



# VIT<sup>®</sup>

**Vellore Institute of Technology**  
(Deemed to be University under section 3 of UGC Act, 1956)

## **Title: Finding Bias in Political News in social media**

*Social and Information Networks*  
*CSE3021*

*Submitted by*

SL. NO.	NAME	REGISTRATION NO.
1.	HARSHVARDHAN MISHRA	19BCB0125
2.	SHREYA RASTOGI	19BCE0756
3.	SHYAM RANJAN BHARTI	19BCE0784

## **1. OBJECTIVE**

There is a growing body of evidence of bias in the media caused by underlying political and socio-economic viewpoints. To classify the partiality of the media, but there is little work on quantifying it, and less still on the nature of this partiality. The vast amount of content published in social media enables us to quantify the inclination of the press to predefined sides of the socio-political spectrum. Media bias is sometimes caused by Columnists neglecting to report all the current stories to the gathering of people. The greater part of times media lack of bias is affected by the administration and its degree fluctuates in various nations. Right now, present another procedure that evaluates the political inclining of news outlets dependent on the computerization of a notable political test.

## **2. SCOPE**

The media impacts how individuals see the world that encompasses them. Increasingly more force has been attributed to the advanced press since its origin, in any event, considering it the "Fourth Estate" accentuating its autonomy and its capacity to give exacting cut-off points to what governments could conceivably do. There are notable instances of the press in any event, toppling governments: the Washington Post in the Watergate embarrassment is maybe the most reverberating model. A survey conducted among news consumers confirms that media bias has an impact on the coverage of controversial topics and that this is perceivable by the general audience. This will empower readers to better reflect on the content provided by their news outlets of choice.

### 3. ABSTRACT

There is a growing body of evidence of bias in the media caused by underlying political and socio-economic viewpoints. Previous studies have tried to classify the partiality of the media, but there is little work on quantifying it, and less still on the nature of this partiality. The vast amount of content published in social media enables us to quantify the inclination of the press to pre-defined sides of the socio-political spectrum. To describe such tendencies, we use tweets to automatically compute a news outlet's political and socio-economic orientation. To show that the media have a measurable bias, and illustrate this by showing the favouritism of Chilean media for the ruling political parties in the country. This favouritism becomes clearer as we empirically observe a shift in the position of the mass media when there is a change in government.

### 4. LITERATURE SURVEY

S.No	Title	Source	Year	Objective	Limitation
1	More Voices Than Ever? Quantifying Media Bias in Networks	<a href="https://arxiv.org/pdf/1111.1227.pdf">https://arxiv.org/pdf/1111.1227.pdf</a>	2011	Proposed empirical measures to quantify the extent and dynamics of “bias” in mainstream and social media (hereafter referred to as News and Blogs, respectively). The measurements are not normative judgment, but examine bias by looking at the attributes of those being mentioned, against a null model of “unbiased” coverage.	They plan to continue work along the lines such as long-term tracking of slant dynamics in the two media, modeling individual outlets’ biases, and leveraging content analysis and multivariate analysis

2	A MEASURE OF MEDIA BIAS	<a href="https://mospace.uconn.edu/xmlui/bitstream/handle/10355/2637/MediaBias.pdf">https://mospace.uconn.edu/xmlui/bitstream/handle/10355/2637/MediaBias.pdf</a>	2005	We count the times that a particular media outlet cites various think tanks and policy groups, then compare this with the times that members of Congress cite the same groups. All of the findings refer strictly to news content; that is, we exclude editorials, letters, and the like.	None has provided a way to link such a measure to ideological measures of other political actors. we omitted editorials, book reviews, and letters to the editor from our sample.
3	Public Perceptions of Media Bias: During the 2012 American Presidential Election	<a href="https://www.elon.edu/u/academics/communications/journal/wp-content/uploads/sites/153/2017/06/05DanielQuackenbushEJFall13.pdf">https://www.elon.edu/u/academics/communications/journal/wp-content/uploads/sites/153/2017/06/05DanielQuackenbushEJFall13.pdf</a>	2013	During that 2012 election time With every election year, the American public continues to perpetuate the stereotype that the American news media is ideologically biased, negatively shaping other citizens' views of the American political system and impacting their willingness to participate in the electoral process.	The author acknowledges that the considerably limited scope of the accumulated results and the consistent reliance on selected sources to provide the bulk of the study's statistical basis does not permit him to speak indisputably about either the source or the depth of the perceptual bias this paper has sought to document.
4	Quantifying Search Bias: Investigating Sources of Bias for Political Searches in Social Media	<a href="https://arxiv.org/pdf/1704.01347.pdf">https://arxiv.org/pdf/1704.01347.pdf</a>	2017	Proposed a framework to quantify these distinct biases and apply this framework to politics-related queries on Twitter. We found that both the input data and	A limiting factor in our study was using the simplifying assumption of considering a user as either neutral, pro-democratic or pro-republican. Under this

				the ranking system contribute significantly to produce varying amounts of bias in the search results and in different ways.	assumption we can not have a user who is partially both pro-republican and pro-democrat.
5	A Machine Learning Pipeline to Examine Political Bias with Congressional Speeches	<a href="https://arxiv.org/pdf/2109.09014.pdf">https://arxiv.org/pdf/2109.09014.pdf</a>	2021	Proposed methods exploit the use of transcripts collected from political speeches in US congress to label the data and achieve the highest accuracy of 70.5% and 65.1% in Twitter and Gab data respectively to predict political bias. We also present a machine learning approach that combines features from cascades and text to forecast cascade's political bias with an accuracy of about 85%.	The proposed data labelling procedure is very generic such that it can be applied to multiple perspectives to study problems like media bias and fake news in the future. Even though the given method is more effective, more thorough case studies with valid ground-truth data can help to understand and update the approach for more robust political bias modelling in social media.

### 3. Overview of the Proposed System

#### 3.1. Introduction

Finding Dataset in which two nodes are connected by an edge where vertex 1 and vertex 2 in the same row represents a line. We found an efficient dataset from. We used Gephi software. We used a simple concept that whichever node twitter or Facebook has more links to the biased news must be having more in degree and also must have more betweenness centrality.

Therefore the node with more betweenness centrality is the social networking site that is more politically biased

The first step in quantifying the bias for political searches on social media and web search is measuring the political bias of an individual result (i.e., a tweet or a weblink). There have been some attempts to infer the political bias of blogs and news stories as well as hashtags on Twitter . However, there has been limited work on inferring the bias of content of short social media posts like tweets. Instead, researchers have inferred the bias of the users posting tweets by modeling how different polarity users use language , or by leveraging the linking behavior of users or by leveraging both textual and network features for political leaning, have quantified the impartiality of social media posts by measuring how easy it is to guess the political leaning of its author. Bond and Messing (2015) inferred the political leanings of Facebook users by observing the endorsements of Facebook Pages of known politicians, while measuring the endorsements in terms of retweeting behavior of users to infer their political leanings. We used supervised methods to classify users into different groups of political activities and showed that it is hard to infer the political leaning of “normal” users. Most of these prior studies make the assumption that the learning of the user is explicit in their language, social connections or endorsements, however this may not always be true. We build upon these prior studies to propose a methodology for inferring the bias of a Twitter user.

- Proposed empirical measures to quantify the extent and dynamics of “bias” in mainstream and social media (hereafter referred to as News and Blogs, respectively ). The measurements are not normative judgment, but examine bias by looking at the attributes of those being mentioned, against a null model of “unbiased” coverage.

- We count the times that a particular media outlet cites various think tanks and policy groups, then compare this with the times that members of Congress cite the same groups. All of the findings refer strictly to news content; that is, we exclude editorials, letters, and the like.
- In order to identify general tendencies and to uncover nuanced findings, news sharing research was analyzed both quantitatively and qualitatively. Three central areas of research— news sharing users, content, and networks— were identified and systematically reviewed. In the central concluding section, the results of the review are used to provide a critical diagnosis of current research and suggestions on how to move forward in news sharing research
- During that 2012 election time, With every election year, the American public continued to perpetuate the stereotype that the American news media is ideologically biased, negatively shaping other citizens' views of the American political system and impacting their willingness to participate in the electoral process.

#### *Bias score of an individual data item-*

We are interested in quantifying the search bias for political queries in the context of US politics. Since there are two primary political parties in the US, each data item (e.g., a tweet or a web-link) can be positively biased (i.e., supportive), negatively biased (i.e., opposing), or neutral towards each of the parties. Therefore the bias score of each item must capture the extent to which it is biased with respect to the two parties. To apply our bias quantification framework in the context of political searches on a search platform, we need a methodology for inferring the bias scores for each data item.

Next, we use these bias scores of individual data items to define the metrics for the input bias, output bias, and ranking bias.

**Input bias-** Once a user issues a query, the search system retrieves a set of items from the whole corpus that are relevant to the query and provides them as an input to the ranking system. Since this input data captures the bias introduced due to the filtering of the relevant items from the data corpus according to the issued query, we measure the input bias for a query as the aggregate bias of all the items relevant to the query in this input data set. In other words, input bias gives a measure of the bias a user would observe if they were shown random items relevant to the query, instead of the output list ranked by the ranking system. Specifically, the Input Bias  $IB(q)$  for query  $q$  is the average bias of the  $n$  data items that are relevant to  $q$

$$IB(q) = \frac{\sum_{i=1}^n s_i}{n}$$

Rank $r$	Bias till rank $r$	Value
1	$B(q, 1)$	$s_2$
2	$B(q, 2)$	$\frac{1}{2}(s_2 + s_4)$
3	$B(q, 3)$	$\frac{1}{3}(s_2 + s_4 + s_5)$
4	$B(q, 4)$	$\frac{1}{4}(s_2 + s_4 + s_5 + s_1)$
5	$B(q, 5)$	$\frac{1}{5}(s_2 + s_4 + s_5 + s_1 + s_3)$
Output bias at rank 5		$\frac{1}{5}[s_2(1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5})$ $+ s_4(\frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5})$ $+ s_5(\frac{1}{3} + \frac{1}{4} + \frac{1}{5})$ $+ s_1(\frac{1}{4} + \frac{1}{5})$ $+ s_3(\frac{1}{5})]$

Fig 1. Explaining the bias metrics with reference to the search bias quantification framework.



*Output bias-* The output bias of a search system is the cumulative bias in the final ranked list of search results presented to the user who issued the search query. Prior studies have shown that not only are the users more likely to browse the top search results, but they also tend to put more trust in them . Therefore, we propose an output bias metric inspired from the well-known metric—mean average precision—which gives more importance to higher ranked search results. For a given search query  $q$ , we first define the bias till a particular rank  $r$  in the ranked results (i.e., the aggregate bias of the top  $r$  results). The bias  $B(q, r)$  till rank  $r$  of the output ranked list is defined as-

$$B(q, r) = \frac{\sum_{i=1}^r s_i}{r}$$

*Ranking bias-* If the internal details of the deployed ranking system were known, then the ranking bias could be measured by auditing the exact features being used for ranking. However, for most of the real-world commercially deployed search engines, the internal details of the ranking system are not known publicly. Therefore, building on previous studies that have adopted a “black-box” view for an algorithmic system while auditing it, we treat the ranking system as a black-box, such that we only observe its inputs and outputs. Therefore, we define the Ranking Bias  $RB(q, r)$  for the query  $q$  as simply the difference between the output bias and the input bias for  $q$ -

$$RB(q, r) = OB(q, r) - IB(q)$$

*Time-averaged bias-* To capture the overall trend in the bias, we collect multiple snapshots of search results, compute the different bias metrics for each snapshot, and then compute the time-averaged values of the aforementioned metrics.

For instance we compute the time-averaged output search bias  $TOB(q,r)$  as the average of the  $OB(q, r)$  values measured at various instants of time. Similarly, we define  $TIB(q)$  and  $TRB(q,r)$  as the time averaged input bias and time-averaged ranking bias for query  $q$  respectively.

### 3.2. Methods Adopted For the Proposed System

#### ***Vertices***

A vertex is an element of a network. The vertices count is the number of people or things in the network.

#### ***Unique Edges***

An edge is a connection between two vertices. The “unique” edge count is the number of connections where multiple connections between A and B are counted only once.

#### ***Edges With Duplicates***

An edge is a connection between two vertices. The “duplicate” edges count is the total number of multiple connections between two vertices.

#### ***Total Edges***

The “total” edges count is the total number of connections where multiple connections between A and B are all counted.

#### ***Self-Loops***

An edge that starts and ends in the same vertex is a self-loop. These are also called isolates.

### ***Reciprocated Vertex Pair Ratio***

When two vertices both link to each other their connection is “reciprocated”. This is the percentage of vertices that have a reciprocal relationship. When an edge from A to B is joined by another edge from B to A then their connection is “reciprocated”.

### ***Reciprocated Edge Ratio***

This is the percentage of edges that have a reciprocal relationship.

### ***Connected Components***

A group of vertices that are all connected is a component. This is the number of separate sets of connected vertices.

### ***Single-Vertex Connected Components***

A vertex that has zero connections is “isolated” or an “island”. This is the count of vertices that have zero connections.

### ***Maximum Vertices in a Connected Component***

A connected component is composed of a number of vertices. This is the count of vertices in the largest connected component.

### ***Maximum Edges in a Connected Component***

A connected component is composed of a number of edges. This is the count of total edges in the largest connected component.

### ***Maximum Geodesic Distance (Diameter)***

A geodesic is a chain or path composed of edges that link two vertices, potentially through intermediate vertices. A “shortest path” is the minimum number of connections needed to link two vertices. The “longest” “shortest path” is the

“Maximum geodesic distance”.

### ***Average Geodesic Distance***

A geodesic is a chain or path composed of edges that link two vertices, potentially through intermediate vertices. The average length of these paths is the “average geodesic distance”.

### ***Graph Density***

Density is the measure of the number edges among a group of vertices over the total possible number if everyone was connected to everyone. A high graph density means that most people are connected to many others. A low graph density means that most people are not connected to many others.

### ***Modularity***

Modularity is a measure of the fitness of the groups that are created in a clustered network. Many group cluster algorithms are intended to find sets of vertices that are strongly connected and relatively separate from other strongly connected groups. Modularity is the measure of the number of edges that leave a group to connect to vertices in a different group. If modularity is low, the clusters or groups created may be of low quality. If modularity is high, the groups are well defined.

### ***Bipartite***

A graph  $G = (V, E)$  is called a bipartite graph if its vertices  $V$  can be partitioned into two subsets  $V_1$  and  $V_2$  such that each edge of  $G$  connects a vertex of  $V_1$  to a vertex  $V_2$ . It is denoted by  $K_{mn}$ , where  $m$  and  $n$  are the numbers of vertices in  $V_1$  and  $V_2$  respectively.

### ***Sampling of our dataset***

We should use sampling with replacement when we have a large dataset. Because if we use sampling without replacement then the probability for

each item to be picked will keep changing and it will be too complex after a certain point. Sampling with replacement can tell us what is more frequent in our data.

### ***Centrality Measures-***

- **Betweenness Centrality**

It is a way of detecting the amount of influence a node has over the flow of information in a graph. It is often used to find nodes that serve as a bridge from one part of a graph to another.

The betweenness centrality of a node  $v$  is given by the expression:

$$g(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

- **Degree Centrality**

Degree centrality of a node refers to the number of edges attached to the node.

In order to know the standardized score, you need to divide each score by  $n-1$  ( $n$  = the number of nodes).

$$CD(j) = \frac{1}{n-1} \sum_{i=1}^n A_{ij}$$

- **Katz centrality**

Katz centrality computes the relative influence of a node within a network by measuring the number of the immediate neighbors (first degree nodes) and also all other nodes in the network that connect to the node under consideration through these immediate neighbors

$$C_{\text{Katz}}(i) = \sum_{k=1}^{\infty} \sum_{j=1}^n \alpha^k (A^k)_{ji}$$

- **Eigenvector centrality**

Eigenvector centrality (also called eigen centrality or prestige score) is a measure of the influence of a node in a network. Relative scores are assigned to all nodes in the network based on the concept that connections to highscoring nodes contribute more to the score of the node in question than equal connections to low-scoring nodes. A high eigenvector score means that a node is connected to many nodes who themselves have high scores.

$$x_i(t) = \sum_j A_{ij} x_j(t-1)$$

### **Clustering Measures**

To measure cluster tendency is to measure to what degree clusters exist in the data to be clustered. Clustering is an unsupervised machine learning algorithm. It helps in clustering data points to groups. If one wants to do clustering with ground truth labels being present, validation methods and metrics of supervised machine learning algorithms can be used.

### **Reciprocity**

Reciprocity of a directed network is the fraction of edges that belong to a loop of length two.

$$R = \frac{\sum_{i,j} A_{i,j} A_{j,i}}{\sum_{i,j} A_{i,j}} = \frac{\text{Tr}(A^2)}{m}$$

### **Transitivity**

If refers to the extent to which the relation that relates two nodes in a network that are connected by an edge is transitive.

$$T = \frac{3 \times \text{number of triangles in the network}}{\text{number of connected triples of nodes in the network}}.$$

### **Degree Distribution**

$P(k)$  of a network is then defined to be the fraction of nodes in the network with degree  $k$ . Thus if there are  $n$  nodes in total in a network and  $n_k$  of them have degree  $k$ .

$$P(k) = \frac{n_k}{n}.$$

**Community Detection- • Clique Percolation Method(Require Parameter  $k$ )**

1. Return Overlapping Communities
2. Cliques $_k$  = find all cliques of size  $k$
3. Construct clique graph  $G(V, E)$ , where  $|V| = |\text{Cliques}_k|$
4.  $E = \{e_{ij} \mid \text{clique } i \text{ and clique } j \text{ share } k - 1 \text{ nodes}\}$
5. Return all connected components of  $G$

• **Girvan–Newman algorithm-**

1. Calculate edge betweenness for all edges in the graph
2. Remove the edge with the highest betweenness
3. Recalculate betweenness for all edges affected by the edge removal.
4. Repeat until all edges are removed

### **3.3. Proposed System Model**

*Overview of our search bias quantification framework-* For a given query  $q$ , a set of data items relevant to the query is first selected. Each individual data item (e.g.,  $i_1, i_2$ ) has an associated bias score (e.g.,  $s_1, s_2$ ). This set of relevant items is input to the ranking system which produces a ranked list of the items.

Our framework includes metrics for measuring the bias in the set of relevant items input to the ranking system (input bias), and the bias in the ranked list output by the ranking system (output bias) . Investigating sources of bias for political searches on social media-

Having described our search bias quantification framework, we next apply it to political searches on Twitter social media for queries related to 2016 US presidential primaries.

With this study, we highlight an important application scenario of our framework, where not only can we observe the search system's output results, but we also can observe the set of relevant items that form the input to the ranking system

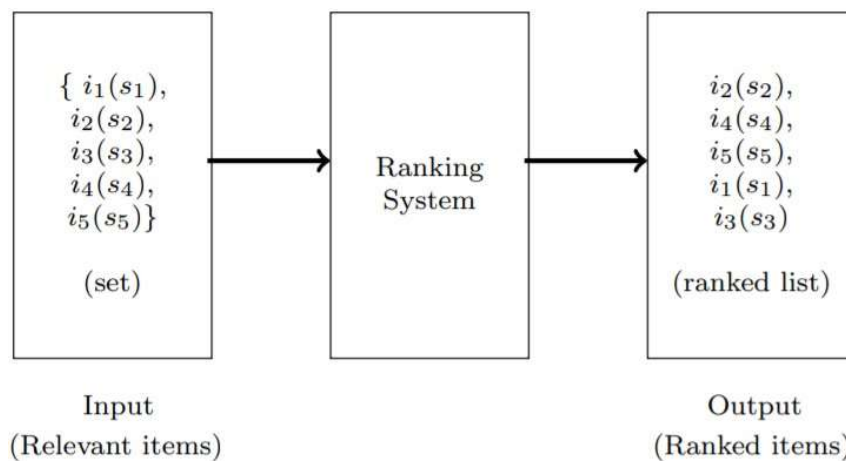


Fig. 2 Overview of our search bias quantification framework

We begin by describing our selected queries and data set for Twitter search, followed by the methodology we used for measuring the political bias of an individual Twitter search result ,and then we finally present our findings about how biased are the search results for political topics on Twitter and where does this bias in the search results came



### ***Collecting Twitter search data -***

Here, we describe the queries we considered and the data gathered from Twitter for conducting the analyses.

### ***Selecting search queries-***

In an ideal scenario, for studying the bias in political searches on Twitter, we would use the actual search queries that people are making on the platform for following news and information related to 2016 US presidential primaries. However, we did not have access to this proprietary data about the queries issued on Twitter. In the absence of the actual search queries issued on Twitter, we followed the methodology used in Koutra et al. (2015) of first identifying a seed set of queries and then expanding them to identify a larger set of potential queries. Our seed set consists of the queries democratic debate, and republican debate, and their shortened versions (dem debate and rep debate) popular on Twitter because of their short lengths. We wanted our expanded set of queries to satisfy two properties: (i) they should be popular and be used by many users, and (ii) they should not be biased towards any particular party, candidate or organization in their formulation, i.e., the leaning of the user issuing the query should not be obvious from the query.

### ***Data collection from Twitter-***

For applying our bias quantification framework to Twitter search, we needed to collect data about the output search results given out by Twitter's ranking algorithm, as well as the set of tweets which were relevant to our selected queries that form the input to the ranking system. For performing our bias analysis, we collected the search data for a one week period in which both a Democratic debate (December 19, 2015) and a Republican debate (December 15, 2015) took place—14–21 December 2015. Even

though Twitter provides multiple different filters for their search functionality, we collected the search snapshots for our set of selected queries for the default filter of “top” search results (<https://twitter.com/search-home>). The “top” search results are the output of Twitter’s proprietary ranking system, which performs ranking based on a multitude of factors, including the number of users engaging with a tweet (<https://help.twitter.com/en/using-twitter/top-search-results-faqs>). During the one-week period, search snapshots were collected at 10-min intervals for each query. Each snapshot consists of the top 20 results on the first page of search results, and we used these snapshots to compute the output bias for the queries. Across all queries, we collected a total of 28,800 snapshots which consisted of 34,904 distinct tweets made by 17,624 distinct users.

*Source bias: inferring political bias of Twitter users-*

Prior studies have shown that people’s political affiliations are correlated with their personality attributes and responses to different stimuli). Based on this knowledge, we propose a methodology for inferring political leaning of Twitter users by leveraging their interests. Therefore, our methodology for inferring the political bias of a Twitter user  $u$ , is based on the following three steps:

- Generating representative sets of Democratic and Republican users
- Inferring topical interests of a user
- Matching user’s interests to interests of Democrats and Republicans

*Comparing relative bias in political searches on the web and social media*

- We apply our bias quantification framework to compare the relative biases of political searches on two different search systems—Twitter social media search and Google Web search. This second study highlights another useful application scenario for our bias quantification framework where we can observe the output search results, but we do not have access to the input

data to the ranking system (as is the case with most commercial search systems). This unavailability of input data makes it infeasible to disentangle the effect of input data and ranking system by measuring input bias and ranking bias separately, however, we can still compare the relative biases of different search systems. Our choice of the two search systems to compare (Google and Twitter search) was driven by the fact that these are two popular channels by which internet users are finding news and information on the Web. Traditional media channels like Fox News or CNN have often been scrutinized by academics. However, the relative biases of newer digital channels like search systems are not as well studied and documented as yet, and thus users may not be taking their relative biases into account while selecting where to get their information from. With this study, we aim to highlight the differences in the bias of these two popular search systems—Twitter social media search and Google Web search. To have a fair comparison, we compare the Google search results with Twitter ‘news’ search results (<https://twitter.com/search-home>), both of which frequently contain results from news media sources.

Finally, the bias score of user  $u$  with interest vector  $I_u$  is given by the difference in the cosine similarities of  $I_u$  with  $I_D$  and  $I_R$ ,

$$Bias(u) = cos\_sim(I_u, I_D) - cos\_sim(I_u, I_R).$$

We max-min normalize the scores such that the bias score of a user lies in the range  $[-1.0, 1.0]$ , with a score closer to  $+1.0$  indicating more Democratic bias, while a score closer to  $-1.0$  indicating more Republican bias.

# IMPLEMENTATION

## CODE

```
# Importing used libraries for this social network graph study
import networkx as nx
import matplotlib.pyplot as plt
import csv
import random
import matplotlib.colors as mcolors
import random
import pandas as pd
import datetime
import io
import array, re, itertools
import numpy as np
import math

import warnings
import matplotlib.cbook
warnings.filterwarnings("ignore", category=matplotlib.cbook.mplDeprecati
on)

# Taking Twitter Dataset in graph and displaying it's features.
g_cp = nx.read_edgelist('/content/drive/My Drive/Colab Notebooks/Datase
t/twitter_combined.txt', nodetype=int, create_using=nx.DiGraph())
print (nx.info(g_cp))

from networkx.algorithms import bipartite
print("Is Biparted? ")
print(bipartite.is_bipartite(g_cp))

# Sampling Social Circles Facebook Dataset
result = []
f = open('/content/drive/MyDrive/twitter_combined_Sample.txt', "w")
with open('/content/drive/MyDrive/twitter_combined.txt') as fh:
    data = fh.readlines()
    for line in data:
        if line:
            words = line.split()
            result.append(words)
sample = random.sample(result, 20000)

for item in sample:
```

```

        f.write(item[0] + "\t" + item[1] + "\n")
print ("Sampled Dataset Information:")
g_sampled = nx.read_edgelist('/content/drive/MyDrive/twitter_combined_Sample.txt', nodetype=int, create_using=nx.DiGraph())
print (nx.info(g_sampled))

from networkx.algorithms import bipartite
print("Is Biparted? ")
print(bipartite.is_bipartite(g_sampled))

# Sampling Social Circles Facebook Dataset
result = []
F = open('/content/drive/MyDrive/facebook_combined_Sample.txt', "w")
with open('/content/drive/MyDrive/facebook_combined.txt') as fh:
    data = fh.readlines()
    for line in data:
        if line:
            words = line.split()
            result.append(words)
sample = random.sample(result, 20000)

for item in sample:
    F.write(item[0] + "\t" + item[1] + "\n")
print ("Sampled Dataset Information:")
G_sampled = nx.read_edgelist('/content/drive/MyDrive/facebook_combined.Sample.txt', nodetype=int, create_using=nx.DiGraph())
print (nx.info(G_sampled))

# Twitter Dataset Sample
pos = nx.spring_layout(g_sampled)
# Degree Centrality
degCent = nx.degree_centrality(g_sampled)
# Betweenness Centrality
betCent = nx.betweenness_centrality(g_sampled, normalized=True, endpoints=True)
#eigen vector centrality
eig_cen=nx.eigenvector_centrality(g_sampled,1000000) # 1000000-->precision
key_max = max(eig_cen.keys(), key=(lambda k: eig_cen[k]))
print('Maximum eigen-vector centrality:',eig_cen[key_max],end=" ")
print('at node =',key_max)
# Katz centrality
l=nx.katz_centrality(g_sampled, alpha=0.1, beta=1.0, max_iter=1000, tol=1.0e-6, nstart=None, normalized=True, weight=None)
l_max = max(l.keys(), key=(lambda k: l[k]))

```

```

print('Maximum Katz centrality:',l[l_max],end=" ")
print('at node =',l_max)

# Clustering Coefficients
print("Clustering Coefficient of Sample (GLocal)")
cc1_sampled = nx.average_clustering(g_sampled)
print (cc1_sampled)
#print("Clustering Coefficient of Sample (Local)")
cl_sampled = nx.clustering(g_sampled)
#print (cl_sampled)
print ("Reciprocity of Sample")
r = nx.overall_reciprocity(g_sampled)
print(r)
print ("Transitivity of Sample")
t = nx.transitivity(g_sampled)
print(t)

# Social Circles Facebook Dataset Sample
pos = nx.spring_layout(G_sampled)
# Degree Centrality
degCent = nx.degree_centrality(G_sampled)
# Betweenness Centrality
betCent = nx.betweenness_centrality(G_sampled, normalized=True, endpoints=True)
# Eigen vector centrality
eig_cen=nx.eigenvector_centrality(G_sampled,1000000) # 1000000-->precision
key_max = max(eig_cen.keys(), key=(lambda k: eig_cen[k]))
print('Maximum eigen-vector centrality:',eig_cen[key_max],end=" ")
print('at node =',key_max)
# Katz centrality
l=nx.katz_centrality(G_sampled, alpha=0.1, beta=1.0, max_iter=1000, tol=1.0e-6, nstart=None, normalized=True, weight=None)
l_max = max(l.keys(), key=(lambda k: l[k]))
print('Maximum Katz centrality:',l[l_max],end=" ")
print('at node =',l_max)

# Clustering Coefficients
print("Clustering Coefficient of Sample (GLocal)")
cc1_sampled = nx.average_clustering(G_sampled)
print (cc1_sampled)
#print("Clustering Coefficient of Sample (Local)")
cl_sampled = nx.clustering(G_sampled)
#print (cl_sampled)
# Reciprocity and Transitivity
print ("Reciprocity of Facebook Sample")
r = nx.overall_reciprocity(G_sampled)
print(r)

```

```

print ("Transitivity of Facebook Sample")
t = nx.transitivity(G_sampled)
print(t)

# Twitter Dataset Sample
dicta = g_sampled.in_degree()
print("The nodes with maximum in-degree are :")
i = max(nx.in_degree_centrality(g_sampled),key=(nx.in_degree_centrality
(g_sampled)).get)
val = dicta[i]
print("Node    inDegree")
for p,r in dicta:
    if(r == val):
        print(str(p) + "\t" + str(r))
print(" ")
print("List of some nodes with their in-degree:")
w = 1
for p,r in dicta:
    if(w <= 5):
        print(str(p) + "\t" + str(r))
        w = w+1

# Social Circles Facebook Dataset Sample
dict1 = G_sampled.in_degree()
print("The nodes with maximum in-degree are :")
i = max(nx.in_degree_centrality(G_sampled),key=(nx.in_degree_centrality
(G_sampled)).get)
val = dict1[i]

print("Node    inDegree")
for p,r in dict1:
    if(r == val):
        print(str(p) + "\t" + str(r))
print(" ")
print("List of some nodes with their in-degree:")
w = 1
for p,r in dict1:
    if(w <= 5):
        print(str(p) + "\t" + str(r))
        w = w+1

# Twitter Dataset Sample
da = dict()
for xa,ya in nx.degree(g_sampled):
    if(ya not in da):

```

```

        da[ya]=1
    else:
        da[ya]+=1
plt.plot(da.keys(),da.values(),"red")
plt.xlabel("Degrees")
plt.ylabel("Number of Nodes")
plt.title("Degree Distribution")
plt.show()
plt.loglog(da.keys(),da.values(),"purple")
plt.xlabel("Degrees")
plt.ylabel("Number of Nodes")
plt.title("Log Degree Distribution")
plt.show()

# Social Circles facebook Dataset sample
d = dict()
for x,y in nx.degree(G_sampled):
    if(y not in d):
        d[y]=1
    else:
        d[y]+=1
plt.plot(d.keys(),d.values(),"red")
plt.xlabel("Degrees")
plt.ylabel("Number of Nodes")
plt.title("Degree Distribution")
plt.show()
plt.loglog(d.keys(),d.values(),"purple")
plt.xlabel("Degrees")
plt.ylabel("Number of Nodes")
plt.title("Log Degree Distribution")
plt.show()

while(len(remaining_activation)):
    node = remaining_activation[0]
    remaining_activation.remove(node)
    nbrs = G_sampled.neighbors(node)
    for child in nbrs:
        if child not in activated_nodes:
            prob = random.uniform(0,1)
            if prob < 0.2:
                activated_nodes.add(child)
                remaining_activation.append(child)

import pandas as pd
import random

```



```

import numpy as np
import networkx as nx
import math
import matplotlib.colors as mcolors
import matplotlib.pyplot as plt
import random
import copy
%matplotlib inline

G = G_sampled

from collections import defaultdict
def get_percolated_cliques(G, k):
    perc_graph = nx.Graph()
    cliques = [frozenset(c) for c in nx.find_cliques(G) if len(c) >= k]
    perc_graph.add_nodes_from(cliques)

    # First index which nodes are in which cliques
    membership_dict = defaultdict(list)
    for clique in cliques:
        for node in clique:
            membership_dict[node].append(clique)

    # For each clique, see which adjacent cliques percolate
    for clique in cliques:
        for adj_clique in get_adjacent_cliques(clique, membership_dict):
            if len(clique.intersection(adj_clique)) >= (k - 1):
                perc_graph.add_edge(clique, adj_clique)

    # Connected components of clique graph with perc edges
    # are the percolated cliques
    for component in nx.connected_components(perc_graph):
        yield(frozenset.union(*component))

def get_adjacent_cliques(clique, membership_dict):
    adjacent_cliques = set()
    for n in clique:
        for adj_clique in membership_dict[n]:
            if clique != adj_clique:
                adjacent_cliques.add(adj_clique)
    return adjacent_cliques

print("")
results = get_percolated_cliques(G, 4)
# nx.draw_networkx(results)

```

```

def _find_between_community_edges(g, partition):
    edges = dict()
    for (ni, nj) in g.edges():
        ci = partition[ni]
        cj = partition[nj]
        if ci != cj:
            try:
                edges[(ci, cj)] += [(ni, nj)]
            except KeyError:
                edges[(ci, cj)] = [(ni, nj)]
    return edges

def _position_nodes(g, partition, **kwargs):
    communities = dict()
    for node, community in partition.items():
        try:
            communities[community] += [node]
        except KeyError:
            communities[community] = [node]
    pos = dict()
    for ci, nodes in communities.items():
        subgraph = g.subgraph(nodes)
        pos_subgraph = nx.spring_layout(subgraph, **kwargs)
        pos.update(pos_subgraph)
    return pos

```

## SCREENSHOTS

The screenshot shows a Jupyter Notebook interface with the following content:

- File Edit View Insert Runtime Tools Help** (Last edited on December 2)
- Data Preprocessing**
  - Importing Datasets and describing them.
  - Working appropriate centrality measures, clustering coefficients (both local and global) and reciprocity and transitivity.
  - Inferences from the output of above values.
- Description of Datasets:**
  - [Patent citation network] <https://snap.stanford.edu/data/ego-Twitter.html>
  - Dataset information:**

This dataset consists of 'circles' (or 'lists') from Twitter. Twitter data was crawled from public sources. The dataset includes node features

Dataset statistics	
Nodes	81306
Edges	1768149
Nodes in largest WCC	81306 (1.000)
Edges in largest WCC	1768149 (1.000)
Nodes in largest SCC	68413 (0.841)
Edges in largest SCC	1685163 (0.953)
Average clustering coefficient	0.5653
Number of triangles	13082506
Fraction of closed triangles	0.06415
Diameter (longest shortest path)	7

```
SAIN_Project_ipynb
File Edit View Insert Runtime Tools Help Last edited on December 2
+ Code + Text
[ ] Is Bipartite?
False

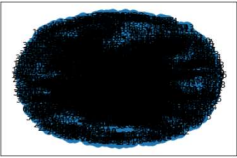
# Sampling Social Circles Facebook Dataset
result = []
f = open('/content/drive/MyDrive/twitter_combined_Sample.txt', "w")
with open('/content/drive/MyDrive/twitter_combined.txt') as fh:
    data = fh.readlines()
    for line in data:
        if line:
            words = line.split()
            result.append(words)
    sample = random.sample(result, 20000)
    for item in sample:
        f.write(item[0] + "\t" + item[1] + "\n")
    print("Sampled Dataset Information:")
    g_sampled = nx.read_edgelist('/content/drive/MyDrive/twitter_combined_Sample.txt', nodetype=int, create_using=nx.DiGraph())
    print(nx.info(g_sampled))

Sampled Dataset Information:
Name:
Type: DiGraph
Number of nodes: 18923
Number of edges: 18627
Average in degree: 0.9844
Average out degree: 0.9844

[ ] from networkx.algorithms import bipartite
print("Is Bipartite? ")
print(bipartite.is_bipartite(g_sampled))

Is Bipartite?
```

```
SAIN_Project_ipynb
File Edit View Insert Runtime Tools Help Last edited on December 2
+ Code + Text
[ ] nx.draw_networkx(g_fb)



# Sampling Social Circles Facebook Dataset
result = []
f = open('/content/drive/MyDrive/facebook_combined_Sample.txt', "w")
with open('/content/drive/MyDrive/facebook_combined.txt') as fh:
    data = fh.readlines()
    for line in data:
        if line:
            words = line.split()
            result.append(words)
    sample = random.sample(result, 20000)
    for item in sample:
        f.write(item[0] + "\t" + item[1] + "\n")
    print("Sampled Dataset Information:")
    g_sampled = nx.read_edgelist('/content/drive/MyDrive/facebook_combined_Sample.txt', nodetype=int, create_using=nx.DiGraph())
    print(nx.info(g_sampled))

Sampled Dataset Information:
```

```
SAIN_Project_ipynb
File Edit View Insert Runtime Tools Help Last edited on December 2
+ Code + Text
[ ] f.write(item[0] + "\t" + item[1] + "\n")
print("Sampled Dataset Information:")
g_sampled = nx.read_edgelist('/content/drive/MyDrive/facebook_combined_Sample.txt', nodetype=int, create_using=nx.DiGraph())
print(nx.info(g_sampled))

Sampled Dataset Information:
Name:
Type: DiGraph
Number of nodes: 3668
Number of edges: 17785
Average in degree: 4.8487
Average out degree: 4.8487

Working on centrality measures, clustering coefficients on samples for both datasets.

# Twitter Dataset Sample
pos = nx.spring_layout(g_sampled)
# Degree Centrality
degCent = nx.degree_centrality(g_sampled)
# Betweenness Centrality
betCent = nx.betweenness_centrality(g_sampled, normalized=True, endpoints=True)
# eigen vector centrality
eig_cen=nx.eigenvector_centrality(g_sampled,1000000) # 1000000-->precision
key_max = max(eig_cen.keys(), key=(lambda k: eig_cen[k]))
print('Maximum eigen-vector centrality:',eig_cen[key_max],end=" ")
print('at node =',key_max)
# Katz centrality
l=nx.katz_centrality(g_sampled, alpha=0.1, beta=1.0, max_iter=1000, tol=1.0e-6, nstart=None, normalized=True, weight=None)
l_max = max(l.keys(), key=(lambda k: l[k]))
print('Maximum Katz centrality:',l[l_max],end=" ")
print('at node =',l_max)
```

```
SAIN_Project_ipynb
File Edit View Insert Runtime Tools Help Last edited on December 2
+ Code + Text
[ ]
print(t)

Maximum eigen-vector centrality: 0.3966795593283609 at node = 34428380
Maximum Katz centrality: 0.09982299190273909 at node = 34428380
Clustering Coefficient of Sample (Global)
0.00324911491381898
Reciprocity of Sample
0.011843876177658143
Transitivity of Sample
0.016179304091578826

# Social Circles Facebook Dataset Sample
pos = nx.spring_layout(G_sampled)
# Degree Centrality
degCent = nx.degree_centrality(G_sampled)
# Betweenness Centrality
betCent = nx.betweenness_centrality(G_sampled, normalized=True, endpoints=True)
# Eigen vector centrality
eig_cen=nx.eigenvector_centrality(G_sampled,1000000) # 1000000-->precision
key_max = max(eig_cen.keys(), key=(lambda k: eig_cen[k]))
print("Maximum eigen-vector centrality:",eig_cen[key_max],end=" ")
print("at node =",key_max)
# Katz centrality
l=nx.katz_centrality(G_sampled, alpha=0.1, beta=1.0, max_iter=1000, tol=1.0e-6, nstart=None, normalized=True, weight=None)
l_max = max(l.keys(), key=(lambda k: l[k]))
print("Maximum Katz centrality:",l[l_max],end=" ")
print("at node =",l_max)

# Clustering Coefficients
print("Clustering Coefficient of Sample (Global)")
ccl_sampled = nx.average_clustering(G_sampled)
print(ccl_sampled)
#print("Clustering Coefficient of Sample (Local)")
```

```
SAIN_Project_ipynb
File Edit View Insert Runtime Tools Help Last edited on December 2
+ Code + Text
[ ]
print(str(p) + "\t" + str(r))
w = w+1

The nodes with maximum in-degree are :
Node inDegree
22462180 60

List of some nodes with their in-degree:
6608332 4
146540920 3
281851529 0
498037630 1
5747582 1

# Social Circles Facebook Dataset Sample
dict1 = G_sampled.in_degree()
print("The nodes with maximum in-degree are :")
i = max(nx.in_degree_centrality(G_sampled),key=(nx.in_degree_centrality(G_sampled)).get)
val = dict1[i]

print("Node inDegree")
for p,r in dict1:
    if(r == val):
        print(str(p) + "\t" + str(r))

print(" ")
print("List of some nodes with their in-degree:")
w = 1
for p,r in dict1:
    if(w < 5):
        print(str(p) + "\t" + str(r))
        w = w+1

The nodes with maximum in-degree are :
Node inDegree
```





SAIN\_Project\_ipynb

File Edit View Insert Runtime Tools Help Last edited on December 2

+ Code + Text

Connect Editing

```
[ ]
Name:
Type: DiGraph
Number of nodes: 3693
Number of edges: 17773
Average in degree: 4.8126
Average out degree: 4.8126
```

An information cascade is defined as a piece of information or decision being cascaded among a set of individuals, where

- individuals are connected by a network and
- individuals are only observing decisions of their immediate neighbors (friends).

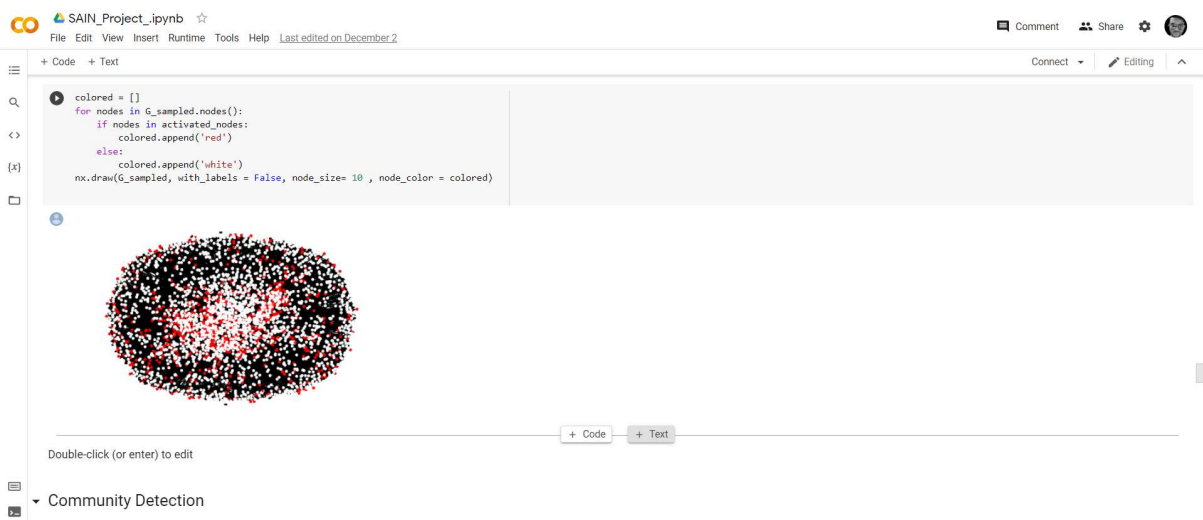
The **independent cascade model (ICM)** that can be utilized to model information cascades. Variants of this model have been discussed in the literature. Below assumptions for this model include the following:

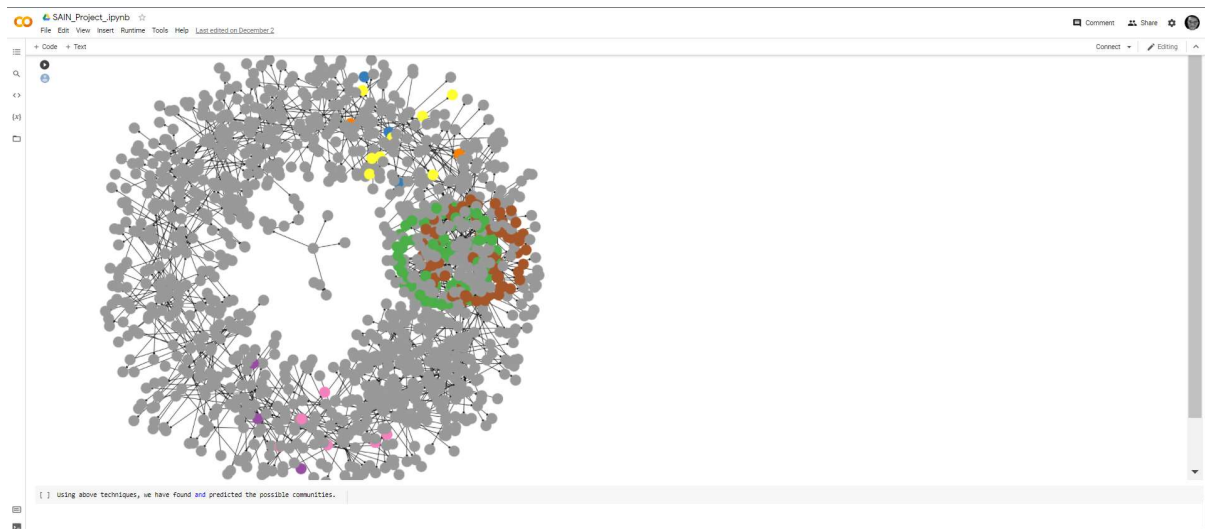
- The network is represented using a directed graph. Nodes are actors and edges depict the communication channels between them. A node can only influence nodes that it is connected to.
- Decisions are binary – nodes can be either active or inactive. An active nodes means that the node decided to adopt the behavior, innovation, or decision.
- A node, once activated, can activate its neighboring nodes.
- Activation is a progressive process, where nodes change from inactive to active, but not vice versa.

Reuirements :

- Diffusion graph  $G(V, E)$ , set of initial activated nodes  $A(0)$ , activation probabilities  $P_{v,w}$

```
initial_activated_nodes = random.randint(1,500)
nodes = list(G.sampled.nodes)
print(initial activated nodes)
```





## **RESULT ANALYSIS AND DISCUSSION**

Twitter has influence over the whole network as compared to twitter because by calculating the extended connections of a node, Eigen Centrality can identify nodes with influence over the whole network, not just those directly connected to it. Facebook has better page rank (most chances of getting a Facebook page while searching) as compared to twitter because this measure uncovers nodes whose influence extends beyond their direct connections into the wider network. We found that factors such as the topic of the query, the phrasing of the query and the time at which a query is issued also impact the bias seen by the users.

We also found that Web search results are typically more favourable for the candidates from the two parties because many of the top results include links to candidate-controlled sources like their own or their party's websites and social media accounts. While we do measure and report the bias introduced by the ranking systems of Twitter and Google search engines, we do not claim that these biases are intentionally added by the platform. To our knowledge, this work presents the first search bias quantification framework which not only quantifies the bias in the output search results but also discerns the contributions of two sources of bias—input data and ranking system. We have applied our framework to investigate the sources of bias for political searches on

Twitter social media and found both input data and the ranking system to be prominent contributors of the final bias seen by the users in the output ranked list of search results. We found that factors such as the topic of the query, the phrasing of the query and the time at which a query is issued also impact the bias seen by the users. We also applied our framework to compare the relative biases of Google Web search and Twitter social media search and found that Web search results are typically more favourable for the candidates from the two parties because many of the top results include links to candidate-controlled sources like their own or their party's websites and social media accounts. While we do measure and report the bias introduced by the ranking systems of Twitter and Google search engines, we do not claim that these biases are intentionally added by the platform.

## **REFERENCES**

- [1] Adamic, L. A., & Glance, N. (2005). The political blogosphere and the 2004 U.S. election: Divided they blog. In Proceedings of the 3rd international workshop on link discovery (pp. 36–43). ACM.
- [2] Cohen, R., & Ruths, D. (2013). Classifying political orientation on twitter: It's not easy!. In Proceedings of AAAI international conference on web & social media, ICWSM 2013, Boston, USA.
- [3] Conover, M., Ratkiewicz, J., Francisco, M., Gonçalves, B., Menczer, F., Flammini, A. (2011b). Political polarization on twitter. In Proceedings of AAAI international conference on web & social media, ICWSM 2011.
- [4] Eslami, M., Vaccaro, K., Karahalios, K., & Hamilton, K. (2017). “Be careful; things can be worse than they appear”: Understanding biased algorithms and users' behavior around them in rating platforms. In Proceedings of AAAI international conference on web & social media, ICWSM 2017 (pp. 62–71).
- [5] Fortunato, S., Flammini, A., Menczer, F., & Vespignani, A. (2006). Topical interests and the mitigation of search engine bias. Proceedings of the National Academy of Sciences (PNAS), 103(34), 12684–12689.
- [6] Garimella, K., De Francisci Morales, G., Gionis, A., & Mathioudakis, M. (2016). Quantifying controversy in social media. In Proceedings of the 9th ACM international conference on web search and data mining, WSDM '16.

[7] Gentzkow, M., & Shapiro, J. (2010). What drives media slant? Evidence from U.S. daily newspapers. *Econometrica*, 78(1), 35–71.

[8] Hannak, A., Sapiezynski, P., Molavi Kakhki, A., Krishnamurthy, B., Lazer, D., Mislove, A., & Wilson, C. (2013). Measuring personalization of web search. In *Proceedings of the 22nd international conference on world wide web, WWW '13* (pp. 527–538). ACM, New York, NY, USA.

[9] Koutra, D., Bennett, P. N., & Horvitz, E. (2015). Events and controversies: Influences of a shocking news event on information seeking. In *Proceedings of the 24th international conference on world wide web, WWW '15, International world wide web conferences steering committee, Republic and Canton of Geneva, Switzerland* (pp. 614–624).

[10] Kulshrestha, J., Eslami, M., Messias, J., Zafar, M. B., Ghosh, S., Gummadi, K. P., & Karahalios, K. (2017). Quantifying search bias: Investigating sources of bias for political searches in social media. In *Proceedings of the 2017 ACM conference on computer supported cooperative work and social computing, CSCW '17* (pp. 417–432). ACM, New York, NY, USA.