

Spring 2019

Prototyping a Social Media Flooding Photo Screening System Based on Deep Learning and Crowdsourcing

Huan Ning

Follow this and additional works at: <https://scholarcommons.sc.edu/etd>

 Part of the [Geography Commons](#)

Recommended Citation

Ning, H.(2019). *Prototyping a Social Media Flooding Photo Screening System Based on Deep Learning and Crowdsourcing*. (Master's thesis). Retrieved from <https://scholarcommons.sc.edu/etd/5253>

This Open Access Thesis is brought to you by Scholar Commons. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of Scholar Commons. For more information, please contact dillarda@mailbox.sc.edu.

PROTOTYPING A SOCIAL MEDIA FLOODING PHOTO SCREENING SYSTEM
BASED ON DEEP LEARNING AND CROWDSOURCING

by

Huan Ning

Bachelor of Engineering
Wuhan University, 2006

Submitted in Partial Fulfillment of the Requirements

For the Degree of Master of Science in

Geography

College of Arts and Sciences

University of South Carolina

2019

Accepted by:

Zhenlong Li, Direct of Thesis

Michael Hodgson, Reader

Cuizhen Wang, Reader

Cheryl L. Addy, Vice Provost and Dean of the Graduate School

© Copyright by Huan Ning, 2019
All Rights Reserved.

DEDICATION

To my advisor, Dr. Zhenlong Li.

To my wife, Weiwei He.

ACKNOWLEDGMENTS

Thanks to my committee: Dr. Zhenlong Li (Chair), Dr. Michael Hodgson, and Dr. Cuizhen (Susan) Wang. They spent a large amount of time to help me establish and polish my thesis.

ABSTRACT

This thesis aims to implement a prototype system to screen flooding photos from social media. These photos, associated with their geographic locations, can provide free, timely, and reliable visual information about flood events to the decision makers. This system is designed for the application to the real social media images, including several key functions: tweets downloading, image downloading, flooding photo detection, and human verification via a WebGIS application. In this study, a training dataset of 5,000 flooding photos was built based on an iterative method; a convolutional neural network (CNN) was then trained and applied to detect flooding photos. Also, the CNN can be re-trained by a larger training dataset after adding the verified flooding photos to the training set. The flooding photo detection result shows that the trained CNN achieved a total accuracy of 93% in a balanced test set (the flooding and non-flooding class have the same number of samples) and precisions of 46% -- 63% in the imbalanced real-time tweets (the number of flooding samples are over 20 times larger than non-flooding), demonstrating the feasibility of the proposed pipeline. The system is flexible to change the classifier, so that detecting other disasters (e.g., tornado) is possible.

PREFACE

Flooding photo detection from social media is a relatively new topic in the hazard management domain. However, I did not purposely step into this field. In August 2017, when Dr. Li told me that there was a team in Florida verifying the flooding related tweets manually, I realized that the popular deep learning technology might aid this task. We quickly initiated an experimental study and obtained some promising preliminary results. Generally, image classification, where flooding photo detection belongs to, is not a difficult issue in computer vision. The main challenge is to build the training dataset – there is no publicly verified flooding dataset at that time. After manually select flooding photos from dozens of thousands of social media images, I found that the variance among those photos and images were large, which imposed a huge challenge to define flooding photos from human beings. Whether or not deep learning can handle this variance shows a great interest to me. Finally, with the encouragement and supervision from Dr. Li, I managed to build up a prototype screening system to detect flooding photos from tweets in real-time. This system can also provide an essential tool to analyze images from Twitter.com for further research.

TABLE OF CONTENTS

DEDICATION	iii
ACKNOWLEDGMENTS	iv
ABSTRACT.....	v
PREFACE.....	vi
LIST OF TABLES.....	viii
LIST OF FIGURES	ix
CHAPTER 1 INTRODUCTION	1
CHAPTER 2 RELATED WORK.....	7
CHAPTER 3 METHODOLOGY	17
CHAPTER 4 RESULTS	30
CHAPTER 5 DISCUSSION AND CONCLUSION	38
REFERENCES	43
APPENDIX A: THE MAIN STRUCTURE OF TWEET JSON.....	49
APPENDIX B: EXAMPLES FOR THE FLOODING AND NON-FLOODING PHOTO.....	52

LIST OF TABLES

Table 3.1 Samples of tweets dataset.	22
Table 3.2 Rules for identifying flooding photo.	25
Table 3.3 Rules for identifying non-flooding photo.	25
Table 4.1 Accuracies of tested CNNs	33
Table B.1 Examples of identifying flooding photo.	52
Table B.2 Examples of identifying non-flooding photo.	54

LIST OF FIGURES

Figure 2.1 A single neuron.....	8
Figure 2.2 Commonly used activation	8
Figure 2.3 A neural network with two fully connected layers.....	9
Figure 2.4 An example of convolution..	12
Figure 2.5 VGG16 Architecture	13
Figure 3.1 The human-in-the-loop workflow of the proposed system.	18
Figure 3.2 System architecture.	19
Figure 3.3 Flowchart of photo downloading.....	22
Figure 3.4 Web pages of the URLs from tweets.....	23
Figure 3.5 A simple CNN used to train the preliminary flooding dataset.	27
Figure 3.6 An example of object detection by YOLO-v3.	28
Figure 4.1 Samples of flooding photos	32
Figure 4.2 Samples of non-flooding photos.....	32
Figure 4.3 Photos identified as flooding by the neural network in Houston Flood 2017 dataset.....	34
Figure 4.4 Non-flooding photos identified by the neural network in Houston Flood 2017 dataset.....	34
Figure 4.5 Samples of flooding photos posted during Hurricane Florence Flood.....	35
Figure 4.6 Tweets with latitude and longitude in Hurricane Florence Flood 2018.	36
Figure 4.7 One located flooding photo.	36
Figure 4.8 The WebGIS application for verification..	37
Figure A.1 A Tweet in JSON format.	49

Figure A.2 Tweet JSON with Twitter Place. 50

Figure A.3 Tweet JSON with exact location 51

CHAPTER 1 INTRODUCTION

1.1 Background

Flood, mainly caused by heavy rainfall, is a huge threat to human's daily life. It dysfunctions human settlements, damages infrastructure, and causes countless losses in the local economy and residential properties. Floods are a common natural hazard in the United States (Union of Concerned Scientists 2018). Moreover, the rainfall pattern is being shifted by global warming, and flood is getting more frequent in the U.S. (Wuebbles, Fahey, and Hibbard 2017). In recent years, this country suffered from several severe floods, such as the Louisiana Flood 2016 and Houston Flood 2017. Hurricane Florence, occurred in September 2018, set at least 28 flood records on stream gages and peaks in North Carolina and South Carolina (Burton 2018). The damage cost of the flood in the U.S. is up to \$60 billion (National Weather Service 2017) in 2017.

Rapid flood situation awareness and inundation mapping are essential to hazard mitigation. Early noticing where flood occurs and how severe it is often takes time for first responders. Inundation maps serve the purposes of flooding extent and severity assessment, flood forecasting, and floodplain mapping (Koenig *et al.* 2016). The U.S. Geological Survey (USGS) normally sends out a team to collect the high water mark and measure water height in the field after a major flood event. These maps are often officially published months after the flood events (Li *et al.* 2018).

A timely approach is therefore needed for rapid flood situation awareness and mapping, and it should be cost-efficient to employ. Volunteer Geographic Information (VGI) is a potential solution for rapid flood mapping (Goodchild and Glennon 2010). In the recent decade, Social media platforms (e.g., Twitter, Instagram) are becoming increasingly popular. Also known as the “human sensors,” social media users collect and broadcast information about their physical and social environment (Sheth 2009; Nagarajan, Sheth, and Velmurugan 2011; Adam, Shafiq, and Staffin 2012), which can be considered as VGI. Recent studies have demonstrated that the real-time, free, and geo-tagged social media posts can be applied in rapid flood situation awareness and mapping (Li et al. 2018; X. Huang, Wang, and Li 2018a; Fohringer et al. 2015; C. Wang, Li, and Huang 2018; X. Huang, Wang, and Li 2018b). Most of these studies view the uploaded photos in flood relevant posts as the critical *in-situ* visual information for enhancing flood situational awareness. For instance, a photo posted by a resident shows a flooded yard is useful for assessing the water height and the working condition of the flood controls nearby. However, efficiently and accurately extracting useful flood photos from the massive amounts of unstructured social media data poses considerable challenges. For example, in November 2018 around 5,000 tweets were posted every second on average (Sayce 2018). Those tweets cover various topics, and most of them would be noises for a specific topic, such as flood.

Keyword-based and manual filtering the flood relevant social media posts are the dominant methods in pioneering research (Li et al. 2018; Fohringer et al. 2015) but with obvious limitations. First, the posts which contain flooding photos might be omitted if there

is no flood-related keyword in the text. Second, manually dealing with the massive social media posts is inefficient, leading the impossibility of real-time analysis.

Deep learning, or multi-layer artificial neural network, has gained a rapid development since 2012 (LeCun, Bengio, and Hinton 2015). It is widely used to identify objects, recognized speech, or match items (LeCun, Bengio, and Hinton 2015). For example, in the 2017 Large Scale Visual Recognition Challenge (Russakovsky et al. 2015), a state-of-the-art method, Squeeze-and-Excitation Networks (Hu, Shen, and Sun 2017), can classify 1000 categories of the image with a low error rate of 2.25%, while the best record in 2011 was 26%. This significant progress has attracted tremendous attention and research resources. Many applications have been employed in the field of visual analysis such as image classification, object detection and localization, semantic segmentation, and image captioning. As a non-manual and efficient filtering method, deep learning is a suitable approach to extracting flood relevant posts from massive social media data (Tkachenko *et al.*, 2017). For example, Feng & Sester (2018) analyzed both the text and image of a post to determine whether it is flood relevant or not. More importantly, the deep learning method can process the massive social media data in real-time, providing timely information for first responses of the local disaster management team.

However, the multifariousness of the social media-posted images challenges the accuracy of the automatic flooding photo detection. The uploaded images include screenshots of text, posters, illustrations, cartoons, advertisements, modified photos to name a few. Most topics of images only take up small portions of the entire dataset, and this is especially true for tweets with geographic information. For instance, the global geo-

tagged tweets with “flood” only take up 0.034% in the data from Stream API operated by Twitter.com.

Meanwhile, the tweeted photos were captured in various devices, angles, and environments, serving a wide variety of purposes. The arbitrariness of photo exacerbates the uncertainty of the detection results. Thus, a fully automatic flooding photo detection method is difficult to be implemented but urgently needed, and a manual final verification stage is helpful for the labeled flooding photo to be used in the flood mapping models. In addition, the location information is critical for hazard situation awareness and responding, so that verifying the location of those flooding photos is required. Currently, there is no feasible method to conduct location verification except manual work. Therefore, a practice approach is to build a system that can automatically filter out irrelevant photos and provide a relatively small amount of flooding photos for manual verification.

Crowdsourcing is a preferred choice for such a verification process. Volunteers or other human operators can verify whether a social media photo shows flooding evidence or not. Crowdsourcing is being increasingly used for problem-solving and task realization (Estellés-Arolas, Navarro-Giner, and González-Ladrón-de-Guevara 2015), where a large number of participants can accomplish a complex task collaboratively at a low cost. Wikipedia (Cox 2011) is a successful example based on massive volunteers, and Amazon Mechanical Turk (Chen et al. 2011) is a common tool to gather a large amount of workforce together to accomplish heavy tasks, such as image classification. Gathering information for disaster response via crowdsourcing is an effective approach to supporting disaster response (Chan 2014; Goodchild and Glennon 2010; Zook et al. 2010; Gao et al. 2011). In this research, crowdsourcing is applied as a refining method for automatic classification

and a necessary stage for location determining. The verified flooding photos provide reliable flooding information for better hazard management.

1.2 Research objective

The goal of this research is to design and develop a prototype system to detect flooding photos from streamed social media (tweets in this research) across the contiguous U.S. in real-time (< 2 minutes). This research has the following objectives:

- 1) Building the training dataset to train a deep learning-based classifier to identify the flooding photos.
- 2) Automatically extracting the flooding photos in real-time steamed geo-tagged tweets in the flood events.

Since the text analyses to social media have been studied extensively for years, this research focuses on images only, which is not well investigated per my best knowledge of current literature. The term “image” referred in this research means the image posted in social media, including photos, screenshot, and other raster files. “Photo” is the image obtained from cameras. The photo records the on-site visual information, while the image may have no relationship with the on-site environment.

1.3 Significance

The proposed system can extract timely flooding photos from social media to support flood situation awareness and inundation mapping. Based on the visual evidence of these photos, decision-makers can more accurately evaluate the situation. The water height can also be estimated from the flooding photos to obtain a timely inundating map without going out to the field, which has been challenging in an ongoing flood event (Li et

al. 2018). Also, the flooded time can be extracted according to the metadata (posted time or the text), hence generating a dynamic inundation map becomes possible. The traditional field survey of high water marks lacks the temporal dimension because the survey is conducted after the flood event, whereas the flooding photos extracted from social media provide high temporal relevance. Since training and classifying are relatively independent processes, this system can be viewed as a social media image analyzing platform that can be applied to extract hazard-related photos in real-time for other disaster events, such as tornados, wildfires, and earthquakes.

CHAPTER 2 RELATED WORK

This section introduces the basic concepts of deep learning, its application in image classification, and the previous research of social media flooding photo classification.

2.1 Neural network

Deep learning is an implement of the artificial neural network, which consists of many simple, connected units called neurons (Zhang, Li, and Mo 2018). Based on the input values from other neurons, a neuron will output a value serving as the input of other neurons, including the previous neurons and the new neurons. The neural network is a branch of machine learning, being developed decades since the 1960s. The key task of a neural network is figuring out the weights used to calculate with the input values and output a new value.

Figure 2.1 shows the operating principle of a single neuron. The weight w_i and the bias b are learnable, they will be changed in the training process. Given the input vector \mathbf{x} , the neuron computes a weighted sum and adds a bias b , then applies an activation function f to the result. The output of f is also the final output of the neuron.

Activation function gives the non-linearity to the neuron. Figure 2.2 gives the commonly used activation functions, and their graphs show that the input of the function is squeezed into a range non-linearly. The Sigmoid function, historically widely-used, relocates the input to $(0, 1)$. A popular activation function, ReLU, simply outputs the input

value if it is larger than 0, significantly reducing the computation then accelerates the training process.

Activation function gives the non-linearity to the neuron. Figure 2.2 gives the commonly used activation functions, and their graphs show that the input of the function is squeezed into a range non-linearly. The Sigmoid function, historically widely-used, relocates the input to (0, 1). A popular activation function, ReLU, simply outputs the input value if it is larger than 0, significantly reducing the computation then accelerates the training process.

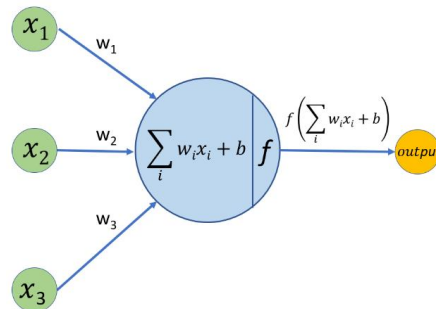


Figure 2.1 A single neuron. x_i : input; w_i : weights, b : bias; f : activation function.

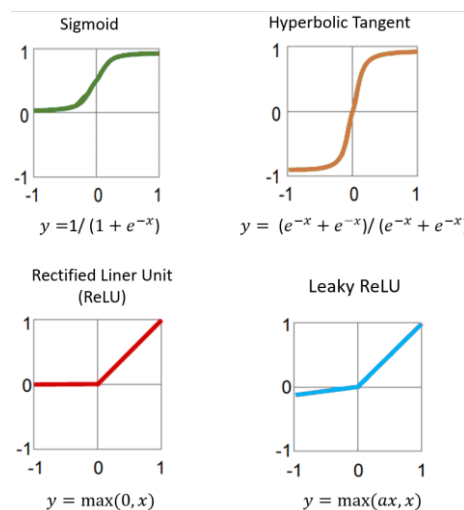
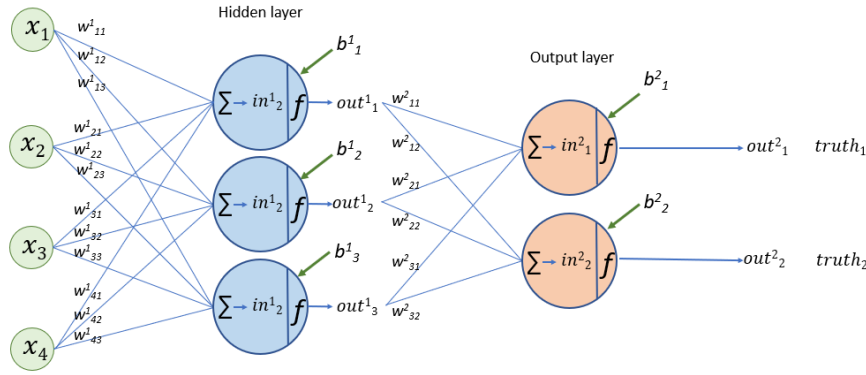


Figure 2.2 Commonly used activation

The basic neurons can form a layer, and these layers can compose the neural network. The outputs of a layer can be the inputs of other layers, or the current layer (Recurrent Neural Networks, RNN). Figure 2.3 is a sample of the neuron network with two fully connected layers: a hidden layer and an output layer, containing 5 neurons. In this network, there are 23 learnable parameters: 12 weights for the first layer, 6 weights for the second layer, and 5 biases for each neuron. The input of the activation function f is notated as in^l , and the output is out^l . The superscript l is the layer number, in this case, $l = 1$ is for the hidden layer, and $l = 2$ for the output layer.



$$in^1 = \sum_{i=1, j=1}^{i=4, j=3} w^1_{ij} x_i \quad (1)$$

$$out^1 = f(in^1 + b^1) \quad (2)$$

$$in^2 = \sum_{j=1, j=k}^{j=3, k=2} w^2_{kj} out^1_j \quad (3)$$

$$out^2 = f(in^2 + b^2) \quad (4)$$

$$E = \frac{1}{2} \sum_{k=1}^{k=2} (out^2_k - truth^k)^2 \quad (5)$$

$$\frac{\partial out^2}{\partial in^2} = f'(in^2) \quad (6)$$

$$\frac{\partial E}{\partial out^2} = \sum_{k=1}^{k=2} (out^2_k - truth^k) \quad (8)$$

$$\frac{\partial in^2}{\partial w^2} = out^1 \quad (7)$$

$$\frac{\partial E}{\partial w^2} = \frac{\partial E}{\partial out^2} \cdot \frac{\partial out^2}{\partial in^2} \cdot \frac{\partial in^2}{\partial w^2} \quad (9)$$

$$\frac{\partial E}{\partial w^2} = \sum_{k=1}^k (out^2_k - truth^k) \cdot f'(in^2) \cdot out^1 \quad (10)$$

Figure 2.3 A neural network with two fully connected layers

The usage of this neural network is that after going through the layers, the input will be transferred into the output layer whose value indicating the specific meaning, for instance, class label. This step is called Forward Propagation (FP), with the known

parameters. The inference stage is an FP process. The inversed process, Back Propagation (BP), is used to calculate the parameters by comparing the output of FP and the ground truth. BP is the main process in training of a neural network.

In the FP stage, the input layer, as a four-dimension vector \mathbf{x} in Figure 2.3, goes through the hidden layer and the output layer, and then is converted to a two-dimension vector \mathbf{out}^2 . Equation (1) - (4) are the computing path to obtain \mathbf{out}^2 . f can be one of the activation functions listed in the Figure 2.2. \mathbf{w}^1 and \mathbf{w}^2 are the weights, \mathbf{b}^1 and \mathbf{b}^2 are the biases.

In the training stage, \mathbf{w}^1 , \mathbf{w}^2 , \mathbf{b}^1 and \mathbf{b}^2 will be determined by the BP process. The training dataset includes input samples and corresponding desired output, annotated by \mathbf{x} and \mathbf{truth} respectively, where $\mathbf{x} \in \mathbb{R}^4$, $\mathbf{truth} \in \mathbb{R}^2$. The objective of the training is obtaining a group of \mathbf{w}^1 , \mathbf{w}^2 , \mathbf{b}^1 and \mathbf{b}^2 to ensure that \mathbf{out}^2 approximates \mathbf{truth} as much as possible. The metric of the approximation is loss function, or cost function, error function. Loss function can be several forms based on the design of the neural network. In this two-layer neural network, the squared error, Equation (5), is used to assess the performance.

The neural networks are usually trained by the iterative gradient descent method. After initializing the parameters (weights and bias) with random values and calculating the loss, the BP algorithm adjusts the parameters by a step, named learning rate, along with the negative gradient, guaranteeing the loss decreases fastest. Specifically, gradient descent proposes new weights $\mathbf{w}' = \mathbf{w} - \epsilon \nabla Loss(\mathbf{w})$, where ϵ is the learning rate. In practice, ϵ is set to small constant, such as 0.01 – 0.001.

Take the updating of w^2 as an example. w^2 first affects in^2 , then out^2 , finally affects the loss E_k , based on the chain rule of derivative, then we have Equation (9). According to the derivative rules, we can get (6) from (4), (7) from (3), and (8) from (5). Put (6), (7), and (8) into (9), we get (10), hence $\nabla Loss(w^2) = \frac{\partial E}{\partial w^2} = \sum_{k=1}^k (out_k^2 - truth^k) \cdot f'(in^2) \cdot out^1$. Using the similar method, we can update w^1 , b^1 and b^2 in the BP process, which will conduct many times until the loss satisfies a threshold or the number of iterations exceed the assigned maximum value.

2.2 Convolutional neural networks

The Convolutional Neural Network (CNN) is used for the structured grid input, usually an image. A CNN layer looks like a serial of stacked image filters with the same height (H) and width (W). With the number of these filters, D , a CNN layer can be notated as a $H \times W \times D$ tensor (three-dimensional array). The weights of a CNN layer are the elements of the filters. These filters slide over the input data, conduct convolutional computation, whose results will go through an activation functions, then record the final output in a tensor. The size of the filters (f), stride (s) and padding (p) is given, not learnable parameters. The output of a CNN layer is a feature map, storing the responses of each scanned region to the filters. Figure 2.4 illustrates a 3×3 convolution scans a 4×4 image and returning a 2×2 image.

Pooling operation in a CNN performs down-sampling to reduce the number of parameters, computation and prevent over-fitting. For instance, max pooling, an often-used pooling layer, returns the max value in a $h \times w$ region. If use a 2×2 max pooling layer, the output image will be $\frac{1}{2}H \times \frac{1}{2}W$ size as before.

For the image classification, fully connected layers are often used at the end of a CNN. The neurons from the previous layer connect to all neurons of the fully connected layer. The output of a fully connected layer is a $1 \times 1 \times n$ tensor, where n denotes the number of the neurons. This tensor cannot keep the spatial information, and is responsible for the high-level reasoning.

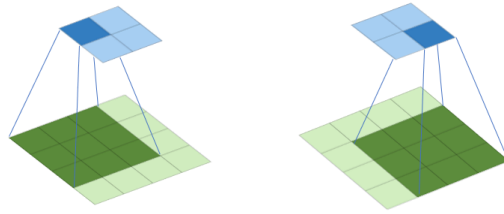


Figure 2.4 An example of convolution.
Filter size $f = 3$, stride $s = 1$ and
padding $p = 0$.

Figure 2.5 presents a successful CNN architecture of image classification -- VGG16 (Simonyan and Zisserman 2014). Its size of the input image is $224 \times 224 \times 3$. After going through 6 convolution layers ($f = 3, s = 1, p = 1$ in all convolutional layers) and 5 max pulling (2×2) layers, the input image was transferred into a $7 \times 7 \times 512$ feature map, then a $1 \times 1 \times 1000$ tensor by the two fully connect layer, 1000 is the number of the image classes.

The softmax layer in VGG16 converts the unnormalized log-probabilities to probabilities P_i of each class. P_i is a value between 0 and 1, and the sum of all P_i is 1, see (11), where C equals the number of classes, Z_i is the log-probability of class i . When training a classification CNN, the cross-entropy function L_{ce} is commonly used to evaluate the loss of the network, see (12), where $y_i = 1$ if the neuron i in the output layer belongs to the class, otherwise $y_i = 0$. For instance, if a training dataset includes 3 class, at a training

step, one sample goes through the neural network and gets an output as [0.1, 0.2, 0.7], and the **truth** is [0, 0, 2]. The $L_{ce} = -(0 \times \log(0.1) + 0 \times \log(0.2) + 1 \times \log(0.7)) = 0.155$.

$$P_i = \frac{e^{z_i}}{\sum_{j=1}^C e^{z_j}} \quad (11)$$

$$L_{ce} = -\sum_{i=1}^C y_i \log(P_i) \quad (12)$$

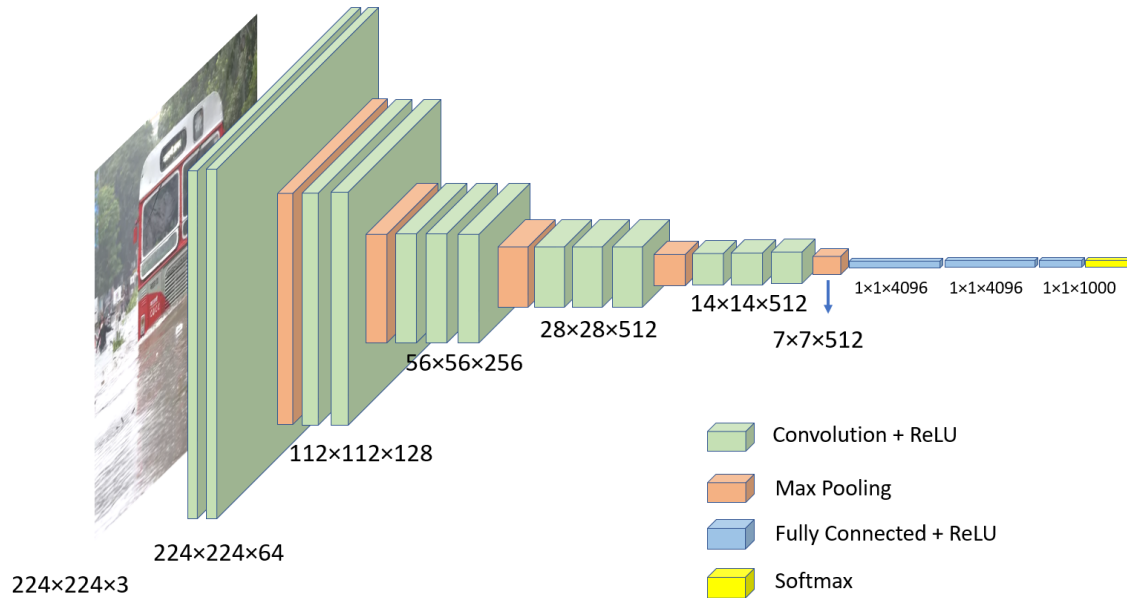


Figure 2.5 VGG16 Architecture

2.3 Image classification based on deep learning

In computer vision, image classification aims at labeling an image to a class according to its content. For example, given a photo of cat, the algorithm will return probabilities to a set of labels, such as cat, dog, or tiger. A qualified algorithm should assign a much higher probability to the cat label than other labels. Before the booming of deep learning, bag-of-words (BoW) is the most popular and successful approach (Druzhkov and Kustikova 2016). The features of the image are extracted by descriptors, such as SIFT (Scale Invariant Feature Transform, Lowe 1999) and SURF (Speeded Up Robust Features,

Bay, Tuytelaars, and Van Gool 2006), then form the vocabulary. The BoW methods treat the features like words and will cluster the images based on their features in the vocabulary. SVM and hierarchical models are popular methods. BoW is hard to keep the spatial context and extract features for various objects in the images.

Deep neural network approaches have made great progress in the past years. AlexNet won the image classification task of ILSVRC 2012 with the accuracy significantly ahead of the second place (16% v.s. 26.2% in error rate). In this challenge, the competitors needed to classify 150,000 testing images into 1,000 classes by training their classifiers with 1.2 million images. In recent years, the error rate has been decreased by more and more complex CNNs, such as VGG (Simonyan and Zisserman 2014) and ResNet (He et al. 2015). Most popular open-sourced deep learning frameworks (e.g., Tensorflow, Pytorch) provide these trained CNNs, and the user can easily apply them to classify images or train the CNNs with the customized training dataset. ILSVRC 2017 is the last image classification challenge in which the error rate was decreased to 2.251%, much better the human performance (5%, Russakovsky et al. 2015). Therefore, the organizer thinks the image classification question was addressed and closed this task.

The developer also can design CNNs for specific tasks. Gebru *et al.* 2017 detected and classified the cars in Google Street View, and acquired a community income prediction with a high correlation to the ground truth ($r = 0.82$). The CNN based on AlexNet recognized 50 million images of 200 largest American cities and labeled the cars into 2600 categories. The authors used the detected cars to conduct a sociological study with the demographics. The fine-grained car detector, trained by 347,811 samples, provides the basic data in this search. Sladojevic *et al.* 2016 trained a CNN to recognize the plant

diseases by leaf image classification. They used 4,483 images to train the CaffeNet, a 1-GPU version of AlexNet embedded in Caffe deep learning framework, to classify 13 leaf diseases of several plants, including apple and peach. The CNN obtained an average accuracy of 96.3%. The image classification based on CNN also is applied or researched in many fields, such as medical image analysis (Tajbakhsh et al. 2016; D. Wang et al. 2016; Bar et al. 2015), and animal detection (Yoon and Yoon 2018).

2.4 Flooding photo classification

Flooding photo classification is one application of image classification and has become a new research topic in hazard management. The Multimedia Satellite Task at MediaEval (Bischke et al. 2017), a competition of disaster photo detection and satellite image classification, works on the promotion of multimedia access and retrieval algorithms. In 2017 and 2018, this task focused on the flooding event. The testers combined the text and photos from social media to determine whether a tweet is flooding related. The top methods in 2017 can get an accuracy higher than 95% (Bischke *et al.* 2017). The training data comes from YFCC100M (Thomee et al. 2016), but did not have a specific criterion about flooding photo. The researchers used the statement such as “unexpected high-water levels in the industrial, residential, commercial and agricultural area” as the definition of flooding photo. The human annotators rated the image to a score range of 1 - 5 according to the strength of the flooding evidence. This contest did not emphasize the application and employment and did not pay attention to data acquisition.

Feng & Sester (2018) use CNN and other methods to classify pluvial flood relevant tweets. Both text and photos in the tweets were combined and classified as relevant and irrelevant. The authors used three subsets (7600 photos each) to train the model. Subset 1

contains common images in social media. These images are flood irrelevant, selected by human annotators. Subset 2 is flooding photos, and the last one consists of water surface images. Subset 3 includes the water surface images, which is used to train a classifier to distinguish the flood and water surface, such as lakes. Two classifiers were trained. For Subset 1 and 2, Xgboost got the highest score of 0.9288. Other methods achieve 0.8876 – 0.9249. When trained by Subset 2 and 3, Xgboost still got the highest F1-score (Goutte and Gaussier 2005) of 0.8774, and others range from 0.8474 to 0.8731. A photo was identified as flood relevant if both classifiers considered it as flooded. This approach was tested by flooding events in Paris, London, and Berlin. When collecting the flooding and water surface images from the internet, this study used search engines and search tools from Twitter.com and Instagram.com. Therefore, these two subsets were not all real data from social media, so they might not be representative.

Although these studies have been conducted on social media flooding photo detection, their objectives of application and standards of data collection are not clear, even using simulated datasets. This research proposes that the main usage of flooding photos should be to provide *in-situ* information directly from the flooding scenes, and the training dataset is built based on this principle. The secondhand information, such as retweets and screenshots from the public media, is not considered in this study. Another reason for this ignoring is the difficulty to map the secondhand information according to the location of the tweet.

CHAPTER 3 METHODOLOGY

This research is to implement a system that can screen the geo-tagged flooding photos from the massive social media posts for rapid flooding situation awareness and better hazard response. The first task is to collect flooding photo samples to build the training dataset for flooding/non-flooding photo classification. Based on a small group of flooding photos, an iterative method is applied to train a pre-selected CNN classifier and use it to collect more flooding photos from social media images. In order to screen the social media photos in real-time, several independent modules are developed for the following sub-tasks: tweets downloading, image downloading, flooding photo detection, and crowdsourcing verification. A MySQL database is used to store and exchange the data from these modules. The system is designed as a general social media image analysis platform that can perform various scene detection and object detection tasks.

Since the location information is critical for flooding response, the system merely downloads the geo-tagged tweets, and the crowdsourcing verification module is designed as a WebGIS application which can also be used to verify the location of flooding photos.

Figure 3.1 demonstrates the workflow of the proposed system. Leveraging parallel computing techniques, a multi-process program is developed to download the massive photos in parallel. These photos are then used to build a training dataset from

scratch iteratively. This study trained several CNN architectures and used the one with the best accuracy to detect the flooding photos. The flooding photos labeled by the CNN can be verified through a WebGIS application, and the CNN will be re-trained after adding the verified photos to the training dataset. The tweets images from two flooding events, Houston Flood 2017 and Hurricane Florence Flood 2018, were used to test the accuracy of the trained CNN.

In the study, the primary development language is Python, and the prototype system runs in the Windows 10 operating system. The deep learning framework used in this research is Pytorch (Paszke et al. 2017). The development environment includes Ubuntu 16.04, Pytorch and Jupyter notebook, and the training and test framework was implemented by Python. The hardware consists of two Nvidia Titan Xp graph cards (12 GB memory each), one Intel i7 CPU, 1.5T SSD, and 64GB memory.

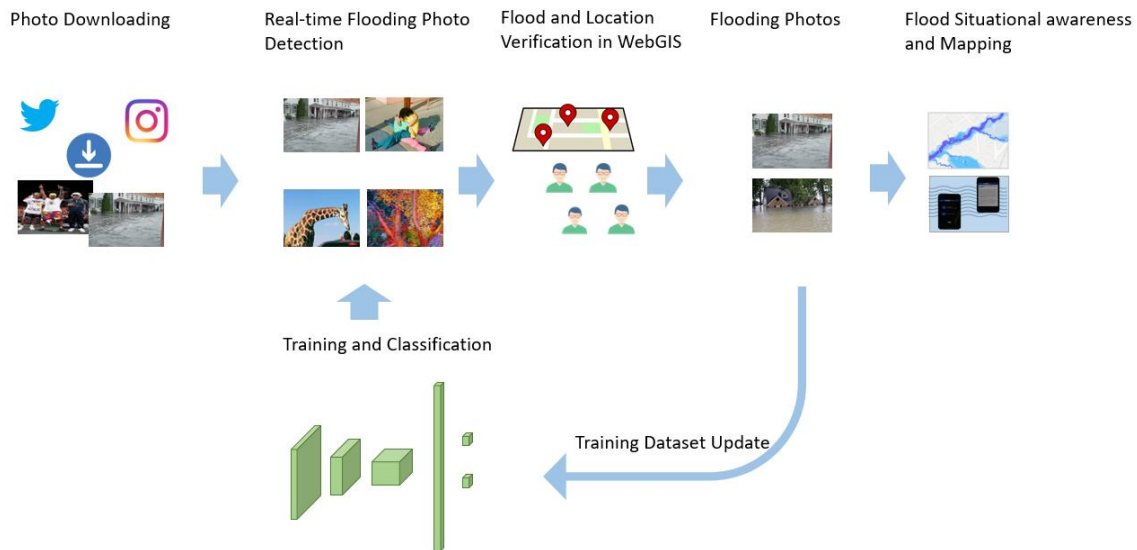


Figure 3.1 The human-in-the-loop workflow of the proposed system.

3.1 System architecture

The screening system is heavily based on massive tweets and their associated images. Moreover, the modules in the workflow need intensive data exchange. Thus, a MySQL database is used to store and retrieve data. The system architecture in Figure 3.2 shows the database-centric workflow: 1) The tweets are downloaded into the database. 2) The photo downloading module obtain the images in the tweets. 3) The images are classified by a trained classifier, and 4) the flooding photos are verified by volunteers via a website and used for flooding mapping. The verified flooding photos serve to flood awareness, mapping, and enlarge the training set.

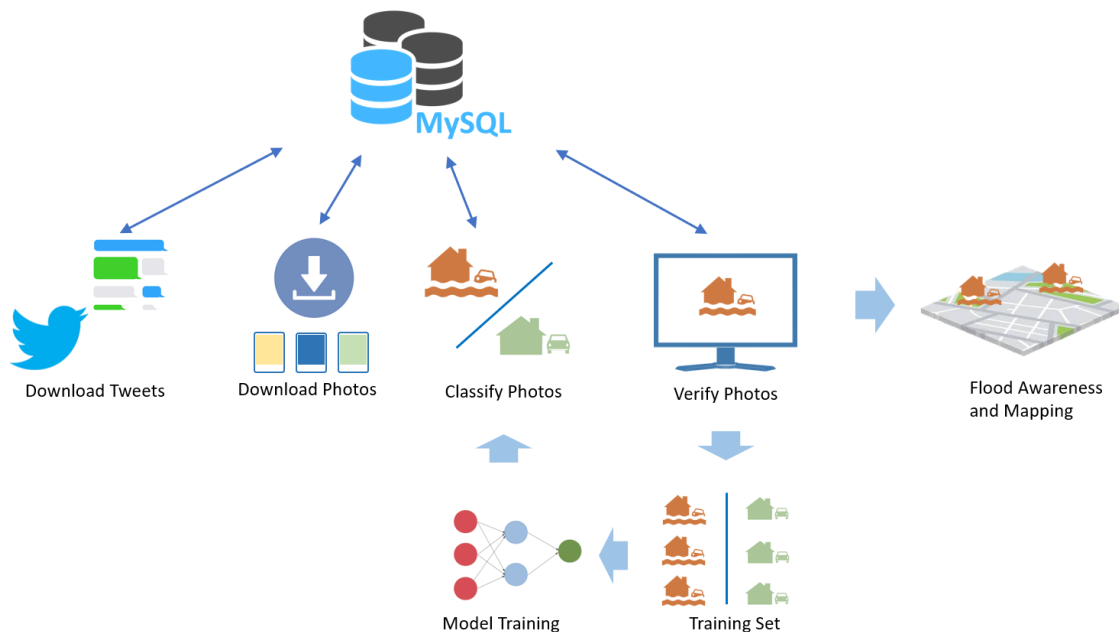


Figure 3.2 System architecture.

Flooding photos classifier is the core module of the system. It is embedded in the system but is trained in another workflow. Also, the classifier can be retrained after the verified flooding photos are added to the training set. A large training set benefits the performance of the classifier. To verify the extendibility of the system, a YOLO-v3

(Redmon and Farhadi 2018) module, which implemented in Tensorflow (Abadi et al. 2016), is attached as an object detector. The social media images will go through two CNNs, and the detecting results are recorded in the database.

3.2 Geotagged tweets downloading

The tweets downloading module uses the Twitter Streaming API to obtain geo-tagged tweets. The downloaded tweets are stored in a table in a MySQL database. This research uses Tweepy, a Python package, to obtain the streaming tweets. Tweepy is an easy-to-use library to access Twitter API, including Streaming API. Twitter Streaming API pushes tweets in JSON (JavaScript Object Notation) format, which is a lightweight data-interchange format and is easy for human reading. Appendix A shows the main structure of a tweet and the Geo object structure in a JSON file. Correctly parsing the tweet JSON is the foundation to store and retrieve the tweets. In the tweets downloading module, every 200-1000 tweets are parsed into the CSV (Comma-Separated Values) format and then are stored in a MySQL table.

3.3 Images downloading

Two scenarios need to be considered in the image downloading module: using the real-time tweets and the tweets in the repository to download the images. The tweets in the repository for this research were collected in 2016-2017 across the contiguous U.S. through the Streaming API. Downloading images from real-time tweets is relatively simple because the posted images have corresponding URL (Uniform Resource Locator) in the tweets JSON. The program can obtain the image directly from the URL.

Using tweets from the repository needs more steps. These tweets were obtained from the Streaming API of Twitter.com and contain the tweet ID, user ID, short URL, and

other fields. Those URLs link to another webpage which the Twitter users want to share or the tweet itself (those have more than 140 characters). About 30% of URLs link to the posts of social media websites (Twitter.com and Instagram.com). The image downloading program extracts the short URLs from tweets and executes them in a browser to get the HTML (Hyper Text Markup Language) files. A few real-time tweets (about 1%) have an URL from Instagram.com, and were also processed in this scenario.

Figure 3.3 shows the flowchart of images downloading. After retrieving tweets from the database, the program extracts a short URL (if any) from a tweet and open it in a browser. Because Twitter.com shortens the full URL into the short (tiny) URL, such as “https://t.co/Qi8Xs5jopp”, the browser needs to open the short URL and get the full URL (e.g., <https://www.instagram.com/p/8WY30zr7F6GkXdywqP7pJJfuLPrMncIjG2yc0/>). To collect the user-generated content (first-hand information), the system only downloads the images from Twitter and Instagram. If the full URL comes from a social media website (Twitter and Instagram in this research), the program will get the HTML page and download all the images embedded in the HTML file. Usually, a single HTML page will contain many images, such as the website logo and the user headshot. The posted photo is just one or several among the embedded images. The downloading program uses a simple strategy to get the posted photos: only save the image with the largest dimension. The saved image will be named with the tweet ID. Images in each day are stored in a single folder.

Table 3.1 lists 4 tweets and their short URLs, two web pages of the URLs in the first two tweets are shown in Figure 3.4. The Streaming API sends about 10 geo-tagged tweets per second (daytime) in the study area, and about 10% among them have accurate

longitude and latitude coordinate. These statistics were recorded in Feb. 2019, and they keep changing.

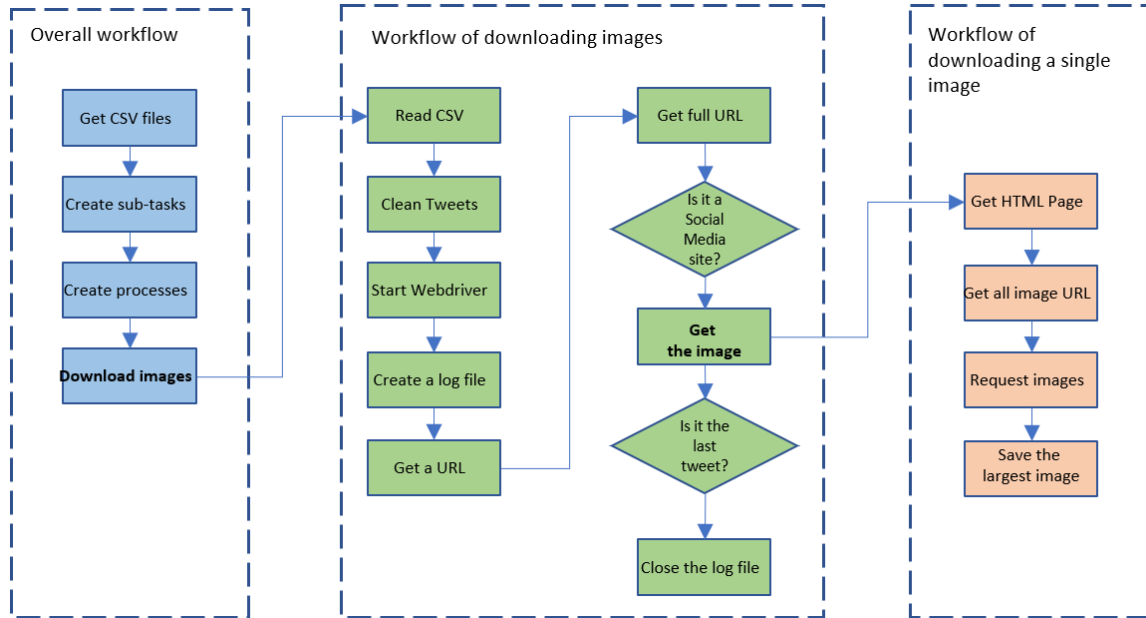


Figure 3.3 Flowchart of photo downloading

Table 3.1 Samples of tweets dataset. The URLs in the posts will be opened to get the HTML files.

Time	Text	URLs
2015/10/2 17:11	if you didn't know, but I are under a flash flood warning??AND today was???????	https://t.co/Qi8Xs5jopp
2015/10/2 17:15	You could not ask for a better cuddle buddy... @ The Gentry???????	https://t.co/QBT04Dlk6p
2015/10/2 17:16	Drinking in the rain. (@ Pearlz Oyster Bar in Columbia; SC)	https://t.co/ZqNykREp30
2015/10/2 17:23	This is what a rainy afterschool Friday afternoon looks like around here. Ahhhh....???????	https://t.co/G9nGFGeJb7
2015/10/2 17:33	At 4:30 PM; Myrtle Beach [Horry Co; SC] DEPT OF HIGHWAYS reports FLOOD	http://t.co/Sr8UHDxWnf

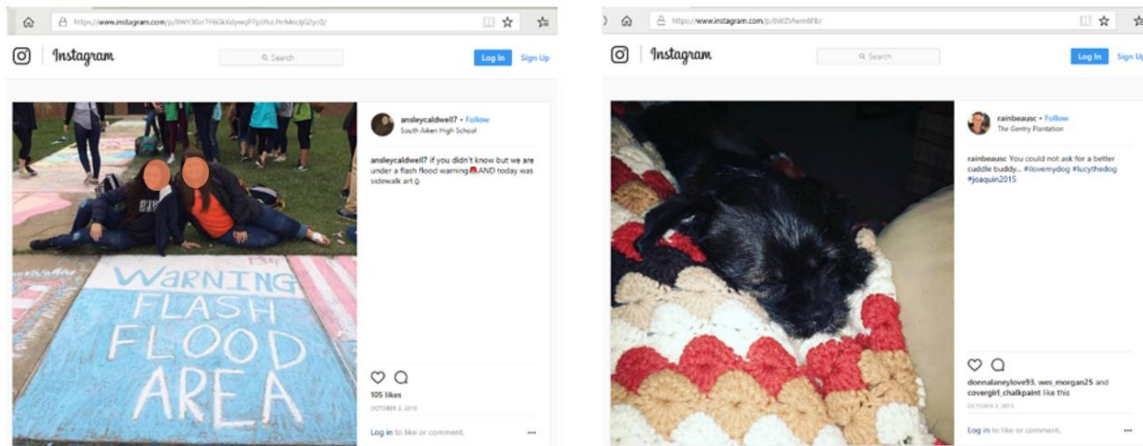


Figure 3.4 Web pages of the URLs from tweets.

The downloading speed is determined by the internet access speed and the performance of the downloading computer. The browser needs about 1 – 3 seconds to get the HTML files, so the downloading program uses the multiprocessing feature of Python to speed up the process. According to the number of CPU cores and the bandwidth of internet access, the program starts 4 – 30 processes to conduct downloading.

3.4 Training and classification

Training and classification is the key module of the system. This research trained several popular CNN architectures, such as AlexNet (Krizhevsky, Sutskever, and Hinton 2012), VGG (Simonyan and Zisserman 2014), and ResNet (He et al. 2015), then use the one with the highest accuracy (the percentage of the true positives in the test set). A neural network could classify two or several types of images. It could be trained from scratch or use transfer learning (Shin et al. 2016), which keep the trained weight in the front-end of a trained CNN. The metrics to evaluate the classification result contain accuracy and recall (the percentage of the true positives in the real flooding photos). When the human verified flooding photos are ready, the CNN can be trained again by a larger training dataset. This

study used accuracy, precision, and recall to evaluate the performance of the CNNs. These metrics are calculated with Equation (13), (14), and (15).

$$accuracy = \frac{true\ positive + true\ negative}{true\ positive + fals\ positive + false\ negative + true\ negative} \quad (13)$$

$$precision = \frac{true\ positive}{true\ positive + fals\ positive} \quad (14)$$

$$recall = \frac{true\ positive}{true\ positive + false\ negative} \quad (15)$$

A training dataset needs to be built to train the CNNs. Since there is no public social media flooding photo dataset, this study needs to collect flooding photos from social media. The flooding photo only takes up a small portion in the whole tweet repository, for instance, the tweets with “flood” keyword only consist of 0.034% of the entire English repository. Manually labeling the flooding photos in the whole repository is unfeasible. This research uses an iterative method to collect flooding photos from the whole repository. Firstly, a list of about 800 tweets was collected, which were manually verified in a flooding event in 2017. A team checked 11,000 geo-tagged tweets and labeled about 800 of them as flood relevant. 430 flooding images were downloaded among these 800 tweets. However, a training dataset of 430 positive samples is not big enough to train a CNN. About 1500 flooding images from the image search engines of Google.com and Bing.com were added to the training dataset. Another 1500 non-flooding images were selected randomly from ImageNet as negative samples. Thought these flooding and non-flooding photos come from different sources, they can form a preliminary training dataset. A sample CNN was trained by this dataset and was used to classify the images from social media. Once the trained CNN is able to classify most flooding photos, it would be applied to detect more flooding photos from real tweets image to obtain more sample for the training set.

3.4.1 The criterion of flooding photo

This research defines a set of criterion to determine whether or not a photo contains on-site information about the ongoing flood. Several rules were established to identify a flooding photo (Table 3.2) or a non-flooding photo (Table 3.3). Appendix B lists some sample photos for each rule. Based on these rules.

Table 3.2 Rules for identifying the flooding photos. If an *in-situ* photo reflects an ongoing flood and provides the firsthand visual information, it can be identified as flooding photo.

No.1	Photos with clear references inundated by water outdoors.
No.2	The indoors photos with clear references inundated by water.
No.3	A mosaic image contains current flooding photos.
No.4	The photos in No.1 – No. 3 with a minor note from the uploader.

Table 3.3 Rules for identifying non-flooding photos. A photo which cannot provide distinguishable visual information about the ongoing flood is a non-flooding photo.

No.1	Description	Flooding photos from other mass media or social network users.
	Reason	Cannot be considered as firsthand information.
No.2	Description	Photos with thin water.
	Reason	The situation is still under control, not a flood.
No.3	Description	The high-water level in a river but inundating nothing.
	Reason	The situation is still under control, not a flood.
No.4	Description	Advertisements or illustrations with flooding photo as background.
	Reason	Cannot indicate an ongoing flood.
No.5	Description	No water in the photo.
	Reason	Cannot indicate an ongoing flood.

No.6	Description	Modified flooding photo.
	Reason	Cannot provide reliable information about the current flood.
No.7	Description	Fake flooding photo.
	Reason	Cannot provide reliable information about the current flood.
No.8	Description	Historical flooding photos.
	Reason	Cannot provide reliable information about the current flood.
No.9	Description	Only water bodies without reference.
	Reason	Cannot judge whether there is a flood.

3.4.2 The iterative method of building the flooding photo training dataset

Building the flooding training dataset is an iterative process. In the beginning, the CNN got a low accuracy due to the imperfection of the preliminary training dataset. Many images are mislabeled. However, the ratio of the flooding photo in the classification results with a “flooding” label is higher than the original distribution. The human annotators can efficiently pick up the real flooding photos in the “flooding” results. The verified flooding photos are moved to the training dataset, and the CNN will be trained again, then classify the remaining twitter images again. In every epoch, the human annotator will move the real flooding photos to the training dataset. After several iterations, most flooding photos are moved to the training dataset, and the human annotator has gradually made up specific rules to identify a flooding photo. Finally, the training dataset will be inspected again based on the rules. In the training stage, the CNN was trained by a balanced dataset, meaning the number of non-flooding photo equals the flooding photo. The training dataset consists of the training set and test set. 75% of the flooding photos are used as the training set, and the remaining 25% is the test set.

3.4.3 The CNN architectures

This research tested several popular CNN architectures to determine which one is most suitable for social media flooding detection according to their performances on the test set. After building the training dataset using a simple, 3-layer CNN, several popular architectures are trained, including VGG16 (Simonyan and Zisserman 2014), ResNet152 (He et al. 2015), DenseNet (G. Huang et al. 2016), and Inception V3 (Szegedy et al. 2016). Their details can be found in the references. All the output layers of these CNNs are modified to two neurons. When training the neural networks, the loss and total accuracy of the test set were printed out every 20 epochs to see whether the network converges or not. If the performance is stable and acceptable, the training will be stopped.

Due to the minor samples of the preliminary training dataset, a simple CNN with a small amount parameter is utilized at first. Figure 3.5 shows its architecture: the input size is 224×224 , followed by three successive convolutional layers. The final feature map has a shape of $28 \times 28 \times 128$, connected by a 1024-dimension fully connected layer. The output layer contains two neurons, indicating the flooding and non-flooding photos categories.

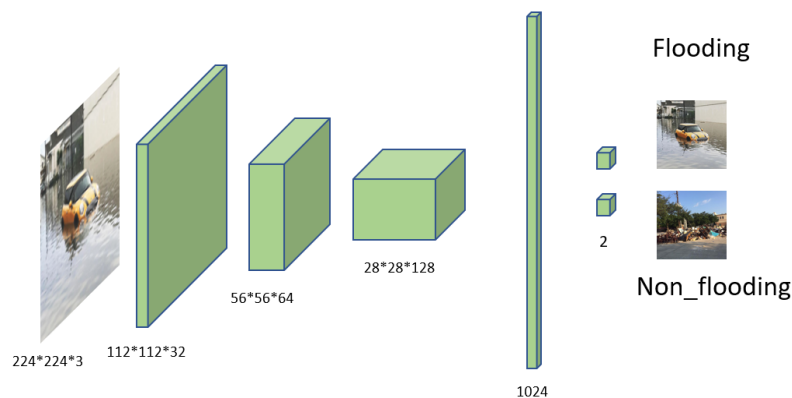


Figure 3.5 A simple CNN used to train the preliminary flooding dataset. The CNN contains 3 convolutional layers, 1 fully connected layer. All the kernel sizes=3, strides= 1, and paddings= 1.

A pretrained YOLO-v3 (Redmon and Farhadi 2018) will be attached to the system, and the 80 classes of common objects in the images can be detected and recorded in the database. Figure 3.6 gives a sample of object detection results by YOLO-v3.

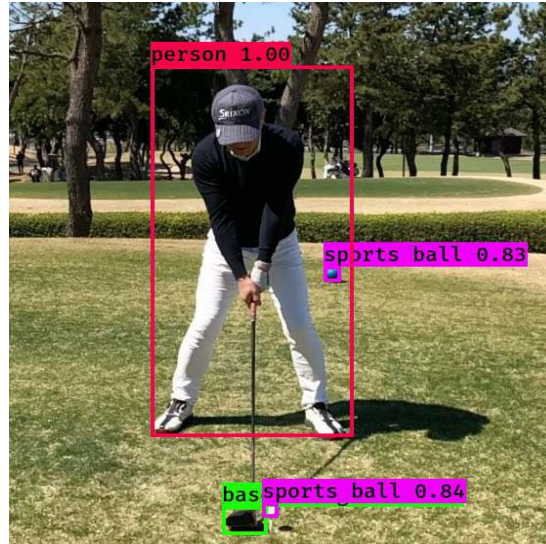


Figure 3.6 An example of object detection by YOLO-v3.

3.5 Crowdsourcing verification module

The deep learning algorithms will mistakenly label a small amount of non-flooding photo as flooding. Because the non-flooding photo takes up the most tweets even in a flooding event (more than 95%), the non-flooding photos will take up a noticeable portion in the labeled flooding photos. Additionally, flooding photos from social media have wide variance. Human knowledge and experience are still needed in a reliable classification. Therefore, the proposed system connects a WebGIS application for human operators to screen the auto-classified result and associate the photos with geographic location. The interface is based on Google Map, displaying the image and showing the tweet to the user. The volunteers can verify the results via the internet access and label the true flooding photos.

When the WebGIS displays a photo on the map, its location of the photo will be determined by the latitude and longitude of the tweet. If the tweet only has a place location, e.g., a geographic bounding box, the application displays the bounding box and locate the photo in the center of the box, so that the user is informed with the region where the photo was taken.

Only the flooding photos will be verified because the non-flooding photos are too many to be checked at a reasonable cost. The WebGIS application can display the flooding photos and the corresponding probabilities. The probabilities are derived from the classifier. Once those photos are verified, they will be added to the training set to further train the classifier.

CHAPTER 4 RESULTS

The prototype system was successfully implemented. It can download the tweets and their embedded images while classifying the images in real-time (< 2 minutes). The classification result can be manually verified via a WebGIS application. When building the training dataset, overall 5,000 flooding photos were collected from Twitter.com and Instagram.com to form a training dataset. Among the tested CNN architectures, the VGG16 obtained the highest accuracy: about 93% in a balanced test set which contains 1,000 flooding photos and 1,000 non-flooding photos. The images from two flood events, Houston Flood 2017 and Hurricane Florence Flood 2018, were classified, and the precisions were 63% and 46%, respectively.

4.1 Geotagged tweets downloading

The tweets downloading module can obtain tweets at a maximum speed of 50 tweets/second (up to the upper limit of Twitter Streaming API). The targeted tweets can be geo-tagged or not, or with a set of keywords. Note that the Streaming API does not support geo-filter and keyword-filter simultaneously, so when applying these two filtering methods at the same time, this module will gain the geo-tagged tweets which contain the assigned keywords and then download the images in the tweets. When setting the search bounding box to the contiguous U.S., This module can capture about 10 tweets/second at daytime. All the downloaded tweets are stored in a MySQL database and labeled as unprocessed.

4.2 Image downloading

The image downloading module retrieves the tweets from the database and downloads the images according to the URLs in the tweets. The number of downloading processes ranges from 1 to 40. Averagely, a downloading process can analyze about 10 tweets/second (about 10% tweets contain images). The number of downloading processes needs to be determined according to the tweet downloading speed. For example, 1 or 2 processes are enough for geo-tagged tweets in the contiguous U.S. (10 tweets/second), and 5 processes can handle the geo-tagged tweets around the world (50 tweets/second).

4.3 Training dataset building and CNN training

The downloading module was applied to obtain about 38,000 images from a list containing 140,000 tweets which have a keyword “flood” in the text. These tweets were posted from January 2016 to November 2017, covering the contiguous U.S. 38,000 downloaded photos were classified into 4,000 flooding and 34,000 non-flooding photos, and 4,000 non-flooding photos were randomly selected along with the all flooding photos to form the training dataset. Figure 4.1 and Figure 4.2 shows some flooding and non-flooding samples.

All the tested architectures achieve overall accuracy of more than 90% (Table 4.1) on the test set, including the simple neural network for building the preliminary training dataset. The result shows the feasibility of the proposed iterative method.

The VGG16 got the highest accuracy when training from scratch. This model was modified slightly by changing the number of neurons in the last fully connected layer from 4096 to 1024. For the tested CNNs, the same hyper-parameters were used: an initial learning rate of 0.002 decayed by a factor of 0.5 when the accuracy stops improving, a

momentum of 0.9, a weight decay of 0.0005, and a batch size of 400. SGD (Stochastic Gradient Descent) was utilized. All the architectures get a 99% accuracy when classifying the training set, indicating a overfitting (the training dataset needs more samples). The overfitting is expected because the CNNs have millions of parameters (e.g., VGG16 have 138 million) and the classes of the training samples might be memorized by the CNNs, leading the loss of generality. When using some data argumentation techniques, such as image flipping, the accuracy did not obtain improvement.

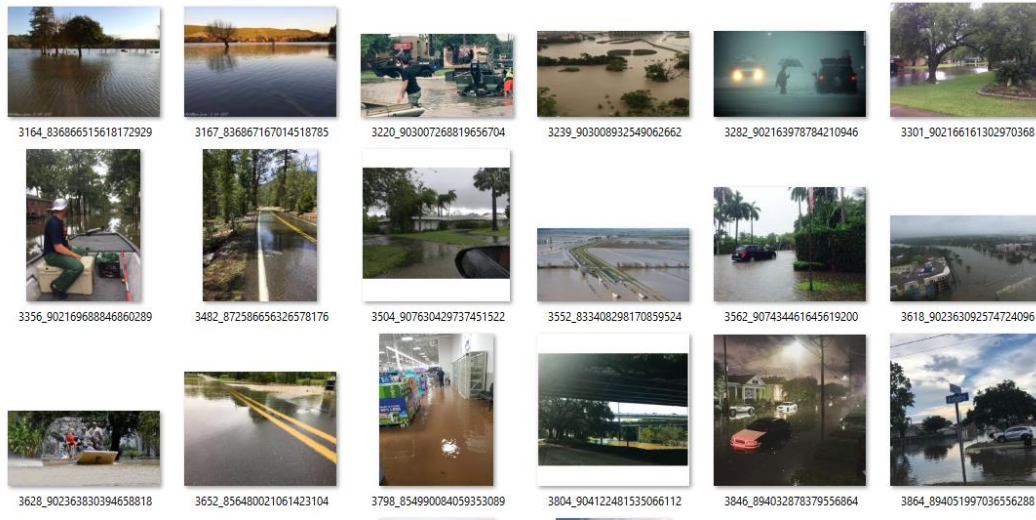


Figure 4.1 Samples of flooding photos

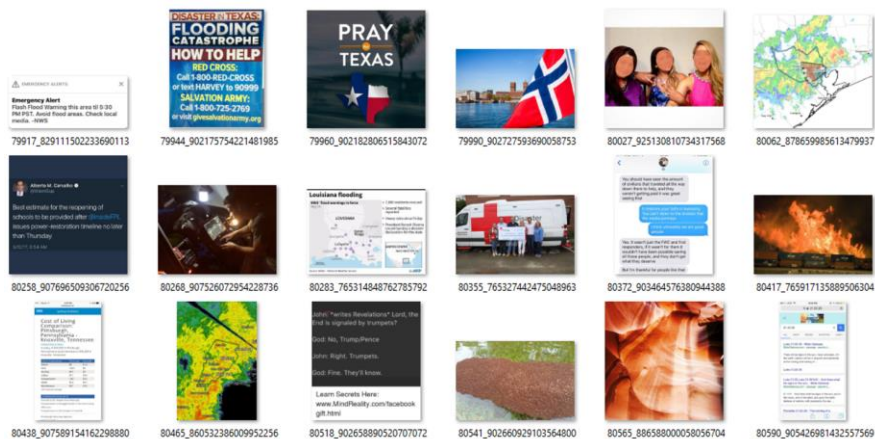


Figure 4.2 Samples of non-flooding photos

Table 4.1 Accuracies of tested CNNs

Network	Method	Total Accuracy
VGG16	Trained from scratch	93%
VGG16	Transfer learning	91%
Inception V3	Transfer learning	91%
ResNet 152	Transfer learning	91%
DenseNet201	Trained from scratch	91%
DenseNet201	Transfer learning	91%
Simple	3 conv, 1 fc	90%

4.3.1 Case study 1: Houston Flood 2017

The trained VGG16 was applied to a tweets dataset of Houston Flood 2017. This dataset has about 140,000 tweets with latitude and longitude in the metropolitan area where suffered an unprecedented flood caused by Harvey Hurricane. The posted time range from August 15, 2017 to October 1, 2017. 39,000 photos were downloaded and classified by the trained VGG16 CNN. 2,237 among them are labeled as flooding. Figure 4.3 and Figure 4.4 present some samples of the detection results. 1,400 of 2,237 are verified as real flooding after a manual check based on the rule in Table 3.2 and Table 3.3. Therefore the precision of flooding photo is 63% (1400/2237). Because of the labor-intensity, only 20% of the non-flooding results have been checked, and 15 flooding photos were found, which means about 75 non-flooding photos were ignored by the trained CNN. The recall of flooding photos is 95% (100% - 75/1475). This indicates that the classifier has an acceptable performance when applying it to the real data with a highly imbalanced distribution.

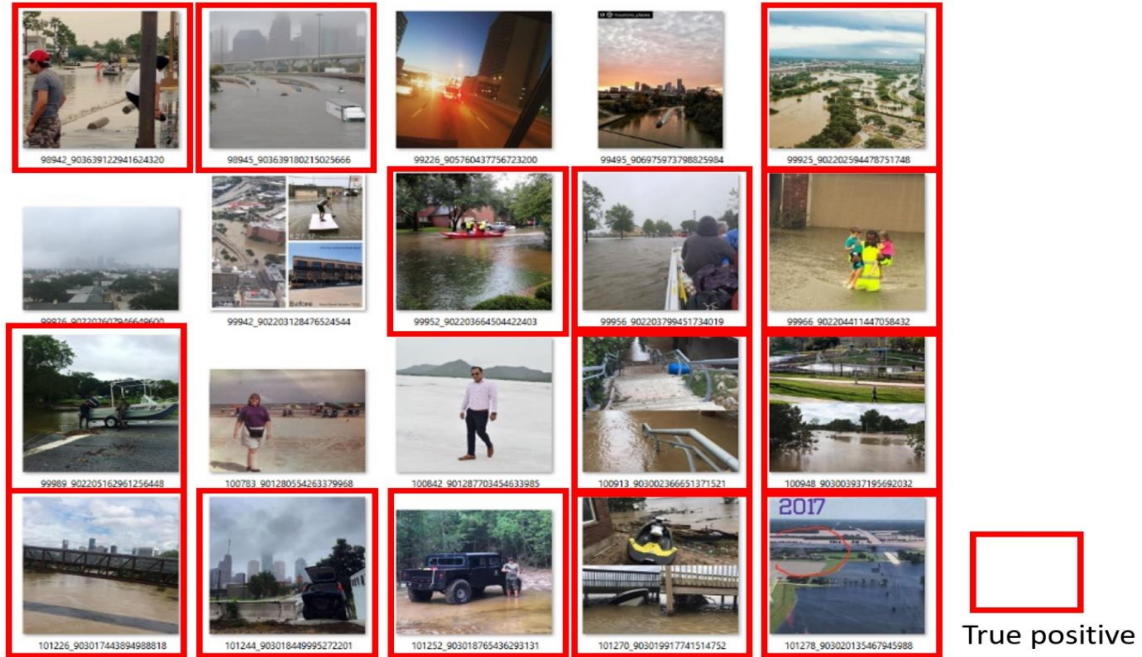


Figure 4.3 Photos identified as flooding by the neural network in Houston Flood 2017 dataset. These randomly selected 25 photos contains 14 real flooding photos (56%), and the whole 2,237 result looks having more than a half contain flood at first glance.

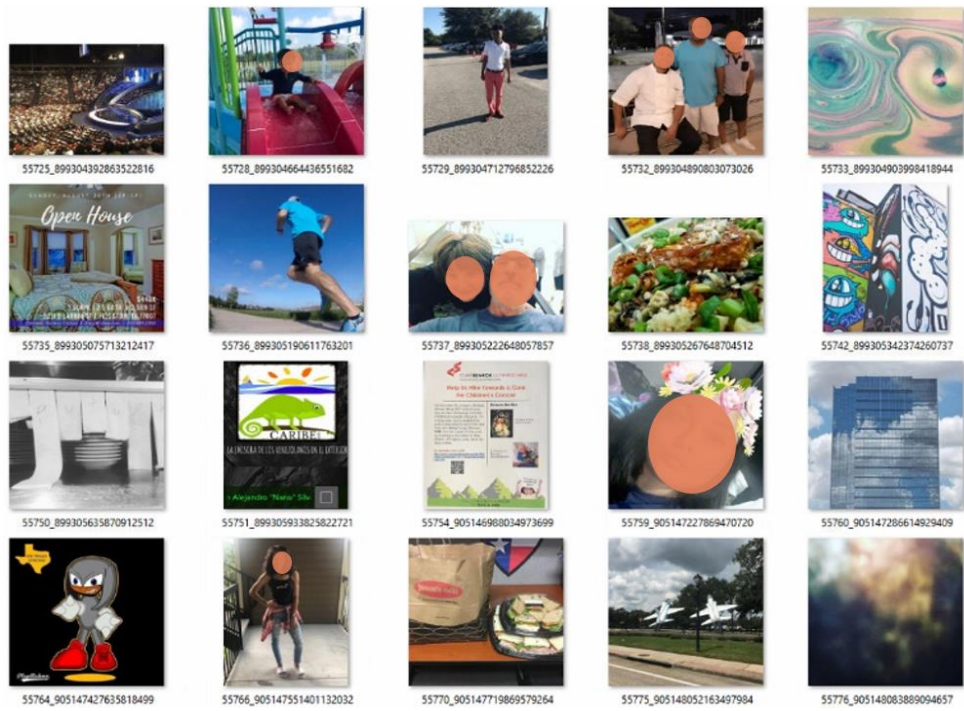


Figure 4.4 Non-flooding photos identified by the neural network in Houston Flood 2017 dataset. These randomly selected 25 photos all are correct, and most of 34,000 photos seem are non-flooding at first glance.

4.3.2 Case study 2: Hurricane Florence Flood 2018

September 14, 2018, Hurricane Florence made landfall near Wilmington, North Carolina, and caused 1,000 years rainfall which leads to broken record floods (Irfan 2018) in North Carolina and South Carolina. 6,975 images were downloaded from 136,000 geo-tagged tweets posted in Carolinas, and the trained VGG16 returned 818 flooding photos. 372 out of the 818 photos are true positives after a manual verification (Figure 4.5), so the precision is 45.5% (373/818), lower than the result in Houston Flood 2017. 44 tweets have latitude and longitude data. Their locations have some overlap with the High Water Mark (HWM) measured by USGS (Figure 4.6). However, the flooding photos are difficult to be located due to the inconsistency location between the shot spot and the posted spot. Another reason is the lack of distinguished buildings or objects in the photos. Figure 4.7 shows one of the located flooding photo based on Google StreetView.

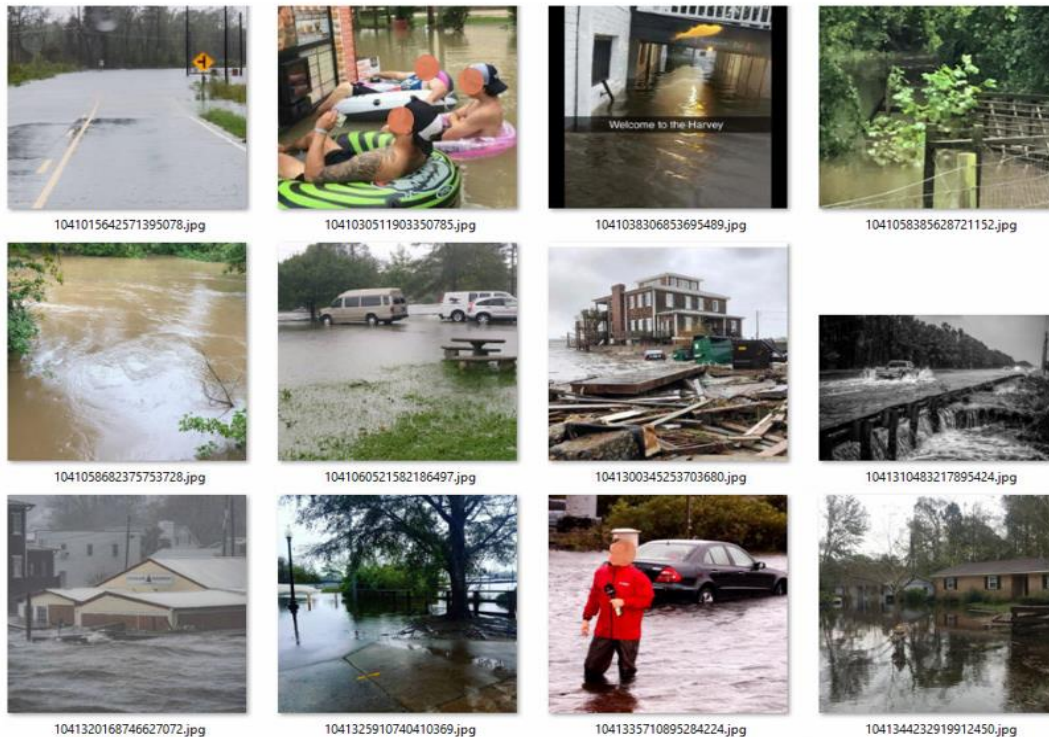


Figure 4.5 Samples of flooding photos posted during Hurricane Florence Flood.

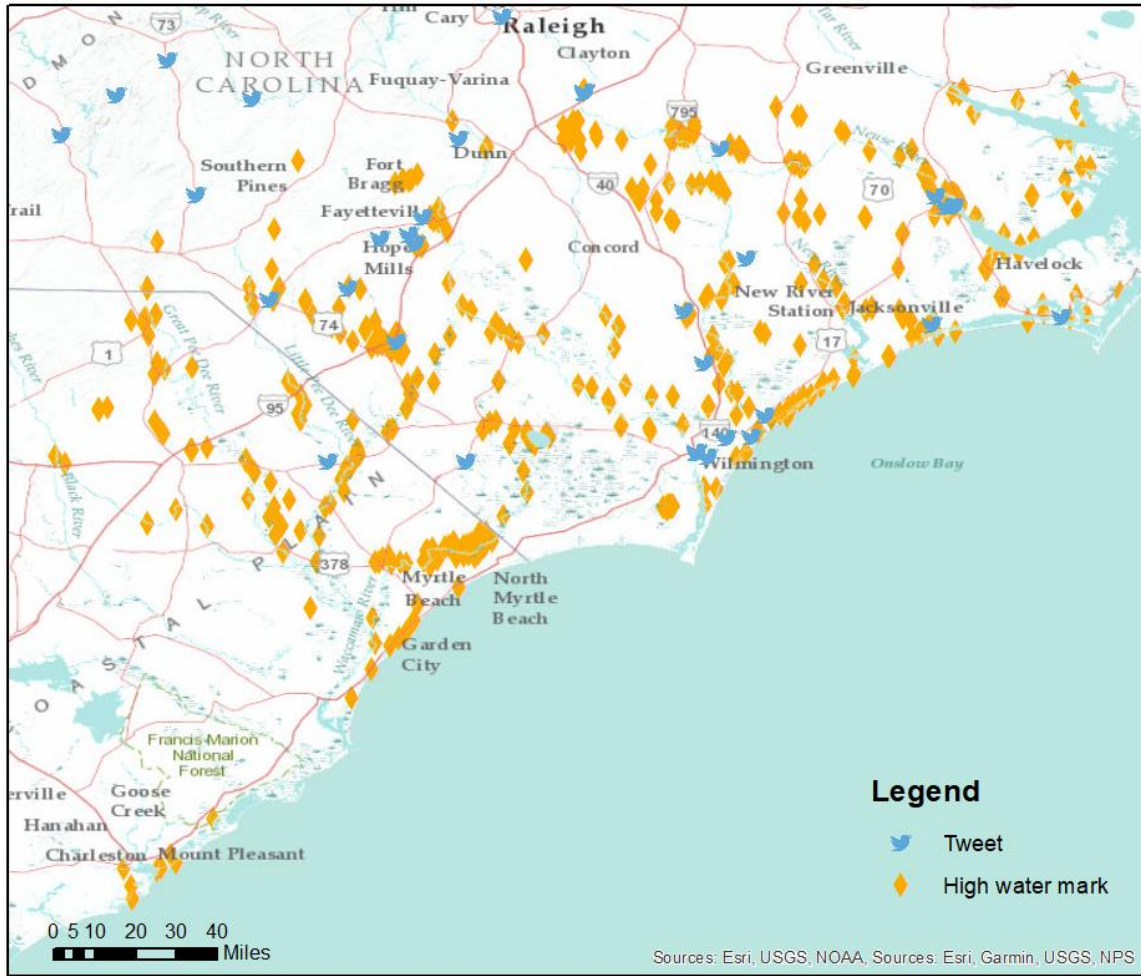
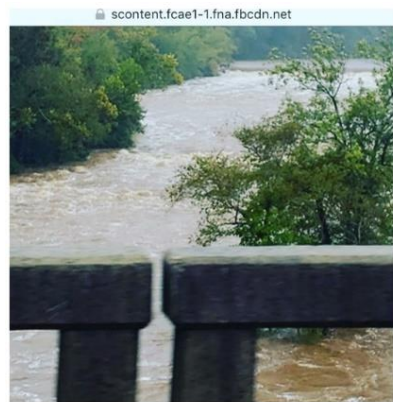
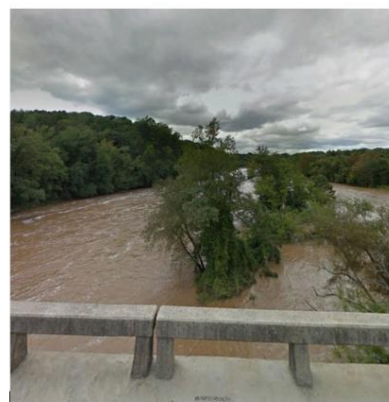


Figure 4.6 Tweets with latitude and longitude in Hurricane Florence Flood 2018.



a) Tweet photo



b) Google Street Map

Figure 4.7 One located flooding photo.

4.4 Crowdsourcing verification module

The crowdsourcing verification module was built based on Google Map (Figure 4.8). This WebGIS application reads the flooding photos from the MySQL database and displays photos and tweet texts on the map. The volunteers can determine whether a photo is flooding or not. If yes, they can record the water height to the database. Also, the location of the image can be verified based on the comparison with the Google Map. The information input by the volunteers would be recorded in the database.

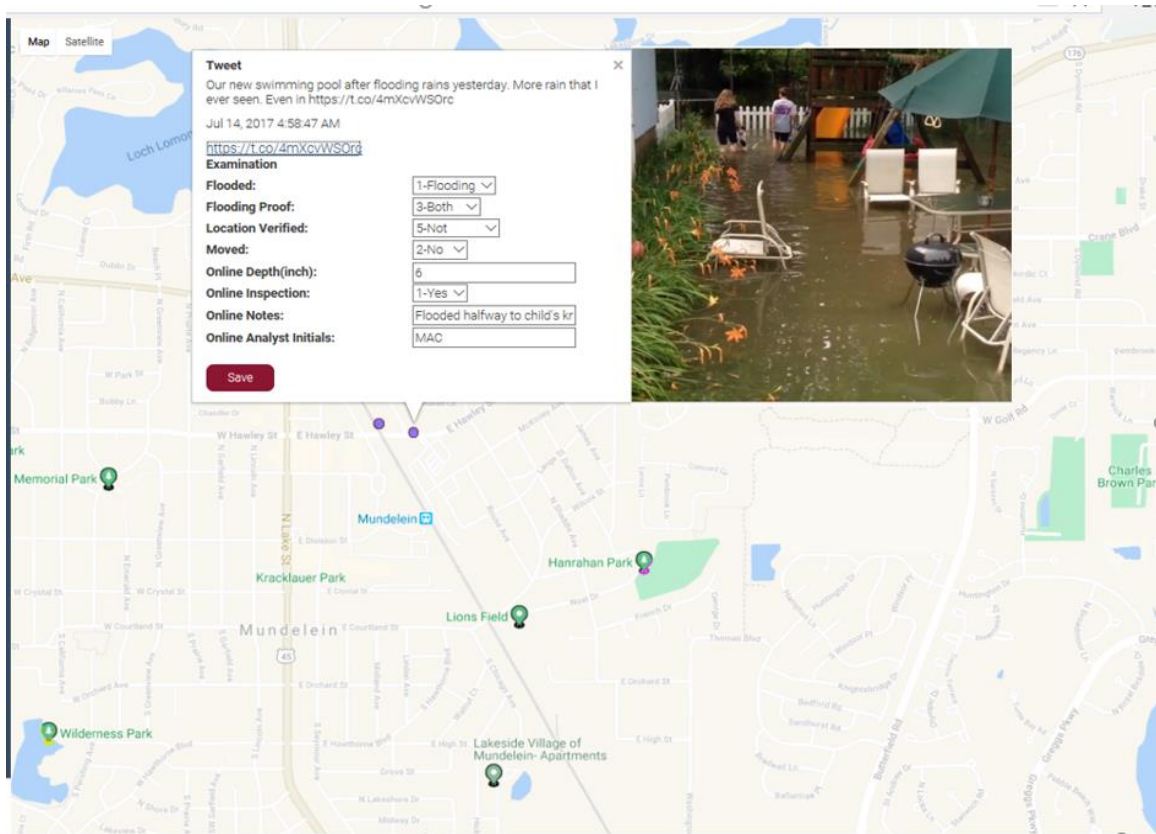


Figure 4.8 The WebGIS application for verification. The volunteer or other human operators can use it to verify the detection results and associate the photos with real geographic location.

CHAPTER 5 DISCUSSION AND CONCLUSION

5.1 Discussion

The research implemented a platform to collect, store, and analyze the images posted on Twitter.com in real-time. It provides a new practical approach to obtaining insights from the massive social media data. Given the current public Streaming API of Twitter.com, the proposed system can receive up to 50 tweets/second, and about 10% of tweets have images. Dozens of flooding photos can be collected each day in the first days of a flood event. This research did not use the paying API (Premium and Enterprise) to collect the tweets, so it is unknown whether the system can obtain more tweets using the paying API. However, the proposed system can process up to 120 tweets/second when using 12 processes so that this prototype system can deal with the tweets from a larger flooding region. Meanwhile, the filter based on keywords and geo-boundary, which provided in the system, can be used to obtain specific tweets and shrink the workload of tweets and images downloading.

About 5,000 flooding photos collected in the research come from the real social media images posted in the contiguous U.S., and these photos are beneficial to train the CNN to detect the newly posted flooding photos in the future flood events. However, these 5,000 flooding photos and the randomly selected non-flooding photos are still under representative for the social media images which have a large variance and a high imbalance of class distribution. This under-representation leads to a low precision when

applying the trained CNN to real-time social media images. Data augmentation, such as flipping and rotation have tested in training, but the results did not obtain improvement. More data augmentation methods need to be tested.

A worrying problem is that only 5,000 flooding photos were collected with only about 2,000 in the two study cases. Acquiring more flooding photos in the flooding events is a challenge in further research. Several approaches may work, such as collecting more images from non-geo-tagged tweets and using enterprise API of Twitter.com to access the entire dataset of the geotagged tweet.

The WebGIS application connected to this system provides a convenient tool to verify the classification results. Volunteers can remove the non-flooding photos, locate the flooding images, and estimate the water height from the image. The information from the photos by human volunteers is valuable for the further applications of the photos. For example, analyzing the false detection -- the non-flooding photos but labeled as flooding by the classifier. If the causes of the wrong labeling are learned and addressed, the performance of the classifier can be improved.

Other visual related studies can be easily conducted based on the system. The flooding classifier currently used in the system can be replaced by other classifier or detectors. A YOLO-v3 model was tested in this study, which can detect 80 classes of common objects (e.g., person, car, and cat). The detected results are stored in the database for further analysis, and the results reveal some interesting phenomena. For instance, in the U.S., cats appear less in social media photos than other places. Other visual-based models can also be used, such as violence detection (Won, Steinert-Threlkeld, and Joo 2017; Kalliatakis et al. 2017), face recognition (Schroff, Kalenichenko, and Philbin 2015), gender

and age extraction (Jia, Lansdall-Welfare, and Cristianini 2016), or skin color analysis (Ryu, Adam, and Mitchell 2017). For instance, according to the downloaded images, the preliminary statistics of race and gender detection based on the human face shows that Indian female appears remarkably less than Indian male, while White, Black, and Asian female and male appear equally. In addition, the system has been well-designed to store the tweets in multiple languages, including emojis. Other research on tweets text analyzes can be embedded into the system and conducted in real-time together with the image analysis. For example, using the text and images together to classify the flooding related tweets (X. Huang et al. 2018). It also has the potential to tackle the representative issues of Twitter data (Jiang, Li, and Ye 2019) by automatically extract the demographic information (e.g., gender, age, and race) from the tweet photos, which will benefit human mobility studies based on social media (Martín, Li, and Cutter 2017; Jiang, Li, and Cutter 2019).

Cross-culture studies based on the system are promising. Images are intuitive and language-free. According to the downloaded geo-tagged tweets, besides English, about 40% tweets are written in over 30 languages, such as Portuguese (13%), Spanish (9%), and Japanese (6%). Research based on the image content does not need to know those languages. A possible research topic is that calculating the frequencies of religious images among cultures. For text mining, the system has connected to the Google Translation API to translate tweets into English or other languages.

5.2 Conclusion

The deep learning-based social media flooding photo screening system provides a real-time pipeline to download the tweets and images, detect flooding photos, and verify the detection results. The images downloading module can process more than 100 tweets

per second, covering a large area, even global. The images in the tweets can be obtained and analyzed within 2 minutes after posting.

A training dataset of flooding photo containing 5,000 flooding samples was built. All the flooding and non-flooding photos are obtained from social media images, so they are representative when applied in analyzing social media images. These flooding photos were selected with a criterion of being on-site photos and containing land reference objects inundated by water. The decision makers can obtain visual information from the on-site photos. The non-user-generated contents, such as posters and advertisements, were not considered in this study, because are useless for disaster response and are difficult to be validated.

When applying the trained CNN by these samples to Houston Flood 2017, the recall of flooding photos is 95%, and the precision is 63%. The precision in Hurricane Florence Flood 2018 is 46%. Compared with the balanced test set, these results show two limits of the trained CNN. The first limit is the lower precision in the highly imbalanced social media images. During the flooding days in these two study cases, the flooding photos consist of less than 5% of the entire image set, which is far less than the test set (50%), the precision of the CNN dropped from more than 90% to about 50%, mislabeling many non-flooding photos as flooding. However, the recall of 95% is acceptable in Houston Flood 2017. Another limit is that, in the severe flood events, the threatened residents will be evacuated, leading a low number of social media posts. Therefore, the detected flooding photos may decrease due to the less social media posts.

A WebGIS application is included in this system for manually verifying the automatically detected flooding photos. The verified flooding photos can be added to the

training set which can further improve the performance of the CNN. Meanwhile, this WebGIS application serves two other purposes: 1) estimating the water height of the flood from the photo, and 2) validate the location of the flooding photo for generating inundation maps.

Not only is the system a useful tool for flooding responding, but also a platform for social media research based on images. The classifier of flooding photo can be replaced by other disaster image classifiers, such as tornados and wildfires. This extensibility has been verified by integrating an object detector (YOLO-v3) and a face detector to the system. In addition, a translation module is embedded to translate non-English tweets for cross-culture research.

REFERENCES

- Abadi, Martín, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, and Michael Isard. 2016. "Tensorflow: A System for Large-Scale Machine Learning." In *OSDI*, 16:265–283.
- Adam, N. R., B. Shafiq, and R. Staffin. 2012. "Spatial Computing and Social Media in the Context of Disaster Management." *IEEE Intelligent Systems* 27 (6): 90–96. <https://doi.org/10.1109/MIS.2012.113>.
- Bar, Yaniv, Idit Diamant, Lior Wolf, and Hayit Greenspan. 2015. "Deep Learning with Non-Medical Training Used for Chest Pathology Identification." In *Medical Imaging 2015: Computer-Aided Diagnosis*, 9414:94140V. International Society for Optics and Photonics. <https://doi.org/10.1117/12.2083124>.
- Bay, Herbert, Tinne Tuytelaars, and Luc Van Gool. 2006. "Surf: Speeded up Robust Features." In *European Conference on Computer Vision*, 404–417. Springer.
- Bischke, Benjamin, Prakriti Bhardwaj, Aman Gautam, Patrick Helber, Damian Borth, and Andreas Dengel. n.d. "Detection of Flooding Events in Social Multimedia and Satellite Imagery Using Deep Neural Networks," 3.
- Bischke, Benjamin, Patrick Helber, Christian Schulze, Venkat Srinivasan, Andreas Dengel, and Damian Borth. 2017. "The Multimedia Satellite Task at MediaEval."
- Burton, Jason. 2018. "USGS: Florence Set at Least 28 Flood Records in Carolinas." November 13, 2018. <https://www.usgs.gov/news/usgs-florence-set-least-28-flood-records-carolinas>.
- Chan, Celeste J. 2014. "Crowdsourcing Disaster Response." *ResearchGate Google Scholar*.
- Chen, Jenny J., Natala J. Menezes, Adam D. Bradley, and T. North. 2011. "Opportunities for Crowdsourcing Research on Amazon Mechanical Turk." *Interfaces* 5 (3).
- Cox, L. P. 2011. "Truth in Crowdsourcing." *IEEE Security Privacy* 9 (5): 74–76. <https://doi.org/10.1109/MSP.2011.145>.
- Druzhkov, P. N., and V. D. Kustikova. 2016. "A Survey of Deep Learning Methods and Software Tools for Image Classification and Object Detection." *Pattern Recognition and Image Analysis* 26 (1): 9–15.

- Estellés-Arolas, Enrique, Raúl Navarro-Giner, and Fernando González-Ladrón-de-Guevara. 2015. "Crowdsourcing Fundamentals: Definition and Typology." In *Advances in Crowdsourcing*, 33–48. Springer, Cham. https://doi.org/10.1007/978-3-319-18341-1_3.
- Feng, Yu, and Monika Sester. 2018. "Extraction of Pluvial Flood Relevant Volunteered Geographic Information (VGI) by Deep Learning from User Generated Texts and Photos." *ISPRS International Journal of Geo-Information* 7 (2): 39. <https://doi.org/10.3390/ijgi7020039>.
- Fohringer, J., D. Dransch, H. Kreibich, and K. Schröter. 2015. "Social Media as an Information Source for Rapid Flood Inundation Mapping." *Natural Hazards and Earth System Sciences* 15 (12): 2725–2738.
- Gao, Huiji, Geoffrey Barbier, Rebecca Goolsby, and Daniel Zeng. 2011. "Harnessing the Crowdsourcing Power of Social Media for Disaster Relief." Arizona State Univ Tempe.
- Gebru, Timnit, Jonathan Krause, Yilun Wang, Duyun Chen, Jia Deng, and Li Fei-Fei. 2017. "Fine-Grained Car Detection for Visual Census Estimation." In *AAAI*, 2:6.
- Goodchild, Michael F., and J. Alan Glennon. 2010. "Crowdsourcing Geographic Information for Disaster Response: A Research Frontier." *International Journal of Digital Earth* 3 (3): 231–241.
- Goutte, Cyril, and Eric Gaussier. 2005. "A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation." In *European Conference on Information Retrieval*, 345–359. Springer.
- He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. "Deep Residual Learning for Image Recognition." *ArXiv:1512.03385 [Cs]*, December. <http://arxiv.org/abs/1512.03385>.
- Hu, Jie, Li Shen, and Gang Sun. 2017. "Squeeze-and-Excitation Networks." *ArXiv:1709.01507 [Cs]*, September. <http://arxiv.org/abs/1709.01507>.
- Huang, Gao, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2016. "Densely Connected Convolutional Networks." *ArXiv:1608.06993 [Cs]*, August. <http://arxiv.org/abs/1608.06993>.
- Huang, Xiao, Cuizhen Wang, and Zhenlong Li. 2018a. "A near Real-Time Flood-Mapping Approach by Integrating Social Media and Post-Event Satellite Imagery." *Annals of GIS*, March. <https://www.tandfonline.com/doi/abs/10.1080/19475683.2018.1450787>.
- . 2018b. "Reconstructing Flood Inundation Probability by Enhancing Near Real-Time Imagery With Real-Time Gauges and Tweets." *IEEE Transactions on*

- Geoscience and Remote Sensing* 56 (8): 4691–4701.
<https://doi.org/10.1109/TGRS.2018.2835306>.
- Huang, Xiao, Cuizhen Wang, Zhenlong Li, and Huan Ning. 2018. “A Visual–Textual Fused Approach to Automated Tagging of Flood-Related Tweets during a Flood Event.” *International Journal of Digital Earth* 0 (0): 1–17.
<https://doi.org/10.1080/17538947.2018.1523956>.
- Irfan, Umair. 2018. “Hurricane Florence’s ‘1,000-Year’ Rainfall, Explained.” Vox. September 20, 2018. <https://www.vox.com/2018/9/20/17883492/hurricane-florence-rain-1000-year>.
- Jia, S., T. Lansdall-Welfare, and N. Cristianini. 2016. “Gender Classification by Deep Learning on Millions of Weakly Labelled Images.” In *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*, 462–67.
<https://doi.org/10.1109/ICDMW.2016.0072>.
- Jiang, Yuqin, Zhenlong Li, and Susan L. Cutter. 2019. “Social Network, Activity Space, Sentiment and Evacuation: What Can Social Media Tell Us?” *Annals of the American Association of Geographers*.
- Jiang, Yuqin, Zhenlong Li, and Xinyue Ye. 2019. “Understanding Demographic and Socioeconomic Biases of Geotagged Twitter Users at the County Level.” *Cartography and Geographic Information Science* 46 (3): 228–42.
<https://doi.org/10.1080/15230406.2018.1434834>.
- Kalliatakis, Grigorios, Shoaib Ehsan, Maria Fasli, Ales Leonardis, Juergen Gall, and Klaus McDonald-Maier. 2017. “Detection of Human Rights Violations in Images: Can... - Google Scholar.”
https://scholar.google.com/scholar?hl=en&as_sdt=0%2C41&q=Detection+of+Human+Rights+Violations+in+Images%3A+Can+Convolutional+Neural+Networks+help&btnG=.
- Koenig, Todd A., Jennifer L. Bruce, Jim O’Connor, Benton D. McGee, Robert R. Holmes Jr., Ryan Hollins, Brandon T. Forbes, et al. 2016. “Identifying and Preserving High-Water Mark Data.” USGS Numbered Series 3-A24. Techniques and Methods. Reston, VA: U.S. Geological Survey.
<http://pubs.er.usgs.gov/publication/tm3A24>.
- Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton. 2012. “ImageNet Classification with Deep Convolutional Neural Networks.” In *Advances in Neural Information Processing Systems 25*, edited by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, 1097–1105. Curran Associates, Inc.
<http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>.
- LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. 2015. “Deep Learning.” *Nature* 521 (7553): 436–44. <https://doi.org/10.1038/nature14539>.

- Li, Zhenlong, Cuizhen Wang, Christopher T. Emrich, and Diansheng Guo. 2018. "A Novel Approach to Leveraging Social Media for Rapid Flood Mapping: A Case Study of the 2015 South Carolina Floods." *Cartography and Geographic Information Science* 45 (2): 97–110.
<https://doi.org/10.1080/15230406.2016.1271356>.
- Lowe, D.G. 1999. "Object Recognition from Local Scale-Invariant Features." In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 1150–57 vol.2. Kerkyra, Greece: IEEE.
<https://doi.org/10.1109/ICCV.1999.790410>.
- Martín, Yago, Zhenlong Li, and Susan L. Cutter. 2017. "Leveraging Twitter to Gauge Evacuation Compliance: Spatiotemporal Analysis of Hurricane Matthew." *PloS One* 12 (7): e0181701. <https://doi.org/10.1371/journal.pone.0181701>.
- Nagarajan, Meena, Amit Sheth, and Selvam Velmurugan. 2011. "Citizen Sensor Data Mining, Social Media Analytics and Development Centric Web Applications." In *Proceedings of the 20th International Conference Companion on World Wide Web*, 289–290. WWW '11. New York, NY, USA: ACM.
<https://doi.org/10.1145/1963192.1963315>.
- National Weather Service. 2017. "Summary of Natural Hazard Statistics for 2017 in the United States."
- Paszke, Adam, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. "Automatic Differentiation in PyTorch," October.
<https://openreview.net/forum?id=BJJsrmfCZ>.
- Redmon, Joseph, and Ali Farhadi. 2018. "YOLOv3: An Incremental Improvement." *ArXiv:1804.02767 [Cs]*, April. <http://arxiv.org/abs/1804.02767>.
- Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, et al. 2015. "ImageNet Large Scale Visual Recognition Challenge." *International Journal of Computer Vision* 115 (3): 211–52.
<https://doi.org/10.1007/s11263-015-0816-y>.
- Ryu, Hee Jung, Hartwig Adam, and Margaret Mitchell. 2017. "InclusiveFaceNet: Improving Face Attribute Detection with Race and Gender Diversity." *ArXiv:1712.00193 [Cs]*, December. <http://arxiv.org/abs/1712.00193>.
- Sayce, David. 2018. "Number of Tweets per Day?" David Sayce. 2018.
<https://www.dsayce.com/social-media/tweets-day/>.
- Schroff, Florian, Dmitry Kalenichenko, and James Philbin. 2015. "FaceNet: A Unified Embedding for Face Recognition and Clustering." In , 815–23. https://www.cv-foundation.org/openaccess/content_cvpr_2015/html/Schroff_FaceNet_A_Unified_2015_CVPR_paper.html.

- Sheth, A. 2009. "Citizen Sensing, Social Signals, and Enriching Human Experience." *IEEE Internet Computing* 13 (4): 87–92. <https://doi.org/10.1109/MIC.2009.77>.
- Shin, H., H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers. 2016. "Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning." *IEEE Transactions on Medical Imaging* 35 (5): 1285–98. <https://doi.org/10.1109/TMI.2016.2528162>.
- Simonyan, Karen, and Andrew Zisserman. 2014. "Very Deep Convolutional Networks for Large-Scale Image Recognition." *ArXiv:1409.1556 [Cs]*, September. <http://arxiv.org/abs/1409.1556>.
- Sladojevic, Srdjan, Marko Arsenovic, Andras Anderla, Dubravko Culibrk, and Darko Stefanovic. 2016. "Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification." Research article. *Computational Intelligence and Neuroscience*. 2016. <https://doi.org/10.1155/2016/3289801>.
- Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. "Rethinking the Inception Architecture for Computer Vision." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2818–2826.
- Tajbakhsh, N., J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang. 2016. "Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning?" *IEEE Transactions on Medical Imaging* 35 (5): 1299–1312. <https://doi.org/10.1109/TMI.2016.2535302>.
- Thomee, Bart, David A. Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. 2016. "YFCC100M: The New Data in Multimedia Research." *Commun. ACM* 59 (2): 64–73. <https://doi.org/10.1145/2812802>.
- Tkachenko, Nataliya, Arkaitz Zubiaga, and Rob Procter. 2017. "WISC at MediaEval 2017: Multimedia Satellite Task," November, 4.
- Union of Concerned Scientists. 2018. "Climate Change, Extreme Precipitation and Flooding: The Latest Science (2018)." Union of Concerned Scientists. June 2018. <https://www.ucsusa.org/global-warming/global-warming-impacts/floods>.
- U.S. Department of the Interior, and U.S. Geological Survey. 2019. "USGS Flood Event Viewer." March 2019. <https://water.usgs.gov/floods/FEV/>.
- Wang, Cuizhen, Zhenlong Li, and Xiao Huang. 2018. "Geospatial Assessment of Flooding Dynamics and Risks of the October'15 South Carolina Flood." *Southeastern Geographer*, Congaree River Watershed, 58 (2): 164–80.

- Wang, Dayong, Aditya Khosla, Rishab Gargeya, Humayun Irshad, and Andrew H. Beck. 2016. “Deep Learning for Identifying Metastatic Breast Cancer.” *ArXiv:1606.05718 [Cs, q-Bio]*, June. <http://arxiv.org/abs/1606.05718>.
- Won, Donghyeon, Zachary C. Steinert-Threlkeld, and Jungseock Joo. 2017. “Protest Activity Detection and Perceived Violence Estimation from Social Media Images.” In *Proceedings of the 25th ACM International Conference on Multimedia*, 786–794. MM ’17. New York, NY, USA: ACM. <https://doi.org/10.1145/3123266.3123282>.
- Wuebbles, Donald J., David W. Fahey, and Kathy A. Hibbard. 2017. “Climate Science Special Report: Fourth National Climate Assessment, Volume I.”
- Yoon, Young-Chul, and Kuk-Jin Yoon. 2018. “Animal Detection in Huge Air-View Images Using CNN-Based Sliding Window.” In *International Workshop on Frontiers of Computer Vision (IWFCV)*. International Workshop on Frontiers of Computer Vision (IWFCV).
- Zhang, Rong, Weiping Li, and Tong Mo. 2018. “Review of Deep Learning.” *ArXiv:1804.01653 [Cs, Stat]*, April. <http://arxiv.org/abs/1804.01653>.
- Zook, Matthew, Mark Graham, Taylor Shelton, and Sean Gorman. 2010. “Volunteered Geographic Information and Crowdsourcing Disaster Relief: A Case Study of the Haitian Earthquake.” *World Medical & Health Policy* 2 (2): 7–33.

APPENDIX A: THE MAIN STRUCTURE OF TWEET JSON

```
{
  "created_at" : "Thu Apr 06 15:24:15 +0000 2017" ,
  "id_str" : "850006245121695744" ,
  "text" : "1\ Today we\u2019re sharing our vision for the future of the T
  "user" : {
    "id" : 2244994945 ,
    "name" : "Twitter Dev" ,
    "screen_name" : "TwitterDev" ,
    "location" : "Internet" ,
    "url" : "https://dev.twitter.com/" ,
    "description" : "Your official source for Twitter Platform news, update
  } ,
  "place" : {
  } ,
  "entities" : {
    "hashtags" : [
    ] ,
    "urls" : [
      {
        "url" : "https://t.co/XweGngmx1P" ,
        "unwound" : {
          "url" : "https://cards.twitter.com/cards/18ce53wgo4h/3xolc"
          "title" : "Building the Future of the Twitter API Platform"
        }
      }
    ] ,
    "user_mentions" : [
    ]
  }
}
```

Figure A.1 A Tweet in JSON format.

```

{
  "geo" : null ,
  "coordinates" : null ,
  "place" : {
    "id" : "07d9db48bc083000" ,
    "url" : "https://api.twitter.com/1.1/geo/id/07d9db48bc083000.json" ,
    "place_type" : "poi" ,
    "name" : "McIntosh Lake" ,
    "full_name" : "McIntosh Lake" ,
    "country_code" : "US" ,
    "country" : "United States" ,
    "bounding_box" : {
      "type" : "Polygon" ,
      "coordinates" : [
        [
          [
            - 105.14544 ,
            40.192138
          ] ,
          [
            - 105.14544 ,
            40.192138
          ] ,
          [
            - 105.14544 ,
            40.192138
          ] ,
          [
            - 105.14544 ,
            40.192138
          ]
        ]
      ]
    } ,
    "attributes" : {
  }
}
}

```

Figure A.2 Tweet JSON with Twitter Place.

```






{
  "geo" : {
    "type" : "Point" ,
    "coordinates" : [
      40.74118764 ,
      - 73.9998279
    ]
  } ,
  "coordinates" : {
    "type" : "Point" ,
    "coordinates" : [
      - 73.9998279 ,
      40.74118764
    ]
  } ,
  "place" : {
    "id" : "01a9a39529b27f36" ,
    "url" : "https://api.twitter.com/1.1/geo/id/01a9a39529b27f36.json" ,
    "place_type" : "city" ,
    "name" : "Manhattan" ,
    "full_name" : "Manhattan, NY" ,
    "country_code" : "US" ,
    "country" : "United States" ,
    "bounding_box" : {
      "type" : "Polygon" ,
      "coordinates" : [
        [
          [
            - 74.026675 ,
            40.683935
          ] ,
          [
            - 74.026675 ,
            40.877483
          ] ,
          [
            - 73.910408 ,
            40.877483
          ] ,
          [
            - 73.910408 ,
            40.683935
          ]
        ]
      ]
    } ,
    "attributes" : {
  }
}
}

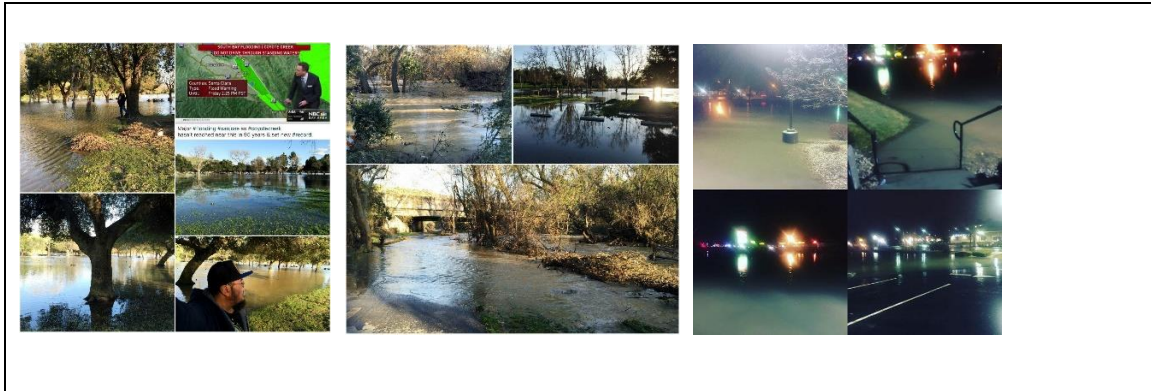
```

Figure A.3 Tweet JSON with exact location

APPENDIX B: EXAMPLES FOR THE FLOODING AND NON-FLOODING PHOTO

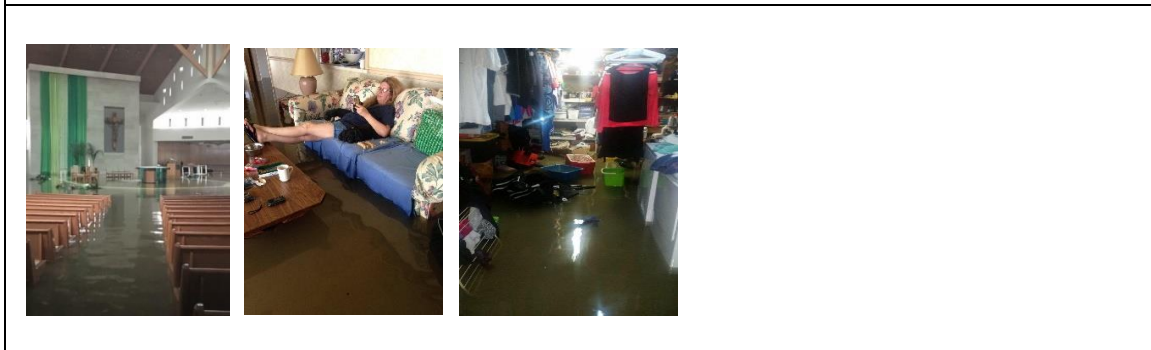
Table B.1 Examples of identifying flooding photo.

No.1	Description	Photos with clear references inundated by water outdoors.
Examples		
<div style="display: flex; flex-wrap: wrap; justify-content: space-around;">      </div>		
No.2	Description	The indoors photos with clear references inundated by water.
Examples		



No.3	Description	A mosaic image contains current flooding photos.
------	-------------	--

Examples






No.4	Description	The flooding photos in No.1 – No. 3 with a minor note from the uploader.
------	-------------	--



Examples



Table B.2 Examples of identifying non-flooding photo.

No.1	Description	Flooding photos from other mass media or social network users.
	Reason	Cannot be considered as the first-hand information.
Examples		
No.2	Description	Photos with thin water.
	Reason	The situation is still under control, not a flood.
Examples		
No.3	Description	The high water level in a river but inundating nothing.

	Reason	The situation is still under control, not a flood.
Examples		
		
No.4	Description	Advertisements or illustrations with flooding photo as background.
	Reason	Cannot indicate an ongoing flood.
Examples		
		
No.5	Description	No water in the photo.
	Reason	Cannot indicate an ongoing flood.
Examples		
		

No.6	Description	Modified flooding photo.
	Reason	Cannot provide reliable information about the current flood.
Examples		
		
No.7	Description	Fake flooding photo.
	Reason	Cannot provide reliable information about the current flood.
Examples		
		
No.8	Description	Historical flooding photos.
	Reason	Cannot provide reliable information about the current flood.
Examples		



No.9	Description	Only water bodies without reference.
	Reason	Cannot judge whether there is a flood.

Examples

