WEB SCIENCE
COURSE WORK 3
Matriculation Number - 2576183S

1. Design an algorithm to detect events from such groups.

1. **What role entities play? How would you reduce the cost of detecting entities?**
2. **If you were to use tf-idf concepts, how would you capture them?**
3. **How do you remove noisy or spam groups?**
4. **How would you identify categories of events?**

Role entities Play

The person involved, or the place an event took place, plays an important role to describe the event because these are the contextual clue which determine specifics of an event. Entities are singular and separate objects that can be a person's, organization or location's name.

Entity detection technology is an expensive process because about half a billion tweets are made in a day which is a lot of data.

Algorithm to Detect Events from such group

The algorithm is designed in such a way that it reduces the cost of detecting entities, use TF-IDF concepts, removes noisy or spam groups and identifies categories of events.

Pre - Processing

1. On text of each tweet perform named entity recognition and extract lemmatized nouns, verbs and named entities (person, location and organization). *Perform Filtering* on the available tweets by removing tweets that does not include any entities identified previously or spam tweets, retweets and non-related tweets. *This will result in a clean dataset with less noise and reduces the amount of data that needs to be processed*.

Clustering

2. Tweets must contain at least one-named entity, based on this we segregate the tweets. Hence, it reduces computational complexity. *Use TF-IDF concept to quantify entities*. Take a tweet t, identify all entities in the tweet t. For each entity e, retrieve list of tweets T from inverted index of e and calculate maximum TF-IDF weight cosine similarity score between t and each tweet in T. If max score in above a threshold, then add t to the nearest neighbour's cluster. If the nearest neighbour does not already belong to a cluster, then a new cluster is created containing both tweets and assigned to entity e. limit the number of tweets that can be retrieved from an entity's inverted index e.g., to 200 and only top 10 TF-IDF weighed terms are used per tweet.

Burst Detection

3. Burst Detection is a technique for distinguishing relevant from duplicate tweets during an event. When an entity's frequency exceeds a predetermined threshold, it may indicate the likelihood of a new event or the introduction of worthless data. The three-sigma rule states that if the difference between the mean and the threshold exceeds three standard deviations, the event will burst. When bursting occurs it means new event occurred. We stop updating

entity frequency information by removing irrelevant and redundant data until the number drops below the threshold. This gives us time to capture any updates or follow up regarding the events.

Cluster Identification

4. The entity that crossed the threshold and was in a burst state is associated with an event. Because most of the tweets sent during the burst are noisy data, the event is left without any tweets. To exclude clusters with background subjects, we associate entity clusters with the event, requiring that the centroid time of a cluster begin after the burst. Every time a new tweet is added, the cluster's centroid time is updated, and we also establish a minimum size threshold to keep noisy clusters out. When all entities are attached to a stopped-bursting event, the event shuts; as a result, the number of clusters in that event is fixed and cannot be changed. To integrate the different events into one, we need to categorise them. This means that we can use a one-way relationship to connect the lesser events that occur around a major event.

Event Merging

5. We merge two events if an entity is mentioned in at least 50% of tweets in one event and the mentioned entity is currently part of another event. This merging process can be repeated indefinitely to create events with multiple entities and many topics/clusters arising from each entity.

2. Assuming that you have developed a full-fledged event detection system, how would you show the effectiveness of your detection system? Unfortunately, you have not test collections to use. Design a crowdsourcing based design to capture ground truth and show the effectiveness of the system. [Hint: discuss collecting data; generating ground truths; what level of annotation process will be used? What measures will be used for comparison and why?]

Despite the popularity of Twitter, there are only a few corpora accessible, none of which are adequate for evaluating event detection algorithms on a broad scale due to their small size or low quantity of events. To establish a pool of events, we first create large-scale event detection corpora utilising state-of-the-art event detection methods and the Wikipedia Current Events Portal. Then, given the pool of events, use crowdsourcing to create relevance ratings.

We'll need to define an event, its meaning, the entities involved, and the target event we're looking for. We'll need to do event detection next, and then use clustering to find clusters of relevant tweets about a specific event or subject. We can run a crowdsourcing test to see how effective this approach is. We perform Collecting Data by using twitter streaming API for around a month. Preform Filtering to remove spams and non-English words and retweets.

We need our tweet cluster detection to be as close to the ground truth as possible in order for the system's effectiveness to be high. We'll have to run a crowdsourcing test on people who are relevant to the event or have some experience in the issue because a random crowd test could jeopardise the event's legitimacy. Following the selection of target participants, they must be provided a set of tweets as well as a questionnaire or survey that allows them to

categorise the tweets into buckets. We can use spam tweets to assess the sincerity of the participants (honey-pot), and the participants can mark them as unsuitable. Another approach is to ask people to mark the tweet as relevant or not.

We'll know if the test is effective after the annotators have marked 50% of the relevant tweets. To assure the test's quality, we can impose a time limit per tweet for participants, allowing us to eliminate the lazy participants who merely click responses at random. By being lazy, we can also see if the participants are consistently clicking "yes" or "no" to identify it as a relevant tweet for the majority of the tweets. These systems can assist us in determining the efficiency of our system. Crowdsourcing, on the other hand, can cut the time and expense of creating large-scale corpora.

For comparison, we can use F-measure which is harmonic mean of Precision and recall. Precision alone just shows how many correct tweets are identified by algorithm. But since we know the ground-truth, recall tells how many of the total right occurrences were correctly marked by algorithm. Precision and recall should be as near as possible.