Twitter use in Natural Disasters:

Using Twitter to Make Calls for Help During Hurricane Harvey

Courtney J. Powers, Aman Dontula, Kaab Ashqeen, Ashwin Devaraj, Amit Joshi, Jay Shenoy,

Dhiraj Murthy

The University of Texas at Austin

Abstract

This research explores the role of Twitter during and in the aftermath of Hurricane Harvey, how Twitter is used in making calls for help and rescue, and how we automate the detection of tweets relevant to first-responders. More specifically, we present a labeling scheme to categorize tweets as relevant or irrelevant and urgent or not urgent and develop machine learning models to classify tweets related to Hurricane Harvey. Our best relevance classifiers (support vector machine and convolutional neural network) capture a larger fraction of relevant tweets than existing non-neural classifiers but underperform compared to convolutional neural networks trained to perform a simpler classification task. Although the highly unbalanced nature of the urgency dataset leads to mediocre model performance, we illustrate the benefit of SVM-SMOTE oversampling in improving classifier performance on unbalanced disaster datasets.

Twitter use in Natural Disasters:

Using Twitter to Make Calls for Help During Hurricane Harvey

When disaster strikes and traditional forms of emergency communication (like US-based 9-1-1 systems) fail, individuals are forced to use whatever means available to communicate and seek help from others (Rhodan, 2017; Glass, 2001). Often, the technology that is easily and readily available and familiar are social media sites and applications, such as Twitter. Previous work finds that individuals used social media during disasters in attempts to communicate, connect, and even coordinate rescue and resource efforts (Paul, 2015; Rhodan, 2017; Lindsay, 2011). When Hurricane Harvey made landfall on the coast of Texas in August 2017 and created catastrophic flooding, this was no different (Robertson, Johnson, Murthy, Smith, & Stephens; in press; Smith, Stephens, Robertson, Li, & Murthy, 2018; Yang, Nguyen, Stuve, Cao, & Jin; 2017). Thousands of people turned to Twitter to connect with others, share information, and even request help during the deadly and record-breaking storm.

Given that social media has been an instrumental communication tool in connecting people in emergency situations, as previous research suggests (Robertson et al., under review; Murthy & Longwell, 2013; Lindsay, 2011; Yang et al., 2017), this study proposal will specifically examine the social media site, Twitter, and how it is used in a major disaster such as Hurricane Harvey. More specifically, this study investigates the role Twitter played in Hurricane Harvey and how people use Twitter to request or coordinate rescue efforts through machine learning techniques. By examining ‘calls for help’ on Twitter through machine learning methods, this research ultimately identifies calls for help among the many other tweets through automatic algorithmic categorization that can ultimately be used to inform first responders how people are using Twitter to request help in urgent, life-threatening disaster situations. In the following sections, we will first introduce Hurricane Harvey as a unique natural disaster, then we will discuss the previous literature about disasters and Twitter use, introduce research questions, the methodology, and the results. Finally, the paper will conclude with a discussion of the findings, implications, and directions for future research.

Hurricane Harvey Overview

Hurricane Harvey made landfall along the mid-Texas coast on August 24, 2017 as a category four hurricane with winds of 130 miles per hour (Blake & Zelinsky, 2018; Carter, 2018). The devastating storm caused catastrophic flooding throughout southeastern Texas, affecting the city of Houston and other neighboring towns (Blake & Zelinsky, 2018). According to Blake and Zelinsky's (2018) report for the National Oceanic and Atmospheric Administration, Hurricane Harvey was "the most significant tropical cyclone rainfall event in United States history both in scope and peak rainfall amounts" and caused damage to thousands of homes and businesses (p. 6). While many people were asked to evacuate in preparation for the storm, many did not and were trapped or stranded and required rescue due to dangerous rising flood waters. By August 31, 2017, FEMA reported that federal forces had rescued over 10,000 people, which does not include the countless rescues made by good samaritans (Gallagher, 2017). This storm, along with the record-breaking rain and flooding, led to an estimated $125 billion dollars' worth of damage and tragically resulted in at least 68 deaths (NOAA, 2018; Blake & Zelinsky, 2018).

Review of the Literature

Twitter Use and Its Role in Disasters

Within a large disaster or emergency situation, communication challenges can arise. Traditional forms of communication like the US-based 9-1-1 emergency system can fail due to an overwhelming influx of calls and inquiries (Rhodan, 2017; Glass, 2001), and people can lose touch with family and friends in the midst of evacuating and/or taking life-saving measures. According to Glass (2001), US-based emergency systems like 9-1-1 were never created or set up to handle the influx of calls that results in a large-scale natural disaster situation. And, in some cases, disaster victims may actually turn to social media as their first venue for assistance or to make a call for help, not even trying to use 9-1-1 (Stephens, Robertson, & Murthy, in press).

According to Palen and Hughes (2010), social media, such as Twitter, affords individuals the opportunity to easily communicate, connect, share and seek information, and even coordinate rescue and relief efforts during disasters and crises. Historically, Twitter as an online microblog has been particularly useful for sharing brief text and images during disasters (Murthy, 2018). Given the widespread global use of social media and its ease of use on both mobile and desktop platforms, Twitter has become a prominent form of communication (Pew Research Center, 2018), especially in disaster situations when other communication channels like emergency helplines can have limited efficacy.

Since Twitter's emergence in 2007, Twitter has been actively used in a variety of different disasters (Murthy, 2018). The platform allows users "the opportunity to post, read, and respond" to short text-based messages and "creates a multi-media platform with constantly updated timelines for wide-open content" (Spence, Lachlan, Lin, & del Greco, 2015, p. 172). Previous research about disasters and Twitter found that people use Twitter to share news and information, reactions and feelings, and even call for help (Paul, 2015; Lindsay, 2011; Yang et al., 2017; Spence et al., 2015). For example, Spence et al. (2015), who examined Twitter use during Hurricane Sandy, found that a large portion of tweets included either information or displays of feelings or reactions such as sorrow, anger, or fear.

Twitter as a Tool to Call for Help in Disasters

For the purpose of this paper, the definition for a call for help on Twitter is largely based on Paul (2015)'s work and is defined as a person issuing a tweet on Twitter during a natural disaster to request help, rescue, medical assistance or information for themselves or for someone else who they believe is affected by the natural disaster. A number of studies have explored social media use during disasters, but only a handful of studies have specifically addressed or mentioned calls for help and rescue on Twitter during a disaster. For example, in their study of social media use in a Japanese earthquake and tsunami, Peary, Shaw, and Takeuchi (2012) found individuals used Twitter to communicate whether they were safe or unsafe, and there were even a number of "calls for help," some of which were inaccurate or fabricated. While Peary et al. (2012) did not provide any examples for calls for help on Twitter, their research gives proof that Twitter is indeed a platform that people post about their safety and make calls for help.

Similarly, in a study of the Ondoy Typhoon and subsequent flooding, Morales (2010) collected Twitter data and performed a content analysis to examine how individuals created rescue networks to help people in need during the disaster. Like Peary et al. (2012), Morales (2010) found that a number of people used Twitter to make a call for help or rescue. In the study, Morales provided a number of examples of users' tweets to demonstrate what calls for help look like and explains "…during Ondoy, users sent a variety of messages pertaining to individual calls for rescue and evacuation and the need for material donations, effectively transforming Twitter from a mere platform of social exchange, into an active facilitator of relief and rescue" (2010, p. 57). Additionally, Morales (2010) found that individuals who were not directly affected by the disaster used Twitter as a means to ask others to check on their loved ones that could be in danger and may be in need of rescue.

Imran, Elbassuoni, Castillo, Diaz, and Meier (2013) collected data from the Joplin tornado and Hurricane Sandy through Twitter's API and automatically coded tweets. Similar to Morales (2010), Irman et al. (2013) also found that Twitter was used to call for help for others who may be in need of rescue from the disaster. While Imran et al. (2013) did not address individuals' calls for their own help or rescue, their research still shows how Twitter was used in attempts to coordinate help and rescue initiatives for people affected by the disaster. A more recent study by David, Ong, and Legara (2016) also found evidence that Twitter was used for rescue coordination in Typhoon Haiyan, noting that, of the total "Spritzer" level data that was collected around the disaster and classified as disaster relief, "12% reported on personal acts of disaster relief, 67% reported on the relief activities of others, 8% were solicitations of specific kinds of help, and 54% are coordinative in nature" (p. 11). Furthermore, because of the use of Twitter for calls of help and help coordination, David et al. consider Twitter to be an important "…venue for mobilizing relief and response on a global scale" in disaster situations (2016, p. 2).

While some studies did not give exact numbers of how many tweets were actually classified as "calls for help," David et al. (2016) went as far as to give percentages of the number of tweets classified as help-seeking, revealing that 8% were solicitations of specific kinds of help of the tweets classified as disaster relief related. Given the previous research, it is clear that Twitter has been used to make calls for help during disasters. Next, we will discuss the limited extant research about calls for help on social media during the disaster of focus, Hurricane Harvey.

Twitter and calls for help during Hurricane Harvey

There has been some research specific to Hurricane Harvey. Given the uniqueness of Harvey, both in scale of destruction as well as the unprecedented levels of volunteer rescues, research has specifically explored how social media was used to make calls for help. According to Robertson et al. "At the time of Hurricane Harvey in 2017, Twitter had 2.46 billion users" (in press, p. 6). Robertson et al. explain that Harvey was considered "a unique natural disaster in which Twitter and other social media provided citizens with a platform to communicate urgent information quickly that ultimately led to life-saving rescues for those who were flooded" (in press, p. 6; Stephens, Li, Robertson, Smith, & Murthy, 2018).

In their study, Robertson et al. (in press) specifically examined Twitter and calls for help during Hurricane Harvey, looking exclusively at images tweeted before, during, and after the landfall of the storm. Robertson and colleagues used a mixed-method approach to compare the accuracy of human-coded and machine coded images. Through using a deep learning VGG-16 convolutional neural network, along with a construction of multilayered perceptron classifiers for classifying the urgency and time period for a given image, Robertson et al. (in press) found that computerized methods can be used to “filter through the noise on social media and identify authentic calls for help or urgent situations during a disaster” (p. 29).

Yang et al. (2017) also studied tweets during Hurricane Harvey, this time looking specifically at the text, instead of images. Yang et al. (2017) argue that 9-1-1 emergency call centers can only serve a certain number of people who need help at a time, as can rescuers with boats during Hurricane Harvey. Because resources are limited and people’s lives are in danger, their study aimed to identify and prioritize those in need. Yang and colleagues (2017) manually labeled 1,000 tweets to train a model to be able to identify tweets that had requests for rescue. Their objective was to triage levels of need by using a 'scheduling algorithm' to try to optimize resource allocation to those who need it most based on tweet data. In their study, 70% of the coded data was used for training, while the rest was used for validation. Their best performing support vector machine classifier has an F-measure of .687 and accuracy of .93, meaning that they had achieved some success in classifying tweets with their relatively small coded dataset.

Research Questions

Previous research found machine learning is effective in categorizing data from tweets related to disasters (O'Neal et al. 2018), with Robertson et al. (in press) and Yang et al. (2017) finding that calls for help during Hurricane Harvey can be identified through deep learning methods and automatic categorization methods. This research will not only incorporate Robertson et al.'s (in preparation) research by using a coding category technique, but this particular study aims to expand upon the findings of Yang et al. (2017) by using a larger Twitter dataset from Hurricane Harvey. While Yang et al. (2017) tried to triage levels of need by using a scheduling algorithm, we are optimizing the part of identifying tweets relevant to rescue/rescuing itself rather than the optimization within. By using a computational machine learning method, this study applies similar categories identified in Robertson et al.'s (in press) study, as well as draws upon other categories regarding how individuals use Twitter to make calls for help in Hurricane Harvey.

Ultimately, by using a computational and machine learning approach, the goal of this research is to determine what calls for help and rescue on Twitter look like in disaster situations and ultimately train a computer to automatically be able to identify calls for help in real-time in a way that will be helpful in saving lives during another disaster. Therefore, we explore the following research questions:

(RQ1): What purpose does Twitter play in a disaster such as Hurricane Harvey?

(RQ2): Do people use Twitter to request or coordinate rescue efforts?

(RQ2A): How do people use Twitter to request or coordinate rescue efforts?

Method

Using established and tested methods that apply machine learning techniques to categorize textual tweets during previous hurricane events [e.g. Ashktorab, Brown, Nandi, & Culotta, 2014; Imran, Castillo, Lucas, Meier, & Rogstadius, 2014], we explore how Twitter was used during Hurricane Harvey and whether Twitter was used to make calls for help. We first performed a preliminary pilot data analysis in order to better understand the dataset. Then, following previous work using machine learning on disaster-related tweets (Morstatter, Lubold, Pon-Barry, Pfeffer, & Liu; 2014; Derczynski Meesters, Bontcheva, & Maynard, 2018), we human coded a subset of the tweets as a team for the purpose of building, training, and fine-tuning a machine-learned model. After, we deployed a larger subset of 4,000 tweets to Amazon Mechanical Turk (MTurk) for additional human coding to be used to refine the computer-based model with the goal of making a more accurate classifier. We employ this crowd-based process of creating a much larger training set of labeled tweets based on recommendations from other successful studies (Li et al., 2015; Appling, Briscoe, Ediger, Poovey, & McColl; 2014) aimed at classifying disaster-related tweets.

Data Sample

This study used a dataset collected at the University of North Texas (UNT) which contains 7,041,866 total tweets related to the Hurricane Harvey disaster and was created using the ‘twarc’ tweet archiving package via Twitter’s API (Phillips, 2017). The data was collected from the Texas-specific location coordinates over a nearly a month-long period (08/18/17 -

09/22/17) in which Hurricane Harvey made landfall and severely impacted the greater Houston area. The dataset is derived from API searches based on seven hashtags and three additional keywords that were included in the search query to collect the Twitter data (Phillips, 2017). These keywords, all of which are related to Hurricane Harvey and its impact on the greater Houston area, are: #Harvey, #Harvey2017, #HarveyStorm, #HoustonFlood, #HoustonFlooding, #HoustonFloods, #HurricaneHarvey, #GulfCoast, Hurricane Harvey, and Twitter.

Data Preprocessing

To prepare the data for the project, a number of established preprocessing techniques were applied. We first tokenized the tweets, which is the process of separating the words into smaller pieces called tokens to make data processing and categorization easier (NLP Tokenization, 2008). This was done using the NLTK natural language processing library's TweetTokenizer (2015), which automated tokenizing steps like removing punctuation, removing username handles (e.g. "@user123"), compressing sequences of more than three identical letters down to length three (e.g. "wwwaaaaaayyyy toooo much" becomes "wwwaaayyy tooo much"), and splitting the processed tweet into a list of word tokens. To denoise the dataset, we also removed a number of stop words, words too frequently used in English to be considered helpful to a machine learning model, from the dataset. The stop words we decided to remove are based on a common list of Twitter stop words provided in the NLTK library (NTLK, 2015). Additional preprocessing included converting all tweets to lowercase to ameliorate case bias, and converting numbers, links, and hashtags into special tokens (<number>, <url>, and <user>

respectively) to use when applying word embeddings to the dataset later on. In this preprocessing, we also undertook lemmatization, which has the goal of transforming inflectional and derivationally related forms of a word to a common base form that can easily be used in computational categorization (NLP Stemming & Lemmatization, 2008). We initially had stemming and decided to remove it since stemming can map words to a shortened version that is not always a valid word, and valid words were necessary for the use of word embeddings as we discuss later. We noticed that many words that seemingly did not appear in the dataset were actually valid words with no spaces between then, such as ‘hurricaneharvey’, which should be ‘hurricane harvey’. In order to extract useful information from such strings, we use a Python API of the SymSpell software’s word segmentation function to split these strings into their component words (Garbe, 2014; Mammothb, 2018). Since the word segmentation function was slow, we indexed the word embeddings first and only used SymSpell on strings that did not appear in the indexed embeddings. The number of unknown words in the dataset decreased dramatically, from around 5% to less than 1%.

In our first pass of the data, we determined that there were a number of retweets in the over seven million tweet dataset. After closer examination of the data, we were able to determine that less than 1.5 million were unique tweets (e.g. not retweets). During preprocessing, we removed tweets with duplicate text bodies. Though this does minimize duplication, some of the tweets in our dataset consisted of unique retweets. We also removed all the metadata associated with the tweets except the raw text itself since our objective was to only use the text for our machine learning models. We especially made sure to remove user ids to protect users' privacy when posting the tweets onto MTurk for labeling. After preprocessing, we obtained a sample of 1,454,297 from the full dataset of over seven million tweets.

Preliminary Analysis

Before building specific models, we decided to explore the data using the topic modeling method latent Dirichlet allocation (LDA) to see if we could easily divide the tweets into meaningful categories solely based on the words they contain. LDA is a technique that, given a set of documents and number of topics, determines the words associated with each topic and for each document determines a probability distribution of topics (Blei, Ng, & Jordan, 2003). Running LDA on the 1.5 million tweets with various numbers of topics resulted in topics with very similar word compositions, so we determined that LDA-based features probably would not be very useful for developing machine learning models since the tweets' features would be too similar to each other to build accurate models. Although this decision was only based on a preliminary exploration, we decided that the following preprocessing step would provide sufficient opportunities for developing good models.

After attempting to analyze the tweets with LDA, we decided to apply another more conventional preprocessing step to the tweets prior to developing text classifiers namely applying word embeddings. Word embeddings were applied to the dataset by converting each tweet into the average of the embeddings representing each of its words and the special tokens <url>, <number>, and <user>. We used embeddings of length 100 (experimentally-chosen length) that were pre-trained on two billion tweets using the GloVe method (Pennington Socher, & Manning, 2014). According to Dalinina (2017), word embeddings can be useful in machine learning because they provide researchers with a "sophisticated way to represent words in numerical space by preserving word-to-word similarities based on context."

Because tweets during disaster events are both high volume and rife with non-relevant posts (Derczynski et al., 2018), we filtered the remaining 1,454,297 tweets to a dataset of around 300,000 by only including tweets that contained one or more of 41 specific words or phrases and did not include three words (displayed in Table 1). For example, one of the words we made sure to exclude was "donate" and "Red Cross", as a large number of the tweets were users and organizations asking people to donate. This filtering helped ameliorate the bias towards non-relevant tweets when searching for catchall hashtags such as #Harvey and increase our chance of coding the types of tweets relevant to our research questions. After the data was filtered, we used it in our following methodology steps, starting with a random random sample of 2,000 individual tweets for manual labeling and preliminary machine learning model selection before moving on to deploying 4,000 tweets for coding to MTurk to ultimately improve our machine learning model.

Table 1

Words Used to Filter Dataset		
Help	Help me	Need help
Please help	Please help me	Emergency
Need	Blanket	Food

Need Rescue	Need to evacuate	Need to leave
Need to get out	Rescue	Rescue me
Please	Danger	In danger
Dangerous	Need boat	Have boat
Water rescue	Medical help	Tree down
Water	Flood	Water in house
House flooded	House is flooding	Water rising
Road blocked	National Guard	Police
Rescuers	Volunteers	Firemen
Police officer	911	Call 911
Ambulance	Damage	-News
-Donate	-Red Cross	

Coding Categories

For the purpose of this study, we had two main coding categories. The first is called relevance and the second is urgency. While the UNT data should ideally be relevant to Hurricane Harvey based on the data collection hashtags, time frame incorporating Hurricane Harvey's descent, and the geographical coordinates, not all of the data in the over seven million tweet dataset, or even our filtered subset is relevant to the hurricane or our research questions. As a result, we relied on Paul's (2015) coding structure, which was used by Robertson et al.'s (in press) study.

Paul’s (2015) typology of social media posts during disasters was our relevance coding categories, and is comprised of the following categories: request, report, and reaction or none. Paul’s (2015) coding typology it a good fit for our study because it allows us to answer our research questions as it pertains to calls for help and how Twitter is used in general in disasters like Hurricane Harvey.

For example, Paul’s (2015) “request” category is essentially a “call for help.” The original definition or criteria we had for hand-coding requests was “Is the user asking for help or assistance as a result of Hurricane Harvey, including material support (such as food, supplies, or shelter), medical assistance, immediate help/rescue, and/or information about a person affected by the storm, area, or something else related to the storm?” However, this definition was adapted a twice after and eventually shortened and simplified to “Asking for help or information as a result of Hurricane Harvey,” with the exclusion of “Donations in the form of money or news sources” for coding from MTurk workers.

It is important to note that in this category, we left the request category relatively loose to account for calls for help for one’s self and calls for help for others who they may feel are in danger and in need of rescuing. This is important because calls for help and rescue can go beyond just the individual person (Irman et al., 2013; Morales, 2010). For example, those who are in immediate danger may not have the resources or ability to tweet out a call for help (such as in the case where flood water is rising or there are power outages that prevent access to technology), so it is plausible to think that others who know or care about them would reach out for help on Twitter on their behalf.

The second coding category from Paul's (2015) typology is report, and was meant to get at individuals' reported thoughts and experiences about Hurricane Harvey. For our team's coding, the report category was stated as "Is the user reporting damage of public property, their own house, or environmental damage from Hurricane Harvey? Is the user reporting about how people in their community are affected by Hurricane Harvey, such as people's mood, behavior, or situation? Is the user reporting updates about Hurricane Harvey, including change in a situation and/or injuries and deaths as a result of the storm?." This definition was adapted and shortened twice, and ultimately read "Reporting damage, updates, injuries or deaths from Hurricane Harvey" for the second iteration of coding done by MTurk workers.

The third coding category from Paul's (2015) typology is reaction, and was meant to assess individuals' feelings, opinions, and thoughts about emergency response workers and volunteers. For our team's coding, this was explained as "Is the user referring to or discussing emergency service (ambulance, police, firemen, National Guard, responders), such as performance of emergency response officials (first responders, federal/state officials)? Is the user reacting to efforts from their community, such as volunteers or food providers? News sources are not included." This code was also adapted twice and substantially shortened to read "Referring to or talking about performance or effectiveness of emergency responders or volunteers." This shortened coding category was used in the second iteration of coding completed by MTurk workers.

The last category for relevance was none, and was meant to be a catch all for all the tweets that were not relevant. Tweets asking for donations to different organizations like the Red Cross and most news sources fell into this category.

The next code that we used was urgency. Guided by Morales (2010), we believe that calls for help generally include some type of urgency, information about where the person is who needs help, and some type of call to action. In the low cues Twitter environment, Morales explains that urgency can be expressed through a variety of different textual elements including "exclamation marks, direct calls for 'help!'," as well as the use of key verbs, or calls to action, such as "'rescue' and 'need'" (2010, p. 39). While our code did not specifically call out these phrases in the description, Morales' (2010) ideas inspired our ultimate categories.

In our initial round of team coding, we have three levels of urgency, which were not urgent, somewhat urgent, and highly urgent. Only tweets that were classified as relevant based on Paul's (2015) categories of request, report, and reaction were then examined and coded for urgency. Highly urgent was defined as "Tweet that suggests an immediate action needs to be taken," and included many examples, which were "Requests for immediate help, material support, and/or information that pertains to someone's livelihood and/or ability to survive. Reports of damage that require immediate attention because it threatens someone's livelihood or ability to survive. Reports of condition changes that impact someone's livelihood or ability to survive and/or require immediate action. Reports of life threatening injuries or deaths." Somewhat urgent was defined as "Tweet that suggests that something may need to be done or some action may need to be taken (but not immediately as no lives are put in immediate danger),"and included the following examples, "For example: Reports of damage will require attention, but it does not need to be fixed immediately. Reports of condition changes that create worry, but do not require immediate action. Reactions about emergency response officials or community efforts that suggest worry, but do not require immediate action or attention." The last initial urgency category was not urgent and was defined as a "Tweet that does not require a response or action by anyone or anything." The examples associated with this category were "General reports or reactions about Hurricane Harvey that do not require any attention or action."

Eventually, these coding categories were refined and distilled down to two categories: urgent and not urgent, or the sake of making it more clear and simple for MTurk workers to be able to classify tweets. This simply asked the user to determine if the tweet "is suggesting that that someone's life in danger?" with a response of yes or no. This definition was purposely left open for interpretation with regards to the one being impacted or in danger, as we wanted to include situations when people are tweeting about things for themselves or on behalf of others. Ultimately, by having a binary classifier, instead of a middle group, tweets were able to be more easily be assigned to a category.

In this study, we chose to solely focus on binary classification, that is, classifying tweets as relevant or not relevant and urgent or not urgent. In this scheme, tweets that were considered to be requests, reports, or reactions were all labeled as 'relevant' and those that weren't in these categories were labeled 'not relevant'; similarly, very urgent and somewhat urgent tweets were simply labeled ‘urgent’ and the rest of the tweets labeled ‘not urgent.’ We focused on binary classification since it is a simpler problem and one we deemed appropriate for determining which types of models generally perform well as relevance and urgency classifiers. A follow-up study could then focus on tuning the selected models to work well in a multi-class setting.

Human Coding

After the data was pre-processed and filtered, we, as the research team, met and hand-coded tweets. For the purpose of coding, single tweets were considered the unit-of-analysis. Because tweets, in essence, are microblogs with short bursts of information, they serve as an ideal unit-of-analysis for classification and research (Ovadia, 2009).

We followed Robertson et al.’s (in press) approach and categorized textual tweets based on relevance per Paul’s (2015) typology of social media posts during disasters, which, as discussed above, is comprised based in the categories of request, report, reactions, or none. Any tweets that were classified into a “request,” “report,” or “reaction,” were considered relevant and further examined for the level of urgency (not urgent, somewhat urgent, and highly urgent).

In the coding process, we all met as a team and coded 150 tweets together to establish a baseline procedure. After, we split into three pairs and each person in the pair independently coded around 445 tweets. In total, our team coded 1,489 of the 2,000 randomly-selected tweets and we had a strong inter-coder reliability of 95.6%. These 1,489 tweets were used for our preliminary model selection.

Overview of Model Performance Metrics

When assessing the performance of machine learning models, only recording accuracy presents an incomplete picture. For example, if a rare disease only occurs in 0.01% of the population, a naive detection system that predicts “not sick” for everyone is over 99% accurate but clearly undesirable since it completely misses the people who are sick. We thus evaluate our models using metrics that quantify both accuracy and how well the model captures truly positive examples. The metrics are accuracy, precision, recall, f1, and area under curve of the receiving operator characteristic (AUC). Accuracy measures the fraction of data that is correctly classified, precision measures the fraction of points labeled positive that are truly positive, recall measures the fraction of positive data that is classified as positive, and f1 is a metric with values between 0 and 1 that depends on both precision and recall. The equations of these metrics are shown below.

$$accuracy \ = \ correct \, / \, total$$

$$precision \ = \ true\,positive \, / \, (true\,positive \ + \ false\,positive)$$

$$recall \ = \ true\,positive \, / \, (true\,positive \ + \ true\,negative)$$

$$f1 \ = \ 2 \cdot precision \cdot recall \, / \, (precision \ + \ recall)$$

The AUC metric measures the overall tradeoff between the false positive and true positive rates of a particular model. Binary classification works by taking some real-valued output of a classifier and converting it into a class prediction based on some threshold value. For example, if the output is between 0 and 1, the threshold value may be 0.5 so that outputs smaller than 0.5 are predicted to be in the negative class and those above 0.5 may be predicted to be in the positive class. As the threshold value is varied, the true and false positive rates of the classifier change, and a plot of the (false positive rate, true positive rate) points generate the receiver operating characteristic (ROC) curve. The area under this curve (AUC) can take on values between 0 and 1 inclusive, and a larger AUC indicates a better classifier.

Although a good model scores well in all five metrics, we prioritize recall since from the perspective of a first responder, it is preferable to sift through some false positive tweets rather than entirely miss many tweets from people who truly do need help or have interesting comments regarding the impact of the storm. We also prioritize f1 since it is a simple formula that captures both precision and recall.

Finally, we measure these metrics for each model using k-fold cross validation. K-fold cross validation works by randomly partitioning the dataset into k subsets and for each subset, training the model on the other k-1 subsets and measuring the metrics on the single chosen test subset. This procedure is done k times by selecting each of the subsets as the test set, and the k values for each metric are averaged. The benefit of cross validation is that since each metric is calculated as an average over many train/test partitions of the data, the measurements have a smaller variance than if they were measured from a single split of the data.

Preliminary Model Selection (Binary Classification)

One of the biggest challenges in developing a machine learning model is the vast number of models available to choose from as well as the numerous hyperparameters to tune in each model. To simplify this task, we first restricted ourselves to the binary classification problem, that is, only classifying a tweet as relevant or irrelevant and urgent or not urgent. We then used the Scikit-learn machine learning package (Pedregosa et al, 2011) to perform k-fold cross-validation (with k = 10) with numerous common classification models and recorded their average accuracy, precision, recall, and f1 across all 10 folds. The suite of classifiers that we tried in bulk are listed below:

- K-nearest neighbors
- Decision tree
- Ensemble methods: AdaBoost (base estimator decision tree), Random forests, Extremely randomized trees
- Support vector machine (linear, polynomial, rbf kernels)
- Gaussian naive Bayes
- Multilayer perceptron

Of these models, AdaBoost, multilayer perceptron, and Gaussian Naive Bayes (displayed in Figure 1) had significantly higher f1 and recall values than the other classifiers for both relevance and urgency, so we decided to narrow our model exploration to those three and a voting classifier consisting of these three models. The performance of these four models, with respect to both binary relevance and urgency classification, is displayed in Table 2.

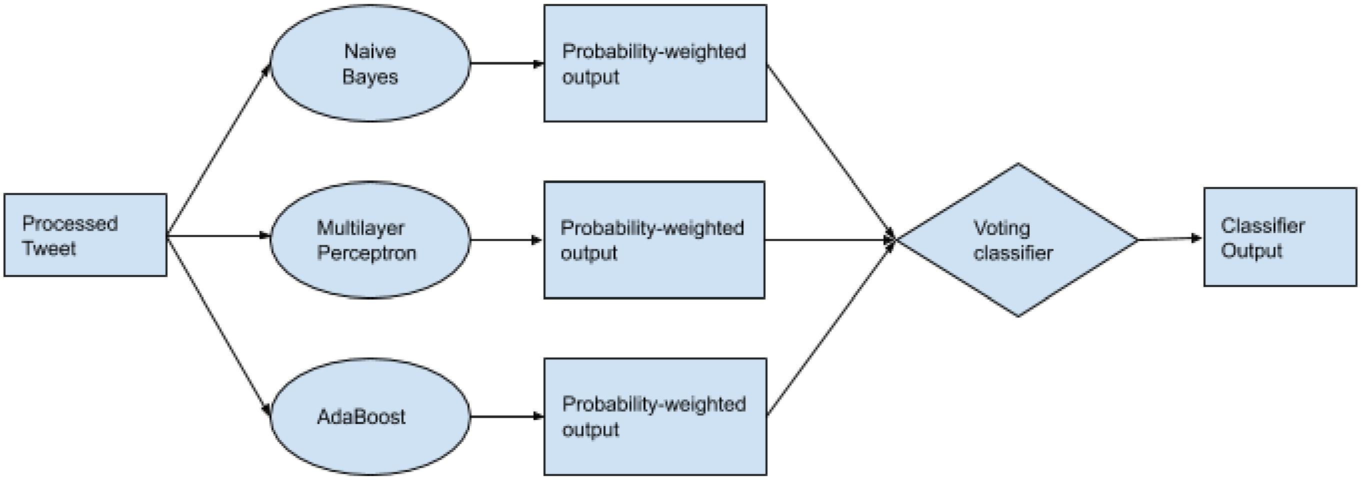

Figure 1: Diagram of how the voting classifier works. For each output, each component model outputs its prediction along with its calculated probability that the prediction is correct. The voting classifier weights each model's prediction with the associated probability and outputs the majority vote.

Classification Results

Method	**F1**	**Precision**	**Recall**	**Accuracy**	**AUC**
Naive Bayes (R)	0.53 ± 0.03	0.43 ± 0.03	**0.68 ± 0.06**	0.69 ± 0.03	0.75
Voting (R)	**0.55 ± 0.04**	0.50 ± 0.05	0.62 ± 0.06	0.74 ± 0.03	0.78
MLP (R)	0.53 ± 0.06	**0.59 ± 0.07**	0.49 ± 0.07	**0.78 ± 0.03**	**0.79**
AdaBoost (R)	0.45 ± 0.08	0.55 ± 0.07	0.38 ± 0.09	0.76 ± 0.03	0.75
Naive Bayes (U)	0.36 ± 0.05	0.26 ± 0.04	**0.61 ± 0.07**	0.77 ± 0.03	**0.78**
Voting (U)	**0.40 ± 0.09**	0.34 ± 0.10	0.48 ± 0.10	0.84 ± 0.04	**0.78**
MLP (U)	0.23 ± 0.10	0.42 ± 0.22	0.17 ± 0.08	0.88 ± 0.02	0.76
AdaBoost (U)	0.27 ± 0.10	**0.42 ± 0.12**	0.21 ± 0.09	**0.89 ± 0.02**	0.73

Table 2: Performance of top four models on preliminary 1,489 tweets with standard deviations. Models marked with (R) are binary relevance classifiers and those marked with (U) are urgency classifiers.

MTurk Coding and Dataset Balancing

One of the shortcomings with the original hand-labeled was the class imbalance: only 25.6% of the tweets are relevant and only 13.6% of the tweets are urgent. This led many of the weaker models to almost naively output a negative prediction, resulting in accuracies close to the proportion of negative tweets in the dataset. This behavior results in decent accuracies but at the cost of poor recall (and f1), since almost none of the relevant or urgent tweets are detected by the models. We thus conjectured that providing a more balanced dataset to the classifiers would prevent this pathological behavior and result in much better recall and f1, even for the four models shown above.

To obtain a balanced dataset for relevance, we decided to submit 3,900 tweets to Amazon's MTurk for rapid, crowd-sourced labeling. To make sure that the dataset submitted to MTurk would have a higher percentage of relevant and urgent tweets, we took tweets from the original dataset of 1.4 million minus the 1,489 that were hand-labeled and ran them sequentially through the multilayer perceptron classifier trained on the 1,489 tweets until 2,400 tweets were encountered that the classifier predicted to be positive and 1,600 were encountered that the classifier predicted to be negative. The multilayer perceptron was chosen for this filtering process since it had the highest recall of all the top four preliminary classifiers. Before submitting 3,900 tweets to MTurk, 100 of the 4,000 tweets in the filtered dataset were used in trial runs to ensure MTurk functioned properly and instructions were understood by MTurk workers.

Our first deployment to MTurk workers for coding included 100 tweets as Human Intelligence Tasks (HITs), which are small tasks that individuals signed up on MTurk can complete for a few cents (Little, Chilton, Goldman, & Miller, 2009). For the first HIT, we created an MTurk survey to randomize the tweets and have two people code each tweet. A worker was asked to read the long descriptions for relevance, which were displayed on the top of the survey, and were then asked to code a tweet. After, they were presented with binary coding categories for urgency (urgency and not urgent). MTurk workers were compensated $.03 per coded tweet. The worker was able to continue coding tweets if they would like, which simply being able to click the yes button. On average, hits took approximately 30 seconds to complete. After looking at the preliminary results and analyzing the amount of time workers spent, as well as the rating the workers had, it was clear that most people were not experienced MTurk workers and they did not spend time fully reading the categories. As a result, we made a number of changes to the format of the MTurk task with the goal of making it more user-friendly to the MTurk workers.

For the second coding deployment on MTurk, the coding categories were drastically shortened to the definitions discussed above for relevance and urgency. In addition to using shortened and simplified descriptions, the coding descriptions were removed from the top of the survey and were placed as answer responses under each tweet to increase the likelihood that the participants would read the code description. We also changed the settings to allow only experienced MTurk workers to complete the HITs, called MTurk Masters. MTurk Masters ensures that only coders who have a very high approval rate will receive our HITs.

We tested these changes by releasing the same 100 tweets to be coded by at least three MTurk Masters each. Coders were compensated $.02 per tweet. The results were more promising and seemed more accurate, but took far too long (around a day), so we decided to increase the rate to $0.05 for each of the first 1,900 tweets, which we split into three batches of size 400, 500, and 500. Each of these batches finished in around three hours, which was much faster than expected, so we used $0.04 for each of the next 2,000 tweets, published in four batches of 500 each. These each took a similar amount of time, totaling around three to four hours to complete.

Results

Improved Non-neural Binary relevance Classifiers

After generating the final dataset, we conducted 10-fold cross-validation with the suite of classifiers as done with the preliminary dataset and as expected, all the models performed much better across all four metrics. Even the support vector machine (SVM) performed much better than before with f1 and recall scores comparable to the top four classifiers selected before. Further experimentation showed that using 100-dimensional word embeddings slightly improved all models' metrics and that using 200-dimensional vectors did not significantly change the results. Thus, from then on, all models were trained using 100-dimensional word embeddings. Table 3 displays the performance of the top four classifiers selected before and the SVM with respect to relevance.

Table 3

Method	**F1**	**Precision**	**Recall**	**Accuracy**	**AUC**
Naive Bayes	0.68 ± 0.02	0.60 ± 0.02	**0.78 ± 0.04**	0.63 ± 0.03	0.71
Voting	**0.69 ± 0.03**	0.63 ± 0.03	0.77 ± 0.04	0.66 ± 0.03	**0.75**
MLP	0.67 ± 0.02	0.68 ± 0.02	0.67 ± 0.04	**0.68 ± 0.02**	0.74

AdaBoost	0.66 ± 0.02	0.65 ± 0.02	0.67 ± 0.03	0.65 ± 0.02	0.71
SVM	0.68 ± 0.02	**0.69 ± 0.02**	0.66 ± 0.03	**0.68 ± 0.02**	0.74

Table 3: Performance metrics (with standard deviations) of binary relevance classifiers on the balanced dataset.

One of the characteristics of our dataset that posed the biggest challenge when developing the classifiers was the nuance in the labels. Each relevance class had very specific constraints that were not immediately obvious to experienced labelers upon examining a tweet. For example, only personal requests and requests on behalf of specific people were considered significant requests; donation requests were excluded. Based on this nuance, examination of MTurk labels, and the extensive discussion some of the tweets required to properly classify, we concluded that the final dataset we created was probably pretty noisy and hypothesized that addressing this with dimensionality-reduction and regularization techniques would slightly improve the performance of the top five selected classifiers.

The first technique we explored was recursive feature elimination with cross-validation (RFECV) as implemented in Scikit-learn using logistic regression, SVM with a linear kernel, and decision tree (Guyon et al., 2002). RFECV works by first running k-fold cross-validation on a dataset to calculate some metric, removing the feature whose coefficient or feature importance attribute in the final model is the smallest, and rerunning the cross-validation with the smaller feature set. This is repeated until a single feature is left, and the "optimal" feature subset is that which produces the highest cross-validated metric value. RFECV only works with models that assign coefficients or feature importances while training, which is why we specifically chose the above models. Running RFECV with ten folds yielded the best performance when done with logistic regression, and the results are shown in Table 4. For naive bayes and the voting classifier, applying RFECV lowered recall by a percentage point but the improvements in both accuracy and precision more than offset this reduction to result in higher f1 scores overall.

Table 4

Method	**F1**	**Precision**	**Recall**	**Accuracy**	**AUC**
Naive Bayes	0.68 ± 0.02	0.60 ± 0.02	**0.77 ± 0.04**	0.63 ± 0.02	0.71
Voting	**0.70 ± 0.02**	0.64 ± 0.02	0.76 ± 0.03	0.67 ± 0.03	**0.75**
MLP	0.69 ± 0.02	0.69 ± 0.02	0.69 ± 0.03	**0.69 ± 0.01**	**0.75**
AdaBoost	0.66 ± 0.02	0.65 ± 0.03	0.68 ± 0.03	0.66 ± 0.02	0.71
SVM	0.69 ± 0.02	**0.70 ± 0.02**	0.68 ± 0.02	**0.69 ± 0.02**	**0.75**

Table 4: Performance of binary relevance classifiers with features pruned using RFECV with logistic regression

The next technique we explored was principal component analysis (PCA), a method of dimensionality reduction which produces an orthonormal basis of the original feature vectors such that the projection of the data onto the first principal component has maximum variance, the projection onto the second component has the second largest variance, and in general, the projection into the nth principal component has the nth largest variance. The benefit of PCA is that since the first few principal components of a dataset capture more of the variance in the dataset than later ones, projecting the dataset onto the first few principal components can still preserve most of the data's variance and retain classifier performance while decreasing the dimensionality of the data and making classifiers less susceptible to overfitting noise. We experimented with projections of the RFECV-pruned features for each tweet onto the first 60, 40, 30, 20, and 10 principal components and found that the 20-dimensional projections were the smallest representation of the dataset that still preserved the performance of the five classifiers. The results are shown in Table 5. Although SVM performance suffered as evident by the 2% decrease in f1, our final model optimization rectified that.

Table 5

Method	**F1**	**Precision**	**Recall**	**Accuracy**	**AUC**
Naive Bayes	0.69 ± 0.02	0.62 ± 0.01	**0.78 ± 0.04**	0.65 ± 0.02	0.72
Voting	**0.70 ± 0.02**	0.66 ± 0.03	0.74 ± 0.02	0.68 ± 0.02	**0.76**
MLP	0.69 ± 0.01	**0.68 ± 0.03**	0.70 ± 0.01	**0.69 ± 0.02**	**0.76**
AdaBoost	0.67 ± 0.02	0.66 ± 0.02	0.68 ± 0.02	0.67 ± 0.02	0.73
SVM	0.67 ± 0.02	0.67 ± 0.03	0.67 ± 0.02	0.67 ± 0.03	0.74

Table 5: Performance of binary relevance classifiers with features first pruned with RFECV using logistic regression and then projected onto their first 20 principal components.

The final fine-tuning step was to find the combination of hyperparameters for each model that maximizes f1 using grid search. Grid search is a method in which a list of candidate values is provided for each hyperparameter of a model, and an instance of the model is trained using cross-validation for every combination of hyperparameter values to find the best combination. We used Scikit-learn's GridSearchCV function with ten folds for this and arrived at the following model configurations, with parameter names corresponding to those in the Scikit-learn implementations of the models. Only hyperparameters not equal to their default values in Scikit-learn version 0.21.1 are displayed in Table 6.

Table 6

Model	**Hyperparameter Values**
Gaussian Naive Bayes	No non-default parameters
Multilayer Perceptron (MLP)	alpha = 0.01
AdaBoost	base_estimator = DecisionTreeClassifier(max_depth=2), n_estimators = 28, learning_rate = 0.35
Support Vector Machine (SVM)	C = 10000, gamma = 0.01
Voting Classifier	Individual learners have same parameters as listed above, voting = "soft"

Table 6: Tuned hyperparameter values of models referred to in Table 5

Table 7 shows the results of the five models on the data projected into the first 20 principal components but after hyperparameter selection with grid search. Notice that the f1 and recall of the tuned SVM are significantly higher than those of the default SVM depicted in Table 5.

Table 7

Method	**F1**	**Precision**	**Recall**	**Accuracy**	**AUC**
Naive Bayes	0.69 ± 0.02	0.62 ± 0.02	**0.78 ± 0.03**	0.65 ± 0.02	0.72
Voting	**0.70 ± 0.02**	0.67 ± 0.02	0.73 ± 0.02	0.68 ± 0.02	**0.76**
MLP	0.69 ± 0.02	**0.68 ± 0.02**	0.71 ± 0.02	**0.69 ± 0.02**	**0.76**
AdaBoost	0.67 ± 0.02	0.66 ± 0.03	0.69 ± 0.03	0.67 ± 0.02	0.73
SVM	**0.70 ± 0.02**	0.67 ± 0.03	0.73 ± 0.02	**0.69 ± 0.03**	0.75

Table 7: Performance of binary relevance classifiers with the same features as in Table 5 but with parameters tuned with grid search.

relevance Classifier Improvement with Deep Convolutional Neural Networks

The natural next step of our exploration of relevance classifiers is to build a model that does not use just the average word embeddings of a tweet—which does not capture the spatial relationship between different words in a tweet—but rather the individual word embeddings arranged in the correct order. An important observation is that the matrix with rows corresponding to word embeddings in correct order can be subject to a convolutional transformation much like a grayscale image as shown in Figure 2. The only difference is that the element in row i and column j in the image is a single grayscale pixel value and the corresponding element in the tweet matrix is the j^{th} element of the word embedding corresponding to the i^{th} word in the tweet.

Figure 2

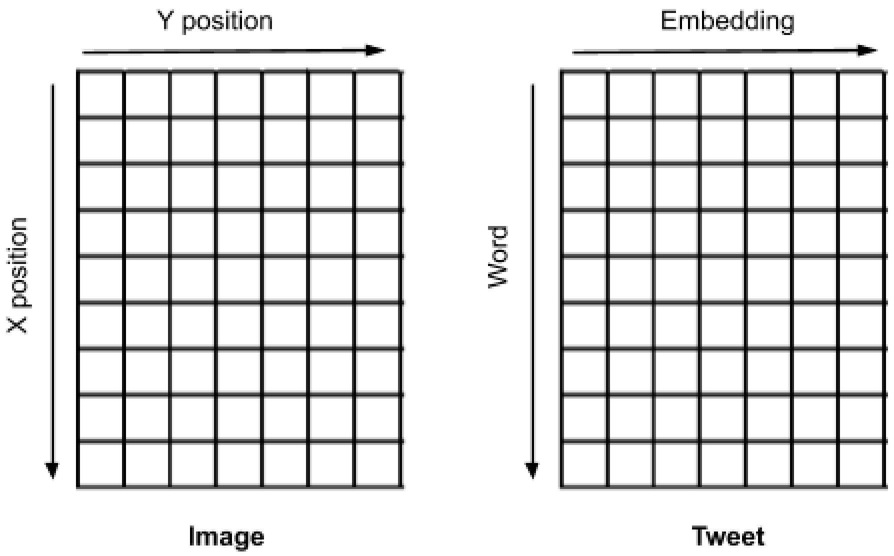

Figure 2: Interpretation of a tweet as an image-like structure with rows representing word embeddings. The only difference is that images may have a third dimension specifying each pixel value.

The convolutional neural network (CNN) we modeled with was the architecture presented by Kim (2014), which consists of a convolutional layer with a fixed width equal to the length of each embedding vector but a variable word window size, a max-pooling layer, and a fully-connected layer with dropout and softmax output. The variable parameters in this architecture were the exact convolutional window sizes used (e.g. [1,2,3], [1], etc.), the number of convolutional filters of each window size, and dropout rate. Variable parameters regarding the training of the neural network were learning rate, batch size, and number of epochs (we fixed the optimizer to Adam for convenience). We implemented the neural network in an open-source deep learning library called PyTorch (Paszke et al., 2017).

Before feeding tweets into the CNN, we padded them with all-one vectors to have the same word count as the largest tweet in the dataset. Grid searching over the hyperparameters of the CNN and training parameters gave us the following optimal configuration:

Table 8

Hyperparameter	**Optimal Value**
Window sizes	[1, 1, 1]
Number of filters of each window size	400
Dropout rate	0.75
Batch size	50
Number of epochs	200
Learning rate	0.0001

Table 8: Optimal CNN hyperparameters

This combination of parameters yielded an f1 of **0.72**, precision of **0.71**, recall of **0.73**, and accuracy of **0.72**. This combination of performance metrics outperforms all the non-neural models developed.

Binary Urgency Classifiers with Balanced Data

The biggest issue with attempting to build an urgency classifier was the heavy imbalance of the dataset. Less than 200 tweets of the 3,900 submitted to MTurk were labeled as urgent and upon inspection, only 51 of the tweets were truly urgent enough to merit the attention of first responders as per our criteria. With such a small number of urgent tweets, making a balanced dataset with sufficiently many tweets to build a classifier was difficult, so we settled with a dataset of 153 tweets, a third of which consisted of urgent tweets. We employed the same methods of model selection, feature selection, and PCA that were used to build relevance classifiers. The best-performing models were the same five as in the relevance case. The best-performing combination of preprocessing and feature selection was as follows: run RFECV with a linear SVM to select a subset of the 100 features for each tweet, do not apply PCA, and run grid search on each model to arrive at the hyperparameters shown in Table 9. For brevity, hyperparameters with values that are default in Scikit-learn version 0.21.1 are not listed. Table 10 shows the performance of these tuned models.

Table 9

Model	**Hyperparameter Values**
Gaussian Naive Bayes	No non-default parameters
Multilayer Perceptron (MLP)	alpha = 0.001
AdaBoost	n_estimators = 100, learning_rate = 0.25
Support Vector Machine (SVM)	kernel = “linear”, C = 1, gamma = 0.001
Voting Classifier	Individual learners have same parameters as listed above, voting = “soft”

Table 9: Optimal hyperparameters of models referred to in Table 10

Table 10

Method	**F1**	**Precision**	**Recall**	**Accuracy**	**AUC**
Naive Bayes	0.62 ± 0.13	0.72 ± 0.15	0.55 ± 0.11	0.78 ± 0.08	0.81
Voting	0.72 ± 0.07	0.80 ± 0.09	0.67 ± 0.10	0.83 ± 0.04	0.89

MLP	0.79 ± 0.06	**0.85 ± 0.06**	0.75 ± 0.11	**0.87 ± 0.02**	**0.91**
AdaBoost	0.61 ± 0.12	0.68 ± 0.11	0.59 ± 0.19	0.76 ± 0.05	0.81
SVM	**0.80 ± 0.02**	0.79 ± 0.04	**0.80 ± 0.03**	0.86 ± 0.02	**0.91**

Table 10: Performance of tuned binary urgency classifiers on features pruned using RFECV with linear SVM

Binary Urgency Classifiers with Unbalanced Data

Even though the SVM and perceptron both performed well on the urgency data, the small dataset size suggests that the good performance could be attributed to overfitting. We thus decided to explore methods of training classifiers on a larger but highly imbalanced urgency dataset. Such techniques would be useful in practice since only a small percentage of disaster tweets are relevant to first responders. Additionally, an initially weak classifier could be run on a very large unlabeled dataset to pick a slightly more balanced subset of tweets (much like was done to generate the final relevance dataset using the preliminary perceptron classifier) which could be used to train a slightly better classifier, which in turn could be used to pick an even more balanced subset from the remaining unlabeled tweets to train an even better classifier, and so on. This bootstrapping method could thus be used to generate better and better classifiers using larger and more evenly split datasets as more unlabeled data is continually collected.

The unbalanced dataset we created consisted of 510 tweets, 10% of which were urgent. The performance of the five models after data preprocessing and model selection are shown in Table 12. The optimal hyperparameters are displayed in Table 11:

Table 11

Model	**Hyperparameter Values**
Gaussian Naive Bayes	No non-default parameters
Multilayer Perceptron (MLP)	alpha = 0.0001
AdaBoost	n_estimators = 50, learning_rate = 1
Support Vector Machine (SVM)	kernel = “rbf”, C = 10, gamma = 1
Voting Classifier	Individual learners have same parameters as listed above, voting = “soft”

Table 11: Optimal hyperparameters for models referred to in Table 12

As expected, the models seem to overpredict “not-urgent” to achieve high accuracy at the expense of recall.

Table 12

Method	**F1**	**Precision**	**Recall**	**Accuracy**	**AUC**
Naive Bayes	0.26 ± 0.17	0.54 ± 0.25	0.18 ± 0.13	0.90 ± 0.02	0.83
Voting	0.26 ± 0.17	0.54 ± 0.25	0.18 ± 0.13	0.90 ± 0.02	0.83
MLP	**0.49 ± 0.03**	**0.75 ± 0.18**	0.37 ± 0.03	**0.92 ± 0.01**	**0.86**
AdaBoost	0.32 ± 0.06	0.37 ± 0.04	0.29 ± 0.08	0.88 ± 0.01	0.75
SVM	0.47 ± 0.10	0.54 ± 0.08	**0.43 ± 0.15**	0.91 ± 0.01	0.83

Table 12: Performance of tuned binary urgency classifiers on an unbalanced dataset with the same preprocessing as listed in Table 10.

To create a more balanced version of the provided dataset, for each training dataset in cross-validation, we randomly sampled from just the set of urgent tweets with replacement until the number of sampled urgent tweets equaled the number of non-urgent tweets, a technique known as oversampling. The model was then trained on this artificially balanced dataset. This additional balancing step was implemented by creating a custom model class in Scikit-learn that takes a base model (like LogisticRegression or AdaBoost) as a parameter and overrides the "fit" method to conduct data balancing as a preprocessing step. We then conducted the same feature-selection and model-selection steps as before and concluded that using RFECV with LogisticRegression followed by projection onto the first 60 principal components generated the best performance. As usual, the top five highest-performing classifiers with respect to f1 and recall were the same as before. The results are shown in Table 14. The optimal hyperparameters tuned for each model are shown in Table 13.

Table 13

Model	**Hyperparameter Values**
Gaussian Naive Bayes	No non-default parameters
Multilayer Perceptron (MLP)	alpha = 0.001
AdaBoost	n_estimators =100, learning_rate = 0.25
Support Vector Machine (SVM)	kernel = "linear", C = 10, gamma = 1
Voting Classifier	Individual learners have same parameters as listed above, voting = "soft"

Table 13: Optimal hyperparameters for models referred to in Table 14

Table 14

Method	**F1**	**Precision**	**Recall**	**Accuracy**	**AUC**
Naive Bayes	0.39 ± 0.03	0.29 ± 0.03	0.63 ± 0.06	0.81 ± 0.02	0.82
Voting	0.46 ± 0.03	0.39 ± 0.06	0.57 ± 0.06	0.86 ± 0.02	**0.84**
MLP	**0.50 ± 0.01**	**0.47 ± 0.05**	0.57 ± 0.10	**0.89 ± 0.02**	**0.84**
AdaBoost	0.33 ± 0.07	0.28 ± 0.02	0.43 ± 0.17	0.84 ± 0.01	0.75
SVM	0.40 ± 0.05	0.30 ± 0.05	**0.61 ± 0.06**	0.82 ± 0.03	0.82

Table 14: Performance of binary urgency classifiers on an unbalanced dataset with features first pruned with RFECV using logistic regression and then projected onto their first 60 principal components. Training data was artificially balanced with oversampling prior to fitting the models.

One shortcoming of naive oversampling is that only existing data points are added over and over again to the final training set; models trained on such a dataset are highly prone to overfitting (Chawla et al., 2002). We attempted to alleviate this problem by experimenting with more sophisticated oversampling methods, namely SMOTE and SVM-SMOTE (Chawla et al., 2002; Nguyen et al., 2009). SMOTE adds "synthetic" points of the minority class to the dataset by perturbing each of the points it oversamples by randomly choosing a point on the line segment connecting each data point to one of its k nearest neighbors (where k is a hyperparameter). SVM-SMOTE is a more sophisticated version of SMOTE that only oversamples the data points that it determines are close to a decision boundary by first fitting an

SVM. We developed models using both of the techniques (with k = 5) and determined that SVM-SMOTE produced better results across all five metrics, so only these results are reported in Table 16. The optimal preprocessing steps are the same as in the naive oversampling case, and the optimal hyperparameters discovered with grid search are displayed in Table 15.

Table 15

Model	**Hyperparameter Values**
Gaussian Naive Bayes	No non-default parameters
Multilayer Perceptron (MLP)	alpha = 0.01
AdaBoost	n_estimators = 100, learning_rate = 0.25
Support Vector Machine (SVM)	kernel = “linear”, C = 1, gamma = 0.001
Voting Classifier	Individual learners have same parameters as listed above, voting = “soft”

Table 15: Optimal hyperparameters for models referred to in Table 16

As expected, balancing the dataset with slight variations of the urgent tweets instead of their exact copies did improve the f1 and recall of all the models.

Table 16

Method	**F1**	**Precision**	**Recall**	**Accuracy**	**AUC**
Naive Bayes	0.40 ± 0.10	0.39 ± 0.04	0.43 ± 0.17	0.88 ± 0.00	0.77
Voting	0.46 ± 0.10	**0.58 ± 0.11**	0.41 ± 0.13	**0.91 ± 0.02**	0.83
MLP	**0.51 ± 0.02**	0.43 ± 0.02	**0.63 ± 0.0**7	0.88 ± 0.01	0.85

AdaBoost	0.38 ± 0.06	0.35 ± 0.08	0.43 ± 0.03	0.86 ± 0.03	0.78
SVM	0.49 ± 0.02	0.42 ± 0.03	0.61 ± 0.07	0.87 ± 0.02	**0.87**

Table 16: Performance of binary urgency classifiers on an unbalanced dataset with features first pruned and projected as described in Table 14. Training data was artificially balanced with SVM-SMOTE prior to fitting the models.

Discussion

By examining ‘calls for help’ on Twitter through machine learning methods, this research first identified how Twitter is used during disasters like Hurricane Harvey. Next, this research determined important attributes of calls for help through automatic computerized categorization. Furthermore, we developed machine learning models to classify tweets related to Hurricane Harvey as relevant and urgent.

We now compare the performance of our best relevance and urgency classifiers to the state of the art non-neural and neural classifiers of disaster-related tweets, namely Tweedr and the Crisis Event Extraction Service (CREES), developed by Ashktorab et al. (2014) and Burel et al. (2018) respectively. Tweedr is a system to extract tweets relevant to natural disasters in real time, and one of its components is a relevance classifier that uses one of Naive Bayes, logistic regression, k-nearest neighbors, decision trees, or latent Dirichlet allocation. CREES uses the same CNN architecture by Kim (2014) to classify tweets by relevance to disaster, type of disaster, and type of information.

All five non-neural models we trained on the balanced relevance dataset have higher f1 scores than the Tweedr classifiers; although our models have lower overall accuracy and precision, they achieve higher f1 with a much higher recall. We believe that a much higher recall at the expense of precision and accuracy is merited since from the point of view of a first responder looking for calls for help or information about a disaster, it is better to capture more of the relevant tweets while suffering the inconvenience of reading through false positives than to miss most of the relevant tweets altogether. The convolutional neural network similarly has better f1 and recall than all the Tweedr classifiers but also maintains a higher accuracy (of 72%), thus attaining a higher f1 than all five of our non-neural models. All of our models including the CNN achieved lower precision, recall, and f1 scores than the CREES classifiers with respect to relevance. We believe our performance was lower despite using the same CNN architecture since our classifiers were trained to identify tweets already related to a disaster as relevant according to specific requirements (request for help or information, report of disaster's effects, or response to first responders' efforts), but the CREES classifiers only identified whether a tweet was relevant to crises at all; our chosen classification task was more nuanced and involved distinguishing between finer semantic information in the tweets (e.g. distinguishing between a donation request and request for first response).

Since there is not research on urgency classification of Twitter data, we compare the performance of the urgency classifiers to the Tweedr and CREES relevance classifiers. The MLP and SVM urgency classifiers trained on the small balanced dataset achieved much higher f1 and recall scores than all the Tweedr classifiers and even achieved better accuracy than most of them; however, our dataset was small and thus these good results were likely a result of overfitting. As expected, even with SVM-SMOTE oversampling, our classifiers trained on the unbalanced dataset showed the pathology of unbalanced training, namely a high accuracy at the cost of recall, though the recall scores were still higher than those of the decision tree and k-nearest neighbor classifiers used in Tweedr. Furthermore, the MLP trained with SVM-SMOTE had a decent recall of 63%, meaning that it would be a good candidate to generate more balanced datasets from new unlabeled data and conduct the "bootstrapping" method discussed earlier.

Our relevance classifier can ultimately be used to inform first responders how people are using Twitter to request help in life-threatening disaster situations by programmatically capturing a high percentage of the tweets that are relevant to them. Although a significant percentage of the captured tweets would be false positives, it is more important that calls for help and reports of damage would not be missed by the classifier. Using our relevance classifiers would allow first responders to go through a much larger number of tweets than they would be able to manually. Our urgency classifier still requires more work to improve recall and f1, and the main avenue of improvement would likely be a larger and more balanced dataset.

Theoretical and Practical Contributions

This study makes several contributions in the field of disaster tweet classification. While Twitter has been found to be an instrumental communication tool during disasters to share news and information, document experiences, and connect with others, perhaps the most impactful and meaningful finding is that Twitter has been used to request and coordinate lifesaving rescue efforts (Robertson et al., in press; David et al., 2016, Rhodan, 2017; Yang et al., 2017). This study examines a particular instance of this phenomenon, namely calls for help during Hurricane

Harvey, and presents machine learning models that classify tweets related to the disaster as relevant or irrelevant and urgent or not urgent. The best relevance classifiers, namely the CNN and SVM, both achieve much higher recall than the Tweedr classifiers, meaning that they capture a higher percentage of the tweets relevant to first responders. While these classifiers do not perform as well as the CNN of identical architecture used in CREES, our classifier task is more robust and complicated (detect tweets relevant to natural disasters vs. detect relevant tweets among those that are all related to natural disasters). Although the urgency classifier did not do as well as the Tweedr or CREES, we still managed to improve the overall model by presenting SVM-SMOTE oversampling as a way to improve the f1 and recall of classifiers trained on heavily unbalanced datasets. Most notably, to our knowledge, this is the first paper that attempts to classify tweets based on urgency in addition to topic.

Limitations and Future Directions

As with any study, there are limitations present. This study only included 4,000 coded tweets to train and inform the computational model, which is less than 1% of the available tweets. Our neural model, which performed markedly better than the non-neural models, would likely have benefited from a very large dataset as deep learning models generally perform well with very large datasets; for example, the CREES neural network was trained on a dataset of 28,000 tweets. A future step could be to submit more tweets to MTurk and see if the performance of our existing neural network approaches that of CREES. Another approach to potentially improve the performance of the neural model would be to add a bidirectional recurrent layer whose output is concatenated to the word embeddings before the convolutional layer is applied (Lai et al., 2015). This technique primarily improves the classification rate of documents longer than tweets, but it is still worth experimenting with in this context.

A major hurdle in the development of both the relevance and urgency classifier was the class imbalance problem; despite the sizeable dataset, the vast majority of tweets were not relevant and even fewer were urgent. In other words, we had few positive examples to train on, and early models learned to virtually always predict not relevant and not urgent, resulting in high accuracy scores (close to the percentage of negative examples in the dataset), but low recall and f1 scores. We were able to partially alleviate this problem for relevance classification by using the MLP trained on the preliminary dataset to sample the 4,000 tweets submitted to MTurk such that at over half were predicted to be relevant. Due to the extreme sparsity of urgent tweets on even this dataset, we resorted to oversampling methods, the most successful of which was SVM-SMOTE. While this helped improve the performance model, it still does not reach Tweedr and CREES. In the future, we could use the bootstrapping method to generate large datasets that are approximately balanced with respect to both urgency and relevance to see if training with a balanced dataset instead of having to simulate one with oversampling would improve urgency classifier performance.

Finally, our classifiers could be made even more useful to first responders if trained in the multi-class setting. In this study, we limited ourselves to the simpler binary classification case for the sake of identifying which types of machine learning models are generally best suited to the classification of disaster tweets. A future study would tune the most promising models we identified in this study to create classifiers that tell first responders not only which tweets are relevant or urgent but what specific category they fall into.

Conclusion

Because Twitter is so readily available in comparison to traditional emergency phone services, it is often used as a way to request help and connect with others during large scale disasters. Using recorded data from Hurricane Harvey, one of the most significant natural disasters in the history of the United States, this study aims to help detect posts on Twitter that are relevant to first responders, namely requests for help, reports of damage, and reactions to other first responders' efforts. Using a variety of neural and non-neural machine learning models trained on 4 thousand unique coded tweets related to Hurricane Harvey, this study developed relevance classifiers which detected relevant tweets at a higher rate than the leading non-neural relevance classifier, Tweedr, but a lower rate than the leading CNN model trained for an easier classification task as part of the CREES system. Although the urgency classifier we trained on the same dataset did not achieve recall or f1 scores competitive with Tweedr or CREES due to class imbalance, this study still presented SVM-SMOTE as a useful oversampling technique to improve the performance of disaster tweet classifiers trained on very imbalanced datasets. We hope that this study can be useful to first responders attempting to quickly identify people in need of rescue, saving more lives during a natural disaster.

References

Appling, S., Briscoe, E., Ediger, D., Poovey, J., & McColl, R. (2014). Deriving disaster-related information from social media. In Proc. 1st KDD Workshop Learn. Emergencies Social Inf. KDD (KDD-LESI) (pp. 16-22).

Ashktorab, Z., Brown, C., Nandi, M., & Culotta, A. (2014, May). Tweedr: Mining Twitter to inform disaster response. In ISCRAM.

Blake, E. S., & Zelinsky, D. A. (2018, May 8). National hurricane center tropical cyclone report: Hurricane Harvey. Retrieved from https://www.nhc.noaa.gov/data/tcr/AL092017_ Harvey.pdf

Burel, G., & Alani, H. (2018). Crisis Event Extraction Service (CREES)-Automatic Detection and Classification of Crisis-related Content on Social Media.

Dalinina, R. (2017, October 10). Word Embeddings: An NLP Crash Course. Retrieved from https://www.datascience.com/resources/notebooks/word-embeddings-in-python

David, C. C., Ong, J. C., & Legara, E. F. T. (2016). Tweeting Supertyphoon Haiyan: Evolving functions of Twitter during and after a disaster event. *PloS one*, *11*(3), e0150190.

Derczynski, L., Meesters, K., Bontcheva, K., & Maynard, D. (2018). Helping crisis responders find the informative needle in the tweet haystack. arXiv preprint arXiv:1801.09633.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, *3*(Jan), 993-1022.

Carter, L. (2018, July 31). Timeline: Hurricane Harvey brings catastrophic rain, flooding to Gulf Coast. Retrieved from https://www.khou.com/article/news/timeline-hurricane-harvey-brings-catastrophic-rain-flooding-to-gulf-coast/285-579150393

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, *16*, 321-357.

Gallagher, J. J. (2017, September 01). Hurricane Harvey wreaks historic devastation: By the numbers. Retrieved from https://abcnews.go.com/US/hurricane-harvey-wreaks-historic-devastation-numbers/story?id=49529063

Garbe, W. (2014, March 25). SymSpell (Version 6.3) [Computer software]. Retrieved April 15, 2019, from https://github.com/wolfgarbe/SymSpell

Glass, T. A. (2001). Understanding public response to disasters. *Public Health Reports*, *116*(Suppl 2), 69.

Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine learning*, *46*(1-3), 389-422.

He, H., Bai, Y., Garcia, E. A., & Li, S. (2008, June). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)* (pp. 1322-1328). IEEE.

Imran, M., Castillo, C., Lucas, J., Meier, P., & Rogstadius, J. (2014, May). Coordinating human and machine intelligence to classify microblog communications in crises. In ISCRAM.

Imran, M., Elbassuoni, S., Castillo, C., Diaz, F., & Meier, P. (2013, May). Practical extraction of disaster-relevant information from social media. In *Proceedings of the 22nd International*

Conference on World Wide Web (pp. 1021-1024). ACM.

Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Kite. (no date). TweetTokenizer. Retrieved from https://kite.com/python/docs/nltk.tokenize. casual.TweetTokenizer.

Lai, S., Xu, L., Liu, K., & Zhao, J. (2015, February). Recurrent convolutional neural networks for text classification. In *Twenty-ninth AAAI conference on artificial intelligence*.

Li, H., Guevara, N., Herndon, N., Caragea, D., Neppalli, K., Caragea, C., Squicciarini, A. & Tapia, A. H. (2015, May). Twitter Mining for Disaster Response: A Domain Adaptation Approach. In ISCRAM.

Lindsay, B. R. (2011). Social media and disasters: Current uses, future options, and policy considerations. *CRS Report for Congress*.

Little, G., Chilton, L. B., Goldman, M., & Miller, R. C. (2009, June). Turkit: tools for iterative tasks on mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation* (pp. 29-30). ACM.

Mammothb. (2018, August 13). SymSpell (Version 6.3) [Computer software]. Retrieved April 15, 2019, from https://github.com/mammothb/symspellpy

Morales, X. Y. Z. G. (2010). *Networks to the rescue: Tweeting relief and aid during Typhoon Ondoy* (Thesis, Georgetown University).

Morstatter, F., Lubold, N., Pon-Barry, H., Pfeffer, J., & Liu, H. (2014). Finding eyewitness tweets during crises. arXiv preprint arXiv:1403.1773.

Murthy, D. (2012). Towards a sociological understanding of social media: Theorizing Twitter. *Sociology*, *46*(6), 1059-1073

Murthy, D. (2018). *Twitter: Social communication in the Twitter age.* Cambridge: Polity Press.

Murthy, D., Gross, A. J., & McGarry, M. (2016). Visual social media and big data: Interpreting Instagram images posted on Twitter. *Digital Culture & Society*, *2*(2), 113-134.

Murthy, D., & Longwell, S. (2013). Twitter and disasters: The uses of twitter during the 2010 Pakistan floods, *Information Communication & Society,* 16, 6, 837-855. doi: 10.1080/1369118X.2012.696123

Nguyen, H. M., Cooper, E. W., & Kamei, K. (2009, November). Borderline over-sampling for imbalanced data classification. In *Proceedings: Fifth International Workshop on Computational Intelligence & Applications* (Vol. 2009, No. 1, pp. 24-29). IEEE SMC Hiroshima Chapter.

NLP. (2008). Stemming and lemmatization. Retrieved from https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html

NLP. (2008). Tokenization. Retrieved from https://nlp.stanford.edu/IR-book/html/htmledition/tokenization-1.html

NOAA. (2018, January 26). Costliest U.S. tropical cyclones tables updated. Retrieved from https://www.nhc.noaa.gov/news/UpdatedCostliest.pdf

NTLK. (2014). Accessing Text Corpora and Lexical Resources. Retrieved from https://www.nltk.org/book/ch02.html

NLTK Tweet Tokenizer (2015). Retrieved April 29, 2019, from http://www.nltk.org/api/nltk.

tokenize.html

O’Neal, A., Rodgers, B., Segler, J., Murthy, D., Lakuduva, N., Johnson, M., & Stephens, K. (2018, December). Training an Emergency-Response Image Classifier on Signal Data. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)* (pp. 751-756). IEEE.

Ovadia, S. (2009). Exploring the potential of Twitter as a research tool. *Behavioral & Social Sciences Librarian*, *28*(4), 202-205.

Palen, L., & Hughes, A. L. (2010). Social media in disaster communication. In H. Rodríguez, W. Donner, & J. E. Trainor (Eds.), *Handbook of Disaster Research – 2nd Edition* (pp. 497-520). Cham, Switzerland: Springer.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., ... & Lerer, A. (2017). Automatic differentiation in pytorch.

Paul, A. (2015). Identifying relevant information for emergency services from twitter in response to natural disaster. Retrieved from https://eprints.qut.edu.au/89220/1/Avijit_Paul_Thesis.pdf

Peary, B. D., Shaw, R., & Takeuchi, Y. (2012). Utilization of social media in the east Japan earthquake and tsunami and its effectiveness. *Journal of Natural Disaster Science*, *34*(1), 3-18.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. *Journal of machine learning research*, *12*(Oct), 2825-2830.

Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1532-1543).

Pew Research Center (2018, March 1). *Social media use in 2018*. Retrieved from http://www.pewinternet.org/2018/03/01/social-media-use-in-2018/

Phillips, M. E. (2017). Hurricane Harvey Twitter dataset, dataset, 2017-08-18/2017-09-22; (https://digital.library.unt.edu/ark:/67531/metadc993940/: accessed January 29, 2018), University of North Texas Libraries, Digital Library, digital.library.unt.edu

Rehurek, R., & Sojka, P. (2011). Gensim–python framework for vector space modelling. NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic, 3(2).

Rhodan, M. (2017, August 30). Please send help: Hurricane Harvey victims turn to Twitter and Facebook. *Time*. Retrieved from http://time.com/4921961/hurricane-harvey-twitter-facebook-social-media/

Robertson, B. W., Johnson, M., Murthy, D., Smith, W. R., & Stephens, K. K. (in press). Using a combination of human insights and 'deep learning' for real-time disaster communication. *Progress in Disaster Science*.

Smith, W. R., Stephens, K. K., Robertson, B. W., Li, J., & Murthy, D. (2018, May). Social media in citizen-led disaster response: Rescuer roles, coordination challenges, and untapped potential. In *Proceedings of the 15th international ISCRAM conference* (pp. 639-648).

Spence, P. R., Lachlan, K. A., Lin, X., & del Greco, M. (2015). Variability in Twitter content across the stages of a natural disaster: Implications for crisis communication. *Communication Quarterly*, *63*, 171-186.

Stephens, K. K., Robertson, B. W., & Murthy, D. (in press). Throw me a lifeline: Articulating mobile social network dispersion and the social construction of risk in rescue communication, *Mobile Media & Communication.*

Stephens, K. K., Li, J., Robertson, B. W., Smith, W. R. & Murthy, D. (2018). Citizens communicating health information: Urging others in their community to seek help during a flood. In K. Boersma & B. Tomaszewski (Eds.), *Proceedings of the 15th International ISCRAM Conference,* Rochester, NY, May 2018.

Yang, Z., Nguyen, L. H., Stuve, J., Cao, G., & Jin, F. (2017, December). Harvey flooding rescue in social media. In *2017 IEEE International Conference on Big Data (Big Data)* (pp. 2177-2185). IEEE.