https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

Natural Hazards and Earth System Sciences
Discussions

Converging Human Intelligence with AI Systems to Advance Flood Evacuation Decision Making

Rishav Karanjit[1,2], Vidya Samadi[3], Amanda Hughes[4], Pamela Murray-Tuite[5], Keri Stephens[6],
1. School of Computing, Clemson University, SC, USA.
2. Amazon Web Service, Seattle, OR, USA.
3. Department of Agricultural Sciences, Clemson University, SC, USA.
4. Computer Science Department, Brigham Young University, UT, USA
5. The Glenn Department of Civil Engineering, Clemson University, SC, USA.
6. Moody College of Communication, The University of Texas at Austin, TX, USA

Corresponding author: samadi@clemson.edu

Abstract. The powers that artificial intelligence (AI) has developed are impressive, with recent success in leveraging human expertise at various stages of model development. AI can attain its full potential only if, as part of its intelligence, it also actively teams with humans to co-create solutions. Combining AI simulation with human intelligence through data convergence can improve decision-making processes and provide a capacity akin to a "teaming intelligence." This research, for the first time, introduces the concepts of Human-AI Convergence (HAC) capabilities for flood evacuation decision-making. The objective of this study was to develop a unique, computationally effective surrogate HAC system for flood evacuation decision-making that integrates the distinctive features of AI with transportation geospatial data, a river hydraulic model, and human data from X (previously Twitter) to visualize flood inundation areas and suggest re-routing. The HAC system is smartly designed to forecast flood stage levels using AI across the US Geological Survey gauging stations and combine the results with Manning's equation results and transportation data, integrated into a web-based Google Earth visualization architecture. The technology has been tested in the Lowcountry of South Carolina, where previous flooding disasters caused considerable damage to the transportation networks and increased traffic on evacuation routes. This state-of-the-art HAC system— a flood evacuation product— stands to advance the frontier of human-AI collaborative research in the context of real-time flood emergency management and response.

Keywords: Artificial Intelligence; Human-AI Convergence; Flood Emergency Management; Evacuation Decision Making and Planning.

1. Introduction

Evacuation is crucial for minimizing the risk of injury or loss of life during flooding events. However, the decision to evacuate can be complex, involving multiple factors including social considerations, resource availability, isolation of location, and capacity of the infrastructure (Kolen et al., 2013). The costs of an evacuation in the case of hurricanes in the United States can exceed 1 million dollars per mile due to losses in commerce, productivity, and direct losses to goods (Wolshon et al., 2005). To reduce this cost, deterministic models such as a heuristically driven flood evacuation planning model (Bennett et al., 2017) and stochastic models such as a Fuzzy logic-based decision support system (Jia et al., 2016) have been developed to aid decision-makers in planning and preparing for the flood evacuation processes. However, when a flood disaster occurs, analyzing complex information and data to make quick evacuation decisions

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

is a challenging task for decision-makers and authorities. Therefore, providing more rapid and accurate evacuation modeling is essential for a safe and smart emergency response and decision-making.

Machine learning approaches are increasingly becoming a viable solution for flood evacuation decisions. Scholars have recently developed machine learning models to forecast flooding and determine safe evacuation routes during emergencies (Sreejith et al., 2022; Wang et al., 2023). The results of these studies are important for rapid evacuation decision-making, however, a mechanism to incorporate decision-makers knowledge and data into machine-learning approaches is lacking. The approach of uniting data from humans and machine learning leverages the strengths of humans and machine learning systems, resulting in more efficient and effective flood evacuation decisions. Indeed, combining machine learning simulation with human understanding and strategic abilities through data convergence may optimize the flood evacuation process and provide a capacity akin to a "teaming intelligence." In this human-AI convergence (HAC) system, humans can perform tasks such as search and rescue, communication, and flood damage validation, which require human knowledge and social skills while machine learning can perform flood forecasting and analyze massive real-time data and information. This cooperation is driven by a shared objective, which necessitates exchanging crucial information through diverse forms of communication, prediction, and the achievement of high-level coordination tasks (McNeese, et al. ,2018).

Previous studies on the concepts similar to HAC have focused on the interaction and effectiveness of human teams with AI working in robotic swarms (Seeber, et al., 2020) with landed aircraft perimeter security (Madni & Madni, 2018) in collaborative games (Ong, et al., 2012), to identify risky human behavior (Stephens et al., 2023), and how various factors such as a person's understanding of the limits or mistakes of a machine learning system might affect team performance in a HAC implementation (Bansal, et al., 2019; Liang et al., 2019;). These studies have led to increased growth in HAC literature, where humans and AI data meet at a point to work together in collaboration and carry out complex tasks as an integrated unit. However, HAC has never been applied for flood response and evacuation problems, and this area could benefit from creative solutions.

The goal of this study is to address this knowledge gap by synthesizing and analyzing HAC competence in flood evacuation decisions and harnessing the potential of machine learning as a partner in real-time decision-making. This research examines the step-by-step structure of employing a HAC system for flood evacuation planning in South Carolina, USA. The intention is not to include all possible algorithms, applications, and techniques, but rather to provide case study applications where HAC system has been successfully implemented. As part of this study, an HAC system was developed for flood evacuation decision-making to provide a general structure for researchers to use HAC concepts to devise effective systems that cooperate well. Additionally, the project evaluates the state-of-the-art in this area, and, in doing so, provides a research agenda and a roadmap for future HAC studies. Our developed HAC system combines machine learning models with human data to predict flood depth and inundation areas and then use these forecasts to determine flood evacuation rerouting decisions. The system includes machine learning approaches for forecasting floods across the US Geological Survey (USGS) gauging stations and incorporates the results into a Height Above Nearest Drainage (HAND) model to calibrate inundation areas in real-time. In addition, we leveraged human data into the HAC system by integrating X (previously

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

Twitter) data into the system. In this integrated HAC framework, the fusion of flood level predictions, a river hydraulic model, transportation data, and real-time X observations heralds an innovative paradigm in flood evacuation prediction and response strategies.

The remainder of this paper is structured as follows: Section 2 provides an introduction to the machine learning models, human data, river hydraulic model, rerouting approach, HAC workflow, and performance metrics. Section 3 presents the results of the HAC applications. Finally, Section 4 presents the discussion, conclusions, and paths for future research.

2. Methodology

2.1. Study Area and Data

Our HAC system was developed and tested for the Lowcountry in South Carolina (SC), USA where frequent flooding caused significant damage to critical infrastructure, properties, and people's lives. The Lowcountry is characterized by a low elevation, flat terrain area prone to inundation conditions and storm surge. Following recent major flooding such as the SC Flood of 2015 and Hurricane Matthew in 2016, many local roads in the Lowcountry were under water, which limited mobility, hampered evacuation response, and in some places isolated communities. Roads in this region were not built high enough to accommodate water flowing around and under them (Phillips, 2020). Consequently, managing flood damage and facilitating evacuations pose major challenges in this region.
We tested the HAC system for multiple USGS gauging stations in the Lowcountry, as case studies (see Figure 1). Rainfall and river data were collected from the USGS and the National Weather Service (NWS). We trained the machine learning models for three USGS gauging stations located in the Lowcountry, including Turkey Creek (USGS02172035), South Fork Edisto River (USGS02173000), and North Fork Edisto River (USGS02173500), shown in Figure 1.

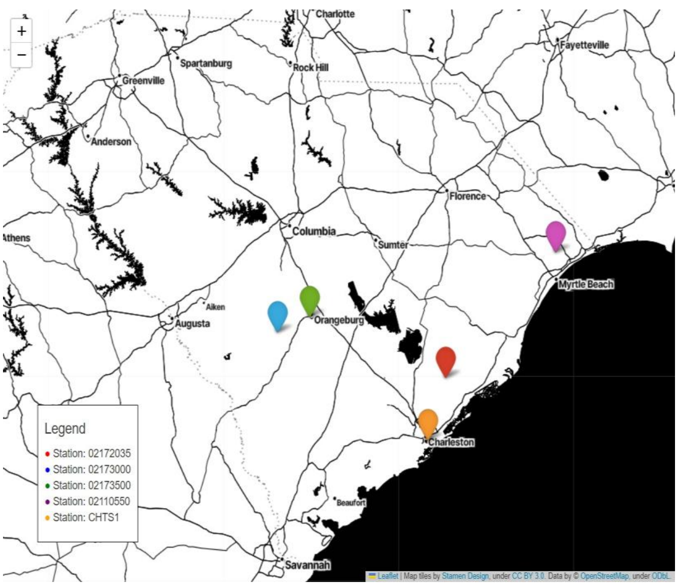

Figure 1: The USGS gauging stations in the Lowcountry, SC used in this research.

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

Historical time series data of precipitation and gauge height obtained from the USGS were used to train machine learning algorithms. During no-flood events, the gauge height of the river was slow-changing. Conversely, gauge height values changed significantly during flooding events over short intervals of 15 minutes. Since the flood prediction task was defined on an hourly basis, we used a pandas (McKinney, 2010) library to calculate the cumulative daily data. We used machine learning algorithms, NWS rainfall forecast data, and a Rational method along with a rating curve conversion tool (explained in the next sections) to predict flood level (or gauge height of the river) in ungauged/poorly gauged watersheds in the Lowcountry, SC.

2.2. Machine Learning Algorithms

We trained two types of RNN models, i.e., Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU; Cho, 2014) using three USGS gauging stations. For each station, we used Optuna (Akiba et al., 2019) to optimize the hyperparameters. We trained 30 models for each station (overall 180 models [2 models, 3 stations and 30 models each]) and used the best model for real-time flood forecasting. During training, we used the pruning technique in Optuna as an early stopping technique. Pruning within the Optuna hyperparameter optimization library stops RNN training early if it is deemed unlikely to produce a better result than the previous best-known model configuration (Akiba et al., 2019). The pruning technique enables user to stop model training efficiently if it is deemed unlikely to produce better results without sacrificing the quality of results. From the 180 models produced during training, the six best models were selected from the Optuna library (for three gauging stations and two models), and then the three best models were chosen manually based on performance metrics.

The architecture of LSTM and GRU variants were specifically modified to address the issue of vanishing gradients that are commonly encountered in conventional recurrent networks. LSTM is equipped with a specialized memory cell capable of retaining information for extended periods. Additionally, this network features three distinct types of gates - namely, the input gate, forget gate, and output gate - which regulate the inflow and outflow of information to and from the memory cell. The gates incorporated in the network facilitate the selective retention or omission of information, rendering it highly appropriate for applications that entail the manipulation and retention of sequential data, such as Natural Language Processing (NLP), speech recognition, and time series prediction. The LSTM network receives input data as a sequential vector set, with each individual LSTM unit processing a single vector at each time step. The output of each LSTM unit is a hidden state vector that is subsequently utilized as input for the following time step. The LSTM model can effectively model intricate sequential data by using gates to regulate the flow of information within the network. This enables the model to retain information from past inputs and leverage it to make informed predictions about future inputs.

Our study employed an LSTM consisting of six layers, a dropout value, and a dense layer. The first three hidden layers were followed by a dropout layer, which was then followed by the remaining LSTM layers. This was succeeded by a flattened layer and a dense layer containing five neurons. The spatial dimensions of the input are reduced to the size of the channel by a flattened layer. The LSTM layer is designed to predict the subsequent 5 data points (5 hours in advance) by utilizing the preceding 48 data points (48 hours)

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

Natural Hazards and Earth System Sciences
Discussions

as "look back" time. The dropout rate, number of units for each layer and epoch number were decided after training 240 models using Optuna. This is because the quantity of water flowing into and out of a river system affects the height of a gauge. The river system receives runoff from precipitation, whereas the runoff output from the system is represented by discharge or gauge height. Briefly, the gauge height of the river was predicted using LSTM and GRU models, the best model was then selected, which was the LSTM, for forecasting gauge height in real-time using NWS rainfall forecast data. Figure 2 illustrates the flood forecasting workflow using LSTM.

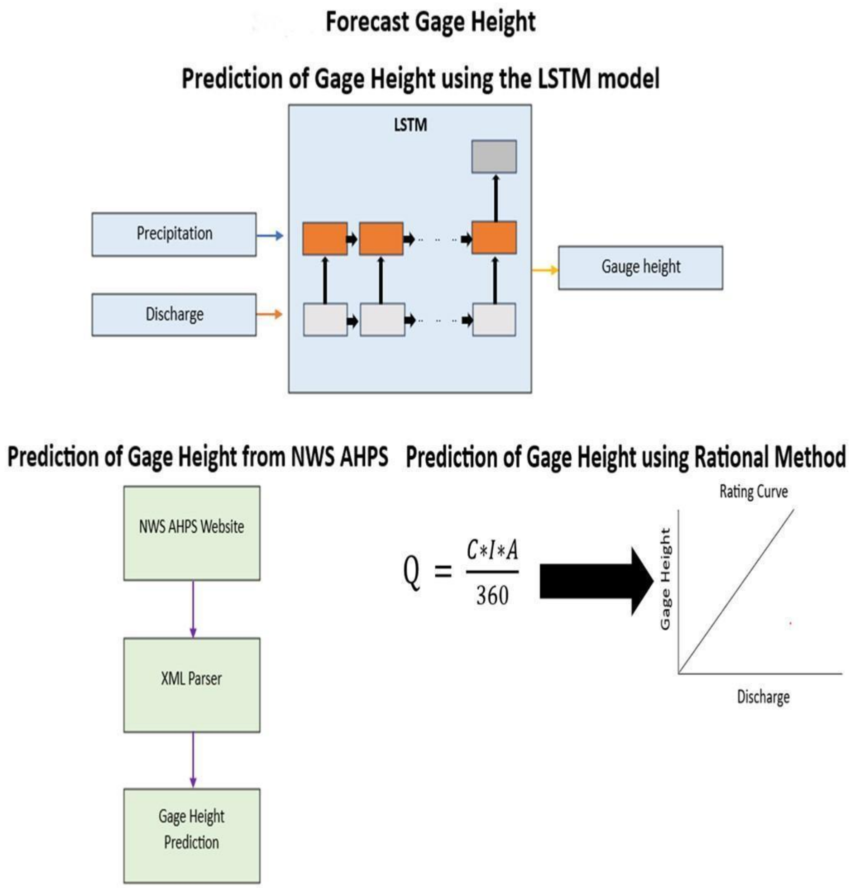

Figure 2: Flood forecasting workflow using LSTM as the best model.

2.2.1 Performance Metrics Used for Machine Learning Modeling Evaluation

Several performance measures are utilized in this research to assess LSTM and GRU performance. They are Mean Square Error (MSE), Mean Absolute Error (MAE), Mean absolute scaled error (MASE), the Nash–Sutcliffe model efficiency coefficient (NSE), and Huber Loss.

MSE (Equation 1) is the average square of the difference between the model's predicted data and the actual data throughout the whole dataset.

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

$$MSE = \frac{1}{n}\sum_{i=1}^{n} \left(Y_i - \hat{Y}_i\right)^2$$ Equation (1)

Where:
$\hat{Y}_i$ is predicted gauge height. Y_i is observed gauge height. *n* is the length of the dataset.

MAE (Equation 2) is the average magnitude of the difference between the model's predicted and observed flood gauge height data for a collection of predictions and observations as a measure of the magnitude of errors for the entire dataset.

$$MAE = \frac{\sum_{i=1}^{n} |Y_i - \hat{Y}_i|}{n}$$ Equation (2)

Where:
$\hat{Y}_i$ is predicted gauge height. Y_i denotes observed gauge height. *n* represents the length of the dataset.

MASE (Equation 3) is an alternative to metrics like MAE to provide a more interpretable scale. To calculate MASE, we divide the MAE of the forecasting method with the MAE obtained when using the previous observation as the forecast for the next observation.

$$MASE = \frac{MAE_{forecast}}{MAE_{naive}}$$ Equation (3)

Where:
$MAE_{forecast}$ is MAE of the forecast method. MAE_{naive} represents MAE obtained when using the previous observation as the forecast for the next observation.

The Huber loss (Equation 4) is a robust loss function used in regression problems. It combines the properties of the MAE and the MSE. The Huber loss is quadratic for small error values (similar to MSE) and linear for large error values (similar to MAE), making it less sensitive to outliers than the MSE. The ideal value of huber loss is zero; closer the value to zero, better the model performance.

$$L_\delta\left(y, f(x)\right) = \begin{cases} \frac{1}{2}\left(y - f(x)\right)^2 & if\ y - f(x) \leq \delta \\ |\ y - f(x)\ | - \frac{1}{2}\delta^2 & otherwise \end{cases}$$ Equation (4)

Where:
y is the observed gauge height. *f(x)* denotes the predicted gauge height. *δ* represents a threshold value.

2.3. Rational Method

The Rational method (Equation 6) is a deterministic hydrological approach commonly used for estimating peak flow rate or discharge in an ungauged watershed. It is based on the principle that the peak flow rate is directly proportional to the rainfall intensity, the area of the catchment, and a runoff coefficient that considers the characteristics of the land use and soil type in the area. This method uses a simple mathematical equation to estimate flood peak discharge (*Q*) based on three inputs: the rainfall intensity (*I*), the drainage area (*A*), and the runoff coefficient (*C*). The equation is given as:

$$Q = \frac{C * I * A}{360}$$ Equation (5)

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

The Equation 5 involves: The measurement of I in inches per hour (in/hr), the expression of A in acres, and the utilization of C as a dimensionless factor.

To obtain the value of Q in cubic feet per second (cfs), it is necessary to divide the product of C, I, and A by 360. The product of the values of C, I, and A is first divided by 12 to convert the unit of measurement from inches to feet and subsequently divided by 60 to convert the unit of measurement from hours to minutes. This calculation yields a factor of 1/720. This value is subsequently multiplied by 3600, which serves to convert minutes to seconds, resulting in a factor of 1/360. The computation of Q in cfs is obtained through division by 360.

The initial step in computing the runoff coefficient involves obtaining the land use data for a specific latitude and longitude through the utilization of an application programming interface (API). A coefficient is assigned to each type of land use. The OpenWeatherMap API is utilized to obtain data on rainfall intensity. The drainage area is obtained by utilizing a digital elevation model (DEM) specific to the low-country region of SC. Initially, the metadata of DEM was extracted, encompassing details such as the pixel dimensions and transformation data. Subsequently, the provided latitude and longitude values were transformed into pixel coordinates utilizing the available transformation data. A threshold value was subsequently employed on the DEM to generate a binary mask that denotes the watershed region. The predicted gauge height is considered as a threshold value for HAND. Finally, the computation of the watershed's drainage area involves the summation of the mask, which is then multiplied by the pixel area. The calculated drainage area is expressed in units of square meters and converted to acres through division by 4047. By utilizing these three variables, it is possible to derive the maximum flood peak rate in a catchment.

A Rating curve approach is then employed to convert the maximum flood peak rate obtained from the Rational method's equation into gauge height. The Rating curve is established by the USGS as an empirical correlation linking the stage of a river to its stream discharge. The rating curve represents the correlation between the height of a measuring instrument and the volume of water flowing in a stream. The Rational method is employed in cases where there is insufficient data to facilitate flood gauge height prediction.

2.4. HAND Model

We developed the HAND model as an inundation mapping approach (Nobre, et al., 2011) in Python to depict the potential extent of flooding. The HAND model is a terrain analysis technique that estimates the elevation of a point above the nearest stream or river. The model is extensively employed for the purpose of ascertaining the flood risk, drainage patterns, and erosion potential of a given region. The HAND model is founded on the principle of surface elevation and the idea that water flows in a downward direction from elevated to lower altitudes, ultimately accumulating water in low-gradient areas with a potential for ponding conditions (see Nobre, et al., 2011). Consequently, the vertical distance between a given point and the closest stream or river is crucial in determining the likelihood of water movement toward the downstream portion. The initial step in generating a HAND model utilizes a DEM model to produce a flow accumulation map. The map portrays the number of cells that contribute to the flow of each cell within the DEM. Typically, the cells exhibiting the greatest flow accumulation are situated in proximity to the streams and rivers. Subsequently, a distance transform algorithm determines the distance between each cell in the DEM and the closest stream or river. Subtracting the elevation of individual cells in the DEM from the distance to the closest stream or river results in the computation of the HAND value for that particular cell. The utilization of HAND values is viable in the creation of a HAND map, which effectively displays the altitude of individual points in relation to the nearest stream or river.

The HAND model utilizes a pair of methodologies on a DEM to normalize the terrain in relation to the hydrological network. The initial stage involves executing a sequence of computations to produce a DEM that adheres to hydrological principles, establishes pathways for water flow, and allocates drainage channels. The subsequent phase entails employing indigenous drain orientations and the drainage system to generate the nearest drainage chart, which will subsequently guide the HAND operator in establishing the normalized topology of the HAND model in a spatial manner. The HAND model is classified into various classes based on flood depth and the severity of inundation. These classes include class 1 (0 to 0.5 meters), class 2 (0.51 to 1 meters), class 3 (1.1 to 1.5 meters), class 4 (1.51 to 2.0 meters), class 5 (greater than 2.0 meters). The HAND model postulates that inundation occurs when the elevation of water surpasses the altitude above the adjacent stream or drainage (see Nobre, et al., 2011). The HAND methodology involves assigning a value to each pixel in a raster, which represents the relative elevation in meters between the pixel and the nearest water stream. Equation 6 provides the map algebra formula for calculating inundation that is equal to or less than the HAND value.

$$HAND\ raster < x \qquad \text{Equation (6)}$$

where x is the gauge height value.

Figure 3 presents a step-by-step example of HAND calculation. DEM is first filled to remove any sinks or pits (Step 1). The D8 flow direction raster file is then generated to determine flood direction. A flow accumulation (Step 3) and a stream raster (Step 4) are then generated to calculate the amount of flood at the outlet of a drainage system. DINF (D-Infinite) is calculated to create a raster of flow direction from each cell to its downslope neighbor or neighbors (see Tarboton, 1997). The flow distance raster is then generated using flow direction raster and the vertical distance (elevation differences). For the last step (i.e., Step 8), Equation 7 is used to calculate the HAND based inundation area and integer 2 is considered the flood depth.

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

Natural Hazards and Earth System Sciences
Open Access
EGU
Discussions

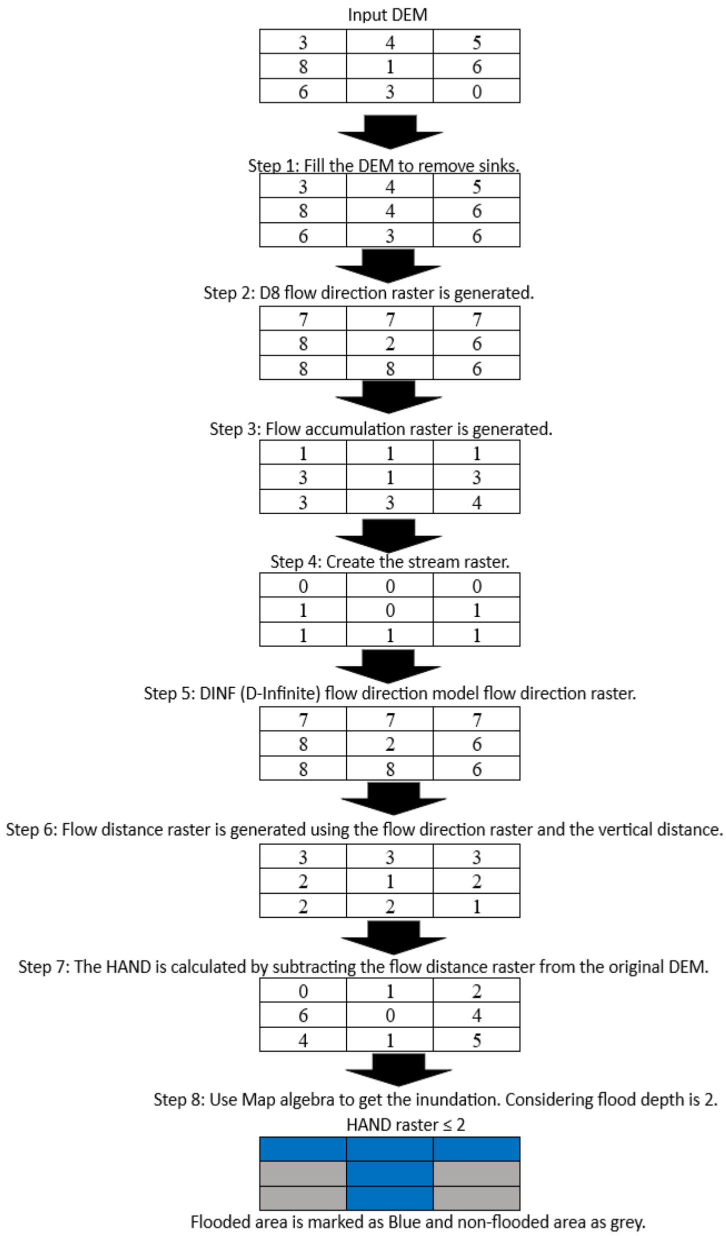

Figure 3: The workflow of HAND model.

Initially, the HAND elevation data was acquired from Liu et al. (2020) and used to generate a hydrological terrain raster, known as HAND, for a Hydrologic Unit Code 6 (HUC6) region in the contiguous United States (CONUS). This was achieved by utilizing a DEM with a 10-meter resolution obtained from the USGS 3-D Elevation Program (3DEP) and the National Hydrography Dataset (NHD) Plus hydrography

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

dataset. The HAND data was then generated (by following the steps explained above) by utilizing geospatial data sources such as the National Hydrography Dataset (NHD). Then, the aforementioned data was amalgamated with the hydraulic property data to generate an all-encompassing dataset. Subsequently, the map algebra approach in Python was employed to compute the HAND value for each raster grid. Next, the HAND map was categorized into several classes to reflect flood depth, as explained above.

2.4. Social Media Text Mining

Human input is a crucial component of HAC architecture. Collecting data about how humans respond to a flooding event is often lengthy and time-consuming; however, with new technology advancements, gathering this human data has become less complicated. This research collected data generated by humans using social media, here X. We used the X API to collect human data including real-time updates, posts/tweets, and contextual information. The X API is a valuable tool for collecting X posts related to flooding because the posts provide near real-time updates on flood conditions and evacuation efforts. The X API enables developers to search for messages using particular keywords or hashtags, making it simple to collect relevant data. The X API also provides metadata about messages, such as location and time, which can be used to filter and analyze the collected data. In this study, only X posts from SC were retrieved.

In addition, we designed and developed a text classification model to filter only those X posts that were deemed relevant to flooding. Specifically, we used Google's Bidirectional Encoder Representations from Transformers (BERT) package to classify X posts. BERT is a cutting-edge, pre-trained NLP model with sophisticated neural network architecture and capacity for contextual text analysis (Khan et al., 2023). The BERT model can generate high-quality representations of natural language text by simultaneously considering the entire input sequence of words to the left and right of the target word, thereby enabling more contextually relevant representations. In contrast to previous NLP models, which only consider the context of the target term's left and right word, this model considers the entire sentence.

To classify posts related to flooding, the BERT model was trained on a labeled dataset of X posts, where each X post was categorized as relevant or irrelevant to flooding. A text classifier was created on top of the BERT model. After the X posts were collected using X API in real-time, we performed text classification of collected X posts. This text classification decided whether the X post was relevant to a flood disaster or not. Figure 4 illustrates the workflow of BERT combined with the HAND model. After collecting relevant X posts using the BERT approach, the outcomes were integrated into the HAND model to validate the inundation areas. We used various geospatial information to integrate HAND and BERT outcomes with the transportation data.

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

Natural Hazards and Earth System Sciences
Open Access
Discussions

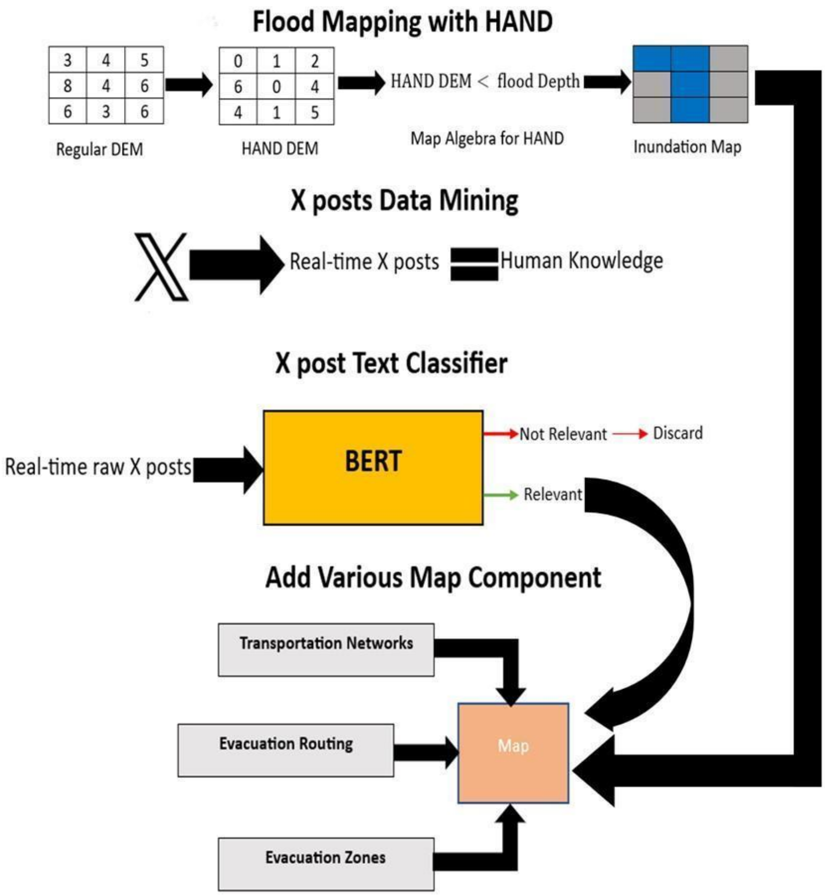

Figure 4: The workflows of the system after LSTM prediction

2.4.1. Performance Metrics Used in X Post Classification

To evaluate the performance of the BERT model for classifying X posts, we used three standard performance metrics: accuracy, precision, and recall. Accuracy measures the model's overall performance and represents the percentage of X posts correctly classified as relevant or irrelevant to flooding. This metric is essential for evaluating the general effectiveness of the model. Precision measures the proportion of correctly classified relevant X posts among all X posts classified as relevant by the model. This metric is essential for evaluating the accuracy of the positive predictions made by the model. Recall measures the proportion of correctly classified relevant X posts among all relevant X posts in the dataset. This metric is essential for evaluating the completeness of the positive predictions made by the model. Equations 7, 8, and 9 are accuracy, precision, and recall formulas, respectively. In these equations, *TP* denotes true positive, *TN* is true negative, *FP* represents false positive, *P* is total positive classes, and *N* denotes total negative classes.

$$Accuracy = \frac{TP + TN}{P + N} \quad \text{(Equation 7)}$$

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

Natural Hazards and Earth System Sciences Discussions

$$Precision = \frac{TP}{TP + FP} \quad \text{(Equation 8)}$$

$$Recall = \frac{TP}{TP + FN} \quad \text{(Equation 9)}$$

2.5. HAC System Structure and Workflow

Figure 5 illustrates the workflow of the HAC system. The system combines multiple modules including a machine learning prediction model, Rational method, river hydraulic model, BERT text mining approach, evacuation re-routing model, and visualization. The system first predicts gauge height using machine learning approaches and uses the Rational method if USGS gauging data is not available for a particular watershed. The estimated gauge height or flood depth is then used in the HAND model for flood inundation mapping. The inundation outcome is integrated into the BERT model and X posts to validate the inundation areas. Finally, the system uses the Grasshopper API to avoid inundated roads and suggest rerouting. To perform evacuation re-routing, a Leaflet routing machine, and a JavaScript library for interactive re-routing in web applications were used to connect with the Graphhopper API. The Graphhopper API provides various re-routing algorithms using the 'alternative_route' algorithm, which generates multiple alternative routes for a given start and end point.

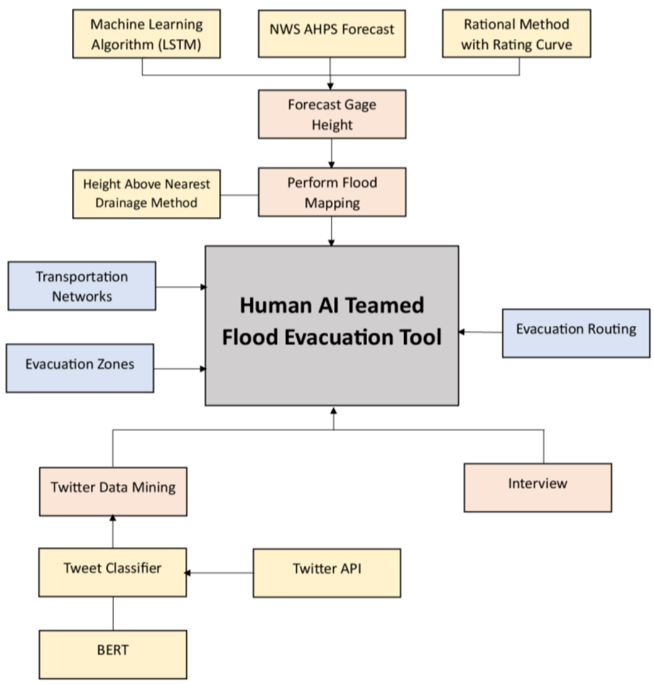

Figure 5: The overall workflow of the HAC flood evacuation system.

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

3. Results and Applications

This section includes flood forecasting using machine learning approaches as well as human data collection, inundation mapping, and evacuation re-routing. The results are presented for the three gauging stations across the Lowcountry, SC.

3.1. Flood Forecasting

We conducted training for both the LSTM and GRU models. A total of 180 models were trained using the Optuna algorithm, from which the top three models were selected. To tune the hyperparameters, we minimized the validation loss function in Optuna. In each gauging station, Optuna computed the number of neurons in each layer, dropout rate, and number of epochs. Optuna trained both LSTM and GRU and optimized hyperparameters. The number of neurons varied significantly among hidden layers with the least number in the 6th layer (5 to 15) and the maximum value in the first hidden layer (100 to 200). The drop-out rate ranged between 0.1 to 0.5 with Epoch number of 50-200. The three gauging stations that were used include Turkey Creek (USGS02172035), South Fork Edisto River (USGS02173000), and North Fork Edisto River (USGS02173500). We used 03/01/2013 to 05/08/2023 datasets to simulate gauge height values for USGS02173500 and USGS0217035. Due to data unavailability, we used 01/01/2020 to 05/08/2023 period to predict and forecast gauge height values at USGS02173000.

The performance of both the LSTM and GRU models exhibited a high degree of similarity. However, the LSTM model exhibited slightly superior performance, particularly on the test dataset. Therefore, only the LSTM model is presented here. LSTM was particularly successful in capturing flood peak rates and time to peak, which are two important factors for flood emergency decisions. Table 1 shows the performance achieved by LSTM as the best model for all gauging stations in testing, validation, and training periods, respectively. The LSTM was proficient in simulating gauge heights across the three gauging stations, with respect to the multiple performance metrics. Error estimation metrics such as Huber Loss, MSE, and MAE was comparably low across different gauging stations. Although, error estimation metrics were particularly low and somewhat close to zero during validation and training periods (see Table 1).

Table 1. LSTM performance across three gauging stations.

Station	MASE	Huber Loss	MSE	MAE
Training Period				
USGS02173500	0.0028	0.0025	0.0050	0.0430
USGS02173000	0.0063	0.0020	0.0040	0.0447
USGS0217035	0.0094	0.0188	0.0890	0.0066
Testing Period				
USGS02173500	0.0019	0.0120	0.02404	0. 1284
USGS02173000	0.0066	0. 0248	0.0498	0.1619
USGS0217035	0.00460	0.0177	0.0354	0.1406
Validation Period				
USGS02173500	0.0027	0.0070	0.0140	0.0887

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

USGS02173000	0.0052	0.0075	0.0150	0.0634
USGS0217035	0.0078	0.0065	0.0131	0.0696

Prediction results of LSTM are visualized in Figures 6, 7, and 8. As shown, the LSTM was able to accurately predict a quick rise and fall of the flood gauge height values, particularly at USGS02173000. Although the slope and behavior of the gauge height data are not well captured in the Turkey Creek gauging station (USGS0217035). In addition, low gauge height values are not well captured by LSTM across all three gauging stations. Specifically, when the gauge height values were less than 4 meters, the LSTM performance dropped significantly. LSTM appears to be sensitive to the widespread scale of data, so the model might learn that the low gauge height values carry no information. In addition, the prediction of low gauge height values is a challenge for machine learning models since those data do not add much value to the learning process.

In addition, LSTM showed overfitting in terms of low gauge height values prediction. This network, with its ability to capture long-term dependencies, is prone to overfitting, especially when the data values are small. To mitigate this issue, we implemented Optuna as an early stopping technique to monitor the model performance during the validation period and stop the training process when the performance begins to degrade. While Optuna allowed us to implement state-of-the-art optimization algorithms to speed up the hyperparameter tuning process, these advanced algorithms are built to efficiently search for the best objective when the cost to iterate the model training process is too expensive. If we train our models with a small amount of data, it is possible that Optuna uses Random Search and Grid Search for hyperparameters tuning which can either spend too much time or can’t even locate the minima. On the other hand, if the data volume increases and models get more complex, the cost of using Random Search and Grid Search to train a set of hyperparameters increases significantly.

Further, LSTM can suffer from vanishing or exploding gradient problem during training. When the gradients become too small, it is hard for the model to learn long-term dependencies in the dataset, resulting in unstable training. One can also note the insignificant differences between modelling performances of USGS02173000 and USGS02173500. These two gauging stations are part of a large Edisto River Basin. This concludes that that LSTM was able to learn the gauge height fluctuations and dependencies across a large basin.

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

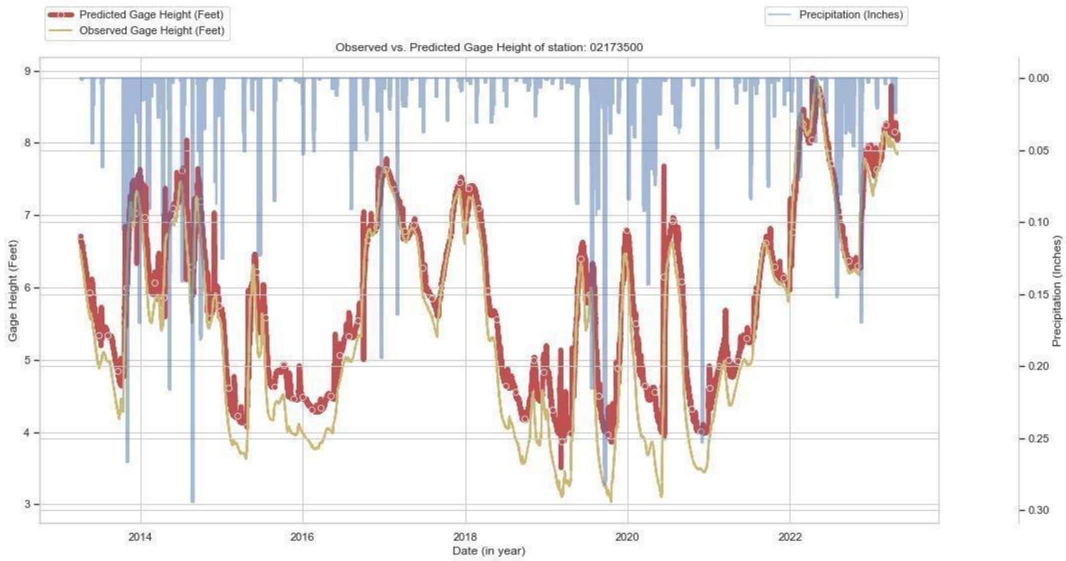

Figure 6: USGS02173500 flood gauge height prediction for training, testing and validation periods (03/01/2013 to 05/08/2023).

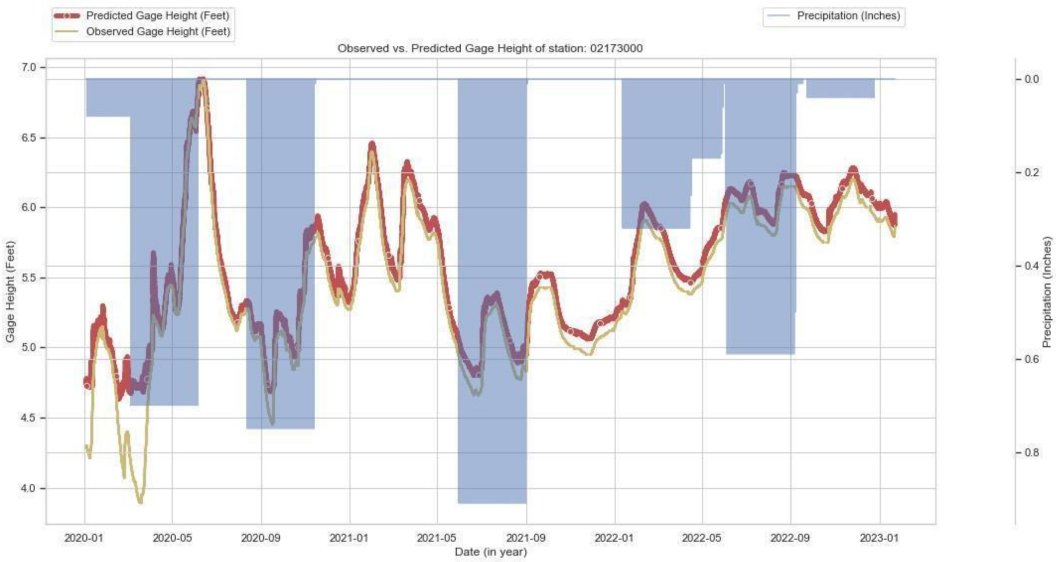

Figure 7: USGS02173000 flood gauge height prediction for training, testing and validation periods (01/01/2020 to 05/08/2023).

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

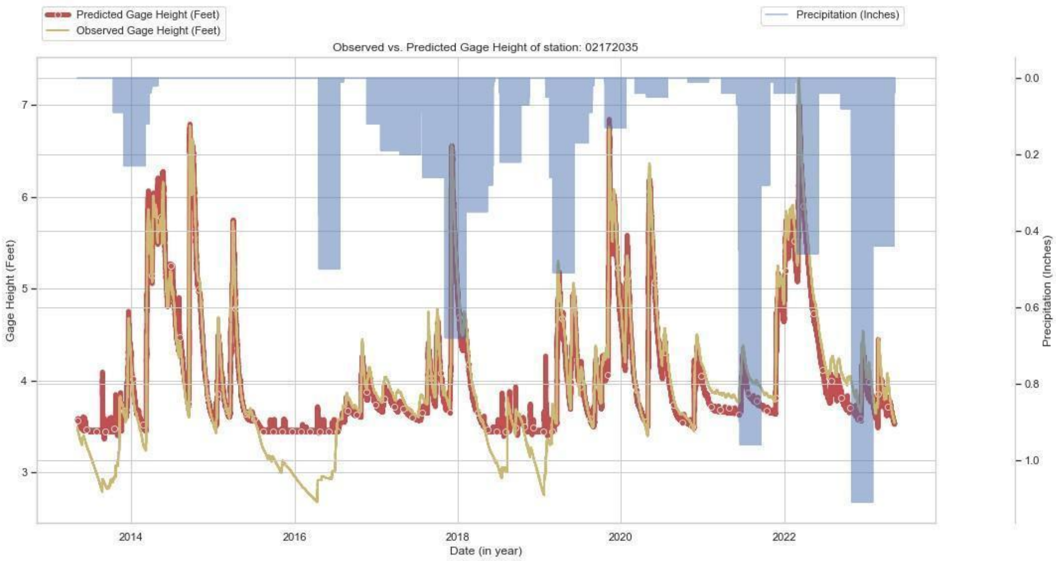

Figure 8: USGS0217035 Flood gauge height prediction for training, testing and validation periods (03/01/2013 to 05/08/2023).

3.2. Social Media Data Analysis

In this research, we leveraged unstructured social media data to enrich the human dimension of the HAC system. The real-time nature of social media, especially X, was instrumental in adding real-time human knowledge to the HAC system. The social media text data provided immediacy and diversity of perspectives, offering contextual richness to incorporating human data into the system. We used the X API to collect and mine text data. The keywords that were used to search for X posts include “floods,” “flood emergency,” “road damage,” and "evacuation". These keywords can be customized by the user to collect data on specific flood events or locations. We filtered X text data and threw away those data that were not relevant to flooding. The filtration was performed via an X post-classifier model that was constructed utilizing an NLP model called BERT algorithm. We first created a text classifier on top of the BERT model. We then collected the dataset from various sources such as WALLACH, (2018), Preda, (2020), Stepanenko and Liubko, (2020), Alam et al., (2021), and Suresh, (2021) to train BERT. Overall, a dataset of approximately 60,000 X text posts was gathered and manually annotated to indicate whether each post was related to flooding or not. This dataset was used to develop an X text post classifier model. The text data was partitioned into two datasets, namely a training set and a testing set, with a ratio of 75%:25%.

These data sources also contain irrelevant X posts so that BERT could learn to distinguish between relevant and irrelevant X posts by accumulating irrelevant X posts alongside relevant ones during training. By including irrelevant X posts in the training data, the model learned to differentiate between various categories of X posts and identified which specific features or keywords indicate whether an X post is relevant or irrelevant to flooding. Each text was then given a category of 0 (not relevant) or 1 (relevant). During the process of fine-tuning, the BERT model learned to identify key flood-related textual features and used them to make accurate predictions.

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

The fine-tuning process involved training the BERT model on the labeled dataset and adjusting its parameters to classify relevant X posts. A backpropagation method was then used for the text fine-tuning process. The model was trained for 30 epochs. The model's predictions were compared with the true labels, and the model's parameters were adjusted to minimize the difference between the predictions and the true labels. Once the BERT model was fine-tuned, it was then used to classify new, unlabeled X posts as either relevant or irrelevant to flooding. The model's output is a probability score indicating the likelihood that the X post is relevant to flooding. If the probability score is above a threshold of 80%, the X post is classified as relevant to flooding. After the X posts were collected using X API in real-time, we performed text classification of collected X posts. This text classification decided whether the X post was relevant to a flood disaster or not.

Figure 9 shows the BERT architecture designed to classify the X posts. The input dimension (i.e., 60,000 X text data) was chosen “None” to set the dimension to any scalar number (Abadi, et al., 2016). So, the input dimension was arbitrary to the input text length. A pre-processed layer obtained from a pre-existing saved text preprocessing layer was then utilized to preprocess the text data in TensorFlow Hub. This layer served as a companion to the BERT model, facilitating the preprocessing of plain text inputs into the specific format that BERT required. The pre-processed layer's output was linked to the input of the BERT encoder layer sourced from a pre-existing TensorFlow Hub model that was trained beforehand. BERT utilized a Transformer architecture and a deep, pre-trained neural network to generate dense vector representations for natural language. The BERT model employed 12 hidden layers, also known as Transformer blocks, with a hidden size of 768 and 12 attention heads. The weights utilized in this model correspond to those disclosed by the primary authors of BERT. The outputs of the encoder consist of two components: the "pooled_output," which served to encapsulate the entirety of the input sequence, and the "sequence_output," which represented each individual token within the context of the sequence. The output obtained from pooling was linked to the dropout layer with a rate of 0.1. The dropout was subsequently linked to a densely connected output layer.

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

Natural Hazards and Earth System Sciences
Discussions

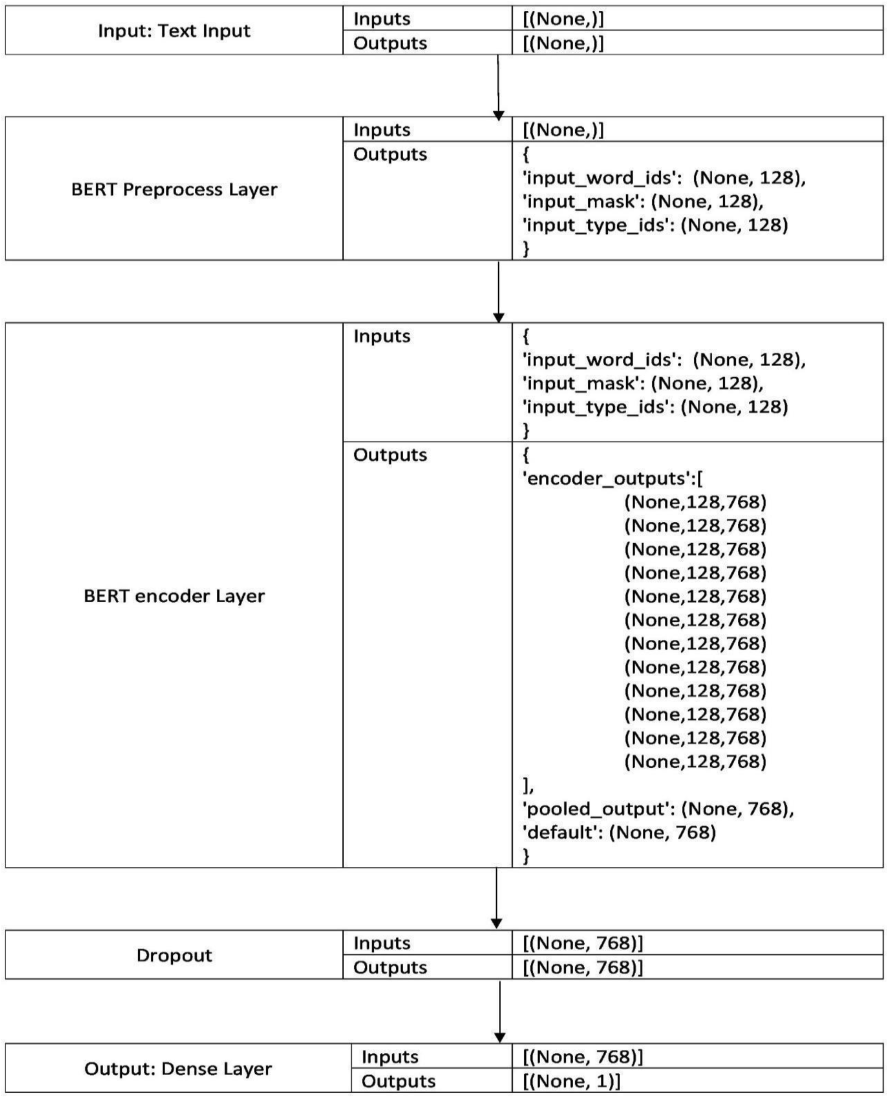

Figure 9: Neural network architecture of X post classifier constructed in this research.

The BERT model attained an accuracy rate of 88.5% along with a precision rate of 0.84% and a recall rate of 0.85% during the training period. Similarly, during the testing phase, the model achieved an accuracy rate of 89%, a precision rate of 81%, and a recall rate of 94%. The count of the number of predicted versus actual of each class was obtained using a confusion matrix (see Table 2). As shown, the number of positive predicted values are much higher that negative values.

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

Table 2. Confusion Matrix for the test set of the BERT model.

		Predicted Values	
		Negative	Positive
Actual Value	Negative	8183	1238
	Positive	319	5209

BERT's architecture, especially its bidirectional mechanism, was pivotal in understanding the context behind text classification. When we examined BERT for real time X post classification, its performance highlighted the model's ability to effectively utilize human-generated data in the context of evolving flood situations. BERT demonstrated its efficacy as a vital tool for contemporary flood prediction systems by efficiently eliminating extraneous data and focusing on relevant flood-related information. Although, the research progress of X post mining was considerably affected by the X decision of not supporting free access to the API.

3.3. Evacuation Re-routing Results

A routing algorithm was included in the HAC system to suggest alternative routes in case of flooding. A Leaflet routing machine through the Graphhopper API was employed to generate evacuation re-routing during a major flooding event on January 10, 2024, in Lowcountry, SC. The Grasshopper API was then integrated into the prototype to calculate the shortest or alternative routes between multiple points. The parameters passed to the API include the algorithm type (alternative_route), maximum number of routes (max_paths), maximum weight factor (max_weight_factor), and maximum sharing factor (max_share_factor). The key parameters used in the Graphhopper API for our case studies are:

(i) Algorithm Type (alternative_route): This parameter specifies the algorithm used for routing, with alternative_route being particularly useful for evacuation as it provides several route options.

(ii) Maximum Number of Routes (max_paths): Determines the number of alternative routes to generate. In evacuation scenarios, having multiple paths ensures that there are options available if the primary route becomes impassable.

(iii) Maximum Weight Factor (max_weight_factor): This parameter influences the maximum weight of the alternative paths, which can be interpreted as a measure of route efficiency in terms of distance.

(iv) Maximum Sharing Factor (max_share_factor): This parameter controls the degree of similarity between the alternative routes. A lower sharing factor tells the algorithm to provide routes that diverge from each other, which can increase the chances of avoiding blocked areas.

These parameters ensure that multiple evacuation routes are generated and that flooded areas are avoided as much as possible. After the API returned the routes, each route was checked to see if it was in the

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

inundation area or not. The fastest route, which was out of the inundation area, was then selected and displayed on the map to guide citizens toward a safe evacuation.

We used 'alternative_route' as a parameter in the Graphhopper API to generate and return multiple routes; the best route that was not inundated was selected as an alternative route. When the API returned data for multiple routes, each route was checked manually to ensure it was out of the flooded/inundation area; the fastest route was then driven from point A to point B (see Figure 10). The optimal route — the fastest one clear of flooding — was displayed on the map to offer a reliable guide for safe evacuation rerouting. By actively avoiding inundated areas, the system ensured that the evacuation routes remain as safe as possible.

Figure 10: Map showing re-routed path in Hwy 41 (near Turkey Creek River) during January 10, 2024, flooding in Lowcountry, SC (Leafmap Python package [see Wu, 2021] was used for interactive mapping and geospatial analysis).

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

4. Conclusions

The HAC system which was created in this study as a flood evacuation system, exhibited a high degree of efficacy in its ability to forecast river gauge height and suggest evacuation re-routing by combining LSTM with a river hydraulic model and relevant human information. The incorporation of social media data added a humanistic dimension to the developed HAC system and facilitated the identification of regions that may require prompt aid and evacuation consideration. In general, the prototype exhibited significant potential for disaster response applications and evacuation endeavors within low-lying regions of SC that can be applied to other flood-prone areas. The high accuracy and precision achieved by the LSTM and BERT models demonstrated the effectiveness of machine learning and NLP in predicting river gauge height values and filtering relevant social media data which could provide ground information to decision makers.

Traditionally, machine learning models rely on historical data to make predictions, but this approach may need to be revised in unpredictable circumstances like flooding, where real-time information is critical. To address this challenge, we introduced a new methodology incorporating machine learning predictions and X data into a geographical representation. The cartographic representation functions as an interface between machine learning and human inputs, facilitating mutual reinforcement and enhancing the precision of predictions. Utilizing X text data enabled the acquisition of contemporary human knowledge, augmenting the predictive capacity of machine learning models. The integration of two sources of information was facilitated by utilizing a visualization map as a platform, creating a cohesive perspective of the flood evacuation situation.
The HAC system is a novel approach developed towards achieving human-AI collaboration in flood evacuation problems. As discussed, the HAC system leverages extant competencies to strategically coordinate the interplay between X text data and machine learning and flood inundation models to analyze the outcomes and suggest evacuation re-routing alternatives. HAC system development involved integrating a range of algorithms, data, and information to test the prototype in real-time across Lowcountry, SC—a flood prone area.

The utilization of the HAC system in flood evacuation decisions has the potential to augment human capabilities and knowledge, thereby increasing the prototype's overall robustness and effectiveness. Human-AI collaboration continues to evolve, and its decision-making and prediction can help teams deal with real-time evacuation decisions. At the same time, societal demands for more accurate flood evacuation decisions will continue to increase; therefore, the need for advanced technologies such as HAC will continue growing. Engineering solutions to flood management problems, including evacuation and warning and real-time decision-making, increasingly rely on sophisticated computational solutions rather than traditional and empirical assessment. At the same time, scientists working in machine learning applications and flood emergencies will increasingly be pushed towards inquiry that is directly relevant to societal decision-making. These include incorporating human factors into machine learning based flood forecasting, which has important consequences for people's safety and protection. In the future, more research is needed to develop additional methods that incorporate human data into the HAC system that consider flood situational conditions; these can inform emergency officials of when they can rely on an AI system and when they need to intervene. This research will serve as a foundation for future studies exploring the potential of human-AI collaboration in flood disaster and response domains. Exploring and testing the HAC approach could unlock new possibilities for achieving more significant breakthroughs in various human-AI teaming applications in flood modeling and management domains.

5. Acknowledgements

This research is supported by the US National Science Foundation (NSF) Directorate of Engineering (Grant # CMMI 2125283). Clemson University (USA) is acknowledged for generous allotment of computing time on the Palmetto cluster. We acknowledge the NSF funded the Advanced Cyberinfrastructure Coordination Ecosystem: Services and Support (ACCESS) program for providing free cloud credits to deploy HAC system. The HAC system is available at http://floodevacuationtool.clemson.edu/. The source code and data are available upon request.

Competing Interests: The authors declare that they have no conflict of interest.

6. References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., . . . others. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, (pp. 2623–2631).

Alam, F., Qazi, U., Imran, M., & Ofli, F. (2021). Humaid: human-annotated disaster incidents data from twitter with deep learning benchmarks. *Proceedings of the International AAAI Conference on Web and Social Media*, *15*, pp. 933–942.

Bansal, G., Nushi, B., Kamar, E., Lasecki, W. S., Weld, D. S., & Horvitz, E. (2019). Beyond accuracy: The role of mental models in human-AI team performance. *Proceedings of the AAAI conference on human computation and crowdsourcing*, *7*, pp. 2–11.

Bhatti, S., Demir, M., Cooke, N. J., & Johnson, C. J. (2021). Assessing Communication and Trust in an AI Teammate in a Dynamic Task Environment. *2021 IEEE 2nd International Conference on Human-Machine Systems (ICHMS)*, (pp. 1–6).

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. *Proceedings of the fifth annual workshop on Computational learning theory*, (pp. 144–152).

Cho, K. a. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Council, N. R., Committee, M. S., & others. (2007). *Successful response starts with a map: improving geospatial support for disaster management.* National Academies Press.

Dawson, C. W., & Wilby, R. (1998). An artificial neural network approach to rainfall-runoff modelling. *Hydrological Sciences Journal*, *43*, 47–66.

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

Demir, M., McNeese, N. J., & Cooke, N. J. (2020). Understanding human-robot teams in light of all-human teams: Aspects of team interaction and shared cognition. *International Journal of Human-Computer Studies, 140*, 102436.

Donratanapat, N., Samadi, S., Vidal, J. M., & Tabas, S. S. (2020). A national scale big data analytics pipeline to assess the potential impacts of flooding on critical infrastructures and communities. *Environmental Modelling & Software, 133*, 104828.

Eglash, R., Robert, L., Bennett, A., Robinson, K. P., Lachney, M., & Babbitt, W. (2020). Automation for the artisanal economy: enhancing the economic and environmental sustainability of crafting professions with human–machine collaboration. *Ai & Society, 35*, 595–609.

Flathmann, C., Schelble, B. G., Rosopa, P. J., McNeese, N. J., Mallick, R., & Madathil, K. C. (2023). Examining the impact of varying levels of AI teammate influence on human-AI teams. *International Journal of Human-Computer Studies, 177*, 103061.

FloodMap Mobile. (2023). *FloodMap Mobile*.

Goodwin, G. F., Blacksmith, N., & Coats, M. R. (2018). The science of teams in the military: Contributions from over 60 years of research. *American Psychologist, 73*, 322.

Han, D., Chan, L., & Zhu, N. (2007). Flood forecasting using support vector machines. *Journal of hydroinformatics, 9*, 267–276.

Hancock, D. Y., Fischer, J., Lowe, J. M., Snapp-Childs, W., Pierce, M., Marru, S., . . . others. (2021). Jetstream2: Accelerating cloud computing via Jetstream. In *Practice and Experience in Advanced Research Computing* (pp. 1–8).

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation, 9*, 1735–1780.

Holstein, K., Aleven, V., & Rummel, N. (2020). A conceptual framework for human–AI hybrid adaptivity in education. *Artificial Intelligence in Education: 21st International Conference, AIED 2020, Ifrane, Morocco, July 6–10, 2020, Proceedings, Part I 21*, (pp. 240–254).

HURREVAC National Hurricane Program. (2023). *HURREVAC National Hurricane Program*.

Integrated Public Alert and Warning System. (2023). *Integrated Public Alert and Warning System*.

Istalkar, P., Kadu, A., & Biswal, B. (2023). Value of process understanding in the era of machine learning: A case for recession flow prediction. *Journal of Hydrology*, 130350.

Khan, W., Daud, A., Khan, K., Muhammad, S., & Haq, R. (2023). Exploring the frontiers of deep learning and natural language processing: A comprehensive overview of key challenges and emerging trends. *Natural Language Processing Journal*, 100026.

Laird, J., Ranganath, C., & Gershman, S. (2020). Future directions in human machine teaming workshop. *Arlington, VA: US Department of Defense*.

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

Natural Hazards and Earth System Sciences
Open Access
EGU
Discussions

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE, 86*, 2278–2324.

Liang, C., Proft, J., Andersen, E., & Knepper, R. A. (2019). Implicit communication of actionable information in human-ai teams. *Proceedings of the 2019 CHI conference on human factors in computing systems*, (pp. 1–13).

Liu, D., Jiang, W., Mu, L., & Wang, S. (2020). Streamflow prediction using deep learning neural network: case study of Yangtze River. *IEEE access, 8*, 90069–90086.

Liu, Y., Tarboton, D. G., & Maidment, D. R. (2020). *Height Above Nearest Drainage (HAND) and Hydraulic Property Table for CONUS.* Tech. rep., Oak Ridge National Lab.(ORNL), Oak Ridge, TN (United States). Oak Ridge

Madni, A. M., & Madni, C. C. (2018). Architectural framework for exploring adaptive human-machine teaming options in simulated dynamic environments. *Systems, 6*, 44.

McCall, F., Hussein, A., Petraki, E., Elsawah, S., & Abbass, H. (2021). Towards a systematic educational framework for human-machine teaming. *2021 IEEE International Conference on Engineering, Technology & Education (TALE)*, (pp. 375–382).

McGrath, J. E. (1984). *Groups: Interaction and performance* (Vol. 14). Prentice-Hall Englewood Cliffs, NJ.

McKinney, W. (2010, June). Data structures for statistical computing in python. In Proceedings of the 9th Python in Science Conference (Vol. 445, No. 1, pp. 51-56).

McNeese, N. J., Demir, M., Cooke, N. J., & Myers, C. (2018). Teaming with a synthetic teammate: Insights into human-autonomy teaming. *Human factors, 60*, 262–273.

Morgan Jr, B. B., Salas, E., & Glickman, A. S. (1993). An analysis of team evolution and maturation. *The Journal of General Psychology, 120*, 277–291.

Nevo, S., Morin, E., Gerzi Rosenthal, A., Metzger, A., Barshai, C., Weitzner, D., . . . others. (2022). Flood forecasting with machine learning models in an operational framework. *Hydrology and Earth System Sciences, 26*, 4013–4032.

Nobre, A. D., Cuartas, L. A., Hodnett, M., Rennó, C. D., Rodrigues, G., Silveira, A., & Saleska, S. (2011). Height Above the Nearest Drainage–a hydrologically relevant new terrain model. *Journal of Hydrology, 404*, 13–29.

Ong, C., McGee, K., & Chuah, T. L. (2012). Closing the human-AI team-mate gap: how changes to displayed information impact player behavior towards computer teammates. *Proceedings of the 24th Australian Computer-Human Interaction Conference*, (pp. 433–439).

Pally, R. J., & Samadi, S. (2022). Application of image processing and convolutional neural networks for flood image classification and semantic segmentation. *Environmental Modelling & Software, 148*, 105285.

Pourreza-Bilondi, M., Samadi, S. Z., Akhoond-Ali, A.-M., & Ghahraman, B. (2017). Reliability of semiarid flash flood modeling using Bayesian framework. *Journal of Hydrologic Engineering, 22*, 05016039.

Preda, G. (2020). COVID19 Tweets. *COVID19 Tweets*. Kaggle. doi:10.34740/KAGGLE/DSV/1451513

Preda, G. (2021). Pfizer Vaccine Tweets. *Pfizer Vaccine Tweets*. Kaggle.

Puig, X., Shu, T., Li, S., Wang, Z., Liao, Y.-H., Tenenbaum, J. B., . . . Torralba, A. (2020). Watch-and-help: A challenge for social perception and human-ai collaboration. *arXiv preprint arXiv:2010.09890*.

Rabbani, M., Oladzad-Abbasabady, N., & Akbarian-Saravi, N. (2022). Ambulance routing in disaster response considering variable patient condition: NSGA-II and MOPSO algorithms. *Journal of Industrial & Management Optimization, 18*.

Seeber, I., Bittner, E., Briggs, R. O., De Vreede, T., De Vreede, G.-J., Elkins, A., . . . others. (2020). Machines as teammates: A research agenda on AI in team collaboration. *Information & management, 57*, 103174.

Stepanenko, V., & Liubko, I. (2020). Disaster Tweets. *Disaster Tweets*. Kaggle. doi:10.34740/KAGGLE/DSV/1640141

Stephens, K. K., Harris, A. G., Hughes, A. L., Montagnolo, C. E., Nader, K., Stevens, S. A., ... & Zobel, C. W. (2023). Human-AI Teaming During an Ongoing Disaster: How Scripts Around Training and Feedback Reveal This Is a Form of Human-Machine Communication. Human-Machine Communication, 6(1), 5.

Sundar, S. S. (2020). Rise of machine agency: A framework for studying the psychology of human–AI interaction (HAII). *Journal of Computer-Mediated Communication, 25*, 74–88.

Suresh, K. (2021). Bitcoin Tweets. *Bitcoin Tweets*. Kaggle.

Tarboton, D. G. (1997). A new method for the determination of flow directions and upslope areas in grid digital elevation models. Water Resources Research, 33(2), 309-319.

Thirumalaiah, K., & Deo, M. C. (2000). Hydrological forecasting using neural networks. *Journal of Hydrologic Engineering, 5*, 180–189.

Turing, A. M. (1950). Computing Machinery and Intelligence. *Mind, 59*, 433–460.

WALLACH, D. A. (2018). Financial Tweets. *Financial Tweets*. Kaggle. Retrieved from https://www.kaggle.com/datasets/davidwallach/financial-tweets

Wee, G., Chang, L.-C., Chang, F.-J., & Amin, M. Z. (2023). A flood Impact-Based forecasting system by fuzzy inference techniques. *Journal of Hydrology, 625*, 130117.

Wenskovitch, J., Fallon, C., Miller, K., & Dasgupta, A. (2021). Beyond Visual Analytics: Human-Machine Teaming for AI-Driven Data Sensemaking. *2021 IEEE Workshop on TRust and EXpertise in Visual Analytics (TREX)*, (pp. 40–44).

Windheuser, L., Karanjit, R., Pally, R., Samadi, S., & Hubig, N. C. (2023). An end-to-end flood stage prediction system using deep neural networks. *Earth and Space Science, 10,* e2022EA002385.

Wu, Q., 2021. Leafmap: A Python package for interactive mapping and geospatial analysis with minimal coding in a Jupyter environment. Journal of Open Source Software, 6(63), p.3414.

Yu, P.-S., Chen, S.-T., & Chang, I.-F. (2006). Support vector regression for real-time flood stage forecasting. *Journal of hydrology, 328*, 704–716.

Zhang, G., Chong, L., Kotovsky, K., & Cagan, J. (2023). Trust in an AI versus a Human teammate: The effects of teammate identity and performance on Human-AI cooperation. *Computers in Human Behavior, 139*, 107536.

Wang, H., Xu, S., Xu, H., Wu, Z., Wang, T. and Ma, C., 2023. Rapid prediction of urban flood based on disaster-breeding environment clustering and Bayesian optimized deep learning model in the coastal city. Sustainable Cities and Society, 99, p.104898.

Sreejith, R. and Sinimole, K.R., 2022. Modelling evacuation preparation time prior to floods: A machine learning approach. Sustainable Cities and Society, 87, p.104257.

Phillips, P. 2020. Flooding forces closure of several Charleston-area roads. Available at https://www.live5news.com/2020/03/05/list-flooding-forces-closure-several-downtown-roads/. Accessed on Jan. 03., 2024.

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

Natural Hazards and Earth System Sciences
Discussions

Appendix A

Pseudocode

Pseudocode 1: Calculate HAND from DEM

```
requires:  flood_depth, TL, BR, hand_DEM_path

hand_dataset <- OPEN_GIS_DATASET(hand_DEM_path)
band <- GET_BAND(hand_dataset, 1)

transform <- GET_DATASET_GEOTRANSFORM(hand_dataset)
inv_transform <- CALCULATE_INVERSE_GEOTRANSFORM(transform)

top_left_pixel <- APPLY_INVERSE_GEOTRANSFORM(inv_transform, TL.longitude, TL.latitude)
bottom_right_pixel <- APPLY_INVERSE_GEOTRANSFORM(inv_transform, BR.longitude, BR.latitude)

start_x <- MAX(0, INT(MIN(top_left_pixel.x, bottom_right_pixel.x)))
end_x <- MIN(GET_DATASET_WIDTH(hand_dataset), INT(MAX(top_left_pixel.x, bottom_right_pixel.x)) + 1)
start_y <- MAX(0, INT(MIN(top_left_pixel.y, bottom_right_pixel.y)))
end_y <- MIN(GET_DATASET_HEIGHT(hand_dataset), INT(MAX(top_left_pixel.y, bottom_right_pixel.y)) + 1)

hand_array_clipped <- READ_RASTER_DATA_AS_ARRAY(band, start_x, start_y, width <- end_x - start_x, height <- end_y - start_y)

no_data_value <- GET_BAND_NO_DATA_VALUE(band)
IF no_data_value IS NOT NONE:
   REPLACE_ARRAY_VALUES(hand_array_clipped, no_data_value, NaN)

inundated_mask <- CREATE_MASK_WHERE(hand_array_clipped IS LESS OR EQUAL TO flood_depth AND NOT NaN)

bins <- DEFINE_BINS([0, 0.5, 1, 1.5, 2], flood_depth)

zone_indices <- ASSIGN_TO_BINS(hand_array_clipped[WHERE inundated_mask], bins)

Zone1, Zone2, Zone3, Zone4, Zone5 <- INITIALIZE_EMPTY_LISTS()
lat_list, lon_list <- INITIALIZE_EMPTY_LISTS()

FOR zone_number FROM 1 TO LENGTH(bins) + 1:
   zone_mask <- IDENTIFY_MASK_FOR_ZONE(zone_indices, zone_number)
   IF zone_mask CONTAINS TRUE VALUES:
      inundated_indices <- GET_TRUE_INDICES(inundated_mask)
      selected_indices <- FILTER_INDICES_BY_ZONE(inundated_indices, zone_mask)
      world_coords <- CONVERT_PIXELS_TO_WORLD_COORDINATES(selected_indices, transform, start_x, start_y)
      APPEND_ZONE_COORDINATES(Zone{zone_number}, world_coords)
      APPEND_LAT_LON_FROM_COORDINATES(lat_list, lon_list, world_coords)

FOR zone_number FROM LENGTH(bins) + 1 TO 5:
   CLEAR_ZONE_LIST(Zone{zone_number})

RETURN Zone1, Zone2, Zone3, Zone4, Zone5, lat_list, lon_list
```

This pseudocode1 requires flood depth (flood_depth), top left clipped rectangle coordinate (TL), bottom right clipped rectangle coordinate (BR) and HAND DEM path (hand_DEM_path). Functions used in this pseudo-code:

- OPEN_GIS_DATASET represents the process of opening the GIS file.

https://doi.org/10.5194/nhess-2024-25
Preprint. Discussion started: 15 February 2024

Natural Hazards and Earth System Sciences Open Access EGU Discussions

- GET_BAND, GET_DATASET_GEOTRANSFORM, and similar functions represent various GIS data operations.
- CALCULATE_INVERSE_GEOTRANSFORM calculates the inverse geotransformation matrix.
- APPLY_INVERSE_GEOTRANSFORM applies the inverse geotransform to convert world coordinates to pixel coordinates.
- READ_RASTER_DATA_AS_ARRAY reads raster data from the GIS file into a numeric array.
- REPLACE_ARRAY_VALUES replaces specific values in an array with another value.
- CREATE_MASK_WHERE creates a boolean mask based on a condition.
- DEFINE_BINS, ASSIGN_TO_BINS, and IDENTIFY_MASK_FOR_ZONE are used for categorizing the data into different zones based on the flood depth.
- CONVERT_PIXELS_TO_WORLD_COORDINATES converts pixel coordinates back to world.

Pseudocode 2: Forecast gauge height in real-time

requires: flood_station, period, scaler_path, model_path

```
TRY:
   herring <- NWIS_WEB_SERVICE(flood_station, 'iv', period)
   data <- EXTRACT_AND_PROCESS_DATA(herring)
CATCH Exception:
   HANDLE_DATA_RETRIEVAL_ERROR(flood_station)
   RETURN NONE

scaler <- LOAD_SCALER(scaler_path)
scaled_data <- PREPARE_DATA_FOR_PREDICTION(data, scaler)

custom_objects <- SETUP_MODEL_CUSTOM_OBJECTS()
regressor <- LOAD_MODEL(model_path, custom_objects)

prediction <- regressor.PREDICT(scaled_data)

inversed_gage_height <- POST_PROCESS_PREDICTIONS(prediction, scaler)

df_predictions <- PREPARE_PREDICTIONS_DATAFRAME(inversed_gage_height, data)

RETURN df_predictions
```

Pseudocode 2 requires identifier for the flood station (flood_station), time period over which to fetch data (period), path to the directory where the scaler files are stored (scaler_path) and path to the directory where the trained machine learning model files are stored (model_path). Function used in this pseudocode are:

- NWIS_WEB_SERVICE: Uses NWIS API to perform GET request and fetch data from it.
- EXTRACT_AND_PROCESS_DATA: Performs operations like fetching data, resampling, and timezone localization.
- HANDLE_DATA_RETRIEVAL_ERROR: Encapsulates error handling for data retrieval issues.
- LOAD_SCALER: Loads and returns scaler.
- PREPARE_DATA_FOR_PREDICTION: Prepares data scaling and preparation for input into the predictive model.
- LOAD_MODEL: Loads and return machine learning model.

- PREDICT: Predict and return predicted data.
- SETUP_MODEL_CUSTOM_OBJECTS: Represents the setup for custom objects required by the model, such as custom loss functions or metrics.
- POST_PROCESS_PREDICTIONS: Performs the post-processing of predictions to convert them from the scaled form back to the original measurement scale.
- PREPARE_PREDICTIONS_DATAFRAME: Prepares the creation of a DataFrame with predictions, including setting up the index with appropriate timestamps.