Final Report **Kenneth Sanders**
COMP 4800-011 8/23/17
Prof. Rumshisky

Abstract

Hashtags on Twitter are an easy and simple way of relating multiple topics to the contents of a tweet. These simple labels also act as fantastic markers for conflict and disagreement through a variety of topics. Hashtags add another layer of relation between tweets other than their semantic meaning, adding to the complexity of the social network. By finding a way to represent the interactions of hashtags on a large scale, we may further understand the spread of opinion and conflict on Twitter. One major way of representing the relevancy and conflict between hashtags is through generation of reliable clusters based on conflicting arguments. This paper aims to explore the clustering of hashtags through network and vector space analysis.

Previous Work

Much work has been done previously on analysis of word and sentence meaning in order to understand and analyze conflict in political and social issues. Additionally, community detection and conflict modeling have been conducted using network and cluster analysis. The following papers are some of the more relevant works.

Distributed Representations of Words and Phrases and Their Compositionality

This paper by Mikolov et al. details the Word2Vec algorithm and the theory behind it, which may be used to help represent the semantic meaning of each individual element of a tweet in the context of tweets containing a hashtag. This is done through the use of a simple neural net and specialized training of that neural net using a skip-gram model. Word2Vec is the basis for the embedding approach to hashtag clustering focused on in this project[14].

Distributed Representations of Sentences and Documents

This paper by Quoc V. Le and Tomas Mikolov expands upon the Word2Vec algorithm in order to form vector representations of sentences and paragraphs, and describes the Paragraph2Vec algorithm. This creation of paragraph embeddings may be used to generate more accurate representations of the tweets and the corpus of text used with hashtags, ideally providing accurate embeddings from which hashtag clusters may be gleaned[9].

Political Polarization on Twitter

Conover et al. discuss interactions on Twitter and how to extract a measurement of polarization by analyzing social network structure, especially in a political environment. This network analysis is similar to the co-occurrence networks that this paper proposes. The measures of polarization using these networks may also be useful when analyzing the effectiveness of the resulting clusters from this project[3].

Dynamic Debates: An Analysis of Group Polarization over Time on Twitter

This paper by Sarita Yardi and Danah Boyd is an analysis of debates on Twitter and how group theory may be used to interpret them. While this project does not focus on group theory some of the models of Twitter as a social network still applies. The dynamics of interactions within and outside of one's own opinion groups are analyzed and shown to strengthen and spread group opinions respectively[20].

Community Identity and User Engagement in a Multi-Community Landscape

Zhang et al. describe many community detection measures using a text corpus to represent each one. Some of these measures may be useful in accounting for the importance, relevance, and distinctions between groups. Since it is hypothesized that separate groups use separate hashtags, the detection of communities and the relevence of a hashtag to the group is certainly important to the focus of this project[21].

Hypothesis

The co-occurrence of hashtags within individual and sets of user tweets can be used to create affinity graphs representing the relationships between hashtags. Hashtags may also be compared using specialized application of Word2Vec to generate embedding representations of hashtags in vector space. Both the affinity graphs and vector space representations may be used to generate clusters of closely related hashtags. Analysis of these hashtag clusters may provide a measure of controversy between the topics that the hashtags focus on, such as euclidean distance and entropy between clusters.

Theory

There are two veins of interest covered in this paper: weighted networks based on hashtag co-occurrence, and the creation of hashtag embeddings through a speciallized sumation of word embeddings. Hashtag co-occurrence networks themselves may be analyzed in three parts: co-occurrence within single tweets, co-occurrence within sets of user tweets, and a combination of the two. The theories behind each of these methods are described below.

Hashtag Co-occurrence Within Tweets

A network may be constructed between hashtags using their co-occurrence within the same tweet as a basis for edges. This tweet co-occurrence network may be represented as an undirected weighted graph with vertices H and edges T. Each vertex represents a single unique hashtag that was used in our data set. The existence of each edge represents the fact that the two hashtags it connects occurred in the same tweet atleast one or more times. The edges themselves may be weighted using the number of times each of the linked hashtags occur in the same tweet.

This graph can be viewed as an affinity matrix M. In more formal terms, each vertex in the graph is mapped to the integer range $[0, n)$ where n is the number of vertices. The matrix M is constructed by assigning each element $M_{i,j}$ the weight for the respective edge between the two verticies mapped to i and j. In other words, each element $M_{i,j} = T_{A,B}$, where $i, j \in [0, n)$, $A, B \in H$, and edge $T_{A,B}$ is the weight of the edge that connects hashtags A and B.

Additionally, a transition matrix may be easily formed from the affinity matrix M. This can be done simply by dividing each element in a column by the sum of the column in M. Formally, given affinity matrix M, to construct transmission matrix T, we may set

$$T_{i,j} = \frac{M_{i,j}}{\sum_{k=0}^{n} M_{k,j}}$$

where n is the number of columns (and rows) in M, and i and j is the respective row number and column number of the element being constructed. This transition matrix may be used for random walk emulation, which Markov clustering uses.

Hashtag Co-occurrence Within Users

Similar to hashtag co-occurrence within tweets, a network may be constructed between hashtags using the number of co-occurrences of the hashtags within the same user's group of tweets (in the data set) as a weight for the edges. Therefore the network can again be represented as an undirected weighted graph with vertices H and edges U. As before, each vertex in H represents a unique hashtag. However, each edge weight represents the number of users that used the two hashtags connected by the edge in our data set. Formally, the weight of edge $e_{h_1,h_2} \in U$ that connects $h_1, h_2 \in H$ has a weight equal to the number of users that used both h_1 and h_2 at any point in the data set. This network may then be stored in the form of an affinity matrix or a transition matrix, using the same formula described for hashtag co-occurrence within tweets.

Hashtag Co-occurrence Within Users and Tweets

We may combine the two previous co-occurrence networks with weights in an attempt to infer more information between the hashtags. For instance, having co-occurrence within tweets be weighted 3 times as much as the co-occurrence within a user since hashtags used in the same tweet are more likely to be closely related than hashtags used by the same user. This also allows us to take advantage of both the benefits of the two

methods. This is sensible since the tweet co-occurrence graph is fairly sparse, but very accurate in relevance, and user co-occurrence is fairly dense, but not very accurate. The resulting network from both methods combined is both fairly dense and accurate.

More formally, assuming a network $N_U = \{V_U, E_U\}$ has been created to represent user co-occurrence, and network $N_T = \{V_T, E_T\}$ has been constructed for tweet co-occurrence. Using the theory described in the previous two sections, we may combine those networks in the following way. We also will define a new network $N_B = \{V_B, E_B\}$ which will come to represent a combination of the two previous networks. For any two hashtags (represented as nodes) in the net h_1 and h_2 such that $h_1, h_2 \in V_T$ and $h_1, h_2 \in V_U$, we may construct an edge $e_{h_1,h_2} \in E_B$, such that $e_{h_1,h_2} = W_U * f_{h_1,h_2} + W_T * g_{h_1,h_2}$, where $f_{h_1,h_2} \in E_U$, $g_{h_1,h_2} \in E_t$, and W_U, W_T are the weights for user and tweet co-occurrence respectively. These weights can be used to adjust the influence from user and tweet co-occurrence. Since gathering methods differ in the way they accumulate the tweets, some sets may have a high number of tweets from a few individual users, or vice versa. The weights allow us to adjust how much the network favors the two different co-occurrence possibilities. It should also be noted that tweet co-occurrences are naturally a subset of user co-occurrences when calculating ideal weights. To this extent, it is sensible to assign a constant unit weight to W_U, and adjust the tweet weight W_T to reflect the ratio of average tweets per user. For our data, an average of 3 tweets per user is approximated, meaning that $W_U = 1$ and $W_T = 2$, since the final weight for any tweet co-occurrence will be approximately $W_T + W_U = 3$.

This new network N_B may then be transformed into an affinity matrix, transition matrix, and clustered using the same methods as the two networks it is derived from.

Hashtag Embeddings

This method assumes the collection of a large number of tweets and their associated hashtags. They will be used to create a graph representing the similarity relationship between each hashtag on a scale of -1 to 1 (-1 is high dissimilarity, 1 is high similarity). Given this set of tweets, each tweet will contain text and a set of zero or more hashtags that were used with the tweet. Each tweet is of course made up of a set of words. Additionally, we may model a hashtag as being made up of a set of tweets that users ascribed with the hashtag.

The words can be modeled as vectors called word embeddings by training a shallow neural net with target-context pairs (a pair of words that co-occur in the same tweet). This procedure is well documented as Word2Vec by Mikolov et al. [14], and a slightly modified algorithm produces consistent and accurate word embeddings for our data set. These embeddings allow us to use cosine similarity (the dot product of two vectors divided by the product of their magnitudes) to estimate the similarity between the two words that the embeddings represent. The goal of this is to bring the same functionality to hashtags by creating embeddings to represent them well.

In theory, we may sum the embedding for every word that is used in a tweet to make a vector representing a tweet embedding, and then sum the embeddings of all the tweets that are used with any given hashtag to create an embedding for that hashtag. Additionally, we may weight each embedding before summing so that tweets and words more relevant to the hashtag will have greater impact than the ones that are not relevant.

More formally, we are given a vocabulary W and set of embeddings $E_W = \{E_w \text{ for } w \in W\}$ For any hashtag h, we can say there exists a set of tweets T_h in which the hashtag was used ($\forall t \in T_h$, h was used in t). We may also say that every tweet $t \in T$, $t = [w_1, w_2 ... w_n], w_i \in W$. We may attempt to construct a tweet embedding in a similar way to Le and Mikolov's Paragraph2Vec model [9] by summing the word embedding of each word used in a tweet t. In terms of w, h, and t, this is

$$E_t = \sum_{w \in t} F_h(E_w)$$

where F_h is a function that attempts to apply a scalar weight to word embedding E_w based on the importance of the word w in tweet t. In this model, specificity [21] is used to measure how specific a word is when used with a hashtag compared to when it is used normally in the data set. For instance, one would expect "defund" to have a relatively high specificity in regards to the hashtag "#defundpp". In other words, since "defund" is very likely to be used with "#defundpp", the word embedding for "defund" should have a greater scalar multiple in order for the "#defundpp" embedding to be skewed in its direction. In contrast, the word "prolife" would be expected to have low specificity in relation to the hashtag "#prochoice", since they represent opposite stances on the topic.

Once we have a set of tweet embeddings $E_T = \{E_t \text{ for } t \in T\}$, we may construct a hashtag embedding

$$E_h = \sum_{t \in T_h} G_h(E_t)$$

where $T_h \subseteq T$. In this case, weight function G_h applies a scalar weight to the tweet embedding E_t based on its relevance to the hashtag h. One form of this weight could be the consistency of the tweet creator's vocabulary in relation to the current tweet, which may be a primitive measurement of a user's authority on the topic. In the project's current form however, this weight is not used, instead opting for a function that reduces each vector to a unit vector (magnitude = 1) in the same direction as the original vector.

Clustering Algorithms

A number of clustering algorithms were tested using the resulting networks and points in vector space from the 4 procedures this paper outlines. The clustering algorithms are breifly discussed below.

Affinity Propagation

This is a clustering algorithm for affinity matrices, which represent a network with affinity between the vertices as elements in the matrix. The affinity propagation method uses the idea of message passing throughout the network. The algorithm uses a set of exemplar points for that the other points may use as their exemplar point, which succinctly results in a node identifying its own cluster. Then, messages are passed between the nodes relaying the "responsibility" and "availability" of an individual node to be an exemplar. This "responsibility" represents how well suited a point is to be an exemplar based on an each individual node's potential exemplars. The "availability" measure

represents how well suited a point is to have another point as an exemplar based on other node's preference of exemplar. This essentially boils down to clustering via representative samples for each cluster, and works fairly well for our application since the clusters for each side of a conflict should have some rather accurate examples as to the central topic of the argument [7].

Markov Clustering

This clustering algorithm is also used on weighted graphs, however the affinity matrix must be transformed into a transition matrix by normalizing the columns such that each value in a column represents the probability of a transition from the vertex that the column represents to the vertex that the row represents. Markov clustering focuses on the idea that the flow between nodes in the same cluster will be high, while the flow between two separate clusters will be low. To do so, the algorithm uses a two step process for each iteration, which are known as "inflation" and "expansion". Expansion is simply applying the matrix power operation to the transition matrix, then returning the resulting square matrix to a transition matrix by normalizing the columns. Inflation is the process of taking a non-negative power of each of the elements in a single column, then re-normalizing that column. The inflation operator emphasizes the weakness or strength of the currents between nodes, while the expansion operator allows flow from different parts of the graph. Executing inflation and expansion repeatedly will ideally converge to a matrix with the only non-zero edges between nodes in the same cluster [5].

K-means

The K-means algorithm is one of the most simple and straight-forward clustering algorithms. It may only be used for vector space graphs however, since the calculation of centroids is only possible in vector space. K-means requires the number of desired clusters as a parameter, since this determines how many centroids to generate. Centroids are generated initially as inexact guesses, and then refined using an error function in order to reduce the average variance of all the points from their nearest centroid [8]. K-means also be expanded to X-means. In X-means the number of clusters is not predefined. Instead the algorithm uses the bayesian information criterion on multiple iterations of K-means to determine the number of clusters that reduces the variance the most [17].

Data Set

The data set used is a collection of over 2,100,000 tweets and 23,000 hashtags that were gathered by the Text Machine Lab at UMass Lowell under the supervision of Prof. Anna Rumshisky. The data is rather raw, and was gathered through the use of keywords on the topic of abortion through the Twitter API [19]. However, the neutrality of the data is not guaranteed, since there may be underlying functionality in Twitter's keyword search that is not publicly disclosed.

Database

Since a relatively large amount of data (2,114,926 tweets) is being processed in $O(n^2)$ time, there is a need to store the product of each step of the process in order to make improvements to the algorithm and allow for analysis of intermediate data, such as the word embeddings and the vocabulary dictionary. Therefore, intermediate data is stored in PSQL tables by its producer, and accessed by its consumer accordingly. A default configuration of each table needed in the database is included in the code base, and schemas may be used to create multiple intermediate and final data sets.

Methodology

Each theorized method of hashtag clustering must be modeled and carried out on a computer. For this, a combination of Python and PSQL are used in order to efficiently and easily manipulate the data and structures needed for these procedures. The following are descriptions of the code that is used in order to carry these procedures out.

Hashtag Co-occurrence Within Tweets

The creation of clusters of hashtags based on their co-occurrence is fairly straightforward. There are only two steps: construction and clustering of the network. Construction of the network simply involves creation of a mapping from hashtags to a list of tweet IDs that they were used in. From this, the weights of edges between two vertices in the network can be found by counting the number of shared tweet ID s between the two lists that each hashtag maps to. This network can be represented as a square affinity matrix, which can then be clustered using graph clustering algorithms such as affinity propagation [7] or Markov cluster analysis [5]. Since the graph created generally has multiple small unconnected sub-graphs due to the relative sparsity of this data set, the largest connected sub-graph is separated and used for the purposes of clustering.

Hashtag Co-occurrence Within Users

Similar to clustering of the network of co-occurrence of hashtags used in the same tweets, we may also cluster hashtags based on their use by the same user. The same methodology is applied, except hashtags are mapped to lists of users that the hashtags were used by. The network is then created between hashtags with weights of edges as the number of users that used both hashtags that the edge connects. This is again transformed to an affinity matrix, and then clustered using the same algorithms as for the tweet version of the network.

Hashtag Co-occurrence Within Users and Tweets

As noted in the theory for this procedure, it is simply a combination of the two previous procedures. In fact, this method may simply use the hashtag co-occurrence within tweets, with an added step. Once the weighted undirected graph for user co-occurrence is created, a list of hashtags for each tweet in the data set is formed. Every possible combination

of two hashtags in each of these lists is generated, and if both hashtags are present in the network, the number of times two hashtags co-occur in a tweet is multiplied by the tweet weight (W_t) and added to the respective edge in the user co-occurrence graph. W_T represents the weight that tweet co-occurrences have compared to the user co-occurrences, as described in the theory section. This network is then expressed in matrix form, and clustered using the previously noted graph clustering algorithms.

Hashtag Embeddings

The creation of word embeddings and subsequent hashtag embeddings is a multi-step process, which can be divided into functions and run by a single main function. Each of these steps is described below.

Splitting and Stemming

Initially, a tweet is a string of up to 140 characters and a unique long integer ID. To be able to create word embeddings, the tweet text must be split into individual words. The Natural Language Tool Kit (NLTK)[10] Python library offers a fairly comprehensive tweet tokenizer, which simply returns an array of strings representing words and symbols. Next, stop words and certain symbols are removed, and a snowball stemming algorithm from NLTK is applied to each word in a tweet in order to reduce each word to its root. Once the text of each tweet is tokenized, filtered and stemmed, the array of strings is inserted into a formatted-tweets table in the database using the tweet ID as an index.

Constructing the Dictionary

Once each tweet is reduced to an array of stemmed words, we can iterate over all the words of all the tweets in the data set and count the number of occurrences of each word. The most common words are mapped to unique integer IDs and inserted into a dictionary table in the database. The number of occurrences of each word is also included in the dictionary table, as it will be useful in generating hashtag embeddings. Any words that are not a part of those added to the database are considered to be unknown and are assigned the ID '0' for the next step.

Integerize Tweets

Once the dictionary table is created, the process of converting the tweets from arrays of strings to an arrays of integers begins. This is done by searching for each word and its corresponding word ID in the dictionary. If the word is not found, it is replaced with '0' instead of a positive integer ID. The tweet IDs and integer arrays are then inserted into an integer-tweets table in the database.

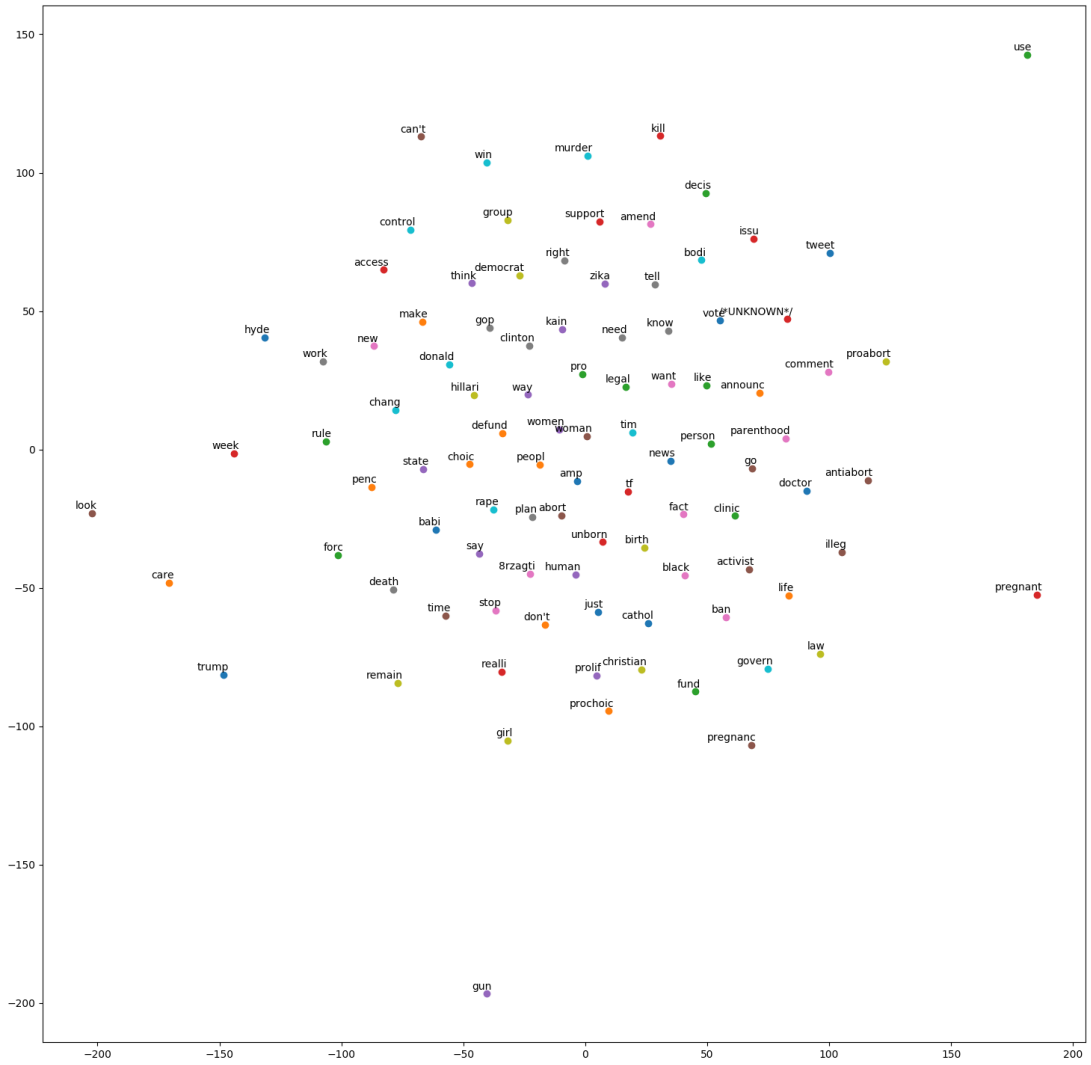


Figure 1: The 100 most common word roots graphed by their dimension-reduced embeddings.

Creating Word Embeddings

Generating word embeddings once the tweets are integerized and the dictionary is created is fairly straight-forward. This is outlined by Mikolov et al. [14] and is easily implemented using the Tensorflow library [1]. We use the skip-gram model of Word2Vec since it scales well with the varying sizes of data sets and does not treat the entire sentence as the context. In our model, each tweet is modeled as a sentence in the training for the neural net, meaning that each target-context pair is tweet specific, which helps maintain the sentiment of each individual tweet. Figure 1 shows the embeddings of the top 100 most common words in the data set. In theory the distance should represent the similarity of the words to each other. However, since the embeddings are reduced from 128 dimensions to 2, this may not always be accurate.

Generating Hashtag Embeddings

After the word embeddings are created, they may be combined into tweet embeddings, and then hashtag embeddings. The tweet embeddings created are temporary, allowing

the specificity of the word to the hashtag in relation to the whole data set to be used as a weight when building the hashtag embedding. Therefore the tweet embeddings are not stored in the PSQL database, as they are hashtag specific, and the required space would be many times the size of the tweet database. Once the tweet embeddings for a hashtag are created, they are unitized (reduced to magnitude = 1 in the same direction) in order to ensure that each tweet begins with the same weight. In the current model, the unitized tweet embedding for each tweet in a hashtag set is simply summed together with no weight to create the hashtag embedding. Therefore, these weights have the potential to improve results easily, assuming an accurate and consistent weighting method can be found. However, creating a weighting function is a difficult task for another paper.

Creating a Hashtag Graph

Once hashtag embeddings are created, we may calculate the cosine similarity between each possible pair of embeddings in order to get the relationship between each pair of hashtags. These relationships can be expressed as a square matrix or a list of n-dimensional points, where n = the number of hashtags. This hashtag graph allows each hashtag to act as a dimension, and the relationship it has to all the other tweets will be the value of that dimension in all the other hashtags in the graph.

Clustering and Graph Analysis

After a hashtag graph is created, we may attempt to analyze the accuracy of the embeddings for use in polarization analysis. The graph is clustered using k-means with k=2, or with X-means. Once the clusters are generated, the within-cluster standard deviation is calculated and the hashtags that belong to each cluster are recorded, along with the size of cluster, standard deviations, and distance between centroids.

Results

The methodologies for each of the procedures were conducted using the abortion data set with a variety of different parameters. These were then compared to a training set of annotated hashtags, allowing for a comparison of ideal clustering with the procedure used. The best parameters for each method were then used to compare to a test set of annotated hashtags completely different from the training set. This allowed for an objective analysis of the results using both entropy measurements and human-conducted comparison of clusters.

Annotated Hashtags

In order to properly analyze the results of the project, a set of annotated hashtags was created. This set includes over 300 hashtags that were deemed to be obviously and intentionally in favor of either pro-choice or pro-life arguments, and were hand selected and labeled out of over 10,000 hashtags. The hashtags gathered were intended to be chosen using unbiased knowledge of the situation. However, in the interest of full disclosure,

it should be noted that the annotator's personal opinion on the matter is pro-choice. Figure 2 shows 50 hashtags from each annotated set.

Pro-Life Hashtags	defundpp praytoendabortion prolife lifefirst ppsellsbabyparts ppvideos chooselife defundplannedparenthood prolifeforalllife stoptheslaughter prolifegen stopabortion babieslivesmatter iamprolife proadoption unbornlivesmatter abortionismurder lifebeginsatconception endabortion adoptionworks prolifeangels prolifeacrossam respectforlife alternativetoabortion prolifeyouth kcpfisprolife noabortion abolishabortion prochoiceisalie prolifewomen prodoption stopabortionnow prolifewomen16 prayfortheunborn endabortionnow abortionhorrors voteprolife abortionalternatives abortionregret feticide standforlife txloveslife supportnotabort righttolife protestpp banabortion prolifeisprowoman prolifewomem16 abortionisunamerican prolifewomen2016 notachoice
Pro-Choice Hashtags	antichoice prochoice womensrights mybodymychoice womenshealth standwithpp sexeducation birthcontrol abortionaccess contraception prochoix reproductivehealth reprorights reprohealth reproductiverights womenrights righttochoose istandwithpp shoutyourabortion keepclinicsopen safeaccess voteprochoice keepabortionsafe abortionisnotacrime ppnycpride waronwomen contraceptives abortionstigma safeabortion wholewomenshealth mybodymyrights wholewomanshealth clinicescort awomansrighttochoose perpetuatingantichoice perpetuatingantichoiceideas myabortionmylife prowoman nouterusnoopinion abortionpositive maternalhealth stickers4choice knickersforchoice legalabortion nowirehangers womenhealth fundpp womenschoice womansrights workforpp womanrights

Figure 2: A sample of 50 hashtags from each category of the annotated hashtags.

Co-occurrence Clustering

The annotated hashtags may be used to compare the effectiveness of different clustering and co-occurrence techniques. This is done by calculating the entropy of the resultant clusters. More formally, the entropy $\gamma = N_L * H(C_L) + N_C * H(C_C)$, where N_L and N_C are the number of annotated pro-life and pro-choice hashtags respectively, and $H(x)$ is the entropy of a singular discrete random variable - in this case, the probability that any given biased hashtag is clustered into each of the clusters formed. It may be noted that C_L and C_C are the discrete random variables representing the distribution of the annotated pro-life and pro-choice hashtags respectively within the resultant clusters. $H(x)$ may be calulated using the following formula: $H(x) = -\sum_{i=0}^{i=n} P(x_i) * Log_b(P(x_i))$, where $P(x_i)$

	Tweet Co-occurrence	User Co-occurrence	Tweet and User Co-occurrence (4:1 weight ratio)	Tweet and User Co-occurrence (2:1 weight ratio)
Markov Clustering (inflation = 1.5)	0.0646213267	0.1269305453	0.1247703308	0.1297406947
Markov Clustering (inflation = 2.0)	0.0815237945	0.0729493063	0.1393248853	0.1165547546
Markov Clustering (inflation = 2.5)	0.0733129091	0.0634337357	0.0893344217	0.0773504305
Affinity Propagation (damping = 0.5)	0.1980770285	0.1158433145	0.1652436074	0.1757028125
Affinity Propagation (damping = 0.7)	0.1970993758	0.1172335695	0.1642711037	0.1745926008
Affinity Propagation (damping = 0.9)	0.2141230878	0.1280909866	0.1757028125	0.1916695552

Figure 3: Entropy of different settings for each clustering algorithm with each network creation method.

is the probability that a hashtag is clustered into cluster i, where there are n clusters. The chart below shows the resulting entropy from multiple different tests on each of the co-occurrence clustering methods, using a training set of annotated hashtags.

In Figure 3, we may note that the affinity propagation method creates much higher entropy clusters, implying that the clusters formed are not as distinct as those created by Markov clustering. Also, we may note that user co-occurrence actually resulted in the least entropy using Markov clustering with an inflation value of 2.5. For tweet co-occurrence, the least entropic paremeters were using Markov clustering with inflation = 1.5. For the combination of hashtag and tweet co-occurrences, the minimal entropy is found using a 2:1 user to tweet co-occurrence weight ratio with Markov clustering using an inflation value of 2.5.

Hashtag Co-occurrence Within Tweets

Due to the sparsity of hashtag co-occurrence within tweets, such that each tweet contains around 2 hashtags on average, it is difficult to cluster the hashtags into large general clusters. Instead, many small clusters were generated, both by Markov clustering and affinity propagation. In essence, co-occurrence within tweets does create a strong relation between hashtags. However, the sparsity results in high affinity between small sets of hashtags, essentially making the desired generation of large clusters as desired very difficult. However, these small clusters may yield some interesting and well defined relationships. Using the test set of annotated hashtags with the best parameters found for this procedure, the resulting entropy was 0.0903140674541, compared to a training

entropy of 0.0646213267. A (human) selection of well defined clusters resulting from this procedure may be noted in Figure 4.

'repealthe8th', 'twowomentravel', 'ireland', 'cblive', 'vinb', 'roseoftralee', 'notacriminal', 'feministagenda', 'loveboth', 'micksbill', 'freesafelegal', 'sydneyrose', 'euref', 'mariestopes', 'celebratethe8th', 'womenhurt', 'extend67', 'womenbetrayed', 'arcmarch16', 'womenmatter', 'someoneyoulove', 'ifpa', 'balance', 'dculawconf', 'notavessel', 'thankyou', 'stoppunishingtragedy', 'iccl', 'dail', 'latelateshow', 'nondirectivecounselling', 'emergency', 'rte', 'ep2016', 'maternity', 'marian', 'awidforum', 'allirelandfinal', 'thenewsclub', 'ffabill', 'abortionbill'
'tcot', 'ccot', 'wakeupamerica', 'dncchecklist', 'teaparty', 'margaretsanger', 'tlot', 'media', 'hillary2prison', 'lnyhbt', 'fridayreads', 'tgdn', 'hillaryproabortion', 'climate', 'defunfpp', 'twisters', 'tiot', 'memphis', 'radioanswer', 'spain', 'ycot', 'bahamas', 'agw', 'tennessee', 'thembng', 'ocra', 'penn', 'electiondayin5words', 'bcot', 'nashville', 'orpuw', 'uniteright'
'nevertrump', 'cruzcrew', 'dumptrump', 'freethedelegates', 'delegaterevolt', 'notpence', 'delegatesunbound', 'madamepresident', 'ca', 'neverpence', 'ichooseted', 'manbaby', 'warrenvp', 'stoptrump', 'caprimary', 'writeincruz', 'exgop', 'ijs', 'makeamericasaneagain', 'neveragain'
'imwithher', 'tntweeters', 'ifmenhadperiods', 'flipitdem', 'bernieorbust', 'bernie', 'dk', 'hillyes', 'mediabias', 'womancard', 'heatofdebate', 'vets', 'stopthemvotedem', 'madampotus', 'nv', 'betterway', 'alsohim', 'attwn', 'dealmein', 'gotv', 'lovetrumpshate', 'votehernothim', '100daysofhillary'

Figure 4: Well defined clusters from the tweet co-occurrence network

'repealthe8th', 'twowomentravel', 'cblive', 'roseoftralee', 'vinb', 'feministagenda', 'micksbill', 'liveline', 'loveboth', 'howwedothingsinireland', 'freesafelegal', 'euref', 'askronanmullen', 'sydneyrose', 'extend67', 'arcmarch16', 'balance', 'raydarcy', 'repeal', 'repeal8th', 'notavessel', 'abortionstories', 'thankyou', 'growingupirish', 'someoneyoulove', 'pknt', 'dail', 'stoppunishingtragedy', 'protectthe8th', 'nondirectivecounselling', 'emergency', 'rot', 'celebratethe8th', 'savita', 'ffa', 'dculawconf', 'rte', 'biaskillsdebate', 'givethemavoicecora', 'openyourbookscora', 'haveyoureportedthemcora', 'abortionbill', 'maternity', 'latelateshow', 'ffabill', 'awidforum', 'allirelandfinal', 'electricpicnic', 'abortolibre', 'ifpa', 'marian', 'marref', 'allthelads', 'theresamaypm', 'rally4life', 'ep2016', 'abortthestigma', 'noregrets'
'imwithher', 'hillyes', 'alsohim', 'attwn', 'stopthemvotedem', 'madampotus', 'dealmein', 'voterregistration'
'nevertrump', 'cruzcrew', 'notpence', 'ifmenhadperiods', 'nevertrumporhillary', 'delegatesunbound', 'neverpence', 'heatofdebate', 'ichooseted', 'manbaby', 'exgop', 'neveragain', 'faketrumpintelligencebriefing', 'ripgop', 'imwithher2016', 'igetdepressedwhen', 'trumppense'

Figure 5: Well defined clusters from the user co-occurrence network

Hashtag Co-occurrence Within Users

This method of clustering produces a much more general and weaker relation between hashtags, since a single user's tweets may not be closely related. However, this relation is much better for creating larger, more general clusters, since users do not use sets of hashtags from the opposing sub-culture. This does result in fewer large clusters than co-occurrence within tweets, since edge weights are much more dense and evenly distributed. This also results in much larger clusters, including a more neutral cluster of hashtags used by all sides on the topic. With the test set of annotated hashtags, the resulting entropy using the best parameters found during training was 0.080863546392, compared to a training entropy of 0.0634337357. Another (human) selection of the most well defined clusters may be noted in Figure 5, alongside a 2-d representation of the user co-occurrence affinity graph colored by cluster in Figure 6.

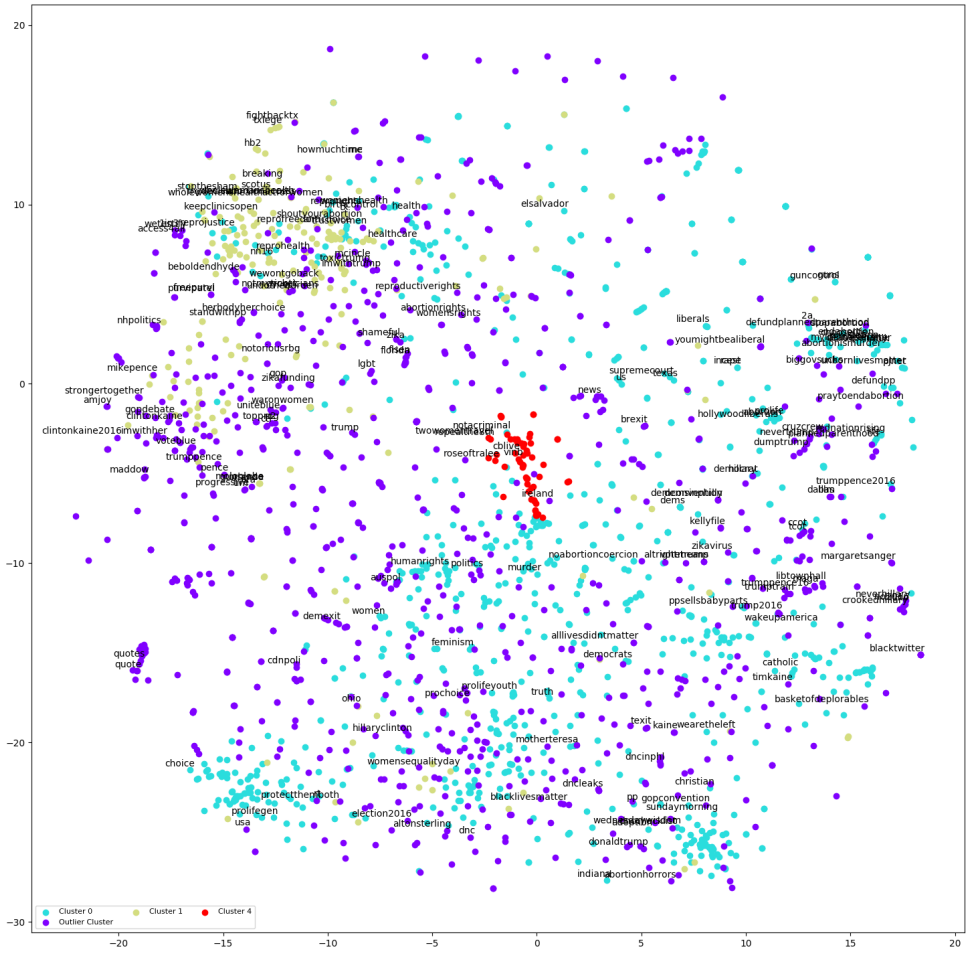


Figure 6: A graph of user co-occurrence hashtags based on their relation to each other

Hashtag Co-occurrence Within Users and Tweets

Since this method attempts to combine the two other co-occurrence methods, it provides a decent middle groud. However, according to the cluster entropy results, this still results in less effective clustering than the user co-occurrence. This may be due to the high entropy found in the tweet co-occurrence, since many of the high affinity relationships in this method are similar to those of the tweet co-occurrence method. However, under certain circumstances this may be desirable, since these high affinity relationships are an important part in understanding the relationships between hashtags. A sample of the resulting clusters may be found in Figure 7.

'pjnet', 'endabortion', 'alllivesmatter', 'defundplannedparenthood', 'stopabortion', 'deathpenalty', 'stoppp', 'mylifematters', 'mlk', 'pro', 'tedcruz', 'defundabortion', 'ntb', 'ham', 'gloriasteinem', 'voteforlife', 'firetrumpin4words', 'kelliward'
'repealthe8th', 'howwedothingsinireland', 'askronanmullen', 'sydneyrose', 'womenbetrayed', 'repeal8th', 'notavessel', 'abortionstories', 'thankyou', 'growingupirish', 'pknt', 'dail', 'ivg', 'rot', 'savita', 'ffa', 'biaskillsdebate', 'avortement', 'awidforum', 'marref', 'irlanda', 'allthelads', 'abortthestigma', 'shoutmyabortion'
'demsinphilly', 'dncinphl', 'demconvention', 'demexit', 'pinkoutthevote', 'imwithhernow', 'corrupthillary', 'jillnothill', 'ttt16', 'openthebigtent', 'dncconvention', 'abortionstigma', 'wikileaks', 'disgrace', 'cbn2016', 'demoncrats', 'blueroom', 'cultofbaal', 'comeyhearing', 'bias'
'nevertrump', 'cruzcrew', 'dumptrump', 'notpence', 'ifmenhadperiods', 'delegaterevolt', 'nevertrumporhillary', 'neverpence', 'voteyourconscience', 'evanmcmullin', 'heatofdebate', 'ichooseted', 'stoptrump', 'ijs', 'ca', 'mcmullin2016', 'exgop', 'neveragain', 'writeincruz', 'ripgop', 'trumppense'
'twowomentravel', 'brexit', 'ireland', 'cblive', 'roseoftralee', 'notacriminal', 'vinb', 'feministagenda', 'micksbill', 'liveline', 'loveboth', 'dumpryan', 'freesafelegal', 'prexit', 'euref', 'extend67', 'arcmarch16', 'balance', 'raydarcy', 'repeal', 'someoneyoulove', 'stoppunishingtragedy', 'nondirectivecounselling', 'emergency', 'celebratethe8th', 'dculawconf', 'rte', 'womenhurt', 'womenmatter', 'maternity', 'latelateshow', 'ffabill', 'allirelandfinal', 'electricpicnic', 'abortolibre', 'ifpa', 'marian', 'theresamaypm', 'ep2016', 'noregrets'

Figure 7: Well defined clusters from the user and tweet co-occurrence network

Hashtag Embeddings

The procedure was run on the set of abortion tweets split by week and with all 10 weeks combined. The produced hashtag graph is clustered and analyzed, resulting in hashtag clusters. These clusters can be interpreted using the reference hashtags manually picked to be representative of the issue being focused on. Some reference hashtags and the clusters they are grouped in can be noted in Figure 8. The expected and ideal results would be for absolutely all of the hashtags in the pro-life group to belong to the opposite cluster from the pro-choice hashtags. This never occurs in the reference hashtags, but this is not surprising given how volatile the meanings of hashtags are,

Week		1	2	3	4	5	6	7	8	9	10	All
Prolife Hashtags	defundpp	1	0	1	0	1	1	0	1	1	0	1
	praytoendabortion	1	0	1	0	1	1	0	1	1	0	1
	prolife	1	0	1	0	1	1	0	1	1	0	1
	ppsellsbabyparts	0	0	1	0	1	1	0	1	1	0	1
	stopabortion	0	1	0	1	1	1	1	1	1	0	0
Neutral Hashtags	abortion	1	0	1	0	1	1	0	1	1	0	1
	pregnant	0	1	0	1	0	0	1	0	0	1	0
	healthcare	0	1	1	0	0	0	1	1	0	1	0
	medical	0	1	0	1	X	0	1	0	0	1	0
	health	1	0	0	1	0	1	0	1	1	0	0
Prochoice Hashtags	prochoice	1	0	1	0	1	1	0	1	1	0	1
	antichoice	0	1	1	0	0	0	1	1	0	1	0
	womensrights	1	1	0	1	0	1	1	0	0	1	0
	mybodymychoice	0	1	0	1	0	0	1	0	0	1	0
	fundpp	1	0	1	X	X	0	X	1	X	0	1

Figure 8: Reference hashtags and their resulting clusters for each week of data. X symbolizes that the hashtag was not found that week.

especially when dependent only upon a week-long interval. The main notable feature in Figure 8 is that there is a distinct (but not absolute) polarization between pro-life and pro-choice hashtags. Specifically, in weeks 2, 5, 6, 7, 9, and 10 the majority of pro-life hashtags are in the opposite cluster from the majority of pro-choice hashtags. While these polarized majorities hardly indicate success, they are promising considering that a random distribution should contain very few of these occurrences. Additionally, none of the neutral hashtags have a ratio higher than 3:1 (aggregate run excluded), indicating that the validity of the clusters remains.

In addition to the reference hashtags, much can be said of the original clusters themselves. Figure 9 shows the sizes and standard deviation of the two clusters formed from the hashtag graph, as well as the distance between them. By taking the squared sum of error for the clusters, we can approximate a distance between the centroids of the clusters relative to the normal probability density of them. Mathematically, we may write N_1 and N_2 to represent the sizes of each cluster and $\sigma(C_i)$ to represent the standard deviation of cluster i. Then the weighted average standard deviation (WASD) will be

$$s(C_1, C_2) = \frac{(N_1\sigma(C_1)+N_2\sigma(C_2))}{(N_1+N_2))}$$

and the relative distance will be quotient of the distance (D) over the WASD, or

$$r(C_1, C_2, D) = \frac{D}{s(C_1,C_2)}.$$

We may note that the relative distance is fairly consistent in the range of 9 to 20, and we may even note a peak in debate during Week 6 (7/29/16 - 8/5/16) which circumstantially followed controversial remarks from U.S. Vice Presidential Candidate Tim Kaine [4][12]

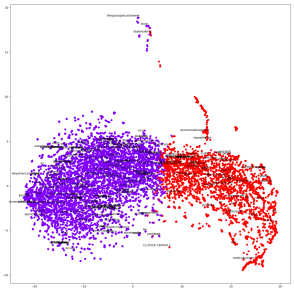
Lastly, we may reduce the hashtag relationship dimensions so that they may be visualized. This is achieved using T-distributed Stochastic Neighbor Embedding (TSNE) [11]. This allows our hashtag graph to be expressed as a set of two dimensional points for each hashtag, which subsequently allows for the hashtag relationships to be plotted on paper. These dimension reduced hashtag graphs can be noted in Figure 10. It should be noted that much of the information is lost in the TSNE reduction, since the data is reduced from about 5,000 dimensions (depending on the number of hashtags in the data set) down to 2. However, there are still clearly defined and consistent clusters despite this loss, which is promising for the validity of the hashtag relationships. Notable features of these graphs are the consistent "weaker" arm on the right side, as well as far outliers at the top. It also appears as if Week 6 was an especially abnormal week considering the extensive growth of the left cluster, which agrees with the data in Figure 9. It is important to note that the swapping of colors between the clusters by week is irrelevant, as this is only based on the number assigned to the cluster, and individual hashtags remain in the same clusters for the most part. Additionally, this is independent data from separate weeks (subfigure K excluded), and the consistency between weeks is another promising indicator of accurate hashtag embeddings.

Week	# of Tweets	C_1 size	$\sigma(C_1)$	C_2 size	$\sigma(C_2)$	D	$s(C_1, C_2)$	$r(C_1, C_2, D)$
1	161,537	6,665	2.61	3,335	2.53	23.36	2.58	9.04
2	150,758	1,902	2	4,290	0.77	11.83	1.15	10.31
3	182,649	4,180	0.77	1,939	2.12	11.87	1.20	9.91
4	170,366	1,752	1.75	3,873	0.69	11.27	1.02	11.05
5	106,860	3,926	0.67	1,678	1.66	11.67	0.97	12.08
6	150,406	4,853	0.16	497	0.86	4.31	0.23	19.15
7	478,441	893	0.93	3,141	0.55	9.5	0.63	14.98
8	141,943	3,734	0.72	1,582	1.76	11.71	1.03	11.37
9	180,676	3,777	0.99	1,820	2.02	14.26	1.32	10.76
10	165,554	1,706	1.86	3,808	0.66	10.85	1.03	10.52
All	2,114,926	5,934	1.66	4,066	2.47	20.89	4.12	5.08

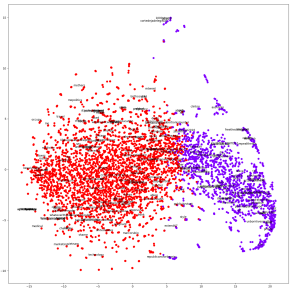
Figure 9: Results of clustering the generated hashtag graph for data on the topic of abortion (weeks of 6/24/16 - 9/2/16 and their aggregate)

Conclusion

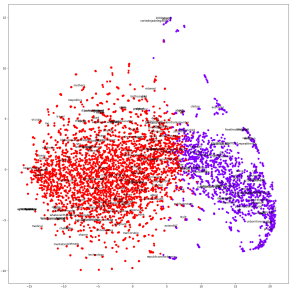
In conclusion, the choice between different methods of identifying clusters of hashtags and the conflict between them is highly dependent on the form of the desired clusters. If large general clusters are desired, then co-occurrence within user's tweets are preferable. If small and highly related clusters are desired, then the co-occurrence between individual tweets is preferable, although this does not provide much insight into hashtag relations outside of clusters. For a decent middle ground, using the combination of user and tweet co-occurrence is preferable, since it allows for the best of both other co-occurrence methods. Lastly, if usable representations of hashtags in vector space is desired, then creation of hashtag embeddings may be the best option presented.



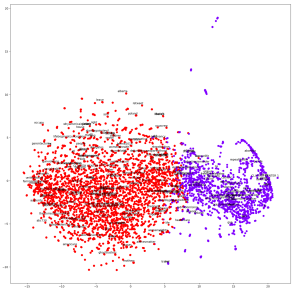
(a) Week 1



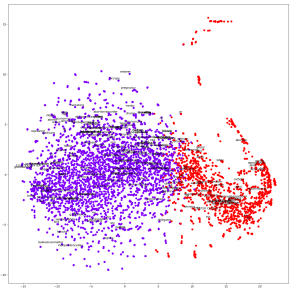
(b) Week 2



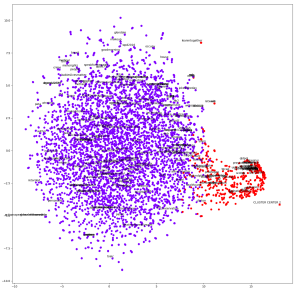
(c) Week 3



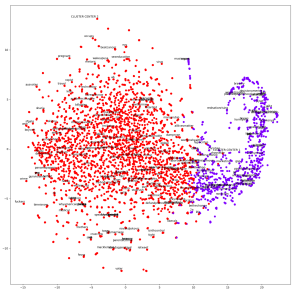
(d) Week 4



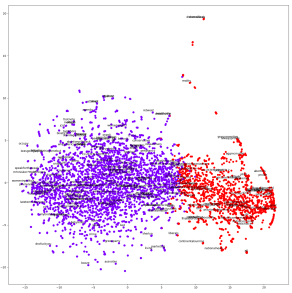
(e) Week 5



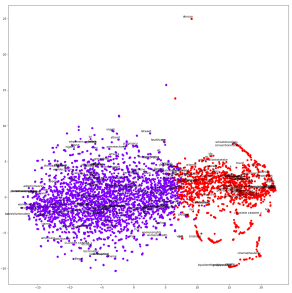
(f) Week 6



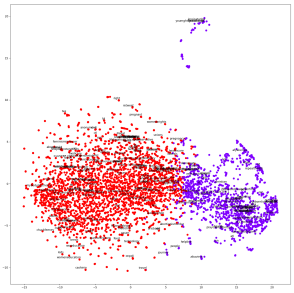
(g) Week 7



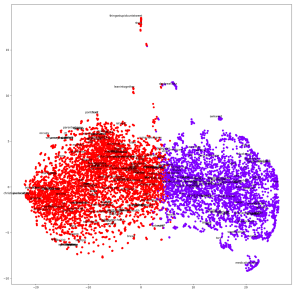
(h) Week 8



(i) Week 9



(j) Week 10



(k) All data

Figure 10: Dimension reduced graphs of abortion hashtag embeddings by week (and all data combined)

Bibliography

[1] Abadi, Martin, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Rafal Jozefowicz, Yangqing Jia, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Man, Mike Schuster, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Vigas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, (Version 1.2) 2015. Web. 7 June 2017.

[2] Abdi. H., & Williams, L.J. (2010). "Principal component analysis". *Wiley Interdisciplinary Reviews: Computational Statistics*. 2 (4): 433459. Web. 6 June 2017.

[3] Conover, M. D. et al. "Political Polarization On Twitter". *Fifth International AAAI Conference On Weblogs And Social Media*. Bloomington: Indiana University. Web. 5 May 2017.

[4] Davis, Susan. "The Policy Split Between Clinton And Kaine On Abortion, Explained." NPR. NPR, 01 Aug. 2016. Web. 07 July 2017.

[5] Dongen, S. van. "Graph clustering by flow simulation." Proefschrift Universiteit Utrecht, 2000. Web. 20 Aug. 2017.

[6] Ester, Martin, Hans-peter Kriegel, Jrg Sander, and Xiaowei Xu. "A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise". N.p.: AAAI, 1996. 226-31. Web. 6 June 2017.

[7] Frey, Brendan J, and Delbert Dueck. "Clustering by Passing Messages Between Data Points." *Science*, vol. 315, no. 5814, 16 Feb. 2007, pp. 972976. Web. 20 August 2017.

[8] Hartigan, J. A., and M. A. Wong. "Algorithm AS 136: A K-Means Clustering Algorithm." *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, 1979, pp. 100108. Web. 5 June 2017.

[9] Le, Quoc V., and Tomas Mikolov. "Distributed Representations of Sentences and Documents." CoRR 4053rd ser. Abs.1405 (2014): n. pag. Web. 7 June 2017.

[10] Loper, Edward and Steven Bird. 2002. "NLTK: the Natural Language Toolkit". *In Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics (ETMTNLP '02)*, Vol. 1. Association for Computational Linguistics, Stroudsburg, PA, USA, 63-70. Web. 6 June 2017.

[11] van der Maaten, L.J.P.; Hinton, G.E. (Nov 2008). "Visualizing High-Dimensional Data Using t-SNE". *Journal of Machine Learning Research*. 9: 25792605. Web. 6 July 2017.

[12] Mellen, Ruby. "Kaine breaks with Clinton on abortion provision." *CNN*. Cable News Network, 01 Aug. 2016. Web. 7 July 2017.

[13] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Jrg Sander. 1999. "OPTICS: ordering points to identify the clustering structure". *In Proceedings of the 1999 ACM SIGMOD international conference on Management of data (SIGMOD '99)*. ACM, New York, NY, USA, 49-60. Web. 7 June 2017.

[14] Mikolov, Tomas and Sutskever, Ilya and Chen, Kai and Corrado, Greg S and Dean, Jeff. "Distributed Representations of Words and Phrases and their Compositionality", *Advances in Neural Information Processing Systems 26*. Eds. C. J. C. Burges and L. Bottou and M. Welling and Z. Ghahramani and K. Q. Weinberger Curran Associates, Inc., 2013, 3111–3119. Web. 7 June 2017.

[15] Novikov, Andrei. Pyclustering Library (Version 0.6), 2017 Web. 7 June 2017.

[16] Pedregosa, F., et al. Scikit-Learn: Machine Learning in Python. *Journal of Machine Learning Research*, vol. 12, 2011, pp. 28252830.

[17] Pelleg, Dau, and Andrew Moore. "X-Means: Extending K-Means with Efficient Estimation of the Number of Clusters." Morgan Kaufmann, *In Proceedings of the 17th International Conf. on Machine Learning*, 2000, pp. 727734.

[18] Per Christian Hansen. 1987. "The truncated SVD as a method for regularization". *BIT* 27, 4 (October 1987), 534-553. Web. 5 June 2017.

[19] "Rust API." Twitter. Twitter, n.d. Web. 07 July 2017.

[20] Sarita Yardi and Danah Boyd, "Dynamic debates: An analysis of group polarization over time on twitter", *Bulletin of Science, Technology & Society 30 (2010)*, no. 5, 316327. Web. 5 May 2017.

[21] Zhang, J. , W. Hamilton, C. Danescu-Niculescu-Mizil, D. Jurafsky, and J. Lescovec. "Community Identity and User Engagement in a Multi-Community Landscape." *AAAI International Conference on Weblogs and Social Media (ICWSM) (2017)*. Web. 7 July 2017.