Fake News data wrangling

Building a dataset containing verified real and fake news

Austin Wilson†
Computer Science and Statistics
California State University of Sacramento
Sacramento, CA US
austin1224@gmail.com

Talib Khwelled
Computer Science
California State University of Sacramento
Sacramento, CA US
tk@csus.edu

ABSTRACT

The overall goal of this project is to build a reliable data set of news. We would like to extend the Fake News Net project. By doing so we would also like to make this code base more user friendly. The news will be both political data and other general news. There are two categories: either fake news or real verified news. We have used several tools in order to create this dataset. The first tool we utilize is the programming language python along with the libraries: pandas, numpy and the twitter API tweepy . We have also used the code base from the project called FakeNewsNet. FakeNewsNet is a tool developed for the purpose of creating reliable datasets for fake and real news. This tool gave us access to twitter data associated with news stories and a reliable classification of fake or real. Once we ran this tool we had to use the twitter API to gather the text data associated with each tweet id to build the final dataset. This dataset can be used in the future to train machine learning models to predict if a piece of news is fake or real.

1 Fake News Net

The first part of this project was using the Fake News Net project to get several datasets. We ran the tool to gather datasets titled "gossipcop_fake.csv", "gossipcop_real.csv", "politifact_fake.csv" and "politifact_real.csv". These datasets are news stories and associated tweet ids from the websites gossipcom.com and politifact.com. Gossipcop.com is a website that fact checks celebrity reporting. Politifact.com is a similar website that fact checks claims by elected officials. By using this tool we were able to gather 432 fake news political stories, 624 real news political stories, 4,898 fake celebrity stories and 15,747 real celebrity stories. After dropping the rows with NA values we ended up with 389 fake news political stories, 373 real news political stories and the same amount of fake/real celebrity stories. It seems that gossipcop.com had more reliable data.

2 Fake News Net Plus

From this point I began to implement my own functions to gather the real text data from twitter. I began by importing the data gathered from gossipcop.com and politifact.com into a python jupyter notebook. After checking the size of the data sets and dropping the NA values I inspected the columns. Each data set has four columns. The columns are the same for gossipcop and politifact datasets.

2.1 Data from Fake News Net

The columns in the dataset imported from Fake News Net are "id", "news_url", "title" and "tweet_ids". The id is in the form politifact<<"number">> or gossipcop-<<"number">>, respectively. This is an id associated with a particular news story. The next column is titled "news_url". This contains a link to the url for the original story. For many of the rows, the url is outdated and the webpage no longer exists. This is probably because news websites don't keep old stories public. The title column is self explnitory. The last column is tweet_ids. This is a string that contains all the tweets ids associated with each particular news story.

2.2 Preparing data for twitter API

The ids are tab separated so I began by converting these values into a list to be passed to the the twitter API. Since we have a list of tweet ids associated with an id, title and url I decided that I should create a new dataframe with each tweet id, username for the tweet, the attribute "in_reply_to_status__id_str" which indicates if the tweet is a response to another tweet, the text from the tweet, the title associated with the tweet, the url associated with the tweet and the id associated with the news story. This approach will result in a dataframe that has quite a bit of repeated data. To be exact, the id (not tweet id), url and title will be repeated for each individual tweet that corresponds to a single news story.

2.3 twitter API

Because we are working with tweet ids, we can make use of the tweepy method "statuses_lookup". This method will accept an array containing up to 100 (no more than 100) tweet ids. This caused a problem because many of the news stories have more than 100 tweets associated with them. We also wanted to give users of this project the option to request only a subset of the tweets from the associated gossipcop or politifact news story. As of writing this report I have not found a way for the user to select a subset of a particular story. Since I am aggregating the data for the whole politifact and gossipcop dataset I was only able to allow the user to limit the number of tweets gathered for the entire dataset. I will elaborate in the following sections.

2.3.1 twitter API credentials

Before using the twitter API you must register as a twitter developer. I will skip the details of this process as it is relatively straightforward. Once you have registered, use the twitter developer website to generate your credentials. You need consumer key, consumer secret, access token and access token secret. Once you have these credentials you can connect to the twitter API and begin fetching data.

2.3.2 get_tweets_small

This function is designed to gather the twitter data for news stories that have less than 100 tweets associated with them. I would have liked to write a single function that could be used in both cases but I was unable to do this successfully and found it more straightforward to separate these requests into small and large for less than or more than 100 tweets, respectively. The parameters for this function are row and number of tweets. Row represents a single row from the politifact or gossipcop data frame. The reason for using a single row is that a row will be passed to this function iteratively in the aggregate_data function later. The number of tweets parameter has the default value False but will contain an integer if the user wants to specify a smaller number of tweets. The next part of the function I descructure the row data by attribute and initialize empty lists to store the twitter API data collected. Again some of these list will simply contain repeated data. Specifically news_id_list, news_url_list and title_list will all contain repeated values. This is done because I am trying to store the unique tweet data associated with each news story in a separate row. There are many tweets associated with a single url, title and news id. The data can later be aggregated by these values so that each story only has one row for all the text. I have not done this part. Next I pass the array of tweets to the twitter API via statuses_lookup and store the response in the result variable. The result object contains a dictionary with tweet data for each tweet id passed to the API. Next I loop through all of the tweets and append id, screen name, text, "in_reply_to_status_id_str", "news_id", "news_url" and title to the lists initialized in the beginning of the function. Once this process is complete I create a dictionary containing a label for each list and the list itself. This dictionary contains the news id, news URL, title, username, text and reply data for all the tweets associated with each news story. Finally, the function returns a data frame created from this dictionary.

2.3.3 get_tweets_large

This function is synonomous with "get_tweets_small" function but is designed to request more than 100 tweets. Since most of the function is the same as "get_tweets_small" I will only elaborate on the code which traverses an array of more than 100 tweets. First of all, the user may want to subset the tweets by a number smaller than 100 we can use "get_tweets_small", which has been written for this and will simply make one request. Now in the case that the user wants to subset by a number larger than 100 we simply need to reassign "row_data" to the subset of "row_data" specified by number of rows. Note that the "+1" is so that the last row is included. Now in order to traverse an array with more than 100 rows by 100 rows at a time, we need to calculated the length of the array integer divided by 100. This will give us the number of subsets of length less than 100 that the array can be divided into. For example if an array has 230 tweet ids, then 230 // 100 gives us 2. This is what we need to traverse the array 100 elements at a time. I then use the range function to do this. Since range starts from 0 in our example range(2) will have 3 iterations. This will allow us to iterate through an arbitrarily long array elements at a time. Then I take a 100 tweet id subset of the tweet id and make a request. The same process that I used for the "get_small_tweets" function is followed. I traverse through the list one tweet at a time and add the associated data to a list which was initialized in the function. I do this with the array of tweets 100 elements at a time until all the data has been collected. Then these lists are stored as a dictionary with their titles and return as a data frame.

2.3.3 aggregate_data

The purpose of the "aggregate_data" function is to create a new data frame which has the data from every single tweet id found in the datasets from Fake News Net. The parameters are df (standard notation for data frame), name and number of tweets. The df to be passed in is the dataset gathered from Fake News Net. The name parameter has the default value "test.csv" for testing purposes. This is the name that the dataframe will have once it is converted to a csv file at the end of the function. The number of tweets parameter will effectively subset all of the rows tweet lists by whatever number is passed. If the user only wants 10 tweets per news story then 10 should be passed as number of tweets. Then the stories that have more than 10

tweets associated with them will only include the first 10 tweets. The news stories that have more than 10 tweets will not be affected. First, I initialize an empty data frame. Then I traverse all the row in the data frame passed into the function. Since "iterrows()" returns a tuple containing the index of the current row and the row we must use "row = row[1]" to get the actual row data. Then we use "get_tweets_small" or "get_tweets_large" depending on the number of tweet ids in the row to retrieve the data from the twitter API as described previously. Then I append the data frame returned from these functions to the empty initialized data frame. Once this process is done for all the rows in the data frame, I convert the resulting data frame to csv and return it at the end of the function.

3 Case study

After running "aggregate_data" four times we are left with four corresponding datasets. We have both fake and real data for gossipcop and politifact. Each dataset has 7 columns. News id column is the id that corresponds to a unique news story. There are many tweets associated with each news id. News URL and title are also associated with a single news story for many tweets. Then we have tweet id which was used by the twitter API to get the data in the first place. We also have username, text and reply. These are self-explanatory. Reply is a Boolean value where a 1 indicates that the tweet is a response and 0 indicates it is not a response. The fake data for politifact has 121,114 rows of data. The real data for politifact has 306,575 rows. The fake data for gossipcop has 436,913 rows. The politifact fake news data has an average of 424 tweets per news story. The politifact fact real news data has an average of 1064 tweets per news story. The gossipcop fake news data has an average of 117 tweets per news stop. The gossipcop real news data has an average of 55 tweets per news story.

3.1 Politifact fake news examples

I randomly chose two specific rows to look at the data. I used index 0 and 20000. The title for the tweet at index 0 is "BREAKING: First NFL Team Declares Bankruptcy Over Kneeling Thugs". This is obviously (to me) fake news. No NFL teams have gone bankrupt in recent history. I also looked at the actual text data for this tweet an found interesting data. The text was the same as the title with the additional spam "-ChristmasGifts https://t.co/WkwXW7pGST\n\nWriter h... https://t.co/tebSFM2hLo. This clearly just a garbage repost. When I looked at the other data associated with this news story I found a very similar trend. The title is included in the tweet test along with a hyperlink. I tried to visit these links unsuccessfully. When I looked at the data around the 20000th index I found that the text data contains data different than the title. The title for the 20,000th row (and the rows nearby) was "The Legislative Process". This is not a very descriptive title. I found that this title is associated with 22,406 tweets. It is hard to distinguish what this fake news story is. The titles of the tweets all had something to do with the US house of representatives. Other titles talked about the democrats taking over the house of representatives. Some of them talk about a government shutdown. Either way this data is not very clear.

3.2 Politifact real news examples

After looking at several different news stories I chose one titled "Donald Trump exaggerates when he says China has 'total control' over North Korea." I found it interesting that Donald Trump showed up in the real news dataset. There are 26 tweets associated with this news article. Oddly enough I found that the text data for these tweets was almost identical to the title of the news stories. These could possibly be retweets. The next news article I selected was titled "School kids taught to praise Obama". After looking at several of the text values associated with this story, I found again that the tweet text was very similar to the story title. The story has 1,126 tweets associated with it.

3.3 Gossipcop fake news examples

The first fake news story I selected was titled "Miley Cyrus Claims 'Satan Is A Nice Guy; He's Misunderstood". This seems obviously fake to me, but it has 76 tweets associated with it anyway. The text data is very similar to the title. One tweet also has the text "The reptilius at work!". This is likely a real user not a bot. Reptilus seems to have no actual meaning other than referencing an old film. The next example I chose was titled "'Stranger Things' Creators Go Inside Seasons 2's Heartbreaking Death". I choose this because I have seen the show and I know this is not true. I also know that (spoiler alert) this does happen in a later season. When I looked at the text data for the tweets associated with this news story, I found some very interesting variation. First of all, some of the text data was in German. Perhaps the original post was from Germany even though this is an American show. The tweet I found most interesting is "will and eleven are gonna die in season 3 stranger things, remember this tweet". This is clearly an actual user (not a bot) who knows this is fake news. Although what this user predicts does not happen, something similar does happen.

3.4 Gossipcop real news examples

The first example I selected a news story titled "Anne Heche and James Tupper split after 10 years together". This news story had 50 tweets associated with it. I found that the text in this dataset was all exactly the same as the title. Some of the tweets had additional URLs. For my second example, I chose a news story titled "Gweneth Paltrow reveals how she broke her foot on Jimmy Kimmel Live". I thought this was interesting

because depending on how you read the sentence it could mean two different things. Did reveal how she broke her foot, or did she break her foot on live TV? Presumably she did not break her foot on live TV. There are 93 tweets associated with this news story. I looked at the text data with the tweets and found similar uniformity. All of the text data contained the news story title, verbatim, some with additional information.

4 Results and Reflection

Overall, the project was a success. I was able to build a dataset that has reliable data for real and fake news. This data can be used to train a machine learning model to make predictions about news in the future. One important observation is that most of the fake news data is repeated. The real news data is not repeated. Typically, the fake news tweet text value is the same as the title of the news story. This makes our data set less useful because we don't have variety in the data. This may be caused by bots on twitter reposting fake news. The real news tweets have quite a bit of variation.

** add references **