

# CS5304 Assignment 3 - Analysis Of Social Graphs

## Group Members

Kuo-wei Tseng (kt535)  
Chong-yee Gan (fc297)

## Instructor

Giri Iyengar

# Table of Contents

## [Table of Contents](#)

### [Section 1: GDELT Dataset](#)

#### [1.1 Data Extraction](#)

#### [1.2 PageRank Construction](#)

#### [1.3 World Map Visualisation](#)

### [Section 2: Reddit Dataset](#)

#### [2.1 Data Extraction](#)

#### [2.2 Undirected Graph Construction](#)

#### [2.3 Directed Graph Construction](#)

# Section 1: GDELT Dataset

The GDELT Dataset, which contained nearly 400 million samples of human society interaction data over 35 years, were extracted using [BigQuery](#). Further details on the GDELT Dataset could be found in the [CAMEO Manual](#).

## 1.1 Data Extraction

In this assignment, we explored the interaction between countries in terms of providing/receiving military aid. This was performed based on the query below, with a “EventCode = 072” filter set to limit the query to military aid events only.

```
1 SELECT
2   Actor1Name, Actor1Geo_Lat, Actor1Geo_Long,
3   Actor2Name, Actor2Geo_Lat, Actor2Geo_Long
4
5 FROM
6   [gdelt-bq:full.events]
7
8 WHERE
9   EventCode = "072" AND EventCode IS NOT NULL
10  AND Actor1Name IS NOT NULL AND Actor1Geo_Lat IS NOT NULL AND Actor1Geo_Lat != 0.0
11  AND Actor1Geo_Long IS NOT NULL AND Actor1Geo_Long != 0.0
12  AND Actor2Name IS NOT NULL AND Actor2Geo_Lat IS NOT NULL AND Actor2Geo_Lat != 0.0
13  AND Actor2Geo_Long IS NOT NULL AND Actor2Geo_Long != 0.0
14
```

Figure 1.1.1: SQL Query on Military Aid GDELT Data

However, as we experimented further with graph/pagerank construction (Section 1.2) and map visualization (Section 1.3), we discovered a need to further process and filter our data to better draw insights from them. Therefore, we decided to query further information on CountryCode and AvgTone:

```
1 SELECT
2   Actor1Name, Actor1CountryCode, Actor1Geo_Lat, Actor1Geo_Long,
3   Actor2Name, Actor2CountryCode, Actor2Geo_Lat, Actor2Geo_Long, AvgTone
4
5 FROM
6   [gdelt-bq:full.events]
7
8 WHERE
9   EventCode = "072" AND EventCode IS NOT NULL
10  AND Actor1Name IS NOT NULL AND Actor1CountryCode IS NOT NULL
11  AND Actor1Geo_Lat IS NOT NULL AND Actor1Geo_Lat != 0.0 AND Actor1Geo_Long IS NOT NULL AND Actor1Geo_Long != 0.0
12  AND Actor2Name IS NOT NULL AND Actor2CountryCode IS NOT NULL
13  AND Actor2Geo_Lat IS NOT NULL AND Actor2Geo_Lat != 0.0 AND Actor2Geo_Long IS NOT NULL AND Actor2Geo_Long != 0.0
14  AND AvgTone IS NOT NULL
15
16 ORDER BY AvgTone
```

Figure 1.1.2: SQL Query on Military Aid GDELT Data with Additional Features

We then further filtered the data to two different time periods, 1990-2000 and 2000-2010, to further draw insights from our pagerank construction and map visualisation in Section 1.2 and Section 1.3 respectively.

```
1 SELECT
2   Year,
3   Actor1Name, Actor1CountryCode, Actor1Geo_Lat, Actor1Geo_Long,
4   Actor2Name, Actor2CountryCode, Actor2Geo_Lat, Actor2Geo_Long, AvgTone
5
6 FROM
7   [gdelt-bq:full.events]
8
9 WHERE
10  EventCode = "072" AND EventCode IS NOT NULL
11  AND Actor1Name IS NOT NULL AND Actor1CountryCode IS NOT NULL
12  AND Actor1Geo_Lat IS NOT NULL AND Actor1Geo_Lat != 0.0 AND Actor1Geo_Long IS NOT NULL AND Actor1Geo_Long != 0.0
13  AND Actor2Name IS NOT NULL AND Actor2CountryCode IS NOT NULL
14  AND Actor2Geo_Lat IS NOT NULL AND Actor2Geo_Lat != 0.0 AND Actor2Geo_Long IS NOT NULL AND Actor2Geo_Long != 0.0
15  AND Year >= 1990 AND Year <= 2000 AND AvgTone IS NOT NULL
16
17 ORDER BY AvgTone
```

Figure 1.1.3: SQL Query on Military Aid GDELT Data with Additional Features from 1990 to 2000

```
1 SELECT
2   Year,
3   Actor1Name, Actor1CountryCode, Actor1Geo_Lat, Actor1Geo_Long,
4   Actor2Name, Actor2CountryCode, Actor2Geo_Lat, Actor2Geo_Long, AvgTone
5
6 FROM
7   [gdelt-bq:full.events]
8
9 WHERE
10  EventCode = "072" AND EventCode IS NOT NULL
11  AND Actor1Name IS NOT NULL AND Actor1CountryCode IS NOT NULL
12  AND Actor1Geo_Lat IS NOT NULL AND Actor1Geo_Lat != 0.0 AND Actor1Geo_Long IS NOT NULL AND Actor1Geo_Long != 0.0
13  AND Actor2Name IS NOT NULL AND Actor2CountryCode IS NOT NULL
14  AND Actor2Geo_Lat IS NOT NULL AND Actor2Geo_Lat != 0.0 AND Actor2Geo_Long IS NOT NULL AND Actor2Geo_Long != 0.0
15  AND Year >= 2000 AND Year <= 2010 AND AvgTone IS NOT NULL
16
17 ORDER BY AvgTone
```

Figure 1.1.4: SQL Query on Military Aid GDELT Data with Additional Features from 2000 to 2010

Further details on our journey and rationale behind using different datasets were explained in Section 1.3. The full documentation of all our queries could be found in the attached “BigQuery sql codes.rtf” file.

## 1.2 PageRank Construction

Events with the “MilitaryAid” EventCode were selected to run the pagerank algorithm. The resultant csv file (queried from our code Figure 1.1.1) was of dimension (n x 6), where n represented the amount of samples and 6 represented the amount of columns/features - Actor1Name, Actor1CountryCode, Actor1Geo\_Lat, Actor1Geo\_Long, Actor2Name, Actor2CountryCode, Actor2Geo\_Lat and Actor2Geo\_Long.

The code for the graph and pagerank construction for different actor names within the dataset were shown in the Figure 1.2.1, whilst the top 10 “actor names” with higher pagerank was shown in Figure 1.2.2.

```

import networkx as nx
import csv
import matplotlib.pyplot as plt
TRAIN_FILE = "militaryAid.csv" #pagerank with actor name
#TRAIN_FILE = "militaryAid_received_byAvgTone.csv" #pagerank with country name

G = nx.DiGraph()
with open(TRAIN_FILE) as f:
    reader = csv.reader(f)
    reader.next()
    for row in reader:
        actor1, actor2 = row[0], row[3] #pagerank with actor name
        #actor1, actor2 = row[1], row[5] #pagerank with country name
        G.add_edge(actor1, actor2)

pr = nx.pagerank(G)
sort_pr = sorted(pr.keys(), key = lambda x :pr[x], reverse=True)
print sort_pr[:10]

```

Figure 1.2.1: Code for Graph Construction and PageRank Algorithm for Actor Names.

```
['MILITARY', 'UNITED STATES', 'GOVERNMENT', 'ARMY', 'IRAQ', 'POLICE', 'AFGHANISTAN', 'SECURITY FORCE', 'RUSSIA', 'PRESIDENT']
```

Figure 1.2.2: Top 10 pageRank Actor Names

As can be seen from Figure 1.2.2, bodies such as “military”, “army”, “police”, “security force”, have high pageRank as they are directly involved in providing military aid on the field. In the meantime, nouns such as “president” and “government” are also placed highly as they are the authorities who/which are involved in the decision of dispatching and controlling the army. Finally, countries such as “United States”, “Iraq”, “Afghanistan”, “Russia” possess high pageRanks as they are often involved in wars and military affairs over the past 35 years, and have probably received/given much military aid to their allies. The most influential Actor1 is, of course, military as they are most involved with the act of providing/receiving military aid.

However, as the pageRank results consisted of many different types (bodies/ authorities/ countries), we decided to run an analysis solely on countries instead. We then further partitioned the data by different time periods (1990-2000 and 2000-2010) to better analyse the military aid trends over the past couple of years.

This was done using the same code as that used in Figure 1.2.1, but with their corresponding csv files (militaryAid\_received\_byAvgTone.csv, militaryAid\_byAvgTone\_1990To2000 and militaryAid\_byAvgTone\_2000To2010). The top 10 “actor countries” with higher pagerank were shown in Figure 1.2.3, Figure 1.2.4 and Figure 1.2.5 respectively.

```
['USA', 'GBR', 'AFG', 'FRA', 'IRQ', 'AFR', 'RUS', 'CHN', 'SOM', 'EUR']
```

Figure 1.2.3: Top 10 pageRank Actor Countries over 35 Years

```
['USA', 'RUS', 'GBR', 'FRA', 'AFR', 'IRQ', 'SOM', 'SAU', 'IRN', 'HTI']
```

Figure 1.2.4: Top 10 pageRank Actor Countries From 1990 to 2000.

```
['USA', 'IRQ', 'AFG', 'GBR', 'FRA', 'AFR', 'RUS', 'EUR', 'SDN', 'SOM']
```

Figure 1.2.5: Top 10 pagerank Actor Countries From 1990 to 2010.

It should be noted that USA was the most influential Actor1 country in all 3 cases. Such was the case as USA is a military powerhouse with numerous allies, and have been involved providing/receiving military aids to/from its many alliances. Throughout the past 35 years, USA, UK, Afghanistan, France, Iraq, Africa, Russia, China, Somalia and Europe were the top 10 pagerank countries as they were involved in alliances and were more consistently active in war.

Compared to the overall list, from 1990 to 2000 Saudi Arabia and Iran were included in the top 10 pagerank entries as they both provided support for several parties in a number of wars throughout that period. These include the [Al-Qaeda insurgency in Yemen](#) and [Afghanistan Civil War](#). On the other hand, Haiti was included in the list due to its involvement in conflicts such as the [Operation Uphold Democracy](#).

From 2000 to 2010, however, Afghanistan and Sudan were in the top 10 list due to their involvement in wars such as the [War in Afghanistan](#) and [War in Darfur](#). Similarly, Europe was included in the list due to its many alliance with countries during the [Iraq War](#), [War in Somalia](#) and so on.

## 1.3 World Map Visualisation

As mentioned in Section 1.1, we first created a visualisation that aimed to answer the question of “Which Country Provided the Most Military Aid”. The Military Aid data was first extracted with additional features on the actors’ country codes. Thereafter, the data was sorted by Actor1CountryCode to ease processing for map visualisation, as shown below:

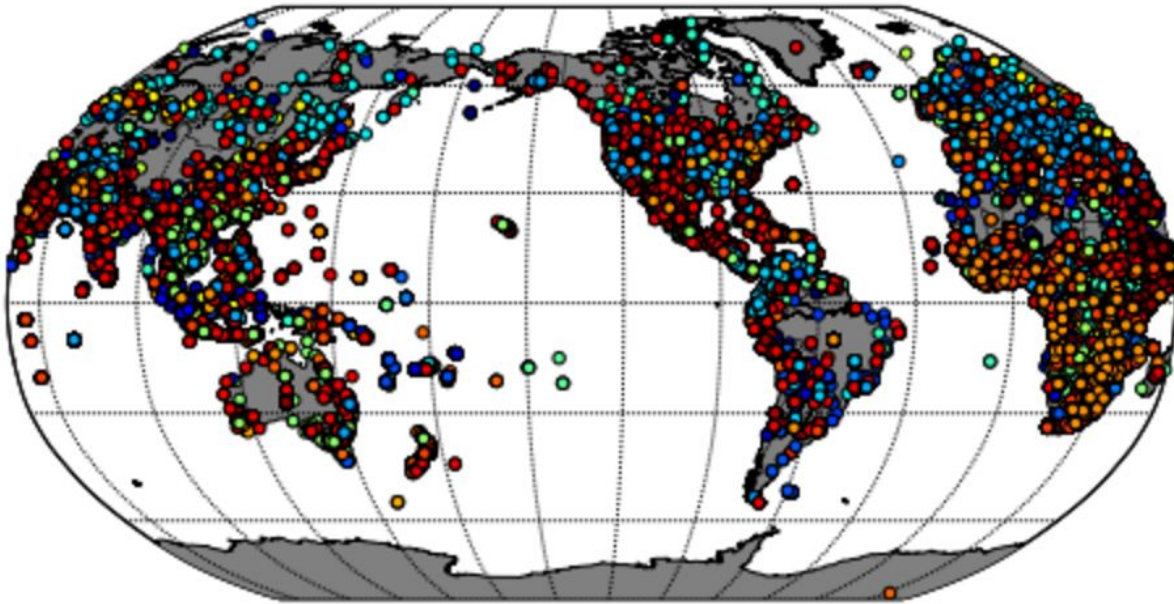
```
1 SELECT
2   Actor1Name, Actor1CountryCode, Actor1Geo_Lat, Actor1Geo_Long,
3   Actor2Name, Actor2CountryCode, Actor2Geo_Lat, Actor2Geo_Long
4
5 FROM
6   [gdelt-bq:full.events]
7
8 WHERE
9   EventCode = "072" AND EventCode IS NOT NULL
10  AND Actor1Name IS NOT NULL AND Actor1CountryCode IS NOT NULL AND Actor1Geo_Lat IS NOT NULL AND Actor1Geo_Long != 0.0
11  AND Actor1Geo_Long IS NOT NULL AND Actor1Geo_Long != 0.0
12  AND Actor2Name IS NOT NULL AND Actor2CountryCode IS NOT NULL AND Actor2Geo_Lat IS NOT NULL AND Actor2Geo_Long != 0.0
13  AND Actor2Geo_Long IS NOT NULL AND Actor2Geo_Long != 0.0
14
15 ORDER BY Actor1CountryCode
```

Figure 1.3.1: SQL Query on Military Aid GDELT Data Sorted By Actor1CountryCode

Having extracted the data, we plotted the location of each Actor2 that received military aid on the map, and color coded these plots by the countries that provided military aid to them (each color represented a country type of Actor1).



The map was visualised using Basemap, with each Actor2 plotted using its scatter function. Note that the `zorder` parameter of `fillcontinents` function was assigned a value of 0, whilst that of the scatter function was assigned a value of 4 to ensure that the plots were overlaid above the continents. Each country was assigned a unique color using numpy's `randint` function and appending them to the `pltColors` array. Each color within `pltColors` was matched to the corresponding country in the `MILITARY_AID` data through the same index. The results were shown in Figure 1.3.2 below.



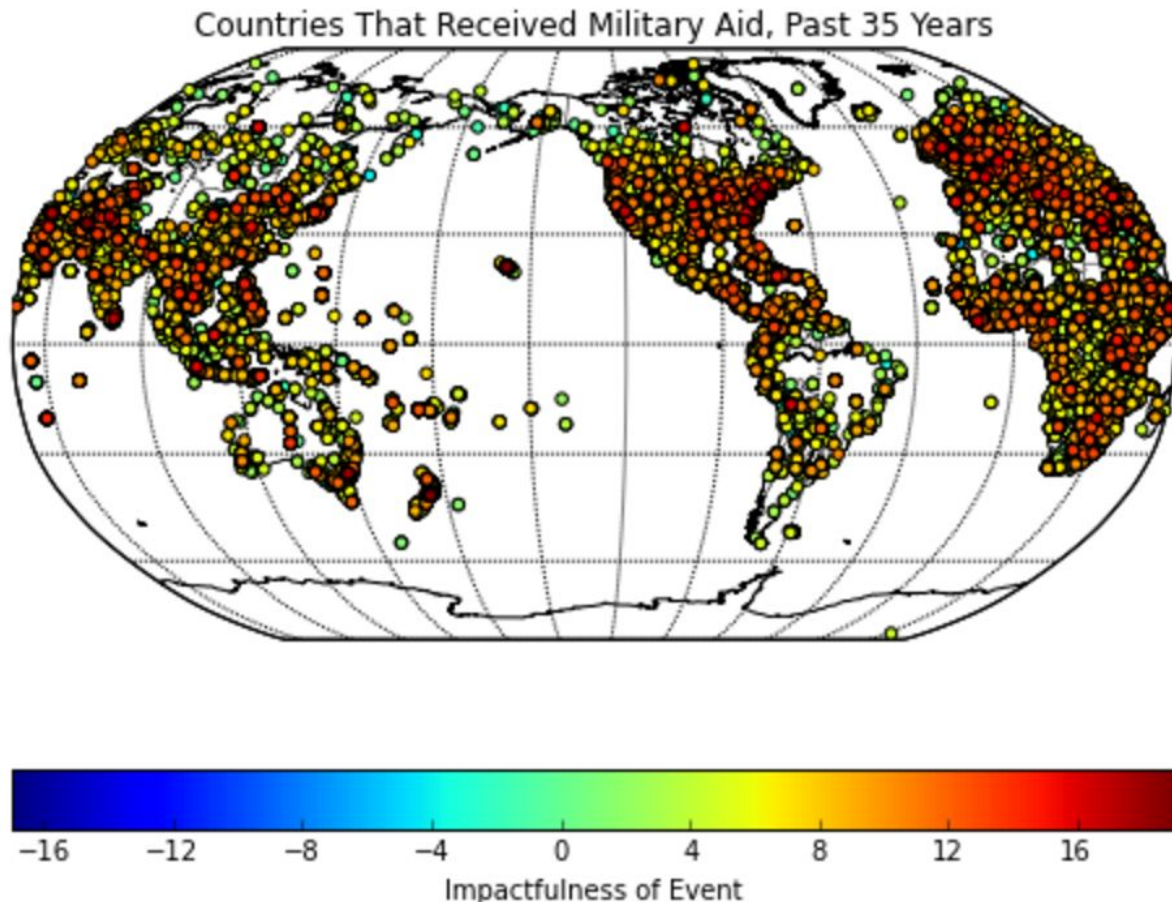
**Figure 1.3.2: Map Visualisation of Countries which Received Military Aid from 212 Different Countries**

However, there were a total of 212 countries that provided military aid, the legend plotted for these countries would be unreadable. This, coupled with the high aggregation of plots above one another (as shown in Africa), made it hard to draw insights from the map as it would be hard to discern and analyse so many unique countries.

We tried to further filter the data by different time periods - 1980 to 1990 and 2000 to 2010. However, the data was still hard to be visualised as they contained 161 and 200 types of countries respectively. Hence, we decided to ask another question which was probably more interesting and easier to draw insights from the map visualisation - "Which Countries Received Military Aids That Were Most Impactful".

This was achieved by first querying the `AvgTone` attribute of each event (Figure 1.1.2), which signified the "tone" of all documents which referred to a particular military aid. As such, a higher score for `AvgTone` signified a more impactful military aid that helped provide stability, whilst a lower or negative value for `AvgTone` signified an ineffective military aid or one that brought negative impacts on the country (as explained by the CAMEO Manual).

The data was visualised in a similar manner as Figure 1.3.2, but with the color adjusted according to the magnitude of the AvgTone/impact of an event. The results were shown in Figure 1.3.3 below.



**Figure 1.3.3: Map Visualisation of Impactfulness of Military Aid Received over the past 35 Years**

The key insight of this map was that most military aid were received in Africa and the Middle East, with most of them having high impact. On the other hand, continents such as Australia and South America received less military aid. By comparison, military aids received by Western Europe was redder as they the units were probably dispatched more efficiently or created stability in a swifter period of time. Military aids received by Africa, on the other hand, were probably less impactful due to corruption, or because it created more unrest within the continent.

However, as a whole, most military aids had high accumulated impactfulness as the data was collected over a period of 35 years. To better visualise the data on a world map, we decided to further filter the data by 2 time periods - 1990 to 2000 and 2000 to 2010. These data were queried using the codes in Figure 1.1.3 and Figure 1.1.4 respectively. The corresponding results were shown in Figure 1.3.4 and Figure 1.3.5 below.



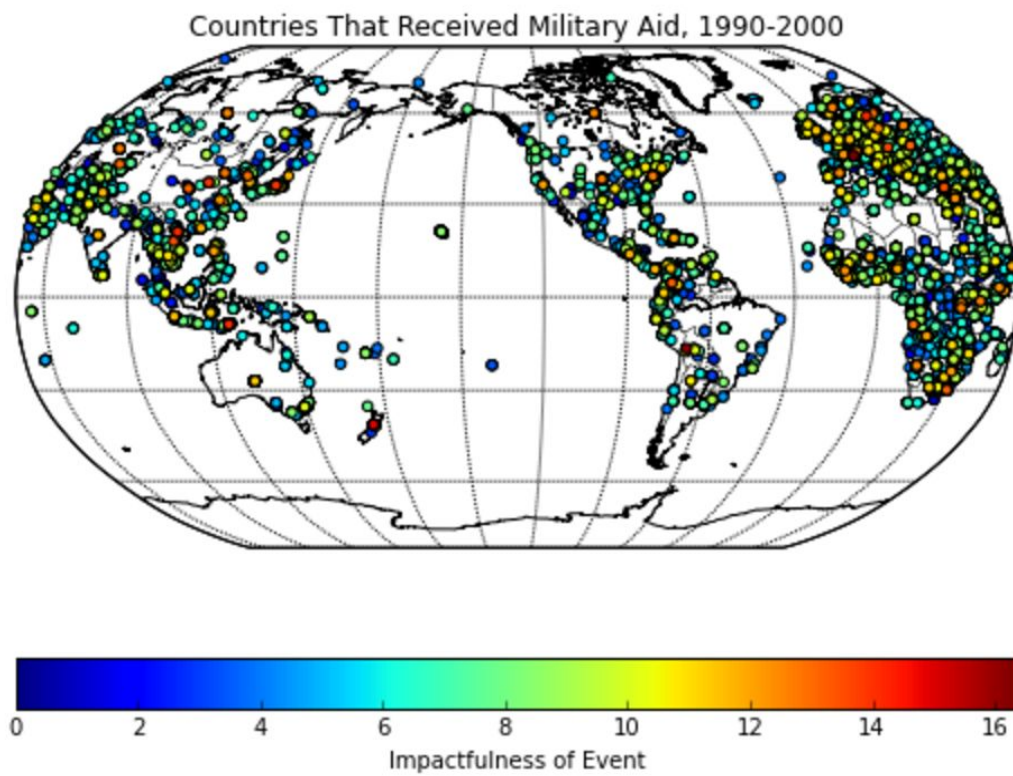
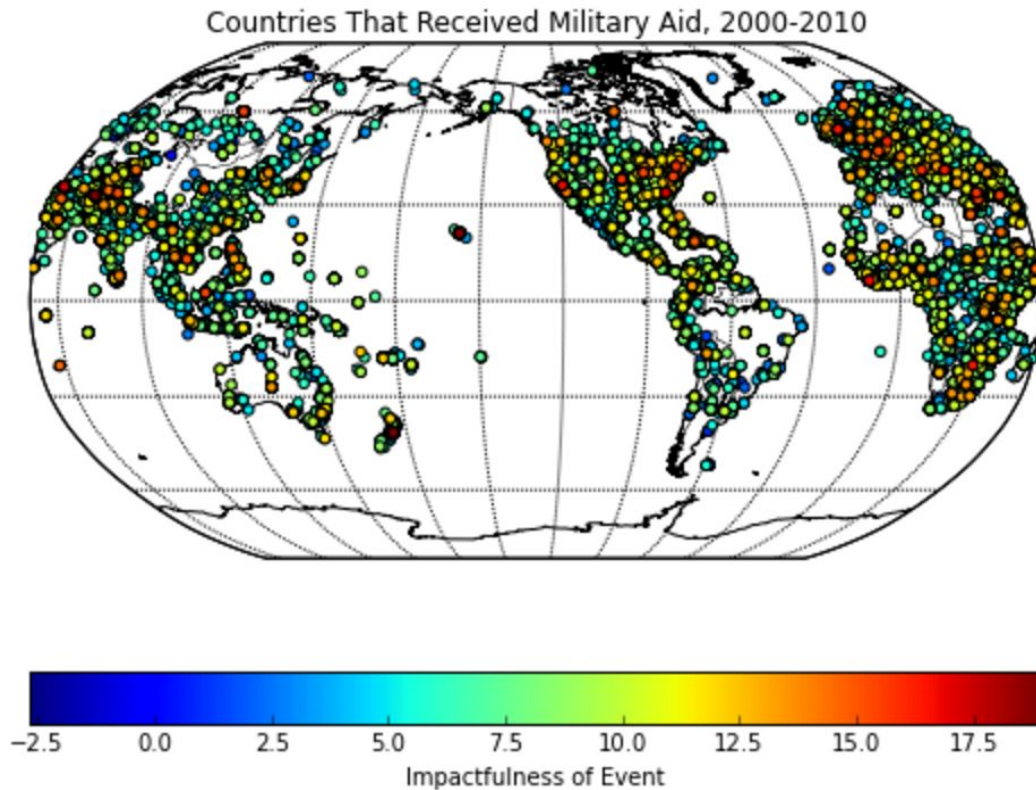


Figure 1.3.4: Map Visualisation of Impactfulness of Military Aid Received from 1990 to 2000



**Figure 1.3.5: Map Visualisation of Impactfulness of Military Aid Received from 2000 to 2010**

As compared to 2000-2010, the 1990-2000 was a more peaceful period, as evidenced by the significantly lesser number of plots. However, most military aids were given in Eastern Middle East and Europe. This insight was in line with our findings in Section 2, where the top pagerank countries were UK, Europe, Afghanistan, Iraq and Somalia. The fact that USA had less plot despite being placed at the top of the pagerank meant that USA was mostly involved in giving military aid, rather than receiving it. In general, the military aids received by these countries were also less impactful as compared to 2000-2010, probably due to the fact that the military aid given was smaller scaled/less effective due to the size of the conflicts, and thus were less effective for the allies to combat their opponents.

By comparison, the period 2000-2010 saw a lot more military aids received across the globe. This was probably due to a higher degree of unrest and war frequency due to the political conflicts in the Middle East and USA/Europe. This was evidenced by a higher concentration of military aids in these three areas. This data was also in line with our findings in Section 1.2. The impactfulness of these military aids were also significantly higher. This was probably due to the fact that the military aids received from both parties were much larger than those in 1990-2000, and were more effective in combatting their enemies.

## Section 2: Reddit Dataset

Similar to Section 1, the 1.7 billion reddit comment dataset was obtained from [BigQuery](#) as well. Further details of the 21 features/columns within this dataset could be found on the [Table Details](#) section of BigQuery.

## 2.1 Data Extraction

For this assignment, we focused on extracting reddit data solely from year [2014](#). This was done to reduce the processing time and computational space needed to extract and analyse the data.

We performed two different types of queries for this section. These included:

1. Directly query first 200k row:

```
1 SELECT
2   author, parent_id, subreddit, subreddit_id, score
3
4 FROM
5   [fh-bigquery:reddit_comments.2014]
6
7 LIMIT 200000
8
```

Because we only query 200k, which is a relatively amount of this table. If we query randomly, the connection between user might be less.

2. Pick up rows in 20 top hot subreddits:

In this query, first queried the top 20 most frequent subreddits:

```
1 SELECT
2   subreddit, COUNT(*) as frequency
3
4 FROM
5   [fh-bigquery:reddit_comments.2014]
6
7 GROUP BY
8   subreddit
9
10 ORDER BY frequency DESC
11
```

Thereafter, the first 200k of filtered data was queried:

```


1 SELECT
2   author, parent_id, subreddit, subreddit_id, score
3
4 FROM
5   [fh-bigquery:reddit_comments.2014]
6
7 WHERE
8   subreddit = "AskReddit" OR subreddit = "funny" OR subreddit = "leagueoflegends" OR subreddit = "AdviceAnimals" OR subreddit = "pics" OR
9   subreddit = "worldnews" OR subreddit = "videos" OR subreddit = "nfl" OR subreddit = "WTF" OR subreddit = "todayilearned" OR
10  subreddit = "gaming" OR subreddit = "nba" OR subreddit = "soccer" OR subreddit = "DotA2" OR subreddit = "news" OR
11  subreddit = "pcmasterrace" OR subreddit = "hockey" OR subreddit = "movies" OR subreddit = "CFB" OR subreddit = "friendsafari"
12
13 LIMIT 200000

```

We only used the comments in the hot topic, which gave the constraint to the topic that the authors wanted to discuss. Therefore, the chance of finding two authors commenting under the same post was much higher, thus increasing the intersection and making the pagerank more accurate.

## 2.2 Undirected Graph Construction

Firstly, we built two unique id dictionary for both posts and authors to convert input string to integers, as it was easier to use them in jaccard matrix evaluation. An example of the resultant dictionary was shown below:

Author	Authro_uid	parent_id	parent_uid
[deleted]	0	t1_ckwv0t7	0
spiiierce	1	t1_ckx3fn5	1
LawlsaurusRex 	2	t1_ckx3fn5	2

Thereafter, for every author, we built a dictionary in which author\_uid was a key and the value was a set of posts' uid posted by this author. We used the native set api, union and intersection, to compute jaccard score.

Author_uid	parent_uid_set
0	set([1,4524, 8753...])
3758	set([384, 765])
23147	set([1, 384, 7769...])

```

comment_dict = defaultdict(set)
#build dict

```

```
#...
#i, j imply the score of author with uid i and author with uid j
tmp_score[0, j] = \
float(len(comment_dict[i] & comment_dict[j])) / len(comment_dict[i] | comment_dict[j])
```

Because we only queried 100k comments to do our test, the jaccard score is relatively low, so if jaccard score between two users > 0, we will add an edge between these two nodes. We ran pagerank with 20k, 50k, and 100k comments:

20k	50k	100k
17014 different users	41005 different users	71688 different users
undirected <b>Pagerank</b> ['[deleted]', 'spiiierce', '2Cuil4School', 'jordanaustin', 'selxxa', 'BergerDog', 'trickyhero', 'msp04', 'The-schnarg', 'wpbops']  directed <b>Pagerank</b> ['[deleted]', 'Kukantiz', 'DonnieNarco', 'GloryMacca', 'crazy_canucklehead', 'mtled', 'fastslowfast', 'CTV49', 'WoWords', 'boognerd']	undirected <b>Pagerank</b> ['[deleted]', '2Cuil4School', 'selxxa', 'dylan833', 'ZombieHilDog', 'BergerDog', 'trickyhero', 'msp04', 'wpbops', 'jordanaustin']  directed <b>Pagerank</b> ['[deleted]', 'Citeh', 'uscjimmy', 'LawlsaurusRex', 'Kukantiz', 'DonnieNarco', 'GloryMacca', 'hummm1n984D93R', 'wetrabbit', 'alysia415']	undirected <b>Pagerank</b> ['[deleted]', 'raouldukehst', 'DatLe_ArtDirector', 'noseonarug17', 'ricodued', 'Thunderkleize', '-khriz-', 'woofers02', 'Sinical89', 'chrishansen24']  directed <b>Pagerank</b> ['[deleted]', 'CleanShirt27', 'Mr_Weebles', 'MishyC', 'brock_lee', 'Mari0us', 'GeebusNZ', 'batistafreed', 'THE_BEST_ANSWER', 'finsnfeathers']

## 2.3 Directed Graph Construction

We defined a metric for our directed graph construction, such that there is a directed edge “b->a” if author “b” comments (with t1 header) under author “a”s posts (with t3 header). Because parent\_id had the header to indicate whether it is the top post, if it is “t3” we added it to a dictionary post\_author\_dict with value of its author, if it is “t1” we add it to the dictionary post\_subauthor\_dict. This was a directed metric as it mapped “b” onto “a”.

post\_author\_dict:

post_uid	Author_uid (with t3 heading)
4154	3512
37414	97123

12374	21374
-------	-------

post\_subauthor\_dict:

post_uid	Author_uid (with t1 and t3 heading)
1016	set([178, 2586,2681])
1414	set([1231,1485,26812])
52234	set([854, 4857,22341])

```

if head == "t3": #means it is the top post
    post_author_dict[post_dict[post_id]] = author_dict[author]
    post_subauthor_dict[post_dict[post_id]].add(author_dict[author])
else:
    post_subauthor_dict[post_dict[post_id]].add(author_dict[author])

```

The code for building directed graph and running pagerank was shown below:

```

def build_diredge_and_pagerank():
    dirG = nx.DiGraph()
    totalkey = post_author_dict.keys() + post_subauthor_dict.keys()
    for key in totalkey:
        if (key not in post_author_dict) or (key not in post_subauthor_dict):
            continue
        actor1_key = post_author_dict[key]
        actor1 = rev_author_dict[actor1_key]
        actor2_key_list = post_subauthor_dict[key]
        if len(actor2_key_list) == 1:
            continue
        for actor_key in actor2_key_list:
            if actor1_key == actor_key:
                continue
            actor2 = rev_author_dict[actor_key]
            dirG.add_edge(actor1, actor2)
    pr = nx.pagerank(dirG)
    sort_pr = sorted(pr.keys(), key = lambda x :pr[x], reverse=True)
    print sort_pr[:10]
    return pr

```

20k	50k	100k
17014 different users	41005 different users	71688 different users



<code>directed Pagerank [['deleted'], 'Kukantiz', 'DonnieNarco', 'GloryMacca', 'crazy_canucklehead', 'mtled', 'fastslowfast', 'CTV49', 'WoWords', 'boognerd']</code>	<code>directed Pagerank [['deleted'], 'Citeh', 'uscjimmy', 'LawlsaurusRex', 'Kukantiz', 'DonnieNarco', 'GloryMacca', 'hummm1n984D93R', 'wetrabbit', 'alysia415']</code>	<code>directed Pagerank [['deleted'], 'CleanShirt27', 'Mr_Weebles', 'MishyC', 'brock_lee', 'Mari0us', 'GeebusNZ', 'batistafreed', 'THE_BEST_ANSWER', 'finsnfeathers']</code>
--	---	--