Human-Guided Flood Mapping: From Experts to the Crowd

Jiongqian Liang
The Ohio State University
liang.420@osu.edu

Peter Jacobs
The Ohio State University
jacobs.269@osu.edu

Srinivasan Parthasarathy
The Ohio State University
srini@cse.ohio-state.edu

ABSTRACT

Hurricane-induced flooding can lead to substantial loss of life and huge damage to infrastructure. Mapping flood extent from satellite or aerial imagery is essential for prioritizing relief efforts and for assessing future flood risk. Identification of water extent in such images can be challenging considering the heterogeneity in water body size and shape, cloud cover, and natural variations in land cover. In this effort, we introduce a novel cognitive framework based on a semi-supervised learning algorithm, called HUman-Guided Flood Mapping (`HUG-FM`), specifically designed to tackle the flood mapping problem. Our framework first divides the satellite or aerial image into patches leveraging a graph-based clustering approach. A domain expert is then asked to provide labels for a few patches (as opposed to pixels which are harder to discern). Subsequently, we learn a classifier based on the provided labels to map flood extent. We test the efficacy and efficiency of our framework on imagery from several recent flood-induced emergencies and results show that our algorithm can robustly and correctly detect water areas compared to the state-of-the-art. We then evaluate whether expert guidance can be replaced by the wisdom of a crowd (e.g., crisis volunteers). We design an online crowdsourcing platform based on `HUG-FM` and propose a novel ensemble method to leverage crowdsourcing efforts. We conduct an experiment with over 50 participants and show that crowdsourced `HUG-FM` (`CHUG-FM`) can approach or even exceed the performance of a single expert providing guidance (`HUG-FM`).

ACM Reference Format:
Jiongqian Liang, Peter Jacobs, and Srinivasan Parthasarathy. 2018. Human-Guided Flood Mapping: From Experts to the Crowd . In *The 2018 Web Conference Companion, April 23–27, 2018, Lyons, France.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3184558.3186339

1 INTRODUCTION

Each year, many areas on earth are impacted by severe flooding, causing serious loss of life and economy [14]. Flood extent mapping can be utilized to guide first responders to where they are most needed. This information can also be used to monitor and predict future flood risk in these areas.

To map flood extent, satellite images can be extremely useful due to their low cost and consistent, repetitive data acquisition capability over large spatial areas [27, 29]. Depending on availability these can also be supplemented with aerial flyover imagery. Compared to sparse *in situ* physical sensing data (e.g., river gauge data and weather station records), satellite images offer a synoptic view of the landscape and provide a comprehensive geospatial perspective on flood events. The challenge here is to correctly identify flooded areas given confounding factors ranging from cloud cover to refractive materials within urban areas.

This problem can be modeled as an image segmentation task [4, 6, 23, 28], where one wants to delineate flooded areas within a region. A challenge for such methods is the need to identify many diversely shaped segments (e.g. sinuous rivers) while accommodating various types of land-cover forms (e.g. swamps). In addition, current techniques generally do not scale well to high-resolution satellite images. Finally, the difference between flooded regions and other regions can be so subtle that human guidance is often required to accurately map floods.

To address these difficulties, we propose a framework based on ideas from *cognitive computing*. *Cognitive computing* refers to "systems that learn at scale, reason with purpose and interact with humans naturally" [15]. City, regional and national emergency systems are increasingly relying on such "smart" systems to simulate human thought process to solve real-world problems with humans and computers interacting with one another and providing necessary decision support to supplement traditional decision making.

Our proposed cognitive framework, called HUman Guided Flood Mapping (`HUG-FM`), integrates ideas from graph clustering and semi-supervised learning with human guidance, to accurately realize post-disaster flood maps from aerial or satellite imagery. Graph clustering approaches are used to divide images into patches (easier to label than individual pixels) and guidance from expert user is utilized to provide initial labels for a few patches (water or land). The method for flood mapping involves segmentation of satellite images of a given area *both* before and after a flood occurs. This is followed by a comparison of these pre-disaster and post-disaster segmentations to identify flooded vs. non-flooded areas. We run `HUG-FM` on satellite images of Chennai, India during the 2015 flood, Houston, Texas after the 2016 flood, and Lumberton, North Carolina during Hurricane Matthew in 2017. Experimental results show that our method can effectively identify flooded areas when compared to state-of-the-art approaches from both the remote sensing and computer vision communities. Our method is also more efficient, enabling real-time incremental learning and providing useful information to help prioritize post-disaster repair and relief activities.

We additionally extend our efforts to develop a crowdsourced variant of `HUG-FM` (called `CHUG-FM`), where we replace domain-expert guidance with the wisdom of the crowd (e.g., crisis volunteers). To test the crowdsourcing platform (`CHUG-FM`), we recruited over 50 volunteers to conduct interactive flood mapping on three different satellite images. We develop and deploy a novel ensemble learning method to integrate the crowdsourcing efforts and find it improves the performance of flood mapping when compared to `HUG-FM` (operating with a domain expert).

WWW'18 Companion, April 23–27, 2018, Lyon, France

ACM ISBN 978-1-4503-5640-4/18/04.
https://doi.org/10.1145/3184558.3186339

2 RELATED WORK

2.1 Image Segmentation

Image segmentation is a long-standing problem in computer vision [1, 2, 4, 6, 23, 28]. Classic methods for image segmentation include thresholding-based techniques wherein a pixel intensity threshold T forces all pixels with intensity above T to be one color, while all pixels with intensity below T become another color [1]. Picking the value of T is the main challenge in such methods and several automated approaches have been developed for this purpose [23, 30]. A popular method in the remote sensing community for picking T, is known as Otsu thresholding [23], involves finding the pixel intensity that creates the greatest separation and least overlap between the modes in the pixel intensity histogram. A weakness of these thresholding methods for remote sensing and flood mapping is that they are susceptible to noise and often generate too many tiny spots (flooded segments).

Other methods of image segmentation include the region merging technique proposed by Baatz et al. [2], which greedily groups similar pixels in an hierarchical fashion. More recently, graph-based methods have been introduced for image segmentation. They formulate the image as a graph, and adopt either spectral or clustering methods to conduct segmentation. Shi et al. [28] create a graph with weighted edges and use the normalized cut criterion to segment the image. Browet et al. [6] also formulate the image as a graph; they use modularity as a criterion to find a segmentation for the image. However, these methods can be computationally expensive and typically are not scalable on large satellite images.

Researchers have also investigated semi-supervised learning approaches for image segmentation [4, 16]. One influential semi-supervised method for mapping floods is the Watershed algorithm, developed by Beucher and Meyer [4]. It requires the user to mark different pixels in the image and utilizes a region growing technique to merge pixels starting from the provided markers. While the method is interactive and incremental in nature, the Watershed algorithm requires at least one marker for each segment, which can be inefficient in the scenario of flood mapping in an urban setting.

2.2 Semi-supervised Methods

Semi-supervised classification uses unlabeled data in addition to some amount of labeled data to learn a classifier. This type of learning has become widely used in recent years [17, 22, 33]. A common approach within this category is graph-based semi-supervised learning [3, 5]. This genre of methods leverages the graph structure, which is either obtained from additional data sources or derived from the original data. In general, these methods use graph structure as a regularizer to the loss function by assuming that nearby nodes in the graph should have similar labels. These methods are not suitable for our problem as they usually require much more labeled data for training and do not scale to high resolution data.

2.3 Flood Mapping

Flood mapping on satellite images has been the focus of much prior work in the remote sensing community. Many of these works rely on variations of the idea of thresholding and work in a purely unsupervised fashion [12, 21]. For example, Giustarini et al. lever a probabilistic flood mapping procedure (based on Gaussian mixture models) to segment flooded regions from dry regions [12].

There exist a few flood mapping approaches that leverage human supervision [19, 31]. Martinez et al. [19] collect labels from aerial images and ground observations while adopting a supervised method on SAR images to map the flood temporal dynamics. Semi-supervised learning [31] methods adopt the idea of region-growing following different ways to model the change of pixel intensity in the image. These methods usually require a large number of labels to achieve desirable performance and do not naturally support interactive and crowdsourced flood mapping.

2.4 Crowdsourcing in Emergency Response

Crowdsourcing has also been shown to be valuable for emergency response during times of disaster [11, 13, 20]. A recent example is the searching for Malaysian Flight 370. Authorities released satellite images and the public helped in efforts to locate the missing aircraft [20]. In addition, some researchers have investigated the important role that crowdsourcing can play during flood disasters. Degrossi et al. studied the usage of social media for collecting useful information such as water height and geo-location information to conduct flood risk management [9]. While existing work does study how crowd-based information can be applied to collect information for a few locations during a flood disaster, we are not aware of other efforts that lever crowdsourcing to generate a holistic flood map for a disaster area.

3 HUMAN-GUIDED FLOOD MAPPING

The design goals for our flood mapping system can be stated as:

(1) **Quality:** Generate accurate flood mappings with only limited supervision.
(2) **Efficiency:** Conduct efficient and scalable flood mapping for large satellite images. Efficiency is necessary to facilitate interactive learning; it is also vital if the method is to be used to help guide emergency first responders in a flood disaster.
(3) **Interactivity and Ease-of-Use:** Effectively incorporate expert guidance from domain experts or from the crowd.

With these desiderata in mind, we now describe our framework for flood mapping in detail below.

3.1 Preprocessing

To label areas of a satellite image as either land or water, we need to decide on a primary unit for labeling. A straightforward technique is to treat each pixel as a unit and conduct pixel-based labeling. The drawback of this method is that we lose information derived from geographic correlations between pixels (a neighboring pixel of a water area is more likely to be water and vice versa). A pixel-based approach is not only sensitive to noise effects (common in such imagery) and but also prone to labeling error [1]. Another alternative is to conduct uniform grouping, which divides the image into many parts of uniform size. However, without using the pixel intensity information from the image, this grouping can go across land-water boundaries leading to cognitive dissonance when labeling patches. To avoid such problems, we adopt an efficient graph-based approach for patch generation, which can both effectively detect regions of

[1] We have empirically verified both these limitations of pixel-based methods on a range of real world flood scenarios, but due to paucity of space cannot include these results in Section 5.

different sizes and largely avoid generating regions across land-water boundaries.

3.1.1 Graph Construction. Graph-based segmentation has been widely studied in the literature [6, 10, 28]. In this paper, we convert a given image into an undirected graph following the approach proposed by Cour et al. [8]. Each pixel of the image is treated as one node and each pixel has edges to nearby pixels within a distance d_{max}, where d_{max} is a pre-defined parameter. The weight of the edge between pixel i and pixel j is defined as follows:

$$w_{ij} = \begin{cases} e^{-\frac{d(i,j)}{\sigma_x^2} - \frac{|F(i)-F(j)|^2}{\sigma_y^2}} & \text{if } d(i,j) < d_{max} \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

where $d(i,j)$ is the Euclidean distance between pixels i and j and $F(i)$ is a feature vector evaluated at pixel i. Depending on the data source, the feature vector can be the grayscale value or the RGB values of the pixel. σ_x, σ_y and d_{max} are parameters that need to be determined ahead of time[2].

Note that the number of nodes in the constructed graph n is equal to the number of pixels in the image, and the number of edges is $m = \alpha * n$, where α is a small constant factor depending on the setting of d_{max}.

3.1.2 Graph Clustering to Generate Patches. After we construct the graph from the image, we cluster the graph to generate patches. Since a satellite image usually contains hundreds of millions of pixels, we need a highly scalable graph clustering algorithm. In this work, we leverage Multi-level Regularized Markov Clustering (MLR-MCL) [24, 25], a scalable graph clustering software[3].

Since the goal of graph clustering is to generate basic units for labeling, we tend to produce a large number of clusters. Empirically, we find the method works well when the average size of a cluster (patch) is a few hundred pixels. Once we obtain the graph clustering results, pixels in the same cluster are considered to be a *patch*.

There are many advantages to producing patches using this approach. First of all, unlike uniform grouping, this approach is better at avoiding cognitively discordant scenarios where clusters traverse natural image boundaries (e.g. water and land in the same patch). Secondly, controlling the number of patches generated is straightforward using MLR-MCL (we do not have to pre-specify number of patches/clusters). Finally, MLR-MCL has time complexity linear to the number of edges and is very efficient when run on the constructed graph, where the number of edges is proportional to the number of nodes.

3.2 Cognitive Expert-guided Labeling

After generating patches, the next step is to ask the human expert to identify and label a few patches. The expert user will place a few markers in the image to label a few patches that they identify as land or water. We rely on the expert to select patches to label – ideally focusing on hard-to-label patches (e.g. sinuous river beds, flooding in urban zones, or swamp regions). We will discuss strategies to automate this process in Section 4. To utilize this human-provided supervision, a binary classifier is then learned and subsequently applied to the rest of the unlabeled patches, discussed next.

[2]They can be decided through empirical cross-validation.
[3]https://sites.google.com/site/stochasticflowclustering/

3.2.1 Learning the Binary Classifier. In this paper, we use k-NN as the classifier because there are only a few interpretable features and we want model training and prediction to be efficient. In particular, we define the distance function between two patches i and j as follows:

$$D(i,j) = \left\| \bar{F}(i) - \bar{F}(j) \right\|_2 * \log(dist(i,j)) \quad (2)$$

Eq. 2 contains two components. The first component compares the features of the two patches while the second component calculates the Euclidean distance between the two patches. To compute the first component, we average the feature vectors of the two patches respectively and compute the L2-norm of their difference. For the second component, we calculate the geographic centroid of both patches and compute the Euclidean distance between the centroids. To decrease the effect of geographic distance, we take the logarithm of the Euclidean distance[4].

To classify an unlabeled patch, we find the k most similar labeled patches based on the distance function in Eq. 2. The classification of the patch is then decided by a vote conducted using the labels of these k most similar labeled patches. We point out that HUG-FM inherently supports **incremental learning**. If the classification result is not desirable and needs further adjustment, the user will be able to improve the result by adding new markers as supervision. The result will then be updated based on the newly added information. In practice, we find that an expert usually only needs to provide 2 to 6 markers to generate high quality results.

3.3 Flood Mapping

After obtaining segmentations of an urban area before and after a flood, flood mapping can be performed through a correspondence of segmentations across time. We use the satellite image collected before the flood as the reference and compare satellite images during and after the flood with this reference. Areas that are not classified as water before the natural disaster but are classified as water after the disaster are considered flooded areas.

4 THE CROWDSOURCED HUG-FM (CHUG-FM)

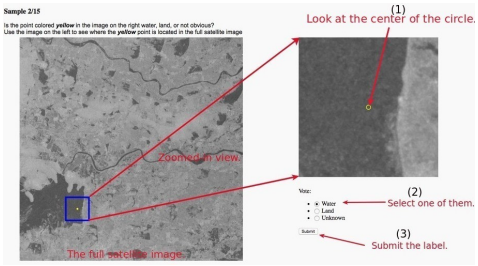
When a disaster strikes, reaction time during and immediately thereafter is of paramount importance. Given both the potential scale of the problem and the number of images one needs to annotate, it may overwhelm available experts. To generate more consistent results in a scalable fashion by leveraging crisis volunteers, we introduce a crowdsourced variant of HUG-FM known as CHUG-FM. Key elements of CHUG-FM include the basic HUG-FM platform and a novel ensembling strategy. The ensembling strategy relies on inference over intermediate results provided by non-expert users.

4.1 Supervision Collection

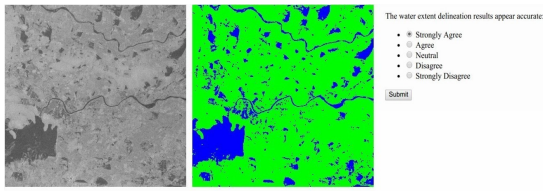
4.1.1 Crowdsourcing Interface. In the crowdsourcing platform, a non-expert user is introduced to a satellite image of an area, such as Houston or Chennai. The user is prompted to label fifteen randomly selected pixels in the image. The interface for pixel labeling is displayed in Figure 1[5]. As shown in the figure, the user is prompted to label the yellow circle in the image; in this case the point is clearly

[4]Logarithm works the best among the alternative functions tested which include: square-root, linear, and quadratic functional variants.
[5]We have elected to keep the interface simple (accommodating both mobile devices and desktops) although our preliminary user study was conducted on desktops.

water. After collecting labels for fifteen markers, the application runs HUG-FM and displays the result to the user, where the user can rate the result (screenshot shown in Figure 1b).



(a) Labeled data provision interface



(b) Flood mapping result based on the provided labels.

Figure 1: CHUG-FM screenshots: a) user labeling interface. The blue square highlights the region around the marker to be labeled. A zoomed-in view of the marker is displayed on the right. The user has three label options: water, land, or unknown. b) the result (segmentation) based on the provided labels is shown to the user and the user can rate the result.

4.1.2 Stratified Generation of Markers. A key difference between HUG-FM and CHUG-FM is that in the former we rely on the human expert to select patches from the image for labeling (see Section 3.2), while in the latter we do not. For the crowdsourced variant, we need to identify key patches that members of the crowd will label since these individuals (i.e non-experts) might not be good at selecting patches (e.g. they may select only land patches or just select patches from a very small region). Pre-selection also offers an efficiency advantage since users do not need to spend time selecting patches. In designing the patch selection procedure for CHUG-FM, we have two priorities in mind. First, both water and land patches should be sampled, regardless of any skew that may exist (e.g. very little water in the image). Second, patches that span the entire image should be sampled (e.g. patches should not only come from on specific area of the image). To ensure these priorities are met, we stratify patches into three groups: likely-water, likely-land, and uncertain. We then sample patches from the three groups separately by adapting the idea of proportional-allocation stratified sampling [7]. Specifically, we convert the image into a gray-scale image and fit a Gaussian Mixture Model (GMM) with two components to the pixel intensity of each patch. We compute the means of the two Gaussian components μ_1 and μ_2 ($\mu_1 < \mu_2$). We then categorize the patches based on their intensity values. If the average pixel intensity of a patch $x \leq \mu_1$, then the patch is regarded as likely-water. If $x > \mu_2$, then it is regarded as likely-land. Otherwise, the patch is categorized as uncertain. We sample 40% of the patches from the likely-land group and likely-water group respectively while drawing the remaining 20% from the uncertain group. If a patch is selected, a marker is placed in its center (and context is provided to ensure cognitive correspondence). Note that sampling uncertain pixels allows us to satisfy our second priority in sampling, which is to cover the entire image in sampling.

We inject two validation markers into the samples for the purpose of eliminating ineffective workers. These two markers are manually selected from the satellite image and are trivial for a human to label as water or land. In total, we generate 2 validation markers and 13 regular markers for each flood mapping task.

4.2 Ensemble Learning

As the intermediate flood mapping results generated by different users can vary greatly in their quality, we use an ensemble learning method known as a voting classifier to aggregate intermediate results into a final result [26]. We remove the participants who do not correctly label all the validation markers. To determine the aggregated label of a patch, we compute the distribution of its labels across all the participants. If the proportion of results labeling it as land exceeds some threshold θ, then the patch is labeled as land; otherwise it is labeled as water. We next discuss the two-fold cross-validation procedure used for selecting θ.

We denote the number of participants as n, each of which provides m markers. At the end of the crowdsourcing experiment, we have n individual HUG-FM results and $n * m$ patches with provided labels. We randomly select $0.5 * n$ users and use their HUG-FM results as the training dataset while treating the $0.5 * n * m$ labeled points from the rest of the users as the validation dataset. We conduct the aggregation on the training dataset, where we vary the threshold θ from 0.0 to 1.0 by 0.01 increments. We then infer the labels of patches in the validation dataset and adopt the user-provided labels as ground-truth for evaluation. We pick the θ that leads to the best performance in the validation dataset[6]. While the selection of θ is governed by the trade-off between precision and recall, we found in experiments that the best performance is generally achieved when θ is between 0.1 and 0.3.

5 EXPERIMENTS ON HUG-FM

In this section, we examine the performance of our framework, HUG-FM, on real-world satellite images to evaluate its performance.

5.1 Experiment Setup

We compare our algorithm with some state-of-the-art algorithms for image segmentation and semi-supervised learning:

(1) Otsu thresholding [23], the most common thresholding-based method.
(2) The Watershed algorithm [4], a semi-supervised region-growing method.
(3) The Normalized cut (N-cut) algorithm [28]. It formulates the image as a graph and uses the normalized cut criterion to segment the image.
(4) Graph-based image segmentation with post-processing. While it also generates patches before segmentation, the subsequent step is unsupervised. It involves continued merging of nearby patches based on the similarity of pairs of patches until the designated number of patches is left.

[6]Here we use F1-score to measure the overall performance.

Image Date	Size of Image	σ_x^2	σ_y^2	d_{max}	# patches
11/24/2015	800×444	3	16	2	12946
10/19/2015	4500×2500	2	16	2	69674
10/31/2015	4500×2500	2	16	2	69674
11/12/2015	4500×2500	2	16	2	69674
11/24/2015	4500×2500	2	16	2	69674
12/06/2015	4500×2500	2	16	2	69674
12/18/2015	4500×2500	2	16	2	69674

Table 1: Datasets and parameter settings.

(5) Support Vector Machine (SVM). This method learns a classifier using the provided markers as training data.
(6) NORM-THR[18, 21]: is a modern split-based automatic thresholding method for water delineation.
(7) Planetoid [32]. This is the state-of-the-art semi-supervised learning algorithm on attributed graphs based on deep neural networks. To apply this method, we construct a graph with node attributes in the same way as `HUG-FM`.

We implement `HUG-FM` and the graph-based method with post-processing using `Python`. We use the OpenCV API for implementing the Watershed algorithm and Otsu's thresholding. For the N-cut algorithm and Planetoid, we employ the source code from the authors. We use SVM-light for the SVM method[7].

5.2 Qualitative Experiment on Chennai Dataset

5.2.1 Water Delineation on Individual Images. For qualitative evaluation, we use satellite images of Chennai, India during the 2015 South Indian Floods[8]. In total we run `HUG-FM` on six satellite images during the flood, focused on the greater Chennai metropolitan area, one for every twelve days. The images come from Sentinel-1, which orbits this region every twelve days. Considering the fact that some baselines (e.g. the N-cuts algorithm and the Watershed algorithm) are very computationally expensive and cannot finish on large satellite images in 24 hours, we downscale the satellite images and run our method and baselines on them for comparison. As an example, we run all the algorithms on the satellite image of Chennai on 11/24/2015, which is re-sized from 4,500×2,500 to 800×444. Basic information about the datasets and parameter settings for our algorithm are displayed in Table 1.

We compare the performance of our algorithm with the baselines on the downscaled image shown in Figure 2a (Chennai area on 11/24/2015). The water-land segmentation results are shown in Figure 2b-f while the execution time is listed in Table 2. The results of NORM-THR, Planetoid and SVM on the Chennai dataset, are omitted here due to lack of space (they perform slightly worse than `HUG-FM`), but a detailed comparison with these approaches will follow in the next section. We highlight the following observations on the Chennai dataset as shown in Figure 2:

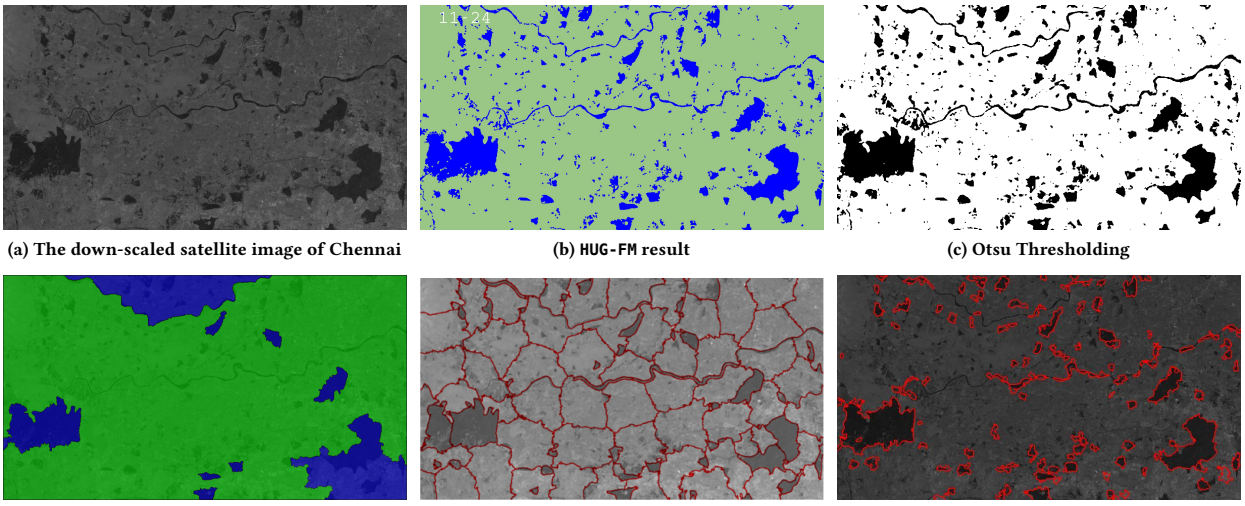
(1) Our method performs the best among all the approaches. Our method can clearly identify most of the water areas and even long thin rivers; most other methods fail to do so. Particularly, our method is good at identifying regions of arbitrary shape while not limiting the size of each segment. We notice that Otsu thresholding in Figure 2c also has similar advantages, but it is very sensitive to noise and tends to generate many tiny partitions (two times as many segments as our method on the image). This is because its segmentation results only depend on the intensity of each pixel and one pixel can be an individual partition if its pixel intensity is far different from its neighboring pixels. Its shortcomings will become more evident shortly when we discuss our quantitative evaluation.
(2) Compared to the Watershed algorithm, our method produces better results while requiring far less human effort (Figure 2b vs. Figure 2d). The Watershed algorithm seems to correctly capture some boundaries but can not segment out small water areas, including the long thin rivers. For the result shown in Figure 2d, the expert user places nine markers in different water areas and two markers in land areas (see Figure 3b). But the segmentation result is still not desirable. On the other hand, using our method, the user only needs to place one marker in water and one in land respectively (see Figure 3a) and the result is much better than the Watershed algorithm. One of the reasons for this difference is that the Watershed Algorithm is a region-growing method and the segments grow from the markers in a local fashion; therefore it requires more manually placed labels to achieve desirable performance.
(3) The N-cuts algorithm tends to generate over-balanced segments and cannot extract segments of long thin shape (shown in Figure 2e). Though it performs well in detecting most of the boundaries, it breaks large areas into pieces that should be in one partition. This can be seen from the split of some large lakes. As a whole, the result is much worse than our algorithm.
(4) Graph-based segmentation with post-processing works well in detecting some large regions (see Figure 2f). However, similar to the N-cut algorithm, it cannot detect small water regions, especially long thin rivers.
(5) As displayed in Table 2, our method is the most efficient method in classifying the image immediately after providing labels. The N-cut algorithm is very slow because it involves expensive computation of the eigenvectors for the Laplacian matrix. The Graph-based segmentation with post-processing method is computationally expensive at the stage of hierarchical merging. Our method is even faster than the simple Otsu's thresholding algorithm since we label the image patch by patch, as opposed to pixel by pixel. We point out that our algorithm requires about 30 seconds for preprocessing to generate patches. However, the preprocessing time is less of a concern here as we only need to conduct it once. The labeling time is a more important factor and `HUG-FM` greatly reduces this amount of time, enabling usage in different scenarios (e.g. online interactive learning and crowdsourcing).

Method	# Markers	Time (s)
`HUG-FM`	2	0.057
Otsu's Thresholding	0	0.077
Watershed Algorithm	11	0.225
N-cuts Algorithm	0	538.615
Graph method w. post-process	0	558.220

Table 2: Running time comparisons for different methods. # markers is the number of markers the human provides for the algorithm.

5.2.2 Dynamic Analysis for Flood Mapping. While we mainly focus on water delineation in the satellite images above, we now

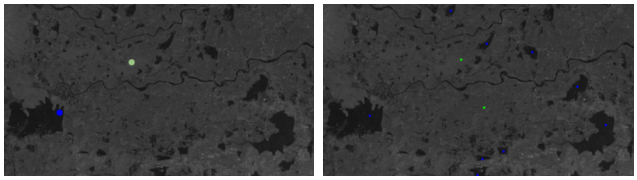
[7] http://svmlight.joachims.org/
[8] https://en.wikipedia.org/wiki/2015_South_Indian_floods



(a) The down-scaled satellite image of Chennai (b) HUG-FM result (c) Otsu Thresholding

(d) Watershed Algorithm (e) N-cuts Algorithm (100 partitions) (f) Graph-based Clustering with Post-processing

Figure 2: Segmentation results of different approaches on satellite image of Chennai on 11/24/2015 (down-scaled).



(a) Two markers provided by the user to our method (b) Eleven markers provided to the Watershed algorithm

Figure 3: Labels that the user provided for the algorithm. Blue points label the areas as water while green points label them as land.

discuss how we adopt the developed water delineation method to detect flooded areas. To this end, we conduct water delineation on all the full-size satellite images of Chennai that are described in Table 1. We then refer to the historical satellite images collected before the flood and conduct dynamic analysis.

Specifically, we use the segmentation results from 10/31/2015 as the baseline and compare this water delineation to the ones from later dates. Figure 4 presents the dynamic changes of water areas[9]. Red color indicates the areas that change from land to water while yellow color indicates the opposite change. From Figure 4, we can clearly observe that 11/24 and 12/06 have the largest number of water areas; water areas seem to decrease following 12/06. Red areas are likely regions affected by the flood. The flood maps are quite consistent with the fact that the South Indian floods lasted from 11/08/2015 to 12/14/2015.

5.3 Quantitative Evaluation

In this section, we quantitatively evaluate HUG-FM on a real-world dataset with ground-truth. For this purpose we rely on a higher-resolution synthetic aperture radar satellite image of Houston collected immediately following a 2016 flood; this dataset has been manually annotated by a domain expert (this annotation is the ground truth). The size of the image is $1,550 \times 2,533$. Similar to the Chennai datasets, there are two raw attributes (HH and HV) from radar and one attribute representing geographic elevation for each pixel. HH and HV measure the polarity of waves reflected by a material and are helpful in distinguishing water from land.

Method	Accuracy	F1 Score
HUG-FM	**0.9552**	**0.8681**
SVM	0.9451	0.8382
Planetoid	0.9445	0.8414
Watershed algorithm	0.8904	0.6796
Otsu's thresholding	0.8977	0.7394
NORM-THR	0.8673	0.8371

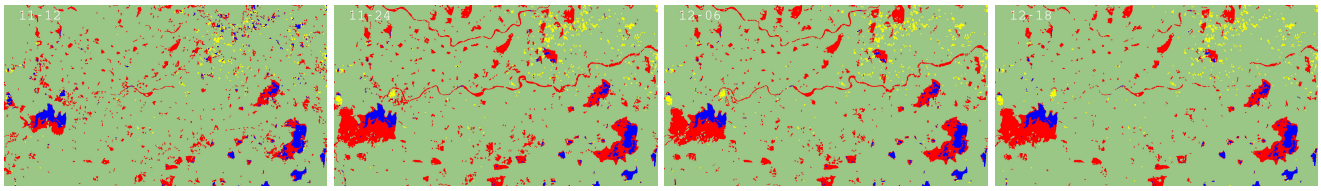
Table 3: Quantitative evaluation on Houston dataset.

We run HUG-FM on this dataset with two representative markers provided by an expert user for water and land area. As baselines, we compare our method with some of the best methods in the previous section[10]. We employ the Watershed algorithm with six carefully selected markers. For SVM and Planetoid, we provide 50 labeled patches for both water and land as they require more labeled data to generate reasonable results (label data is sampled from the ground-truth). We leverage the ground-truth provided by the domain expert and evaluate different methods using accuracy and F_1 score. Table 3 shows the performance of these methods. It can be seen that HUG-FM substantially outperforms other methods with the highest values on both evaluation metrics. Both Otsu's algorithm and the Watershed algorithm have very low precision and therefore low F_1 score, while NORM-THR has more balanced performance. Among all the baselines, NORM-THR, SVM and Planetoid are the strongest. We point out that even though SVM and Planetoid use much more labeled data (100 training examples as opposed to 2 in HUG-FM), our method still outperforms them.

We also visualize the results of these methods in Figure 5. Our method nicely captures most of the water areas of various shapes. Otsu's algorithm, on the other hand, tends to mistakenly classify

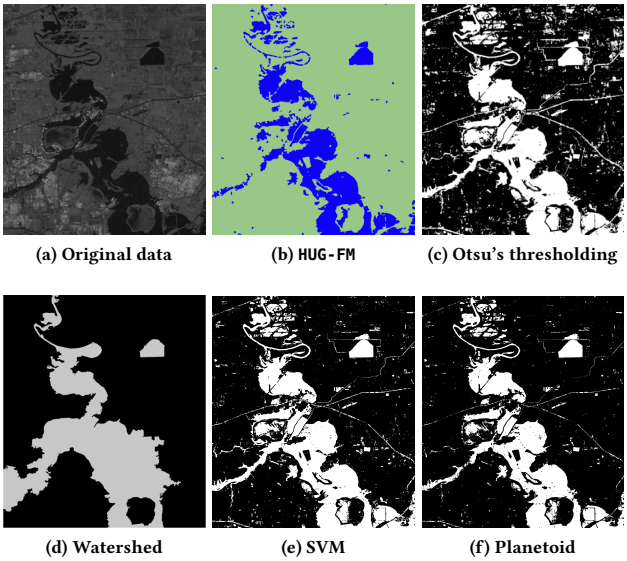
[9]Dynamic images can be seen at http://jiongqianliang.com/HUGFM/

[10]The N-cut algorithm and the graph-based method with post-processing algorithm cannot finish running in 24 hours and hence are not reported here.



(a) 11/12 (b) 11/24 (c) 12/06 (d) 12/18

Figure 4: Water area changes from 11/12/2015 to 12/18/2015 using 10/31/2015 as the baseline. One image for every 12 days. Red color indicates areas that were land on 10/30/2015 but were water on the given date, while yellow color indicates areas that were water on 10/30/2015 but were land on the given date. Blue and green represent areas that were originally water or land on 10/30/2015 and remain so on the given date.



(a) Original data (b) HUG-FM (c) Otsu's thresholding

(d) Watershed (e) SVM (f) Planetoid

Figure 5: Flood mapping results on Houston dataset (truncated).

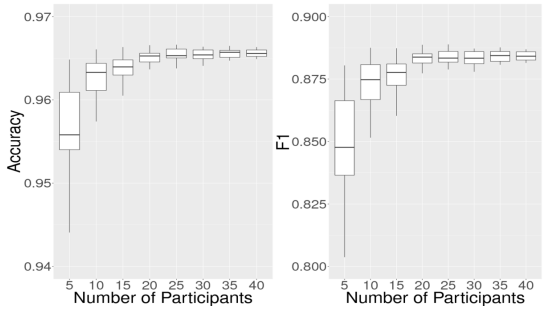
Method	Accuracy	F1 Score
HUG-FM (by the expert)	0.9552	0.8681
Avg. HUG-FM (by the crowd)	0.9380 (±0.061)	0.8290 (±0.094)
CHUG-FM	**0.9655**	**0.8840**

Table 4: Results of CHUG-FM compared with others on Houston.

many regions as water and the result looks very messy. The Watershed algorithm generates an overly smooth result and cannot accurately capture small regions of water. The SVM and Planetoid results look visually similar to HUG-FM (as does NORM-THR, whose image is not shown due to limit of space). However, all three incorrectly classify many land regions as water.

5.4 Crowdsourcing Experiment Setup

To evaluate our crowdsourced cognitive framework, CHUG-FM, we recruited a broad mix of college-educated participants (53) in our user study (OSU-IRB # 2015B0249) which follows standard Nielsen Norman Group guidelines. Participants were given time to acclimatize themselves with the interface and further had the opportunity to work with some test images that were distinct from any of the satellite images used in the experiment. Each participant was provided three different satellite images to work with and a 15-minute time limit. The first image is the Houston image during the 2016



(a) Accuracy (b) F_1 score

Figure 6: Box plots showing the differential performance of CHUG-FM while varying number of participants. The x-axis is the number of the participants and y-axis represents the evaluation measurements of accuracy and F_1 respectively.

Measurements	Median	Average	S.D.
Time Spent (minutes)	5.78	6.08	1.85
Rating Easiness of using the website	5.00	4.68	0.52
Rating Clearness of the instructions	5.00	4.73	0.50

Table 5: Summary of users' feedback on the crowdsourcing experiment. The rating scale is from 1 to 5 with 5 being the highest.

Flood as discussed in the previous section. The other two images are for the city of Lumberton in North Carolina before and after Hurricane Matthew respectively (both dimensions are 2031×2500). Before the experiment step-by-step instructions are shown to the participants on the front page of the website. At the end of each task, the participants are asked to rate the results of HUG-FM and provide feedback for our crowdsourcing platform.

5.5 CrowdSourced Results Analysis

5.5.1 Ensemble Learning Results. Once the participants finish the experiment, we conduct ensemble learning following the procedures aforementioned. We remove 5 ineffective participants from the database as they fail to correctly label both the validation markers. We start by discussing the results on the Houston data which includes ground-truth labels for all the pixels. The optimal threshold θ for this dataset, tuned through two-fold cross-validation, is 0.21. We use this threshold to perform aggregation on all the HUG-FM results from these participants. We compare the aggregated results with the ones from each individual and the domain expert in Table 4. We can observe that CHUG-FM performs significantly better than each individual HUG-FM (row 2 vs. row 3 in Table 4). Even compared with HUG-FM conducted by a domain expert with carefully selected markers, CHUG-FM still brings us 0.0385 improvement in

terms of F_1 score. This reveals the power of crowdsourcing in conducting flood mapping. Though each individual does not perform very well in mapping the flood, we can obtain high-quality results by aggregation as long as errors do not correlate across users.

We also analyze how the performance of CHUG-FM changes by varying the number of participants. We vary the number of participants from 5 to 40. For each particular number of participants c, we sample c from the total 48 valid participants and run the same aggregation as above, using 0.21 as the θ. We repeat the same process 60 times and compute the average performance and the corresponding standard deviations for each c. We show the results in the box plots in Figure 6. We can observe from Figure 6 that as the number of participants increases, the performance of CHUG-FM improves. Expectedly, the variance of the performance is lower when there are more participants.

We also visualize the results to conduct qualitative evaluation for all three images, including the two satellite images from before and after Hurricane Matthew which do not contain ground-truth label (images omitted due to limit of space). We observe that CHUG-FM can capture the water areas more accurately than HUG-FM. It generates a more smooth result and is less prone to noise.

5.5.2 User Experience Analysis. We conduct a comprehensive analysis on the user experience for the crowdsourcing platform. We summarize the results in Table 5. We can see that it only took 6.08 minutes on average (with maximum of 11.33 minutes and minimum of 3 minutes) to finish the three flood mapping tasks. According to the feedback, the participants appreciate the usability and cognitive correspondence of our crowdsourcing platform. More than 75% of participants gave the highest possible score on rating the easiness of using the website and the clearness of the instructions. 14 participants submitted detailed comments by using the textbox on the website and 5 of them mentioned our crowdsourcing platform is "easy"/"straightforward" to use. Here are a few representative positive comments from the participants:

(1) User 1: "Very interesting exercise. Very easy to use and follow..."
(2) User 2: "Pretty neat idea..."
(3) User 3: "I think it was cool that ... I really enjoyed this."

Some participants indicated that some markers lies between water and land and are difficult to label. This is expected as we intentionally generate a few markers in uncertain patches to ensure we cover the space of the entire image in sampling. In the future, we plan to improve the user interface so that it can be easier for participants to label these markers (e.g. supporting interactive zoom-in view).

6 CONCLUSION

In this paper, we describe a novel cognitive framework to the flood mapping problem by effectively coupling human guidance with machine learned models. Our results with guidance from experts show that our algorithm can correctly segment out water and land areas with less error compared to state-of-the-art approaches. To enable usage during disaster, we develop a novel crowdsourcing platform and ensemble algorithm to utilize the wisdom of the crowd (crisis volunteers). Our crowdsourcing experiment, with over fifty participants working on three different tasks, consistently shows that the crowdsourced variant performs well – producing noise-tolerate flood maps comparable to those produced by domain experts.

Acknowledgments: This work is supported by National Science Foundation under grants EAR-1520870 and DMS-1418265. Computational support was provided by Ohio Supercomputer Center under grant PAS0166. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their sponsors. We thank Jiayong Liang and Desheng Liu for useful discussions.

REFERENCES

[1] S. S. Al-Amri, N. V. Kalyankar, et al. Image segmentation by using threshold techniques. *Journal of Computing*, 2010.
[2] M. Baatz and A. Schäpe. Multiresolution segmentation: an optimization approach for high quality multi-scale image segmentation. *Angewandte Geographische Informationsverarbeitung XII*, 58:12–23, 2000.
[3] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *JMLR'06*.
[4] S. Beucher and F. Meyer. The morphological approach to segmentation: the watershed transformation. *Mathematical Morphology in Image Processing*, 1992.
[5] A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. 2001.
[6] A. Browet and *et al.* Community detection for hierarchical image segmentation. In *International Workshop on Combinatorial Image Analysis*, 2011.
[7] W. G. Cochran. *Sampling techniques*. John Wiley & Sons, 2007.
[8] T. Cour, F. Benezit, and J. Shi. Spectral segmentation with multiscale graph decomposition. In *CVPR*, volume 2, pages 1124–1131. IEEE, 2005.
[9] L. C. Degrossi et al. Flood citizen observatory: a crowdsourcing-based approach for flood risk management in brazil. In *SEKE*, 2014.
[10] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.
[11] H. Gao, G. Barbier, and R. Goolsby. Harnessing the crowdsourcing power of social media for disaster relief. *IEEE Intelligent Systems*, 26(3):10–14, 2011.
[12] L. Giustarini et al. Probabilistic flood mapping using synthetic aperture radar data. *TGRS*, 2016.
[13] M. F. Goodchild and J. A. Glennon. Crowdsourcing geographic information for disaster response: a research frontier. *IJDE*, 2010.
[14] S. Hallegatte, C. Green, R. J. Nicholls, and J. Corfee-Morlot. Future flood losses in major coastal cities. *Nature climate change*, 2013.
[15] J. Kelly. Computing, cognition and the future of knowing. In *Whitepaper*. IBM Research, 2015.
[16] G. A. Lazarova. Semi-supervised image segmentation. In *Artificial Intelligence: Methodology, Systems, and Applications*, pages 59–68. Springer, 2014.
[17] J. Liang, P. Jacobs, J. Sun, and S. Parthasarathy. Seano: semi-supervised embedding in attributed networks with outliers. *arXiv preprint arXiv:1703.08100*, 2017.
[18] J. Liang and D. Liu. Image fusion for flood detection based on surface change probability (iffd): an example of modis and landsat. *American Association of Geographers, AAG Annual Meeting*, 2017.
[19] J.-M. Martinez and T. Le Toan. Mapping of flood dynamics and spatial distribution of vegetation in the amazon floodplain using multitemporal sar data. *Remote sensing of Environment*, 2007.
[20] M. Martinez, D. Williams, and D. Simon. Crowdsourcing volunteers comb satellite photos for malaysia airlines jet. *Cable News Network*, 12, 2014.
[21] S. Martinis, A. Twele, and S. Voigt. Towards operational near real-time flood detection using a split-based automatic thresholding procedure on high resolution terrasar-x data. *Natural Hazards and Earth System Sciences*, 9(2):303–314, 2009.
[22] K. Nigam and *et al.* Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2-3):103–134, 2000.
[23] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, Jan 1979.
[24] Y. Ruan, D. Fuhry, J. Liang, Y. Wang, and S. Parthasarathy. Community discovery: Simple and scalable approaches. In *User Community Discovery*. 2015.
[25] V. Satuluri and S. Parthasarathy. Scalable graph clustering using stochastic flows: applications to community discovery. In *SIGKDD*. ACM, 2009.
[26] R. E. Schapire et al. Boosting the margin: A new explanation for the effectiveness of voting methods. *The annals of statistics*, 26(5):1651–1686, 1998.
[27] S. B. Serpico et al. Information extraction from remote sensing images for flood monitoring and damage evaluation. *Proceedings of the IEEE*, 2012.
[28] J. Shi and J. Malik. Normalized cuts and image segmentation. *PAMI*, 2000.
[29] S. P. Simonovic and P. Eng. Role of remote sensing in disaster management. 2002.
[30] D.-M. Tsai. A fast thresholding selection procedure for multimodal and unimodal histograms. *Pattern Recognition Letters*, 16(6):653–666, 1995.
[31] A. Twele and *et al.* Sentinel-1-based flood mapping: a fully automated processing chain. *International Journal of Remote Sensing*, 2016.
[32] Z. Yang, W. W. Cohen, and R. Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. *ICML*, 2016.
[33] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *ACL'95*.