natural Language Processing (pp. 782-792). Association for Computational Linguistics.

[8] Hofmann, T. (1999, July). Probabilistic latent semantic analysis. InProceedings of the Fifteenth conference on Uncertainty in artificial intelligence(pp. 289-296). Morgan Kaufmann Publishers Inc.

[9] Hoffart, J., Suchanek, F. M., Berberich, K., & Weikum, G. (2013). YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. Artificial Intelligence, 194, 28-61.

[10] Hulpus, I., Hayes, C., Karnstedt, M., & Greene, D. (2013, February). Unsupervised graph-based topic labelling using DBpedia. In Proceedings of the sixth ACM international conference on Web search and data mining (pp. 465-474). ACM.

[11] Khare, P., & Heravi, B. R. (2014). Towards Social Event Detection and Contextualisation for Journalists. COLING 2014, 54.

[12] Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to information retrieval (Vol. 1, p. 6). Cambridge: Cambridge university press.

[13] Meij, E., Weerkamp, W., & de Rijke, M. (2012, February). Adding semantics to microblog posts. In Proceedings of the fifth ACM international conference on Web search and data mining (pp. 563-572). ACM.

[14] Mihalcea, R., & Csomai, A. (2007, November). Wikify!: linking documents to encyclopedic knowledge. In Proceedings of the sixteenth ACM conference on Conference on information and knowledge management (pp. 233-242). ACM.

[15] Milne, D., & Witten, I. H. (2008, October). Learning to link with wikipedia. In Proceedings of the 17th ACM conference on Information and knowledge management (pp. 509-518). ACM.

[16] Muthukrishnan, S. (2005). Data streams: Algorithms and applications. Now Publishers Inc.

[17] Osborne, M., Petrovic, S., McCreadie, R., Macdonald, C., & Ounis, I. (2012, August). Bieber no more: First story detection using Twitter and Wikipedia. InProceedings of the Workshop on Time-aware Information Access. TAIA (Vol. 12).

[18] Parikh, R., & Karlapalem, K. (2013, May). Et: events from tweets. InProceedings of the 22nd international conference on World Wide Web companion (pp. 613-620). International World Wide Web Conferences Steering Committee.

[19] Petrović, S., Osborne, M., & Lavrenko, V. (2010, June). Streaming first story detection with application to twitter. In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the

Association for Computational Linguistics (pp. 181-189). Association for Computational Linguistics.

[20] Ritter, A., Clark, S., & Etzioni, O. (2011, July). Named entity recognition in tweets: an experimental study. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (pp. 1524-1534). Association for Computational Linguistics.

[21] Ritter, A., Etzioni, O., & Clark, S. (2012, August). Open domain event extraction from twitter. In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 1104-1112). ACM.

[22] SanJuan, E., Moriceau, V., Tannier, X., Bellot, P., & Mothe, J. (2012). Overview of the INEX 2012 tweet contextualization track. Initiative for XML Retrieval INEX, 148.

[23] Thater, S., Fürstenau, H., & Pinkal, M. (2010, July). Contextualizing semantic representations using syntactically enriched vector models. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (pp. 948-957). Association for Computational Linguistics.

[24] Tjong Kim Sang, E. F., & De Meulder, F. (2003, May). Introduction to the CoNLL-2003 shared task: Language independent named entity recognition. InProceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4 (pp. 142-147). Association for Computational Linguistics.

[25] Zaki, M. J., & Meira Jr, W. (2011). Fundamentals of data mining algorithms.