Automatic Segmentation of Sinkholes Using a Convolutional Neural Network

Muhammad Usman Rafique[1], Junfeng Zhu[2,3], Nathan Jacobs[4]

[1]Kitware Inc., Clifton Park, New York, 12065
[2]Kentucky Geological Survey, University of Kentucky, Lexington, Kentucky, 40506
[3]Department of Earth and Environmental Sciences, University of Kentucky, Lexington, Kentucky, 40506
[4]Department of Computer Science, University of Kentucky, Lexington, Kentucky, 40506

Key Points:

- Image segmentation models using elevation and aerial images were trained to locate sinkholes.
- Model's out-of-distribution generalization was assessed.
- DEM gradient images provide the best input for training sinkhole segmentation models.

Corresponding author: Junfeng Zhu, junfeng.zhu@uky.edu

Abstract

Sinkholes are the most abundant surface features in karst areas worldwide. Understanding sinkhole occurrences and characteristics is critical for studying karst aquifers and mitigating sinkhole-related hazards. Most sinkholes appear on the land surface as depressions or cover-collapses and are commonly mapped from elevation data, such as digital elevation models (DEMs). Existing methods for identifying sinkholes from DEMs often require two steps: locating surface depressions and separating sinkholes from non-sinkhole depressions. In this study, we explored deep learning to directly identify sinkholes from DEM data and aerial imagery. A key contribution of our study is an evaluation of various ways of integrating these two types of raster data. We used an image segmentation model, U-Net, to locate sinkholes. We trained separate U-Net models based on four input images of elevation data: a DEM image, a slope image, a DEM gradient image, and a DEM shaded relief image. Three normalization techniques (Global, Gaussian, and Instance) were applied to improve the model performance. Model results suggest that deep learning is a viable method to identify sinkholes directly from images of elevation data. In particular, DEM gradient data provided the best input for U-net image segmentation models to locate sinkholes. The model using the DEM gradient image with Gaussian normalization achieved the best performance with a sinkhole intersection-over-union (IoU) of 45.38 % on the unseen test set. Aerial images, however,were not useful in training deep learning models for sinkholes as the models using an aerial image as input achieved sinkhole IoUs below 3 %.

Plain Language Summary

Sinkholes are very common in areas with limestone rocks. Sinkholes can damage roads, buildings, and other infrastructure and sometimes even cost human lives. Sinkhole maps are needed for land use planning and hazard mitigation. Because sinkholes often occur in large numbers, often in the thousands, accurately mapping each of them manually is expensive and laborious. In this study, we applied deep learning, a form of artificial intelligence, to build computer models to automatically locate sinkholes from images created from elevation data. These models used the image segmentation technique to label every pixel in an image as either sinkhole or non-sinkhole. We used images of elevation, slope, elevation gradient, and shaded relief as inputs to models. Model results suggested that deep learning offered a viable way to automatically locate sinkholes from elevation data. In particular, models using elevation gradient information performed the best. We also evaluated aerial imagery to train the models and found that aerial images were not useful in training deep learning models for sinkhole identification.

1 Introduction

Approximately 15 % of the world's ice-free land surface is underlain by carbonate rocks, and a recent estimate suggests that 1.3 billion people lived on these rocks in 2019 globally (Goldscheider et al., 2020). Almost all the carbonate rock areas have developed karst, a landscape characterized by sinkholes, sinking streams, springs, and caves (Monroe, 1970). Sinkholes are the most abundant surficial features in karst and are formed when soil or other overburden material subsides or collapses into subsurface voids created by the dissolution of soluble rocks. Hydrologically, sinkholes collect rainfall and drain it internally to the subsurface, serving as fast recharge routes for karst aquifers. More commonly, sinkholes are known as a geohazard. Sinkholes, especially suddenly occurring collapse sinkholes, cause significant damage to homes, buildings, highways, and other infrastructure (Weary, 2015). Therefore, knowledge of detailed distribution and characteristics of sinkholes is essential for protecting karst aquifers and mitigating sinkhole-related hazards in karst areas.

Most sinkholes appear on the land surface as depressions or cover-collapses, and are traditionally mapped from topographic maps. In the United States, the topographic maps used for mapping sinkholes are low in resolution and were mostly created prior to the 1970s. As a result, many small or newly formed sinkholes were missed (Zhu et al., 2014). The increasing availability of high-accuracy and high-resolution remote sensing data, especially LiDAR (Light Detection and Ranging), has led to the discovery of significantly more sinkholes in many karst areas (Rahimi & Alexander, 2013; Zhu et al., 2014; Wu et al., 2016, e.g.). For instance, using LiDAR data, Zhu et al. (2014) found three times more sinkholes than previously identified from topographic maps in Floyds Fork watershed, central Kentucky. Inconveniently, sinkholes are not the only surficial features showing as depressions on the surface. Many nature features such as stream channels, meander cutoffs, and more commonly man-made structures such as farm ponds, road culverts, and swimming pools, also appear as depressions. Processing LiDAR data to locate sinkholes also extracts these non-sinkhole depression features, so separating sinkholes from non-sinkhole depressions becomes a necessary step. While this step can be done using a manual process of visual inspection and classification of each depression (Zhu et al., 2014), the manual process can be laborious and time-consuming because 1) thousands of surface depressions can be extracted from LiDAR data in a small area and 2) sinkholes are an only small portion of the extracted depressions. Finding efficient methods to separate sinkholes from other depressions remains a challenge.

Machine learning is a branch of artificial intelligence that constructs computer-based systems that improve automatically through training experience (Jordan & Mitchell, 2015). Machine learning methods have been applied to automatically identify sinkholes or evaluate sinkhole hazards (Miao et al., 2013; Zhu & Pierskalla, 2016; Taheri et al., 2019; Kim et al., 2019; Zhu et al., 2020, e.g.). These studies applied conventional, or shallow machine learning methods that rely on feature datasets to train because the conventional machine learning methods have limited ability to process raw data (LeCun et al., 2015). These feature datasets are created by extracting feature variables deemed relevant to a problem of interest from available data; therefore, the extracted variables are often subjective, depending on a researcher's experience and their understanding of the original data. For instance, Kim et al. (2019) used topographic variables, such as elevation, aspect, and curvature, to train a logistic regression sinkhole model. Zhu et al. (2020) used morphometric variables of the depressions, such as surface area, depth, and circularity, to train machine learning methods for identifying sinkholes from surface depressions that were previously extracted by processing LiDAR elevation data. In a sense, the machine learning methods applied in Zhu et al. (2020) did not directly learn from elevation data. Deep learning methods, on the other hand, can directly learn from images, text, videos, and sounds through multiple processing layers to learn representations with multiple levels of abstraction (LeCun et al., 2015). Convolutional neural networks (CNNs) are the most widely used deep learning methods for image classification. Because of their tremendous success in classifying conventional photographic images, CNNs have also been applied for landscape classifications recently (Hu et al., 2015; Buscombe & Ritchie, 2018; Li et al., 2020, e.g.). In particular, Vu et al. (2020) trained sinkhole detection CNN models using thermal images for eight manually dug holes. These studies used mainly multispectral remote sensing images in which different landscape features are easily discernible. Elevation data are not commonly used for landscape classification. Li et al. (2020) found remote sensing images provide best information in loess landform classification while digital elevation models can help distinguish ridges and hills. Sinkholes, on the other hand, are small-scale topographic features that are difficult to see from multispectral remote sensing images. In this study, we trained a convolutional neural network to perform image segmentation on LiDAR elevation data and their derivative images to locate sinkholes. We also tested multispectral remote sensing images in finding sinkholes.

There are many types of CNNs that can be used for image segmentation (Minaee et al., 2021). We select a commonly used architecture known as U-Net (Ronneberger et

al., 2015), which has been shown to work well across a broad range of tasks. U-net has also been applied to detect sinkholes. For instance, Vu et al. (2020) used U-net as a weak but fast classifier to find areas with high probability of sinkholes. Our focus in this work is on evaluating various ways of pre-processing the input data. This includes whether or not a particular input modality is included and different forms of input pre-processing and standardization.

2 Study Area and Input Images

The study area is located in the Inner Bluegrass Region of central Kentucky, a mature karst environment developed on the Middle Ordovician Lexington Limestone (Cressman & Peterson, 1986). The region features gently rolling topography with numerous sinkholes across the landscape (Paylor & Currens, 2004). The climate is temperate with an average annual temperature of 13.0 °C and an average precipitation of 1170 mm. The land use is mainly agricultural with some urban and suburban regions (University of Kentucky College of Agriculture Food and the Environment, 2011). Sinkholes in the region have been mapped from LiDAR data (Kentucky Geological Survey, n.d.). In this study, we selected a rectangular area of 625 km^2 in the region to generate input images (Figure 1). This area covers part of Fayette, Franklin, Scott, and Woodford Counties and is 21.74 km long in the x direction (west-east) and 28.83 km long in the y direction (south-north). There are 2177 sinkholes mapped in the rectangular area.

The input data for the deep learning models consist of three images: a LiDAR-derived digital elevation model (DEM) image, an aerial image, and a binary label image (Figure 2). All the images are 14268 x 18851 pixels and each pixel is 1.524 m x 1.524 m (5 ft x 5 ft) in size. The DEM and the aerial image are downloaded from Kentucky's Elevation Data and Aerial Photography Program (KyFromAbove, n.d.). The DEM image has one channel with values ranging from 158 m – 308 m (518 ft – 1003 ft). The aerial image is a four channel National Agriculture Imagery Program (NAIP) image from 2018. The original NAIP image is in 0.610 m (2 ft) resolution and is resampled to 1.524 m (5 ft) resolution to have the same resolution as other input images. The binary label image was created using the sinkhole mapping results (Kentucky Geological Survey, n.d.). Any pixel located inside a sinkhole is valued as 1 and as 0 otherwise. Note that only 2 % of pixels are valued as 1 because even though sinkholes are widespread, their areas are so small that they only occupy a small fraction of the total land surface.

In addition to directly using the DEM image as input, we also prepared three images derived from the elevation data: a slope image, a DEM gradient image, and a shaded relief image (Figure 3). The slope image was created from the DEM using ArcGIS Pro's Planar Slope method, which calculates the slope as the maximum rate of change in elevation from a cell to its immediate neighbors. The slope image has one channel with values ranging from 0 – 85 degrees. The slope as calculated in ArcGIS Pro is the maximum slope among the neighboring cells. The DEM gradient image is calculated using central difference and it has two channels, one for elevation gradient in the x direction and the other for elevation gradient in the y direction. Therefore, the two-channel DEM gradient image preserves directional slope information otherwise lost in the traditional slope image. The shaded relief image is a single illumination hillshade with an azimuth of 315 degrees and an altitude of 45 degrees. The shaded relief image is prepared as an RGB image with three channels. We created the shaded relief image because sinkholes are highly visible on the shaded relief of DEMs (Zhu et al., 2014).

3 Methods

There are several formulations for the task of image recognition. Image classification is the task of assigning one label for the entire image. On the other hand, image segmentation is the task of assigning a class label to every pixel. While image segmenta-

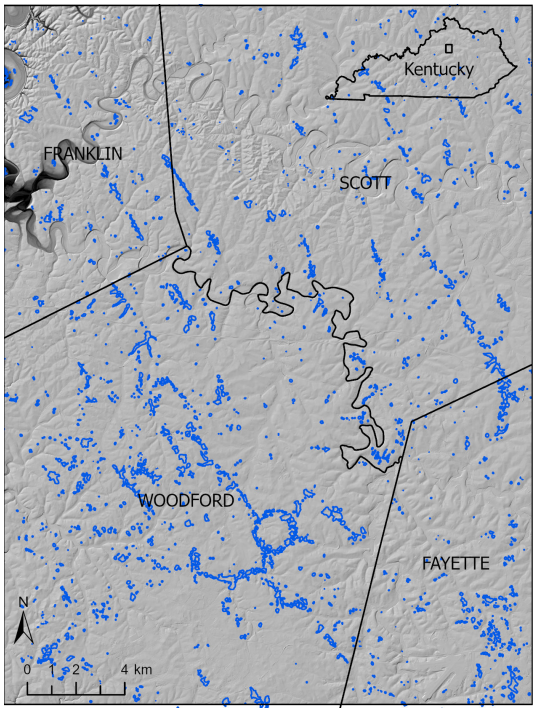
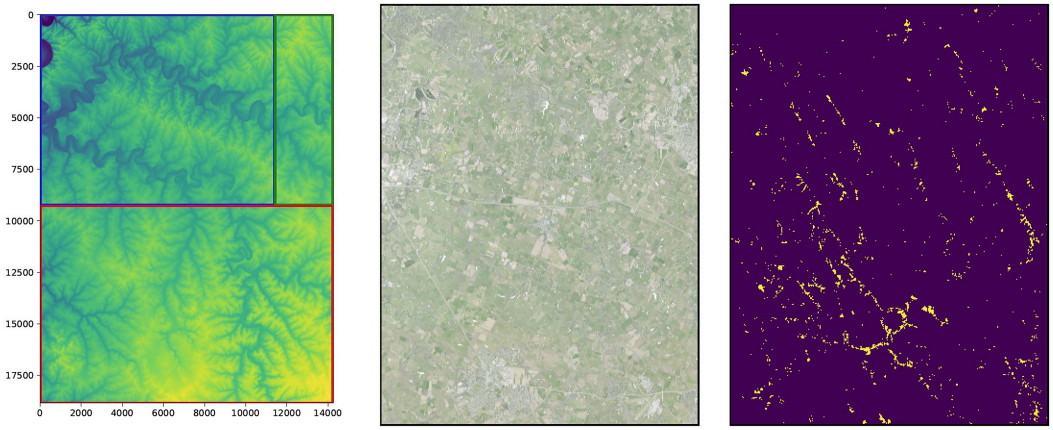
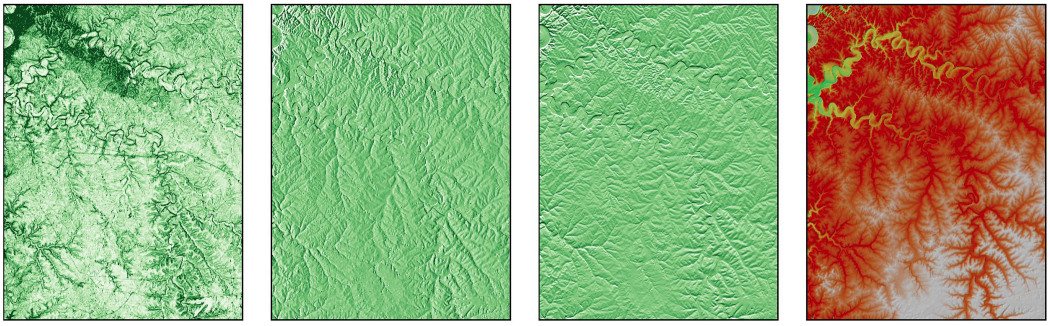


Figure 1. Study area. Blue lines depict mapped sinkholes



(a) DEM (b) NAIP (c) Sinkhole Label

Figure 2. Input data: (a) DEM, (b) NAIP, and (c) sinkhole label. Data splits are illustrated in the DEM image: training set in blue, validation set in green, and test set in red. Axis labels on (a) DEM are in pixels.



(a) Slope (b) DEM gradient X (c) DEM Gradient Y (d) Shaded Relief

Figure 3. Images derived from DEM data: (a) Slope, (b) DEM gradient X direction, (c) DEM gradient Y direction, and (d) Shaded relief.

tion provides more detailed output, this formulation requires more labeling effort. Since we have dense labels for sinkholes, derived from LiDAR, we formulate the task as a segmentation problem. Our task is a binary segmentation task that classifies every pixel as sinkhole or non-sinkhole. We use convolutional neural networks (CNNs) for the task of image segmentation. The input to the segmentation model is a smaller patch of size 400 x 400 pixels. However, the model can be used for arbitrarily large regions, as shown in Figure 1, by feeding a batch of such patches to the model and stitching the results back.

3.1 Data Normalization

Images often are stored in various formats resulting in different input ranges. For instance, our DEM image has a range of 518 – 1003 (elevation in ft) while our shaded relief image and aerial image has a range of 0 – 255 in each channel. It is a standard practice to normalize pixel values to a small range to improve training by gradient descent (LeCun et al., 2012). We evaluated three alternative normalization methods:

- Global [0, 1] normalization: we normalized all values in the range [0, 1] based on the maximum and minimum values. This normalization was done based on the statistics of the training data.
- Gaussian whitening: for an input channel x, the normalized value was given by: $\hat{x} = \frac{x - \mu}{\sigma}$ where μ and σ are mean and standard deviation of the training data.
- Instance normalization: we normalized every patch separately into the range [0, 1]. As opposed to the Global normalization, in this case the normalization was performed on every patch.

Figure 4 shows a visualization of the three normalization methods on the DEM.

3.2 Network Architecture

Many CNN architectures have been proposed for image segmentation ranging from FCN (Long et al., 2015) to DeepLabV3+ (Chen et al., 2018) and HR-Net (Wang et al., 2020). While these networks achieve state-of-the-art results for urban scenes and indoor images, for medical and remote sensing images, U-Net (Ronneberger et al., 2015) often performs better. We modified the U-Net for our task of binary segmentation based on the number of input channels we have for different input image types. In our case, the size of the output is the same as the input, which is not the case in the original U-Net

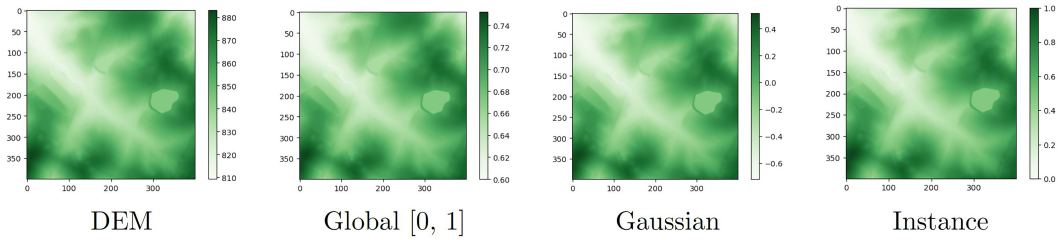


Figure 4. Visualization of three normalization methods on a DEM patch. Axis labels are in pixels. Note that the different ranges are defined for the colormap of each image.

model. The output layer has two channels: one for sinkhole pixels and the other for non-sinkhole pixels.

The network architecture is shown in Figure 5. The input is patch of spatial size I (400 x 400 pixels in our case). There are several convolutional layers, each having a filter size of 3×3, followed by a *BatchNorm* layer (Ioffe & Szegedy, 2015). For every convolutional layer, there are different numbers of filters - in our implementation, we use 1/4 the number of filters than the original U-Net (Ronneberger et al., 2015). For example, the first block has two convolutional layers, each with 16 filters. The left half of the network, also referred to as the encoder, feature maps are reduced in spatial size by applying *MaxPool* (Nagi et al., 2011). The feature maps are reduced to the size $\frac{I}{16}$, i.e., 1/16th the spatial size of input patch I, in the bottleneck section, shown in the middle in Figure 5. The right side of the network, also known as the decoder, increases the spatial size of feature maps. In the decoder, at each level, the feature maps from the encoder are copied over as input, as shown by arrows on the top. All layers use *ReLU* (Nair & Hinton, 2010) as the activation function except the last layer. In the last layer, we have a two-channel output, one for sinkhole and the other for non-sinkhole. We apply the *Softmax* activation function that results in a proper probabilistic prediction (also called soft prediction): the score for sinkholes is $\hat{y}$ and the score for non-sinkholes prediction is $1-\hat{y}$.

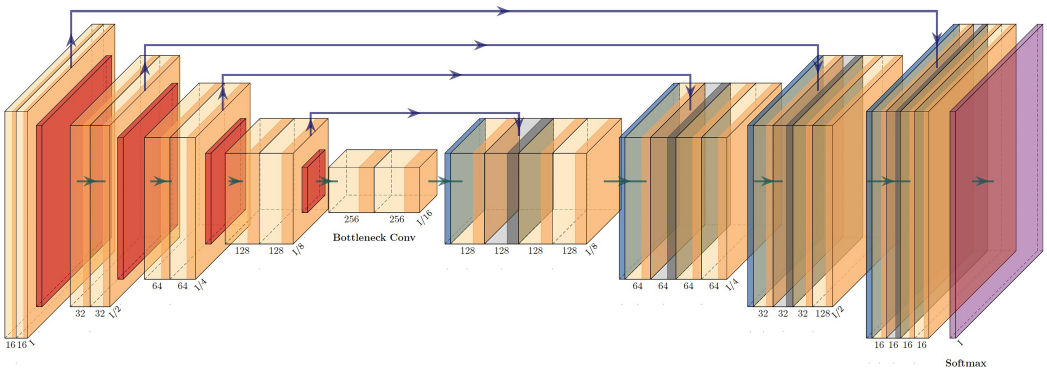


Figure 5. The U-Net architecture used for sinkhole segmentation. Here, I denotes the spatial size of the input image patch. Visualization generated using (Iqbal, 2018).

3.3 Loss Function

For training, it is common to use the cross entropy loss function

$$l(\hat{y}, y) = -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y}), \tag{1}$$

where $\hat{y}$ is the prediction, y is the target label indicating non-sinkhole (0) or sinkhole (1) pixel. The sinkhole label image is highly imbalanced with 98 % pixels belonging to the non-sinkhole category and only 2 % belonging to the sinkhole category. A network treating both categories equally will result in a trivial local minimum such that the network only predicts the majority class (non-sinkhole region) and gets a very low loss. To address this, we use different loss weighting factors for non-sinkhole and sinkhole pixels:

$$l(\hat{y}, y) = -w_s y \log(\hat{y}) - w_n (1 - y) \log(1 - \hat{y}), \tag{2}$$

where w_s and w_n are the loss weights for sinkhole and non-sinkhole pixels, respectively. We use a higher weight for sinkhole, w_s to encourage the network to make better sinkhole predictions. We found that using $w_s = 1.0$ and $w_n = 0.05$ gives better results than other weight ratios.

3.4 Implementation Details

We implemented our approach using PyTorch (Paszke et al., 2019), which is a freely available software library. Please see `https://mvrl.github.io/SinkSeg/` for the source code, installation instructions, access to the image dataset, and scripts for training and inference. The image dataset can be also downloaded directly from `https://doi.org/10.5281/zenodo.5789436`. For training and evaluation, we used a patch of size 400 x 400 pixels. We randomly cropped patches from training images as a data augmentation strategy because it can generate a large number of unique examples for training. For validation and testing, we made non-overlapping patches that covered the respective region completely. We can run our trained model on arbitrarily large regions by sequentially feeding batches of non-overlapping smaller patches to the model and stitching the results back. In total, we had 644 patches for training, 161 for validation, and 840 for testing. For training, we used a batch size of 14 and trained all models for 100 epochs using an L_2 regularization of 1×10^{-6}. We set the initial learning rate of 5×10^{-4} and reduced the learning rate by a factor of 0.9 after every 3 epochs. During training, we saved the model checkpoint with the lowest loss on the validation set as the best model and used that for evaluation. Training one epoch (of the area approximately 239 km^2) of our model took around 14 seconds on a single NVIDIA Titan RTX GPU. A trained model can be used for inference on validation data (having an area around 60 km^2) in 2 seconds and on the test set (having an area around 312 km^2) in 7 seconds using the same GPU.

4 Evaluation and Results

Using the five different types of input images and three normalization methods, we trained 15 image segmentation sinkhole identification models. We also trained three additional models with non-normalized images of DEM, slope, and gradients. We did not train non-normalized shaded relief and aerial images because they are regular RGB images and normalization is standard for these images in deep learning. We then evaluated and compared these 18 models to find the best data and normalization method as described below.

4.1 Evaluation Metrics

We report several commonly used metrics for image segmentation (Long et al., 2015) including intersection over union (IoU), mean accuracy, average precision, and area under the ROC curve (AUC). As there is a severe class imbalance and we are primarily interested in the identification of sinkholes, we report sinkhole IoU separately as well.

Intersection over union, also known as the Jaccard index, can be written as:

$$IoU(y, \hat{y}) = \frac{y \cap \hat{y}}{y \cup \hat{y}} \tag{3}$$

where $y \cap \hat{y}$ is the intersection (overlap) and $y \cup \hat{y}$ is the union of prediction and true label. Accuracy is given as:

$$Acc(y, \hat{y}) = \frac{TP + TN}{T} \tag{4}$$

where TP is number of true positive, TN is number of true negative, and T is the total number of pixels. We show receiver operating characteristics curve (ROC) as well two methods of summarizing the curve, area under the ROC curve and average precision. Average precision is given as

$$AP = \sum_i (R_i - R_{i-1} P_i) \tag{5}$$

where P_i and R_i are precision and recall computed at the threshold value i. Precision (P) and recall (R) are given as:

$$P = \frac{TP}{TP + FP} \qquad R = \frac{TP}{TP + FN} \tag{6}$$

where TP, TN, FP, and FN are numbers of true positive, true negative, false positive, and false negative, respectively.

4.2 Results

The model can predict the probability of each pixel being part of a sinkhole ($\hat{y}$) in an image. However, in practice, we need to make a binary prediction for whether or not a pixel is within a sinkhole if $\hat{y} > t$ for the threshold t. For all models, we find the optimum threshold that gives the highest sinkhole IoU on the validation set. We use this threshold to compute metrics of the respective model on the test set. Figure 6 shows how sinkhole IoU varies with different thresholds for the model using the elevation gradient image with Gaussian normalization. We can see that for this model, the optimum threshold is 0.9, as shown in Figure 6. A visualization of varying binary predictions as the threshold changes is shown in Figure 7 for the validation set.

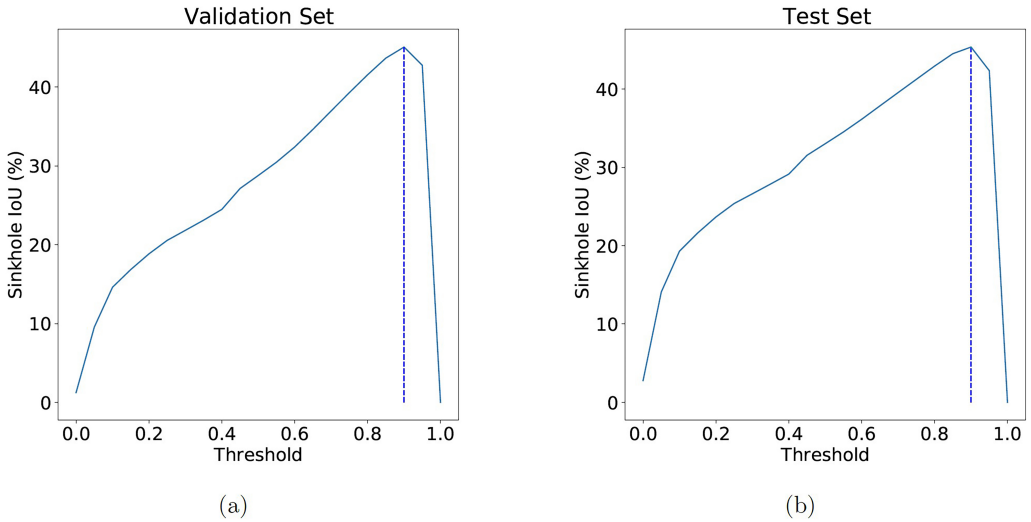


Figure 6. Analysis of varying threshold on (a) the validation set and (b) the test set.

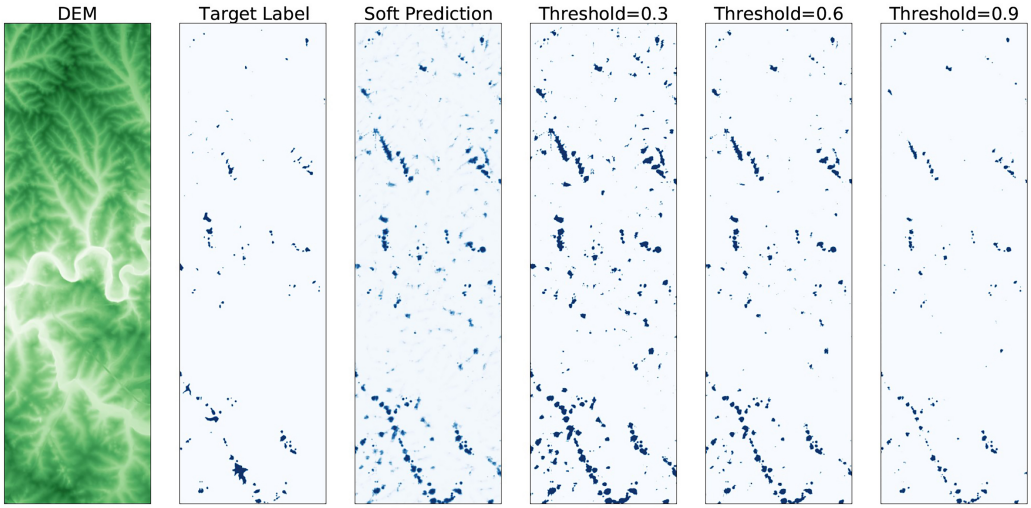
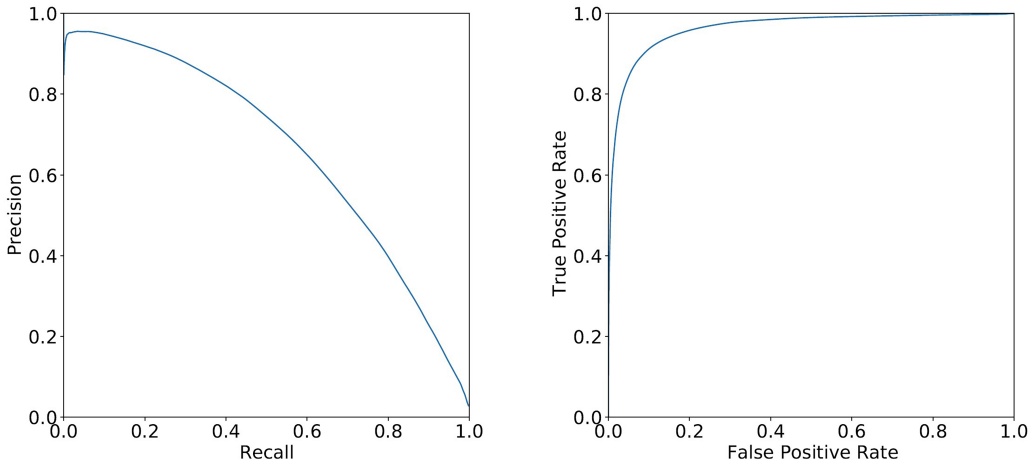


Figure 7. Qualitative results on the full validation set for several threshold values. We also show soft predictions without applying any threshold.



(a) (b)

Figure 8. Test set evaluation: (a) Precision recall curve and (b) receiver operating characteristics curve (ROC).

After selecting the optimum threshold, we calculated the five metrics introduced in section 4.1 on the test set for each model. Among the five metrics, sinkhole IoU, mean IoU, and mean accuracy were calculated using the optimum threshold while average precision and AUC were integrated over the entire threshold range. Figure 8 shows a precision-recall curve used for calculating the average precision and a receiver operating characteristic curve for calculating AUC for the model using the elevation gradient image with Gaussian normalization.

Although all the five metrics were used in comparing the models, we selected sinkhole IoU as the indicator metric because IoU is a widely used metric in evaluating image segmentation models and the other four metrics are consistent with sinkhole IoU. Comparing the metrics of all the 18 models (Tables 1, 2, 3, 4, 5), the model using ele-

vation gradient with Gaussian normalization performed the best, with a sinkhole IoU of 45.38 %, followed by the model using elevation gradient without normalization, which achieved a sinkhole IoU of 43.61 % (Table 3). Other models that achieved sinkhole IoU above 40 % were elevation gradient with Global normalization (Table 3) and DEM with Instance normalization (Table 1). In contrast, models using NAIP image performed the worst with sinkhole IoU values below 3 % in all normalization methods (Table 5). The models using DEM slope (Table 2) and the models using shaded relief image (Table 4) were better than the models using the NAIP image. However, with their sinkhole IoUs in the range of 20 % – 30 %, these models can only be considered to be moderately successful.

Normalization	Sinkhole IoU (%)	Mean IoU (%)	Mean Accuracy (%)	Avg. Precision (%)	AUC
None	19.18	57.43	66.94	24.33	0.8627
Global [0, 1]	25.45	60.80	72.13	29.14	0.8968
Gaussian	23.47	60.29	65.31	32.52	0.7954
Instance	40.83	69.15	80.02	60.21	0.9508

Table 1. Evaluation metrics of image segmentation models using DEM as input.

Normalization	Sinkhole IoU (%)	Mean IoU (%)	Mean Accuracy (%)	Avg. Precision (%)	AUC
None	27.55	62.24	69.90	40.31	0.9076
Global [0, 1]	25.12	60.73	70.77	36.54	0.8987
Gaussian	26.57	61.41	74.92	40.83	0.9044
Instance	27.42	62.20	69.52	40.18	0.8946

Table 2. Evaluation metrics of image segmentation models using DEM slope as input.

Normalization	Sinkhole IoU (%)	Mean IoU (%)	Mean Accuracy (%)	Avg. Precision (%)	AUC
None	43.61	70.68	80.03	65.39	0.9610
Global [0, 1]	41.26	69.36	80.89	60.25	0.9513
Gaussian	45.38	71.65	79.87	66.29	0.9645
Instance	26.35	60.94	76.62	39.15	0.3915

Table 3. Evaluation metrics of image segmentation models using DEM gradient as input.

The results of the best performing model, elevation gradient with Gaussian normalization, on the test set are illustrated in Figure 9. The figure shows prediction of the model with the optimum threshold of 0.9 as well as predictions for thresholds 0.3 and 0.6 and the soft prediction. The soft prediction was the actual prediction result, which was the probability of each pixel being part of a sinkhole. The soft prediction and results with three different thresholds largely matched the sinkhole label image in pattern, but the one with the 0.9 threshold closely resembled the sinkhole label.

Normalization	Sinkhole IoU (%)	Mean IoU (%)	Mean Accuracy (%)	Avg. Precision (%)	AUC
Global [0,1]	26.05	60.91	74.97	40.00	0.9149
Gaussian	23.18	59.29	72.00	34.78	0.8859
Instance	21.32	58.47	69.63	29.47	0.8486

Table 4. Evaluation metrics of image segmentation models using shaded relief as input.

Normalization	Sinkhole IoU (%)	Mean IoU (%)	Mean Accuracy (%)	Avg. Precision (%)	AUC
Global [0,1]	2.97	5.58	53.19	2.99	0.5473
Gaussian	2.90	4.05	52.02	3.40	0.5882
Instance	2.98	12.81	53.01	3.19	0.5537

Table 5. Evaluation metrics of image segmentation models using NAIP image as input.

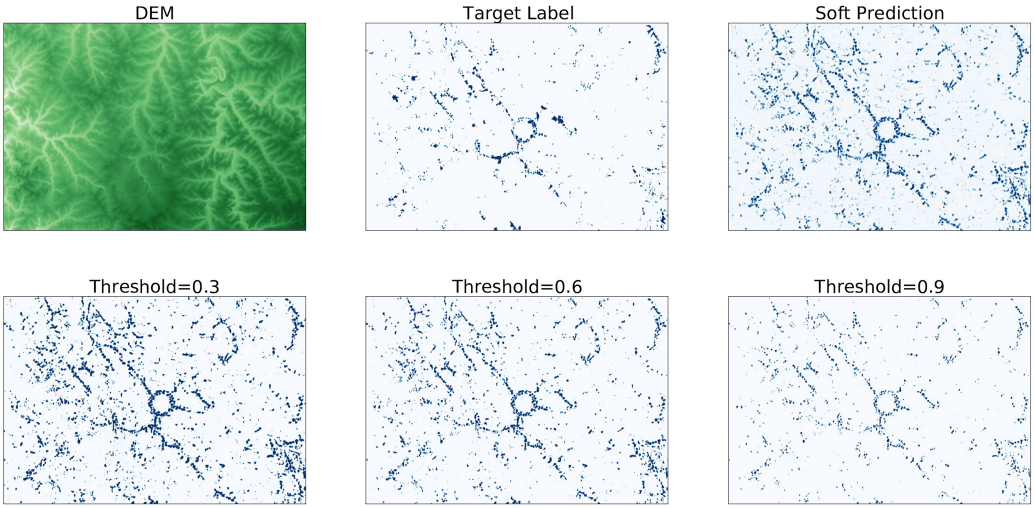


Figure 9. Test set results. We show qualitative results on the full test set for several threshold values and soft predictions without applying any threshold.

A close view of the prediction results is shown in Figure 10. It shows results from seven randomly selected patches on the test data. Each row in Figure 10 shows a single patch that is passed through the network. Overall, the prediction matched the true sinkhole label quite well. However, there were some mismatches, as shown in the last three rows of Figure 10.

5 Discussion

We trained 15 sinkhole segmentation models using images created from LiDAR-derived digital elevation data. We found that, with proper data pre-processing and normalization, the CNN-based image segmentation method can extract sufficient information from the LiDAR-derived elevation data to build decent models to automatically identify sinkholes. However, when the raw DEM data were directly used without normalization, the model performed poorly. The raw DEM data had the largest range of val-

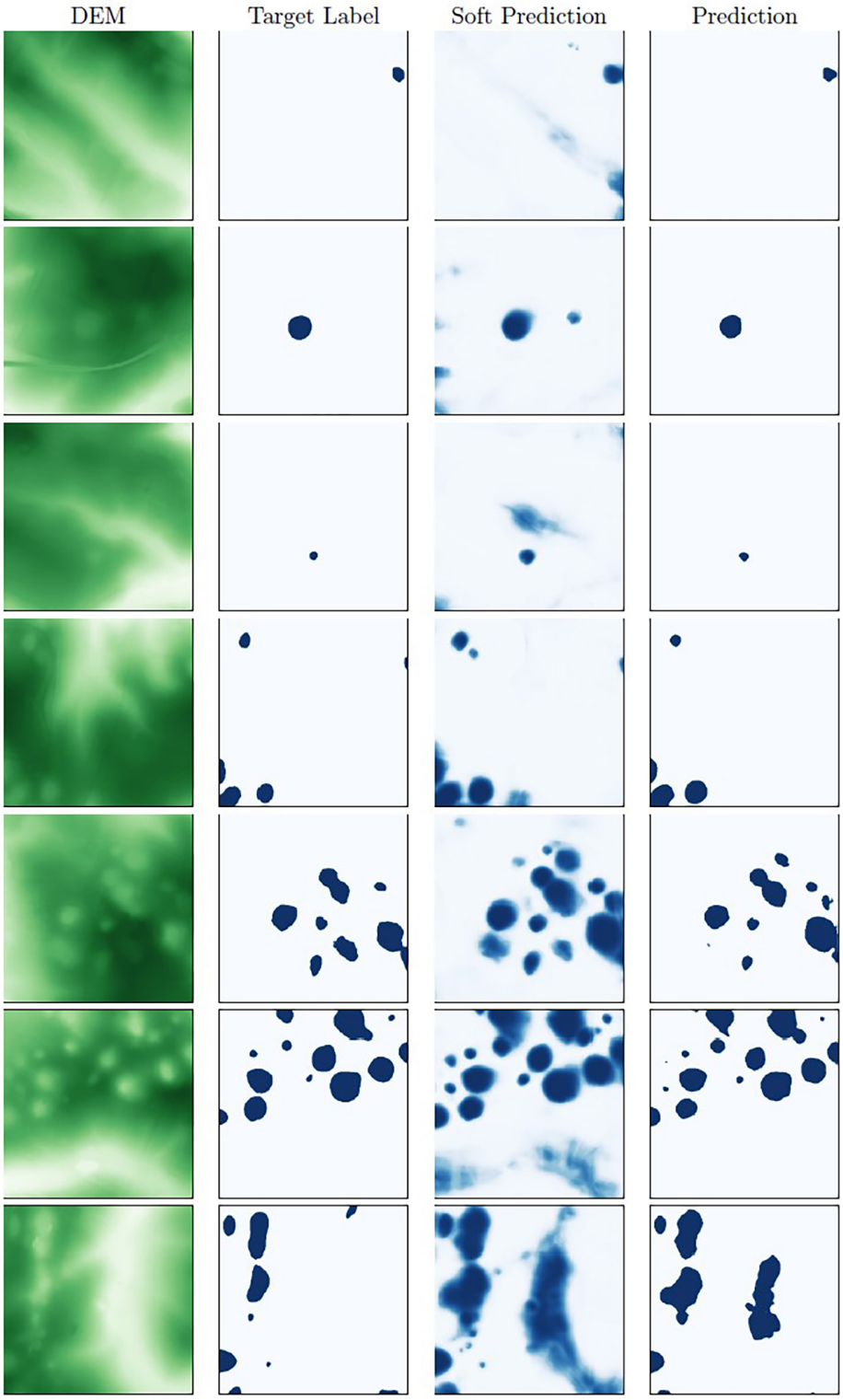


Figure 10. Qualitative results on patches from the unseen test set.

ues (518-1003) among all the inputs and there is an overall trend in elevation where the elevation is highest in the southeast and dips into the northwest. We speculate that the large range and the trend create a difficulty to translate the results from the training area (northwest region) to the test area (south region). Both Global and Gaussian normalizations reduced the overall range of the data, but the overall trend remained. This is evident as both normalizations only slightly improved the model. On the other hand, the Instance normalization reduced the range and also removed the overall trend, therefore provided an additional improvement.

Models trained on DEM slope with and without normalization yielded similar poor results with sinkhole IoUs of around 25 % – 27 % (Table 2). The Planar Slope method combines the slope values in x and y directions into one value, leading to possible information loss. To test if the information loss attributes to the poor performance, we created an image with two channels, one for elevation gradient in the x direction and the other for the y direction. Models trained on this 2-channel DEM gradient image (Table 3) performed much better than the models trained on DEM slope. Using the 2-channel gradient image, the model without normalization achieved a sinkhole IoU of 43.61 % and Gaussian normalization slightly improved the model with a sinkhole IoU of 45.38 %.

In the models using the raw DEM as input, all normalization methods improved model performance (Table 1). However, these normalization methods did not yield noticeable improvements when slope data or elevation gradient were used as inputs. For models using the slope data, all normalization methods had little impact (Table 2). For models using the elevation gradient data, Gaussian and Global normalizations had little impact whereas the Instance normalization decreased sinkhole IoU to 26.35 % (Table 3). The slope and DEM gradient data removed the overall trend in the DEM and converted elevation values to a smaller range of 0 – 90 degrees for the 1-channel image and a smaller range for the elevation gradient image, which might explain why all the additional normalization methods did not improve the model. The poor performance of Instance normalization on the elevation gradient data was a stark contrast to the method's improvement on models using the raw DEM data. In normalizing each patch to a range [0, 1], the Instance normalization requires different scaling factors for every patch, therefore lacking consistency across the entire training image. As a result, the Instance normalization can be more sensitive to noise in the DEM.

We find that these normalization choices result in large difference in final system performance. We expect that our performance metrics could be further improved by using different segmentation architectures and further tuning of training hyperparameters. To facilitate future studies in this regard, we make available code to facilitate easy training and inference.

Metrics of models using the shaded relief of the DEM are shown in Table 4. The shaded relief image is a three-channel color image. For color images, Global normalization and Gaussian normalization are universally used in machine learning. Consequently, we did not run the model from shaded relief without normalization. The results of the three normalization methods were quite similar and sinkhole IoUs ranged from 21 % to 26 %. The results were also similar to several models using raw DEM and the models using the slope data. It suggests that a shaded relief image does not provide additional information from the raw DEM to the segmentation models despite it being useful for manual visual inspection.

Results of the three sinkhole segmentation models using NAIP imagery as input showed the aerial image provided weak cues for segmenting sinkholes (Table 5). For all normalization methods, the models could not correctly identify sinkholes and achieved sinkhole IoUs of merely 2.98 %. Since most sinkholes cannot be seen directly on aerial images such as NAIP, models using NAIP images alone perform poorly. However, visible surface features on aerial images, such as tree clusters, ponds, roads, and residen-

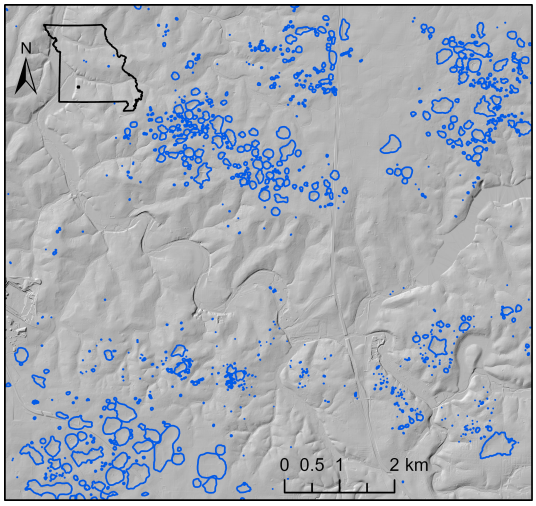


Figure 11. Out-of-distribution evaluation area in Missouri. Blue lines depict mapped sinkholes

tial houses, can be used to help separate sinkholes from other forms of surface depression (Zhu et al., 2014). In future research, we will explore methods that combine elevation data and aerial images to improve their ability to segment sinkholes.

Deep learning models trained in one area can perform unexpectedly when applied to a geographic region with different landscape characteristics. This issue is an example of so-called out-of-distribution problems commonly encountered in deep learning. To evaluate if the models trained using data from Kentucky are applicable to other karst regions, we applied our best performing sinkhole segmentation model to the Springfield Plateau in southwest Missouri, USA. The Springfield Plateau is a prominent karst region with abundant karst features, such as sinkholes, caves, and springs. The region is underlain by the Mississippian Burlington and Keokuk Limestones (Martin & Pratt, 1991), which are roughly 100 million years younger than the Lexington Limestone underlying the area in Kentucky where our models were trained. We selected a rectangular area of 86.4 km^2 in Greene County in the Springfield Plateau (Figure 11). The area is 9.6 km in the x direction (west-east) and 9 km in the y direction (south-north). LiDAR DEM of 1 m resolution were obtained from MSDIS (Missouri Spatial Data Information Service, n.d.) and were resampled to 1.524m (5 ft) to match the resolution of the images used for model training. The range of elevations in this area is 90 m (306 - 396 m) whereas the elevation range in the training area is 150 m (158 - 308 m). A total of 1021 sinkholes have been mapped in the area (City of Springfield, Missouri, n.d.) and were used to create a binary label image to evaluate model prediction results.

Our best-performing model was the one that used DEM gradients as inputs with Gaussian normalization. The prediction results of applying this model to the Missouri area show that the model predicted sinkhole areas closely matched with mapped sinkhole areas (Figure 12). The evaluation metrics (Table 6) confirmed the model's good performance in the new area. The sinkhkole IoU was 42.38 %, which was only slightly lower than the Kentucky test sinkhole IoU of 45.38 % (Table 3). Note that while the threshold producing the highest sinkhole IoU for Kentucky was 0.9, the threshold for the highest sinkhole IoU for the Missouri area was 0.5. The difference in optimal thresholds appeared to be corresponding to different criteria in mapping sinkholes. In Kentucky, surface depression features less than 46.45 m^2 (500 ft^2) were excluded from consideration for sinkholes (Zhu et al., 2014), but approximately 15 % of the mapped sinkholes in the

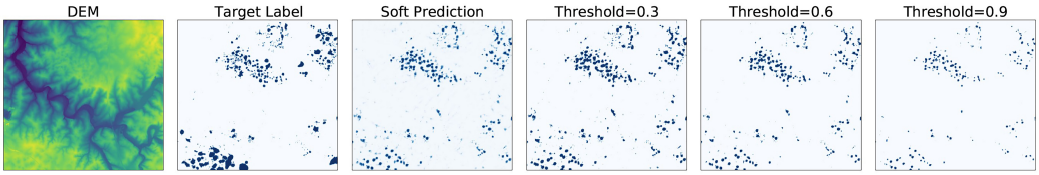


Figure 12. Qualitative results on the data from the Missouri Area. These results are generated from the model trained on DEM gradients of the Kentucky data and no information from the Missouri area is provided to train the model.

Missouri area were less than the minimum area of 46.45 m^2 used in Kentucky. Even though our model generalizes well for a different region, an appropriate threshold separating sinkholes from non-sinkholes requires existing sinkhole data for the region. Because we had a sinkhole dataset for the Missouri area, we were able to find an optimal threshold. If the trained image segmentation model is used to predict sinkholes to an area where sinkholes are not mapped, we suggest that a small sub-area should be mapped manually so that a suitable threshold can be determined.

Normalization	Sinkhole IoU (%)	Mean IoU (%)	Mean Accuracy (%)	Avg. Precision (%)	AUC
Gaussian	42.38	68.85	77.01	61.78	0.8665

Table 6. Evaluation metrics of applying the best image segmentation model to the Missouri region. The first three metrics were calculated with a threshold of 0.5.

6 Conclusions

Sinkholes are the most prevalent topographic features in karst areas worldwide. Understanding their occurrence and characteristics is critical for studying karst aquifers and mitigating sinkhole-related hazards. In this study, we explored image segmentation for automatically locating and delineating sinkholes from high-accuracy, high-resolution LiDAR DEMs. We trained convolutional neural network models based on the U-Net architecture and performed image segmentation to label each pixel in an image as sinkhole or non-sinkhole. We evaluated how three normalization methods impacted model performance. Furthermore, we explored the usefulness of aerial images as input for training deep learning sinkhole identification models. We also applied our model to a karst area in Missouri to test our model's out-of-distribution generalization. Our study suggests:

- Deep learning-based image segmentation is a promising tool to identify karst sinkholes directly from DEMs.
- Slope and DEM gradient data provide better information than the raw DEM in identifying sinkholes. Shaded relief of DEMs, on the other hand, does not enhance model performance.
- While Global and Gaussian normalization methods have the potential to improve deep learning models, Instance normalization should be used with caution as it can worsen model performance.

- The sinkhole segmentation models trained using data from Kentucky show good out-of-distribution generalization and can potentially be applied to other karst areas.
- Aerial images alone did not prove to be useful as input to the proposed segmentation model.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. (IIS-1553116). The second author is supported by the National Science Foundation Grant No. (EAR-1933779). We thank Nicole Wong and Aram Ansary Ogholbake for testing the code. We appreciate Dr. William Odom and Dr. Daniel Buscombe for their instructive reviews, which have greatly improved the manuscript.

Data Availability Statement

Model source code, installation instructions, and scripts for training and inference are in Github at `https://mvrl.github.io/SinkSeg/`. The image dataset used in the model is deposited at `https://doi.org/10.5281/zenodo.5789436`. Data sources used to derive the image dataset are available in these in-text data citation references: aerial imagery from the National Agriculture Imagery Program (KyFromAbove, n.d.), [public domain]; digital elevation model derived from LiDAR data (KyFromAbove, n.d.), [public domain]; binary label image derived from Kentucky LiDAR-derived sinkholes (Kentucky Geological Survey, n.d.), [public domain]; digital elevation model for Missouri from Missouri Spatial Data Information Service (Missouri Spatial Data Information Service, n.d.), [public domain]; and sinkhole data for Greene County, Missouri from City of Springfield, Missouri (City of Springfield, Missouri, n.d.), [public domain].

References

Buscombe, D., & Ritchie, A. (2018). Landscape classification with deep neural networks. *Geosciences*, *8*(7), 244. doi: 10.3390/geosciences8070244

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the european conference on computer vision (eccv)* (pp. 801–818).

City of Springfield, Missouri. (n.d.). *Sinkhole boundaries for Greene County, Missouri.* Author. Retrieved from `https://gisdata-cosmo.opendata.arcgis.com/datasets/COSMO::sinkhole-boundaries/about` (accessed: 06.25.2021)

Cressman, E. R., & Peterson, W. L. (1986). Ordovician system. In R. C. McDowell (Ed.), *The geology of Kentucky: a text to accompany the geologic map of Kentucky.* US Geological Survey. doi: 10.3133/pp1151h

Goldscheider, N., Chen, Z., Auler, A. S., Bakalowicz, M., Broda, S., Drew, D., ... Veni, G. (2020). Global distribution of carbonate rocks and karst water resources. *Hydrogeology Journal*, *28*(5), 1661-1677. doi: 10.1007/s10040-020-02139-5

Hu, F., Xia, G.-S., Hu, J., & Zhang, L. (2015). Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery. *Remote Sensing*, *7*(11), 14680–14707. doi: 10.3390/rs71114680

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning* (pp. 448–456).

Iqbal, H. (2018). *Harisiqbal88/plotneuralnet v1.0.0.* Zenodo. Retrieved from `https://zenodo.org/record/2526396` doi: 10.5281/ZENODO.2526396

Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, *349*(6245), 255–260. doi: 10.1126/science.aaa8415

Kentucky Geological Survey. (n.d.). *KGS Geologic Map Information Service.* Kentucky Geological Survey, University of Kentucky. Retrieved from `https://kgs.uky.edu/geomap/` (accessed: 06.10.2020)

Kim, Y. J., Nam, B. H., & Youn, H. (2019). Sinkhole detection and characterization using LiDAR-derived DEM with logistic regression. *Remote Sensing*, *11*(13), 1592. doi: 10.3390/rs11131592

KyFromAbove. (n.d.). *Kentucky's Elevation Data & Aerial Photography Program.* Commonwealth Office of Technology, Kentucky. Retrieved from `https://kyfromabove.ky.gov/` (accessed: 06.16.2020)

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. doi: 10.1038/nature14539

LeCun, Y., Bottou, L., Orr, G. B., & Müller, K.-R. (2012). Efficient backprop. In *Neural networks: Tricks of the trade.* Springer.

Li, S., Xiong, L., Tang, G., & Strobl, J. (2020). Deep learning-based approach for landform classification from integrated data sources of digital elevation model and imagery. *Geomorphology*, *354*, 107045. doi: 10.1016/j.geomorph.2020.107045

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 3431–3440).

Martin, J. A., & Pratt, W. P. (1991). *Geology and mineral-resource assessment of the springfield 1 degree x 2 degrees quadrangle, missouri, as appraised in september 1985* (Tech. Rep.). Retrieved from `https://doi.org/10.3133/b1942` doi: 10.3133/b1942

Miao, X., Qiu, X., Wu, S.-S., Luo, J., Gouzie, D. R., & Xie, H. (2013). Developing efficient procedures for automated sinkhole extraction from lidar DEMs. *Photogrammetric Engineering & Remote Sensing*, *79*(6), 545–554. doi: 10.14358/pers.79.6.545

Minaee, S., Boykov, Y. Y., Porikli, F., Plaza, A. J., Kehtarnavaz, N., & Terzopoulos, D. (2021). Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1-1. doi: 10.1109/TPAMI.2021.3059968

Missouri Spatial Data Information Service. (n.d.). *Missouri LiDAR Data.* the University of Missouri-Columbia. Retrieved from `https://msdis.missouri.edu/data/lidar/` (accessed: 06.25.2021)

Monroe, W. H. (1970). *A glossary of karst terminology* (- ed.; Tech. Rep.). (Report) doi: 10.3133/wsp1899K

Nagi, J., Ducatelle, F., Di Caro, G. A., Cireşan, D., Meier, U., Giusti, A., ... Gambardella, L. M. (2011). Max-pooling convolutional neural networks for vision-based hand gesture recognition. In *2011 ieee international conference on signal and image processing applications (icsipa)* (pp. 342–347).

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Icml.*

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... others (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems.*

Paylor, R., & Currens, J. C. (2004). *Royal springs karst groundwater travel time investigation* (Tech. Rep.). Lexington, KY: Kentucky Geological Survey. (A report prepared for Georgetown Municipal Water and Sewer Service, Lexington, KY)

Rahimi, M., & Alexander, C. (2013). Locating sinkholes in LiDAR coverage of a glacio-fluvial karst, Winona County, MN. In *Full proceedings of the thirteenth multidisciplinary conference on sinkholes and the engineering and environ-*

mental impacts of karst. National Cave and Karst Research Institute. doi: 10.5038/9780979542275.1158

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International conference on medical image computing and computer-assisted intervention* (pp. 234–241).

Taheri, K., Shahabi, H., Chapi, K., Shirzadi, A., Gutiérrez, F., & Khosravi, K. (2019). Sinkhole susceptibility mapping: A comparison between bayes-based machine learning algorithms. *Land Degradation & Development*, *30*(7), 730–745. doi: 10.1002/ldr.3255

University of Kentucky College of Agriculture Food and the Environment. (2011). *Cane run and royal spring watershed-based plan, version 5. epa project number c9994861-06.* Retrieved from `https://www.uky.edu/bae/sites/www.uky.edu.bae/files/Cane_Run_WBP_2011.pdf` (accessed: 02.16.2021)

Vu, H. N., Pham, C., Dung, N. M., & Ro, S. (2020). Detecting and tracking sinkholes using multi-level convolutional neural networks and data association. *IEEE Access*, *8*, 132625-132641. doi: 10.1109/access.2020.3010885

Wang, J., Sun, K., Cheng, T., Jiang, B., Deng, C., Zhao, Y., ... others (2020). Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*.

Weary, D. (2015). The cost of karst subsidence and sinkhole collapse in the united states compared with other natural hazards. In *Sinkholes and the engineering and environmental impacts of karst: Proceedings of the fourteenth multidisciplinary conference.* University of South Florida Tampa Library. doi: 10.5038/9780991000951.1062

Wu, Q., Deng, C., & Chen, Z. (2016). Automated delineation of karst sinkholes from LiDAR-derived digital elevation models. *Geomorphology*, *266*, 1–10. doi: 10.1016/j.geomorph.2016.05.006

Zhu, J., Nolte, A. M., Jacobs, N., & Ye, M. (2020). Using machine learning to identify karst sinkholes from LiDAR-derived topographic depressions in the Bluegrass Region of Kentucky. *Journal of Hydrology*, *588*, 125049. doi: https://doi.org/10.1016/j.jhydrol.2020.125049

Zhu, J., & Pierskalla, W. P. (2016). Applying a weighted random forests method to extract karst sinkholes from LiDAR data. *Journal of Hydrology*, *533*, 343–352. doi: 10.1016/j.jhydrol.2015.12.012

Zhu, J., Taylor, T., Currens, J., & Crawford, M. (2014). Improved karst sinkhole mapping in Kentucky using LiDAR techniques: a pilot study in Floyds Fork watershed. *Journal of Cave and Karst Studies*, *76*(3), 207–216. doi: 10.4311/2013es0135